Improving Performance of Synchronous Transmission-Based Protocols Using Capture Effect over Multichannels

MOBASHIR MOHAMMAD, MANJUNATH DODDAVENKATAPPA, and MUN CHOON CHAN, National University of Singapore

10

Synchronous transmission has been exploited recently for accelerating fundamental operations of wireless sensor networks like data dissemination, data collection, and network-wide agreement by an order of magnitude. Although these protocols (e.g., Glossy) have shown to be highly reliable for small packet sizes, the use of large packet sizes coupled with an excessive number of simultaneous transmissions can reduce reliability significantly. Our work is motivated by the observation that capture effect can improve the reliability of synchronous transmission. We experimentally study the effect of physical layer capture and identify guidelines in which it can be exploited to enhance reliability. Based on these observations, we propose Syncast, a data dissemination protocol that improves over Glossy by exploiting capture effect over multichannels. The evaluation shows that Syncast provides a robust, reliable, and scalable data dissemination of large packets with lower end-to-end delay and up to 92% reduced radio-on-time.

CCS Concepts: • **Networks** → **Sensor networks;**

Additional Key Words and Phrases: Wireless sensor networks, synchronous transmission, multichannel, radio interference, time synchronization, capture effect

ACM Reference Format:
Mobashir Mohammad, Manjunath Doddavenkatappa, and Mun Choon Chan. 2017. Improving performance of synchronous transmission-based protocols using capture effect over multichannels. ACM Trans. Sen. Netw. 13, 2, Article 10 (April 2017), 26 pages.
DOI: http://dx.doi.org/10.1145/3043790

1. INTRODUCTION

Synchronous transmission has gained popularity in recent years because of its capability to perform control data dissemination and network synchronization as shown in Glossy [Ferrari et al. 2011]. In synchronous transmission, multiple transmitters simultaneously transmit the same packet to multiple receivers in a synchronized fashion without the need to perform any carrier sensing. By eliminating most of the overhead involved in channel access, a single message can be disseminated to the entire network in time that is proportional to the diameter of the network. For example, in a testbed with 94 hops, a single message of 8 bytes can be disseminated to all of the nodes with an average latency of 2.4ms.

Based on the idea of synchronous transmission, several protocols have been proposed for data dissemination [Doddavenkatappa et al. 2013; Du et al. 2015], data collection [Ferrari et al. 2012], network-wide agreement [Landsiedel et al. 2013], and bulk data

This work was supported in part by the Agency for Science, Technology and Research (A*STAR), Singapore, under SERC grant 1224104049.
Authors' addresses: M. Mohammad, M. Doddavenkatappa, and M. C. Chan, School of Computing, National University of Singapore; emails: {mobashir, doddaven, chanmc}@comp.nus.edu.sg.

DOI: http://dx.doi.org/10.1145/3043790

transfer [Doddavenkatappa and Choon 2014]. However, it has been observed that the reliability of synchronous transmission degrades significantly when the synchronization error exceeds a certain threshold and that the degradation gets magnified when packet sizes increase [Ferrari et al. 2011; Dutta et al. 2008]. The performance also degrades with an increase in the number of transmitting nodes, as it becomes harder to maintain precise timing among multiple synchronous transmitters, resulting in the so-called scalability problem [Wang et al. 2013; Ji et al. 2013].

In this article, we propose Syncast, with an objective to make synchronous transmission reliable even when the packet size is large and/or when the network is dense, which results in the scalability problem. Our approach is motivated by the observations made in Chaos, where physical layer capture[1] helps in successful packet reception even when multiple nodes synchronously send different packets. The use of capture effect to address the scalability problem provides two benefits. First, to exploit capture effect, the synchronization requirement can be relaxed by up to 320 times (160μs for capture effect in concurrent transmissions vs. 0.5μs for constructive interference in synchronous transmissions). Second, the number of synchronous transmitters can be reduced significantly, thus lowering the cost of transmissions as well as the risk of encountering synchronization errors due to fewer transmitters.

In this work, we make the following contributions:

—We experimentally study the effect of physical layer capture on synchronous transmission. In particular, our measurements show that the existence of capture effect can be predicted with high probability by using the infrequently measured RSSI values. Based on this estimation procedure, a significantly smaller set of synchronous transmitters can be selected so that capture effect can be successfully exploited.
—Our measurement results show that links with the highest RSSI values are fairly stable over time (from hours to days) and that if a link of high packet reception ratio (PRR) has a high RSSI value on one channel, it is very likely to have high PRR on other channels as well. Hence, if capture effect can be observed on one channel on a link, the link is likely to be good on the other channels.
—The preceding two observations made it possible to design Syncast, a synchronous transmission–based protocol that uses capture effect over multichannels to scale data dissemination to a large network with many synchronous transmitters and to support even the largest-size packets. In particular, Syncast exhibits reliability at least as high as Glossy, with significantly lower energy consumption due to shorter radio-on duration and shorter end-to-end delay. Furthermore, since Syncast relies on successful capture effect instead of constructive interference for packet reception, we could lower the synchronization requirement up to 320 times. In addition, as Syncast considers only the strongest links for data dissemination, which tends to be more stable, it features resilience to channel variations compared to Glossy or Glossy-based single-channel protocols.

The article is organized as follows. In Section 2, we present related work. In Section 3, we provide an overview of synchronous transmission and capture effect. Measurement results of capture effect are presented in Section 4, and the design of Syncast is presented in Section 5. Evaluation results are provided in Section 6, followed by our conclusion in Section 7.

2. RELATED WORK

The redesigning of common wireless sensor network protocols started when Ferrari et al. [2011] showed the possibility of achieving synchronous transmissions on an

[1]The terms *capture effect* and *physical layer capture* will be used interchangeably.

IEEE 802.15.4 radio module through their work, Glossy. They achieved tight time synchronization and showed that if the nodes of the network have the same small-size control packet to transmit and are time synchronized to an order of 0.5μs, then instead of contending with each other for channel acquisition, the nodes are allowed to transmit concurrently. The packets then interfere constructively and a reception reliability of up to 99.99% can be obtained [Ferrari et al. 2011]. Using it for large data packets, however, still remained a challenge.

Many-to-one. Motivated by Glossy, numerous protocols have been revamped, achieving a drastic improvement in protocol performance, especially in terms of latency and energy efficiency. LWB [Ferrari et al. 2012] is one such synchronous transmission–based data collection protocol capable of handling low, high, and fluctuating traffic. It showed better reliability and energy efficiency in comparison to the standard collection tree or DODAG-based protocols [Gnawali et al. 2009; Mohammad et al. 2016] over A-MAC [Dutta et al. 2010], LPL [Polastre et al. 2004], or CSMA/CA for small packets having an application payload of 15 bytes. Since LWB is based on synchronous flooding, it promised flexibility to be extended to various traffic patterns like one-to-many, many-to-one, and many-to-many.

One-to-many. Quite often, a large data object needs to be disseminated over a wireless sensor network. The key requirements for such dissemination protocols are low latency and high reliability, both of which are severely hampered if contention-based MAC protocols like CSMA/CA or TDMA-based protocols are employed. Before the advent of synchronous transmissions, Deluge [Hui and Culler 2004] had been the state of the art. However, Splash [Doddavenkatappa et al. 2013] outperformed it by up to 20 times in terms of speed. It achieved such improvement by exploiting synchronous transmissions along with tree pipelining, channel cycling, opportunistic overhearing, and XOR coding. However, despite the many techniques incorporated, a final round of CSMA/CA-based local recovery, which takes more than 50% of the total dissemination time, was still needed. Pando [Du et al. 2015] incorporated fountain codes to Splash-like pipelined transmission to eliminate this long tail problem and was able to achieve an even faster data dissemination. However, protocols like Splash and Pando are suited for the dissemination of a large file that can be decomposed into multiple small packets to be able to fill a packet pipeline. Reliability of a single packet transmission will suffer from the scalability problem nevertheless due to a lack of being able to exploit techniques like network coding and/or pipelined data transmission.

One-to-one. Synchronous transmission is exploited in the context of one-to-one communication primitives as well. P^3 [Doddavenkatappa and Choon 2014] improved PIP [Raman et al. 2010], the state-of-the-art protocol for high throughput bulk data collection by five times on average, achieving an end-to-end average goodput of 177.8Kbps. However, as the maximum observed utilization is capped at 94.1% and much lower in many cases, it hints at some limitations on how synchronous transmission techniques have been applied in the preceding protocols.

Many-to-many. Recently, Landsiedel et al. [2013] showed in Chaos that synchronous transmissions not only are limited to the same data transmission but also can work in cases where different nodes transmit different data. In the latter cases, capture effect plays a significant role in ensuring successful packet delivery in such a collision-prone environment. Flash flooding [Lu and Whitehouse 2009] and Chorus [Zhang and Shin 2010] are other popular flooding protocols that have exploited capture effect in the past. Since these protocols exploit capture effect by chance, they suffer significant packet collisions amidst occasional successful reception. Our approach, on the contrary, makes an attempt to maximize the occurrence of physical layer capture through careful multichannel transmissions.

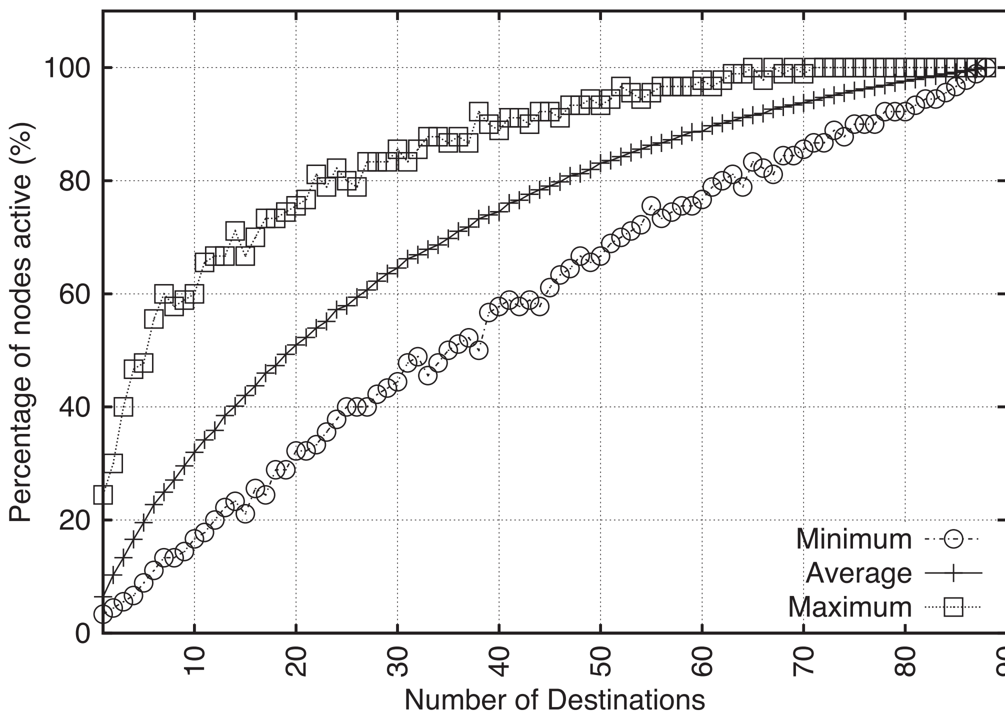


Fig. 1. Percentage of active nodes required to transmit when disseminating to a specific set of network nodes.

Despite the order of magnitude improvements that can be achieved by reimplementing existing protocols using synchronous transmission, some issues still remain. The first is the well-known scalability problem, which was first identified by Wang et al. [2013]. They argued that the reliability of packet reception decreases when the number of synchronous transmitters increases. The same was reported by other subsequent works [Yuan et al. 2014; Carlson et al. 2013; Doddavenkatappa et al. 2013; Ji et al. 2013; Du et al. 2015], which provided various heuristic-based solutions to mitigate the scalability problem. Splash [Doddavenkatappa et al. 2013], for instance, tried to reduce the number of synchronous transmitters by forcing only the nonleaf nodes of the dissemination tree to transmit in the first round of dissemination. Pando [Du et al. 2015] incorporated a similar technique to reduce the number of synchronous transmitters by allowing leaf nodes to relay the received packets only when the sequence number of that packet is even. Since more than 50% of the nodes in the underlying data collection tree are leaf nodes [Doddavenkatappa et al. 2013], the number of synchronous transmitters is significantly reduced.

Sparkle [Yuan et al. 2014] employs another interesting approach to solving the scalability problem, but for a specific case of one-to-one communication. It does so by identifying the nodes that are in a direct path from source to destination using the WSNShape technique, which relies on capture effect and later turning off all nodes that are not in the path identified previously. To test Sparkle for a more general case of network-wide data dissemination, we implemented the WSNShape algorithm for Contiki OS [Dunkels et al. 2004] and ran it with a path count size of 1 on channel 26 of an Indriya testbed and illustrated in Figure 1, the percentage of active nodes that are required to transmit synchronously when we vary the number of destination nodes k

from 1 to the entire network. For each selection of k random destination nodes, we run it 1,000 times and plot the minimum, maximum, and average values. It can be seen that although a large number of nodes can be turned off to disseminate data to a small section of the network, for network-wide dissemination the WSNShape technique does not solve the scalability issue, as all nodes are required to remain active and participate in packet transmission.

Such algorithms or heuristics may mitigate the problem for a very specific setting and would not eliminate the problem altogether. From our experimental results, the scalability problem is more severe than thought since sometimes only two synchronous transmitters can potentially reduce the reception reliability to as low as 10%. The second and equally important issue with synchronous transmissions is the reduction in reliability with the increase in packet size [Ferrari et al. 2011].

Since synchronous transmission has shown tremendous potential, our work focuses on understanding the limitations in its current usage and making it as reliable as possible to enable its large-scale adoption. A key challenge of our work is to identify the nodes that can potentially degrade reception reliability and show how high reliability can be maintained, even for the largest packet size.

3. BACKGROUND

In this section, we give some background on how synchronous transmission works in low-power wireless networks and the requirements for physical layer capture to ensure high reception reliability in a collision-prone environment.

3.1. Synchronous Transmission

To successfully decode a packet sent by synchronous transmitters, the nodes must send the packets with a temporal offset of not more than $0.5\mu s$ so that the signals can interfere constructively at the receiver. Glossy has been able to achieve this level of time synchronization following the basic principle that the receivers should relay the received packet immediately on reception. Assuming the propagation delay to be negligible, the neighbors of a node should receive the packet almost at the same time with a negligible time difference and should, in turn, relay the packet almost instantaneously leading to a less than $0.5\mu s$ time difference, thus causing them to constructively interfere at the receiver.

Such synchronous transmissions for the IEEE 802.15.4 standard is possible because on reception of the same packet, the SFD pin (a pin that goes high on detecting the reception of the start field delimiter of the incoming packet) of cc2420 (a 2.4GHz IEEE 802.15.4 RF transceiver) rise and fall almost at the same time. The nondeterminism arises only due to the difference in the time when an interrupt is generated and when it is served because on MSP430 (a low-power microcontroller), an interrupt is served only after the completion of the current instruction, which requires one to six clock cycles to complete. Thus, the issue of transmission request can be made synchronous by inserting a certain number of no-operation instructions (NOPs) at the beginning of the interrupt handler, which depends on which cycle the currently executing instruction is in. Although the timing requirement for physical layer capture is much less stringent, slight delays in the serving of interrupts can be detrimental to time-sensitive constructive interference.

3.2. Capture Effect

In simple terms, *capture effect* is a phenomenon in which a receiver is able to decode a packet correctly from a sender in the presence of interfering signals if the signal strength from the sender is sufficiently stronger than that from the interfering sources.

However, in a scenario where there are multiple transmitters and receivers, having one sender whose signal strength is distinctly stronger than the rest becomes less likely.

For a successful physical layer capture to occur, there are two key requirements. The first requirement, as mentioned previously, is that the strongest signal strength P_s has to be significantly stronger than P_n, the sum of the n weaker interfering signals strengths. In fact, it has been noted that for capture effect to occur, the received signal from the strongest node should be about 3dB stronger than the sum of the received signals from all other nodes [Arnbak and Van Blitterswijk 1987; Dutta et al. 2010; Son et al. 2006].

The second is a strict timing requirement where the stronger signal should arrive at the receiver along with the other n weaker interfering signals within 160μs of the time difference called *capture window* t_w, which in fact is the propagation time for the IEEE 802.15.4 packet's preamble and SFD [Ji et al. 2013; Landsiedel et al. 2013]. Thus, a packet reception during concurrent transmissions[2] in an environment that is free from external interference would be unsuccessful if

$$10 * \log_{10}(P_s/P_n) < 3dB \text{ during } t_w \text{ with } n > 0. \quad (1)$$

In a scenario where two nodes are transmitting concurrently, failure of reception happens when a node with weak signal strength at the receiver starts to transmit, followed by the node with strong signal strength after the 160μs capture window. In such a case, the strong signal acts as a strong interferer, leading to packet corruption. However, if the nodes are time synchronized to the order of a few tens of microseconds and transmit concurrently, then irrespective of which node starts to transmit first, the receiver will almost always be able to decode the packet correctly if the received signal strength from one node dominates the other.

4. MEASUREMENT RESULTS

Unless otherwise stated, all measurements are performed on channel 26 of IEEE 802.15.4 to minimize the effect of external interference from commonly used Wi-Fi channels, as they do not overlap with each other [Liang et al. 2010]. In all of our experiments, we employ TelosB [Polastre et al. 2005] devices with CC2420 transceivers.

4.1. Reliability of Synchronous Transmissions

The reliability of synchronous transmission is influenced by two factors: packet size and the number of transmitters. To measure the impact of these two factors, we conducted an experiment on the Indriya testbed [Doddavenkatappa et al. 2011a] for multiple packet sizes and for varying numbers of synchronous transmitters. Figure 2 shows the reception reliability, averaged over 300,000 packet transmissions involving 150 scenarios. Each scenario is described by a different number of synchronous transmitters (two to seven synchronous transmitters), different packet size (8 bytes to 128 bytes), and a random node selection. For each set of selected nodes (one randomly selected receiver and two to seven of its one-hop neighbors randomly selected to be synchronous transmitters), 2,000 packets are transmitted. The combined effect of a large number of synchronous transmitters and large packet size leads to a dramatic reduction in packet reception rate. Note that the degradation is much more pronounced for large packets, whereas it is not so serious for smaller packets. This observation does not contradict the measurement in the Glossy paper, as the result shown in Ferrari et al. [2011] is for small packets (8 bytes) with capture effect included.

[2]We use concurrent transmission to refer to transmissions with a 160μs timing requirement, whereas synchronous transmission is used for transmissions requiring a 0.5μs time synchronization.

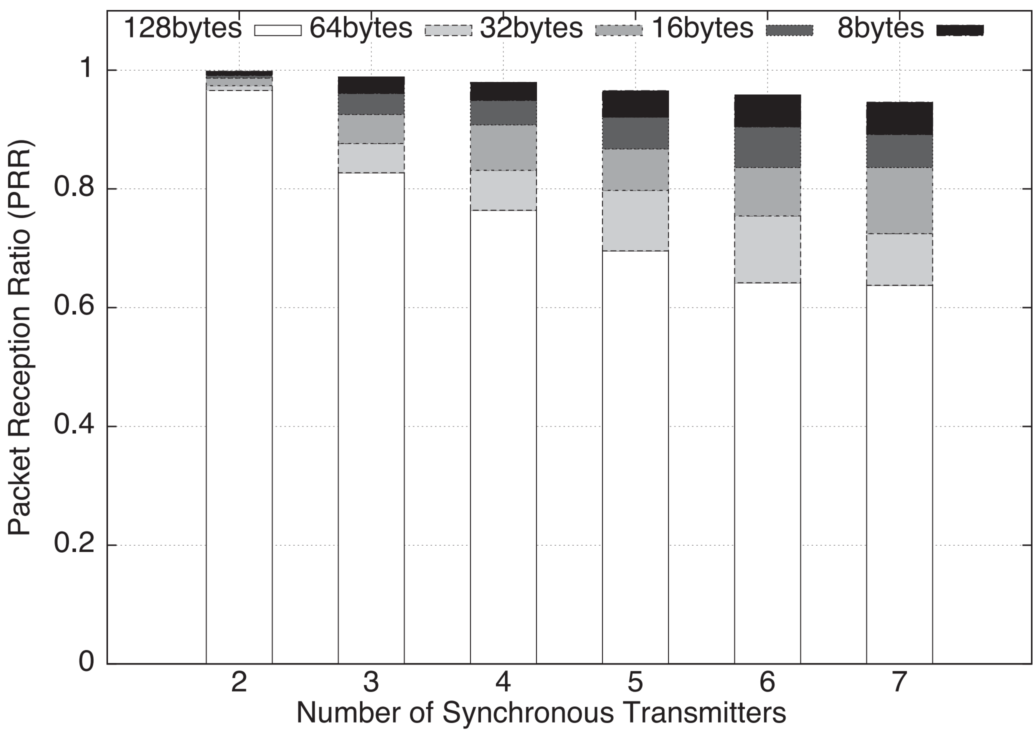


Fig. 2. Impact of packet size on reception reliability with increasing synchronous transmitters.

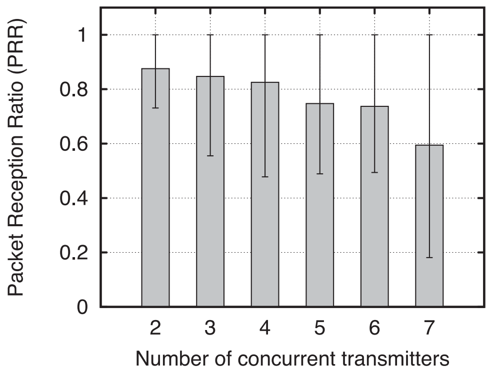
To further validate our observation, measurements on three additional environments are performed:

—A 25-node deployment in an indoor environment (laboratory),
—A 25-node deployment in an outdoor environment (soccer field)
—The Twist testbed [Handziski et al. 2006].

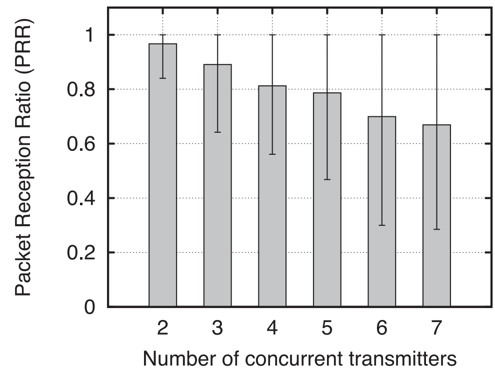
To check if reception reliability decreases with the number of synchronous transmitters, we randomly selected one of the nodes as the initiator and triggered it to broadcast 1,000 maximum-size packets. A few randomly selected one-hop neighbors of the initiator on successful reception relayed the same packet synchronously back to the initiator. We repeated the experiment by varying the number of synchronous transmitters and the set of senders and the receiver.

Figure 3 plots the reception reliability averaged over 10 different combinations of synchronous transmitters. For each combination, 1,000 packets are transmitted. The maximum and minimum PRRs are also shown. The results provide further support for the observation that reception reliability decreases as the number of synchronous transmitters increases. In fact, on the Twist testbed, PRR as low as 0.142 can be observed with seven synchronous transmitters.

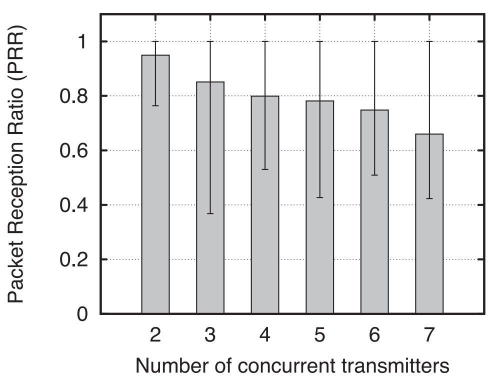
This decrease in reliability with an increasing number of synchronous transmitters can be explained by the increasing likelihood of larger temporal displacements among synchronous transmitters [Dutta et al. 2008], which can be caused by a combination of software, hardware, and signal propagation delays [Wu et al. 2014]. As illustrated in Ferrari et al. [2011], the impact of timing jitters as small as $0.25\mu s$ is enough to cause a drastic drop in reception reliability.



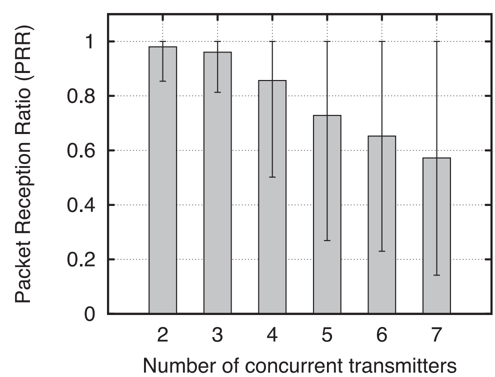
(a) Indoor Setup



(b) Outdoor Setup



(c) Indriya



(d) Twist

Fig. 3. PRR versus the number of concurrent transmitters for different setups, including an indoor environment, an outdoor environment, and two large-scale testbeds.

4.2. Quantifying Capture Effect Through RSSI

Our approach is based on exploiting capture effect, which imposes a much less stringent timing requirement ($<160\mu$s) than constructive interference ($<0.5\mu$s). However, to have an effective capture effect, the strongest signal needs to be much stronger than the sum of all other weaker transmissions. Hence, the number and choice of the allowable concurrent transmitters are needed to be managed to ensure that Equation (1) is satisfied. Concurrent transmission is used here to distinguish from synchronous transmission where constructive interference is necessary for packet reception.

We first conducted measurements to demonstrate the presence and effect of physical layer capture on concurrent transmission. We define Δ as the difference between the largest and the second largest RSSI values observed at the receiver when multiple nodes transmit concurrently. This Δ value provides an indicator for the presence of capture effect, as a large value would indicate a high likelihood that Equation (1) can be satisfied.

We performed experiments on the four testbeds mentioned in the previous section (indoor, outdoor, Indriya, and Twist). A random node was selected as initiator and a subset of one-hop neighbors were randomly selected as concurrent transmitters, sending packets back to the initiator. We varied the number of concurrent transmitters from two to seven and recorded the packet reception reliability at the initiator.

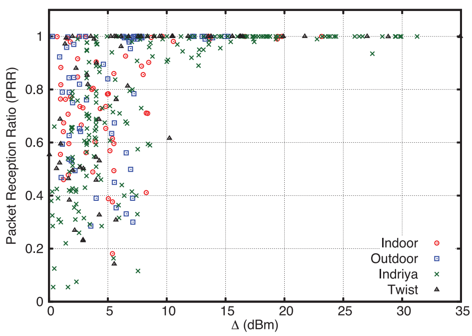


Fig. 4. Dependency of reception reliability on Δ when nodes send the same data synchronously.

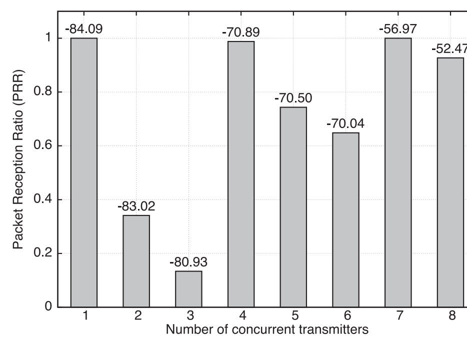


Fig. 5. Increase in reception reliability on adding nodes having capturing ability.

Figure 4 plots the outcome of the preceding experiment, where each point corresponds to the packet reception reliability averaged over 1,000 transmitted packets for a given network and a given number of concurrent transmitters. A total of 400 different sets of sender/transmitter is shown in the figure. The result shows that the packet reception reliability stays consistently above 90% for a Δ exceeding 10dBm (for up to seven concurrent transmitters). For Δ values below this threshold, where capture effect is less likely to occur, reliability varies significantly.

To validate the ability to control capture effect using the RSSI information, we selected a particular scenario having multiple transmitters with known RSSI values and added transmitters to the list of concurrent transmitters one at a time in the order of increasing RSSI values. We look for a sudden surge in reliability and compare the observation with the prediction of successful capture effect according to Equation (1). Figure 5 illustrates one such experiment performed on Indriya using maximum-size packets. We begin with just one node transmitting a large packet (with average RSSI –84.09dBm; the reception reliability is good at close to 99%. On adding nodes 2 and 3 with small Δ values of 1.7 and 2.09, respectively, the reception reliability drops to 37% and 13.37%, respectively. However, when we add one more node, providing a large Δ value of 10.04, the reception reliability shoots from 13.37% to 98.78%, thus demonstrating the ability of capture effect alone in guaranteeing high reception reliability. Next, nodes 5 and 6 are added. As the Δ values are small again, reliability decreases. On adding node 7, with a large Δ of 14, reliability again increases significantly to 100%, again due to successful physical layer capture.

4.2.1. Selection of Δ. The next question we asked is what Δ value is suitable to be able to exploit capture effect in a practical setting. Note that if Δ is set too large, the opportunity to exploit capture effect becomes small, as it is difficult to find configurations with such large RSSI separation. On the other hand, if Δ is too small, the number of allowable concurrent transmitters may become too low. To find an appropriate threshold, we consider two large-scale testbeds, Indriya and Twist, and measure the distribution of RSSI values of the links in the network. Figure 6 shows the fraction of links on the two networks belonging to various RSSI ranges.

Using the distribution of the link RSSI collected from the preceding two testbeds, we ran a simulation where we randomly selected a transmitter T and a receiver R from the testbed. To this configuration, we add another set of N synchronous transmitters whose signal strengths were selected from the above distributions while ensuring that $\Delta \geq 5$dBm, $\Delta \geq 10$dBm, and $\Delta \geq 15$dBm. For each Δ, we check whether the physical layer capture was successful or not based on Equation (1). As an additional baseline,

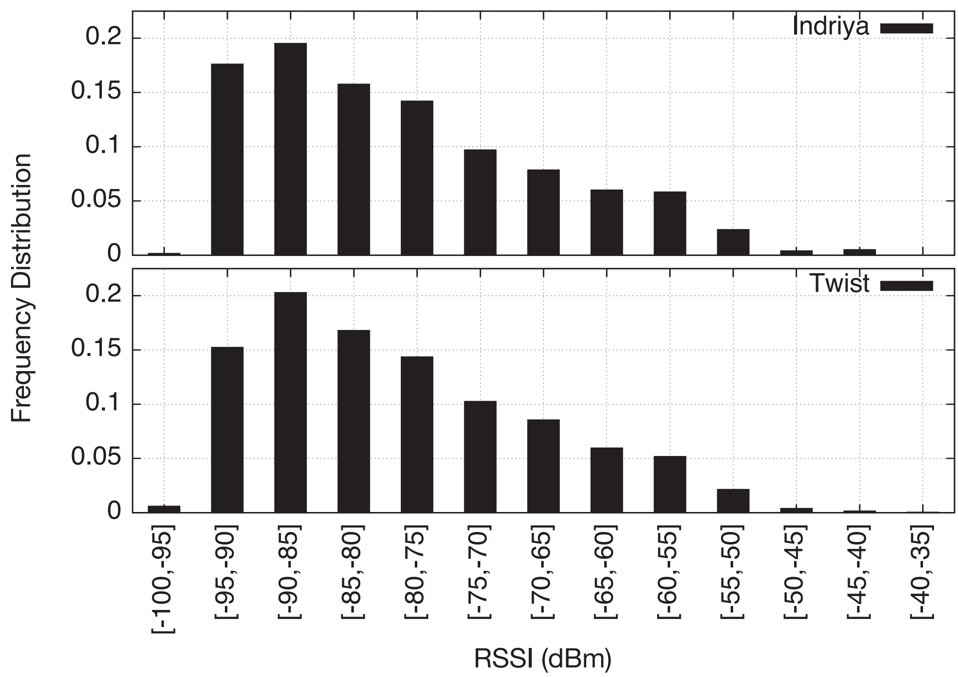


Fig. 6. Distribution of RSSI values of the links in Indriya and Twist.

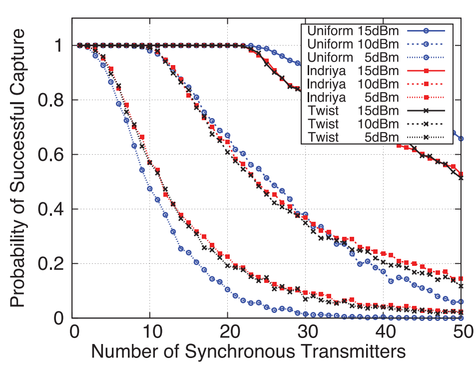


Fig. 7. Probability of successful capture for a given Δ on a given network.

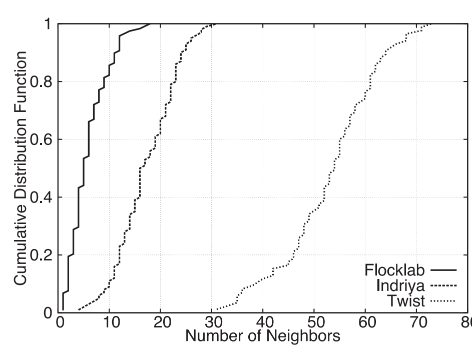


Fig. 8. CDF of the number of neighbors for each node on different testbeds.

we include the case where RSSI values are uniformly distributed over the given range. Figure 7 shows the probability of successful capture when different Δ values are used with an increasing number of concurrent transmitters. With a small value of 5dBm, once the number of concurrent transmitters increases beyond 3, the likelihood of successful capture drops quickly. However, with a Δ of 10dBm and 15dBm, the average maximum number of concurrent transmitters that can be supported increases to 10 and 22, respectively. Depending on expected network density, different values should be selected.

For instance, to identify the appropriate Δ value, we perform neighbor discovery on channel 26 at the maximum power level for three popular wireless sensor network testbeds (Flocklab [Lim et al. 2013], Indriya, and Twist). Figure 8 plots the

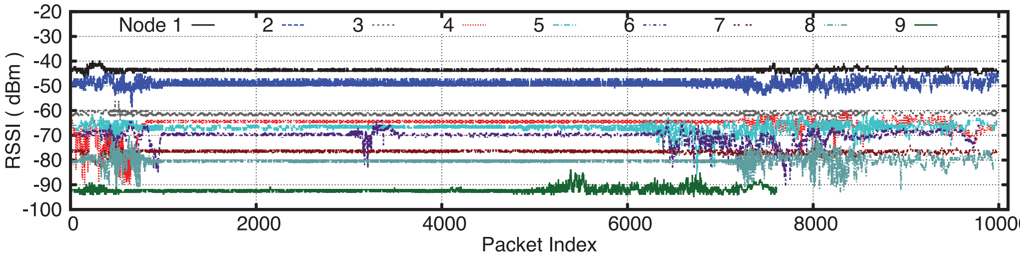


Fig. 9. RSSI variations over different links for a packet size of 128 bytes.

neighborhood distribution, showing Flocklab to be the sparsest (with 5 neighbors on average) and Twist to be the densest (with more than 50 neighbors on average). For Indriya (with up to 30 neighbors), assuming that transmissions are spread uniformly over four channels, we expect up to 8 neighbors to transmit concurrently on the same channel to a common receiver, For such a deployment, a Δ of 10 suffices. On the other hand, for Flocklab, a lower Δ value would be sufficient, whereas for Twist, transmitting at a lower power to reduce neighborhood count would be necessary, as we will see in Section 5.3.3.

4.3. Stability of RSSI Values

For this set of measurements, we follow the terminology of Srinivasan et al. [2008], where a link is referred to as "good," "intermediate," and "poor" if the PRRs are above 0.9, in the range from 0.1 to 0.9, and below 0.1, respectively.

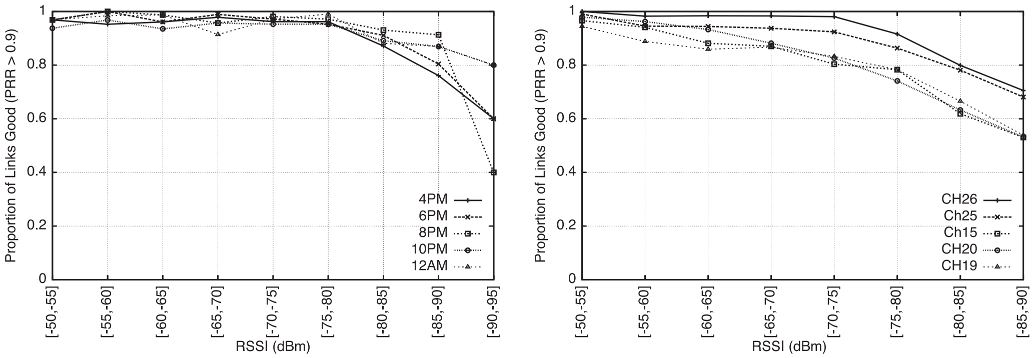
Although the results in the previous section demonstrate the possibility of using RSSI values to exploit capture effect, there is the question of whether such RSSI values can be obtained efficiently with very low overhead. Another key question is on the stability of RSSI values over time. To gain further insight into these issues, we perform different experiments to record RSSI values over different links, time periods, and channels.

4.3.1. Stability over Links. In Figure 9, a randomly selected node on Indriya broadcasts 10,000 packets at a rate of 1 packet per second to its neighbors on channel 26 over a period of more than 2.5 hours. We record RSSI values of these packets at the receivers. We can clearly observe that the fluctuation of RSSI values for the stronger links—the links of interest for the purpose of exploiting capture effect—tend to be very stable. The weakest link did not even receive the packets toward the end during strong interference.

4.3.2. Stability over Time. Next, we measure how the PRR of a strong link varies over time. First, we measure the RSSI and PRR of all links (averaged over 100 packets) from all nodes to all nodes at 2 pm. We then consider only links with PRR = 1.00 and group these links into bins of 5dBm width on their RSSI values ranging from –95dBm to –50dBm. For the next 10 hours (until midnight), we measured the average PRR of all considered links in each bin every 2 hours. Figure 10(a) shows the result.

It is clear that a significantly large proportion of links with PRR = 1.00 and RSSI ≥ -80dBm remains "good" even 10 hours later. Only the weak links (RSSI ≤ -80dBm) have their PRRs see a significant drop.

4.3.3. Stability over Different Channels. It has been observed that the PRRs on different channels are not correlated [Doddavenkatappa et al. 2011b]. This observation is detrimental to the protocols that perform link measurement only on a single channel and yet need to operate on multiple channels [Doddavenkatappa and Choon 2014; Doddavenkatappa et al. 2013; Raman et al. 2010]. For optimal performance, these protocols ideally should perform link measurement on multiple channels, but such measurements incur significant overhead.



(a) Variation over the day

(b) PRR Variation on different channels

Fig. 10. Variation in the quality of the link over the day or across different channels.

Fortunately, it may not be necessary to obtain measurements for all of the channels. What is needed, at least for our purpose, is to know if a link has high RSSI and PRR on one channel. Then there is a high probability that the link is also "good" on another channel.

To check if the said relation is true, we performed link measurements on Indriya on channels 26, 25, 20, 19, and 15 back to back. As each set of measurements over a single channel takes 100 seconds, there is a time difference of roughly 8.33 minutes from the first measurement on channel 26 to the last measurement on channel 15.

We selected links with PRR = 1.00 on one channel with a minimum RSSI value ranging from –90dBm to –50dBm and plotted the proportion of those links that remain "good" on the other channels in Figure 10(b). It can be inferred from the result that if we consider the RSSI threshold of strong links to be more than –75dBm, it is highly probable ($\geq$80%) that the same links on other channels are "good" as well. If we consider lesser RSSI values, the link quality on the other channel becomes more unpredictable. Based on this observation, a communication tree can be constructed where we select only links with PRR = 1.00 and RSSI ≥ -75dBm and claim with high confidence that the transmissions on these, when performed on other channels, remain reliable.

All observations made in this section are incorporated into our algorithm to be presented in the next section.

5. DESIGN

From the previous section, we see that capture effect can be used in conjunction with multichannel allocation to achieve reliable reception in a collision-prone environment of synchronous transmissions. The basic idea is to ensure that every receiver is captured by exactly one transmitter on a good (PRR = 1.00) and strong (RSSI ≥ -75dBm) link while maintaining $\Delta \geq \Delta_{th}$ constraint so that the other signals that try to interfere are just too weak.

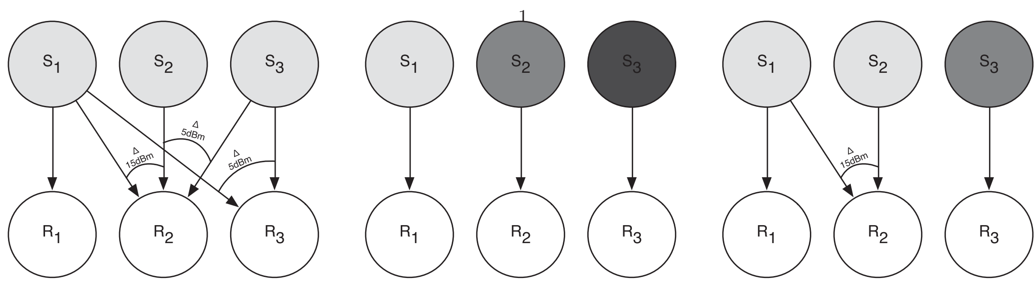
We start by presenting a naive approach of using a separate channel for each link and then introduce the idea of channel reuse that is incorporated in Syncast, the proposed solution.

5.1. Naive Approach

When the objective is to rely on a good and strong link for packet receptions, it is natural to consider dissemination on a tree topology. To mitigate the scalability and the interference problem from affecting the reception reliability, the simplest approach

Nodes	S_1	S_2	S_3
R_1	-45	0	0
R_2	-63	-48	-53
R_3	-52	0	-47

RSSI values in dBm of the packets received in the given setting. Zero value signifies no link between the corresponding sending and receiving nodes



(a) Same Channel Transmission (b) Naive Approach (c) Channel Reuse

Fig. 11. Channel allocation strategies.

would be for all nodes in the same hop to transmit on orthogonal channels so that there is only one transmitter per channel.

Consider a simple setup of three senders and three receivers all transmitting synchronously on the same channel as shown in Figure 11(a). There is an arrow between two nodes if there is a communication link between them. Let us assume that PRR = 1.00 for all of the links. Since R_1 can hear from only S_1, the reception reliability at this node is expected to be very high. However, this transmission will result in degradation of reliability for R_2 and R_3, who will be hearing from three (S_1, S_2, S_3) and two (S_1, S_3) senders, respectively. In the naive approach, each of the three senders can be allocated a different channel on which to transmit, as shown in Figure 11(b). However, such allocation is only possible when there is a large number of available channels. Since the number of channels is often much smaller than the number of possible synchronous transmitters at the same hop, especially in typical dense sensor network deployments, a different channel allocation scheme involving channel reuse is required.

5.2. Channel Reuse

For a dense sensor network like Indriya, a node can have as many as 20 neighbors. IEEE 802.15.4 is the commonly used networking technology in low-power sensor networks. There are only 16 channels available, out of which most overlap with the three commonly used Wi-Fi channels (channels 1, 6, and 11) causing a lot of external interference to most of the IEEE 802.15.4 channels [Liang et al. 2010]. To be precise, there are only four IEEE 802.15.4 channels (channels 15, 20, 25, 26) that do not overlap with Wi-Fi and a few channels that partially overlap, as shown in Figure 12.

Naive channel allocation is clearly not feasible due to the limited number of channels available. Taking a cue from cellular networks, one possibility is to allow channel reuse among nodes. The challenge is to determine when a channel can be reused. The Δ factor defined previously provides the means to make this decision.

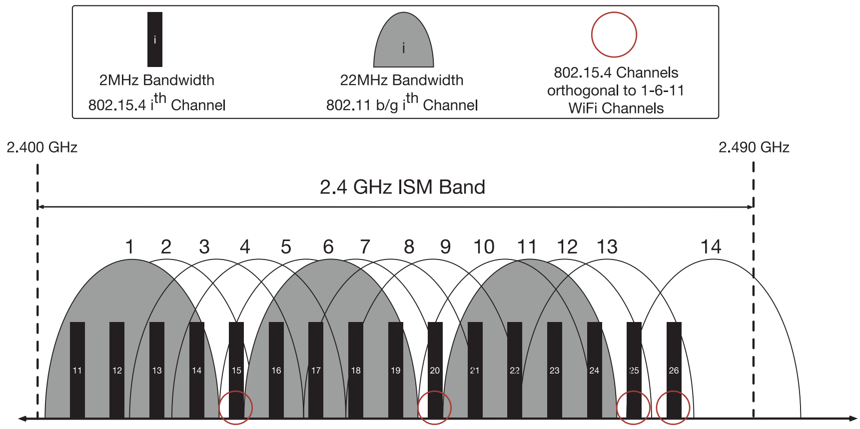


Fig. 12. Collocated 802.11 b/g and 802.15.4 channels.

Consider the same setup of three senders and three receivers shown in Figure 11(a). As observed previously, when there are multiple synchronous transmitters, if there is a gap of Δ exceeding a certain threshold in RSSI between the strongest and the next strongest link, high reception reliability can be maintained with large probability. Let the required threshold on Δ be Δ_{th}. The channel reuse policy works as follows:

—If there are two synchronous transmitters with a difference in RSSI values at the receiver more than Δ_{th}, then both transmitters can transmit on the same channel and capture effect will cause transmission from the stronger link to succeed with higher probability.
—However, if the difference in RSSI values at the receiver is less than Δ_{th}, then the two transmitters need to transmit on different channels to avoid interference in case the synchronization error exceeds 0.5μs.

In Figure 11(a), since the difference in the RSSI value of the packets received from S_1 and S_2 at R_2 is more than $\Delta_{th} = 10$dBm, they can use the same channel to transmit. However, since the difference in the RSSI value of the packets received from S_2 and S_3 at R_2 is less than Δ_{th}, they need to transmit on different channels. In this case, only two instead of three channels are sufficient, as shown in Figure 11(c). The reduction in the number of channels required is much more significant when a large sensor network is considered and channel reuse is allowed.

5.3. Syncast

Based on the experimental results obtained and the various observations made so far, we can now present the design of Syncast. It is a robust, energy-efficient protocol that reliably disseminates a packet (of size up to the maximum packet size supported by IEEE 802.15.4) to all nodes in the network. It can be thought of as an enhancement layer to Glossy. The key difference between Syncast and Glossy is that dissemination in Syncast is organized along a multichannel tree topology exploiting channel diversity, with transmission allowed only during the assigned time slot to minimize the energy consumption and maximize the success of physical layer capture. Syncast provides a robust, reliable, energy-efficient, and scalable data dissemination with reduced end-to-end delay.

Syncast consists of an initial link selection phase that measures the RSSI and PRR on all of the links. Although this phase incurs extra overhead, based on the results shown in the previous section, this measurement only needs to be performed infrequently. The core part of Syncast is the tree topology construction and channel allocation algorithms. Once the topology and channel assignments are determined, synchronous transmission can proceed. Note that since transmission is performed on the tree, unlike Glossy, each node will only receive and transmit once at a fixed time interval based on its position on the tree.

5.3.1. Topology Construction. Based on the link measurements available, we select only links that have PRR = 1.00 and strong RSSI ($>= -75$dBm). We call this set of links L and the set of all nodes N. The tree topology for a given source node s is constructed as shown in Algorithm 1.

ALGORITHM 1: Topology Construction Algorithm

```
Data: Link set: L, Node set: N, Source node: s
S ← s;
T ← N − S;
count ← 1;
hopcount(s) = 0;
while T ≠ φ do
    foreach t ∈ T do
        if t can connect to a node in S using link in L then
            T = T − t;
            S = S + t;
            hopcount(t) = count;
        end
    end
    count++;
end
```

5.3.2. Link Classification. Given the topology generated previously, we classify all links in the graph into three categories (G, B, and W), which can be organized into a link classification matrix *LCM* as shown in the following:

—Starts with hopcount $i = 1$.
—For each node with hopcount i, find the (incoming) link with the strongest RSSI value RSSI_s and classify this link as "good." Denote this link as G in *LCM*.
—The other links connected to this node with $\text{RSSI}_s - \text{RSSI} < \Delta_{th}$ is classified as "bad." Denote such links as B in *LCM*.
—For the rest of the incoming links of this node with $\text{RSSI}_s - \text{RSSI} \geq \Delta_{th}$, classify them as "don't care," since the signal strength is (most likely) too weak to affect the transmission on the strongest link. Denote such links as W in *LCM*.
—Increment hopcount by 1 and repeat the preceding classifications until all nodes are considered and all links are labeled.

An example of *LCM* is shown in Figure 13. We use $\Delta_{th} = 10$dBm in our algorithm implementation. From the results obtained in the previous section, this threshold can support up to a maximum of 10 synchronous transmitters. The number of neighbors in a dense network can be larger than 10 (e.g., some nodes in Indriya have up to 20 neighbors). Therefore, in a single-channel system, a 10dBm threshold would not be sufficient. However, in a multichannel system such as Syncast, the average number of neighbors sharing the same channel is significantly lower and a 10dBm threshold suffices.

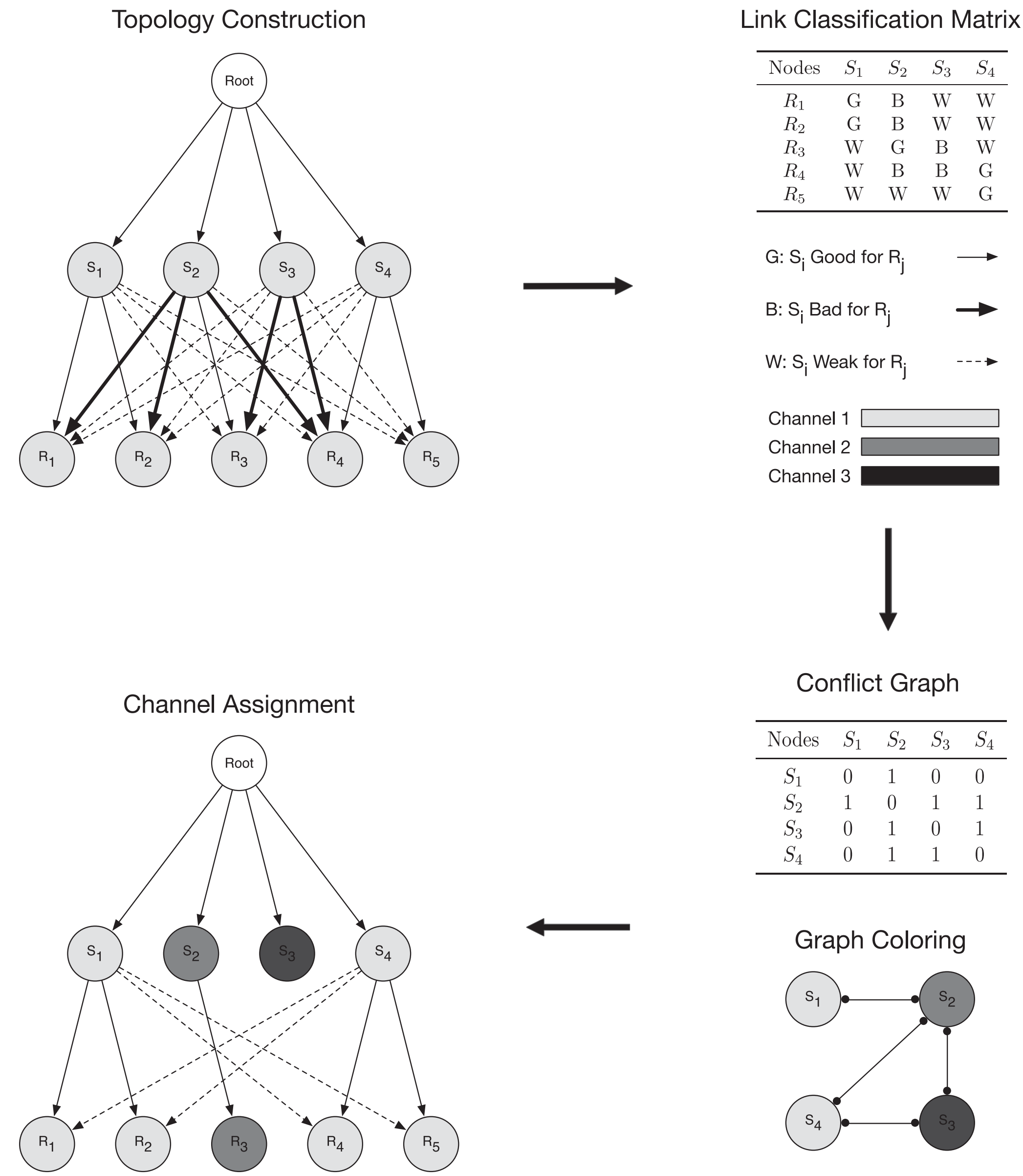


Fig. 13. Flow of Syncast.

5.3.3. Channel Assignment. For channel assignment, we construct a conflict graph CG as shown in Figure 13. Every node in the network has a corresponding node in the CG. There is an edge between any two nodes if one prevents the other from capturing a receiver. The construction of CG is as follows.

Initially, when hopcount is 1, there is only one sender and multiple receivers. Thus, a single channel is sufficient. For subsequent hopcounts, consider the set of senders (S) and receivers (R) for the given hopcount. All nodes in S are inserted in the conflict graph with no edges initially. For each receiver $r \in R$, let $s_G \in S$ be the node where the

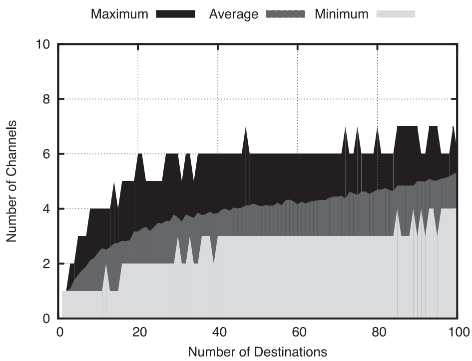


Fig. 14. Minimum, average, and maximum number of channels required for channel assignment averaged over 100 runs for random destinations on the Indriya testbed.

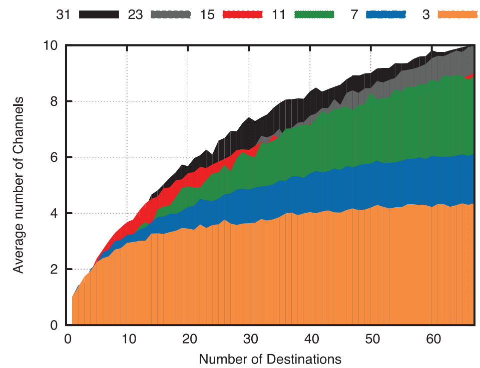


Fig. 15. Number of channels required for Syncast when averaged over 100 runs for random destinations and for different power levels on the Twist testbed.

link from s_G to r is marked G in LCM. For any other sender $s_B \in S$ such that the link from s_B to r is marked B in LCM, add an edge e_{s_G,s_B} to the conflict graph CG for this hopcount.

Once all nodes in r have been considered, the construction of CG is completed for this hopcount and channel assignment can proceed for the rest. Note that when two senders have a common edge in CG, they cannot transmit simultaneously on the same channel, because when one sender tries to capture the receiver, the other acts as a strong interference if there is sufficient temporal displacement between their signals. Hence, the channel allocation problem can be thought of as a graph coloring problem where two nodes joined by an edge should not have the same color (i.e., should not transmit on the same channel).

We do not seek the optimal solution for the graph coloring problem. Instead, we use a greedy heuristic that sorts the nodes in CG in descending order of their degrees, and nodes with higher degrees are colored early while respecting the coloring constraints. The resulting solution provides a channel allocation that allows multiple senders to transmit synchronously on the same channel while maximizing the chances of physical layer capture.

To check the performance of the greedy channel assignment in terms of the number of channels needed, we took the link measurements on Indriya and simulated 100 runs of tree formations and channel assignment for randomly selected destinations. The number of destinations per tree varies from 1 to 100. The minimum, average, and maximum number of channels required for the scheme are shown in Figure 14. It can be seen that up to six channels are required for delivering packets to up to 100 destination nodes. In most cases, four or five channels are sufficient. With the proposed channel assignment scheme, the limited number of usable channels available in IEEE 802.15.4 suffices.

However, for dense deployments with each node having many neighbors, the number of channels required for Syncast operation might exceed the number of noninterfered channels available. The Twist testbed is one such deployment, where each node may have up to 50 neighbors on average at the maximum transmission power level. A simple approach to solving this channel explosion problem would be to find a suitable transmission power level. In Figure 15, we show the number of channels required when we vary the number of destinations per tree from one to the entire Twist network. At a maximum transmission power level of 31 for the CC2420 radio on TelosB, 10 channels

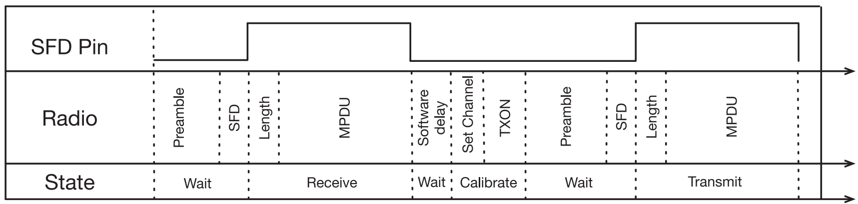


Fig. 16. Incorporation of channel hopping to a tightly synchronized protocol involves execution of switch channel instruction, adding a fixed calibration time for all of the nodes.

are needed. However, the required number of channels reduces as the transmission power is reduced. A suitable power level (3 for Twist) can thus be chosen to limit the number of channels required for Syncast.

5.3.4. Channel Switching Latency. One challenge was to incorporate channel hopping into the tightly synchronized reception/transmission schedule. Since each node may have a different reception and transmission channel, it is important for the nodes to quickly change to the transmission channel immediately upon successful packet reception. Moreover, the channel hop should not add significant latency to the packet relay. Fortunately, channel switching time for the cc2420 radio is on the order of microseconds (time to change the channel number register plus the time to strobe to TX state again and finally 192μs to calibrate). It has also been exploited in the past in Splash [Doddavenkatappa et al. 2013] and Pando [Du et al. 2015]. This adds a fixed, deterministic, and small calibration time for every hop, as shown in Figure 16.

6. EVALUATION

We designed Syncast by modifying Glossy, which is implemented in Contiki OS for the TelosB platform and is a framework for many synchronous flooding-based protocols like Low-Power Wireless Bus [Ferrari et al. 2012].

We consider three performance metrics:

—*Reliability*: The number of data/control packets received successfully out of the total packets sent
—*Latency*: The delay between the start of packet dissemination to its actually reception by a node represented in terms of hopcount, which translates directly to time based on the packet-on-air time since the delays are highly deterministic
—*Energy*: The total duration of the flooding phase for which the radio is turned on for packet reception and transmission.

We evaluate Syncast through extensive experiments against Glossy, the basic synchronous transmission protocol on a real-life testbed, Indriya, and show the performance improvement. Syncast not only makes synchronous communication robust to channel variations but also help in reducing the latency and overall energy consumption by significant amounts.

6.1. Reliability

Reliability is the foremost concern for protocols involving dissemination of data or control instructions. In this section, we investigate the reliability of Syncast with respect to channel variations and packet sizes over multiple days.

6.1.1. Influence of Channel Variations. To check the impact of channel variations on the reception reliability of synchronous transmissions, we disseminated 10 rounds of 500 maximum-size IEEE 802.15.4 packets per day for five consecutive days. To take Wi-Fi

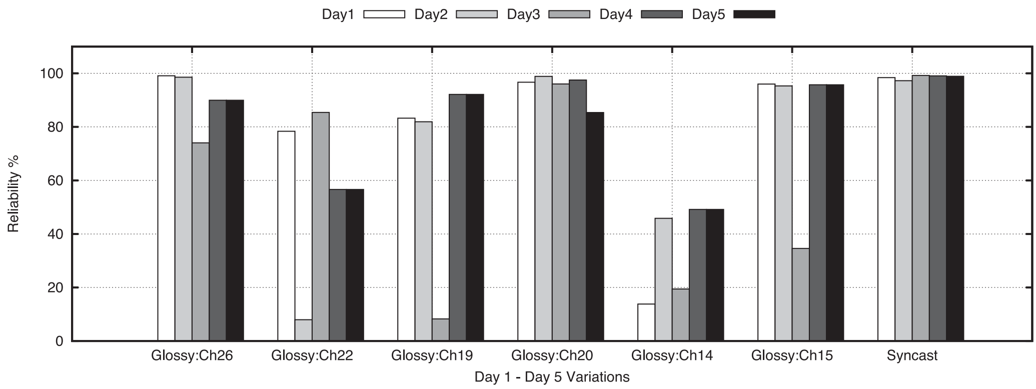


Fig. 17. Minimum reception reliability recorded on five consecutive days.

		Day 1		Day 2		Day 3		Day 4		Day 5		Summary	
		Avg. (%)	Min. (%)	Avg. (%)	Min. (%)	Avg. (%)	Min. (%)	Avg. (%)	Min. (%)	Avg. (%)	Min. (%)	Avg. (%)	Min. (%)
Glossy	Channel 26	99.3	99.08	98.89	98.55	78.48	74.02	92.52	89.94	79.68	78.89	89.774	74.02
	Channel 22	83.61	78.34	8.25	7.95	87.58	85.4	67.01	56.62	55.29	68.02	60.348	7.95
	Channel 19	84.99	83.26	83.18	81.9	17.28	8.25	94.89	92.14	86.31	84.73	73.33	8.25
	Channel 20	96.82	96.69	99.09	98.85	98.66	96.02	98.17	97.5	85.73	85.37	95.694	85.37
	Channel 14	18.66	13.82	64.62	45.82	32.11	19.42	55.91	49.16	57.82	71.98	45.824	13.82
	Channel 15	96.3	96.01	95.48	95.3	49.97	34.57	96.39	95.73	97.27	97.81	87.082	34.57
Syncast	Multichannel	99.03	98.39	98.2	97.25	98.9	99.22	98.39	99.06	99.04	98.9	98.712	97.25

interference into consideration, which is predominant during weekdays, the experiment spanned over both weekdays and weekends.

We evaluate Syncast, which is a multichannel protocol against Glossy operating over different IEEE 802.15.4 channels and observed the average and minimum reception reliability. Since Glossy operates on a single channel, it cannot exploit channel diversity. Thus, when a particular channel condition deteriorates, the reception reliability drops considerably.

On the other hand, Syncast not only tries to disseminate using only the highly reliable communication links but also over multiple channels. Thus, even if a channel gets interfered, Syncast overcomes it by exploiting capture effect over multiple channels.

The experimental results reported in the table highlights that Glossy is indeed channel dependent. The reliability goes as low as 7.95% on IEEE 802.15.4 channel 22, which is highly influenced by external Wi-Fi interference. Even on IEEE 802.15.4 channel 26, which does not overlap with the commonly used Wi-Fi channels, reliability reduces to 74.02% on day 3.

Syncast, which uses all five channels simultaneously, performs consistently well on all 5 days. This shows that the use of strong and good links can mitigate the effect of channel variations. The variation of reception reliability on different channels can be compared visually in Figure 17, where we plot the minimum observed reception reliability of Glossy on a set of IEEE 802.15.4 channels against the minimum observed reception reliability of Syncast for the same 5 days. It can be seen that Syncast maintains consistent performance despite problems on certain channels experienced by Glossy.

6.1.2. Influence of Packet Size. It is well known that large packets are more influenced by channel dynamics and susceptible to packet corruptions [Sankarasubramaniam et al. 2003]. To see the effect of packet size on Syncast, we generated a dissemination tree based on the link measurements on day 1 and flooded 500 packets, each of length 8, 64,

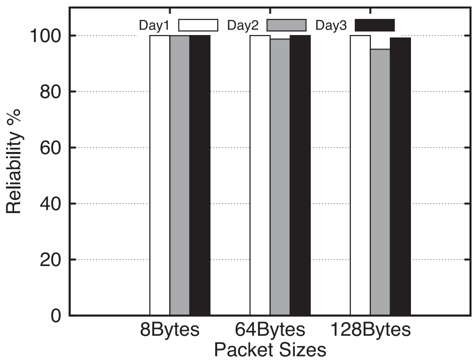


Fig. 18. Influence of packet size on reception reliability and the stability of route links.

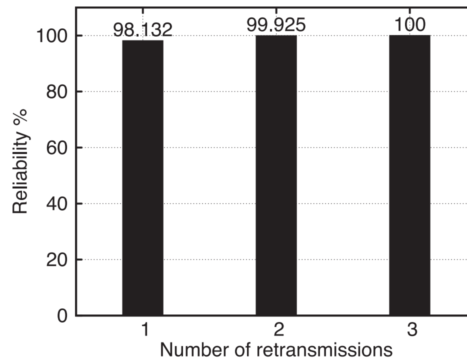


Fig. 19. Impact of packet retransmissions on the reception reliability.

and 128 bytes over the same dissemination tree for three consecutive days. Looking at Figure 18, we find that the reception reliability for Syncast remains good for all disseminated packet sizes. In addition, since the same dissemination tree is used for all 3 days, the result also shows that the links selected in the topology construction remain good and stable over a long period.

6.1.3. Influence of Retransmissions. Since we allow only one transmission per node, there is always the issue of reduced reliability due to the presence of cross-technology interference. Like Glossy, which maintains high reliability by allowing multiple retransmissions and by exploiting spatial diversity, Syncast can benefit from the same. To see the impact of retransmissions of packets on Syncast, we performed dissemination of 5,000 packets on Indriya and recorded the reception reliability by varying the maximum number of transmissions allowed per node from one to three. Even though a single transmission gives good packet reception reliability, as expected, packet retransmission results in significant improvement, as can be observed from Figure 19.

However, when retransmissions are allowed in Syncast to improve reliability, the question of latency and energy consumption arises, which we discuss next.

6.2. Latency

In applications involving flooding of control packets for which synchronous transmission was first used, latency is an important issue. It not only makes the system responsive but also helps indirectly in making the protocols energy efficient, as we soon witness.

Flooding of a data packet leads to the formation of a dissemination tree implicitly. This tree is different for different packet flows because of the dynamic nature of the communication channel. Any network deployment can be defined in terms of the maximum number of hops by which all nodes would be reachable. The number of hops translates directly to the latency since the leaf nodes and the intermediate relay nodes have to wait for that number of hops for the packet to arrive in a multihop network. Moreover, for a synchronized protocol, packet relay over a single hop takes a deterministic amount of time even after incorporating channel hopping. as shown in Figure 16.

Every node has an expected hop number at the maximum transmission power level at which it should receive the packets. Any hop number greater than that expected can count toward additional latency for which the radio needs to remain awake. At the maximum transmission power level, Indriya is a five-hop network if we choose the

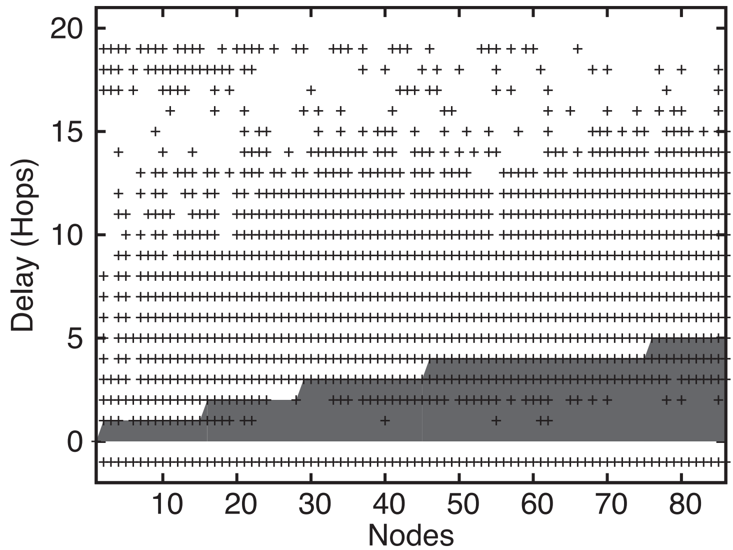


Fig. 20. Comparison of latency incurred for Syncast and Glossy.

strong links. Since Syncast floods a packet along a dissemination tree and is perfectly synchronized, all nodes know the hop at which it would either receive the packet or miss it. We compare the latency incurred at the nodes in terms of hops for Syncast and Glossy with NTX (the maximum number of transmissions allowed per node per flooding phase) fixed at 1. The task is to disseminate 5,000 maximum-size packets over IEEE 802.15.4 channel 19.

We plot the results in Figure 20, where the x-axis represents the node ID and the y-axis represents the delay in terms of hops. The shaded area indicates the hop at which the corresponding nodes receive the disseminated packet for Syncast. Each plotted point on the figure corresponds to the hopcount at which that node received one of the 5,000 disseminated packets. For any node, a point indicating a hopcount of –1 denotes that it did not receive that packet successfully. Although Syncast is able to deliver its packet reliably according to the allocated time slot with a maximum latency of 5 hops, it is rather surprising that the latency for Glossy can be as large as 19 hops. The combined effect of the external interference and the scalability problem can increase the latency further due to a larger number of required retransmissions. Interestingly, there are several cases where packet reception times for Glossy are shorter than Syncast. This is because these nodes receive their packets making use of intermediate quality links [Doddavenkatappa et al. 2011b], which can have a longer range than the strong links used in Syncast.

Figure 21 shows the distribution of the depth of the 5,000 trees dynamically generated by Glossy for each of the 5,000 disseminated packets. It can be seen that if we limit the dissemination latency to a maximum of 5 hops, only 47 or 0.0094% of dissemination would have successfully completed. With a maximum of 10 hops, 352 or 0.7% of the disseminations would still remain incomplete. Hence, to have high reliability in Glossy, there is a need to set a high threshold for maximum latency.

To understand the latency-reliability trade-off better, we analyze how packet dissemination occurs in more detail. Let us consider Figure 22, which shows an illustration of a simple synchronous transmission–based protocol in action. Optimally, all nodes are expected to receive the packets with a maximum latency of two hops as

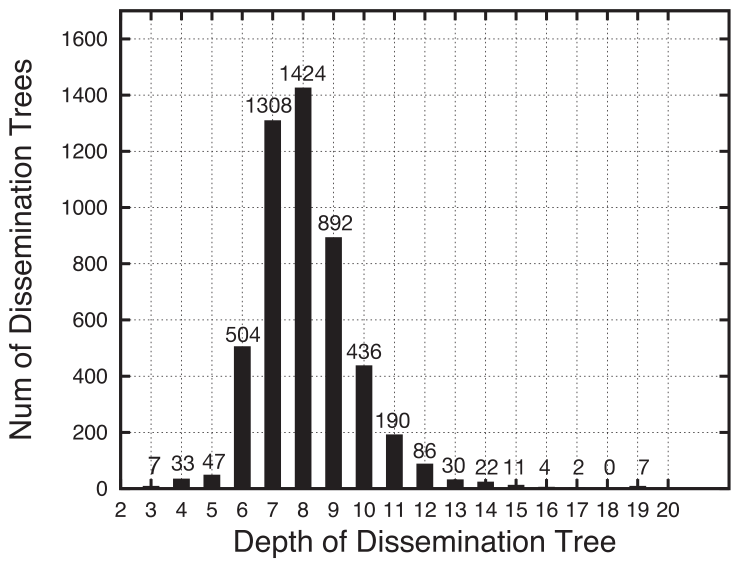


Fig. 21. Number of trees having a certain depth.

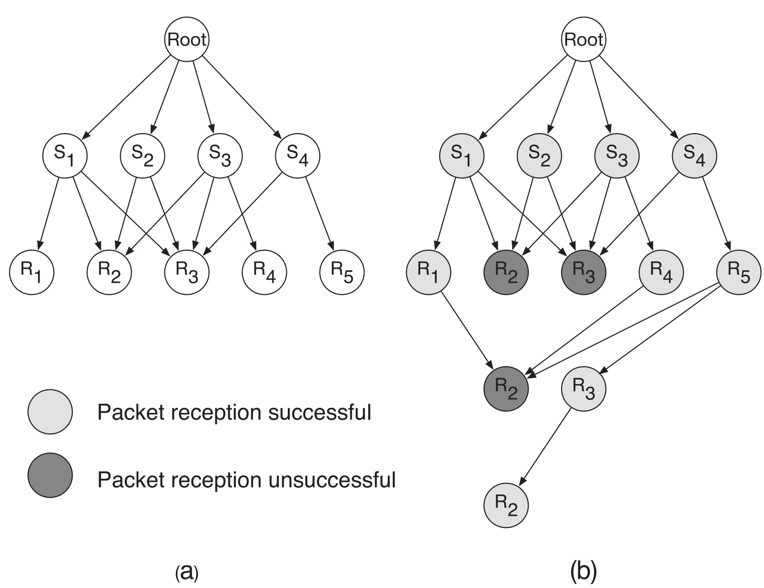


Fig. 22. Synchronous transmission–based protocol in action.

shown, assuming PRR = 1.00 for all of the links. When the root broadcasts the first packet, all of the first hop nodes receive the packet reliably since only one transmitter is active and there is no interference. However, at the second hop, only nodes R_1, R_4, and R_5 could receive reliably, whereas nodes R_2 and R_3 experienced packet loss. These unfortunate nodes, still waiting to receive a packet, get "demoted" to the next hop. At this hop, only node R_3 could successfully receive from node R_5, whereas node R_2 again fails to receive due to interference from nodes R_1, R_4, and R_5. Finally, in the fourth

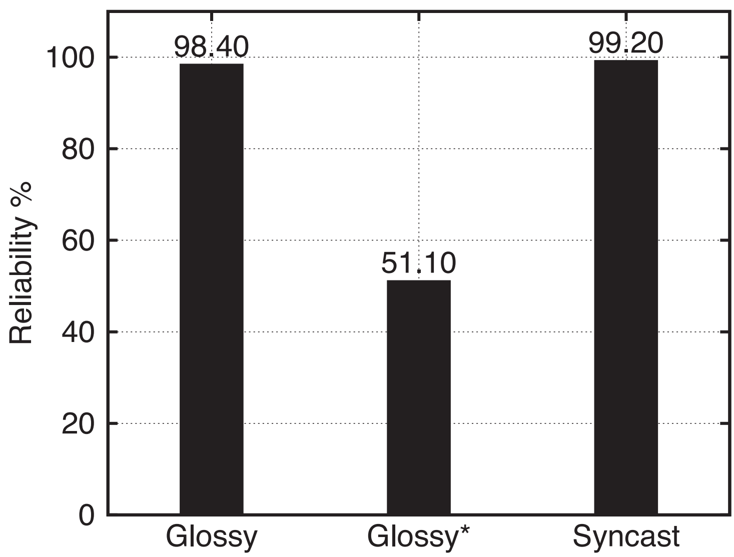


Fig. 23. Reception reliability comparison.

hop, it receives the packet from R_3 after an additional latency of two hops. In practice, such delays in packet reception are even more severe, as can be seen from Figure 21.

One question to consider is whether the latency of Glossy can be improved if the nodes only transmit/receive along a tree topology, similar to Syncast. The difference is that instead of running over multiple channels to minimize interference as in Syncast, this modified version of Glossy, which we call *Glossy**, uses only one channel. This result is shown in Figure 23. Since no smart channel assignment strategy is involved, restricting transmission to a single-channel topology leads to a significant performance drop because of the pronounced scalability issue. The reception reliability of Glossy* was low at 51.1%, whereas both Glossy and Syncast achieve reliabilities of 98.4% and 99.2%, respectively.

6.3. Energy

The final evaluation parameter considered is the energy consumption. We approximate energy consumption by the average radio-on-time over all nodes.

For Syncast, since the hop at which a node would receive the disseminated packet in Syncast is deterministic due to the underlying tree and the dissemination is synchronized, radio-on-time for the nodes is two times the packet-on-air time (time to receive and transmit), which is roughly 2*4.09ms = 8.18ms for 128 bytes IEEE 802.15.4 packets. When the node knows that it is not supposed to be transmitting or receiving, it can turn off its radio.

For Glossy, the actual transmission and reception times vary. Furthermore, before a node meets its maximum transmission count (NTX), it has to stay awake. The likelihood and exact time of completing NTX transmissions remain uncertain. Hence, the radio-on-time cannot be determined and is measured through experiments. We ran Glossy for various values of NTX per node and fixing the flooding phase duration to a sufficiently large value of 250ms to eliminate packet losses due to nodes turning their radio off on timeout. Table I shows the recorded average, maximum, and minimum radio-on-time for the nodes on Indriya for the two protocols.

Table I. Comparison of the Radio-on-Time for a Typical Glossy Flooding Against Syncast

	NTX	Rel. (%)	Avg. T (ms)	Max. T (ms)	Min. T (ms)
Syncast	1	98.57	8.18	8.18	8.18
	2	99.98	16.36	16.36	16.36
	3	99.99	24.54	24.54	24.54
Glossy	2	95.02	56.84	179.14	17.82
	3	98.52	63.96	210.13	26.77
	4	99.14	79.52	236.12	35.71
	5	99.36	92.49	241.60	44.66
	6	99.45	105.23	244.45	53.60

Whereas for NTX_2 the average radio-on-time for Syncast was 16.360ms, the average radio-on-time for Glossy for the same NTX_2 was observed to be 56.841ms. In the worst case, it went as high as 179.144ms because the nodes had to keep their radio on until they transmit at least twice. We can conclude that Syncast not only proves to be extremely reliable but is also responsive and energy efficient, with energy savings varying from 52% to 92% depending on the configuration.

7. CONCLUSIONS

In this work, we presented observations on physical layer capture and show that it occurs on links that are good (high PRR) and strong (high RSSI values). As these links retain performance over time and on different channels, protocols can be designed to take advantage of these observations by restricting transmissions only along these links. We propose Syncast, a protocol to improve synchronous transmission by exploiting capture effect over multichannels. The evaluation shows that Syncast provides a robust, reliable, and scalable data dissemination of large packets with lower end-to-end delay and up to 92% reduced radio-on-time.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for valuable comments and suggestions that helped improve the quality of presentation.

REFERENCES

Jens C. Arnbak and Wim Van Blitterswijk. 1987. Capacity of slotted ALOHA in Rayleigh-fading channels. *IEEE Journal on Selected Areas in Communications* 5, 2, 261–269.

Doug Carlson, Marcus Chang, Andreas Terzis, Yin Chen, and Omprakash Gnawali. 2013. Forwarder selection in multi-transmitter networks. In *Proceedings of the 9th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'13)*. IEEE, Los Alamitos, CA, 1–10.

Manjunath Doddavenkatappa and Mun Choon. 2014. P^3: A practical packet pipeline using synchronous transmissions for wireless sensor networks. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (IPSN'14)*. IEEE, Los Alamitos, CA, 203–214.

Manjunath Doddavenkatappa, Mun Choon Chan, and Akkihebbal L Ananda. 2011a. Indriya: A low-cost, 3D wireless sensor network testbed. In *Proceedings of the 7th International Conference on Testbeds and Research Infrastructures (TRIDENTCOM'11)*. 302–316.

Manjunath Doddavenkatappa, Mun Choon Chan, and Ben Leong. 2011b. Improving link quality by exploiting channel diversity in wireless sensor networks. In *Proceedings of the 32nd IEEE Real-Time Systems Symposium (RTSS'11)*. IEEE, Los Alamitos, CA, 159–169.

Manjunath Doddavenkatappa, Mun Choon Chan, and Ben Leong. 2013. Splash: Fast data dissemination with constructive interference in wireless sensor networks. In *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI'13)*. 269–282.

Wan Du, Jansen Christian Liando, Huanle Zhang, and Mo Li. 2015. When pipelines meet fountain: Fast data dissemination in wireless sensor networks. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys'15)*. ACM, New York, NY, 365–378.

Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. 2004. Contiki—a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*. IEEE, Los Alamitos, CA, 455–462.

Prabal Dutta, Stephen Dawson-Haggerty, Yin Chen, Chieh-Jan Mike Liang, and Andreas Terzis. 2010. Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10)*. ACM, New York, NY, 1–14.

Prabal Dutta, Razvan Musaloiu-e, Ion Stoica, and Andreas Terzis. 2008. Wireless ACK collisions not considered harmful. In *Proceedings of the 7th ACM Workshop on Hot Topics in Networks (HotNets'08)*. ACM, New York, NY, 1–6.

Federico Ferrari, Marco Zimmerling, Luca Mottola, and Lothar Thiele. 2012. Low-power wireless bus. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (SenSys'12)*. ACM, New York, NY, 1–14.

Federico Ferrari, Marco Zimmerling, Lothar Thiele, and Olga Saukh. 2011. Efficient network flooding and time synchronization with Glossy. In *Proceedings of the 10th International Conference on Information Processing in Sensor Networks (IPSN'11)*. IEEE, Los Alamitos, CA, 73–84.

Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. 2009. Collection tree protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09)*. ACM, New York, NY, 1–14.

Vlado Handziski, Andreas Köpke, Andreas Willig, and Adam Wolisz. 2006. Twist: A scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *Proceedings of the 2nd International Workshop on Multi-Hop Ad Hoc Networks: From Theory to Reality (REALMAN'06)*. ACM, New York, NY, 63–70.

Jonathan W. Hui and David Culler. 2004. The dynamic behavior of a data dissemination protocol for network programming at scale. In *Proceedings of the 2nd Conference on Embedded Networked Sensor Systems (SenSys'04)*. ACM, New York, NY, 81–94.

Xiaoyu Ji, Yuan He, Jiliang Wang, Kaishun Wu, Ke Yi, and Yunhao Liu. 2013. Voice over the dins: Improving wireless channel utilization with collision tolerance. In *Proceedings of the 21st IEEE International Conference on Network Protocols (ICNP'13)*. IEEE, Los Alamitos, CA, 1–10.

Olaf Landsiedel, Federico Ferrari, and Marco Zimmerling. 2013. Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys'13)*. ACM, New York, NY, 1.

Chieh-Jan Mike Liang, Nissanka Bodhi Priyantha, Jie Liu, and Andreas Terzis. 2010. Surviving Wi-Fi interference in low power Zigbee networks. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10)*. ACM, New York, NY, 309–322.

Roman Lim, Federico Ferrari, Marco Zimmerling, Christoph Walser, Philipp Sommer, and Jan Beutel. 2013. Flocklab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In *Proceedings of the 12th International Symposium on Information Processing in Sensor Networks (IPSN'13)*. IEEE, Los Alamitos, CA, 153–165.

Jiakang Lu and Kamin Whitehouse. 2009. *Flash Flooding: Exploiting the Capture Effect for Rapid Flooding in Wireless Sensor Networks*. IEEE, Los Alamitos, CA.

Mobashir Mohammad, XiangFa Guo, and Mun Choon Chan. 2016. Oppcast: Exploiting spatial and channel diversity for robust data collection in urban environments. In *Proceedings of the 15th ACM/IEEE International Symposium on Information Processing in Sensor Networks (IPSN'16)*. IEEE, Los Alamitos, CA, 1–12.

Joseph Polastre, Jason Hill, and David Culler. 2004. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*. ACM, New York, NY, 95–107.

Joseph Polastre, Robert Szewczyk, and David Culler. 2005. Telos: Enabling ultra-low power wireless research. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN'05)*. IEEE, Los Alamitos, CA, 364–369.

Bhaskaran Raman, Kameswari Chebrolu, Sagar Bijwe, and Vijay Gabale. 2010. PIP: A connection-oriented, multi-hop, multi-channel TDMA-based MAC for high throughput bulk transfer. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10)*. ACM, New York, NY, 15–28.

Y. Sankarasubramaniam, I. F. Akyildiz, and S. W. McLaughlin. 2003. Energy efficiency based packet size optimization in wireless sensor networks. In *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications (SNPA'03)*. IEEE, Los Alamitos, CA, 1–8.

Dongjin Son, Bhaskar Krishnamachari, and John Heidemann. 2006. Experimental study of concurrent transmission in wireless sensor networks. In *Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys'06)*. ACM, New York, NY, 237–250.

Kannan Srinivasan, Maria A. Kazandjieva, Saatvik Agarwal, and Philip Levis. 2008. The β-factor: Measuring wireless link burstiness. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08)*. ACM, New York, NY, 29–42.

Yin Wang, Yuan He, Xufei Mao, Yunhao Liu, and Xiang-Yang Li. 2013. Exploiting constructive interference for scalable flooding in wireless networks. *IEEE/ACM Transactions on Networking* 21, 6, 1880–1889.

Dingming Wu, Chao Dong, Shaojie Tang, Haipeng Dai, and Guihai Chen. 2014. Fast and fine-grained counting and identification via constructive interference in WSNs. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (IPSN'14)*. IEEE, Los Alamitos, CA, 191–202.

Dingwen Yuan, Michael Riecker, and Matthias Hollick. 2014. Making glossy networks sparkle: Exploiting concurrent transmissions for energy efficient, reliable, ultra-low latency communication in wireless control networks. In *Proceedings of the 11th European Conference on Wireless Sensor Networks (EWSN'14)*. 133–149.

Xinyu Zhang and Kang G. Shin. 2010. Chorus: Collision resolution for efficient wireless broadcast. In *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM'10)*. IEEE, Los Alamitos, CA, 1–9.

Received July 2015; revised August 2016; accepted November 2016