Reliability Through Frequency Diversity: Why Channel Hopping Makes Sense*

Thomas Watteyne, Ankur Mehta, Kris Pister
Berkeley Sensor & Actuator Center
University of California
Berkeley, CA 94720-1774
{watteyne,mehtank,pister}@eecs.berkeley.edu

ABSTRACT

Wireless sensor networks (WSNs) face the challenge of ensuring end-to-end communication while operating over individually unreliable wireless links. This paper addresses channel hopping, a class of frequency diverse communication protocols in which subsequent packets are sent over different frequency channels. Channel hopping combats external interference and persistent multipath fading, two of the main causes of failure along a communication link.

This paper is, to our knowledge, the first to address the impact of channel hopping on routing. We simulate the performance of channel hopping and single channel solutions on connectivity traces gathered from a real-world office WSN deployment.

Results indicate that the most basic channel hopping protocol increases connectivity along communication links, improving network efficiency (measured by the expected transmission count ETX) by 56% and network stability (measured by the average churn) by 38%. Further improvement can be achieved through the use of whitelisting – selective channel hopping over a subset of the available frequencies.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

General Terms

Reliability, Experimentation, Algorithms

Keywords

Channel Hopping, Reliability, Experimental Results

*This work was partially supported by the California Energy Commission.

PE-WASUN'09, October 28–29, 2009, Tenerife, Canary Islands, Spain.

1. INTRODUCTION

Commercial products and real-world deployments of wireless sensor networks (WSNs) are faced with harsh reliability issues, which are mainly attributed to the unreliable nature of individual wireless links. Reliability can be achieved by exploiting diversity – connection redundancy over different link parameters. A good routing protocol provisions alternate paths which are used to circumnavigate areas of poor or inconsistent connectivity. Such *path diversity* can be combined with *frequency diversity* which exploits the fact that external interference and persistent multipath fading vary across frequency channel. This paper evaluates the benefits of frequency diversity with a particular focus on its impact on network routing.

Specifically, we study *channel hopping*, a solution which exploits frequency diversity by sending subsequent packets on different channels. Because links are coherent across short time scales but not frequency [1], a transmission failure on a particular channel is likely to result in a failed retransmission whereas a different channel would behave independently. Channel hopping requires time synchronization to ensure that the transmitter and its intended recipient are always operating on the same frequency.

Channel hopping can come in several flavors. In the simplest *blind channel hopping*, each node uniformly hops over all available channels (16 in the case of an IEEE802.15.4 radio). A more advanced variant uses *whitelisting* on link-by-link basis. In this variant, two neighbor nodes agree upon a subset of the available channels and hop only on that subset. Choosing an optimal subset requires nodes to maintain statistics to decide which channels to use and when to update the whitelist.

Using channel hopping at the MAC layer has non-intuitive impact on routing. On the one hand, because the links between nodes have different characteristics from channel to channel, the connectivity graph is different as a node changes channel, which could lead to routing instabilities and poor performance. On the other hand, because channel hopping causes the performance of a link to be averaged over multiple channels, channel hopping could, in fact, yield more stable routing paths.

How does channel hopping affect the routing protocol? Can channel hopping over only a subset of whitelisted channels further increase efficiency? If yes, how many channels should such a whitelist contain? This paper contributes to answering those questions by simulating the performance of a channel hopping network on connectivity traces gathered from a real-world office WSN deployment. We quantify the impact of channel hopping on routing in the canonical case of a network with a single sink. We choose gradient routing because (1) it identifies optimal routes according to some metric and hence abstracts a best-case routing protocol and (2) it is currently standardized by the IETF workgroup ROLL[1] and is therefore expected to be widely used in future commercial products and deployments.

The contributions of this paper are:

- We show that, compared to using a single channel solution, blind channel hopping increases network connectivity by 26%, improves efficiency by 56% and network stability by 38%. Network efficiency and network stability are quantified by the average ETX and network churn, respectively, two indicators defined in Section 2.2.

- We show how whitelisting can further increase the performance of the network. Performance depends on the size of the whitelist; a whitelist of 6 channels yields the best results. Whitelisting requires state to be maintained and exchanged between neighbors. Whether the performance increase justifies this overhead is an open question.

The remainder of this paper is organized as follows. Section 2 details related work on frequency agile communication and gradient routing. It shows how current radio chips efficiently handle frequency agility, and how channel hopping and gradient routing are being standardized and used in commercial products. Section 3 details the experimental setting used in this paper. We justify the use of connectivity traces and show the impact of external interference on connectivity. Section 4 presents the main results of the paper. It shows that channel hopping improves connectivity, efficiency and network stability, as detailed above. We also discuss the overhead of using whitelisting. Section 5 concludes this paper and presents future work.

2. RELATED WORK

2.1 Frequency-Agile Communication

Hardware Support.

Frequency-agile communication requires nodes to change the frequency channel they transmit on or listen to often (e.g. every few tens of milliseconds). Luckily, radio chips have become very efficient at doing this. As an example, all IEEE802.15.4-compliant [2] radio chips – the *de facto* standard for WSN hardware – switch channels in less than $192\mu s$. Such chips live in most popular platforms, such as TelosB, MICAz, IRIS, SUNspot, IMote or EPIC. Moreover, non-IEEE802.15.4 chips such as Texas Instruments' CC1100, CC1101 and CC2500 [3] feature low turn-around times of $88.4\mu s$. Combined with the fact that typical clocks drift by $10ppm$ or less, fast channel hopping capabilities have made frequency-agile communication efficient.

With nodes continuously changing channels, senders and receivers need to be tightly synchronized for their radios to be on the same channel when communicating. Protocols presented below allow for relative de-synchronization of up to a few milliseconds. As a result, the vast majority of frequency-agile MAC protocols are Time Division Multiple Access (TDMA)-based: time is cut into slots and nodes maintain synchronization with their neighbors. We present a comprehensive overview of the latest frequency-agile MAC protocols and standardization efforts below.

Frequency-Agile MAC Protocols.

Lightweight MAC [4] (**LMAC**) assigns slots to nodes in a distributed way. **Multichannel LMAC** [5] proposes, when all slots are assigned, to pick a slot on another frequency. The number of potential slots is roughly multiplied by the number of frequency channels, which allows more nodes to communicate than LMAC. Omnet++ simulations show that the use of multiple channels decreases the number of active nodes while reducing collisions.

Y-MAC [6] is primarily designed to decrease latency. Nodes are synchronized and reception slots are assigned to each node on a common base channel. In case multiple packets need to be sent between neighbor nodes, successive packets are sent, on a different frequency. As a result, bursts of messages ripple across channels, which reduces latency. Y-MAC was implemented in the RETOS operating system on the TmoteSky motes and compared to LPL. With 8 sec resynchronization period and 5 frequency channels, the idle duty cycle when sending one packet every $10s$ is around 7%.

Time Synchronized Mesh Protocol (**TSMP**)[7] was designed to improve reliability. TSMP employs frequency hopping: different links use different frequency channels and the same link hops during its lifetime across different channels. This reduces the impact of narrow-band interference and persistent multipath fading. [8] presents experimental results in which 44 nodes run TSMP for 26 days in a printing facility. The authors show how channel hopping, combined with a retransmission policy, yields an end-to-end delivery ratio of 99.999%. TSMP uses a central coordinator which retrieves the list of nodes, their neighbors and their traffic requirements. This allows it to construct a schedule which is then communicated back to the network.

Critical applications require reliable solutions, and channel hopping is one answer to this need. With the success of proprietary solutions such as TSMP, standardization bodies have been working on similar solutions, as detailed in the next section.

Standardization Efforts.

IEEE802.15.1 [9] is the technology used by the Bluetooth consortium. The physical layer features 79 1-MHz channels in the 2.4GHz ISM band. Devices wishing to communicate group around a leader and synchronize to that leader's clock. Time is sliced up into $625\mu s$-long slots, and a hashing function translates the leader's address into a channel hopping pattern. All nodes follow that pattern, changing channels roughly 1600 times per second. A user can configure a device to hop on only a subset of the 79 channels, a simple form of whitelisting.

The HART Communication Foundation standardizes embedded networking solutions for industrial applications. Their wireless extension, called **WirelessHART** [10], uses a central controller to schedule communication. WirelessHART uses IEEE802.15.4 radios to hop on 15 frequency channels in the 2.4GHz band. Similarly to TSMP, reliability is increased by having each node maintain connectivity to at least two parent nodes in the routing graph, enabling the

[1] www.ietf.org/html.charters/roll-charter.html

network to resist link failures. Additionally, whitelisting is a user configurable feature of the controller, based on the proximity of other wireless networks that are in the same physical environment.

Another industrial wireless standardization body is the ISA100 Wireless Compliance Institute. Their latest standard **ISA100a** [11] is similar in essence to TSMP or WirelessHART, yet features interesting channel hopping mechanisms. Successive channels in the hopping pattern are separated by at least 15 MHz (three IEEE802.15.4 channels). When retransmissions occur, they will not encounter or cause interference in the same IEEE802.11 (Wi-Fi) channel. Moreover, whitelisting limits operation to a subset of channels. At a global scope, a system manager can block certain radio channels that are not working well or are prohibited by local policy. At a local scope, Adaptive Channel Hopping (ACH) enables whitelisting on a link-by-link basis. The MAC layer of a node bans channels that it deems problematic due to a history of poor connectivity, potentially with granularity of a specific channel used to communicate with a specific neighbor.

The workgroup **IEEE802.15.4E** focuses on enhancing the MAC protocol proposed in IEEE802.15.4, while keeping the same physical layer. In its current proposal [12], nodes can switch between different hopping sequences. Similarly to TSMP, slots can be added/removed during the lifetime of the network. Note that an open-source implementation of this proposal, called TSCH, is presented in [13].

Summary.

In summary, protocols and standards use channel hopping to combat narrow-band interference (e.g. in the presence of IEEE802.11 networks) and persistent multipath fading (mostly present in indoor deployments). All protocols use a TDMA approach to agree upon a schedule. In most, whitelisting can be applied globally (IEEE802.15.1, WirelessHART, ISA100a, IEEE802.15.4E); in some, it can be applied locally on a link-by-link basis (ISA100a). Channel hopping mechanisms have been designed and are being standardized. However, whereas [14] identifies the problem, it is unclear what the impact of channel hopping on routing protocols is.

We evaluate the impact of channel hopping on gradient routing, which is currently standardized by the IETF and is therefore expected to be widely used in future commercial products and deployments. We provide a comprehensive overview of gradient routing in the following section.

2.2 Gradient Routing

We consider the network uses a gradient routing protocol to find multihop paths from the nodes to the sink. We choose gradient routing because (1) it identifies optimal routes according to some metric and hence abstracts a best-case routing protocol and (2) it is currently standardized by the **IETF** – through his work-group **ROLL** – and is therefore expected to be widely used in future commercial products and deployments. Because gradient routing is particularly suitable for convergecast WSNs, it has been in use for almost a decade, and is known as Gradient Based Routing [15, 16, 17], Gradient Broadcast [18], Tree Discovery [19] or Collection Tree Routing [20].

Gradient Set Up.

In gradient routing, all nodes have an internal variable called `height`, a neighbor table, and a means to measure the cost of communicating with each neighbor. Nodes periodically send out beacons containing their `height` and listen for beacons sent by neighbors. Upon receiving a beacon, a node extracts the `height` field, to which it adds the cost of sending a packet to that neighbor. It places this `neighbor height` value in its neighbor table. A node sets its own `height` to the smallest `neighbor height` value in its neighbor table. The sink node always keeps its `height` to zero.

This creates a gradient of heights increasing from the sink outwards. When a node sends a message, it sends it to the neighbor which features the smallest `neighbor height` in its neighbor table. As long as the connectivity graph does not change (i.e. links do not (dis)appear), the gradient ensures messages reach the sink following the shortest path. Depending on how link cost is evaluated, *shortest* takes different meanings. If each existing link has a unit cost, the shortest path is the path with the smallest number of hops.

Network Efficiency.

Consider a link connecting node A to node B. The Packet Delivery Ratio (PDR $\in$ [0..1]) of that link is the fraction of packets sent by A which are received by B. Assuming Bernoulli behavior, this means that A needs to retransmit on average $\frac{1}{PDR}$ times before B successfully receives the packet. $\frac{1}{PDR}$ is called Expected Transmission Count (ETX)[21]. It is a good metric to capture the cost of a link, as packet retransmission is a major source of energy and time expenditure.

When using ETX as a link metric for setting up a gradient, the height of a node indicates how many times a message sourced at that node is transmitted before it reaches the sink. These transmissions include the hops from node to node, as well as the retransmissions needed upon link failure. An example topology is shown in Fig. 1. Assuming the message generation rate is homogeneous among all nodes, the lower the average height of the nodes, the better. We therefore use **average height** as a metric to quantify *network efficiency.*

Network Dynamics.

The routing gradient needs to reflect the changes in network topology when links come and go. As stated in [20], the beaconing period poses a basic tradeoff. A small period reduces how stale information can be, but uses more bandwidth and energy. A large period uses less bandwidth and energy but can let topological problems persist for a long time. The Collection Tree Protocol (**CTP**)[20] uses the Trickle algorithm [22] to regulate the beaconing interval. In the absence of topological changes, this interval is regularly doubled until it reaches a maximum value which triggers only a few beacons per hour. Upon topological changes, the interval is reduced to allow for fast gradient re-convergence. Experimental results on 12 different testbeds show that CTP requires 73% fewer beacons than a solution with a fixed 30-second beacon interval, for an idle duty cycle of 3%.

The more dynamic the network, the more often beacons have to be exchanged; this increases energy consumption and network congestion. Network dynamics should therefore be kept low. One metric to quantify *network dynamics* is **network churn**, i.e. the portion of nodes in the network

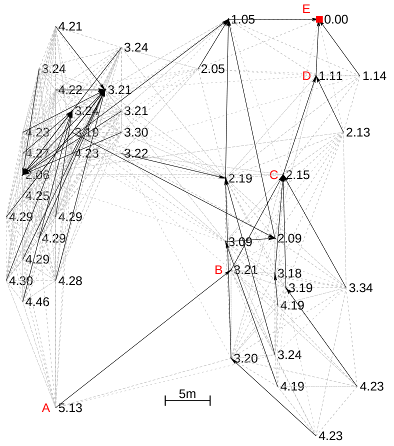

Figure 1: Topology at run 1, on channel 11, as detailed in Section 3. Dotted lines indicate links with a strictly positive PDR. Numbers indicate the heights around sink node E. Arrows connect nodes with their parent: a message sent from A follows path $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$.

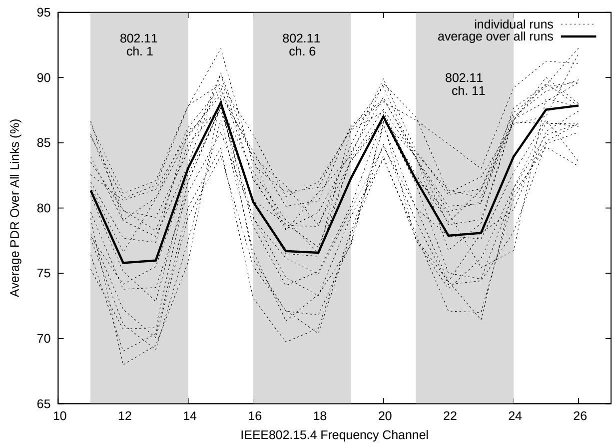

Figure 2: Nearby IEEE802.11 networks impact the performance of the testbed.

that change primary routing parent between two instances in time.

3. EXPERIMENTAL SETTING

We replay the behavior of single channel and channel hopping MAC protocols over connectivity traces gathered in a real-world deployment. Traces ensure fairness in the comparison as only the MAC protocol changes; such fairness is hard to achieve when rebuilding a different deployment for each protocol.

The connectivity traces are collected by J. Ortiz and D. Culler in a UC Berkeley office space[2]. They are obtained as follows. 46 IEEE802.15.4-compliant TelosB motes are deployed in a $50m \times 50m$ indoor environment, and are constantly listening for packets. One after the other, each mote transmits a burst of 100 packets – with an $20ms$ inter-packet time and a transmission power of $0dBm$ – on each of the 16 frequency channels. Timers are used to ensure that all nodes switch channels simultaneously. Note that, because bursts are sent in sequence, there are no collisions. All non-transmitting nodes record the timestamp of the packets received, their source address, and the frequency channel the packets are received on. After all 46 nodes have sent a burst, each node reports what packets it has received. This process is repeated in 17 runs. A single run completes in 13 minutes; several hours separate subsequent runs.

A total of 12 million packet receptions form the dataset used for this paper. By counting how many of the 100 sent packets are received, one can know the Packet Delivery Ratio (PDR) between any pair of nodes in the network, on any of the 16 channels, at any of the 17 runs.

Several IEEE802.11 (Wi-Fi) networks run in the deployment area, using IEEE802.11 channels 1, 6 and 11. Fig. 2 shows the impact of this external interference on the PDR averaged over all links. Only IEEE802.15.4 channels 11, 15, 25 and 26 are free from IEEE802.11 interference, making them good choices for single channel MAC solutions.

Several approaches can be taken when designing a MAC protocol:

- in a **single channel** solution, all nodes in the network operate at the same pre-defined channel for the lifetime of the network.
- in a **channel-hopping** solution, subsequent packets "hop" from channel to channel according to a hopping pattern common to both end-points of a single-hop link. In **blind channel hopping**, the hopping pattern uses all 16 available channels. A more elaborate approach, called **whitelisting**, uses only a subset of best channels. We consider whitelisting at a link-by-link granularity. This requires nodes to keep state about the quality of each channel, and some signaling overhead to make the choices by sender and receiver coherent. These overheads will be discussed in section 4.4.

4. RESULTS

At a given run, the dataset informs of the PDR of a link between any pair of nodes. As detailed in Section 2.2, we use $ETX = \frac{1}{PDR}$ as a gradient metric. The results in this section show that a channel-hopping MAC protocol achieves performances which are better than a single-channel solution operating over any of the 16 channels. In particular, we show that channel hopping requires 26% less nodes to be deployed in order to cover a given region (Section 4.1), and achieve up to 63% lower average ETX (Section 4.2) and 47% lower churn (Section 4.3). Section 4.4 discusses the overhead associated with whitelisting.

4.1 Network Connectivity

We first look at how many nodes are needed to cover the $50m \times 50m$ area with a given probability, while forming a fully connected network. We therefore randomly pick n out of the 46 nodes, and determine whether they are fully connected. They are connected when there exists a multi-hop path from any node to any node, such that each hop has a strictly positive PDR. We repeat this 500 times approximate the probability for those n nodes to be fully connected. This value is calculated for a single channel solution (operating

[2] As an online addition to this paper, the connectivity traces used in this paper are made available by the authors.

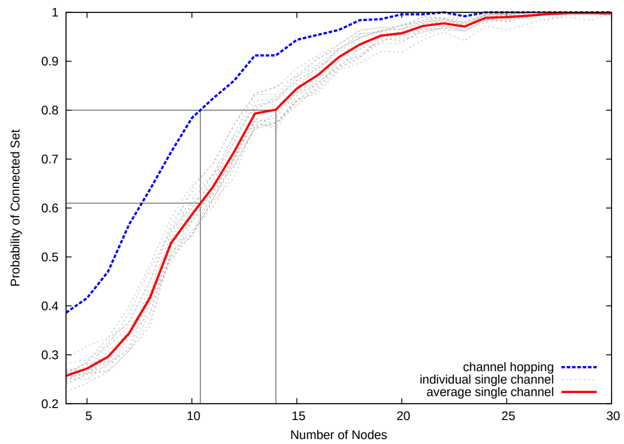

Figure 3: A channel hopping solution requires less nodes to form a fully connected network in a given area, than a single channel solution.

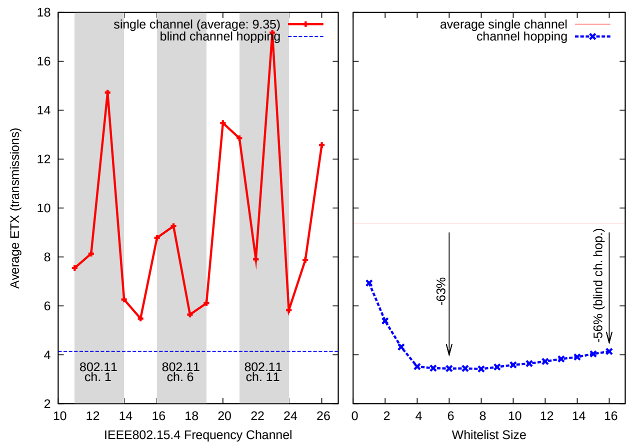

Figure 4: Blind channel hopping – whitelist size of 16 – features a 56% lower average ETX than the average single channel solution. Whitelisting enables further improvements, with the lowest average ETX measured for a whitelist size of 6.

at all possible channels) and a blind channel-hopping solution. The results, averaged over all 17 runs, are presented in Fig. 3.

Fig. 3 indicates that, to form a fully connected network with a probability of 80%, 14 nodes need to be deployed when running the network on a single channel; a blind channel hopping solution requires only 10 nodes, 26% less. Similarly, deploying 10 nodes running on a single channel yields a probability of them being fully connected of only 61%.

Channel hopping improves connectivity. This is particularly interesting when deciding how many nodes to deploy in a environmental monitoring environment. Note that this discussion is somewhat artificial as we study the probability for the network to be connected, not the probability of the sensors mounted on the nodes to sufficiently sample the physical phenomenon under study. The next sections assume all 46 nodes participate in the network, and quantify the efficiency and the dynamics of the network.

4.2 Average ETX

We use the average ETX as a primary metric to quantify the efficiency of a network. It is the number of transmissions (i.e. successful transmissions to hop from node to node, and retransmissions due to link failure) one can expect a message requires when sourced at an arbitrarily chosen node. ETX captures both the energy and the time spent for sending a packet.

Wireless links are unreliable in essence and their PDR changes over time. In gradient routing, a gradient is used after it is set up. Even if the gradient is constantly maintained, it is always slightly outdated as link characteristics change. We mimic this behavior by using, at run $i+1$, the gradient set up at run i.

We use the following steps to compute the average ETX. At run i, we obtain the topology of the network, i.e. the PDR values for all links. We choose the upper-right node to be the sink (see Fig. 1) and we compute the `heights` of all nodes in the network; each node is assigned a routing parent. At run $i+1$, we have each node send a message which follows the gradient established in run i. Because the topology has changed between runs i and $i+1$, each message may end up consuming a number of transmissions different from the number predicted at run i.

Fig. 4 depicts the average ETX when using a single channel (left) or a channel hopping solution (right). For statistical validity, results are averaged over all runs. Fig. 4 shows the impact of the chosen channel in a single channel solution, and the impact of the size of the whitelist when using channel hopping.

The average ETX when using a single channel solution varies significantly with the channel. While one could expect channels free from IEEE802.11 interference to present lower average ETX, this is not the case for channels 19 and above. This may be attributed to multipath fading, but demonstrates the danger of assuming specific channel behavior.

The average ETX of a channel hopping solution depends on the size of the whitelist. With a whitelist size of 1, each link operates at the best channel, determined at run i. At run $i+1$, however, the PDR of this channel may have dropped, yielding less favorable results. In this case, a larger whitelist makes sense as it reduces the impact of PDR variations. On the other hand, with a whitelist size of 16 (i.e. blind channel hopping), a link ends up using channels with a low PDR, yielding less favorable results. The best whitelist size is located in-between those extremes. Fig. 4 shows that the average ETX is lowest for a whitelist size of 6. This size offers the most resilience against PDR variance while excluding channels which constantly perform poorly.

It is worth noting that, even without whitelisting (which requires some state and communication overhead, as discussed in Section 4.4), blind channel hopping performs better than a single channel solution operating on any channel. It yields a 56% lower average ETX than the average single channel solution. This value increases to 63% when using a whitelist size of 6.

4.3 Network Churn

As detailed in Section 2.2, network churn – the portion of nodes in the network which change routing parent between two instances in time – quantifies the stress induced by the MAC protocol on routing. A high churn requires more in-

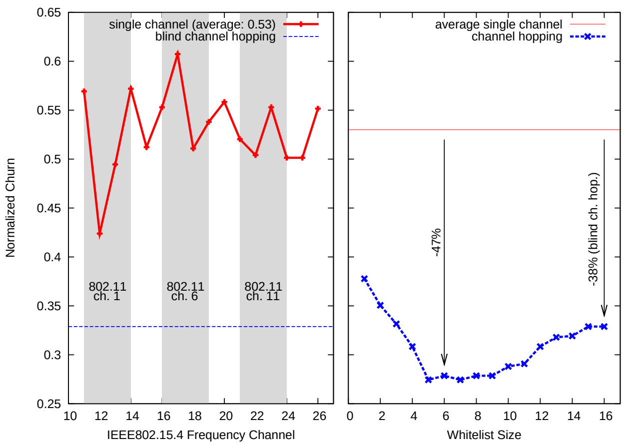

Figure 5: Channel hopping reduces network churn when compared to a single channel solution. Blind channel hopping yields a 38% decreases in network churn; using whitelisting can further decreases network churn.

network signaling to reconstruct the routing tables of the nodes. In CTP [20], this translates into a smaller beaconing interval. The lower the network churn, the better.

To measure network churn, we compare the parent of each node at runs i and $i+1$, and determine the portion of nodes that changed parent. We repeat this operation for $i \in [1..16]$ and present averaged results. Fig. 5 depicts the network churn for a single channel solution operating on the various channels, and for a channel hopping solution with different whitelist sizes. Network churn is normalized by the number of nodes in the network, i.e. a network churn of 0.3 indicates that on average 30% of the nodes change parent between successive runs.

In a single channel solution, network churn varies with the channel the network operates on, with an average of 0.53 (i.e. on average 53% of the nodes – about 24 nodes – change parents between successive runs). Blind channel hopping reduces this number by 38%. The network is inherently more stable as individual link dynamics are absorbed by averaging over several channels. Similarly to Fig. 4, network churn can be further reduced by filtering out the worst channels on each link through whitelisting. The network churn is 47% lower than the average single channel solution when using whitelist size of 6.

4.4 Discussion

Table 1 summarizes the observations made in previous sections. It shows that channel hopping makes sense. In particular, blind channel hopping (with a whitelist containing all channels) achieves better connectivity, lowers average ETX and lowers network churn compared to a single channel solution operating on any channel. It also shows that these indicators can be improved when using whitelisting, i.e. when filtering out the worst channels on a link-by-link basis. Performance in this case depends on the size of the whitelist – how many channels are not filtered out; our results advocate for a whitelist size of 6.

Blind channel hopping has been implemented and standardized, as detailed in Section 2. The cost of using blind channel hopping is that nodes need to keep synchronization, and follow the same hopping pattern. While the latter is typically a compile-time parameter which presents no real overhead, the former is often regarded as a major source of energy expenditure.

Nodes keep synchronization by periodically exchanging messages containing timestamps. The period at which this must happen depends upon the accepted de-synchronization. A typical channel hopping solution such a TSCH [13] uses timeslots of $24ms$, with guard times which allow up to $2ms$ of de-synchronization. Current crystal oscillators drift by $10ppm$; two nodes hence drift by up to $20ppm$ relative to each other (one going fast, the other slow). This translates into a maximum re-synchronization period of 100 seconds. Considering the worst case in which synchronization requires a node to be on for a complete slot, this yields an idle duty cycle of 0.024%. We therefore argue that keeping synchronization represents a small overhead.

Yet, one need to be aware that, because packets are constantly sent on different channels, a node takes more time to join a network than when using a single channel solution. A node wanting to join listens for packets, and slowly scans the different channels. It can not listen on a single channel because this might undergo deep multipath fading. Joining hence takes time (e.g. in the order of a few tens of seconds in TSCH [13]). Moreover, because nodes set up schedules to communicate to neighboring nodes, node mobility is usually poorly handled by channel hopping.

Table 1 points out the benefits of whitelisting. Whitelisting, however, comes at a price. First, it requires each node to keep statistics for all channels and for each of the links to neighbors. These statistics may consist of counting the number of received packets, which can be translated back into PDR. Assuming PDR is an integer number between 0 and 255 and a IEEE802.15.4 radio with 16 channels, if a node has 15 neighbors, these statistics require 240 bytes of RAM. A typical microcontroller, such as the MSP430 used by the popular TelosB mote, contains 10kB of RAM. From a practical perspective, the overhead of maintaining statistics is hence small.

When using whitelisting on a link-by-link basis, the nodes on each end of a link need to ensure their whitelists for that link contain the same subset of channels. If this is not the case, they may end up communicating on different channels, leading to deafness. Maintaining coherent whitelists hence requires some form of negotiation between neighbors. This needs to happen at least each time one of the neighbors decides to update its whitelist. To our knowledge, there exists no protocol which dynamically negotiate whitelists.

As a summary, channel hopping makes sense as it improves connectivity, efficiency and stability compared to a single channel solution. Blind channel hopping has been standardized, and commercial products prove that this technology can be implemented efficiently. Whitelisting further improves the performance of channel hopping, yet this requires nodes to keep state and neighbor nodes to share this state in a coherent fashion. We believe that the performance improvements of doing so is worth the price, yet designing a MAC protocol which includes these mechanisms is needed to provide an informed answer.

	Single Chan. (average)	Channel Hopping	
		w.l.⋆=16	w.l.⋆=6
number nodes 80% connection probability	14	10 (-26%)	*N.A.*
Average ETX	9.35	4.14 (-56%)	3.44 (-63%)
Norm. Churn	0.53	0.33 (-38%)	0.28 (-47%)

⋆Size of the whitelist.

Table 1: Blind channel hopping (whitelist size of 16) achieves better connectivity, network performance and network stability than a single channel hopping solution over any channel. This can be further improved by using whitelisting.

5. CONCLUSIONS

Interference and persistent multipath fading cause the PDR of a single link to vary with the channel. In an office building, a site survey can determine which of the available channels is least affected; this channel can then be used to run the WSN. A different approach is to use channel hopping, in which each packet is sent on a different channel. In this case, transient failures in links on a given channel are handled gracefully, and persistent link failures that develop after the site survey do not destabilize the network. Nonetheless, the topology of the network (the collection of links interconnecting the nodes) is different from channel to channel. To our knowledge, this is the first paper to quantify the impact of channel hopping on routing.

In this paper, we simulate the performance of single channel and channel hopping solutions on traces from a real-world office WSN deployment. We show that blind channel hopping improves connectivity, while reducing ETX by 56% and network churn by 38%. The performances can be further improved by using whitelisting on a link-by-link basis. When using whitelisting, each node maintains statistics for all channels to each of its neighbors; neighbor nodes then agree upon a subset of channels – the whitelist – over which to hop. Using this principle, we show that a whitelist containing 6 channels yields best results: average ETX is reduced by 63% and network churn by 47% compared to the average single channel solution. It is unclear whether these increased performances are worth the price of keeping state and sharing state in a coherent fashion. If not, blind channel hopping may suffice.

Designing a MAC protocol which performs whitelisting will help answering that question. We therefore plan to add whitelisting functionality in TSCH and evaluate its performances experimentally. Furthermore, we would like to determine how the best whitelist size is impacted by the gradient refresh period. If the gradient is refreshed often, the induced network churn is small, and a smaller whitelist size may trigger the best results.

Acknowledgments

The authors thank Jorge Ortiz and Prof. David E. Culler from the University of California at Berkeley for making the connectivity traces available.

6. REFERENCES

[1] K. Srinivasan, M. A. Kazandjieva, S. Agarwal, and P. Levis, "The beta-factor: Measuring Wireless Link Burstiness," in *6th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Raleigh, NC, USA, 5-7 November, 2008 2008.

[2] *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements. Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, IEEE Std., Rev. 2006, 8 September 2006.

[3] CC2500, *CC2500, Low-Cost Low-Power 2.4 GHz RF Transceiver (Rev. B)*, Texas Instruments, Inc., 13 September 2007, data Sheet SWRS040B [available online].

[4] L. van Hoesel and P. Havinga, "A lightweight medium access protocol (LMAC) for wireless sensor networks: Reducing Preamble Transmissions and Transceiver State Switches," in *International Conference on Networked Sensing Systems*, 2004.

[5] O. D. Incel, S. Dulman, and P. Jansen, "Multi-channel Support for Dense Wireless Sensor Networking," in *First European Conference on Smart Sensing and Context (EuroSSC) Lecture Notes in Computer Science 4272. Springer.*, Enschede, the Netherlands, 25-27 October 2006, pp. 1–14.

[6] Y. Kim, H. Shin, and H. Cha, "Y-MAC: An Energy-efficient Multi-channel MAC Protocol for Dense Wireless Sensor Networks," in *International Conference on Information Processing in Sensor Networks (IPSN)*. St. Louis, Missouri, USA: IEEE, April 22-24 2008, pp. 53–63.

[7] K. Pister and L. Doherty, "TSMP: Time Synchronized Mesh Protocol," in *Parallel and Distributed Computing and Systems (PDCS)*, Orlando, Florida, USA, 16-18 November 2008.

[8] L. Doherty, W. Lindsay, and J. Simon, "Channel-Specific Wireless Sensor Network Path Data," in *16th International Conference on Computer Communications and Networks (ICCCN)*. Turtle Bay Resort, Honolulu, Hawaii, USA: IEEE, August 13-16 2007, pp. 89–94.

[9] *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements. Part 15.1: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs)*, IEEE Std., Rev. 2005, 14 June 2005.

[10] *HART Field Communication Protocol Specifications, Revision 7.1, DDL Specifications*, HART Communication Foundation Std., 2008.

[11] ISA, *ISA100.11a:2009 Wireless systems for industrial automation: Process control and related applications*, International Society of Automation Std., 2009, [Draft standard, in preparation].

[12] C. S. Kang, K. H. Chang, R. Enns, K. Pister, L. Winkel, C. Powell, and J. Gutierrez, "MAC

Enhancements for Time Slotted Channel Hopping," 14 May 2009, iEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) [work in progress].

[13] T. Watteyne, D. Zats, and K. Pister, "TSCH: Time Synchronized Channel Hopping," under review.

[14] J. Ortiz and D. Culler, "Exploring Diversity: Evaluating the Cost of Frequency Diversity in Communication and Routing," in *Conference on Embedded Networked Sensor Systems (SenSys)*. Raleigh, NC, USA: ACM, 5-7 November 2008.

[15] J. Faruque, K. Psounis, and A. Helmy, "Analysis of Gradient-based Routing Protocols in Sensor Networks," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*. Marina del Rey, CA, USA: IEEE/ACM, June 2005, pp. 258–275.

[16] K.-H. Han, Y.-B. Ko, and J.-H. Kim, "A Novel Gradient Approach for Efficient Data Dissemination in Wireless Sensor Networks," in *IEEE 2004 International Conference on Vehicular Technology Conference (VTC)*, Fall 2004.

[17] O. Powell, A. Jarry, P. Leone, and J. Rolim, "Gradient Based Routing in Wireless Sensor Networks: a Mixed Strategy," arXiv.org automated e-print archives, Report CS-0511083, Tech. Rep., 2005. [Online]. Available: http://arxiv.org/abs/cs/0511083

[18] F. Ye, G. Zhong, S. Lu, and L. Zhang, "GRAdient Broadcast: A Robust Data Delivery Protocol for Large Scale Sensor Networks," *ACM Wireless Networks*, vol. 11, pp. 285–298, 2005.

[19] P. Thubert, T. Watteyne, Z. Shelby, and D. Barthel, "LLN Routing Fundamentals," IETF ROLL, IETF Internet-Draft, 9 April 2009, draft-thubert-roll-fundamentals-01 [work in progress].

[20] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "CTP: Robust and Efficient Collection through Control and Data Plane Integration," Stanford University, Tech. Rep. SING-09-01, 28 April 2009.

[21] D. S. J. De Couto, "High-Throughput Routing for Multi-Hop Wireless Networks," Ph.D. dissertation, Massachusetts Institute of Technology, June 2004.

[22] P. Levis, N. Patel, C. David, and S. Shenker, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks," in *Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, USA, 29-31 March 2004.