Multichannel RPL Variant

Noradila Nordin
University College London
Email: noradila.nordin.12@ucl.ac.uk

Richard G Clegg
Imperial College London
Email: richard@richardclegg.org

Miguel Rio
University College London
Email: miguel.rio@ucl.ac.uk

***Abstract*—This paper proposes a new multi-channel tree building protocol for ad-hoc sensor networks. Low power radios such as IEEE 802.15.4 are a relatively short range transmission standard radio technology in the 2.4Ghz band. Unfortunately, the frequency band is shared with WiFi and Bluetooth which cause problem for Wireless Sensor Networks that require minimal packet loss, interference and delay. Our protocol alleviates the effect of interference which results in improved network efficiency and stability, link reliability and minimised latency. Our proposal takes into account all available channels to utilise the spectrum. It checks the condition of all the channels before deciding on a channel to switch into. The successful transmission rate of the channels are stored externally from the sensors which can be accessed when require. This information is used to limit the channels to be considered when channel switching is invoked. The channel that is selected is checked for any changes in its condition that might had taken place after it was checked previously before committing to the channel. The results and decisions are informed to the other nodes to update their neighbour table. We use random channel selection and two hops neighbour strategy to avoid collision. By basing our protocol in routing protocol for low power and lossy networks (RPL), packets can be sent to the destination the same way as a single channel RPL but with less loss. RPL is a gradient based routing protocol forming any-to-any routing for low power IPv6 networks rooted at a single destination called the Low Power and lossy network Border Router (LPBR) with no cycles. The topology is formed by choosing the minimum rank which is the distance from the node to the root based on the minimum expected transmission count (ETX) metric. All nodes are battery operated except for the LPBR. This enables a centralised channel switching processes at the LPBR. The channel switching processes take place after the topology is formed to further improve the transmission rate on the best paths. We implement and evaluate our solution using the Contiki framework. Our experimental results demonstrate an increased resilience to interference, significant higher throughput making better use of the total available spectrum and link stability.**

I. Introduction

Low power radios such as IEEE 802.15.4 is a relatively short range transmission standard radio technology in the 2.4Ghz band. Unfortunately, the frequency band is shared with WiFi and Bluetooth which cause problem for WSN that require minimal packet loss, interference and delay. The problem arise as all 16 channels overlap with channels used by WiFi. However, the overlapping problem can be solved if WiFi only uses the European non-overlapping channel set which are channels 1, 7 and 13, leaving channels 15, 16, 21 and 22 to be used by 802.15.4 [1]. Another solution is to use multichannel.

Multichannel communication in wireless networks can alleviate the effects of interference which as a result, improve the network efficiency and stability, link reliability and minimise latency. It also enables communication between nodes to occur simultaneously without the risk of collision. However, not all channels are free from interference, thus, the need to hop to another channel when the quality of the channel deteriorates. There are two types of channel hopping [23], blind channel hopping and whitelisting. In blind channel hopping, the node will hops over all available channel. Whitelisting on the other hand, filters out the worst channel. Many studies make use of channel whitelisting such as [23] claimed that channel 11, 15, 25 and 26 are free from Wi-Fi, [25] channel 11, 19 and 25, Chrysso [12] uses channel 11, 14, 20, 22 and 26, and MiCMAC [2] uses channel 15, 20, 25 and 26. From these studies, it can be concluded that it is impossible to determine a single channel that is not affected by interference. Our proposed work takes into account all available channels to utilise the spectrum and checks the condition of the channels before hopping as interference varies over time.

There are many studies that were done in multichannel where most of them concentrated on using multichannel MAC layer. Despite there are many multichannel MAC layer that are available in the literature, multichannel is still not widely implemented even though it has many potential benefits for wireless networks. This might be due to the complexity of the solutions to be implemented.

In this paper, we propose a multichannel RPL variant. As most multichannel is implemented in MAC layer, our work concentrates on the application layer. The nodes are given different channels by the Low Power and lossy network Border Router (LPBR) after the topology is formed to avoid collision in a single channel. LPBR has the knowledge of the whole topology which enables it to assign channel to the nodes. As a result, synchronisation is not require. The nodes communicate on the transmission channel and are always listening on their listening channel. The control messages are sent to the nodes on their listening channel as unicast which eliminate the need for a separate control channel. In Contiki, a fast turnaround time is supported where the channel switching delay of 128μs is negligible.

The rest of the paper is organised as follows: Section II presents related work to multichannel protocols. Section III describes the key idea of our proposed protocol and the high-level design and the implementation of the protocol in Contiki. We describe and evaluate the experimental results in Section IV. Finally, we conclude in Section V.

II. Related Work

Radio duty cycling mechanism can be classified into two categories; synchronous and asynchronous. Multichannel synchronous protocols for such as MC-LMAC [11] which uses time slot to transmit on a particular channel and Y-MAC

[13], EM-MAC [17] and TSCH that depend on the neighbouring nodes to synchronise with each other. Multichannel asynchronous protocols such as MuChMAC [5], Chrysso [12], MiCMAC [2] and our protocol are independent of time slot and synchronisation.

//!!why asynchronous vs synchronous important? This is important as in synchronous multichannel, the nodes need to be globally synchronise which is hard to achieve with frequent channel changes as the nodes need to maintain a tight synchronisation and need to synchronise the network clock periodically not to drift in time. Need to have a time-scheduled communication. Asynchronous multichannel on the other hand, can be either sender or receiver initiated communications. This shifts the global synchronise to local and does not affect the whole network. -simple setup, self-configurable as it does not need tight synchronisation.

//distributed vs centralised Most of the (available?) multichannel protocols are fully distributed (self configuration?) and does not has a centralised system. This is the way it should be as sensors have limited battery life and memory but centralised would enable real time decisions more efficient - moving decisions making to a central point and reduce unnecessary processing on the nodes.

Chrysso [12] is a multichannel protocol for data collection applications. The nodes are organised into parent-children groups where each parent-children uses two channel for transmitting and receiving packets. Both parent and children nodes can hop to another channel when interference is detected based on the channel switching policies. If a node loses connectivity, Chrysso calls the scan mode to enable neighbour discovery over multiple channel. Chryssos functionality comprises a set of channel switching policies that interface to both the MAC layer and the network layer (Collect) //?Chrysso concentrates on data collection while our work tries to improve RPL single channel into multichannel without dealing with the MAC layer. RPL is typically used in ContikiMAC which is a single channel.

ContikiMAC [7] is the default radio duty cycling protocol in Contiki that is responsible for the node wake-ups period. ContikiMAC is a power-saving radio duty cycling protocol. It was proved to be efficient in a single channel [2][8]. ContikiMAC uses periodical wake-ups to listen to the neighbours transmission packet. It has a phase-lock mechanism to learn the neighbours wake-up phase to enable efficient transmissions and a fast sleep optimisation in case of spurious radio interference is detected.

The sender use the knowledge of the wake-up phase of the receiver to optimise its transmission. When a packet is successfully received, the receiver sends a link layer acknowledgement. The sender repeatedly sends its packet until it receives a link layer acknowledgement from the receiver.

ContikiMAC relies on retransmissions for reliable transmissions. A Carrier Sense Multiple Access, CSMA is a MAC protocol that performs retransmissions when the underlying MAC layer has problems with collisions. When the sender does not receive the link layer acknowledgement, CSMA will retransmit the packets three times before dropping it from the buffer queue.

//loss and retransmission - how it deals with it

//be clear about when protocol suffers a loss

MiCMAC [2] is an asynchronous protocol, ContikiMAC [7] channel hopping variant. On every wakeup cycle, the channel is periodically switched according to a pseudo-random sequence. MiCMAC introduces channel lock for the channel reception at the sender. There is a dedicated broadcast channel for a duration at every wake up period. MiCMAC can be used with RPL without any changes to RPL.

In order to maximise the good use of (???) multichannel, routing topology play a big role - for scalability, to save energy (efficient) while sending on an optimised routes, maximise the chance of packets being received (minimise loss).

RPL do topology maintenance*.

There are many studies that were done on routing protocol such as tree based LEACH [3], PEGASIS [15], CTP [10] and RPL which is designed largely based on CTP.

Dynamic/static routing topology?

However, only recent multichannel protocols such as MiCMAC uses RPL as the routing protocol. Chrysso uses Contiki collect which is a CTP-like data collection protocol in Contiki. We choose to use RPL as it is the standard for IPv6 routing in low power and lossy networks. RPL forms the network topology dynamically.

Routing protocol for low power and lossy networks (RPL) is a gradient based routing protocol forming any-to-any routing for low power IPv6 networks. RPL topology is a Destination-Oriented Directed Acyclic Graph (DODAG), rooted at LPBR with no cycles. The root has the overall view of the network. The other nodes however, only has knowledge of its neighbours and default router. RPL is a rooted topology which any-to-any traffic is directed towards the root unless the common ancestor is found which the traffic is then routed downwards towards the destination. This strategy is used in order to scale large networks by reducing the routing overhead at the cost of increased hop count through common ancestor.

In RPL terminology, the node distance to the root and other nodes is defined as the node's rank. RPL finds the path with the minimum number of transmissions that a node expect to successfully deliver a packet to the destination and switches only if it is less than the current rank to prevent frequent changes [9].

Our proposed protocol takes into account RPL topology formation scheme and the control messages exchange between nodes that take place frequently to maintain the quality of the tree. The RPL control messages are sent through unicast in order to reduce unnecessary transmitting in broadcast. Our work makes use of RPL topology formation and improves on the channel within the topology formed. The nodes do not need to sync with one another as they would know the listening channel of the other nodes.

//why ours is better?? compare with MiCMAC, Chrysso, MuChMac. "We are different because because we are cross layer with a centralised co-ordinator. RPL is typically used with ContikiMAC, a single channel protocol"

We don't blacklist any channels; checks before using the channel. Centralised channel change decision; at LPBR. Nodes decide if the channel recheck LPBR decision of the channel and change. LPBR has the intelligence in choosing the channel for the node as it has the full overview of the condition of all nodes based on a periodic information from the nodes. LPBR will have the information of the good and bad channel for each node. Cross layer; channel changes on network layer while others use layer 2 (?), the MAC layer. This enables us to do cleverly? and deciding the channel cross layer - can do complicated decisions than if with MAC layer channel.

III. Multichannel RPL Protocol

Multichannel RPL concentrates on finding the best channels for nodes to listen and transmit on, given policies that needs to be complied.

A. Overview

The design of Multichannel RPL are based on several crucial observations:

- Channel assignment - Sensors have limited memory and battery capabilities. In order to maximise the sensors lifetime, a centralised LPBR that has larger memory and fully powered is opted. LPBR has a complete knowledge of the topology which enables it to make good channel assignment decisions based on the criteria that are explained in the next part.
- Interference - External interference cannot be predicted, thus channels cannot be allocated beforehand as it varies over time and locations. It is impossible to determine a single channel that is free from interference at any location. Our protocol checks the channel condition each time before deciding on a channel change to reduce interference and maximise throughput.
- Frequency diversity - RPL is typically used with ContikiMAC which is a single channel protocol. By using multichannel, we increase the robustness of the network towards interference. However, applying multichannel to the existing RPL may hinder neighbour detection and RPL processes to maintain the network topology as it does not switch to the correct channel. We overcome this problem by enabling unicast in neighbour detection and RPL control messages. We assume that no new nodes should join the topology after the initial setup.

Multichannel RPL focuses on the network and application layer of the protocol. This allows channel assignment decisions to be made thoroughly without being limited by the low layer complexity. The channel assignment processes take place once the topology tree has been formed by LPBR and stabilise.

B. Channel Selection Strategy

LPBR uses a two hops neighbour strategy to select a channel to be assigned to a node. This allows fair load balancing on the channels and reduces congestion on the channel that could occur when several nearby nodes start transmitting at similar time. A sensor node integrated an onboard antenna that covers a transmission range of 50 metres indoor and 125 metres outdoors between the sensors. Two hops from the node would cover some ranges between the nodes to reduce transmission interference to a minimum. This allows transmission to happen at the same time without the risk of packet loss due to collision.

However, at initialisation, all nodes are initialised to channel 26 by default. The nodes need to be on the same channel for neighbour discovery and enable RPL to exchange control messages that are required to form an optimised topology before channel assignments can take place. The nodes will only be on the same channel once during the initial setup. Channel 26 is chosen as the initial channel as it usually does not overlap with WiFi and is relatively in a cleaner frequency than the other channels. The studies in [12][2][23] use several channels in their experiments and have channel 26 in common.

In two hops neighbour strategy, LPBR chooses a random channel from channel 11 to 26 for a node unless LPBR has full knowledge of the channels condition. If LPBR has knowledge of all the channels, LPBR can limit the channels range to consider only channels that the node has used in the past that gives good transmission results. LPBR keeps the information of each node neighbours and channels in a table. Before LPBR sends a channel change message to the node, it first checks if the node is LPBR neighbours. If it is, LPBR checks if the new channel value is the same as LPBR channel. If it is not, LPBR checks if the node is a neighbour of other nodes. The new channel is compared with the current channel. If the channels are the same, the first hop of the two hop strategy has failed. LPBR will try going through the same steps with another new channel.

If the first hop succeeds, LPBR will check the nodes that are two hops away from the node the new channel is for. If the node is LPBR neighbour, LPBR checks the new channel value with all LPBR neighbours channels. Otherwise, the node neighbours neighbours channels are checked. If the channel values are not the same, LPBR will send the new channel value to the node to proceed with channel quality checking. If the two hops failed, LPBR retries with a new channel value. The steps from the first hop is repeated. LPBR tries to find the channel that is two hops free within four tries. If a two hops free channel is not found, the default channel 26 is used.

C. Channel Quality Checking

The channel quality checking is invoke each time the node receives the channel change message from LPBR. The node informs all the route neighbours in turn, of the new channel it will be listening on. The route neighbours will update their neighbour table to hold the new channel value as the node current channel for transmission. The node sends a *STARTPROBE* message to the route neighbour and switches to the listening channel ready to receive any incoming messages. The node's route neighbour starts sending the probe messages to the node and the node collects the information on the success or failure rate of the channel. These information is then, sent to LPBR to updates LPBR's knowledge on the channel condition for the specific node and its neighbours. The node uses the results from probing to decide if the new channel is better than the current channel. All the neighbours and LPBR are informed of the decision and update the node's channel.

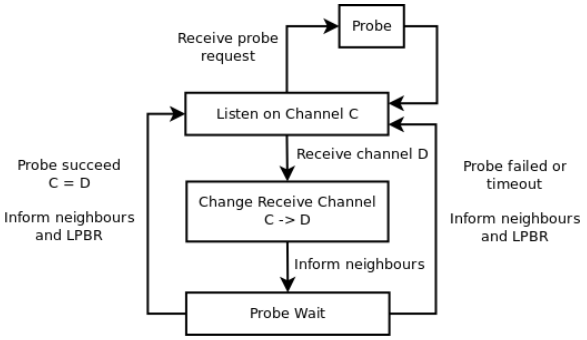

Fig. 1. Channel switching process

A channel is not chosen as a good channel to change into either if it timed out or the probing messages received are below a threshold. The node will revert to the previous channel as it is better than the new channel selected. The interference channel might be used by some nodes if during the probing, all probing messages are received. The channel change will be invoked again in this case.

Probing is an important part in making the channel changes decision. It gives a quick overview of the channel condition based on the number of probing messages received. It is worth noting that probing is only done between the node and the route neighbours. Neighbours that are not route neighbours will not use the node as a route during their transmission thus there is no need for probing to take place for those neighbours. However, the neighbours still need to know the channel value as RPL control messages are sent to neighbours directly without using the routes.

D. Channel Switching

Figure 1 shows the states in channel switching. As explained in the previous sections, LPBR chooses a channel based on the two hops strategy and sends the change channel message to the node. The node saves it's current and new channel separately to allow the channel to be restored if require. The node contacts its route neighbours and start the channel quality checking. The channel that is decided on is informed to the neighbours and LPBR and waits for acknowledgement. If the acknowledgement does not arrive after the timeout, the change channel confirmation message is retransmitted. The neighbour node is ignored if the retransmissions has timed out.

IV. Evaluation

The results of our multichannel RPL protocol is compared against a single-channel tree protocol.

A. Experimental Setup

We evaluate the protocol in Cooja simulated environment with emulation of TMote sky nodes that feature the CC2420 transceiver, a 802.15.4 radio. The nodes run on IPv6, using UDP with standard RPL and 6LoWPAN protocols. The network consists of 16 nodes are used to run the simulation where we have 1 border router node, 1 interference node, and 14 duty cycled nodes that act as UDP clients to send packets to LPBR. RPL border router is used as LPBR in order to move most processing decisions on a PC as it has more RAM and better processing capabilities than a sensor. TelosB has limited RAM and ROM of 10K bytes and 48K bytes of flash memory. By using a border router, this allows channel changing to be decided in real time without draining the memory and battery on a sensor.The border router also acts as the root of the tree.

We simulated a controlled interference node that generates semi-periodic bursty interference to resemble a simplified WiFi or Bluetooth transmitter on channel 22. The interference model that we use is described in [4]. The interference has two states, clear state and interference state. In clear state where it does not do anything, the process stays in the state for a time that is uniformly distributed between $9/16$ seconds and $15/16$ seconds. In interference state, the interference node generates packets for a time that is uniformly distributed between $3/4{*}clearTime$ and $5/4{*}clearTime$ where $clearTime$ refers to the rate of interference. We use a single channel interference in our simulation to show our hypothesis that multiple channels can help avoid interference where we consider a scenario where an RPL system is subject to interference on its channel after set up has successfully completed.

We evaluate multichannel RPL variant using end to end packet delivery performance metric. The transmission success rate is calculated from the sender to the receiver over multiple hops. We also look at the loss over time to observe the protocols react to interference and set up overhead.

We run the simulation for a duration of 60-70 minutes to send 700 packets; 50 packets for each node. When the nodes are switched on for the first time, all nodes are initialised to channel 26 by default. RPL runs the initial network set up for a few minutes before it is stable. We set the RPL set up time to be 5 minutes. RPL topology is formed in a minute. We wait for another 4 minutes to allow trickle timer to doubles the interval length so that RPL control messages are less frequently invoke. We then let our multichannel protocol runs for 10 minutes. In our 15 nodes simulation, our protocol takes 7-8 minutes to run the channel change set up. We allow another 2 minutes wait time if channel changes retransmission happen. In a single channel simulation, all the nodes are changed to channel 22 after 5 minutes of RPL set up time. This allows RPL to have enough time to discover all nodes to form an optimised topology. The topology formation does not formed completely if the interference node interferes from the beginning. The interference node starts sending packets to interfere after 3 minutes the system is switched on so that the interference channel is involve in the channel changes decision. We proved that our protocol tries to avoid from changing to the interference channel through time out and probing failures. After 15 minutes, the client nodes will send a normal packet periodically every 30-60 seconds to LPBR. This is done in order to avoid collision of the nodes sending at the same time. The simulation is repeated 10 times.

B. Effect of Multi-channel

We vary the interference rate, which is referred as *clear_time* in [4] to 0% (no interference), 25% (extreme), 50% (moderate) and 75% (mild) where the percentage is the ratio of the time the channel is cleared from interference. The test is done to evaluate our protocol behaviour in different interference rate and to compare the result with a single channel

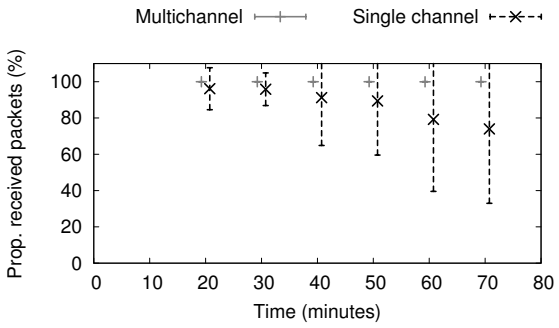

Fig. 2. Channel switching process

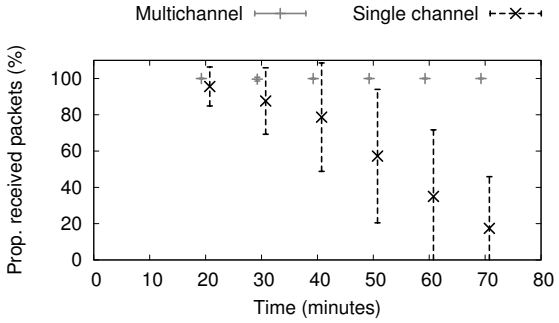

Fig. 3. Channel switching process

case. Figure (xxxxx) shows the averaged results from three runs that we did. We observed that during high interference and moderate interference, when the LPBR generate a random two hop channel for a node to change into, the receiving node will probe on the channel. It will either timed out or the probing messages received are less than a threshold that allows for the node to change it's listening channel to the new channel. This is as expected as our protocol checks the channel each time before deciding on the new channel to avoid interference channel. By doing this, we can be sure that the node listening channel is a good channel. This enable us to use all available channels without blacklisting any channel until we are sure it is a bad channel through our probing process. The channel quality table is built at the LPBR that over time, can be used to learn good and bad channels based on several probing processes.

In the mild interference case, all probing messages are received even though there are interference in that channel.

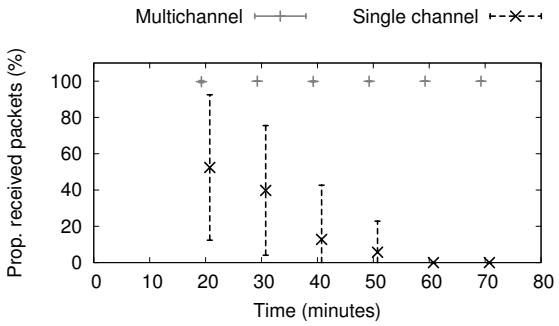

Fig. 4. Channel switching process

This is because, the probing gives good result which means that the channel can be used. As the interference rate is mild, all packets are received. This is also the case with a single channel. The interference does not affect the transmissions as the interference is not frequent enough that enables the node to recover. However, the interference would slightly effect the packet transmission over time. We plan to run channel change processes periodically to avoid this from happening.

In the single channel, the node does not have enough time to recover from the interference and drops all packets. Figure (xxxx)shows that there are more packets drop over time and it stops receiving packets as it doesn't have enough buffer to store the incoming packet and the channel becomes congested. However, as the interference rate increases (less interference), the single channel performance improves as it has more time to recover.

C. Setup Overhead

Our protocol introduces extra set up overhead to the system. Default RPL on ContikiMAC has an overhead of average 281 packets while our protocol has an overhead of 844 including RPL overhead. Our protocol introduces 563 overhead. However, the overhead is a one-off which the channel changes might be use for hours or days which is not a lot in that comparison. (negligible?) RPL forms the network while our protocol improves the network at a small cost in terms of messages while leaving the network functional. Our protocol can transmit and receive messages during the set up phase - which means, normal transmission can run as usual while the set up is taking place at the same time.

V. Conclusion

We introduced Multichannel RPL, a channel changes protocol as an extension to the existing RPL with ContikiMAC at the network and application layers. The nature of our protocol that decides/run the decision making to the application layer gives us more opportunity to add the decision complexity - intelligence? and do real time channel checking. Our protocol maintains high reliability during heavily interfered periods where ContikiMAC showed a low throughput of //////result?.

We are continuing with the work to further develop this protocol. The next stages that we plan to pursue is to improve the interference model that we used in testing to cover multiple channel interference model. Why? Testing close to the real world where interference happen at many channels. We also plan to test our implementation to real hardware and to allow continual updates on packet loss for each node so that channels can be changed dynamically when interference occurs.

Acknowledgment

References

[1] IEEE standard for information technologytelecommunications and information exchange between systems local and metropolitan area networksspecific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pages 1–2793, March 2012.

[2] B. Al Nahas, S. Duquennoy, V. Iyer, and T. Voigt. Low-power listening goes multi-channel. In *2014 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 2–9, May 2014.

[3] Asaduzzaman and Hyung Yun Kong. Energy efficient cooperative leach protocol for wireless sensor networks. volume 12, pages 358–365, Aug 2010.

[4] Carlo Alberto Boano, Thiemo Voigt, Nicolas Tsiftes, Luca Mottola, Kay Römer, and Marco Antonio Zúñiga. Making sensornet mac protocols robust against interference. In *Proceedings of the 7th European Conference on Wireless Sensor Networks*, EWSN'10, pages 272–288, Berlin, Heidelberg, 2010. Springer-Verlag.

[5] Joris Borms, Kris Steenhaut, and Bart Lemmens. Low-overhead dynamic multi-channel mac for wireless sensor networks. In *Proceedings of the 7th European Conference on Wireless Sensor Networks*, EWSN'10, pages 81–96, Berlin, Heidelberg, 2010. Springer-Verlag.

[6] Thang Vu Chien, Hung Nguyen Chan, and Thanh Nguyen Huu. A comparative study on operating system for wireless sensor networks. In *2011 International Conference on Advanced Computer Science and Information System (ICACSIS)*, pages 73–78, December 2011.

[7] Adam Dunkels. The ContikiMAC radio duty cycling protocol, 2011.

[8] Simon Duquennoy, Olaf Landsiedel, and Thiemo Voigt. Let the tree bloom: Scalable opportunistic routing with ORPL. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, page 2:12:14, New York, NY, USA, 2013. ACM.

[9] Omprakash Gnawali. The minimum rank with hysteresis objective function, rfc6719. 2012.

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

[11] Ozlem Durmaz Incel, Lodewijk van Hoesel, Pierre Jansen, and Paul Havinga. Mc-lmac: A multi-channel mac protocol for wireless sensor networks. *Ad Hoc Netw.*, 9(1):73–94, January 2011.

[12] V. Iyer, M. Woehrle, and K. Langendoen. Chrysso - a multi-channel approach to mitigate external interference. In *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 449–457, June 2011.

[13] Youngmin Kim, Hyojeong Shin, and Hojung Cha. Y-mac: An energy-efficient multi-channel mac protocol for dense wireless sensor networks. In *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, pages 53–63, April 2008.

[14] Philip Levis, T Clausen, Jonathan Hui, Omprakash Gnawali, and J Ko. Rfc6206: The trickle algorithm. *Internet Engineering Task Force (IETF) Request For Comments, http://ietf. org/rfc/rfc6206. txt*, 2011.

[15] S. Lindsey and C.S. Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. In *Aerospace Conference Proceedings, 2002. IEEE*, volume 3, pages 3–1125–3–1130 vol.3, 2002.

[16] Lanny Sitanayah, Cormac J. Sreenan, and Szymon Fedor. A cooja-based tool for maintaining sensor network coverage requirements in a building. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, page 70:170:2, New York, NY, USA, 2013. ACM.

[17] A. Sivanantha, B. Hamdaoui, M. Guizani, Xiuzhen Cheng, and T. Znati. Em-mac: An energy-aware multi-channel mac protocol for multi-hop wireless networks. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2012 8th International*, pages 1159–1164, Aug 2012.

[18] Luigi Alfredo Grieco Thomas Watteyne, Maria Rita Palattella. Using ieee802.15.4e tsch in an lln context: Overview, problem statement and goals. 2014.

[19] Pascal Thubert. Objective function zero for the routing protocol for low-power and lossy networks (rpl), rfc6552. 2012.

[20] Nicolas Tsiftes, Joakim Eriksson, Niclas Finne, Fredrik Osterlind, Joel Hglund, and Adam Dunkels. A framework for low-power IPv6 routing simulation, experimentation, and evaluation. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, page 479480, New York, NY, USA, 2010. ACM.

[21] Tsvetko Tsvetkov. Rpl: Ipv6 routing protocol for low power and lossy networks. *Sensor Nodes–Operation, Network and Application (SN)*, 59:2, 2011.

[22] J Vasseur, M Kim, K Pister, N Dejean, and D Barthel. Routing metrics used for path calculation in low power and lossy networks. *draft-ietf-roll-routing-metrics-19 (work in progress)*, 2011.

[23] Thomas Watteyne, Ankur Mehta, and Kris Pister. Reliability through frequency diversity: Why channel hopping makes sense. In *Proceedings of the 6th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, PE-WASUN '09, page 116123, New York, NY, USA, 2009. ACM.

[24] T Winter, P Thubert, T Clausen, J Hui, R Kelsey, P Levis, K Pister, R Struik, and J Vasseur. Rpl: Ipv6 routing protocol for low power and lossy networks, rfc 6550. *IETF ROLL WG, Tech. Rep*, 2012.

[25] Yafeng Wu, J.A. Stankovic, Tian He, and Shan Lin. Realistic and efficient multi-channel communications in wireless sensor networks. In *IEEE INFOCOM 2008. The 27th Conference on Computer Communications*, pages –, April 2008.