PC-RPL: Joint Control of Routing Topology and Transmission Power in Real Low-Power and Lossy Networks

14

HYUNG-SIN KIM, University of California Berkeley
JEONGYEUP PAEK, Chung-Ang University
DAVID E. CULLER, University of California Berkeley
SAEWOONG BAHK, Seoul National University

We present *PC-RPL*, a transmission power-controlled IPv6 routing protocol for low-power and lossy wireless networks that significantly improves the end-to-end packet delivery performance under heavy traffic compared to the standard RPL. We show through actual design, implementation, and experiments that a multihop wireless network can achieve better throughput and routing stability when transmission power and routing topology are "jointly and adaptively" controlled. Our experiments show that the predominant "fixed and uniform" transmission power strategy with "link quality and hop distance"–based routing topology construction (i.e., RPL) loses significant bandwidth due to hidden terminal and load imbalance problems. We design an adaptive and distributed control mechanism for transmission power and routing topology, named *PC-RPL*, on top of the standard RPL routing protocol for hidden terminal mitigation and load balancing. We implement *PC-RPL* on real embedded devices and evaluate its performance on a 49-node multihop testbed. *PC-RPL* reduces total end-to-end packet losses by approximately sevenfold without increasing hop distance compared to RPL with the highest transmission power, resulting in 17% improvement in aggregate bandwidth and 64% improvement for the worst-case node by successfully alleviating both hidden terminal and load imbalance problems.

CCS Concepts: • **Networks** → *Network protocol design*; Network design principles;

Additional Key Words and Phrases: Bandwidth, multihop, routing topology, transmit power, low-power lossy network (LLN), RPL, Internet of Things

An earlier version of this article appeared in the proceedings of the 13th IEEE International Conference on Distributed Computing in Sensor Systems (DCoSS'17), June 2017 [36]. However, this article has been significantly extended from that conference version.
This work was supported partly by the Department of Energy (Grant No. DE-EE0007685); California Energy Commission; Intel Corporation; partly by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT, and Future Planning (No. 2017R1E1A1A01074358); and partly by the Basic Science Research Program through the NRF funded by the Ministry of Education (NRF-2017R1D1A1B03031348).
Authors' addresses: H.-S. Kim, Department of Data Science, Seoul National University, Seoul, South Korea; email: hyungkim@snu.ac.kr; D. E. Culler, Computer Science Division, University of California Berkeley, Berkeley, CA 94720; email: culler@cs.berkeley.edu; J. Paek (corresponding author), School of Computer Science and Engineering, Chung-Ang University, Seoul, 06974, Republic of Korea; email: jpaek@cau.ac.kr; S. Bahk (corresponding author), Department of Electrical and Computer Engineering and INMC, Seoul National University, Seoul, Republic of Korea; email: sbahk@snu.ac.kr.

1550-4859/2020/03-ART14 $15.00
https://doi.org/10.1145/3372026

ACM Reference format:
Hyung-Sin Kim, Jeongyeup Paek, David E. Culler, and Saewoong Bahk. 2020. PC-RPL: Joint Control of Routing Topology and Transmission Power in Real Low-Power and Lossy Networks. *ACM Trans. Sen. Netw.* 16, 2, Article 14 (March 2020), 32 pages.
https://doi.org/10.1145/3372026

1 INTRODUCTION

Low-power and lossy wireless networks (LLNs), comprised of a large number of embedded networking devices, is an emerging technology that can be used in a variety of applications including smart grid automated metering infrastructures (AMIs) [1, 6], industrial monitoring [15, 21], and wireless sensor networks [29, 52, 54, 61]. Combined with IPv6 and Internet standard protocols such as RPL (IPv6 Routing Protocol for LLN) [64] and 6LoWPAN [49], LLNs are expected to become a major part of the Internet of Things (IoT) with a number of devices interconnected through multihop mesh networks.

Due to the low power constraint, an LLN usually forms multihop topology in practice, which makes routing an important aspect of the LLN. Given that *lossy* link is a dominant characteristic of the LLN, the subtlest aspect of routing in LLNs is topology formation, the process of determining dynamically the set of pairwise links that are reliable enough to be considered when selecting paths and used in forwarding. In modern LLN routing protocols, such as RPL [64] and collection tree protocol (CTP) [16], the goal of this process is to form a directed acyclic graph (DAG) rooted at one or more border routers, typically connected to local or wide area networks and thereby part of a private or public Internet. Each node discovers potential neighbors through communication events and computes certain statistics to determine whether to include the candidates to its neighbor table [2]. Nodes use link characteristics in combination with path metrics, such as hop count or expected transmission count (ETX) [10], to select a small subset of neighbors that are closer to the roots to serve as parents [16, 64]. The dominant traffic pattern is then generating and forwarding packets through parents toward border routers and beyond (i.e., upward/collection traffic). All aspects of this process, link capacity, neighbor table size, routing table size, and queue size, are severely constrained.

The wireless network topology is primarily determined by a host of external factors, such as physical placement of the nodes, presence of obstacles, media attenuation, and multipath effects. All that a node can control is its transmission power and the logic it uses to compute statistics and apply thresholds in the topology formation process. Many LLN studies have considered varying transmission power to increase energy efficiency by minimizing transmission power, often in theoretical models or simulations [58]. Assuming that an external protocol sets routing topology, these studies aim to maximize nodes' battery lifetime by using transmit power *just enough* for reliable packet delivery over the given wireless link, whereas delivered bandwidth (throughput) has been overlooked assuming that the LLN typically generates light traffic.

Target performance metric (throughput). Throughput is becoming another important performance metric in LLNs, sometimes more important than energy efficiency. For example, in indoor environments with available power outlets, there is no reason to avoid plugging in nodes. In this context, Thread [20] and BLE mesh [19], recently standardized multihop LLN protocols, force LLN routers to be plugged in.

In addition, differently from traditional LLN applications such as environmental monitoring [46, 63], modern IoT applications generate heavy traffic. For example, machine health monitoring for a smart factory requires a vibration sensor attached on a machine to frequently report its data [26]. The heating, ventilation, and air-conditioning (HVAC) system may include anemometry

(with ~1 Hz/node reporting frequency) to diagnose problems in a building [41]. Some applications require image transmission [32, 52]. Even a typical sense-and-send application, where each node generates light traffic, might require nodes near a border router to relay heavy traffic as its deployment scale or node density increases. For example, Cisco's CG-Mesh network [8] for smart grid constructs a large scale multihop LLN with approximately 5,000 nodes. In these cases, an LLN is required to provide high throughput for collection traffic patterns rather than energy efficiency.

Challenges. Unfortunately, improvement of throughput is more complicated than that of energy efficiency. Specifically, using just enough transmission power, the approach for improving energy efficiency, does not improve throughput [23]. Although this approach allows more spatial multiplexing on the same routing topology, this would be expected to bring no value for collection traffic patterns, since every packet that is generated needs to arrive at a border router eventually; collection points are the bottleneck.

Instead, maximizing transmission power tends to maximize link reliability and minimize routing distance [34],[1] which has become a common design choice in LLNs. Furthermore, uniform transmission power setting for all nodes is commonplace in LLNs for simplicity and ease of management, and has been used in almost all published articles that present experimental results with real embedded devices (e.g., [7, 48]). An empirical study verified that under heavy traffic, this simple maximum/uniform transmission power strategy outperforms an adaptive power control mechanism designed for energy efficiency [23]. Therefore, to the best of our knowledge, there have been no power control mechanisms in LLNs that target throughput improvement for multihop collection traffic.

Last, although most transmission power control mechanisms built on theoretical/simulation models (e.g., [47]) contribute to intellectual depth of the research community, they cannot directly be applied in practice [60] since many assumptions for generating these models do not hold in the real world. For example, in terms of channel modeling, wireless link quality is not necessarily determined by physical distance between a node pair [65, 66]. Many proposals assume that each node has important information (e.g., topology, link quality, or distance) timely and accurately, which is actually hard to provide. Thus, a practical mechanism should be proved to run on real embedded devices in real wireless link environments.

Our approach. In this article, we argue that throughput improvement in multihop LLNs cannot be done merely with transmission power control. Instead, we show that substantial gains in throughput and fairness result from dynamic adaptation of transmission power in combination with adaptive routing topology formation, all in a *distributed* manner. We study this problem in the context of RPL on a substantial, multihop testbed comprised of nodes in wide use for more than a decade [57]. We develop a new routing protocol called *PC-RPL* (power-controlled RPL) that uses purely local rules in a simple control loop to adapt routing topology and transmission power in concert to improve throughput and fairness. In doing so, the rules recognize the interrelated effects of link loss, queue loss, and routing distance, all of which are measured locally by each node, and balance them to eliminate loss.

In RPL [64], topology formation is driven by destination-oriented directed acyclic graph (DODAG) information object (DIO) messages, distinct from application data packets, so the two forms of communication can easily utilize distinct transmission power settings. *PC-RPL* uses maximum transmission power for (infrequent) DIOs to gain the most information about the node neighborhood, and it prunes (or expands) the set of candidate parents in a manner consistent with time-varying connectivity by adjusting the threshold used in parent selection. To reduce *hidden*

[1]Note that the maximum transmission power in an LLN is significantly limited (e.g., 0 dBm in the case of IEEE 802.15.4). Therefore, an LLN typically forms a multihop topology even when using maximum transmission power.

terminal effects at its parent, a *PC-RPL* node may elect to prune its parent list and select a parent "closer to it" by increasing the threshold and then adjust data transmission power while maintaining link reliability. At the same time, *PC-RPL* balances these benefits against potentially increased path length and path loss. Further *load balancing* benefits are gained by analogous threshold adjustments vis-a-vis children, to cause some of the children of an overloaded parent to shed it in favor of a more lightly loaded, good alternative. Critical to these adaptive mechanisms is identifying how a node can reasonably infer whether loss it observes is due to hidden terminal or queue overload effects. This is achieved by simple and local rules in *PC-RPL*.

Organization. We develop and evaluate a localized, adaptive topology and transmission power control protocol in stages. After defining our empirical methodology in Section 2, we conduct a preliminary study of forwarding performance and loss characteristics, particularly topologies that bring out the underlying system factors, giving rise to the opportunity to improve throughput and fairness through these adaptive mechanisms. This study further sharpens the empirical methodology by identifying the traffic regime germane to this study. At very low offered load, the functional benefits are present but less pronounced because contention and queue overflow are unlikely. At some offered load, they are present and increase the saturation point, but at very high load, saturation effects dominate.

Building on this, in Section 4 we study uniform transmission power control with conventional topology formation in the full multihop setting. This proves the existence of benefits of power control on collection traffic and permits a sequence of observations that lead to our adaptive, non-uniform strategy. It should be noted that the benefits of *a priori* transmission control, while present on any specific deployment, are not directly actionable because the control point depends on particulars of the deployment. If it is to be determined *in situ* by nodes observing particular events and taking particular actions, it is actually more natural to employ non-uniform adaptation, which is also much more effective. Real deployments are typically non-uniform. In Section 5, we study how to achieve this non-uniform power control in real environments: what we should do and not do. Specifically, we show why a geographical distance–based graph, used in most topology control schemes, may not work in complex real-world wireless environments. Link quality information should be directly measured to be used for transmission power control.

In Section 6, we develop the *PC-RPL* adaptive topology and power control protocol and evaluate its effectiveness in Section 7 on a 49-node testbed. It provides a sevenfold reduction in end-to-end packet loss rate, resulting in 17% improvement in aggregate bandwidth and 64% improvement for the worst-case node. The control algorithm stabilizes quickly with key adjustments occurring in a minority of the nodes. Related work is discussed in Section 8, and Section 9 concludes the article.

Contributions. The contributions of this work are fourfold:

- We experimentally show that adaptive and differentiated power control is needed to improve throughput performance because (1) fixed and uniform transmit power is prone to induce link congestion and load imbalance under relatively high traffic loads, and (2) there is no single transmit power that fits all deployment scenarios when equal transmit power is used for all nodes.
- We design *PC-RPL*, a lightweight distributed and adaptive routing/power control mechanism within RPL, which is the first practical control mechanism aiming for better *throughput* in LLNs, to the best of our knowledge.
- We provide novel local rules for *PC-RPL*, where each node detects its own problems by combining its link loss and queue loss, and adjusts its transmission power and routing parent/child together to mitigate hidden terminal and load imbalance problems.

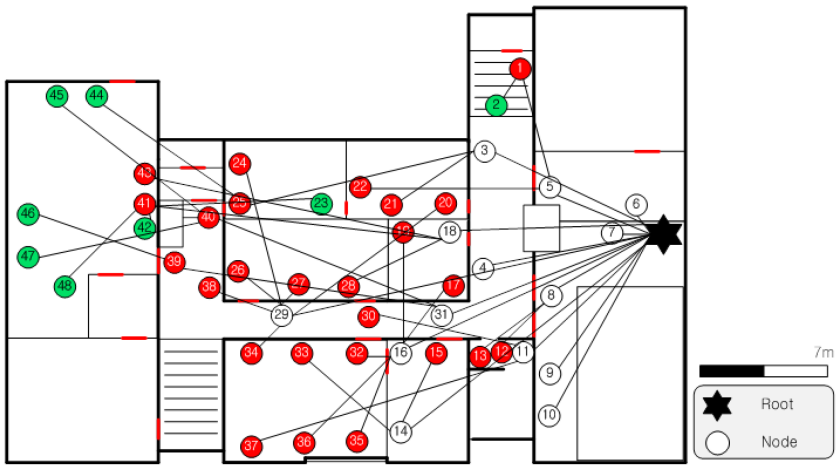


Fig. 1. Testbed topology with a snapshot of routing paths given by RPL when using transmission power of 0 dBm.

- We implement *PC-RPL* on embedded devices and experimentally evaluate its performance against the standard RPL and QU-RPL [35] on a real 49-node testbed. Our evaluation shows that *PC-RPL* achieves significantly better packet delivery performance, routing stability, and transmission overhead than RPL while using lower and heterogeneous transmit power and providing similar routing distance.

2 SYSTEM MODEL AND EXPERIMENTAL SETUP

We consider a wireless system where many embedded devices form a multihop wireless mesh rooted at a border router, connecting the wireless network to a wide area network to form an IP-based internetwork. In this scenario, LLN endpoints utilize IEEE 802.15.4 links to communicate with each other and use RPL to construct routes toward the LLN border router (LBR). On top of that, endpoints use IPv6 to communicate with servers.

To study throughput of the wireless multihop network, we configured a testbed environment as depicted in Figure 1, where 48 nodes and one border router (marked with a star) are in an office environment. Each node is a TelosB clone [57] with an MSP430 microcontroller and a CC2420 radio (IEEE 802.15.4 radio with the maximum transmit power of 0 dBm). The border router is connected to a Linux PC through a serial link. Embedded software is TinyOS 2.1.2 with an IPv6/6LoWPAN [49] stack and an RPL routing protocol implementation, *BLIP* and *TinyRPL*, respectively [38]. The *pp-prouter* stack is used for the border router serial link. Each node employs the TinyOS default CSMA/CA, which avoids packet collision with random backoff and clear channel assessment (CCA). CCA detects an ongoing transmission if its received signal strength indicator (RSSI) is higher than −77 dBm. Each node uses up to five link-level retransmissions and a FIFO transmit queue size of 10 packets. To focus on the impact of transmission power and routing topology, we disabled the use of duty cycling mechanism [50].[2] We also focus on upward traffic from individual nodes to the border router (i.e., data collection). Each result comes from an hour-long testbed experiment during the nighttime.

[2]Furthermore, in many practical LLN deployments [8, 54], LLN routers are wall powered and free from the energy constraint (i.e., no duty cycling).

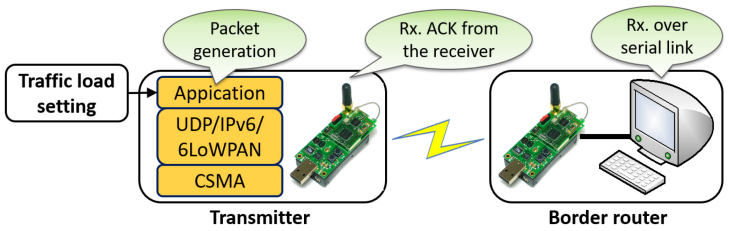


Fig. 2. Experimental setup for the preliminary throughput (delivered bandwidth) study.

3 PRELIMINARY FORWARDING STUDY

Given that throughput is affected by several factors, such as wireless bandwidth, serial bandwidth, and processing delay, we perform a preliminary study before looking into the throughput of multihop networks. This study aims at leading us, throughout the rest of this article, to not only get meaningful insight from experimental results on our 49-node testbed but also solely focus on the impact of *routing topology* and *transmission power* on wireless throughput. For example, we should be able to decouple our experimental results from the effect of processing delay and serial bandwidth. Specifically, we first study throughput performance of the border router with simple smaller topologies. This is because, no matter how we change transmission power and routing topology, all packets eventually pass through wireless and serial links at the border router; the bottleneck is likely to be around the border router.

For a detailed analysis, we divide throughput into four types: *ideal*, *app-gen*, *wireless*, and *serial*, each of which is measured as shown in Figure 2. *Ideal* is the input traffic load that we set, and *app-gen* is the packets generated at the transmitter's application layer. The gap between *ideal* and *app-gen* indicates that a transmitter fails to generate some packets due to processing delay. *Wireless* is the packets successfully delivered through the wireless link and counted by ACK reception at the transmitter. The gap between *app-gen* and *wireless* comes from the packet losses at the transmitter's queue and wireless link. *Serial* is the packets delivered to the Linux PC, and the gap between *wireless* and *serial* means the loss within the border router (serial link or queue).

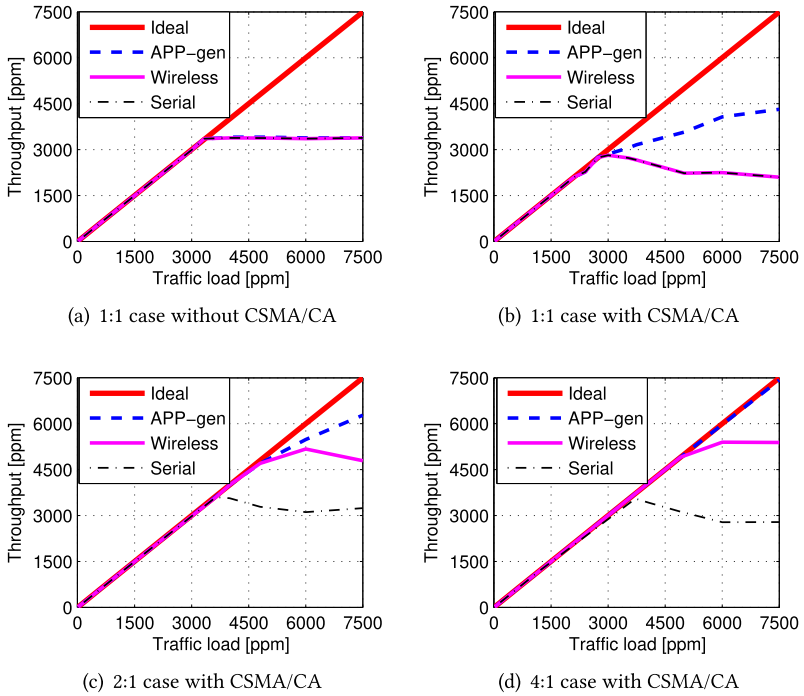
3.1 Throughput of Single-Hop Topologies

Figure 3 depicts throughput at the border router with periodic traffic (packet size of 112 bytes) in various single-hop topologies. Note that wireless link quality is good enough to cause no packet loss.

The simplest network throughput without CSMA/CA in Figure 3(a) shows that the 1:1 case (one transmitter and one receiver case) is bounded by 3,380 packets per minute (pkts/min), approximately 20% of the ideal throughput.[3] Interestingly, the bottleneck is neither wireless link nor serial link but *the transmitter's software stack*. Due to the limited processing capability of the MCU, 18 ms elapses from a packet generation at the application layer to its transmission (including IP, link, and PHY layer stacks). Figure 3(b) shows that with CSMA/CA, throughput drops to 2,815 pkts/min. This is limited by the link layer due to CSMA/CA backoff delay, which exists to improve multiple channel access rather than single stream delivery. When the application generation rate exceeds the link-layer bandwidth, the intervening queue overflows and the excess is loss.

Figure 3(c) and (d) show the throughput when increasing the number of transmitters to two and four, respectively. All transmitters generate the same amount of traffic (i.e., 1/n of total load). The

[3]Given that we are using a CC2420 radio that implements IEEE 802.15.4 with a maximum data rate of 250 kbps, the ideal throughput is 16,740 pkts/min.



(a) 1:1 case without CSMA/CA

(b) 1:1 case with CSMA/CA

(c) 2:1 case with CSMA/CA

(d) 4:1 case with CSMA/CA

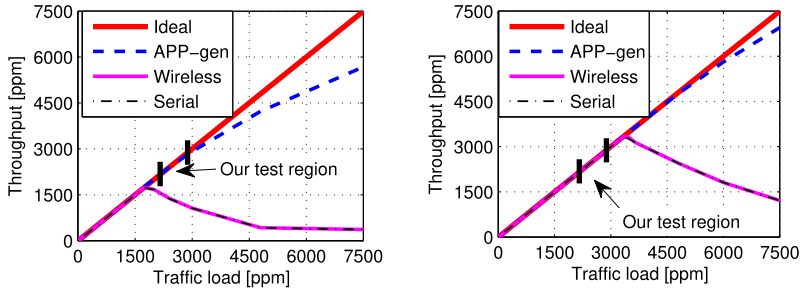
Fig. 3. Throughput at the border router in single-hop networks. The results show that delivering traffic through multiple nodes improves throughput by amortizing processing delay and CSMA/CA backoff delay.

results are quite different from 1:1 cases. Throughput increases to 3,600 pkts/min, and more importantly, the bottleneck moves to the receiver side; serial link capacity.[4] The aggregate bandwidth received on the wireless link exceeds that in the 1:1 case. We generally assume that throughput in a wireless network is maximized when a single node fully utilizes the channel, but here we find receiver bandwidth exceeds transmitter bandwidth, so *multiple* transmitters are required to saturate it. This is because the use of multiple transmitters enables to distribute traffic generation to multiple computing resources, which amortizes the processing delay issue of individual transmitter. In addition, the receive bandwidth at the border router exceeds that of its upstream serial link, resulting in loss at the intervening queue. We see this as a unique characteristic of embedded devices that have much poorer computing capability than LTE and WiFi devices. Last, we note that the bandwidth of wireless link improves significantly when the number of transmitters increases from one to two (5,200 pkts/min) but remains similar when it increases from two to four (5,400 pkts/min).

3.2 Throughput of 2-Hop Topologies

Next, we measure the throughput at the border router in various 2-hop topologies to have a glimpse of multihop characteristics. We set the distance such that each wireless link is good enough to cause no packet loss due to channel error. Figure 4(a) shows the throughput performance of 2-hop line topology (1:1:1), which is only half of that of the 1:1 single-hop topology. This is because the 1-hop node has to relay all packets to the border router but cannot access a wireless channel frequently due to contention with the second hop node.

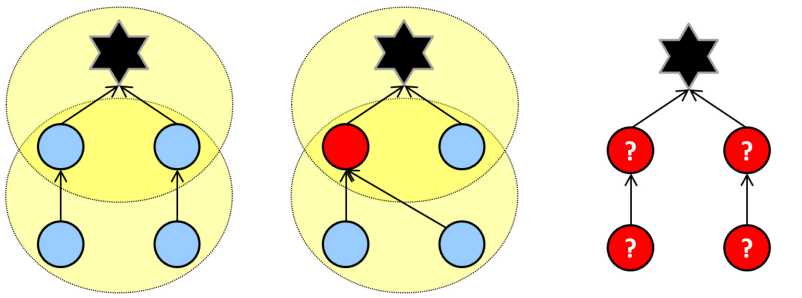
[4]Serial baud rate is 115,200, the default for TelosB motes.



(a) 1:1:1 (line topology) (b) 2:2:1 ideal

(c) 2:2:1 load imbalance (d) 2:2:1 hidden terminal

Fig. 4. Throughput at the border router in 2-hop networks. Delivering traffic through multiple branches improves throughput of multihop networks, but hidden terminal and load imbalance problems degrade throughput.



(a) Ideal case (b) Load imbalance case (c) Hidden terminal case

Fig. 5. Various 2:2:1 topologies for bandwidth study of 2-hop networks. Each link is good enough to incur no link loss, and nodes in a shaded circle are within each other's transmission range.

Figures 4(b) through (d) show the throughput performance when delivering traffic through multiple branches by using the three types of 2:2:1 topologies depicted in Figure 5. In Figure 5, nodes within a shaded ellipse can detect and receive each other's transmissions.

First of all, we configure an ideal case as in Figure 5(a), where traffic load is equally distributed to the two branches and each node detects all CSMA/CA contenders' transmissions with CCA (i.e., no hidden terminal). Compared to Figure 4(a), Figure 4(b) reveals that, as in the single-hop case, the use of multiple branches improves throughput significantly (3,260 pkts/min) in multihop networks.

To take a deeper look at the effect of topology on throughput, the load unbalanced topology in Figure 5(b) and 4(c) shows that this degrades throughput significantly. Note that this topology is also free from hidden terminal since all contenders are within each other's CCA range. Given that the achievable bandwidth using a single branch is strictly limited as shown in Figure 4(a), overloading a single relay node causes queue losses. Combining the results of Figure 4(b) and (c), we can see that delivering traffic through multiple branches can improve throughput; however, to maximize the throughput, gain load should be balanced among the branches. Furthermore, without hidden terminal, traffic congestion incurs queue losses rather than link losses.

Last, in Figure 5(c), we configure a hidden terminal (a representative problem of CSMA/CA) scenario by adjusting the CCA range. Compared to Figure 5(a), the only difference of this scenario is that each node cannot detect contenders' transmissions. By comparing Figure 4(d) to Figure 4(b), we confirm that the hidden node problem degrades throughput. A more interesting observation is that, in contrast to the other cases in which almost all packet losses occur at the queue, the hidden terminal case incurs considerable retransmission failures (link losses) from packet collisions. In other words, when a node experiences link losses, this implies that the node suffers from hidden nodes.

We move on to the throughput study of a more complex "49-node multihop network" while keeping in mind the two findings from this preliminary study:

- Load balancing among multiple branches improves throughput by addressing limited processing delay and queue size of embedded devices.
- A hidden terminal problem degrades throughput by causing packet collisions. Moreover, link losses at a good-quality wireless link imply the presence of hidden terminals.
- A multihop LLN should be able to achieve the throughput range marked as "our test region" in Figure 4 when multiple branches are well balanced and the hidden terminal is well mitigated.

4 LIMITATIONS OF ROUTING WITH UNIFORM TRANSMISSION POWER

In this section, we provide an experimental measurement study of our 49-node multihop testbed network using standard RPL for topology construction while varying transmission power settings uniformly for all nodes. To evaluate this strategy, we generate upward traffic at a rate from 720 pkts/min (15 pkts/min/node) to 2,880 pkts/min (60 pkts/min/node). Note that, as marked in Figure 4 ("our test region"), these traffic loads can be delivered when multiple branches are effectively used. Given that our testbed provides sufficient 1-hop nodes (i.e., branches) that can transmit packets directly to the border router as shown in Figure 1, throughput depends on the network protocol; traffic load used for this study is suitable for investigating the impact of *a network protocol* on throughput.

Although a traditional LLN application usually generates low rate traffic at each node, RPL does need to deliver *heavy traffic* since many modern LLN applications require frequent data reporting [26], image delivery [52], or large scale/density deployment [8, 32]. Therefore, our traffic load setting is applicable to practical LLN scenarios. Observations presented in this section will provide the motivation for designing *PC-RPL*, a distributed power and topology control mechanism that enables adaptive and non-uniform power transmissions among nodes.

4.1 RPL: IPv6 Routing Protocol for an LLN

We first briefly describe RPL's characteristics. RPL is designed for resource-constrained embedded devices to meet the requirements of a wide range of LLN applications [5, 12, 25, 27, 64] while supporting bi-directional IPv6 communication between network devices, leading to the emerging

IoT. RPL is a distance vector type of routing protocol that builds DAGs based on routing metrics and constraints. In most deployment scenarios, RPL constructs a quasi-forest routing topology called *DODAG* rooted at an LBR.

Each node in RPL advertises routing metrics and constraints through DIO messages, and builds a DAG according to its objective function (OF) and routing information in DIO messages. Upon receiving DIO messages from its neighbors, a node chooses a routing parent according to its OF and local policy and then constructs a routing topology (i.e., DODAG). DIO messages are broadcasted using the *TrickleTimer* [42] to achieve a balance between control overhead and fast recovery. RANK is defined and used by the OF to represent the routing distance from a node to the LBR, and link and node metrics (e.g., ETX) are used for RANK calculation and parent selection. RPL allows implementations to customize OF based on application requirement, and various OFs are available for flexibility [14, 17, 18].

Once each node selects a route toward the LBR, RPL uses destination advertisement object (DAO) messages for reverse route construction, which advertise routing information on how other nodes can reach various destinations and prefixes within the network when traveling down the RPL DODAG. Each node generates a new DAO message whenever it changes its routing parent, and periodically when updates are required. How a DAO message is processed by each node and the LBR depends on whether the network is using RPL's "storing mode" or "non-storing mode" for downward routing [39]. The basic idea is for ancestor nodes to process and store the information in DAO messages to create routing entries for the nodes in the subtree.

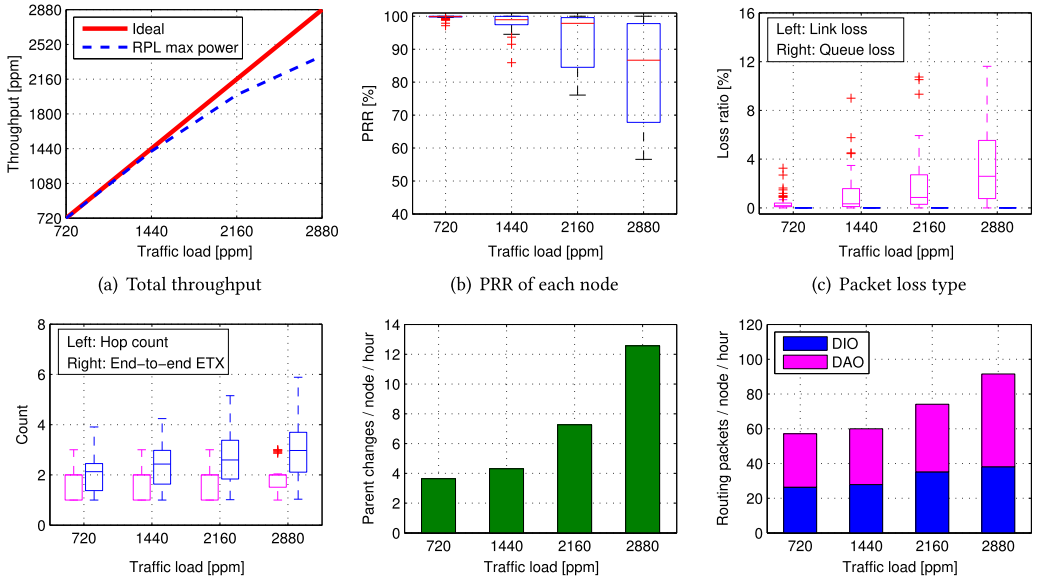
Our experiments uses TinyRPL, the RPL implementation on TinyOS that has been used in several studies [33, 38]. TinyRPL implements OF0 [62] OF with hop count and ETX metrics. Specifically, RANK of node k is calculated only based on its hop count from the LBR (i.e., $RANK(p_k) = hop\_count(k) + 1$), but the routing metric for a parent candidate node of node k, p_k, is calculated as "$RANK(p_k) + ETX(k\ to\ p_k)$." A node initializes the ETX value for a new neighbor entry as "1" (the best link quality), given that it has no prior knowledge of the new neighbor. Then, the node updates the ETX value for the neighbor node after each transmission to that node using an EWMA filter. TinyRPL operates in storing mode for downward routing. The more detailed operation and bugfixes of TinyRPL are described in Kim et al. [30, 31].

4.2 Effect of Traffic Load

Figure 6(a) through (f) plot relevant performance metrics of RPL with varying traffic load when nodes use transmit power of 0 dBm, which is the maximum allowed by the IEEE 802.15.4 standard. Figure 6(a) shows that RPL successfully delivers 99.7% of the generated traffic when traffic load is light (i.e., 720 pkts/min). This shows that, in our testbed environment, RPL establishes a reliable routing topology where each link quality is good enough to deliver a packet within five retransmissions.

However, the gap between traffic load and achieved throughput increases with traffic load. Specifically, when traffic load is 2,880 pkts/min, throughput becomes 2,400 pkts/min, which means that the network loses 17% of packets despite good link quality. Compared to the ideal results in Figure 4(b), the results imply that the full-power multihop network experiences some factors (other than link quality) that degrade the throughput. Furthermore, Figure 6(b) shows that the end-to-end packet reception ratio (PRR) from each node becomes significantly unfair as traffic load increases. Under traffic load of 2,880 pkts/min, the worst-performing node experiences only 57% PRR, whereas some nodes still achieve nearly 100% PRR.

To investigate the reasons for this severe and unbalanced packet loss, we check the routing topology by using Figure 7. However, we observe that traffic seems to be well distributed to many



(a) Total throughput (b) PRR of each node (c) Packet loss type

(d) Hop distance and ETX (e) Parent changes (f) Routing overhead

Fig. 6. Performance of RPL with varying traffic load when all nodes exploit maximum transmit power (0 dBm). RPL suffers from the hidden terminal problem, which not only degrades throughput but also churns routing topology.

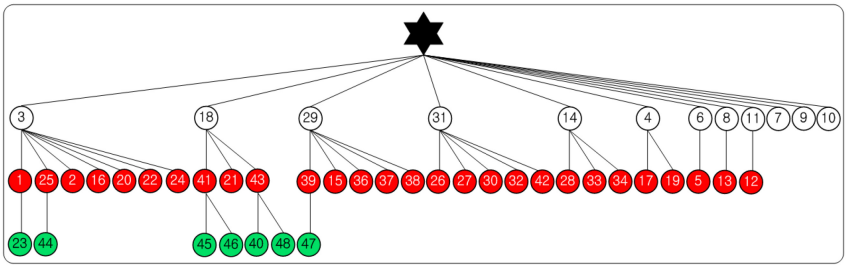


Fig. 7. Routing topology snapshot of RPL when using transmit power of 0 dBm at traffic load of 2,880 pkts/min.

branches and the most heavily loaded node (node 3) has only nine descendant nodes. The packet loss does not seem to come from a load imbalance problem.

What then is wrong in this network? To figure this out, we divide losses into two types—link loss and queue loss—and plot them in Figure 6(c). Here, we observe that as traffic load increases, most packet losses occur at links rather than at queues.[5] From the second finding of our preliminary study, we can infer that this multihop network has hidden terminals that cause packet losses due to collisions.

This can also be seen in Figure 6(d), which plots both the hop distance and ETX from each node to the border router. It shows that ETX increases as traffic load increases, whereas the hop distance remains constant (i.e., transmissions per hop increases). This indicates that increase in ETX under heavy load is solely caused by more link-layer retransmissions rather than more hop count, which

[5]Queue loss ratio is hard to see in Figure 6(c) because all of their values are close to zero.

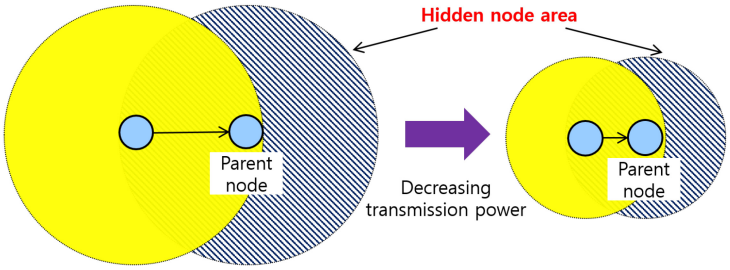


Fig. 8. Impact of transmission power on a hidden terminal in a multihop network. A smaller transmission power can decrease the number of hidden nodes by reducing transmission range and letting a routing protocol select a closer parent.

implies the presence of hidden terminals. Another observation is that even though the link layer is reflecting link status using ETX, RPL fails to provide reliable routes as shown in Figure 6(b).

To take a deeper look at the routing layer's behavior, Figure 6(e) and (f) show parent change frequency and routing control packet overhead, respectively. Figure 6(e) shows that RPL nodes change their parent more frequently as traffic load increases. This is because RPL uses ETX when selecting a parent, ETX reflects link congestion toward the parent, and each RPL node tries to avoid congestion by selecting an alternative parent node. Furthermore, Figure 6(f) shows that routing overhead (both DIO and DAO)[6] also increases with traffic load. Extra DAO traffic is a direct consequence of more frequent parent changes since an RPL node transmits a new DAO after each parent change to set up its downward route. Given that RPL resets *TrickleTimer* [42] when it detects routing inconsistency, additional DIO traffic implies more frequent route failures. However, when combined with the results of Figure 6(a) and (b), these efforts simply result in routing topology churn, without performance improvement.

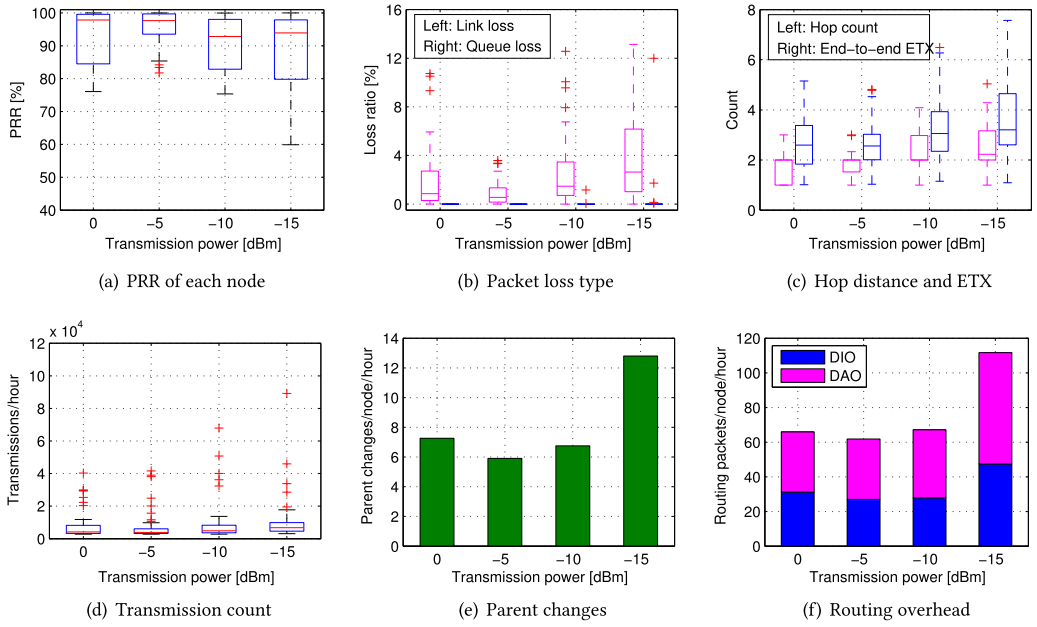
Overall, we find that RPL with the maximum transmit power suffers severely from the hidden terminal problem at heavy traffic load even though both the routing and link layers put effort into alleviating it. A significant amount of traffic is lost and extra energy is expended due to retransmissions and routing control overhead. We have the following observations:

- *Observation 1*: RPL experiences severe performance degradation at *heavy traffic load*: low throughput, PRR unfairness, unstable topology, and high routing overhead.
- *Observation 2*: RPL cannot deal with the *hidden terminal problem* even though it adjusts routing topology with an ETX-based routing metric.

4.3 Effect of Transmission Power

Based on the preceding findings, we ask the question, "if there is a severe hidden terminal problem in the network, can we alleviate it and improve throughput by adjusting the transmission power?" Although using a smaller transmission power may degrade throughput by increasing hop distance, our conjecture is that it can also improve throughput by alleviating a hidden terminal. When all nodes use a smaller transmission power, each RPL node may select a closer parent node with a smaller transmission range, which decreases the hidden node area as depicted in Figure 8. Furthermore, selecting a slightly closer node does not necessarily increase hop distance. To seek an answer, we conducted additional experiments at a load of 2,160 pkts/min (45 pkts/min/node) with varying transmit power from 0 dBm to −15 dBm, on all nodes. Figure 9 plots various performance metrics with this configuration.

[6]Please refer to RFC6550 [64] if unfamiliar with default RPL control messages.



(a) PRR of each node (b) Packet loss type (c) Hop distance and ETX

(d) Transmission count (e) Parent changes (f) Routing overhead

Fig. 9. Performance of RPL with varying transmit power at traffic load of 2,160 pkts/min (45 pkts/min/node). If all nodes use equal transmit power, simply reducing transmit power is not an effective solution to improve throughput.

First, we observe in Figure 9(a) that transmit power of −5 dBm provides better PRR performance than 0 dBm. The maximum allowed transmit power is not optimal; using a smaller transmit power can improve throughput by mitigating the hidden terminal problem. However, PRR degrades rapidly when transmit power is reduced too much. This provides an initial indication that we need a balance. We have the following observation:

- *Observation 3*: *Adaptive* transmission power control has the potential to improve throughput when the hidden terminal problem is severe with maximum transmission power.

To understand this phenomenon in more detail, Figure 9(b) divides packet losses into link loss and queue loss. It shows that link loss decreases at −5 dBm but increases again as transmit power is reduced further. Specifically, when using transmit power of −10 dBm or −15 dBm, link loss becomes larger than with 0 dBm despite fewer neighbor nodes. In addition, queue loss increases as transmit power is reduced. Importantly, severe queue losses occur only at a few nodes (one or two nodes in Figure 9(b)). By inspecting the routing topology, we identify the bottleneck nodes as the nodes with large unbalanced subtrees. For example, Figure 10 depicts a snapshot of RPL's routing topology when using transmit power of −15 dBm. Node 29 has a subtree with 27 nodes and suffers from very severe queue loss.

This load imbalance comes from RPL's use of only link quality and hop distance for parent selection [35]. In our testbed environment, obstacles such as doors, windows, and walls complicate wireless link quality, which leads to signal diversity even though all nodes use the same transmit power. Real-world deployments are likely to be subject to similar uneven signal density due to physical deployment and wireless link characteristics. This provides the following intuition:

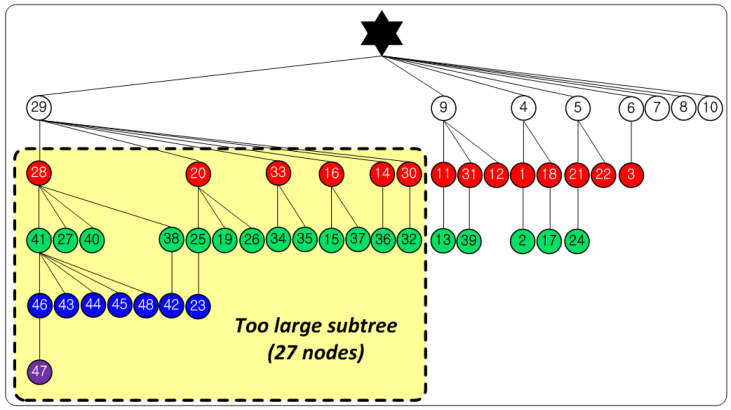


Fig. 10. Routing topology of RPL when using transmit power of −15 dBm, which clearly shows that RPL has a load imbalance problem.

- *Observation 4*: *Non-uniform* transmission power control has the potential to improve throughput by *load balancing*.

Figure 9(c) and (d) show why more link loss occurs when transmission power is too small even though this tends to reduce the number of hidden nodes. First, hop distance remains similar when transmission power decreases from 0 dBm to −5 dBm, which explains why −5 dBm provides better PRR than 0 dBm: fewer hidden nodes without hop distance increase. When transmission power is reduced further, hop count eventually increases, resulting in more forwarding overhead. In conjunction with the load imbalance problem, increase in forwarding overhead worsens link congestion, which leads to increased packet collision, packet loss, and resultant increase of link-layer retransmissions. As a result, end-to-end ETX increases faster than hop distance as transmit power decreases, as shown in Figure 9(c). Figure 9(e) and (f) show that the severe link congestion causes chaos on the RPL routing topology.

Our results show that as transmission power decreases, the number of neighbor nodes decreases, which mitigates link congestion. But at the same time, if it decreases too much, hop distance increases, which may result in more congestion. Thus, when designing a power control mechanism, we need to carefully mitigate hidden terminal and load imbalance problems while avoiding hop distance increase.

To confirm the results, we conducted the same experiments on another physically distinct 31-node testbed depicted in Figure 11. Figure 12 shows that PRR performance has similar trends as Figure 9(a) from a 49-node testbed. For example, when using 0 dBm transmission power, the network suffers the hidden terminal problem, which degrades PRR performance (Figure 12) and causes nodes 1, 3, 4, 5, 6, and 7 to select node 2 as the parent (temporarily) even though they are close to the LBR (Figure 11). Note that under the hidden terminal problem, RPL nodes (e.g., nodes 1, 3, 4, 5, 6, and 7) keep changing their parents (e.g., between nodes 2 and the LBR) since the ETX metric does not properly represent link quality anymore. As transmission power increases, the PRR performance first gets better and eventually worse again.

Moreover, it shows that the network achieves best PRR at −3 dBm, rather than −5 dBm, indicating that performance on different topologies cannot be optimized with same value of transmit power. Thus, we need a method that can automatically and dynamically find a good transmission power configuration for its topology at runtime. Better yet, this adaptive method should allow

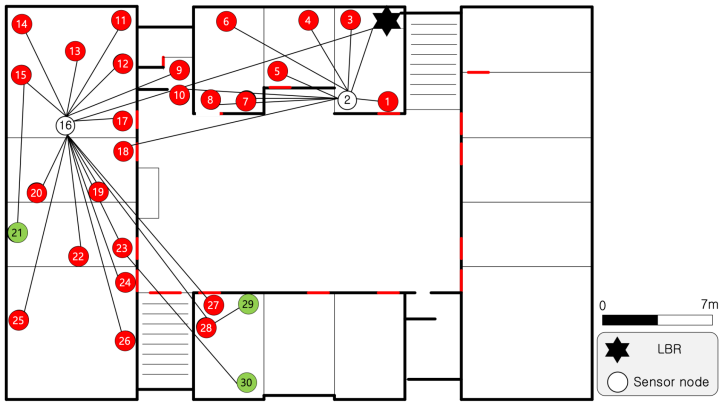


Fig. 11. Topology of another distinct 31-node testbed with a snapshot of routing paths given by RPL when using transmission power of 0 dBm.

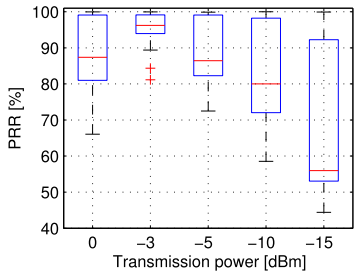


Fig. 12. PRR result from a different and smaller topology (31 nodes) at traffic load of 2,160 pkts/min (72 pkts/min/node).

each node to find a good transmission power autonomously in a distributed manner, thus relaxing the globally "uniform" constraint to achieve better performance. We have the following observations:

- *Observation 5*: It is difficult to find a single transmission power that fits all deployment scenarios when equal transmission power is used for all nodes.
- *Observation 6*: Adaptive transmission power control needs to be continued during runtime.

As a final note, all results show that throughput degradation and PRR unfairness happen together (i.e., some nodes are still healthy, whereas others are suffering), which implies that we may improve throughput by relieving a *few* suffering nodes from their problematic situations. These observations may be intuitive conceptually and theoretically, but solving the problem in real implementation is challenging to realize. This motivate us to design a distributed and adaptive control mechanism for transmission power and routing topology, which addresses both hidden terminal and load imbalance problems to not lose precious throughput. Specifically, in this work, we design and implement this control mechanism on top of the standard RPL and call this *PC-RPL* (power-controlled RPL).

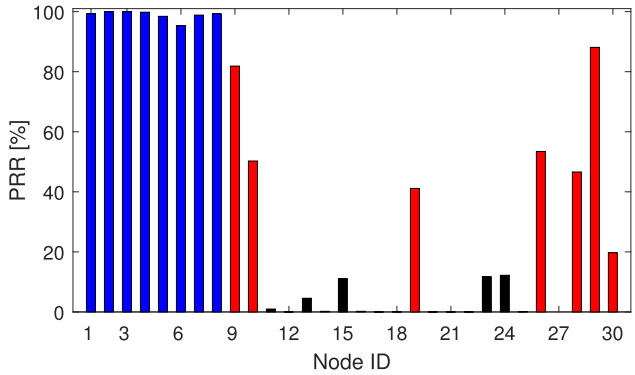


Fig. 13. PRR performance of 30 nodes when the LBR broadcasts a packet every 50 msec with transmission power of 0 dBm.

5 POWER CONTROL: OUT OF IDEAL GRAPH MODELS INTO THE WILD

Section 4 confirmed the need for adaptive power control in RPL-based multihop LLNs. Adaptive power (or topology) control, however, is not a new topic, and there have been several studies, which are summarized in Santi [58] and Stein et al. [60]. Then, why do we need a "new" power control mechanism? We can get a hint from Santi [58]: "despite many theoretical and simulation-based evidences of the effectiveness of topology control techniques in reducing energy consumption and/or increasing network capacity, to date there is little experimental evidence that topology control can actually be used to these purposes." Specifically, many theoretical approaches, such as Matsuura [47], use geographical distance as an important metric for power control with the assumption that link quality degrades as distance increases. Unfortunately, this assumption may not be true in real environments with various obstacles, such as doors, walls, windows, and furniture; a farther node may experience better link quality due to the shadowing effect![7]

To verify the risk of using distance for power control, we have conducted link measurements on the 31-node testbed (Figure 11) using one transmitter (marked as the star) and 30 receivers during a 24-hour period. The transmitter broadcasts a packet with transmission power of 0 dBm every 50 msec, and the receivers measure PRR for 24 hours. Thus, each PRR value comes out of 1,728,000 packet transmissions, which eliminates the effect of short-term channel variation. Only path loss and shadowing effect remains. Figure 13 depicts the PRR for each node averaged over the 24-hour period. It shows that there are 8 good nodes (blue bars) with PRR above 95% and 14 bad nodes (black bars) with PRR below 20%, whereas the remaining nodes (red bars) are in between. It is easy to find counterexamples to the preceding assumption from this figure. For example, node 27 underperforms nodes 28 and 29, although it is closer to the transmitter than these two nodes. Node 26 outperforms nodes 10 through 24 even though it is farther from the transmitter than all of them. The results confirm that we should not hastily convert distance to link quality in real-world environments with various obstacles. We have the following observation:

- *Observation 7*: Estimating link quality via an indirect metric (e.g., distance) is risky in complex real environments; to be used for a practical power control mechanism, wireless link quality should be measured directly.

[7]Link quality is determined by path loss, shadowing, and multipath fading. Path loss is related to distance, but the other two are affected by various environmental factors.

Building on this observation, *PC-RPL* does not rely on the distance information but uses RSSI, directly measured from signal received from each neighbor node. This enables *PC-RPL* to operate in practice without an unrealistic assumption.

6 PC-RPL: ADAPTIVE TOPOLOGY AND POWER CONTROL MECHANISM

In this section, we describe *PC-RPL* design in detail. In brief, *PC-RPL* is differentiated from prior power control mechanisms for LLN [13, 23, 53, 58, 60] as follows:. First, *PC-RPL* aims to mainly improve *throughput*, rather than energy efficiency, by alleviating hidden terminal and load balancing issues under heavy traffic. Second, based on the observations in Section 4, *PC-RPL* controls both routing topology and transmission power *jointly* to solve the problems more effectively. Third, *PC-RPL*'s control mechanisms do not rely on any unrealistic assumption but are built on a systematic, extensive empirical study exploiting the real world as the system model (e.g., wireless link, embedded hardware, and operating system).

6.1 High-Level Overview

PC-RPL's design comes from the basic intuitions derived from our previous experimental studies:

- When a node suffers from a packet loss, it can *self-detect* which kind of problem it is experiencing: queue loss or link loss.
- Assuming that RPL has constructed routes with good link quality, many link losses imply that a node may be experiencing a hidden terminal problem, and many queue losses imply that it may have unbalanced load due to excessively many children.
- It may be possible to adjust transmit power and routing topology to achieve load balancing and hidden terminal mitigation, resulting in better throughput.

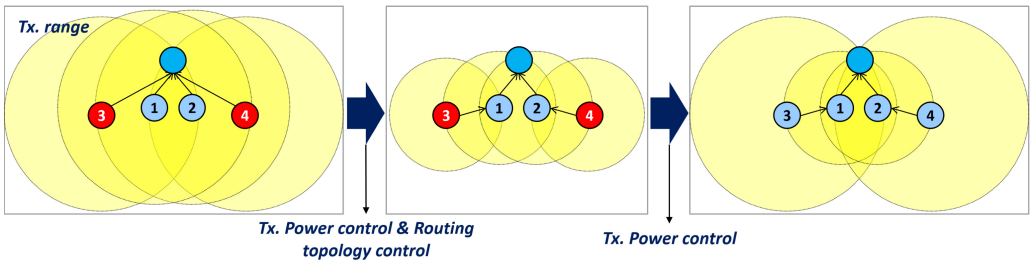
Based on these ideas, *PC-RPL* employs a new parent selection mechanism that considers new adaptive RSSI thresholds and a reference RSSI value of a parent candidate node, in addition to the default rules in RPL (i.e., hop distance and ETX-based parent selection). Specifically, *PC-RPL* has two additional RSSI thresholds—*parent selection RSSI threshold* and *children control RSSI threshold*—and allows each node to include a neighbor node in its parent candidate set only when its *reference RSSI* is higher than both the thresholds. *PC-RPL* controls these RSSI thresholds adaptively to mitigate hidden terminal problems and achieve load balancing. Furthermore, *PC-RPL* minimizes data transmit power according to the reference RSSI and transmission results (success or failure), which reduces hidden terminals and link congestion without sacrificing reliability.

6.2 Key Concepts and Parameters

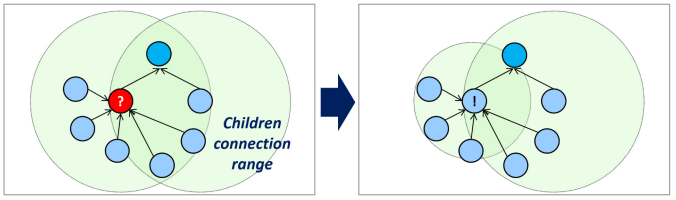
Compared to the standard RPL, the most distinct feature of *PC-RPL* is that it uses a reference RSSI value and adaptive RSSI thresholds for parent selection. We first define these key parameters and provide an overview of how they work for throughput improvement, before looking into the details of our algorithm. Although RSSI is known as a rough link-quality indicator, using this metric for *PC-RPL* is valid since it is not used for optimizing transmission parameters but for simply attaching or detaching a parent candidate node.

6.2.1 Reference RSSI Value. We define $RSSI_{\text{ref}}(k, n_k)$ as the *reference RSSI value* that node k holds as the reference link distance information to its neighbor node n_k. To obtain this reference RSSI value, *PC-RPL* makes all nodes transmit RPL DIO messages with *maximum* transmit power (0 dBm) without transmission power control.[8] Maximum power for DIO allows the network to

[8]We use 0 dBm for DIO transmission, but it can be any allowable maximum value within the limits of local regulations.



(a) Hidden terminal mitigation by adapting *parent selection RSSI threshold* and transmission power



(b) Load balancing by adapting *children control RSSI threshold* and detaching children nodes

Fig. 14. Effect of *PC-RPL*'s joint control of routing topology and transmission power on hidden terminal and load balancing issues.

seek for more link connectivity, if needed, even if only a subset is utilized for data communication. In addition, given that the amount of data traffic is far greater than that of DIO traffic in high traffic scenarios, the cost of using high transmit power for DIO messages can be easily amortized without causing link congestion.

When a node k receives a DIO message from a neighbor n_k, it uses the RSSI of the received DIO message as $RSSI_{\text{ref}}(k, n_k)$ and records this value in its neighbor table. While doing this, we smooth this reference RSSI value of each neighbor node using an EWMA filter in a way similar to ETX. Based on this $RSSI_{\text{ref}}(k, n_k)$, *PC-RPL* controls the three key parameters, $RSSI_{th}^{PS}(k)$, $RSSI_{th}^{CC}(k)$, and $N_{desired}^{SST}(k)$, described below to build a multihop topology that is free from load imbalance and hidden terminal problems.

6.2.2 Parent Selection RSSI Threshold. $RSSI_{th}^{PS}(k)$ is the *parent selection RSSI threshold* that node k maintains and uses for parent selection. Node k includes its neighbor node n_k in its parent candidate set $\mathbf{P}_k$ only when

$$RSSI_{\text{ref}}(k, n_k) > RSSI_{th}^{PS}(k). \tag{1}$$

PC-RPL uses this threshold for hidden terminal mitigation. As shown in Figure 14(a), when node k suffers from a hidden terminal problem, it increases $RSSI_{th}^{PS}(k)$ to select a "closer" parent node. With the help of transmission power control (explained in Section 6.5), this simple local behavior eventually adapts transmission range of all nodes, which alleviates the hidden terminal problem around the neighborhood of node k, and eventually in the whole network. Note that selecting a closer parent does not necessarily increase hop distance since the closer parent may have the same hop distance as the previous parent. Otherwise (if it is free from link loss), node k decreases $RSSI_{th}^{PS}(k)$ to select a longer distance (lower RSSI) node to reduce hop distance.

6.2.3 Children Control RSSI Threshold. $RSSI_{th}^{CC}(k)$ is the *children control RSSI threshold* maintained and sent by node k, and used by its neighboring nodes (potential children nodes) n_k for parent selection. To this end, node k propagates $RSSI_{th}^{CC}(k)$ to its neighbor nodes through DIO messages. A neighbor (potential child) node n_k can add node k to its parent candidate set $\mathbf{P}_{n_k}$

only when

$$RSSI_{\text{ref}}(n_k, k) > RSSI_{th}^{CC}(k). \tag{2}$$

Note that $RSSI_{\text{ref}}(n_k, k)$ is the reference RSSI value that node n_k has for node k.

PC-RPL exploits this threshold for load balancing as shown in Figure 14(b). When node k detects load imbalance (too many children) from frequent queue losses, it increases $RSSI_{th}^{CC}(k)$ to detach its children nodes (farthest located, first detached). Note that a detached child node does not necessarily increase its hop distance since it may have an alternative parent that has the same hop distance as the previous parent. Furthermore, it immediately transmits a DIO (with maximum transmit power) to fast propagate the increased $RSSI_{th}^{CC}(k)$ value to its children nodes. Then, a subset of the children nodes that have weak signal strength to node k (i.e., low reference RSSI value) are forced to change their parents, which results in reduced traffic load at node k.

However, queue loss may still occur even in a load-balanced network if traffic load is higher than achievable throughput. For example, even with a perfectly balanced topology shown in Figure 5(a), we observed that queue loss starts to occur as traffic load exceeds achievable throughput. In this case, increasing $RSSI_{th}^{CC}(k)$ worsens the performance by increasing hop count meaninglessly. To treat the overloaded situation differently, we use another parameter, $N_{desired}^{SST}(k)$, which is described next.

6.2.4 Desired Number of Sub-Subtree Nodes. $N_{desired}^{SST}(k)$ is the desired number of sub-subtree nodes (of node k) for each child node of node k. In our implementation, node k calculates $N_{desired}^{SST}(k)$ using the total number of nodes in its subtree,[9] divided by the number of 1-hop (direct) children nodes. Then, it propagates $N_{desired}^{SST}(k)$ to its children nodes via DIOs.[10]

When node k experiences a high queue loss rate, it distinguishes load imbalance from excessive traffic by using the $N_{desired}^{SST}(p_k)$ value received from its parent node p_k. Specifically, when its subtree size $N_{subtree}(k)$ is larger than $N_{desired}^{SST}(p_k)$, it detects load imbalance problem and increases $RSSI_{th}^{CC}(k)$. Otherwise, it assumes excessive traffic and takes no action to maintain stability.

6.3 RSSI Threshold Control

Figure 15 describes *PC-RPL*'s RSSI threshold control algorithm, which operates on each node in a fully distributed manner. *PC-RPL* classifies transmission results into three categories: success, link loss, and queue loss. It accumulates these results during a period T. If the total number of transmission results, $N_{tx}(k)$, is smaller than a *sampling threshold* α, our algorithm repeats this counting procedure during another period T to avoid a hasty decision with few samples.[11] These parameters can be chosen based on the maximum achievable bandwidth since, if the traffic load is very low (far below bandwidth), we do not need such a control to improve throughput.

If *PC-RPL* gathers a sufficient number of samples, it checks whether the total loss rate is larger than a *loss rate threshold* $\beta = 5\%$—that is,

$$R_{QL}(k) + R_{LL}(k) > \beta, \tag{3}$$

where $R_{QL}(k)$ and $R_{LL}(k)$ are queue and link loss rate, respectively. If not, *PC-RPL* takes no action and will fall back to default RPL since bandwidth is enough to handle the traffic. Otherwise, *PC-RPL* detects that node k is suffering severe packet loss[12] and tries to figure out which problem

[9]Each RPL node can obtain its subtree size from the number of downward routing entries in the routing table.
[10]DIO has 16 reserved bits, from which we use 8 bits to deliver $RSSI_{th}^{CC}(k)$ and the remaining 8 bits to deliver $N_{desired}^{SST}(k)$.
[11]We have used $T = 30$ seconds and $\alpha = 50$ in our experiments, which are large enough to mitigate hasty decisions.
[12]Given that link-layer retransmission and RPL's parent change mechanisms can handle most link dynamics and noise under reasonable traffic load, the severe packet loss comes from congestion: hidden terminal or load imbalance.

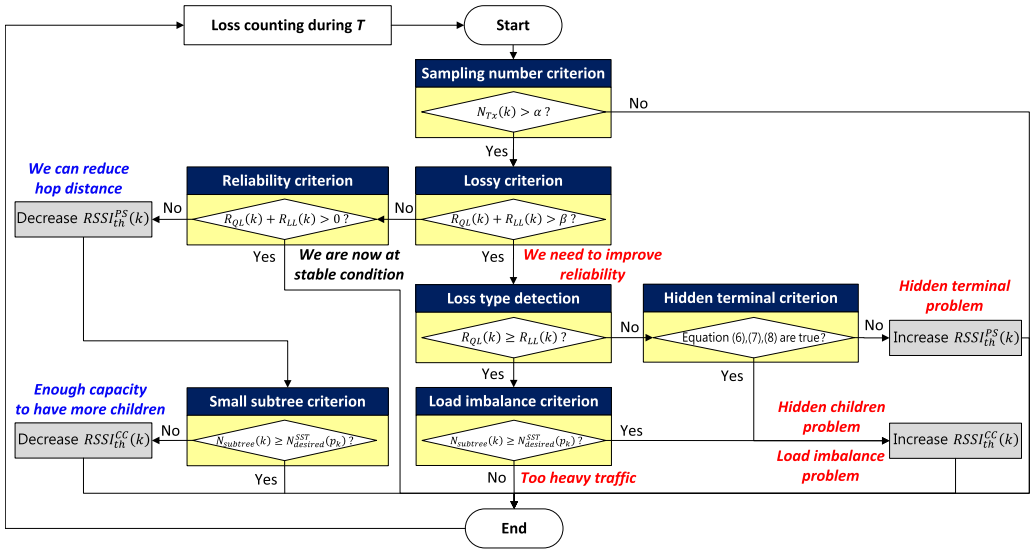


Fig. 15. *PC-RPL*'s RSSI threshold control algorithm. Each node *self-detects* the link state it is experiencing—good or bad, hidden terminal or load imbalance problem—and handles each case differently in a fully decentralized manner.

it is experiencing by using the "loss type criterion" :

$$R_{QL}(k) \geq R_{LL}(k). \tag{4}$$

If Equation (4) holds (more queue losses than link losses), *PC-RPL* checks the "load imbalance criterion" to see if

$$N_{subtree}(k) > N_{desired}^{SST}(p_k). \tag{5}$$

If this condition is satisfied, *PC-RPL* detects that node k is experiencing a load imbalance problem and increases $RSSI_{th}^{CC}(k)$ to reduce its subtree size. Specifically, it increases $RSSI_{th}^{CC}(k)$ just enough to detach the farthest child node for the stability of routing topology. Otherwise, if the subtree sizes are already balanced, *PC-RPL* determines that the traffic load exceeds achievable bandwidth and does not take any action.

However, if Equation (4) does not hold (more link losses than queue losses), *PC-RPL* makes a decision on how to alleviate the link losses. Specifically, it considers a "hidden terminal criterion" consisting of the following three conditions to determine whether or not it should increase $RSSI_{th}^{PS}(k)$ to seek for an alternative parent:

$$RSSI_{th}^{CC}(k) \leq CCA_{th} \tag{6}$$

$$N_{subtree}(k) > N_{desired}^{SST}(p_k) \tag{7}$$

$$|\mathbf{P}_k| = 1, \tag{8}$$

where CCA_{th} is the CCA threshold (−77 dBm by default) used for CSMA/CA.

If Equation (6) holds, node k may have children nodes with reference RSSI less than CCA_{th}, which means that they cannot detect node k's data transmissions (to its parent node p_k) with

CCA.[13] This may incur link losses due to collisions between node k's ACK receptions (from node p_k) and the children nodes' data transmissions. In this case, increasing $RSSI_{th}^{PS}(k)$ is meaningless since hidden terminals are children nodes. If Equation (7) holds, node k has large number of children nodes that are affected by its parent change, and thus its decision may have critical impact on the stability of routing topology. Last, if Equation (8) holds, further increasing $RSSI_{th}^{PS}(k)$ may incur route inconsistency since node k has only one node in its parent candidate set.

If all three conditions are true, it means that node k has an excessive number of children nodes, their transmissions may interfere with node k's transmissions (ACK reception, to be precise), and there is no alternative parent with equal RANK. Thus, *PC-RPL* determines that the main cause of the link losses is due to having a large number of *hidden* children nodes and decides to increase $RSSI_{th}^{CC}(k)$ to detach the farthest child node. Otherwise, *PC-RPL* detects that node k is suffering from a hidden terminal problem, and decides to increase $RSSI_{th}^{PS}(k)$ just enough to exclude the current parent node from the parent candidate set and obtain an alternative parent node with stronger signal strength.

If node k experiences packet loss below the loss rate threshold (i.e., $0 \leq R_{QL}(k) + R_{LL}(k) \leq \beta$), it keeps all of its thresholds unchanged to favor the stability of routing topology. However, if node k experiences no packet loss at all during period T, it attempts to relax the two RSSI thresholds to improve efficiency (in terms of hop distance and forwarding traffic) while maintaining its current reliability. In other words, *PC-RPL* reduces $RSSI_{th}^{PS}(k)$ just enough to include the "closest" neighbor node that can provide smaller hop distance than the current parent node p_k in the parent candidate set. Furthermore, if its subtree size is smaller than $N_{desired}^{SST}(p_k)$, it linearly (additively) decreases $RSSI_{th}^{CC}(k)$ to allow more children nodes. This helps other nodes that have large subtrees relieve their forwarding burden. Lastl, whenever a node detects route inconsistency, it re-initializes both $RSSI_{th}^{CC}(k)$ and $RSSI_{th}^{PS}(k)$ to −90 dBm.

To summarize, $RSSI_{th}^{CC}(k)$ is increased to detach far-away children nodes when a load balancing action is required, and it is decreased to attract children and provide shorter path length when load balancing action is not required. $RSSI_{th}^{PS}(k)$ is increased to select a parent with better signal strength when a node is experiencing severe link losses, and it is decreased to select a farther-away parent in a hope to reduce hop distance when its link reliability is good.

6.4 Parent Selection

Once all threshold values are determined, parent selection process of *PC-RPL* is a straightforward extension of the standard RPL. When a *PC-RPL* node k determines whether to include a neighbor node n_k to its parent candidate set $\mathbf{P}_k$, it first considers hop distance ($hop(n_k) < hop(k)$) and link-layer ETX as in standard RPL. Additionally, *PC-RPL* requires the following condition to be satisfied:

$$RSSI_{\text{ref}}(k, n_k) > max\left\{RSSI_{th}^{PS}(k), RSSI_{th}^{CC}(n_k)\right\}. \tag{9}$$

By the definitions and the control mechanisms of the two RSSI thresholds, a *PC-RPL* node k will do its best to select a parent node such that it avoids both hidden terminal and load imbalance problems.

However, it is possible that a node k needs to increase its $RSSI_{th}^{PS}(k)$ due to a hidden terminal problem but has only the current parent node p_k in its parent candidate set without any alternative. To allow this node to escape from hidden terminals, unlike RPL, *PC-RPL* allows this node to temporarily relax the hop distance condition to $hop(n_k) \leq hop(k)$ for the current selection process

[13] Given that CC2420's receiver sensitivity is −95 dBm, a node can receive a packet even though its RSSI is lower than CCA_{th}. Thus, it is possible that a node receives a DIO message from a neighbor node whose reference RSSI value is lower than CCA_{th} and selects it as the parent node.

(i.e., temporary increase of parent candidates). The reason we relax this condition only *temporarily* is to avoid routing loops. Last, when a node experiences hop distance increase after changing its parent node, it immediately transmits a DIO message to fast propagate this information and avoid any potential routing loop.

6.5 Transmission Power Control

After each parent change to parent node p_k, a node k configures its *data packet transmission power* to p_k, $P_{tx}^{data}(k)$, as

$$P_{tx}^{data}(k) = P_{tx}^{DIO} - \left(RSSI_{\text{ref}}(k, p_k) - RSSI_{th}^{def}\right). \tag{10}$$

As aforementioned, DIO transmission power P_{tx}^{DIO} is fixed to 0 dBm (max), and $RSSI_{\text{ref}}(k, p_k)$ is the reference RSSI value that node k calculates when receiving a DIO from its parent node p_k (sent using P_{tx}^{DIO}). Default RSSI threshold $RSSI_{th}^{def}$ is set to −77 dBm (= CCA_{th}), the default CCA threshold in our implementation. By setting $P_{tx}^{data}(k)$ larger than this $RSSI_{th}^{def}$ value, we allow node k's data transmissions to be detected by CCA of its parent node p_k (i.e., not hidden). This selection of $P_{tx}^{data}(k)$ allows node k to use just enough transmit power to reach its parent p_k while maintaining reliability.[14] When coupled with the RSSI threshold control, this transmit power control enables alleviation of a hidden terminal and link congestion.

This initial configuration of $P_{tx}^{data}(k)$ based on reference RSSI value is fast but needs further optimization since RSSI could be an imperfect metric due to external interference [29] and link asymmetry [3]. To this end, after this initial configuration of $P_{tx}^{data}(k)$, a node uses the following set of rules to adaptively control the transmit power. If the node succeeds in transmitting M consecutive packets without any retransmission (where M is the *good channel threshold*, 20 by default in our implementation), it decreases the transmit power linearly by 1 dBm (or one level allowed by the radio) to probe for a lower reliable power. Otherwise, if a node fails to transmit a packet at the link layer, it increases the transmit power additively by 2 dBm (or two levels allowed by the radio) and also increases the good channel threshold exponentially by $M \leftarrow M * 2$. The purpose of this exponential increase of M is to reduce the transmit power more conservatively when a packet loss is followed by a recent transmit power increase, thus suppressing repeated cycles of increases and decreases. Otherwise, it maintains its current transmit power. Finally, whenever a node detects route inconsistency, it re-initializes the transmit power and M to the initial values.

The goal of our transmission power control is to use just enough transmit power to reach its parent reliably. By coupling the *data transmission power* and *parent selection RSSI threshold* control together, *PC-RPL* mitigates both hidden terminal and link congestion, resulting in throughput improvement. For example, in the left part of Figure 14(a), all nodes initially use maximum transmission power and nodes 3 and 4 are hidden from each other. Given that they cannot further increase transmission power, the hidden terminal problem cannot be addressed without changing routes. To alleviate the problem, *PC-RPL* makes nodes 3 and 4 select nodes 1 and 2 (i.e., closer nodes) as their parents, respectively. Each node sets its transmission power just enough to reach its parent node by using the reference RSSI value. However, the hidden terminal still exists since node 2 is hidden from node 3 and node 1 is hidden from node 4. Then nodes 3 and 4 experience link losses and increase their transmission power to remove hidden nodes, resulting in the topology in the right part of Figure 14(a).

As a final note, unlike data transmissions, we fix the transmit power for ACK packets (from a parent to a child, for upward traffic) to 0 dBm (max). It is for successful ACK delivery to all

[14] If $RSSI_{\text{ref}}(k, p_k) < RSSI_{th}^{def}$, we set $P_{tx}^{data}(k)$ to 0 dBm (max).

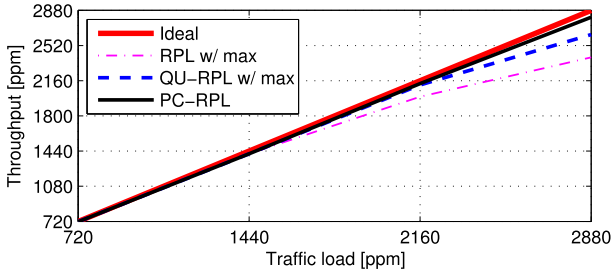


Fig. 16. Bandwidth of RPL, QU-RPL, and *PC-RPL*. *PC-RPL* successfully delivers 97.5% of traffic with packet losses that are 7 times less than RPL and 3.5 times less than QU-RPL.

children nodes without sophisticated child-by-child ACK transmission power control that requires significant management overhead. Given that ACKs are automatically transmitted by the radio hardware,[15] *PC-RPL* cannot know when and to whom an ACK is transmitted. Last, since an ACK is transmitted after a node succeeds in channel access and data packet transmission, using maximum transmit power for ACKs rarely incurs link contention.

7 EVALUATION

In this section, we evaluate the performance of *PC-RPL* on the same testbed setup as in Section 4 and compare it against RPL and QU-RPL [35], a queue-utilization-based RPL recently proposed to tackle the load imbalance problem of RPL (without power control nor addressing the hidden terminal issue). We also discuss the details of how *PC-RPL* adjusts its parameters to improve throughput.

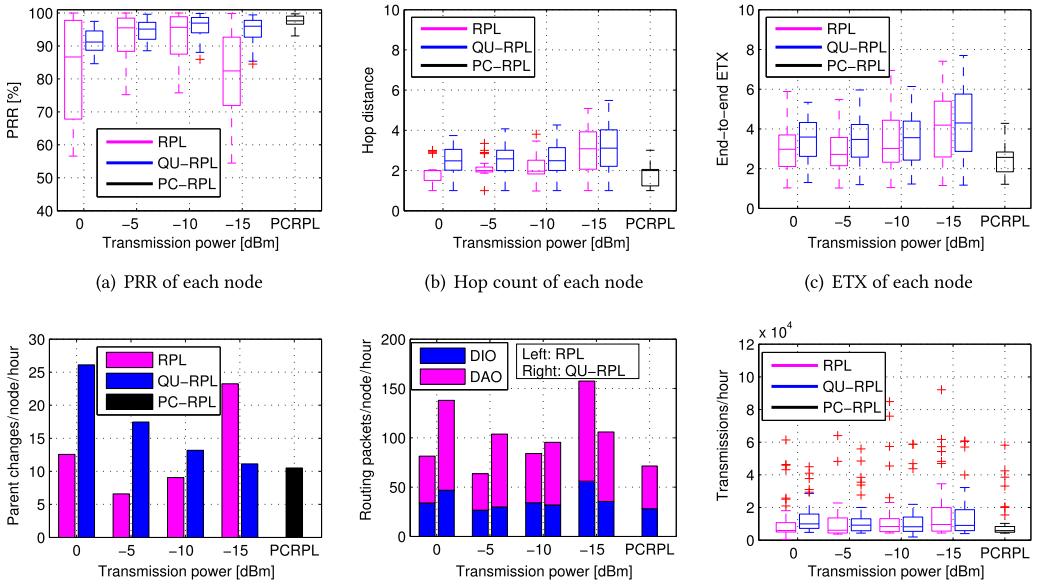
7.1 Packet Delivery Performance

Figure 16 plots the aggregate throughput of RPL (at 0 dBm), QU-RPL (at 0 dBm), and *PC-RPL*, and shows that *PC-RPL* provides better throughput than the other two. At a load of 2,880 pkts/min, *PC-RPL* achieves 2,810 pkts/min of aggregate throughput (i.e., 17% more than RPL and 7% more than QU-RPL). In other words, *PC-RPL* reduces packet loss rate by 7 times compared to RPL and 3.5 times compared to QU-RPL.

Figure 17 plots various performance metrics of RPL, QU-RPL, and *PC-RPL* at a traffic load of 2,880 pkts/min. Several important observations can be made from the PRR results in Figure 17(a). First of all, RPL suffers from low PRR (as already shown in Section 4.3, Figure 9(a)), whereas QU-RPL improves PRR significantly, compared to RPL, at all transmit power settings. Given that the goal of QU-RPL is to achieve load balancing by exploiting queue utilization as a metric, this implies that load balancing alone has a significant impact on improving throughput. Second, *PC-RPL* also provides dramatic PRR improvement over RPL, which is higher than the best case of RPL with uniform transmit power (−5 dBm). In the perspective of PRR fairness, *PC-RPL* improves 64% PRR for the worst-case node compared to RPL with 0 dBm. This shows that the use of adaptive and non-uniformly distributed transmit power achieves better throughput than using any equal transmit power for all nodes.

Third, *PC-RPL* outperforms QU-RPL in terms of PRR, better than the best case of QU-RPL with uniform transmit power at −10 dBm. Furthermore, QU-RPL's PRR performance degrades as transmit power increases, and further investigation reveals that most packet losses are link losses since QU-RPL cannot alleviate the hidden terminal problem. This confirms that *PC-RPL*'s adaptive control mechanism is at least as good as QU-RPL in resolving the load imbalance problem without

[15] We can use software-driven ACK, but it incurs delays in ACK transmissions and degrades throughput.



(a) PRR of each node (b) Hop count of each node (c) ETX of each node

(d) Parent changes (e) Routing overhead (f) Transmission count

Fig. 17. Performance of RPL, QU-RPL, and *PC-RPL* with varying transmit power at traffic load of 2,880 pkts/min (60 pkts/min/node). *PC-RPL* alleviates hidden terminal and load imbalance problems, and thus outperforms RPL and QU-RPL in terms of throughput and routing stability.

the use of queue utilization information, and in fact it achieves better throughput by alleviating a hidden terminal and link congestion as well.

Finally, note from the figure that the best-case transmit power for QU-RPL is −10 dBm, and RPL provides similar PRR when using the transmit power of −5 dBm and −10 dBm. Recall from Section 4.3 that the best transmit power for RPL under 2,160 pkts/min was −5 dBm (Figure 9(a)), and that from another 31-node topology was −3 dBm (Figure 12). This provides evidence that a single transmit power cannot provide the best performance for all scenarios, and also that the use of adaptive and non-uniform transmit power can achieve better throughput than using any equal transmit power for all nodes. More importantly, our *PC-RPL* achieves this improvement automatically and adaptively without requiring a system designer to manually optimize transmit power, which is lacking in both RPL and QU-RPL.

7.2 Link- and Routing-Layer Behavior

Figure 17(b) and (c) plot the hop count and end-to-end ETX of each node. From the former, we observe that the hop distance under both RPL and QU-RPL increases as transmit power decreases due to shorter transmission range. Furthermore, we can see that QU-RPL requires larger hop distance than RPL since it constructs balanced tree topology by sacrificing hop distance. Interestingly, combined with the results of Figure 17(a), the results show that in some cases, PRR can be better even with larger hop distance. QU-RPL provides better PRR than RPL in all transmission power settings even with larger hop distance since its load balancing gain exceeds the loss. In addition, as transmission power decreases from 0 dBm to −10 dBm, QU-RPL's PRR is improved even though hop distance continuously increases. It seems that when load balancing is given, the effect of hidden node mitigation can exceed the defect of hop distance increase. We can confirm that throughput

performance is not determined by a single factor, such as hop distance, but by the combination of various factors including hop distance, hidden terminal, and load balancing.

However, *PC-RPL* does not increase hop distance compared to RPL and provides end-to-end ETX lower than both RPL and QU-RPL regardless of their transmit power levels. This means that, while having similar hop distance to the border router, *PC-RPL* reduces link-layer retransmissions by alleviating both the hidden terminal and load imbalance problems. Given that *PC-RPL* requires fewer transmissions than RPL and QU-RPL for an end-to-end packet delivery, we estimate that *PC-RPL* may also improve latency performance compared to the two schemes.

Figure 17(d) and (e) show the routing layer's behavior of the three protocols. We first observe that QU-RPL's parent change frequency is larger than that of RPL in almost all cases since it detects the load imbalance problem and tries to avoid it by changing parent nodes. Combined with the results in Figure 17(a), this implies that RPL's relatively fewer parent changes are not because it provides good and stable topology but because it maintains bad topology due to lack of knowledge. RPL also suffers from meaningless parent changes when link congestion becomes severe (at −15 dBm).

Furthermore, QU-RPL's parent change frequency increases with transmit power. This is because, when using high transmit power, QU-RPL cannot stabilize the routing topology due to the hidden terminal problem. In contrast, parent change frequency of *PC-RPL* is smaller than that of QU-RPL in all transmit power settings while maintaining a high PRR. This reveals that *PC-RPL* avoids meaningless parent changes while load balancing by jointly controlling transmission power and routing topology. *PC-RPL*'s parent change frequency is even similar to the RPL's best case, which shows its efficient operation.

In addition, Figure 17(e) shows that routing control overhead is roughly proportional to the parent change frequency. This implies that QU-RPL and *PC-RPL* achieve PRR improvement at the expense of more routing overhead. However, given that each node generates 3,600 data packets per hour in our scenario, this increase in routing overhead (10∼20 extra control packets per hour compared to the RPL's best case) is a negligible cost compared to significant performance improvement.

Figure 17(f) plots the total transmission count (including data and control packets) of each node. Interestingly, in all cases, QU-RPL requires more transmissions on average but reduces that of the most bottlenecked (worst-case) node compared to RPL. The former result comes from the larger hop distance of QU-RPL and the latter from its load balancing effect. However, *PC-RPL* does not lose any of hop distance, ETX, and load balance, which reduces both transmission overhead for the bottlenecked node (−5.4%) and average transmission overhead (−4.4%) than the RPL's best case. Although *PC-RPL* mainly targets throughput improvement, our results show that *PC-RPL* also improves energy efficiency under heavy traffic.

Finally, Figure 18 is a snapshot of *PC-RPL*'s routing topology during an experiment. Compared to Figure 10, this figure shows that the previously largest subtree (with 27 nodes) has been divided into four smaller subtrees where the largest one (i.e., node 29's subtree) has 11 nodes, resulting in a shallower and relatively more balanced network. Quantitatively, *PC-RPL* reduced the standard deviation of the subtree size per node from 4.26 to 1.94, and average subtree size from 1.56 to 0.77. This means that *PC-RPL*'s load balancing function has taken effect as desired.

7.3 Transmission Power and RSSI Thresholds Control

Figure 19(a) plots the PRR experienced by *PC-RPL* for an hour as time goes by during an experiment, and Figure 19(b) through (d) present snapshots of the parent selection threshold $RSSI_{th}^{PS}$, children control threshold $RSSI_{th}^{CC}$, and transmission power of each node during the same experiment. These figures show that although *PC-RPL* experiences some losses during the beginning of the experiment when all threshold values and transmit power were uniform (default) among

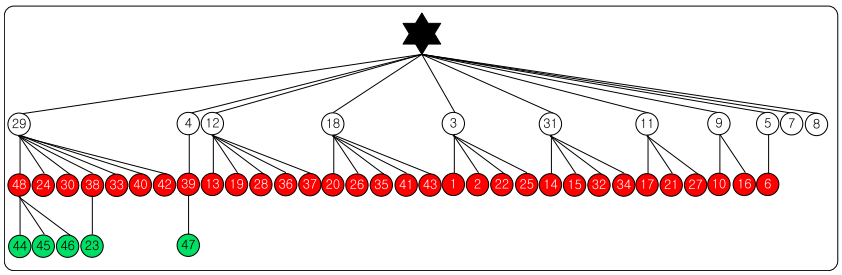
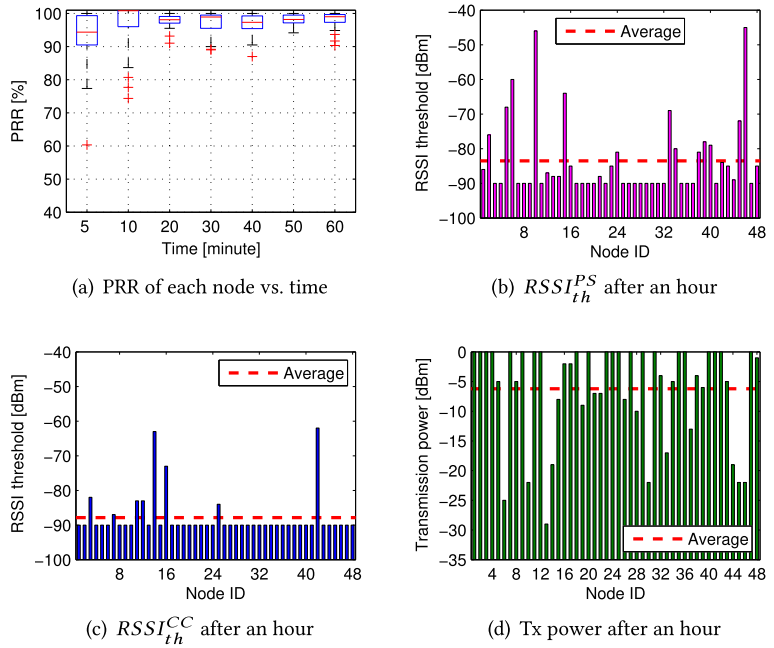


Fig. 18. A snapshot of *PC-RPL*'s routing topology at traffic load of 60 pkts/min/node. Compared to Figure 10, it shows that *PC-RPL* has taken load balancing action.



(a) PRR of each node vs. time

(b) $RSSI_{th}^{PS}$ after an hour

(c) $RSSI_{th}^{CC}$ after an hour

(d) Tx power after an hour

Fig. 19. Detailed operation of *PC-RPL*. *PC-RPL* improves PRR as time progresses by adjusting the two RSSI thresholds and transmission power. It results in a heterogeneous power network.

nodes, it improves throughput as time goes by through distributed and adaptive control of the two thresholds and transmission power.

Specifically, it increases $RSSI_{th}^{PS}$ from −90 dBm to −83.52 dBm on average, and $RSSI_{th}^{CC}$ from −90 dBm to −87.85 dBm on average. More interestingly, *PC-RPL* adjusted the two thresholds for only a small subset of nodes. Recall that $RSSI_{th}^{PS}$ is used mainly to alleviate hidden terminal problem, and $RSSI_{th}^{CC}$ is used for load balancing. Also recall that increasing the $RSSI_{th}^{PS}$ "reduces" the data transmission power of that node. This verifies that, indeed, a small subset of nodes suffered severely from load imbalance and hidden terminal problems, and *PC-RPL* successfully relieved them without disrupting other happily operating nodes.

Figure 19(d) clearly shows that *PC-RPL* constructs a multihop network with heterogeneous transmit power. In this experiment, *PC-RPL* reduces transmit power from 0 dBm to −6.21 dBm on average. Recalling that *PC-RPL* provides similar hop distance to RPL's 0 dBm case (Figure 17(b))

with reduced total number of transmissions (Figure 17(f)), we can see that maximum transmission power does not necessarily result in minimal hop distance and transmission overhead. Furthermore, we can confirm that, in complex "real-world" wireless environments, "non-uniform" transmission power is needed for more balanced topology.

8 RELATED WORK

Several studies have investigated topology control in the wireless multihop network, and Santi [58] provides an excellent survey of those works. In this survey, topology control via transmission power control is divided into "per-link power control" and "clustering/hierarchy" -based topology control, and the goal of topology control is described as reducing energy consumption and radio interference while maintaining some graph property (e.g., *k*-connectivity). The survey also provides a taxonomy of topology control techniques based on the constraints on the range assignment and on the type of information that is available to the network nodes. According to this taxonomy, *PC-RPL* can be regarded as a "non-homogeneous" approach that does not restrict same transmit power to all nodes, and also a "neighbor-based" approach that uses information from neighboring devices instead of location or direction information.

However, most of the work in Santi [58] (or not, e.g., Pan et al. [53]) is graph-theoretical work based on idealized graph models and node distributions without real implementation. The survey itself writes, "despite many theoretical and simulation-based evidences of the effectiveness of topology control techniques in reducing energy consumption and/or increasing network capacity, to date there is little experimental evidence that topology control can actually be used to these purposes." A more recent survey [60], published in 2016, still notes the practicality issue of topology control mechanisms. For example, Matsuura [47] aims to build power efficient multihop topology based on RPL routing. However, the proposal, a centralized topology construction mechanism, is built on unrealistic/ideal assumptions that are challenging to achieve in practice. First, link quality between two nodes is determined and fixed by the geographical distance between them. Second, each node somehow knows its link quality from all of its neighbor nodes, and the LBR can have a complete "bird's-eye view" of the whole topology. Although these mechanisms seem to work well in their simulation environments, it is challenging to implement and utilize them in practice as they are. In contrast, as a piece of "systems" research, this work aims to investigate topology control with an emphasis on practicality and applicability.

How can we build a practical topology control mechanism for LLNs? We answer this statement by designing and implementing a distributed topology control scheme for improved delivered bandwidth on real embedded devices, built into the Internet standard IPv6 routing protocol, and provide extensive experimental evidence of its effectiveness on a real multihop network. *PC-RPL* also meets the requirements of a topology control protocol suggested by the survey, such as being distributed, asynchronous, and localized, without requiring a manual power configuration.

A few prior works have investigated transmission power control within LLNs using real implementation, either to manage network topology [23, 59] or to reduce energy consumption and improve spatial reuse [40, 43]. In both cases, control mechanisms were designed to be simple to respect the resource constraints of LLN devices. In particular, they limited transmission power decisions to those based on RSSI [43, 59] or trial-and-error active probing [13, 23, 40]. RSSI-based methods typically assume stable channel conditions, whereas active-probing-based schemes are targeted at dynamic channel conditions. All of them assume that LLN generates light traffic and all nodes are battery powered, focusing on minimizing transmission power (energy consumption) while maintaining good link quality. In other words, none of these works consider throughput performance under high traffic scenarios. The power control mechanism in Hackman et al. [23] even worsens the packet delivery performance under high traffic compared to the use of maximum

transmission power. In contrast, *PC-RPL* goes beyond by coupling power control with routing to explicitly tackle hidden terminal and load imbalance problems under high traffic scenarios while combining ideas of RSSI-based and active-probing-based approaches.

The load imbalance problem of RPL in congested scenarios has been investigated in several pieces of work. Ha et al. [22] investigated the load imbalance problem in RPL when using multiple gateways. They proposed MLEq and compared its performance to that of RPL. However, they reduced traffic congestion only by using additional gateways and do not address the load imbalance problem in an LLN with a single gateway. Liu et al. [44] proposed LB-RPL, a variant of RPL that improves load balancing of RPL by allowing a node to prioritize its parent candidates by considering their queue utilization. Lodhi et al. [45] designed M-RPL, which detects traffic congestion through RPL control messages and provides two parent nodes for traffic distribution. However, these works provide neither implementation on real embedded devices nor experimental results. Boubekeur et al. [4] proposed BD-RPL, which restricts the subtree size of each node to relieve congestion. Kim et al. [31, 35] proposed QU-RPL, which uses queue utilization as an indicator to resolve congestion and load imbalance in RPL. These two pieces of work provide experimental evidence on testbeds comprising real embedded devices. More generally, an extensive RPL survey was recently published [33], which discusses 95 carefully selected works including LB-RPL, BD-RPL, and QU-RPL. None of them examined the use of transmission power control within RPL over a real multihop LLN testbed for jointly mitigating hidden terminal and load imbalance problems.

With respect to implementation, Ko et al. [38] experimentally evaluated the performance of RPL in TinyOS and showed that its performance is similar to that of CTP, the *de facto* data collection protocol in TinyOS. Two pieces of work in Kim et al. [30] and Park and Paek [55] investigated the performance of TCP when used over RPL via testbed experiments, and Ko et al. [39] investigated interoperability of two modes of operations (MOPs) defined in the RPL standard. In addition, Clausen et al. [9] provided a critical evaluation of RPL regarding limitations, trade-offs, and suggestions for improvements. None of these works examined the use of transmission power control within RPL over a real multihop LLN testbed for jointly mitigating link congestion and load imbalance.

Finally, we point out that our transmission power and topology control scheme is designed to be independent of the radio duty-cycling mechanism such as the low-power probing (LPP) [51] or low-power listening [56] schemes. Furthermore, given that the load imbalance issue occurs in large-scale applications where LLN routers are usually wall powered [8, 25, 54], prior work on RPL's load balancing did not use a duty-cycling mechanism [4, 31, 35]. Likewise, this work is orthogonal to the external interference issue, such as WiFi. Given that a number of frequency hopping–based interference avoidance mechanisms have been developed in the LLN research community [11, 37, 48], interaction between *PC-RPL* and those mechanisms would be an important future work.

9 CONCLUSION

We presented an adaptive and distributed mechanism for jointly controlling transmission power and routing topology in low-power multihop wireless networks. We have shown how routing topology and the hidden terminal problem affect achievable bandwidth in a multihop network, identified cases where a uniform transmission power configuration can be improved for better bandwidth, and quantified our findings through implementation and testbed experiments. We then implemented our control mechanism on top of the standard RPL protocol, called *PC-RPL*, which aims to achieve better bandwidth compared to RPL. *PC-RPL* tackles the hidden terminal and load imbalance problems in a multihop network by controlling the routing topology via transmission power and RSSI threshold control. We evaluated *PC-RPL* through extensive experiments on a real

49-node testbed, and our results showed that *PC-RPL* alleviates the packet loss problem, which led to significant improvement in achievable bandwidth and routing stability.

The adaptive techniques presented here are likely to be of value to most wireless routing protocols, but we do see some important future work. First, an experimental study on other modern platforms, such as CC2650 [24] and Hamilton [28], would be valuable since these fast and capable platforms may reduce processing delay significantly; the results in Section 3 could be different with these platforms. Second, this work does not tackle the wireless interference problem; synergistic integration with an advanced frequency hopping and resource scheduling technique would be beneficial to resolve load balancing, hidden terminal, and interference issues at the same time. Third, investigating this topology control issue on a larger-scale testbed with hundreds of nodes has a significant value since it conveys real-world link characteristics that large-scale simulations do not have, and reveals the scalability aspect that a small-scale testbed cannot show.

REFERENCES

[1] E. Ancillotti, R. Bruno, and M. Conti. 2013. The role of the RPL routing protocol for smart grid communications. *IEEE Communications Magazine* 51, 1 (Jan. 2013), 75–83.

[2] E. Ancillotti, R. Bruno, and M. Conti. 2014. Reliable data delivery with the IETF routing protocol for low-power and lossy networks. *IEEE Transactions on Industrial Informatics* 10, 3 (Aug. 2014), 1864–1877.

[3] Nouha Baccour, Anis Koubâa, Luca Mottola, Marco Antonio Zúñiga, Habib Youssef, Carlo Alberto Boano, and Mário Alves. 2012. Radio link quality estimation in wireless sensor networks: A survey. *ACM Transactions on Sensor Networks* 8, 4, Article 34 (Sept. 2012), 33 pages.

[4] Fadwa Boubekeur, Lélia Blin, Remy Leone, and Paolo Medagliani. 2015. Bounding degrees on RPL. In *Proceedings of the 11th ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet'15)*. 123–130.

[5] A. Brandt, J. Buron, and G. Porcu. 2010. Home Automation Routing Requirements in Low-Power and Lossy Networks. *RFC 5826* (April 2010). Retrieved January 7, 2020 from https://tools.ietf.org/html/rfc5826.

[6] M. Buevich, D. Schnitzer, T. Escalada, A. Jacquiau-Chamski, and A. Rowe. 2014. Fine-grained remote monitoring, control and pre-paid electrical service in rural microgrids. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'14)*.

[7] Marco Cattani, Andreas Loukas, Marco Zimmerling, Marco Zuniga, and Koen Langendoen. 2016. Staffetta: Smart duty-cycling for opportunistic data collection. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'16)*.

[8] Cisco Systems Inc.[n.d.]. Connected Grid Networks for Smart Grid—Field Area Network/CG-Mesh. Retrieved January 7, 2020 from http://www.cisco.com/web/strategy/energy/field_area_network.html.

[9] T. Clausen, U. Herberg, and M. Philipp. 2011. A critical evaluation of the IPv6 routing protocol for low power and lossy networks (RPL). In *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking, and Communications (WiMob'11)*.

[10] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. 2003. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom'03)*.

[11] Simon Duquennoy, Beshr Al Nahas, Olaf Landsiedel, and Thomas Watteyne. 2015. Orchestra: Robust mesh networks through autonomously scheduled TSCH. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'15)*.

[12] T. Winter (Ed.), P. Thubert (Ed.), A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. P. Vasseur, and R. Alexander. 2009. Routing Requirements for Urban Low-Power and Lossy Networks. *RFC 5548* (May 2009). Retrieved January 7, 2020 from https://tools.ietf.org/html/rfc5548.

[13] Yong Fu, Mo Sha, G. Hackmann, and C. Lu. 2012. Practical control of transmission power for wireless sensor networks. In *Proceedings of the 2012 20th IEEE International Conference on Network Protocols (ICNP'12)*.

[14] O. Gaddour, A. Koubaa, N. Baccour, and M. Abid. 2014. OF-FL: QoS-aware fuzzy logic objective function for the RPL routing protocol. In *Proceedings of the 2014 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'14)*. 365–372. DOI: https://doi.org/10.1109/WIOPT.2014.6850321

[15] German Federal Ministry of Education and Research. [n.d.]. Project of the Future: Industry 4.0. Retrieved January 7, 2020 from http://www.bmbf.de/en/19955.php.

[16] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. 2009. Collection tree protocol. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'09)*.

[17] O. Gnawali and P. Levis. 2010. The ETX Objective Function for RPL. draft-gnawali-roll-etxof-01 (May 2010). Retrieved January 7, 2020 from https://tools.ietf.org/id/draft-gnawali-roll-etxof-01.html.

[18] O. Gnawali and P. Levis. 2012. The Minimum Rank with Hysteresis Objective Function. *RFC 6719* (Sept. 2012). Retrieved January 7, 2020 from https://tools.ietf.org/html/rfc6719.

[19] Mesh Working Group. January 2019. Mesh Profile v1.0.1. https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx?doc_id=457092.

[20] Thread Group. 2015. Thread Stack Fundamentals. Retrieved January 7, 2020 from http://threadgroup.org/.

[21] V. C. Gungor and G. P. Hancke. 2009. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *IEEE Transactions on Industrial Electronics* 56, 10 (Oct. 2009), 4258–4265. DOI: https://doi.org/10.1109/TIE.2009.2015754

[22] Minkeun Ha, Kiwoong Kwon, Daeyoung Kim, and Peng-Yong Kong. 2014. Dynamic and distributed load balancing scheme in multi-gateway based 6LoWPAN. In *Proceedings of the 2014 IEEE International Conference on Internet of Things (iThings'14), IEEE Green Computing and Communications (GreenCom'14), and IEEE Cyber, Physical, and Social Computing (CPSCom'14)*.

[23] Gregory Hackmann, Octav Chipara, and Chenyang Lu. 2008. Robust topology control for indoor wireless sensor networks. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'08)*.

[24] Texas Instruments. 2016. CC2650 SimpleLink™ Multistandard Wireless MCU. Retrieved January 7, 2020 from http://www.ti.com/lit/ds/symlink/cc2650.pdf.

[25] J. Martocci (Ed.), P. De Mil, N. Riou, and W. Vermeylen. 2010. Building Automation Routing Requirements in Low-Power and Lossy Networks. *RFC 5867* (June 2010). Retrieved January 7, 2020 from https://tools.ietf.org/html/rfc5867.

[26] Deokwoo Jung, Zhenjie Zhang, and Marianne Winslett. 2017. Vibration analysis for IoT enabled predictive maintenance. In *Proceedings of the IEEE 33rd International Conference on Data Engineering (ICDE'17)*. IEEE, Los Alamitos, CA, 1271–1282.

[27] K. Pister (Ed.), P. Thubert (Ed.), S. Dwars, and T. Phinney. 2009. Industrial Routing Requirements in Low-Power and Lossy Networks. *RFC 5673* (Oct. 2009). Retrieved January 7, 2020 from https://tools.ietf.org/html/rfc5673.

[28] Hyung-Sin Kim, Michael P. Andersen, Kaifei Chen, Sam Kumar, William J. Zhao, Kevin Ma, and David E. Culler. 2018. System architecture directions for post-soc/32-bit networked sensors. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. ACM, New York, NY, 264–277.

[29] Hyung-Sin Kim, Hosoo Cho, Myung-Sup Lee, Jeongyeup Paek, JeongGil Ko, and Saewoong Bahk. 2015. MarketNet: An asymmetric transmission power-based wireless system for managing e-price tags in markets. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'15)*.

[30] Hyung-Sin Kim, Heesu Im, Myung-Sup Lee, Jeongyeup Paek, and Saewoong Bahk. 2015. A measurement study of TCP over RPL in low-power and lossy networks. *Journal of Communications and Networks* 17, 6 (Dec. 2015), 647–655.

[31] Hyung-Sin Kim, Hongchan Kim, Jeongyeup Paek, and Saewoong Bahk. 2017. Load balancing under heavy traffic in rpl routing protocol for low power and lossy networks. *IEEE Transactions on Mobile Computing* 16, 4 (April 2017), 964–979.

[32] Hyung-Sin Kim, JeongGil Ko, and Saewoong Bahk. 2017. Smarter markets for smarter life: Applications, challenges, and deployment experiences. *IEEE Communications Magazine* 55, 5 (2017), 34–41.

[33] Hyung-Sin Kim, JeongGil Ko, David E. Culler, and Jeongyeup Paek. 2017. Challenging the IPv6 routing protocol for low-power and lossy networks (RPL): A survey. *IEEE Communications Surveys and Tutorials* 19, 4 (Sept. 2017), 2502–2525. DOI: https://doi.org/10.1109/COMST.2017.2751617

[34] Hyung-Sin Kim, Myung-Sup Lee, Young-June Choi, Jeonggil Ko, and Saewoong Bahk. 2016. Reliable and energy-efficient downward packet delivery in asymmetric transmission power-based networks. *ACM Transactions on Sensor Networks* 12, 4 (Sept. 2016), Article 34, 25 pages.

[35] Hyung-Sin Kim, Jeongyeup Paek, and Saewoong Bahk. 2015. QU-RPL: Queue utilization based RPL for load balancing in large scale industrial applications. In *Proceedings of the 2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON'15)*.

[36] Hyung-Sin Kim, Jeongyeup Paek, David E. Culler, and Saewoong Bahk. 2017. Do not lose bandwidth: Adaptive transmission power and multihop topology control. In *Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'17)*.

[37] Seohyang Kim, Hyung-Sin Kim, and Chongkwon Kim. 2019. ALICE: Autonomous link-based cell scheduling for TSCH. In *Proceedings of ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'19)*.

[38] Jeonggil Ko, Stephen Dawson-Haggerty, Omprakash Gnawali, David Culler, and Andreas Terzis. 2011. Evaluating the performance of RPL and 6LoWPAN in TinyOS. In *Proceedings of Extending the Internet to Low Power and Lossy Networks (IP+SN'11)*.

[39] Jeonggil Ko, Jongsoo Jeong, Jongjun Park, Jong Arm Jun, Omprakash Gnawali, and Jeongyeup Paek. 2015. DualMOP-RPL: Supporting multiple modes of downward routing in a single RPL network. *ACM Transactions on Sensor Networks* 11, 2 (March 2015), Article 39, 20 pages. DOI: https://doi.org/10.1145/2700261

[40] JeongGil Ko and Andreas Terzis. 2010. Power control for mobile sensor networks: An experimental approach. In *Proceedings of the 2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON'10)*.

[41] Sam Kumar, Michael P. Andersen, Hyung-Sin Kim, and David E. Culler. 2018. TCPlp: System design and analysis of full-scale TCP in low-power networks. arXiv:1811.02721.

[42] Philip Levis, Neil Patel, David Culler, and Scott Shenker. 2004. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the 1st Conference on Networked Systems Design and Implementation (NSDI'04)*.

[43] Shan Lin, Jingbin Zhang, Gang Zhou, Lin Gu, John A. Stankovic, and Tian He. 2006. ATPC: Adaptive transmission power control for wireless sensor networks. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'06)*.

[44] Xinxin Liu, Jianlin Guo, G. Bhatti, P. Orlik, and K. Parsons. 2013. Load balanced routing for low power and lossy networks. In *Proceedings of the 2013 IEEE Wireless Communications and Networking Conference (WCNC'13)*. 2238–2243. DOI: https://doi.org/10.1109/WCNC.2013.6554908

[45] M. A. Lodhi, A. Rehman, M. M. Khan, and F. B. Hussain. 2015. Multiple path RPL for low power lossy networks. In *Proceedings of the 2015 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob'15)*. 279–284.

[46] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. 2002. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*. ACM, New York, NY, 88–97.

[47] Hiroshi Matsuura. 2014. New routing framework for RPL: Constructing power-efficient wireless sensor network. In *Proceedings of the 2014 IEEE Network Operations and Management Symposium (NOMS'14)*. IEEE, Los Alamitos, CA, 1–9.

[48] 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 ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'16)*.

[49] Gabriel Montenegro, Nandakishore Kushalnagar, Jonathan Hui, and David Culler. 2007. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. *RFC 4944* (Sept. 2007). Retrieved January 7, 2020 from https://tools.ietf.org/rfc4944.

[50] David Moss, Jonathan Hui, and Kevin Klues. [n.d.]. Low Power Listening. *TinyOS TEP 105*. Retrieved January 7, 2020 from https://www.academia.edu/2784300/Low_power_listening.

[51] Răzvan Musăloiu-E., Chien-Jan Mike Liang, and Andreas Terzis. 2008. Koala: Ultra-low power data retrieval in wireless sensor networks. In *Proceedings of the 2008 International Conference on Information Processing in Sensor Networks (ISPN'08)*.

[52] Jeongyeup Paek, John Hicks, Sharon Coe, and Ramesh Govindan. 2014. Image-based environmental monitoring sensor application using an embedded wireless sensor network. *Sensors* 14, 9 (2014), 15981–16002.

[53] Jianping Pan, Y. Thomas Hou, Lin Cai, Yi Shi, and Sherman X. Shen. 2003. Topology control for wireless sensor networks. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom'03)*.

[54] Jaeyeon Park, Woojin Nam, Taeyoung Kim, Jaewon Choi, Sukhoon Lee, Dukyong Yoon, Jeongyeup Paek, and JeongGil Ko. 2017. Glasses for the third eye: Improving clinical data analysis with motion sensor-based filtering. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems (SenSys'17)*. 99–112.

[55] Mingyu Park and Jeongyeup Paek. 2019. TAiM: TCP assistant-in-the-middle for multihop low-power and lossy networks in IoT. *Journal of Communications and Networks* 21, 2 (April 2019), 188–195.

[56] Joseph Polastre, Jason Hill, and David Culler. 2004. Versatile low power media access for wireless sensor networks. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'04)*.

[57] 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)*.

[58] Paolo Santi. 2005. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys* 37, 2 (June 2005), 164–194. DOI: https://doi.org/10.1145/1089733.1089736

[59] Dongjin Son, B. Krishnamachari, and J. Heidemann. 2004. Experimental study of the effects of transmission power control and blacklisting in wireless sensor networks. In *Proceedings of the 2004 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*.

[60] M. Stein, T. Petry, I. Schweizer, M. Brachmann, and M. Muhlhauser. 2016. Topology control in wireless sensor networks: What blocks the breakthrough? In *Proceedings of the 2016 IEEE 41st Conference on Local Computer Networks (LCN'16)*.

[61] R. Szewczyk, A. Mainwaring, J. Anderson, and D. Culler. 2004. An analysis of a large scale habitat monitoring application. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys'04)*.

[62] P. Thubert. 2012. Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL). *RFC 6552* (March 2012). Retrieved January 7, 2020 from https://tools.ietf.org/html/rfc6552.

[63] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. 2006. Fidelity and yield in a volcano monitoring sensor network. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*. 381–396.

[64] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. P. Vasseur, and R. Alexander. 2012. RPL: IPv6 routing protocol for low-power and lossy networks. *RFC 6550* (March 2012). Received January 7, 2020 from https://tools.ietf.org/html/rfc6550.

[65] Alec Woo, Terence Tong, and David Culler. 2003. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*. ACM, New York, NY, 14–27.

[66] Gang Zhou, Tian He, Sudha Krishnamurthy, and John A. Stankovic. 2004. Impact of radio irregularity on wireless sensor networks. In *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services*. ACM, New York, NY, 125–138.

Received December 2017; revised July 2019; accepted October 2019