Andreas Reinhardt, Christian Renner: *Remote Node Reconfiguration in Opportunistic Data Collection Wireless Sensor Networks*. In: Proceedings of the IEEE Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM), pp. 1-3, IEEE Press, June 2014.

Remote Node Reconfiguration in Opportunistic Data Collection Wireless Sensor Networks

Andreas Reinhardt
The University of New South Wales, Sydney, Australia
andreasr@cse.unsw.edu.au

Christian Renner
Universität zu Lübeck, Lübeck, Germany
renner@iti.uni-luebeck.de

***Abstract*—Traditionally, wireless sensor networks collect readings from distributed embedded sensing systems and forward them to one or more sink nodes. While many energy-efficient data collection protocols have emerged as a result, the transmission of control commands from a sink to individual nodes in the network is generally unsupported by these solutions. We present how the receiver-initiated opportunistic ORiNoCo data collection protocol can be extended to allow for the reconfiguration of nodes at minimal additional energy overhead. When our solution is being applied, adaptations of the sensor sampling rates or node sleep cycles can be easily controlled by the base station during runtime.**

I. Introduction

Numerous data collection protocols for the energy-efficient data transfer from multiple nodes to a sink have been proposed for wireless sensor networks (WSNs), e.g., [1, 2]. As they primarily target to optimize the data flow from nodes towards the sink, however, these protocols do not provide support for the reconfiguration of nodes during runtime (e.g., to change their sampling rates). Instead, flooding or dissemination protocols like Trickle [3] or DIP [4], which distribute messages to all nodes in the WSN, are generally applied to this end. Unfortunately, these protocols neither support the addressing of individual nodes nor have they been designed with a focus on the energy-efficient integration with collection protocols.

In this paper, we thus present a solution that synergistically fuses data collection and control command distribution. The approach is based on the opportunistic receiver-initiated no-overhead collection (ORiNoCo) protocol [5]. In a nutshell, ORiNoCo achieves its low power consumption by duty-cycling the radio and low-power probing. Once a node wants to send data, it activates its radio and collects beacons from its neighbors that carry a path cost metric (e.g., their distance to the sink). The node then chooses the neighbor with the best path cost, sends its data there, and returns to sleep mode. In essence, ORiNoCo thus establishes a loosely coupled tree-like routing structure, solely using the path cost metric to ensure that messages are relayed towards the sink. It can hence inherently cope better with changing channel qualities than existing data collection protocols like [1] and [2].

Our presented bidirectional routing approach synergistically adds command and routing information to ORiNoCo messages. By leveraging the existing message types for its operation, only a minimal energy overhead for the reconfiguration of individual devices is introduced. Moreover, its very small and constant memory overhead to all nodes in the WSN also distinguishes our approach from existing routing protocols like RPL [6] or ORPL [7], where much more routing information must be stored locally at the nodes. As a result, the presented seamless integration of routing capabilities into ORiNoCo is simple and lightweight, while it adds the new feature of remote node reconfiguration in data collection WSNs.

II. Remote Node Reconfiguration in ORiNoCo

As our primary objective is to retain ORiNoCo's ultra-low power consumption, avoiding the energy overhead introduced by additional packet transmissions is of utmost importance. We thus present in this section which new data fields are required and how they are symbiotically combined with existing ORiNoCo messages.

A. Required Additional Fields

Destination Addressing: The sink is by convention the destination for all data packets in the WSNs under consideration. Control messages, in contrast, are emitted by the sink and addressed to devices in the network. For the distribution of a control command, it is thus vital to be able to specify the intended recipient, for which purpose a two-byte field for the destination address is required.

Command Definition: The destination of a reconfiguration message must be able to determine which action to take. To minimize additional delays, command identifiers are stored within the control message and can hence directly be extracted from the message upon its reception. For this purpose, a field of one byte has been added to each control message, for which we have defined an initial configuration of frequently used commands, including, e.g., requests to change a node's sampling rate or its transmission intervals.

Duplicate Detection: Control messages may reach the same node multiple times, although they should generally only be executed once upon their first reception. In order to prevent the repeated execution of commands, we have incorporated a version number field. Its value is incremented whenever the destination node or the command identifier have changed. As the version unambiguously refers to the recipient and requested command, it serves both as a means to avoid the duplicate execution of commands and to identify if a neighboring node needs updating. Version numbers are solely assigned by the sink node, hence their consistency is guaranteed throughout the network.

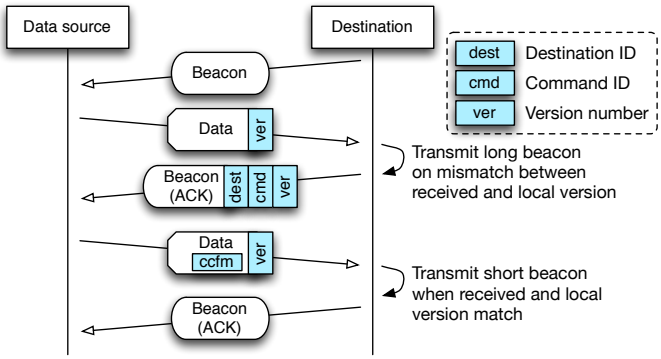

Fig. 1. Communication sequence for the transmission of 2 data packets (newly introduced fields are highlighted).

B. Modified and New Message Types

To retain ORiNoCo's ultra-low power operation, we focus on adding command information to its existing message types where possible instead of defining new message types. The following modifications to existing packets are required.

Beacon Messages: Beacons represent the most frequently transmitted packet type in ORiNoCo, and we thus leverage them in order to quickly disseminate control messages into the WSN. To this end, support to carry the aforementioned newly required fields is added. By default, however, these optional fields are not part of transmitted beacons; beacon sizes thus are unchanged in comparison to ORiNoCo and its beaconing energy efficiency is retained. Only when sender and receiver of a data message feature different versions (see below), these fields are added to the beacon in order to update the receiving node with the latest information. We term the beacon messages without the newly introduced fields as *short beacons*, whereas beacons that contain these three entries are called *long beacons*. To enable the receiver to interpret the beacons correctly, a flag was added to the previously existing ORiNoCo beacon flags field to indicate the presence of the command fields in the beacon.

Data Messages: While beacons originate from the sink, the second-most frequently packet type, data packets, travel in the opposite direction, i.e., towards the sink. By adding a node's current version to all transmitted data packets, the recipient (which is closer to the sink by definition) can determine whether a version mismatch exists, i.e., if a new command should be propagated to the data source. If this is the case, a long acknowledgment beacon can be easily used to update the data source to the current routing version. In case both devices share the same version of the routing information, the data message is acknowledged using a short beacon.

Command Confirmation Messages: Finally, we added a new message type, allowing nodes to acknowledge to the sink that they have received and executed a command. This *confirmation* message is sent as a regular data packet with the command confirm (*ccfm*) flag set and contains the control message version number for verification.

C. Communication Flow

In Fig. 1, the resulting communication flow is shown, and any differences to the transmitted information by ORiNoCo are highlighted. The operation shown on the right-hand side represents the comparison between the received and node's local version. After the first data transmission, the destination has detected a mismatch between its local and the received version, such that the returned acknowledgment beacon is augmented by the control command data. Subsequently transmitted data packets immediately reflect the newest version number, and thus the destination only transmits short acknowledging beacons for all successive packet transmissions. Because control commands originate from the sink only, whereas the direction of data traffic is opposite, direct neighbors of the sink receive the updated routing information as soon as they have transmitted a packet. Iteratively, control commands are being disseminated through the network, and move further away from the sink with every node that transmits data and receives the corresponding acknowledgment beacon.

III. Evaluation

We focus on a practical evaluation of our presented approach in a testbed setting in order to prove its potential to route control messages to individual nodes in the network. More precisely, we practically evaluate typical delays for control commands to reach their destinations as well as the time required until the command confirmation is received at the sink again. We base our practical evaluations on the WiseBed testbed [8], comprised of 54 TelosB nodes. Nodes are located in office rooms at the University of Lübeck and cover an area of approximately $15\,\mathrm{m} \times 50\,\mathrm{m}$.

A. Testbed Evaluation Setup

We have used 49 nodes in the testbed experiments as data sources, and configured them to create a new data packet with sensor readings every $2\,\mathrm{min}$. All nodes are furthermore supplied with outbound packet buffers that can accommodate up to 30 packets in order to cater for the intermittent disconnection of nodes due to poor channel conditions or when incoming packets need to be buffered before they can be forwarded again. Still, unless congestion occurred, all data in the buffer were generally sent at the next transmission opportunity. To retain ORiNoCo's high energy efficiency, all nodes in the field only wake up from sleep mode once every $750\,\mathrm{ms}$ in order to announce their path cost by means of a beacon. The wake-up interval is furthermore randomly varied by up to $10\,\%$ in order to avoid continuous beacon collisions. To allow for higher delivery rates, the sink is the only node in the network that has been configured to provide opportunities to receive data (by sending beacons) every $375\,\mathrm{ms}$. The sink issues a new command addressed to a single node in a round-robin fashion every $10\,\mathrm{min}$. In order to highlight the impact of the network topology, we have configured the nodes to use different transmission power levels ($0\,\mathrm{dBm}$, $-7\,\mathrm{dBm}$, and $-15\,\mathrm{dBm}$) in three separate runs, such that topologies with different depths result.

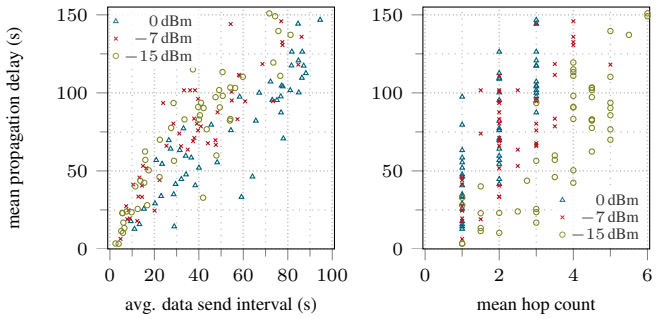

Fig. 2. Effect of network density on the relationship of propagation delay and send interval and hop count, respectively.

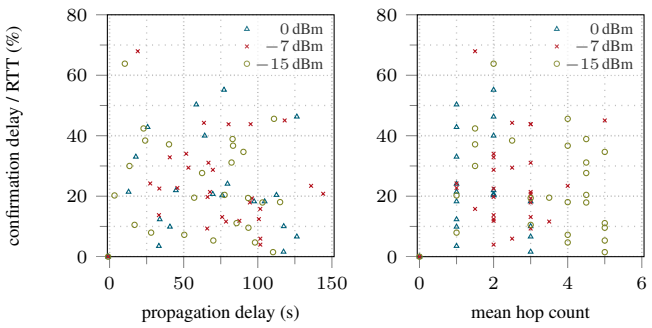

Fig. 3. Effect of network density on confirmation delay relative to round-trip time (RTT) vs. propagation delay and hop count.

B. Evaluation Results

Command Propagation Delay: Our first analysis considers the command propagation delay, or reaction time, of a node. As highlighted above, nodes can only receive beacons when they have data to send. Hence, their opportunities to receive a long beacon with updated command information significantly depends on the data transmission interval. Figure 2 analyzes the impact of the network density. For dense networks (i.e., when the transmission power is set to 0 dBm), hop counts are lower while transmission intervals are longer, because shorter paths result in less traffic per node. For sparse networks (transmission power settings of −7 dBm and −15 dBm), hop counts are higher while transmission intervals are shorter, because longer paths result in more traffic per node. Nodes close to the sink with a high traffic load achieve extremely low propagation delays, whereas nodes with high distance to the sink are faced with longer propagation delays. However, the propagation delay versus hop count is lower in sparse networks, showing that shorter transmission intervals make up for the increased path lengths.

Command Execution Confirmation: We now consider the command execution confirmation that is sent whenever a node has received a control message. As this confirmation travels in the usual direction (i.e., where all collected data flows) and represents an individual packet, its collection is considerably faster than the command message propagation in most cases, i.e., the confirmation delay makes up for less than 50% of the round-trip time. We assessed the round-trip time from the time when the sink issues a new command (i.e., updates the routing version number) and finally receives the confirmation.

Due to the low hop count and the high number of potential parents, confirmations are reliably and quickly transported to the sink. However, the confirmation delay dominates the round-trip time in some instances. This is supported by Fig. 3, which portrays the fraction of the round-trip time (RTT) caused by the confirmation. The figure reveals that, when only a single node is addressed at a time, the ratio between confirmation delay and round-trip time is not affected by the hop count, i.e., the distance to the sink.

Command and Confirmation Success Rates: To assess the quality of command dissemination in our approach, we analyzed the success rates of command and confirmation reception. For the former, we calculated the percentage of received command messages (regardless of their destination) per node. In all experiments, the average value (over all nodes) ranges from 97% to 100%. The percentage of command confirmations received at the sink is between 99% and 100%.

IV. Conclusion

Ultra-low power data collection protocols are vital to achieve long operational times of WSNs. However, emitting commands to reconfigure individual nodes is beyond the capabilities of such protocols, making changes to individual node configurations during runtime next to impossible. We have thus introduced a lightweight extension to the ultra-low power ORiNoCo protocol that allows the sink to route control messages to deployed sensors nodes during the network's operation. Despite ORiNoCo's opportunistic nature, testbed experimentation has proven that command dissemination success rates of 97–100% can be achieved. Moreover, the newly designed protocol bears no additional overhead on regular beacon transmissions and only a single field added to each data packet, resulting in a very small additional energy requirement.

References

[1] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," in *Proc. SenSys*, 2003.

[2] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proc. SenSys*, 2009.

[3] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks," in *Proc. NSDI*, 2004.

[4] K. Lin and P. Levis, "Data Discovery and Dissemination with DIP," in *Proc. IPSN*, 2008.

[5] S. Unterschütz, C. Renner, and V. Turau, "Opportunistic, Receiver-Initiated Data-Collection Protocol," in *Proc. EWSN*, 2012.

[6] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, 2012.

[7] S. Duquennoy, O. Landsiedel, and T. Voigt, "Let the Tree Bloom: Scalable Opportunistic Routing with ORPL," in *Proc. SenSys*, 2013.

[8] G. Coulson, B. Porter, I. Chatzigiannakis, C. Koninis, S. Fischer, D. Pfisterer, D. Bimschas, T. Braun, P. Hurni, M. Anwander, G. Wagenknecht, S. P. Fekete, A. Kröller, and T. Baumgartner, "Flexible Experimentation in Wireless Sensor Networks," *Communications of the ACM*, vol. 55, no. 1, 2012.