Routing Over Multiple Technologies with RODENT

Brandon Foubert, Nathalie Mitton

Routing Over Multiple Technologies with RODENT

Brandon Foubert and Nathalie Mitton
Inria - Lille, France
{brandon.foubert, nathalie.mitton}@inria.fr

***Abstract*—Wireless Sensor Networks (WSN) are limited by the characteristics of the Radio Access Technologies (RAT) they are based on. What we refer to as Multiple Technologies Network (MTN) is a network composed of nodes able to use several RAT. The management of the RAT and routes must be handled by the nodes themselves, in a local way, with a suited communication protocols stack. Each stack's layer has to take the technologies' heterogeneity of the devices into account. In this demonstration paper, we show the practical implementation of our custom routing protocol Routing Over Different Existing Network Technologies (RODENT), designed for MTN. It enables dynamic (re)selection of the best route and RAT based on the data type and requirements that may evolve over time. To assess its performance, we have implemented a functional prototype on real WSN hardware, Pycom FiPy devices.**

***Index Terms*—LPWAN, WSN, MTN, RODENT, multi RAT, heterogeneous, routing, multi flow, Pycom FiPy**

I. Introduction

Wireless Sensor Networks (WSN) enable a remote monitoring of various metrics and many more use cases [1]. Such networks usually rely on a medium distance Radio Access Technology (RAT) (*e.g.,* IEEE 802.15.4) and a multi-hop topology. A specific subset of WSN, Low Power Wide Area Networks (LPWAN), usually rely on a long distance RAT (*e.g.,* LoRaWAN) and star topology. When deployed, those kinds of networks use a single RAT shared by all nodes. Deployments are thus constrained by the limits of RAT chosen, in terms of coverage and throughput. Some RAT are even so constrained that they may not be able to comply with specific data requirements such as delay-intolerant data. Additionally, outdoor nodes have to bear the weather changes (*e.g.,* rain) which greatly impact the wireless links' quality.

Multiple Technologies Networks (MTN) can overcome the aforementioned issues [2]. With several RAT built-in, MTN's nodes are able to select the best technology and route available. The choice is based on the routes availability and costs, in terms of energy, money, etc. If the environment changes, and the selected route's quality decreases, a node can select a better route and RAT. Nodes that have several data requirements (*e.g.,* temperature and video monitoring) can use several path accordingly. Network resiliency is increased, as in case of RAT failure, a node can switch with an alternative technology.

Currently available routing protocols are not suited for MTN. Therefore, we designed a novel Routing Over Different Existing Network Technologies protocol (RODENT). In this demonstration paper, we present an MTN prototype composed of Pycom FiPy devices running a custom implementation of RODENT.

II. Protocol overview

RODENT takes as input the list of available links from the stack's link layer. Each link is associated to a neighbor and a RAT. Depending on its RAT, each link has associated costs and performances in terms of energy, speed, etc. Each node selects its best route between its own routes and the ones shared by its neighbors. The selection is made independently of link's RAT and based only on the routes' costs and performances. The definition of best is based on the node's data requirements. The network model and RODENT main functionalities are explained in this section.

A. Network model & assumptions

We assume that the nodes communicate following a convergecast pattern [3], from nodes to sinks. We assume that the network is a connected graph if we consider every link from every node independently of their RAT. This means that every node must own at least one common RAT with one of its neighbors. Nodes can meet several data requirements (*e.g.,* monitoring, alarm, etc.), as long as they are pre-configured on every node. We assume that the low layers' protocols are adapted to MTN. Specifically, the physical and link layers assess the quality of the links available between a node and its neighbors. Our network layer takes a link matrix as input, composed of these information. The nodes transmit data as soon as its available, without Medium Access Control (MAC).

B. MTN building

The devices boot without any knowledge of their surroundings. The link layer scans the environment with every RAT and builds a single link matrix LM composed of the link's cost. Based on LM, the sink nodes' network layer starts to build the route matrix RM. Links from nodes to base stations are registered in RM as single-hop routes. Sink nodes then select their best routes, for each of their data requirements. The selection is made with a custom lighten Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) method [4]. It takes as input the route matrix RM and a data requirements vector RV. We consider monitoring to be the most common use case. The best routes are used to transmit data accordingly to the associated data requirements. Best routes are shared to neighbors trough piggybacking on data packets, or dedicated control packets. Upon reception, neighboring nodes construct their own routes by adding the route's cost to the link's cost it originated from. In turn, nodes select their best routes and advertise them.

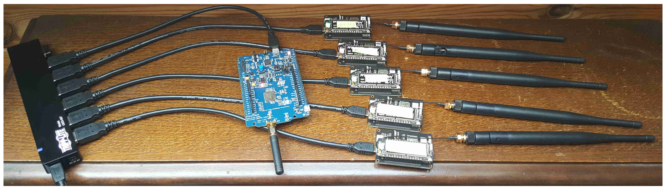

Fig. 1. Experimental setup.

III. Implementation

Our implementation of RODENT is done on Pycom FiPy devices [5]. The specificity of FiPy devices is that they offer five different RAT. These nodes take part in the MTN and offload data to WiFi and LoRa base-stations (BS). The hardware and firmware used are detailed in this section.

A. Hardware

Pycom FiPy nodes are composed of WSN hardware: wireless RAT, ESP32 CPU, few memory available which allows ultra–low power usage. The available RAT are WiFi, LoRa, Sigfox, LTE-M, NB-IoT and Bluetooth Low Energy (BLE). Each RAT comes with different performances in terms of energy consumption, economical cost, bit-rate, etc. RODENT performs route selection based on these characteristics. FiPy are coupled with Pytrack sensor shields which provide an accelerometer, a GPS and a micro-USB port.

The LoRa BS is a B-L072Z-LRWAN1 board [6]. The WiFi BS is an Edimax EW-7811Un dongle [7] connected to the main computer. A Trip Lite U223-007 (7-Port USB Hub) is used to connect every devices. The main computer is a Dell Latitude 5590. It powers devices, collects and analyses results.

B. Firmware

A port of MicroPython available as firmware for the FiPy allowed us to implement RODENT in Python. Upon boot, a node compute its unique ID. Based on its LM it boots up the needed RAT and constructs routes. The node is then locked up in the main loop: *i)* select best route for each RV, *ii)* add next payload to transmission buffer *iii)* send every payload in buffer. Neighbor's route are added in RM upon reception. Neighbor's payload are appended in the transmission buffer. Nodes print on the serial port the characteristics of packet sent. Upon Pytrack's button press, nodes switch between the two RV implemented: monitoring and alarm.

The LoRa BS's firmware is implemented in C. It listens constantly for LoRa transmissions. Upon reception of a RODENT packet, it is unpacked and its characteristics are printed on the serial port. The WiFi BS is coded in Python. It listens for RODENT WiFi transmissions, unpacks them and prints characteristics on stdout.

IV. Experiments

To assess the performances of RODENT, we run experiments on real hardware. We configured the nodes to follow a specific scenario and measured the results. The experimental setup and scenario are presented in this section.

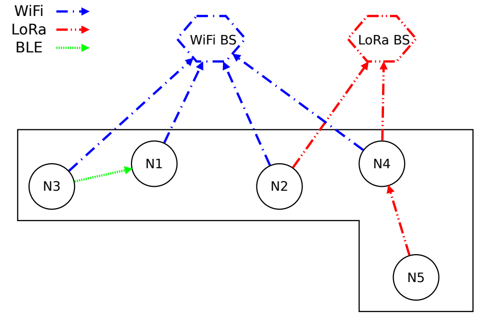

Fig. 2. Farm monitoring scenario.

A. Setup

All devices are connected to the main computer through the USB hub. Every node and BS are powered at the same time and boot up immediately. As visible in Figure 1, every device is laying very close to each other. The main computer reads the stdout of the WiFi BS and the serial ports of the nodes and LoRa BS. Results are then computed off-line, post-experiment.

B. Scenario

We simulate a farm monitoring use case. Nodes collect accurate parcel-specific weather-related data, such as temperature, rainfall, etc. This helps farmers accurately forecast the weather (*e.g.,* risk of frost) and make sound decisions for the crop cycle (requirements for watering, fertiliser, etc). In our scenario, five nodes are deployed throughout a field. The simulated setup is illustrated in Figure 2. Nodes have to offload numerical data on a regular basis while saving up power. They may have to send an alarm if a metric becomes off chart, putting the crops at risk (*e.g.,* temperature).

Out of the five FiPy's RAT, we are using WiFi, LoRa and BLE in this scenario. Sigfox and LTE-M/NB-IoT are not open technologies, so we could not use them directly. LoRa and BLE links are more interesting in terms of energetic savings than WiFi. Each node (Nx) is in a different situation. $N1$ is the control node, it only has a WiFi link with the WiFi BS. $N2$ can reach the WiFi BS and benefits from the LoRa link when RODENT is active. $N3$ have to choose between reaching the WiFi BS directly at a high energy cost or forwarding its data to its neighbor $N1$ via a BLE link. $N4$ needs to be able to send regular monitoring data as well as alarms, via WiFi or LoRa. $N5$ is an isolated node, deployed too far away to directly communicate with the WiFi BS. Farms are usually located in wide rural environments, unfriendly to wireless waves because of tall crops (*e.g.,* corn). Thus isolated nodes are common. Using RODENT, $N5$ can forward its data to $N4$ using LoRa.

We run three types of experiments. First, RODENT is inactive and nodes only use WiFi links, depicted in blue in Figure 2. Second, RODENT is active, allowing nodes to switch to LoRa and BLE links, depicted in red and green in Figure 2. Third, RODENT is active, LoRa transmission are doubled and BLE transmission tripled, which increase the network's reliability. An experiment run is available on video [8].

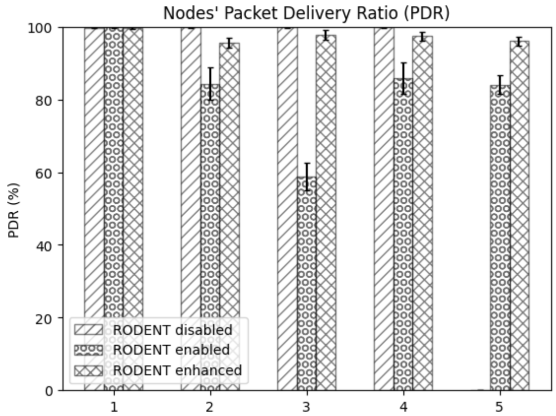

Fig. 3. Packet Delivery Ratio per node.

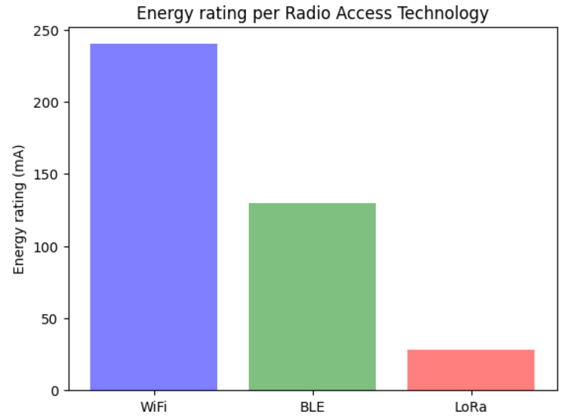

Fig. 4. Energy consumption per RAT.

V. Results

Topology, Packet Delivery Ratio (PDR) and energy consumption are measured. Nodes transmit at an interval randomly picked in $[2;4]$ seconds. We consider a population of 20 experiments for 10 minutes each. Small population is sufficient because of the low standard deviation. Longer experiments are not relevant because the network stabilizes after few messages exchanged. In this section we present the results obtained.

A. Topology

With the use of RODENT, the topology changes. $N1$ does not change its link because it can only reach the WiFi BS. $N2$ uses the LoRa link instead of the WiFi link, because it cost less energy. $N3$ decides to use the BLE link to offload its data to $N1$, which in turn forwards it to the WiFi BS. $N4$ offload its monitoring data to the LoRa BS, to reduce energy consumption compared to WiFi. It can still use the WiFi link to forward alarms, that needs a quicker RAT at the expense of higher energy cost. $N5$ is not isolated anymore, as it forwards its data to $N4$ through LoRa which will offload it to the LoRa BS in turn.

B. Packet Delivery Ratio

The Packet Delivery Ratio (PDR) is the ratio between the total packets received and the total packets sent. The PDR of every node taking part in the MTN is depicted in Figure 3. $N1$'s PDR does not changes, as its route remains the same. Without RODENT, $N5$'s PDR is null as the node is isolated and cannot offload a single data packet. The PDR of $N2$, $N4$ and $N5$ is around 80% with RODENT which allows them to use LoRa. It is not the same as WiFi because of collisions as nodes does not use a proper MAC. $N3$'s PDR is around 60% with RODENT. The node forwards its data through BLE to $N1$. We achieved BLE raw transmissions through the use of single BLE advertisements, hence the packet loses. With the enhanced RODENT, we can see a better PDR for all nodes, close to the one obtained with only WiFi.

C. Energy consumption

Physical measurement of the Pycom FiPy's energy consumption is hazardous since it suffers from design problems which lead to erroneous measurements [9]. We choose to stick to the energy ratings given in the components data-sheets [5], [10] to get a general idea, which are showed in Figure 4. Compared to WiFi, BLE needs approximately half-less current and LoRa a tenth. With the Pycom FiPy's CPU, WiFi and BLE offers the same bit-rate. LoRa's bit-rate is much slower leading to longer transmission for a same amount of data. WiFi and BLE require a heavier traffic control than LoRa does, which allows LoRa to consume less energy. Thus, we can assume that RODENT enable significant energy savings.

VI. Conclusion

In this work we demonstrate the feasibility and utility of MTN with a prototype based on a custom implementation of the RODENT protocol. It increases network flexibility and reliability, decreases energy consumption and maintains a good PDR. Future work includes accurate energy consumption assessment and support for downlink communication.

Acknowledgments

This work was partially supported by a grant from CPER DATA, Sencrop, FEDER, I-SITE and Lirima Agrinet project.

References

[1] D. Ye, D. Gong, and W. Wang, "Application of wireless sensor networks in environmental monitoring," in PEITS, 2009.
[2] B. Foubert and N. Mitton, "Autonomous Collaborative Wireless Weather Stations: A Helping Hand for Farmers," ERCIM News, no. 119, 2019.
[3] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey," IEEE Communications Surveys Tutorials, vol. 15, no. 2, 2013.
[4] B. Foubert and N. Mitton, "Sélection d'interface de communication dans les réseaux de capteurs multi-technologies," in CoRes, 2020.
[5] FiPy Specsheet, Pycom, 2018, version 1.0.
[6] UM2115 User manual, STMicroelectronics, 2018, rev 5.
[7] EW-7811Un Datasheet, Edimax, 2015.
[8] "Rodent experiment recording," http://chercheurs.lille.inria.fr/bfoubert/ressources/rodent.mp4, [Online; accessed 27-Nov-2020].
[9] T. Andersén, "Energy-efficient adaptive sensing in low power wide area networks," Master's thesis, NTNU, 2018.
[10] ESP32 Series Datasheet, Espressif Systems, 2020, version 3.3.