Energy-Efficient Radio Selection and Data Partitioning for Real-Time Data Transfer

Di Mu, Mo Sha, Kyoung-Don Kang, Hyungdae Yi
Department of Computer Science
State University of New York at Binghamton
{dmu1, msha, kang, hyi2}@binghamton.edu

***Abstract*—The importance of real-time wireless data transfer is rapidly increasing for Internet of Things (IoT) applications. For example, smart glasses worn by a doctor need to transmit real-time data to a hospital information system, which performs face detection and recognition, for real-time interaction with recognized patients within a certain deadline, which is ideally a few hundred milliseconds. Other emerging IoT applications, e.g., structural health monitoring, clinical monitoring, and industrial process automation, also require real-time wireless data transfer. Those applications have critical demands for real-time and energy-efficient communication through wireless medium. However, it is very challenging to support stringent timing constraints energy-efficiently through wireless medium due to its inherent unreliability and timing-unpredictability. Fortunately, heterogeneous radios are becoming increasingly available in modern embedded devices, offering new opportunities to use multiple wireless technologies to accommodate the needs of real-time applications. In this paper, we first formulate the runtime radio selection and data partitioning for real-time IoT applications as an Integer Linear Programming (ILP) problem and then present (1) an *optimal* algorithm that makes quick and optimal decisions when selecting between two radios and (2) a *heuristic* algorithm for the platforms with more radios. Experimental results show that our heuristic algorithm provides optimal selections to 94.4% of the cases and makes the decisions 336∼1412 times faster than an ILP problem solver.**

***Keywords*-Real-Time Data Transfer, Radio Selection, Data Partitioning, Energy Efficiency, IoT**

I. Introduction

The importance of real-time wireless data transfer is rapidly increasing for the Internet of Things (IoT) applications. For example, smart glasses worn by a doctor need to transmit real-time data to a hospital information system, which performs face detection and recognition, for real-time interaction with recognized patients within a certain deadline, which is ideally a few hundred milliseconds [1]. As another example, periodic sensor readings from unmanned aerial vehicles (UAVs) should be delivered every second to a georeferencing system that analyzes the data to determine the real-time position and altitude of UAVs [2]. Other emerging IoT applications, e.g., structural health monitoring [3], clinical monitoring [4], and industrial process automation [5], [6], also require real-time wireless data transfer. In such applications, missing data delivery deadlines may result in cognitive distraction, injury, structural damage, or safety hazard. However, it is very challenging to support stringent timing constraints through wireless medium due to its inherent unreliability and timing-unpredictability. Moreover, the energy constraints significantly amplify the challenge, since most of those IoT devices are battery-powered and achieving high energy efficiency is critical for those applications.

Fortunately, embedded system hardware and radio technologies are advancing fast in recent years. As a result, more and more embedded devices are equipped with heterogeneous radios. For example, Firestorm [7] supports ZigBee and Bluetooth Low Energy (BLE) in one device and TI CC2650 [8] integrates those two radios on a single chip. IOT-Gate-iMX7 [9] is an industrial IoT gateway, which supports 4G/LTE, WiFi, Bluetooth, and Zigbee. LX Cellular Core [10] is a small-sized IoT platform, which features 2G/3G, WiFi, BLE, ANT+, LoRa, Taggle, and SigFox. Heterogeneous radios are becoming increasingly available in modern embedded devices, offering new opportunities to use multiple wireless technologies for real-time applications. However, using multiple heterogeneous radios may enhance the timeliness at the expense of higher energy consumption or vice versa. It is even more challenging to strike a good balance between the two potentially conflicting requirements.

This paper aims to address the previously stated challenges and presents an energy-efficient radio switching and bundling solution to minimize the energy consumption of battery-powered IoT devices[1] for real-time applications, leveraging the above-stated hardware advancements. To assure the timeliness, we target at a single-hop application scenario, since most existing solutions relying on multi-hop mesh networks suffer from long latency and high complexity. Our approach conforms to the advanced wireless network technology trend as the industry is investing heavily in network infrastructure to support IoT visions such as smart cities. As a result, more and more access points and edge servers are becoming readily available to support various IoT applications. Specifically, this paper makes the following contributions:

- We formulate the runtime radio switching and bundling as an Integer Linear Programming (ILP) problem;
- We design the *Real-Time radio Selection (RT-Select)* algorithm that optimally and quickly selects between two radios and partitions data between them at runtime to minimize the energy consumption;

[1]In this paper, we focus on minimizing the energy consumption on the sender side (IoT end devices), since the IoT gateways are usually not or much less energy-constrained.

- Based on RT-Select, we design the *RT-Select-General* algorithm for the platforms with more radios.
- We develop the *Real-time Radio Switching and Bundling (RRaSB)* system that runs on our embedded platform equipped with five heterogeneous radios, selectively makes a subset of radios available at runtime, and allows dynamic radio switching and bundling among them;
- We implement RT-Select and RT-Select-General in RRaSB and evaluate them experimentally; these efforts demonstrate the unique benefits of runtime radio switching and bundling.

The remainder of the paper is organized as follows. Section II introduces our problem formulation. Section III presents the design of RT-Select and RT-Select-General. Section IV describes RRaSB. Section V presents our experimental evaluation. Section VI reviews related work and Section VII concludes the paper.

II. Problem Formulation

A. Optimization Formulation

In this section, we formulate the runtime radio selection and data partitioning for real-time applications as an ILP problem. We first introduce some related radio characteristics and then define the objective function and constraints of the ILP problem.

We assume that m radios, $R_1, ..., R_m$, are available on an IoT end device. The characteristics of each radio $R_i (1 \leq i \leq m)$ are separated into two categories:

1) variable characteristics related to the bandwidth and reliability of the wireless link between R_i and the IoT gateway:
   - throughput, TH_i, is the maximum number of data packets which R_i is able to successfully deliver to the IoT gateway per second;
   - expected transmission count, ETX_i, is the average number of transmission(s) which R_i needs to attempt to successfully deliver a packet to the IoT gateway.
2) constant characteristics related to energy and time consumption of R_i:
   - switching energy, E_{sw_i}, is the total energy consumed to switch R_i on and off[2].;
   - switching time, T_{sw_i}, is the time taken to switch R_i on[3];
   - radio base power, P_{rb_i}, is the base power consumed by R_i when the radio is on and idle;
   - per-transmission energy, E_{ta_i}, stands for the additional energy consumed by R_i for each packet transmission attempt.

[2] R_i is turned off by default after it transmits all assigned packets if the future traffic demand is unknown.

[3] The time taken to switch R_i off is not included since the radio can be turned off after the deadline if it is not selected for use in the next period.

Our optimization goal is to minimize the radio energy consumption while meeting the data rate and deadline requirement. We define the deadline miss rate as the number of data transfers which are not completed before their deadlines divided by the total number of data transfers. Please note that we minimize the deadline miss rate instead of the absolute latency since the deadline miss rate is a more direct metric reflecting the performance of real-time applications. To achieve the objective, we select the radio(s) and assign the data packets to them. We assume that there are N packets required to be delivered by deadline D. Let us also assume that X_i packets are assigned to radio R_i, where $0 < X_i \leq N$ if R_i is selected or $X_i = 0$ if R_i is not selected. The objective function to minimize is the sender's energy consumption E, which is the sum of the radio switching energy, radio base energy, and radio transmission energy consumed by the selected radios as shown in Eq. 1, where the radio base energy is P_{rb_i} multiplied by the transmission time (X_i/TH_i), the radio transmission energy is E_{ta_i} multiplied by ETX_i and X_i, and the set S is composed of the indices of all selected radios:

$$E = \sum_{i \in S} \left(E_{sw\_i} + P_{rb\_i} \times \frac{X_i}{TH_i} + E_{ta\_i} \times ETX_i \times X_i\right) \quad (1)$$

There are three constraints on variable X_i (the number of packets assigned to R_i): (i) X_i is a non-negative integer not greater than N as specified in Eq. 2 (ii) X_i should not exceed the maximum packet delivery capacity of the radio link (X_{max_i}) for the deadline D as stated in Eq. 3 and (iii) the total number of packets assigned to all radios should be equal to N as specified in Eq. 4. Therefore, the following constraints should be met to satisfy the traffic demand and deadline requirements:

$$0 \leq X_i \leq N \ (X_i \in \mathbb{N}) \quad (2)$$

$$X_i \leq X_{max\_i} \equiv (D - T_{sw\_i}) \times TH_i \quad (3)$$

$$\sum_{i=1}^{m} X_i = N \quad (4)$$

In addition, let us introduce a Boolean variable, Y_i, to indicate whether or not the radio R_i is selected. $Y_i = 1$ if R_i is selected ($X_i > 0$) and $Y_i = 0$ if R_i is not selected ($X_i = 0$).

Given Eq. 2–4, we simplify the objective function E in terms of variables X_i and Y_i as well as coefficients A_i and B_i as follows:

$$\boldsymbol{min} \left(\sum_{i=1}^{m} \left[A_i Y_i + B_i X_i\right]\right) \quad (5)$$

where

$$\begin{aligned} A_i &= E_{sw\_i} \\ B_i &= \frac{P_{rb\_i}}{TH_i} + E_{ta\_i} \times ETX_i \end{aligned} \quad (6)$$

Eq. 2–6 form an ILP problem, which is NP-hard.

Many resource-constrained IoT devices cannot afford to execute an ILP solver to solve the problem at runtime for real-time applications. This motivates us to develop lightweight

algorithms tailored for the runtime radio selection and data partitioning problem.

III. Algorithm Design

One of the primary design goals of our algorithms is to be time-efficient. With the consideration of the demand of fast responses, our decision-making strategies can be fast processed by the IoT devices to guide the runtime radio selection and data partitioning in response to the current wireless link state and application timing requirement. Specifically, we first design the *RT-Select* algorithm that optimally solves the two-radio case of the problem and prove its optimality. Then, based on the insights from the design of RT-Select, we design the *RT-Select-General* algorithm to solve the general form of the problem involving m radios. Both of our algorithms take the inputs of the traffic demand (i.e., N packets) and the delivery deadline D specified by the application and the pre-measured radio characteristics and output the radio selection decision. For simplicity, we use RC_i to represent the characteristics of each radio R_i including $TH_i, ETX_i, E_{sw\_i}, T_{sw\_i}, P_{rb\_i}$ and E_{ta_i} (see Section II).

Please note that an embedded device may not allow to use some of its radios simultaneously due to hardware conflicts. For example, the ZigBee and BLE radios on the TI CC2650 [8] cannot operate simultaneously, since they share a single DSP modem and a digital PLL. The hardware conflicts are added into our algorithms as constraints.

A. RT-Select Algorithm for Selection between Two Radios

Algorithm 1: RT-Select

Input : N, D, RC_1, RC_2
Output: X_1, X_2

1 *Compute* A_1, B_1, X_{max_1}, A_2, B_2, and X_{max_2} ;
2 $(idx\_1, idx\_2) = \boldsymbol{sort}\{A_i + B_i \times N \mid i = 1, 2\}$;
3 $(idx\_1', idx\_2') = \boldsymbol{sort}\{B_i \mid i = 1, 2\}$;
4 **if** $X_{max\_(idx\_1)} \geq N$ **then**
5 $\quad X_{idx\_1} \leftarrow N$;
6 **else if** $X_{max\_(idx\_1)} < N$ **and** $X_{max\_(idx\_2)} < N$ **then**
7 $\quad X_{idx\_1'} \leftarrow X_{max\_(idx\_1')}$;
8 $\quad X_{idx\_2'} \leftarrow N - X_{idx\_1'}$;
9 **else**
10 $\quad$ **if** $B_{idx\_2} \leq B_{idx\_1}$ **or** $A_{idx\_1'}/(B_{idx\_2} - B_{idx\_1'}) > X_{max\_(idx\_1')}$ **then**
11 $\quad\quad X_{idx\_2} \leftarrow N$;
12 $\quad$ **else**
13 $\quad\quad X_{idx\_1'} \leftarrow X_{max\_(idx\_1')}$;
14 $\quad\quad X_{idx\_2} \leftarrow N - X_{idx\_1'}$;
15 $\quad$ **end**
16 **end**

Algorithm 1 shows RT-Select algorithm that selects between two radios to minimize the energy consumption, while meeting the application specified traffic demand and deadline requirements. RT-Select first computes the A_i, B_i, and X_{max_i} values for both radios based on Eq. 6 and Eq. 3 (line 1). It then sorts the two radios based on the energy consumption for each radio to transmit N packets by itself $(A_i + B_i \times N)$ and stores the radio indices to (idx_1, idx_2) in ascending order. Therefore, the radio R_{idx_1} is more energy-efficient than R_{idx_2}. Similarly, RT-Select sorts the two radios based on the average energy consumption per packet B_i without considering radio switching energy consumption A_i and stores the radio indices to (idx_1', idx_2') in ascending order. Therefore, the radio $R_{idx\_1'}$ is more energy-efficient than $R_{idx\_2'}$ without considering radio switching energy consumption A_i. Finally, RT-Select makes radio selection decisions based on three different cases:

1) if the more energy-efficient radio R_{idx_1} can deliver all packets before the deadline by itself, RT-Select uses it alone and assigns all N packets to it (line 4-5).
2) if none of the radios can deliver all packets before the deadline by itself, RT-Select has to use both radios. Therefore, RT-Select assigns $X_{max\_(idx\_1')}$ packets to $R_{idx\_1'}$ and the rest to the other radio (line 6-8).
3) if only the less energy-efficient radio R_{idx_2} can deliver all packets before the deadline, RT-Select needs to decide whether to use it alone or use both radios. In case R_{idx_2} has the smaller B_i of the two radios or $X_{max\_(idx\_1')}$ is smaller than $A_{idx\_1'}/(B_{idx\_2} - B_{idx\_1'})$ [4], RT-Select uses the less energy-efficient radio R_{idx_2} alone and assigns all packets to it (line 10-11). Otherwise, RT-Select selects both radios and assigns $X_{max\_(idx\_1')}$ packets to $R_{idx\_1'}$ and the rest to the other radio (line 12-14).

The proof of optimality can be found in Appendix A.

B. RT-Select-General for Selection among Multiple Radios

Based on the insights collected during our algorithm design for the two-radio special case, we design RT-Select-General that solves the general form of the problem involving m radios. As shown in Algorithm 2, RT-Select-General first computes the A_i, B_i, and X_{max_i} values for all m radios in line 1. Similar to RT-Select, RT-Select-General sorts all m radios based on the energy consumption to transmit N packets for each single radio $(A_i + B_i \times N)$ and stores the sorted radio indices to $(idx\_1, ..., idx\_m)$ in ascending order (line 2). RT-Select-General sorts all radios again based on the average energy consumption per packet B_i without considering radio switching energy consumption A_i and stores the radio indices to $(idx\_1', ..., idx\_m')$ in ascending order (line 3).

RT-Select-General makes radio selection decisions based on three cases similar to RT-Select:

1) if the most energy-efficient radio R_{idx_1} can deliver all packets before the deadline by itself, RT-Select-General uses it alone and assigns all N packets to it (line 4-5).

[4]This comparison decides whether it consumes less energy to use the less energy-efficient radio alone. The equation comes from the optimality proof in Appendix A.

Algorithm 2: RT-Select-General

Input : $N, D, RC_1, RC_2, ..., RC_m$
Output: $X_1, X_2, ..., X_m$

1 *Compute* $\{A_i, B_i, X_{max\_i} \mid i = 1, ..., m\}$;
2 $(idx\_1, ..., idx\_m) = \boldsymbol{sort}\{A_i + B_i \times N \mid i = 1, ..., m\}$;
3 $(idx\_1', ..., idx\_m') = \boldsymbol{sort}\{B_i \mid i = 1, ..., m\}$;
4 **if** $X_{max\_(idx\_1)} \geq N$ **then**
5 $\quad X_{idx\_1} \leftarrow N$;
6 **else if** $\boldsymbol{max}\{X_{max\_(idx\_i)} \mid i = 1, ..., m\} < N$ **then**
7 $\quad$ **for** $i = 1$ **to** m **do**
8 $\quad\quad$ **if** $X_{max\_(idx\_i')} < N - \boldsymbol{sum}\{X_{idx\_k} \mid k < i\}$ **then**
9 $\quad\quad\quad X_{idx\_i'} \leftarrow X_{max\_(idx\_i')}$;
10 $\quad\quad$ **else**
11 $\quad\quad\quad X_{idx\_i'} \leftarrow N - \boldsymbol{sum}\{X_{idx\_k} \mid k < i\}$;
12 $\quad\quad\quad$ **break**;
13 $\quad\quad$ **end**
14 $\quad$ **end**
15 **else**
16 $\quad$ **for** $i = 2$ **to** m **do**
17 $\quad\quad$ **if** $X_{max\_(idx\_i)} < N$ **then**
18 $\quad\quad\quad$ **continue**;
19 $\quad\quad$ **end**
20 $\quad\quad$ **if** $B_{idx\_i} = \boldsymbol{min}\{B_i \mid i = 1, ..., m\}$ **or** $A_{idx\_1'}/(B_{idx\_i} - B_{idx\_1'}) > X_{max\_(idx\_1')}$ **then**
21 $\quad\quad\quad X_{idx\_i} \leftarrow N$;
22 $\quad\quad$ **else**
23 $\quad\quad\quad X_{idx\_1'} \leftarrow X_{max\_(idx\_1')}$;
24 $\quad\quad\quad X_{idx\_i} \leftarrow N - X_{idx\_1'}$;
25 $\quad\quad$ **end**
26 $\quad$ **end**
27 **end**

2) if none of the radios can deliver all packets before the deadline by itself, RT-Select-General has to use multiple radios. Similar to RT-Select, RT-Select-General prefers to use the radios with small B_is, thus it selects the radios one by one based on the sorted indices $(idx\_1', ..., idx\_m')$ and lets them transmit with their maximum capacity until the selected radios can deliver all N packets before the deadline (lines 6-14).
3) if there exists a radio R_{idx_i} which can deliver all packets before the deadline by itself but is not the most energy-efficient one $(i > 1)$, then RT-Select-General needs to decide whether to use it alone or combine it with another radio[5]. Inspired by Algorithm 1, if R_{idx_i} has the smallest B_i of all radios or $X_{max\_(idx\_1')}$ is smaller than $A_{idx\_1'}/(B_{idx\_i} - B_{idx\_1'})$, RT-Select-General selects R_{idx_i} only and assigns all packets to it. Otherwise, RT-Select-General combines R_{idx_i} with $R_{idx\_1'}$ (the one with the smallest B_i) and let $R_{idx\_1'}$ transmit with its maximum capacity and assigns the rest packets to R_{idx_i} (line 16-26).

[5]We select at most two radios in this case in consideration of designing a light-weight algorithm for runtime use.

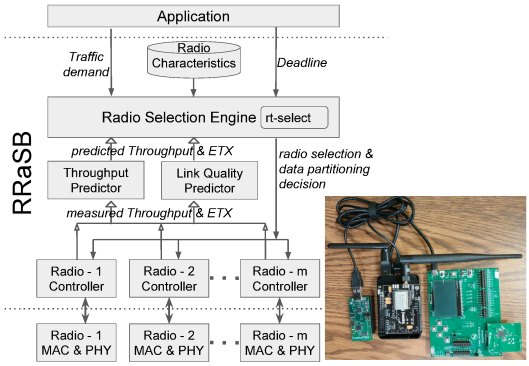


Fig. 1. System architecture and the platform supporting five radios.

The constraints reflecting the hardware conflicts can be added into case 2) and case 3) of Algorithm 1 and Algorithm 2. RT-Select-General behaves identically to RT-Select when $m = 2$, making the latter a special case providing optimal selections. The time complexity of RT-Select-General is $O(m \log m)$ (dominated by the complexity of sorting), which is acceptable to support real-time decision-making since m is not expected to be very large in practice ($m \leq 16$ today to our knowledge). Our experimental results show that RT-Select-General provides optimal selections to 94.4% of the cases (See Section V-E) and makes the decisions 336∼1412 times faster than an ILP problem solver (See Section V-A).

IV. System Design and Implementation

To realize our designs, we develop the RRaSB system that makes multiple radios available at runtime and allows dynamic radio switching and bundling among them. Figure 1 shows the system architecture. The radio characteristics including energy consumption of radio switching (E_{sw}), radio switching time (T_{sw}), power consumption when the radio is idle (P_{rb}), and average energy consumption per transmission attempt (E_{ta}) are measured offline and stored in the **Radio Characteristics** component, serving as inputs to the radio selection algorithm. The **Throughput Predictor** predicts the throughput in the next period based on the historical data and the **Link Quality Predictor** estimates the expected transmission counts (ETX) in the next period based on previous ETX measurements using the Holt-Winters method that is one of the most effective time series forecasting algorithms [11]. If a radio has not been used for a long time, Link Quality Predictor transmits some probing packets through it to keep its link quality measurements updated. The **Radio Selection Engine** takes radio characteristics, estimated throughput and ETX, and traffic demand and deadline specified by the application as inputs and runs the radio selection algorithm to select the radio(s) that is/are best suited for the current network traffic and operating conditions and then assigns packets accordingly. Multiple **Radio Controller** modules exist in RRaSB. Each Radio Controller controls the on/off state of a radio based on the decision made by the Radio Selection Engine and measures the actual throughput and ETX fed into the Throughput

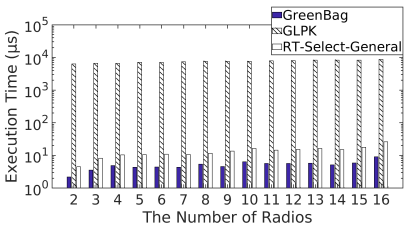
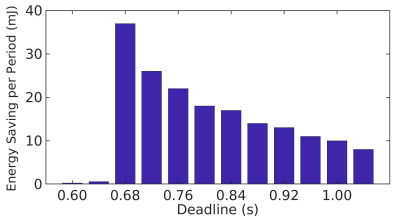
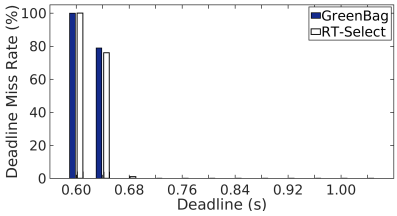


Fig. 2. Execution time of RT-Select-General compared with GreenBag and GLPK.



(a) Energy saving of RT-Select over GreenBag.



(b) Deadline miss rates.

Fig. 3. Performance under RT-Select and GreenBag with two radios when the application transmits at a fixed data rate with different deadlines.

Predictor and Link Quality Predictor, respectively. RRaSB is configured to perform the radio selection in each period based on the measured throughput and ETX of the radio links as well as the traffic demand and deadline specified by the benchmark application. If the current radio selection is found to be the best-suited, it is retained; otherwise, our system switches to a new best-suited setting. Radios are turned off after the last transmission in each period if they are not selected for use in the next period and the unselected ones are kept off to reduce energy consumption. If multiple transmitters exist, they access the channel in a TDMA fashion. We have implemented RRaSB in Raspbian Linux [12] and Contiki [13] and two prototypes: one with two radios and the other with five radios. A power monitor from Monsoon Solutions [14] is connected to the sender to measure the energy consumption. More implementation details can be found in Appendix B.

V. Evaluation

To examine the efficacy of our radio selection and traffic partitioning solution, we perform a series of experiments on our embedded platform presented in Section IV. We start by demonstrating the time efficiency of RT-Select-General and the effectiveness of the throughput and link quality predictors. We then run experiments to measure the radio energy consumption and deadline miss rates with our prototype hosting two radios and repeat the experiments with five radios. Finally, we perform a simulation study in which we demonstrate the effectiveness and benefits of our radio selection and bundling approach for real-time IoT applications at various combinations of traffic demand and deadline. We compare our approaches against two baselines: GreenBag (GB-E) [15] (a practical state-of-the-art radio selection approach designed for real-time applications to reduce energy consumption) and GLPK (GNU Linear Programming Kit) [16] providing optimal results to the ILP problems. Please note that GLPK cannot be used for real-time applications with short deadlines because of its heavy computation overhead as presented in Section V-A. We run GLPK offline and exclude its energy consumption in the results of optimal solutions (Figure 6(a), 7(a), and 8).

In all experiments, we deploy two real-time benchmark applications on top of our system which generate data packets periodically. The first benchmark application (benchmark application A) emulates a health care scenario where doctors use smart glasses to take ambient pictures or videos of patients and send them to the hospital information system for real-time face detection and recognition [1]. In this application, a fixed traffic demand is employed by the smart glasses but the application may specify different deadlines based on its quality of service (QoS) needs. The second benchmark application (benchmark application B) emulates a real-time georeferencing scenario where UAVs capture images of the land from the air and transmit them together with GPS locations to a ground station [2]. In this application, a fixed deadline (e.g., 1 second) of image delivery is adopted by the UAVs to ensure the accuracy of the real-time location but the traffic demand (image size) may vary to meet different needs. Both benchmark applications generate periodic traffic whose deadline is equal to its period. The two benchmark applications allow us to examine the performance of our system (i) at a fixed data rate with different data delivery deadlines and (ii) at various data rates with a fixed deadline.

A. Time Efficiency of RT-Select-General

We first measure the execution time of RT-Select-General and two baseline approaches (GreenBag and GLPK) on the Raspberry Pi 3 with a 1.2 GHz 64-bit quad-core ARMv8 CPU. We measure the time duration between feeding the input into the Radio Selection Engine and receiving the output from it. We repeat the experiments 20 times using random inputs for each m (the number of radios). Figure 2 shows the average execution time of GreenBag, GLPK and RT-Select-General for different number of radios (m ranging from 2 to 16) in the logarithmic scale. As Figure 2 shows, the average execution time of RT-Select-General increases from 4μs to 26μs when m increases from 2 to 16, which is slightly (2$\sim$17μs) longer than what GreenBag uses. The average execution time of GLPK ranges from 6267μs to 8670μs, which is 336$\sim$1412 times longer than what RT-Select-General consumes. Therefore, it is not feasible to use the time-consuming GLPK to support the real-time applications with short deadlines, especially when running on the platforms with limited harware resources. As a comparison, our RT-Select-General can time-efficiently make decisions achieving performance close to what GLPK offers (see Section V-D and Section V-E).

B. Effectiveness of Link Condition Predictors

We then perform a set of controlled experiment to evaluate the effectiveness of our Throughput Predictor and Link Quality Predictor employing the Holt-Winters method. In this set of experiments, we measure the throughput and ETX of radio links under controlled interference and compare them against the predicted values. Figure 4 plots the example traces showing the throughput and ETX changes of a WiFi link when

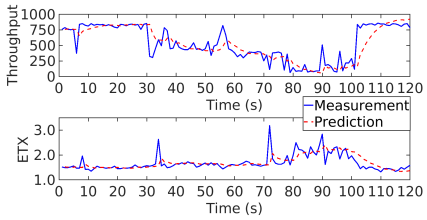
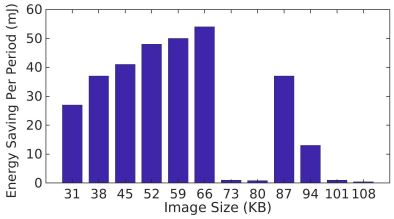
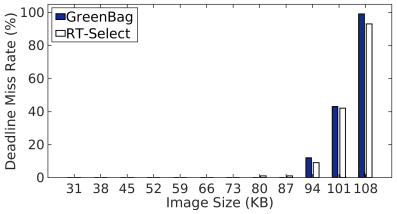


Fig. 4. Throughput and ETX predictions vs. ground truth in a 120-second WiFi link condition trace.



(a) Energy saving of RT-Select over GreenBag.



(b) Deadline miss rates.

Fig. 5. Performance under RT-Select and GreenBag with two radios when the application transmits at different data rates with the same deadline.

encountering the controlled interference. An interferer begins the transmission in the same channel from the 31st second to the 100th second. As Figure 4 shows, the predictions are very close to the measurements during the process. The standard deviation on the throughput difference is 152 packets/s and 80% of the prediction errors are less than 125 packets/s. The standard deviation on the ETX difference is 0.25 and 80% of the prediction errors are less than 0.2.

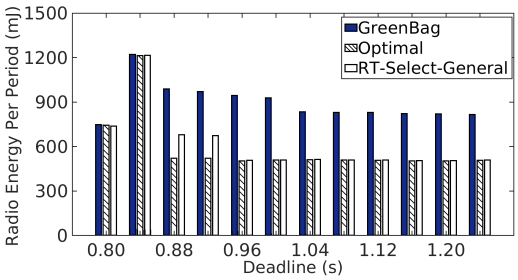
C. Experiments with Two Radios

We run experiments on our prototype hosting two radios (i.e., the CC2650 ZigBee radio and the RT5370 WiFi radio) to evaluate the effectiveness of RT-Select and its impact on radio energy consumption and real-time performance. Since the output of RT-Select is proved to be optimal in Section A, we only compare RT-Select against GreenBag in this set of experiments.

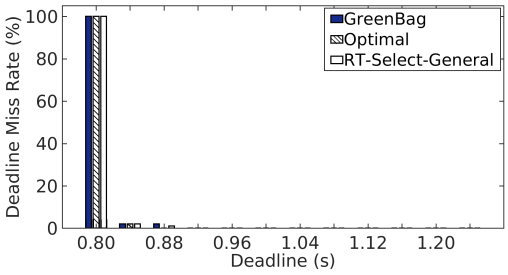
We configure the benchmark application A to transmit a 23KB image (480×480 JPEG) in every period and repeat the experiments with 12 different deadlines ranging from $0.60s$ to $1.04s$ according to the response time of Amazon face recognition applications [17]. Figure 3(a) shows the energy saving of RT-Select over GreenBag per period and Figure 3(b) plots the deadline miss rates. RT-Select shows significant energy saving (ranging from $8mJ$ to $37mJ$[6]) when the deadline is greater than $0.64s$ with the deadline miss rates no higher than 1%. The energy savings benefit from RT-Select's decision on keeping only the WiFi radio active rather than using both radios suggested by GreenBag. High deadline miss rates are observed under both RT-Select and GreenBag when the deadline is shorter than $0.68s$, not enough to turn on the WiFi radio or send all packets using the ZigBee radio. The results show that RT-Select consistently outperforms GreenBag under various deadlines.

Similarly, we configure the benchmark application B to transmit a JPEG image with the fixed deadline ($0.80s$) in every period, and repeat the experiments with 12 image sizes ranging from 31KB (640×480 JPEG) to 108KB (1280×720 JPEG). As Figure 5(a) and Figure 5(b) show, RT-Select consumes $27\sim54mJ$ less energy compared to GreenBag without missing any deadline when the image size is between 31KB and 66KB. The energy savings benefit from RT-Select's decision on keeping only the WiFi radio active rather than using both

[6]As a comparison for energy saving values, the CC2650 radio consumes $30mW$ power when transmitting at $5dBm$ [8].



(a) Comparison on radio energy consumption.



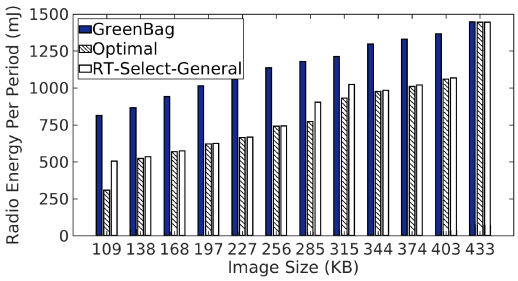
(b) Comparison on deadline miss rates.

Fig. 6. Performance of GreenBag, Optimal and RT-Select-General solutions with five radios when the application transmits at a fixed data rate with different deadlines.

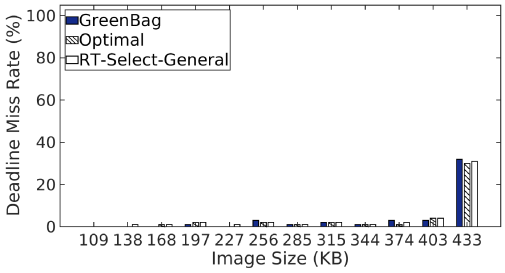
radios suggested by GreenBag. The energy saving is marginal when the image size is 73KB or 80KB. This is because both RT-Select and GreenBag decide to use only the WiFi radio when it becomes the more energy-efficient radio under high traffic demand and can deliver all data packets by the deadline. When the image size is 87KB, both RT-Select and GreenBag suggest using both radios. However, RT-Select assigns 94.6% of packets to the WiFi radio and 5.4% to the ZigBee radio and lets WiFi transmit for the entire period and ZigBee finish early, while GreenBag assigns 85.9% of packets to the WiFi radio and 14.1% to the ZigBee radio and lets both radios finish their transmissions at the same time, resulting RT-Select consumes $37mJ$ less energy than GreenBag. High deadline miss rates are observed under both RT-Select and GreenBag when the image size is larger than 87KB, beyond the capacity of two radios with the consideration of radio switching overhead. The results show that RT-Select always provides the better radio selections on various data rates.

D. Experiments with Five Radios

In this set of experiments, we examine the effectiveness of RT-Select-General with our prototype device hosting five radios (see Appendix B). We compare RT-Select-General



(a) Comparison on radio energy consumption.



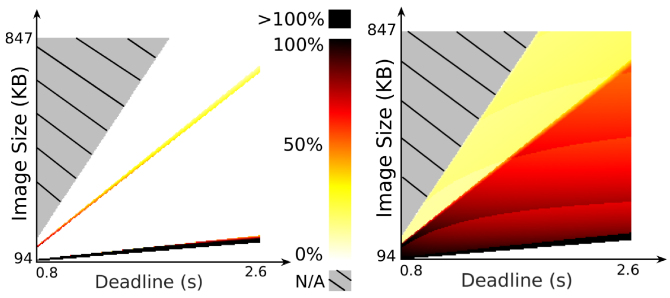
(b) Comparison on deadline miss rate.

Fig. 7. Performance of GreenBag, Optimal, and RT-Select-General with five radios when the application transmits at different data rates with the same deadline.

against GreenBag and Optimal.

We first explore RT-Select-General's performance under a fixed traffic demand with different deadline requirements. We configure the benchmark application A to transmit a 109KB image (1280×720 JPEG) in each period and repeat the experiments with 12 different deadlines ranging from 0.80s to 1.24s. Figure 6 shows the comparisons on radio energy consumption and deadline miss rate under GreenBag, Optimal, and RT-Select-General, respectively. As Figure 6(a) and Figure 6(b) show, all three methods suggest using all radios to accommodate the tight deadlines (i.e., 0.80s and 0.84s). High deadline miss rates are observed when the deadline is 0.80s, beyond the capacity of all five radios together when considering radio switching overhead. When the deadline is larger than 0.84s, RT-Select-General achieves significant energy savings ranging from 308mJ to 436mJ compared to GreenBag with the deadline miss rates no higher than 1%. RT-Select-General makes the optimal selections for all deadlines except 0.88s and 0.92s. In those two cases, RT-Select-General selects to use the BCM43438 radio as the secondary radio based on the sorting of B_i (see Section III-B), while Optimal decides to use the CC2420 radio instead.

We also evaluate RT-Select-General's performance under various traffic demands with a fixed deadline. We configure the benchmark application B to transmit a JPEG image with a fixed deadline (1.44s) in each period and repeat the experiments with 12 different image sizes ranging from 109KB (1280×720 JPEG) to 433KB (1920×1080 JPEG). As Figure 7(a) shows, RT-Select-General consistently consumes less energy (298mJ on average) compared to GreenBag and performs close to what Optimal offers (30mJ difference on average). RT-Select-General provides optimal selections to nine cases among the 12 cases. Please note that high deadline



(a) RT-Select-General over Optimal. (b) GreenBag over Optimal.

Fig. 8. Radio energy comparisons with five radios at various combinations of traffic demands and deadlines. The grey shaded areas denote the invalid combinations that the optimal deadline miss rate is higher than 5%. The colors in each subfigure denote the percentages of more energy consumed than Optimal, i.e., $(E(RT\_Select\_General)-E(Optimal))/E(Optimal)$ and $(E(GreenBag)-E(Optimal))/E(Optimal)$, respectively.

miss rates are observed under all three methods when the image size is 433KB, beyond the capacities of all radios operating simultaneously when considering radio switching overhead. The results demonstrate the effectiveness of RT-Select-General in reducing the energy consumption, while meeting satisfactory real-time requirements.

E. Large-scale Simulation Study

Relying on the radio characteristics measured on our platform with five radios, we also perform a large-scale simulation study to measure radio energy consumption and deadline miss rate at various combinations of traffic demands and deadlines. In this set of experiments, we uniformly select 200 image sizes ranging from 94KB (1280×720 JPEG) to 847KB (3840×2160 JPEG) and 200 deadline samples ranging from 0.8s to 2.6s and then simulate radio energy consumption of running Optimal, GreenBag, and RT-Select-General, respectively, under all valid combinations of traffic demands and deadlines (optimal deadline miss rate no higher than 5%).

Figure 8(a) is a heat map plotting the energy consumption difference between RT-Select-General and Optimal and Figure 8(b) shows the difference between GreenBag and Optimal. The white areas of Figure 8(a) shows the cases (94.4% of deadline and image size combinations) where RT-Select-General makes the optimal radio selections and traffic partitions. GreenBag only makes the optimal decisions in 5.4% of combinations, as shown in Figure 8(b). The mean energy consumption difference between RT-Select-General and Optimal is 7.1%, while the difference between GreenBag and Optimal is 60.8%. The simulation results confirm that RT-Select-General can provide optimal selections to most cases and significantly outperforms GreenBag under various combinations of data rates and deadlines.

VI. Related Work

Bandwidth aggregation for a device with multiple network interfaces has also been studied for years in the literature and many techniques are readily available [18]. For instance, multipath TCP (MPTCP) [19] is one of the most widely used

techniques and now a new standardized transport protocol that allows a device to take advantage of data transfer through multiple network interfaces simultaneously. Those early efforts are not directly applicable to embedded wireless devices with power constraints, since they were not designed to provide energy-efficient wireless radio interfaces [20], [21].

There has been increasing interest in studying the energy-aware bundling or switching between WiFi and 3G/4G radios on smartphones. For instance, Bui et al. used WiFi and/or LTE to minimize playback halts due to the buffer underflow when a stored video is streamed to a smartphone [15]. Pering et al. enabled automatically switching between WiFi and Bluetooth to extend battery lifetime [22]. There exists commercial software, e.g., VideoBee, Super Download Lite-Booster, MPTCP in iOS, and KT's GiGA LTE, that supports concurrent use of WiFi and cellular radios. More recently, research efforts have begun to pay more attention to energy efficiency in the context of smartphones and IoT applications. For instance, Nirjon et al. developed a system to support runtime switching between WiFi and 3G radios to save energy [23], [24]. Lim et al. [20] extended MPTCP to support energy-aware data transfers over WiFi and LTE radios. Nikraves et al. conducted a real-world study of multipath for mobile settings and developed a flexible software architecture to enhance the performance of MPTCP on smartphones [21]. Nika et al. developed an energy model for smartphones to support energy-aware WiFi and LTE radio bundling [25]. Mu et al. developed a radio and transmission power selection system for IoT applications to meet their QoS requirements [26]. These existing approaches are either unaware of timing constraints or limited to mainly WiFi and 3G/4G on smartphone platforms, thus they are not directly applicable to support timely, energy-efficient data transfer using heterogeneous radios in various IoT embedded platforms.

For real-time wireless data deliveries, novel methods (e.g., [27], [28]) have recently been explored to meet timing constraints via real-time MAC, packet scheduling, and routing based on the centralized Time Division Multiple Access (TDMA) scheme. However, most of them consider neither energy efficiency nor heterogeneous radios. In contrast to these real-time approaches, our work aims to support stringent timing constraints with minimal energy consumption by effectively leveraging heterogeneous radios. Our work is therefore orthogonal and complementary.

VII. Conclusion

Heterogeneous radios are becoming increasingly available in modern embedded devices. This paper presents two algorithms which select radios and partition data at runtime to minimize the energy consumption for real-time data transfer. Experimental results show that the proposed solution can significantly reduce the radio energy consumption over the state of the art, while meeting the application specified traffic demand and deadline requirement.

Acknowledgment

This work was supported by the NSF through grant CRII-1657275 (NeTS) and CNS-1526932.

References

[1] J. Ruminski, M. Smiatacz, A. Bujnowski, A. Andrushevich, M. Biallas, and R. Kistler, "Interactions with Recognized Patients Using Smart Glasses," in *IEEE International Conference on Human System Interactions (HSI)*, 2015.

[2] C. Eling, L. Klingbeil, and H. Kuhlmann, "A Direct Georeferencing System for Real-Time Position and Attitude Determination of Lightweight UAVs," in *FIG Working Week*, 2015.

[3] J. Lynch, C. Rarrar, and J. Michaels, "Structural Health Monitoring: Technological Advances to Practical Implementations," *Proceedings of the IEEE, Special Issue on Structural Health Monitoring*, vol. 104, no. 8, 2016.

[4] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis, and M. Welsh, "Wireless Sensor Networks for Healthcare," *Proceedings of the IEEE, Special Issue on Sensor Network Applications*, vol. 98, no. 11, 2010.

[5] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems," *Proceedings of the IEEE, Special Issue on Industrial Cyber Physical Systems*, vol. 104, no. 5, 2016.

[6] T. Watteyne, V. Handziski, X. Vilajosana, S. Duquennoy, O. Hahm, E. Baccelli, and A. Wolisz, "Industrial Wireless IP-Based Cyber Physical Systems," *Proceedings of the IEEE, Special Issue on Industrial Cyber Physical Systems*, vol. 104, no. 5, 2016.

[7] M. P. Andersen, G. Fierro, and D. E. Culler, "System Design for a Synergistic, Low Power Mote/BLE Embedded Platform," in *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2016.

[8] TI CC2650 SimpleLink Multi-Standard 2.4 GHz Ultra-Low Power Wireless MCU. [Online]. Available: http://www.ti.com/product/CC2650

[9] IOT-GATE-iMX7 - Industrial Internet of Things Gateway. [Online]. Available: https://www.compulab.com/products/iot-gateways/iot-gate-imx7-nxp-i-mx-7-internet-of-things-gateway/

[10] LX IoT Cores. [Online]. Available: https://lx-group.com.au/iot-cores/

[11] Holt-Winters Forecasting Method. [Online]. Available: https://www.ons.gov.uk/ons/guide-method/user-guidance/index-of-services/index-of-services-annex-b--the-holt-winters-forecasting-method.pdf

[12] Raspbian Linux. [Online]. Available: https://www.raspbian.org/

[13] Contiki. [Online]. Available: https://github.com/contiki-os/contiki/

[14] Power Monitor by Monsoon Solutions Inc. [Online]. Available: https://www.msoon.com/

[15] D. H. Bui, K. Lee, S. Oh, I. Shin, H. Shin, H. Woo, and D. Ban, "GreenBag: Energy-efficient Bandwidth Aggregation for Real-time Streaming in Heterogeneous Mobile Wireless Networks," in *IEEE Real-Time Systems Symposium (RTSS)*, 2013.

[16] GNU Linear Programming Kit (GLPK). [Online]. Available: https://www.gnu.org/software/glpk/

[17] Analyzing Performance for Amazon Rekognition Apps. [Online]. Available: https://aws.amazon.com/blogs/compute/analyzing-performance-_for-amazon-rekognition-apps-written-on-aws-lambda-using-aws-x-ray/

[18] K. Habak, K. A. Harras, and M. Youssef, "Bandwidth Aggregation Techniques in Heterogeneous Multi-homed Devices," *Computer Networks*, vol. 92, no. P1, 2015.

[19] IETF RFC 6824. [Online]. Available: https://tools.ietf.org/html/rfc6824

[20] Y.-S. Lim, Y.-C. Chen, E. M. Nahum, D. Towsley, R. J. Gibbens, and E. Cecchet, "Design, Implementation, and Evaluation of Energy-Aware Multi-Path TCP," in *International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2015.

[21] A. Nikravesh, Y. Guo, F. Qian, Z. M. Mao, and S. Sen, "An In-depth Understanding of Multipath TCP on Mobile Devices: Measurement and System Design," in *Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2016.

[22] T. Pering, Y. Agarwal, R. Gupta, and R. Want, "Coolspots: Reducing the Power Consumption of Wireless Mobile Devices with Multiple Radio Interfaces," in *MobiSys*. ACM, 2006, pp. 220–232.

[23] S. Nirjon, A. Nicoara, C.-H. Hsu, J. P. Singh, and J. A. Stankovic, "MultiNets: Policy Oriented Real-Time Switching of Wireless Interfaces on Mobile Devices," in *RTAS*, 2012.

[24] ——, "MultiNets: A System for Real-Time Switching between Multiple Network Interfaces on Mobile Devices," *ACM Transactions on Embedded Computing Systems*, vol. 13, no. 4S, 2014.
[25] A. Nika, Y. Zhu, N. Ding, A. Jindal, Y. C. Hu, X. Zhou, B. Y. Zhao, and H. Zheng, "Energy and Performance of Smartphone Radio Bundling in Outdoor Environments," in *International World Wide Web Conference (WWW)*, 2015.
[26] D. Mu, Y. Ge, M. Sha, S. Paul, N. Ravichandra, and S. Chowdhury, "Adaptive Radio and Transmission Power Selection for Internet of Things," in *ACM/IEEE International Symposium on Quality of Service (IWQoS)*, 2017.
[27] Y.-H. Wei, Q. Leng, S. Han, A. K. Mok, W. Zhang, and M. Tomizuka, "RT-WiFi: Real-Time High-Speed Communication Protocol for Wireless Cyber-Physical Control Applications," in *IEEE Real-Time Systems Symposium (RTSS)*, 2013.
[28] R. Jacob, M. Zimmerling, P. Huang, J. Beutel, and L. Thiele, "End-to-end Real-time Guarantees in Wireless Cyber-physical Systems," in *IEEE Real-Time Systems Symposium (RTSS)*, 2016.
[29] Raspberry Pi 3 Model B. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/
[30] Libpcap Library. [Online]. Available: http://www.tcpdump.org/
[31] LoRaWAN Technology for Raspberry Pi. [Online]. Available: https://www.cooking-hacks.com/documentation/tutorials/lorawan-for-arduino-raspberry-pi-waspmote-868-900-915-433-mhz/

APPENDIX A
OPTIMALITY PROOF OF RT-SELECT

We prove that our RT-Select gives an optimal solution under all three cases. We assume to have two radios (R_1 and R_2) and R_1 consumes less or equal energy compared to R_2 when delivering N packets by itself:

$$A_1 + B_1 \times N \leq A_2 + B_2 \times N \tag{7}$$

In case 1), R_1 can deliver all N packets before the deadline by itself. Thus, RT-Select uses R_1 alone and assigns all packets to it. We prove the solution is optimal by contradiction. Suppose there exists a solution consuming less energy by assigning x ($0 < x \leq N$) packets to R_2. If $x = N$, then

$$A_2 + B_2 \times N < A_1 + B_1 \times N \tag{8}$$

or if $x < N$, then

$$\begin{gathered} A_1 + B_1 \times (N - x) + A_2 + B_2 \times x < A_1 + B_1 \times N \\ Thus,\ (B_2 - B_1) \times x + A_2 < 0 \end{gathered} \tag{9}$$

Eq. 8 contradicts Eq. 7. If $B_1 \leq B_2$, Eq. 9 is invalid since $A_2 > 0$ and $x > 0$. If $B_1 > B_2$, from Eq. 9, we have

$$\begin{gathered} \frac{A_2}{B_1 - B_2} < x < N \\ Thus,\ A_2 + B_2 \times N < B_1 \times N \end{gathered} \tag{10}$$

Eq. 10 contradicts Eq. 7 since $A_1 > 0$. We have reached a contradiction. Hence, our assumption that there exists a solution consuming less energy by assigning x ($0 < x \leq N$) packets to R_2 was wrong. RT-Select therefore produces an optimal solution for case 1).

In case 2), neither R_1 nor R_2 can deliver all N packets by itself before the deadline. RT-Select therefore uses both radios. We assume that RT-Select assigns x packets ($0 < x < N$) to R_1 and $N - x$ packets to R_2. Thus, the total energy consumption is

$$\begin{gathered} A_1 + B_1 \times x + A_2 + B_2 \times (N - x) \\ = A_1 + A_2 + B_2 \times N + (B_1 - B_2) \times x \end{gathered} \tag{11}$$

To minimize Eq. 11, x needs to be maximized if $B_1 \leq B_2$ and minimized if $B_1 > B_2$. This is why RT-Select sorts the two radios based on B_i in line 3 in Algorithm 1 and schedules the radio with a smaller B_i to transmit with its maximum capacity (line 7). Thus, RT-Select provides an optimal solution in case 2).

In case 3), only R_2 can deliver all N packets before the deadline. RT-Select decides between (i) using only R_2 to transmit all packets and (ii) using both radios and partitioning the packets between them. We assume that RT-Select assigns x packets ($0 \leq x < N$) to R_1 and $N - x$ packets to R_2. Using both radios consumes less energy only when Eq. 12 is met.

$$A_1 + B_1 \times x + A_2 + B_2 \times (N - x) < A_2 + B_2 \times N \tag{12}$$

thus,

$$(B_2 - B_1) \times x > A_1 \tag{13}$$

Since $x \geq 0$ and $A_1 > 0$, Eq. 13 is met only when $B_1 < B_2$ and $x > \frac{A_1}{B_2 - B_1}$ are both met to assure that using both radios is more energy-efficient than using R_2 alone. When the above condition is met, maximizing x will minimize the left side of Eq. 12, i.e., the energy consumption of two radios. Thus, RT-Select uses R_2 alone if $B_1 \geq B_2$ or $\frac{A_1}{B_2 - B_1}$ is beyond the packet delivery capacity of R_1 (line 10-11). Otherwise, RT-Select uses both radios and schedules R_1 to transmit with its maximum capacity (line 12-14). Hence, RT-Select provides an optimal solution in case 3) too.

Therefore, our RT-Select gives the optimal solution under all three cases.

APPENDIX B
RRASB IMPLEMENTATION

To support the realization of RRaSB and mimic real-world scenarios, we have built a new embedded platform supporting the dynamic switching and bundling among five separate radios with very different characteristics. As Figure 1 shows, our platform is built by instrumenting a Raspberry Pi 3 Model B [29] with five commercial off-the-shelf (COTS) radio hardwares. The Raspberry Pi's onboard BCM43438 radio operates at 2.437GHz. The RT5370 radio operates at 2.462GHz and is connected to the Raspberry Pi through a USB port. The CC2420 and CC2650 radios operate at 2.480GHz and 2.475GHz and are connected to the Raspberry Pi through UART interfaces. The RN2903 radio operates at 915MHz and is connected to the Raspberry Pi through an SPI bus.

We have implemented RRaSB in Raspbian Linux [12] on Raspberry Pi. To support WiFi, our RRaSB implementation adopts the 802.11 physical and MAC layers provided by the Linux kernel and employs the libpcap library [30] for sending and receiving packets to/from the MAC layer. Similarly, we adopt the implementations of 802.15.4 physical and MAC layers in Contiki [13] to support ZigBee. Our implementation also employs the arduPi and LoRaWAN libraries provided by Cooking Hacks Electronic [31] to support LoRa communication. The power consumption in each state of those radios is measured offline by a Monsoon power monitor [14].