Ad Hoc Networks 107 (2020) 102251

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Radio selection and data partitioning for energy-efficient wireless data transfer in real-time IoT applications☆

Di Mu, Mo Sha*, Kyoung-Don Kang, Hyungdae Yi

Department of Computer Science, State University of New York at Binghamton, USA

ARTICLE INFO

Article history:
Received 15 September 2019
Revised 12 June 2020
Accepted 15 June 2020
Available online 20 June 2020

Keywords:
Real-time data transfer
Radio selection
Data partitioning
Energy efficiency
IoT

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 formulate the runtime radio selection and data partitioning for real-time IoT applications as an Integer Linear Programming (ILP) problem and present an *optimal* algorithm that makes quick and optimal decisions when selecting between two radios, a *heuristic* algorithm for the platforms with more radios, and a *runtime* algorithm that reduces deadline miss ratio when facing tight deadlines.

1. 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 [2]. 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 [3]. Other emerging IoT applications, e.g., structural health monitoring [4], clinical monitoring [5], and industrial process automation [6,7], 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 [8] supports ZigBee and Bluetooth Low Energy (BLE) in one device and TI CC2650 [9] integrates those two radios on a single chip. IOT-Gate-iMX7 [10] is an industrial IoT gateway, which supports 4G/LTE, WiFi, Bluetooth, and Zigbee. LX Cellular Core [11] 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

☆ Part of this article was published in Proceedings of the DCOSS [1]

* Corresponding author.
E-mail addresses: dmu1@binghamton.edu (D. Mu), msha@binghamton.edu (M. Sha), kang@binghamton.edu (K.-D. Kang), hyi2@binghamton.edu (H. Yi).

https://doi.org/10.1016/j.adhoc.2020.102251
1570-8705/

devices[1] for real-time applications and reduce the deadline miss ratio when facing tight deadlines, leveraging the aforementioned 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;
- Based on RT-Select, we design the *RT-Select-General* algorithm for the platforms with more radios.
- We design the *Real-Time traffic Balance (RT-Balance)* algorithm that balances the traffic assigned to different radios at runtime to reduce deadline miss ratio when facing tight deadlines.
- 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, RT-Select-General, and RT-Balance in RRaSB and evaluate them experimentally; experimental results show that our RT-Select and RT-Select-General significantly outperform the baseline (GreenBag) and RT-Balance effectively help RT-Select and RT-Select-General reduce deadline miss ratios.

The remainder of the paper is organized as follows. Section 2 introduces our problem formulation. Section 3 presents the design of RT-Select, RT-Select-General, and RT-Balance. Section 4 describes RRaSB. Section 5 presents our experimental evaluation. Section 6 reviews related work and Section 7 concludes the paper.

2. Problem 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, $R_1, \ldots, R_m$, are available on an IoT end device. The characteristics of each radio $R_i (1 \le i \le m)$ 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.

We define the deadline miss ratio as the number of data transfers which are not completed before their deadlines divided by the total number of data transfers. Since the deadline miss ratio directly reflects the performance of real-time applications, we minimize the deadline miss ratio instead of the absolute latency. Thus, our optimization goal is to minimize the radio energy consumption, while meeting the data rate and deadline requirements. To achieve the objective, we select the radio(s) and assign 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 $0 < X_i \le N$ 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:

$$\textbf{\textit{min}} \left\{ \sum_{i \in S} \left(E_{sw\_i} + P_{rb\_i} \times \frac{X_i}{TH_i} + E_{ta\_i} \times ETX_i \times X_i \right) \right\} \tag{1}$$

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:

$$0 \le X_i \le N \ (X_i \in \mathbb{N}) \tag{2}$$

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

$$\sum_{i=1}^{m} X_i = N \tag{4}$$

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:

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

where

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

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.

[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.

[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.

3. 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 processed fast 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. Finally, we design the *RT-Balance* algorithm that balances the traffic assigned to different radios at runtime to reduce deadline miss ratio when facing tight deadlines. All 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. While RT-Select and RT-Select-General output the radio selection decision, RT-Balance adjusts the traffic assignments at runtime and outputs the result whether the deadline is met successfully. 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 2).

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 [9] cannot operate simultaneously, since they share a single DSP modem and a digital PLL. Our algorithms always consider such hardware conflicts when selecting radios.

3.1. RT-Select Algorithm for selection between two radios

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. We have proven the optimality of Algorithm 1 [1]. 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 (Line 2). 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 (Line 3). Therefore, the radio $R_{idx\_1'}$ is more energy-efficient than $R_{idx\_2'}$ without considering radio switching energy consumption A_i. The radio hardware conflict checker "*Conflict*()" gets the boolean information on whether there is a hardware conflict between the two radios which prevents them from being used simultaneously. 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 R_{idx_1} 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 attempts to use both radios. First, RT-Select assigns $X_{max\_(idx\_1')}$ packets to $R_{idx\_1'}$. Then, the remaining packets are assigned to the other radio if there is no hardware conflict between the two radios. (Line 6–10)
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 N packets to it. If there exists a hardware conflict between the two radios, R_{idx_2} is also used alone to avoid the conflict. Otherwise, RT-Select selects both radios and assigns $X_{max\_(idx\_1')}$ packets to $R_{idx\_1'}$ and the remaining packets to the other radio. (Line 12–17)

Algorithm 1: RT-Select.

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

1 *Compute* $A_i, B_i, X_{max\_i} | i = 1, 2$;
2 $(idx\_1, idx\_2) = \textbf{\textit{sort}}\{A_i + B_i \times N \mid i = 1, 2\}$;
3 $(idx\_1', idx\_2') = \textbf{\textit{sort}}\{B_1, B_2\}$;
4 **if** $X_{max\_(idx\_1)} \geq N$ **then**
5 $X_{idx\_1} \leftarrow N$;
6 **else if** $X_{max\_(idx\_1)} < N$ **and** $X_{max\_(idx\_2)} < N$ **then**
7 $X_{idx\_1'} \leftarrow X_{max\_(idx\_1')}$;
8 **if** !*Conflict()* **then**
9 $X_{idx\_2'} \leftarrow N - X_{idx\_1'}$;
10 **end**
11 **else**
12 **if** $B_{idx\_2} < B_{idx\_1}$ **or** $A_{idx\_1'}/(B_{idx\_2} - B_{idx\_1'}) > X_{max\_(idx\_1')}$ **or** *Conflict()* **then**
13 $X_{idx\_2} \leftarrow N$;
14 **else**
15 $X_{idx\_1'} \leftarrow X_{max\_(idx\_1')}$;
16 $X_{idx\_2} \leftarrow N - X_{idx\_1'}$;
17 **end**
18 **end**

3.2. RT-Select-General Algorithm 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 (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 ($idx\_1, \ldots, idx\_m$) 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 ($idx\_1', \ldots, idx\_m'$) in ascending order (Line 3). The radio hardware conflict checker "*Conflict*(R_x, R_y)" gets the boolean information on whether there is a hardware conflict between the radio R_x and any radio in R_y, where R_y is a set that consists of one or more radios.

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)
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 ($idx\_1', \ldots, idx\_m'$) and lets them transmit with their maximum capacity until the selected radios can deliver all N packets before the deadline. If there exists a radio hardware conflict between $R_{idx\_i'}$ and any radio R_k which has already been

[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 [1].

Algorithm 2: RT-Select-General.

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

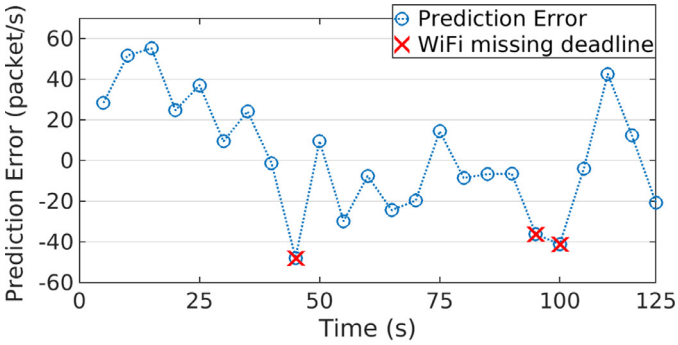
1 *Compute* $\{A_i, B_i, X_{max\_i} \mid i = 1, \ldots, m\}$;
2 $(idx\_1, \ldots, idx\_m) = \textbf{\textit{sort}}\{A_i + B_i \times N \mid i = 1, \ldots, m\}$;
3 $(idx\_1', \ldots, idx\_m') = \textbf{\textit{sort}}\{B_i \mid i = 1, \ldots, m\}$;
4 **if** $X_{max\_(idx\_1)} \geq N$ **then**
5 $X_{idx\_1} \leftarrow N$;
6 **else if** $\textbf{\textit{max}}\{X_{max\_(idx\_i)} \mid i = 1, \ldots, m\} < N$ **then**
7 **for** $i = 1$ **to** m **do**
8 **if** $Conflict(R_{idx\_i'}, \{R_k | X_k > 0\})$ **then**
9 **continue**;
10 **end**
11 **if** $X_{max\_(idx\_i')} < N - \textbf{\textit{sum}}\{X_{idx\_k} \mid k < i\}$ **then**
12 $X_{idx\_i'} \leftarrow X_{max\_(idx\_i')}$;
13 **else**
14 $X_{idx\_i'} \leftarrow N - \textbf{\textit{sum}}\{X_{idx\_k} \mid k < i\}$;
15 **break**;
16 **end**
17 **end**
18 **else**
19 **for** $i = 2$ **to** m **do**
20 **if** $X_{max\_(idx\_i)} < N$ **then**
21 **continue**;
22 **end**
23 **if** $B_{idx\_i} = B_{idx\_1'}$ **or** $A_{idx\_1'}/(B_{idx\_i} - B_{idx\_1'}) > X_{max\_(idx\_1')}$ **or** $Conflict(R_{idx\_i}, R_{idx\_1'})$ **then**
24 $X_{idx\_i} \leftarrow N$;
25 **else**
26 $X_{idx\_1'} \leftarrow X_{max\_(idx\_1')}$;
27 $X_{idx\_i} \leftarrow N - X_{idx\_1'}$;
28 **end**
29 **break**;
30 **end**
31 **end**

selected ($X_k > 0$), the radio $R_{idx\_i'}$ is skipped to avoid the conflict. (Line 6–17)

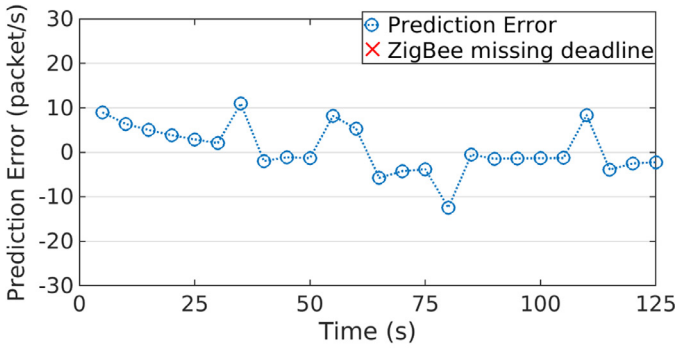
3. if there exists a radio R_{idx_i} which can deliver all packets before the deadline by itself but is not the most energy-efficient one ($i > 1$), then RT-Select-General needs to decide whether to use it alone or combine it with another radio[5] Inspired by Algorithm 1, we consider the radio $R_{idx\_1'}$ (the one with the smallest B_i of all radios) for the possible combination with R_{idx_i}. If R_{idx_i} has the smallest B_i 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. If there exists a hardware conflict between R_{idx_i} and $R_{idx\_1'}$, R_{idx_i} is also selected to be used alone. Otherwise, RT-Select-General combines R_{idx_i} with $R_{idx\_1'}$ and let $R_{idx\_1'}$ transmit with its maximum capacity and assigns the remaining packets to R_{idx_i}. (Line 19–30)

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).

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



(a) WiFi link.



(b) ZigBee link.

Fig. 1. Throughput prediction errors. The deadline misses are marked as crosses.

3.3. RT-Balance Algorithm for runtime traffic balancing

As discussed in Section 3.1 and Section 3.2, RT-Select and RT-Select-General are designed to ensure that all packets can be delivered to their destination by the deadline if they can find feasible radio selection and data partitioning solutions with the assumption that the actual runtime throughput follows the predicted value TH_i. In reality, there does not exist any throughput predictor which achieves 100% prediction accuracy. To study the impact of inaccurate throughput prediction, we perform an empirical study. We use Holt-Winter predictor [12], one of the most effective time series forecasting algorithms, to predict throughput based on historical measurements, run RT-Select to select radios and partition the traffic, and record the deadline misses. We observe that a deadline miss occurs when the traffic assigned to the radio R_i is close to its maximum packet delivery capacity X_{max_i} and the actual throughput of the radio R_i is smaller than the predicted value in that period. Fig. 1 plots the throughput prediction errors when both the WiFi and ZigBee radios are selected by RT-Select to transmit 500 packets (64KB data) with a deadline of 0.8s. Based on line 7–8 in Algorithm 1, the traffic assigned to the WiFi radio has about 478 packets, which is very close to the WiFi radio's capacity, while only about 22 packets are assigned to the ZigBee radio. As Fig. 1(a) shows, the packet deliveries through the WiFi link miss the deadline in three periods (45s, 95s, and 100s), when the actual throughput measurements are smaller than the predictions by at least 30$packets/s$. Fig. 1(b) shows that the packet deliveries through the ZigBee link always meet the deadline because the traffic assigned to the ZigBee radio is far below its capacity. From the results, we can see that the deadline misses occur when the traffic assigned to a radio is very close to its capacity.

To address this issue, we reserve a small portion of the predicted throughput (e.g., 5%) as a guard space, compute X_{max_i} based on the rest (e.g., 95%), and design a runtime algorithm, namely RT-Balance, which balances the traffic assigned to different radios. Algorithm 3 shows the RT-Balance algorithm. When facing

Algorithm 3: RT-Balance.

Input : $N, D, RC_1, \ldots, RC_m$
Global Var: $seq \leftarrow 0$

1 *Compute* $\{X_{max\_i} \mid i = 1, \ldots, m\}$;
2 **if** $\sum_{i=1}^{m} X_{max\_i} > N$ **then**
3 *goto* RT-Select(-General) ;
4 **end**
5 **for** $i = 1$ **to** m **do**
6 **if** *fork*() > 0 **then**
7 **continue** ;
8 **end**
9 **while** $seq < N$ **do**
10 **if** *isReady* (R_i) **then**
11 *Tx* $(R_i, ++seq)$;
12 **end**
13 **if** *time*() $> D$ **then**
14 **return** *FAIL* ;
15 **end**
16 **end**
17 **return** *OK* ;
18 **end**

tight deadlines, RT-Balance creates a process for each radio that repeatedly transmits a packet when it is ready (Line 5–18). In this way, RT-Balance minimizes the latency to meet the deadline and achieves natural load balance among the radios. Specifically, a global variable "*seq*", storing the sequence number of the current packet assigned for transmission, is shared by all processes and initialized as 0. Algorithm 3 first computes the packet delivery capacity (X_{max_i}) of each radio R_i (Line 1), where only the radios without hardware conflict are considered. Then, if the sum of all radios' packet delivery capacities is larger than the traffic demand, RT-Select or RT-Select-General is used to select radios and partition data (Line 2–3). Otherwise, the load balancing is invoked and m child processes are created for the m radios using "*fork*()" (Line 5–7). Each child process uses a loop to request packets for transmission until all packets have been assigned. If there is any unassigned packet and the radio R_i is ready to transmit, *seq* is incremented to be the sequence number of a new packet, which is assigned to the radio R_i for transmission (Line 9–11). The time that has passed since the program starts is checked in each loop. If the deadline has passed before all packets have been transmitted, the child process terminates and indicates that the deadline has been missed (Line 13–14). Otherwise, the child process finishes after all transmission is complete (Line 17).

4. 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. Fig. 2 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

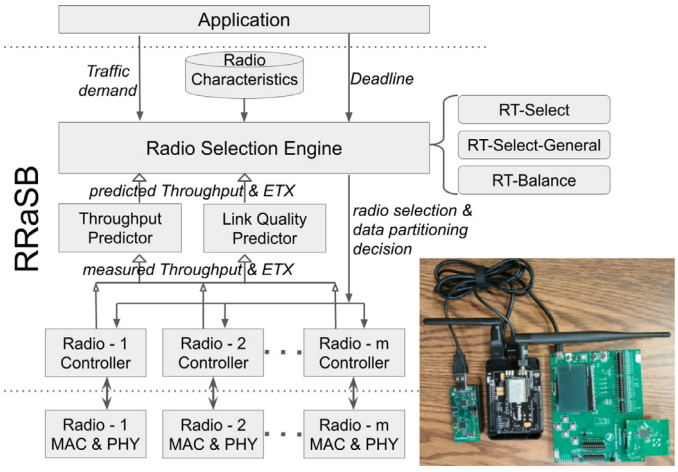


Fig. 2. System architecture and the platform supporting five radios.

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. 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 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 [13] and Contiki [14] and two prototypes: one with two radios and the other with five radios. A power monitor from Monsoon Solutions [15] is connected to the sender to measure the energy consumption. More implementation details can be found in [1].

5. 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 4. 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 ratio with our prototype hosting two radios and repeat the experiments with five radios. We compare our approaches against two baselines: GreenBag using GB-E configuration [16] and GLPK (GNU Linear Programming Kit) [17]. GreenBag is a practical state-of-the-art radio selection approach designed for real-time applications. GreenBag supports multi-radio mode and single-radio mode under GB-E and GB-P configurations. In multi-radio mode, GreenBag seeks to minimize the transmission time by balancing the load

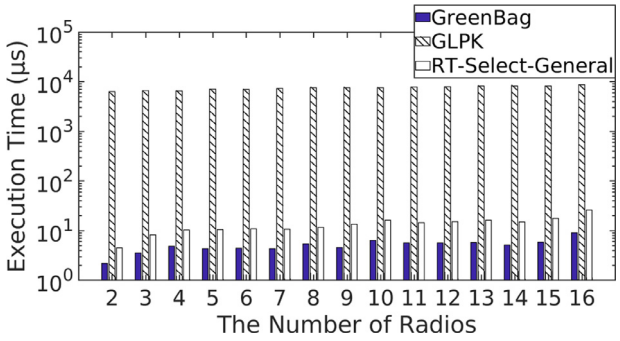


Fig. 3. Execution time of RT-Select-General compared with GreenBag and GLPK.

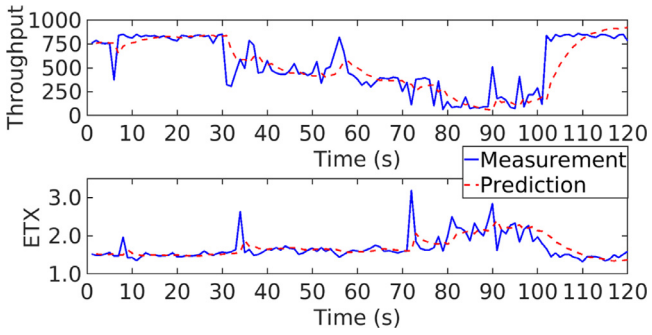


Fig. 4. Throughput and ETX predictions vs. ground truth in a 120-second WiFi link condition trace.

on multiple radios based on link throughput prediction, while the most energy-efficient radio is selected in single-radio mode. GB-E chooses single-radio mode to reduce the energy consumption and switches to multi-radio mode when the bandwidth is insufficient, while GB-P uses multi-radio mode only. GLPK provides the 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 5.1. We run GLPK offline and exclude its energy consumption in the results of optimal solutions (Fig. 8a and 9 a).

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 [2]. 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 [3]. 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.

5.1. 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). Fig. 3 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 Fig. 3 shows, the average execution time of RT-Select-General increases from $4\mu s$ to $26\mu s$ when m increases from 2 to 16, which is slightly ($2 \sim 17\mu s$) longer than what GreenBag uses. The average execution time of GLPK ranges from $6267\mu s$ to $8670\mu s$, which is 336 ~ 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 5.4).

5.2. 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. Fig. 4 plots the example traces showing the throughput and ETX changes of a WiFi link when encountering the controlled interference. An interferer begins the transmission in the same channel from the 31st second to the 100th second. As Fig. 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.

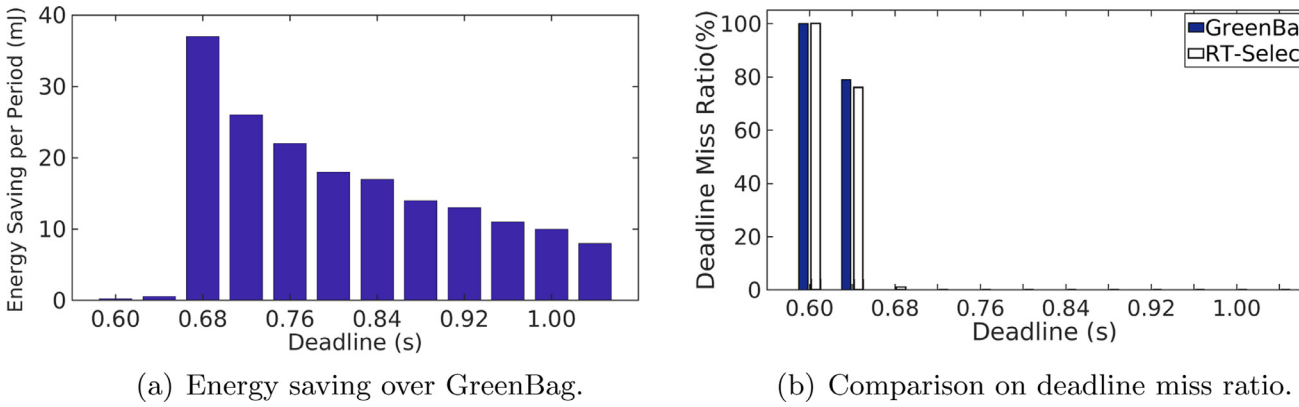
5.3. Experiments with two radios

We run experiments on our prototype hosting two radios [1] (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, 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 [18]. Fig. 5a shows the energy saving of RT-Select over GreenBag per period and Fig. 5b plots the deadline miss ratio. RT-Select shows significant energy saving (ranging from 8mJ to 37mJ[6]) when the deadline is greater than 0.64s with the deadline miss ratios 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 ratios 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.

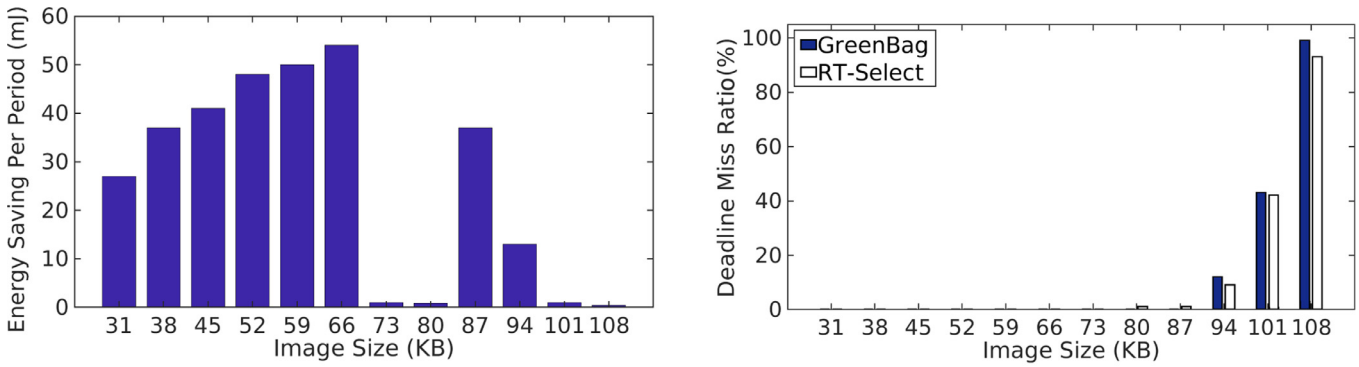
Similarly, we configure the benchmark application B to transmit a JPEG image with the fixed deadline (0.80s) in every period, and repeat the experiments with 12 image sizes ranging from 31KB (640 × 480 JPEG) to 108KB (1280 × 720 JPEG). As Fig. 6a

[6] As a comparison for energy saving values, the CC2650 radio consumes 30mW power when transmitting at 5dBm [9].



(a) Energy saving over GreenBag. (b) Comparison on deadline miss ratio.

Fig. 5. Performance under RT-Select and GreenBag with two radios when the application transmits at a fixed data rate with different deadlines.



(a) Energy saving over GreenBag. (b) Comparison on deadline miss ratio.

Fig. 6. Performance under RT-Select and GreenBag with two radios when the application transmits at different data rates with the same deadline.

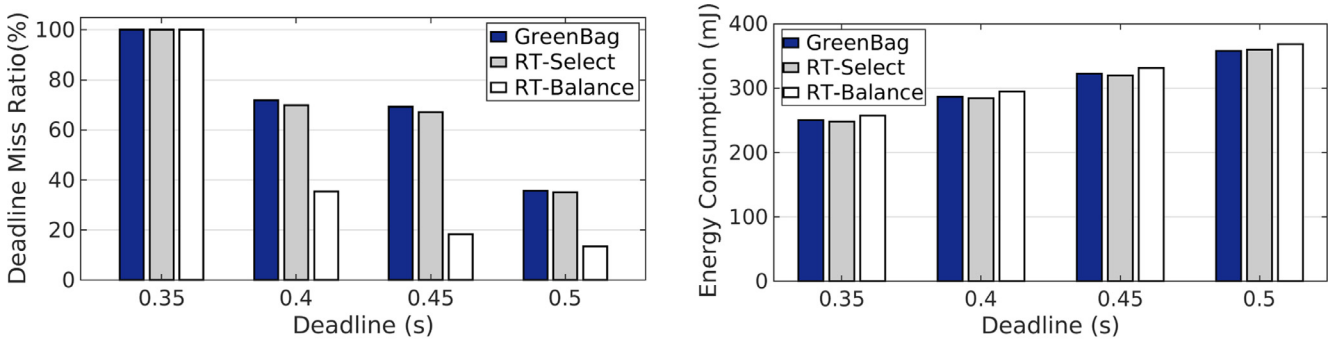
and Fig. 6b show, RT-Select consumes 27 ~ 54*mJ* less energy compared to GreenBag without missing any deadline when the image size is between 31KB and 66KB. The energy savings benefit from RT-Select's decision on keeping only the WiFi radio active rather than using both radios suggested by GreenBag. 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 37*mJ* less energy than GreenBag. High deadline miss ratios 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.

To evaluate the performance of RT-Balance, we configure the benchmark application A to transmit a fix sized image of 64KB with some tight deadlines ranging from 0.35*s* to 0.50*s*. Since the deadlines are very tight, both radios have to keep active for the entire period. As Fig. 7a and 7 b show, RT-Balance significantly reduces the deadline miss ratio by 34.5%, 48.9% and 21.7% compared to RT-Select when the deadlines are 0.40*s*, 0.45*s* and 0.50*s*, respectively. At these deadlines, RT-Balance only increases the energy consumption by 11*mJ*, 12*mJ* and 8*mJ* per period. The slight increase in energy consumption is in exchange for a proportionally much-larger reduction in deadline miss ratio. The reduction on the deadline miss ratio benefits from RT-Balance's runtime traffic balancing between the two radios, in contrast to RT-Select and GreenBag which assign packets to each radio before transmission based on throughput prediction. The deadline miss ratios are 100% for all approaches when the deadline is 0.35*s*, which is too short for the two radios.

5.4. Experiments with five radios

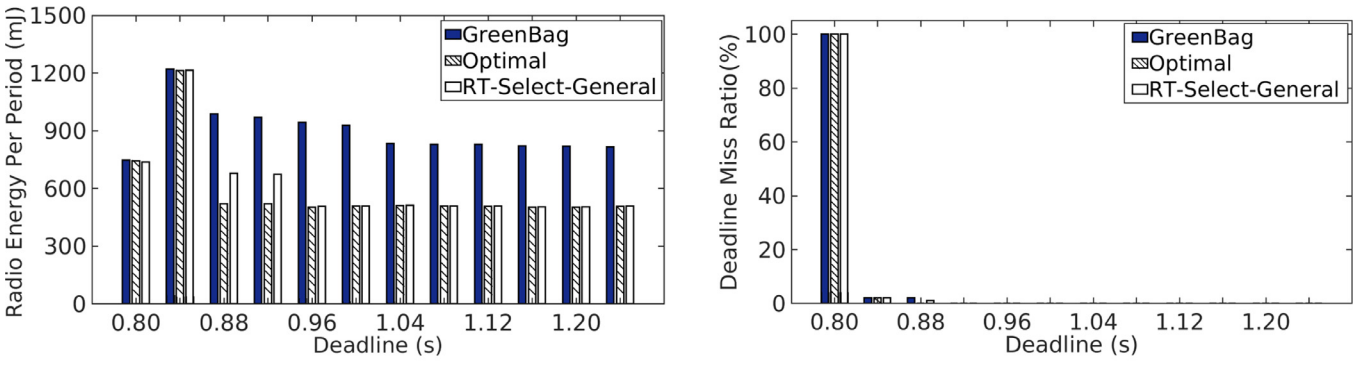
In this set of experiments, we examine the effectiveness of RT-Select-General with our prototype device hosting five radios [1]. We compare RT-Select-General against GreenBag and Optimal.

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 0.80*s* to 1.24*s*. Fig. 8 shows the comparisons on radio energy consumption and deadline miss ratio under GreenBag, Optimal, and RT-Select-General, respectively. As Fig. 8a and b show, all three methods suggest using all radios to accommodate the tight deadlines (i.e., 0.80*s* and 0.84*s*). High deadline miss ratios are observed when the deadline is 0.80*s*, beyond the capacity of all five radios together when considering radio switching overhead. When the deadline is larger than 0.84*s*, RT-Select-General achieves significant energy savings ranging from 308*mJ* to 436*mJ* compared to GreenBag with the deadline miss ratios no higher than 1%. RT-Select-General makes the optimal selections for all deadlines except 0.88*s* and 0.92*s*. In those two cases, RT-Select-General selects to use the BCM43438



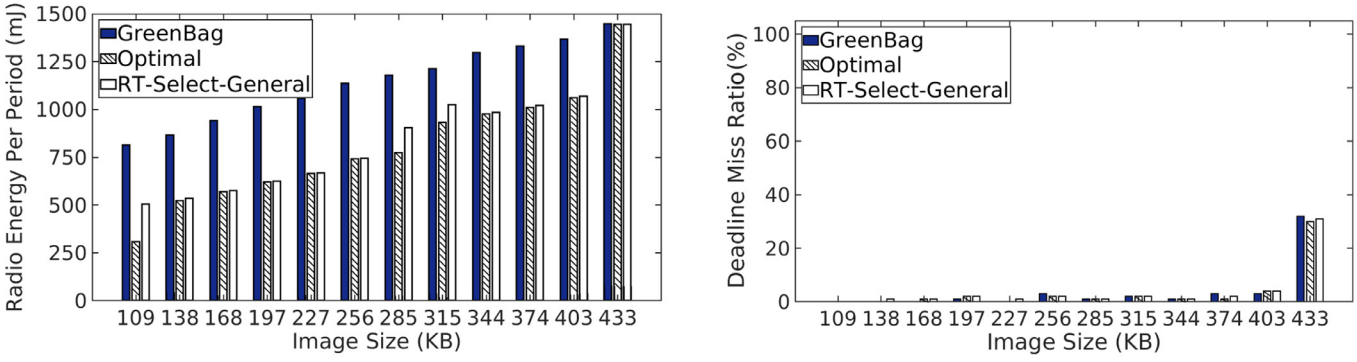
(a) Comparison on deadline miss ratio. (b) Comparison on energy consumption.

Fig. 7. Performance of GreenBag, RT-Select, and RT-Balance with two radios when the application transmits at a fixed data rate with different deadlines.



(a) Comparison on energy consumption. (b) Comparison on deadline miss ratio.

Fig. 8. Performance of GreenBag, Optimal and RT-Select-General solutions with five radios when the application transmits at a fixed data rate with different deadlines.



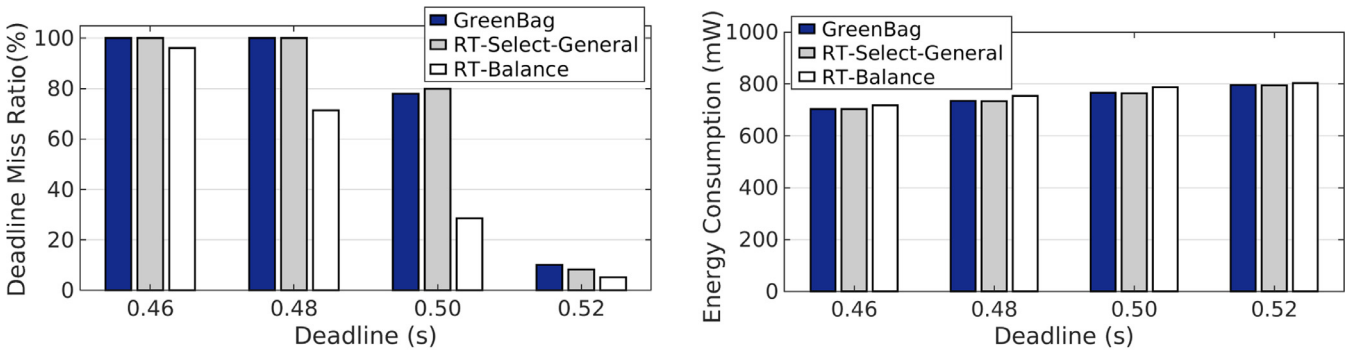
(a) Comparison on energy consumption. (b) Comparison on deadline miss ratio.

Fig. 9. Performance of GreenBag, Optimal, and RT-Select-General with five radios when the application transmits at different data rates with the same deadline.

radio as the secondary radio based on the sorting of B_i (see Section 3.2), while Optimal decides to use the CC2420 radio instead.

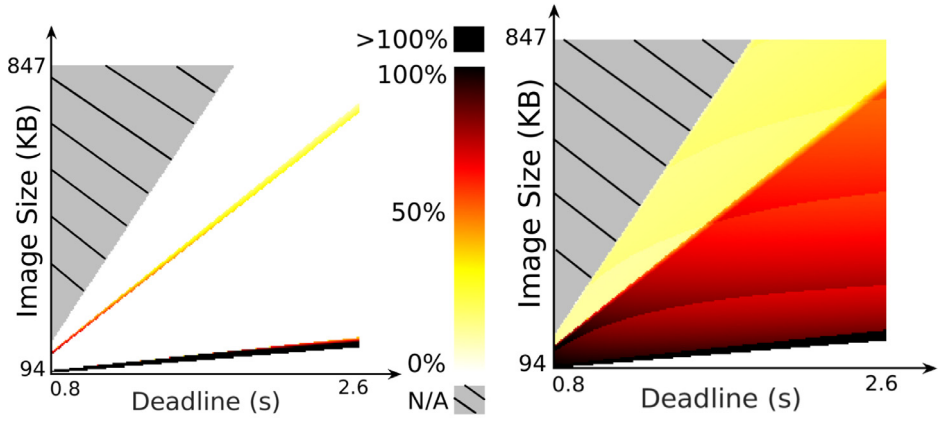
We also evaluate RT-Select-General's performance under various traffic demands with a fixed deadline. We configure the benchmark application B to transmit a JPEG image with a fixed deadline (1.44*s*) in each period and repeat the experiments with 12 different image sizes ranging from 109KB (1280 × 720 JPEG) to 433KB (1920 × 1080 JPEG). As Fig. 9a shows, RT-Select-General consistently consumes less energy (298*mJ* on average) compared to GreenBag and performs close to what Optimal offers (30*mJ* difference on average). RT-Select-General provides optimal selections to nine cases among the 12 cases. Please note that high deadline miss ratios 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. We also perform trace-driven simulations and observe similar improvements at various combinations of traffic demand and deadline [1]. The results demonstrate the effectiveness of RT-Select-General in reducing the energy consumption, while meeting satisfactory real-time requirements.

To evaluate the performance of RT-Balance, we configure the benchmark application A to transmit a fix sized image of 128KB with tight deadlines ranging from 0.46*s* to 0.52*s*. Since the deadlines are very tight, all five radios have to keep active for the entire period. As Fig. 10a and b shows, RT-Balance significantly reduces the deadline miss ratio by 28.6% and 51.4% when the deadlines are 0.48*s* and 0.50*s*, respectively, while only increases the



(a) Comparison on deadline miss ratio. (b) Comparison on energy consumption.

Fig. 10. Performance of GreenBag, RT-Select-General, and RT-Balance with five radios when the application transmits at a fixed data rate with different deadlines.



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

Fig. 11. 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 ratio 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.

energy consumption by 19mJ and 22mJ per period compared to RT-Select-General. The slight increase in energy consumption is in exchange for a proportionally much-larger reduction in deadline miss ratio. The reduction on the deadline miss ratio benefits from RT-Balance's runtime traffic balancing between the five radios, in contrast to RT-Select-General and GreenBag which assign packets to each radio before transmission based on throughput prediction. The deadline miss ratios are nearly 100% for all approaches when the deadline is 0.46s, which is too short for the five radios.

5.5. 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 ratio 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 ratio no higher than 5%).

Fig. 11 a is a heat map plotting the energy consumption difference between RT-Select-General and Optimal and Fig. 11b shows the difference between GreenBag and Optimal. The white areas of Fig. 11a 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 Fig. 11b. 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.

6. 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 [19]. For instance, multipath TCP (MPTCP) [20] 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 data transfers [21,22].

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 [16]. 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, Lim et al. [23] 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 [24]. Mu et al. developed a radio and transmission power selection system for IoT applications to meet their QoS requirements [25]. Wu et al. designed an energy-efficient WiFi and LTE bandwidth aggregation method for video services on mobile devices [26]. Gu et al. developed a low-power LoRa-based control plane bundled with a ZigBee-based data-plane network [27]. 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., [28–30]) have recently been explored to meet timing constraints via real-time MAC protocols, 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.

7. Conclusion and future work

Heterogeneous radios are becoming increasingly available in modern embedded devices, offering new opportunities to use multiple wireless technologies energy-efficiently to accommodate the needs of real-time applications. This paper formulates the runtime radio switching and bundling for real-time IoT applications as an optimization problem and presents three 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.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

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

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.adhoc.2020.102251

References

[1] D. Mu, M. Sha, K.-D. Kang, H. Yi, Energy-Efficient Radio Selection and Data Partitioning for Real-Time Data Transfer, in: IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), 2019.
[2] J. Ruminski, M. Smiatacz, A. Bujnowski, A. Andrushevich, M. Biallas, R. Kistler, Interactions with Recognized Patients Using Smart Glasses, in: IEEE International Conference on Human System Interactions (HSI), 2015.
[3] C. Eling, L. Klingbeil, H. Kuhlmann, A direct georeferencing system for real-time position and attitude determination of lightweight UAVs, FIG Working Week, 2015.
[4] J. Lynch, C. Rarrar, J. Michaels, Structural health monitoring: technological advances to practical implementations, Proceedings of the IEEE, Special Issue on Structural Health Monitoring 104 (8) (2016) 1508–1512.
[5] H.H. Nguyen, F. Mirza, M.A. Naeem, M. Nguyen, A review on IoT healthcare monitoring applications and a vision for transforming sensor data into real–time clinical feedback, in: IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD), 2017.
[6] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, Y. Chen, Real-time wireless sensor-actuator networks for industrial cyber-physical systems, Proceed. IEEE, Spe. Iss. Ind. Cyber Phys. Syst. 104 (5) (2015) 1013–1024.
[7] T. Watteyne, V. Handziski, X. Vilajosana, S. Duquennoy, O. Hahm, E. Baccelli, A. Wolisz, Industrial wireless IP-based cyber physical systems, Proceed. IEEE, Spec. Iss. Ind. Cyber Phys. Syst. 104 (5) (2016) 1025–1038.
[8] M.P. Andersen, G. Fierro, 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.
[9] Texas Instruments, CC2650 SimpleLink Multi-Standard 2.4 GHz Ultra-Low Power Wireless MCU, (2016). https://www.ti.com/product/CC2650 (accessed 20 June 2020).
[10] CompuLab, IOT-GATE-iMX7 - Industrial Internet of Things Gateway, (2020). https://www.compulab.com/products/iot-gateways/iot-gate-imx7-nxp-i-mx-7-internet-of-things-gateway/ (accessed 20 June 2020).
[11] LX Group, IoT Cores, (2018). https://lx-group.com.au/iot-cores/ (accessed 20 June 2020).
[12] P.S. Kalekar, Time series forecasting using holt-Winters exponential smoothing, Kanwal Rekhi School of Information Technology, 2004.
[13] Raspbian Team, Raspbian Operating System, (2020). https://www.raspbian.org/ (accessed 20 June 2020).
[14] Contiki Team, Contiki Operating System, (2018). https://github.com/contiki-os/contiki/ (accessed 20 June 2020).
[15] Monsoon Solutions, Low Voltage Power Monitor, (2019). https://www.msoon.com/lvpm-product-documentation/ (accessed 20 June 2020).
[16] D.H. Bui, K. Lee, S. Oh, I. Shin, H. Shin, H. Woo, D. Ban, GreenBag: Energy-Efficient Bandwidth Aggregation for Real-time Streaming in Heterogeneous Mobile Wireless Networks, IEEE Real-Time Systems Symposium (RTSS), 2013.
[17] Andrew Makhorin, GNU Linear Programming Kit, (2012). https://www.gnu.org/software/glpk/ (accessed 20 June 2020).
[18] Bharath Kumar, Analyzing Performance for Amazon Rekognition Apps, (2017). https://aws.amazon.com/blogs/compute/analyzing-performance-for-amazon-rekognition-apps-written-on-aws-lambda-using-aws-x-ray/ (accessed 20 June 2020).
[19] K. Habak, K.A. Harras, M. Youssef, Bandwidth aggregation techniques in heterogeneous multi-homed devices: A Survey, Comput. Netw. 92 (1) (2015) 168–188.
[20] Internet Engineering Task Force (IETF), RFC 6824 - TCP Extensions for Multipath Operation with Multiple Addresses, (2013). https://tools.ietf.org/html/rfc6824/ (accessed 20 June 2020).
[21] A. Nikravesh, Y. Guo, F. Qian, Z.M. Mao, S. Sen, An In-depth Understanding of Multipath TCP on Mobile Devices: Measurement and System Design, in: ACM International Conference on Mobile Computing and Networking (MobiCom), 2016.
[22] M.J. Shamani, W. Zhu, S. Rezaie, On the Energy Inefficiency of MPTCP for Mobile Computing, in: International Conference on Wired/Wireless Internet Communication (WWIC), 2016.
[23] Y.-S. Lim, Y.-C. Chen, E.M. Nahum, D. Towsley, R.J. Gibbens, E. Cecchet, Design, Implementation, and Evaluation of Energy-Aware Multi-Path TCP, in: ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT), 2015.
[24] A. Nika, Y. Zhu, N. Ding, A. Jindal, Y.C. Hu, X. Zhou, B.Y. Zhao, H. Zheng, Energy and Performance of Smartphone Radio Bundling in Outdoor Environments, in: International World Wide Web Conference (WWW), 2015.
[25] D. Mu, Y. Ge, M. Sha, S. Paul, N. Ravichandra, S. Chowdhury, Adaptive radio and transmission power selection for Internet of Things, ACM/IEEE International Symposium on Quality of Service (IWQoS), 2017.
[26] J. Wu, B. Cheng, M. Wang, J. Chen, Energy-efficient bandwidth aggregation for delay-constrained video over heterogeneous wireless networks, IEEE J. Sel. Areas Commun. 35 (1) (2017) 30–49.
[27] C. Gu, R. Tan, X. Lou, D. Niyato, One-hop out-of-band control planes for low-power multi-hop wireless networks, in: IEEE Conference on Computer Communications (INFOCOM), 2018.
[28] R. Jacob, M. Zimmerling, P. Huang, J. Beutel, L. Thiele, End-to-end real-time guarantees in wireless cyber-physical systems, IEEE Real-Time Systems Symposium (RTSS), 2016.

[29] T. Zhang, T. Gong, Z. Yun, S. Han, Q. Deng, X.S. Hu, FD-PaS: a fully distributed packet scheduling framework for handling disturbances in real-time wireless networks, IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), 2018.
[30] C. Wu, D. Gunatilaka, M. Sha, C. Lu, Real-time wireless routing for industrial Internet of Things, in: IEEE/ACM International Conference on Internet-of-Things Design and Implementation (IoTDI), 2018.

Di Mu is a PhD candidate in the Department of Computer Science at the State University of New York at Binghamton. His research interest is adaptive wireless networking for Internet of Things.

Mo Sha is an Assistant Professor in the Department of Computer Science at the State University of New York at Binghamton. He received his Ph.D. degree in Computer Science from Washington University in St. Louis in 2014. Prior to his PhD, he received a M.Phil. degree from City University of Hong Kong in 2009 and a B.Eng. degree from Beihang University in 2007. His research interests include wireless networks, Internet of Things, embedded and real-time systems, and Cyber-Physical Systems.

Kyoung-Don (KD) Kang is a Professor in the Department of Computer Science at the State University of New York at Binghamton. His research areas include real-time data services, performance management, and security in cyber-physical systems and the Internet of Things.

Hyungdae Yi is a PhD candidate in the Department of Computer Science at the State University of New York at Binghamton. His research interest includes real-time database, sensor network, and stream database.