DIPS: Dual-interface Dual-pipeline Scheduling for Energy-efficient Multi-hop Communications in IoT

Hua Qin, Weihong Chen, Buwen Cao, Min Zeng, Jessica Li and Yang Peng

***Abstract*—The future Internet of Things (IoT) will enable Internet connectivity for a vast amount of battery-powered devices, which usually need to communicate with each other or to some remote gateways through multi-hop communications. Although ZigBee has become a widely-used communication technology in IoT, Wi-Fi, on the other hand, has its unique advantages such as high throughput and native IP compatibility, despite its potentially higher energy consumption. With the development of IoT, more and more IoT devices are equipped with multiple radio interfaces, such as both Wi-Fi and ZigBee. Inspired by this, we propose a Dual-Interface Dual-Pipeline Scheduling (DIPS) scheme, which leverages an activation pipeline mainly constructed by low-power ZigBee interfaces to wake up a data pipeline constructed by high-power Wi-Fi interfaces on demand, towards enabling multi-hop data delivery in IoT. The objective is to minimize network energy consumption while satisfying certain end-to-end delay requirements. Extensive simulations and prototype-based experiments have been conducted. The results show that the energy consumption of DIPS is 96.5% and 92.8% lower than that of the IEEE 802.11's standard power saving scheme and a state-of-the-art pipeline-based scheme in moderate traffic scenarios, respectively.**

***Index Terms*—IoT, energy efficiency, dual radio, pipeline.**

I. Introduction

IN the Internet of Things (IoT), a wide variety of devices, such as entertainment electronics, health appliances, wearable gadgets and industrial sensors, are powered by batteries, and need to wirelessly communicate with each other or some remote IoT gateways through multi-hop communications. To realize this, many wireless technologies can be employed. On one hand, the IEEE 802.15.4 standard (or ZigBee) has been proposed and widely used for home and building automation, smart metering and IoT in general, due to its low-cost and low-power features. On the other hand, the IEEE 802.11 standard (or Wi-Fi) dominates the present-day consumer electronics fields because of its high data rate and long communication range. Any IoT device that connects to smartphones, tablets, digital cameras, TVs and PCs would benefit from Wi-Fi connectivity for reliability and high throughput. The economic gains of reusing the existing Wi-Fi infrastructure drive and facilitate faster deployment with Wi-Fi than other competing technologies. More importantly, Wi-Fi has the advantage of native compatibility with IP, which is the key for IoT communications [1]. The feasibility of connecting battery powered sensors to the IoT using off-the-shelf Wi-Fi chips has already been demonstrated in [2], [3]. Nevertheless, due to its energy-hungry nature, Wi-Fi is often not recommended for short range multi-hop communications in IoT.

To improve the energy efficiency of multi-hop networks based on Wi-Fi, numerous protocols have been proposed. Generally, they can be classified into two categories: *synchronous* approaches and *asynchronous* approaches. The IEEE 802.11 DCF standard specifies a power saving management (PSM), which is the most widely studied synchronous approach. The basic idea is to let all nodes wake up every certain time interval, called beacon interval or BI, to exchange beacon frames for time synchronization and then ATIM (Ad-hoc Traffic Indication Message) frames to announce possible incoming data packets to the receivers. To further improve the performance of PSM, lots of protocols [1], [4]–[6] have been proposed by using some heuristics to adjust the behaviors of PSM (e.g., modifying ATIM frame, rescheduling beacon frame, turning on/off PSM on demand, etc.). The basic idea in asynchronous approaches [7]–[10] is to let the device that has no outgoing packets stay asleep as long as possible, while periodically waking up to send beacon frames to announce its availability for receiving incoming packets. Hence, low latency requires high wakeup frequency and thus more energy.

Although these approaches can greatly reduce power consumption on using Wi-Fi, they also suffer the problem of high data delivery delay, which is a significant barrier to use Wi-Fi in long-term but time-sensitive IoT applications such as environmental surveillance, localization and tracking, and intrusion detection. The fundamental reason is that a Wi-Fi radio must perform high-power idle listening (comparable to transmission and reception [9]) continuously in order to prepare for *unpredictable* incoming traffic, which renders a dilemma that more (less) frequent wake-up of Wi-Fi interface results in shorter (longer) delay but higher (lower) energy consumption. This dilemma has driven recent research on leveraging co-located ZigBee or Bluetooth interface to assist Wi-Fi transmission [11]–[14].

Unfortunately, most of existing dual-interface schemes focus on centralized, single-hop WLANs, and their transmission coordination mechanisms must be accomplished directly be-

This work was supported in part by the National Natural Science Foundation of China under Grant 61602164, in part by the Hunan Provincial Natural Science Foundation of China under Grant 2018JJ3023 and 2018JJ2024, and in part by the Research Foundation of Education Bureau of Hunan Province under Grants 17B050 and 17A037.

Hua Qin, Weihong Chen, Buwen Cao and Min Zeng are with the College of Information and Electronic Engineering, Hunan City University, Yiyang, Hunan, China (e-mail: qinhua_hcu@163.com, paney@hnu.edu.cn, cbwchj@126.com, mindytsang@163.com).

Jessica Li and Yang Peng are with the Division of Computing and Software Systems, University of Washington Bothell, Bothell, WA, USA (e-mail: jxli@uw.edu, yangpeng@uw.edu).

tween data source and destination. Moreover, network traffic investigated in these schemes is more uniform among different devices. Such a traffic pattern is unrealistic for multi-hop networks with dynamic traffic over multiple data flows, where conflicts/contentions may occur. Furthermore, the unreliability of ZigBee communications has also become an obstacle to achieve efficient multi-hop communications, which has not been appropriately tackled in existing dual-interface schemes. As a result, these limitations make existing schemes inapplicable to achieving low-energy and low-delay multi-hop data delivery in IoT.

To address the above issues, we propose a **D**ual-**I**nterface Dual-**P**ipeline **S**cheduling (DIPS) scheme, aiming to minimize the overall network energy consumption for multi-hop communications in IoT while satisfying stringent end-to-end delay requirements. The design of DIPS follows the basic paradigm of leveraging the control flow managed by the low-power ZigBee radios to dynamically schedule the high-power Wi-Fi radios for delivering data flows generated by IoT devices, as illustrated in Fig. 1. Following this paradigm, DIPS adopts the below key ideas: (i) low-power ZigBee radios are utilized to construct an activation pipeline to wake up high-power Wi-Fi radios for on-demand data delivery; (ii) the Wi-Fi radios of all devices on the same flow are waken up in a pipelined manner to form a data pipeline for fast and energy-efficient multi-hop data delivery; (iii) the reliable Wi-Fi radios are used when necessary to assist unreliable ZigBee radios for prompt activation of data pipeline and thus delay-bounded data delivery. Particularly, the DIPS framework is equally applicable to other IoT systems, in which high-power Wi-Fi interface co-exists with other low-power interface such as Bluetooth and Z-Wave [15]. In this paper, we only investigate the pipeline scheduling of Wi-Fi and ZigBee.

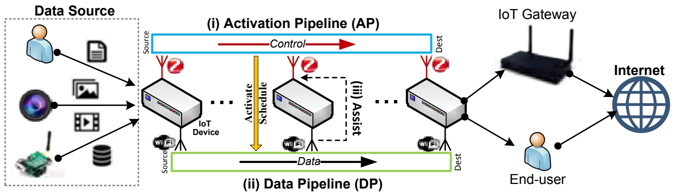


Fig. 1: A dual-interface paradigm for multi-hop communications in IoT with Wi-Fi enabled.

Several pipeline scheduling techniques [16]–[18] have already been proposed for energy-efficient and low-latency data delivery in wireless sensor networks, where sensor nodes usually form a tree rooted at the sink and the nodes along the path between a leaf node and the sink deliver data packets in a pipeline manner using their ZigBee interfaces. However, these techniques cannot be applied to multi-hop communications in IoT, where there may exist multiple destination nodes and the potential high data traffic needs to be handled by high-rate interface like Wi-Fi.

The major design challenges in DIPS appear from the following aspects. Due to the unreliable nature of ZigBee channel, ZigBee control frames may get lost, which requires joint scheduling of the wake-up behaviors of Wi-Fi and ZigBee interfaces, in order to satisfy the end-to-end requirements of various IoT applications. This, combined with the fact that there may exist multiple data flows intertwined with each other and causing collisions/contentions at some intersection nodes, makes the minimization of energy consumption pretty challenging. Besides, unlike the traditional pipeline scheduling schemes where there is only one type of pipeline formed by one type of interface, DISP deals with two heterogeneous interfaces, which collaborate with each other on two different types of pipelines.

The performance of DIPS has been evaluated in large-scale networks via extensive simulations. The results show that DIPS can effectively reduce network energy consumption in a wide range of scenarios. In addition, a prototype system with 10 dual-radio IoT devices has been built to evaluate the performance of DIPS in practical scenarios. The results have demonstrated that the energy consumption of DIPS is 96.5% and 92.8% lower than that of PSM and a state-of-the-art scheme in moderate traffic scenarios, respectively.

The key contributions of this work are listed below.

- To the best of our knowledge, this is the first work on studying the dual-interface pipeline scheduling problem in IoT context. The unique end-to-end delay and energy consumption models have been formalized and thoroughly analyzed.
- The proposed technique can not only construct novel activation and data pipelines for delay-guaranteed data delivery, but also adjust the usage of Wi-Fi and ZigBee interfaces for high energy efficiency under different channel conditions. The guaranteed performance of low energy and low delay on data delivery make the proposed scheme suitable for a wide range of IoT applications.
- The proposed scheme has been implemented and extensively evaluated in both simulation and testbed experiments, through which the effectiveness and the efficiency of the proposed scheme have been verified.

In the rest of the paper, Section II first summarizes the related work. Then, Section III presents the preliminaries. Next, Section IV conducts theoretical analysis. Based on the guidelines obtained in Section IV, a practical design of DIPS is described in Section V. The results of simulations and prototype implementation are reported in Sections VI and VII, respectively. Finally, Section VIII concludes the paper.

II. Related Work

In this section, we discuss the existing techniques that can be used for energy-efficient multi-hop communications in Wi-Fi enabled IoT, each followed by a comparison with our proposed DIPS scheme. Finally, a brief summary of these techniques is given to highlight the unique features of DIPS.

A. PSM Schemes

1) Non-pipelined PSM: The IEEE 802.11power saving management (PSM), adopting a *proactive* wake-up strategy, has been standardized for ad hoc mode. The key idea is to let all nodes wake up every certain time interval, called beacon interval or BI, to exchange beacon frames for time

TABLE I: Comparison among different schemes

	Scheduling method	Wi-Fi wake-up strategy	Energy efficiency	End-to-end latency	Adaptability	Cost	Reusability
Non-pipelined PSM	Synchronous	Proactive	Low	High	Mid	Low	Yes
Pipelined PSM	Synchronous	Proactive	Mid	Low	Mid	Mid	Yes
Multi-channel Scheduling	Synchronous	Reactive	Mid	Mid	High	High	No
Energy-efficient MAC	Asynchronous	Proactive	Mid	Mid	Low	Mid	No
Wake-up Receiver	Asynchronous	Reactive	High	Mid	High	High	No
Cross-interface Approach	Synchronous	Reactive	High	High	High	Low	Yes
DIPS	Synchronous	Reactive	High	Low	High	Low	Yes

synchronization and then ATIM (Ad-hoc Traffic Indication Message) frames to announce possible incoming data packets to the receivers. Although the standard PSM has the advantages of flexibility and simplicity, its energy efficiency varies significantly in different traffic scenarios. To improve the performance of PSM, a large amount of research [4]–[6] has been conducted. Basically, they share the common idea of minimizing the node's time spent in idly listening by adopting certain mechanisms (e.g,. rescheduling beacon frame, modifying ATIM frame, changing the length of ATIM window, etc.) and put nodes into asleep mode as much as possible. However, it has been shown that going into the asleep mode too early may increase link delay, which eventually leads to even longer end-to-end delay [1].

2) Pipelined PSM: Based on the idea of forwarding ATIM frames along multiple hops in a single BI [19], MH-PSM [1] has been proposed to enable low-latency multi-hop data delivery in IoT. However, as we have found in our experiments, the energy efficiency of such pipelined PSM scheme is still limited especially in low-traffic scenarios, though it can deliver a very low end-to-end delay. The main reason is that MH-PSM is built on top of the proactive PSM and thus it has to wake up high-power Wi-Fi interface periodically even if there is no incoming traffic.

Compared with the above PSM-based schemes, DIPS is essentially a *reactive* wake-up scheme, which can wake up Wi-Fi interface on demand of current traffic. Thus, DIPS has the advantage of being able to adapt to traffic dynamics, which makes it applicable to a wider range of IoT applications.

B. Multi-channel Scheduling

Several multi-channel scheduling schemes [8], [10] have been proposed to improve the energy efficiency for multi-hop networks through using one channel for controlling while another channels for data transmission. By sacrificing extra frequency bands, they allow the nodes to wake up on demand for data transmission. Such reactive schemes can avoid the unnecessary periodical wake-up that inherently exists in proactive schemes and thereby improves energy efficiency. Although DIPS shares the similar idea of using a secondary channel for controlling, it leverages the already-existing and lower-power ZigBee interface and thus can deliver higher energy efficiency.

C. Energy-efficient MAC Protocols

Many MAC protocols [7], [9], [20] have been designed for energy-efficient multi-hop communications. The core idea is to let the device that has no outgoing packets stay asleep as long as possible, while periodically waking up to send beacon frames to announce its availability for receiving data packets. However, their energy efficiency is limited because of frequent wake-ups for transmitting beacon frames. To reduce energy consumption, nodes need to wake up less frequently, which can lead to larger end-to-end delay. Unlike these approaches, DIPS can carefully trade off between energy and end-to-end delay, which makes it suitable for more IoT applications.

D. Wake-up Receiver Solutions

Originally designed for power saving in wireless sensor networks, *wake-up receiver*, a low-power radio, has been also developed for energy-efficient WLAN [21]–[23]. In such wake-up receiver equipped WLANs, each access point (AP) is equipped with a wake-up receiver, which can detect wake-up signals transmitted by stations and wakes up the main Wi-Fi radio of the AP for data transmission. Although the above approaches can be used to achieve energy-efficient and low-latency multi-hop communications in a similar way, an extra wake-up receiver needs to be installed in each IoT device, which is costly [21]. In contrast, DIPS relies on an independent ZigBee interface that has already been widely used in IoT and continuously supported by the IEEE 802.15.4 standard (which can also work with 6LoWPAN to enable the IPv6 on energy-constrained IoT devices [24]). The reuse of the existing Wi-Fi and ZigBee standards offers key cost savings and can facilitate the faster deployment of our proposed dual-interface system for multi-hop communications in IoT.

E. Cross-interface Approaches

Recent research has been conducted to utilize co-located interfaces to assist Wi-Fi data transmission, called *cross-interface* approaches. Blue-Fi [25] and CONET [13] leverage co-located Bluetooth to improve energy efficiency of WLANs. ZiFi [26] has been designed to use ZigBee interfaces to detect the existence of WLANs. WiZi-Cloud protocols [14] make use of Wi-Fi and ZigBee interfaces on smartphones and access points to realize energy-efficient, ubiquitous, real-time intra-device/inter-AP handover. ZPSM [12] has been proposed to use low-power ZigBee radio to wake up high-power Wi-Fi radio on demand of current traffic for improved energy efficiency in WLANs. Since these schemes are mainly designed for power saving, their latencies are relatively long and thus unsuitable for multi-hop communications. Although DIPS is also a cross-interface scheme, it aims at the optimization of energy efficiency with end-to-end delay requirements for multi-hop communications in IoT. Moreover, DIPS first adopts the pipeline scheduling mechanism of low end-to-end latency in dual-interface context, which technically differs from the previous work.

F. Summary

To highlight the difference among the above-discussed schemes, we briefly summarize their key features in Table I. Since DIPS is a pipelined cross-interface scheme, it naturally inherits the advantages of both pipelined schemes and cross-interface approaches. Consequently, as shown in the table, it is the only one that can achieve high energy efficiency and low end-to-end latency simultaneously while possessing several advantages of reusability, adaptability and low costs.

III. Preliminaries

A. System Model

In this paper, we consider a multi-hop network of IoT devices, which are time synchronized with each other. Each IoT device (called node hereafter) has both Wi-Fi (IEEE 802.11) and ZigBee (IEEE 802.15.4) interfaces. When a node needs to communicate with another node, a routing protocol may be employed to set up a route between them. The data delivery along a route is referred to as a *data flow*. A data flow f_j has a desired *end-to-end delay bound* (denoted by D_j^{e2e}), which requires that the time elapse from the arrival of the packet at the source node to the reception of the packet at its destination node is no more than D_j^{e2e}.

B. Design Objective

The design objective is to jointly schedule the duty cycles of both Wi-Fi and ZigBee interfaces of all nodes so as to minimize the network energy consumption while satisfying the end-to-end data delivery delay requirements of all flows. The network energy consumption is defined as: the total energy consumption (including both Wi-Fi and ZigBee) of all nodes divided by the total number of data packets received by all destination nodes in the network. Particularly, the lower (higher) the network energy consumption, the higher (lower) the energy efficiency. The end-to-end data delivery delay requirement of a flow is defined as: the *delay meet ratio*, which is the ratio of the number of data packets that meet delivery deadlines to the number of packets transmitted by the source node, stays above a certain large value.

C. Design Framework

In our system, there are two types of pipelines: *data pipeline* (DP) for delivering data packets and *activation pipeline* (AP) for delivering activation frames to activate an upcoming DP, as shown in Fig. 2. Each node v_i wakes up every certain interval, called *AP interval* (denoted by I^{AP}). At each wake-up, node v_i stays awake for a certain duration, which contains two slots of equal length (denoted by τ), called *wake-up slots*. The first slot is used for receiving activation frames and called *rx-slot*, while the second slot is used for transmitting activation frames and called *tx-slot*. An activation pipeline is formed by letting any node v_i wake up one wake-up slot earlier than its next-hop node v_{i+1} on the same data flow. In addition, nodes v_i and v_{i+1} shall communicate using the same radio interface (either Wi-Fi or ZigBee, but not both) during their wake-up slots. To save energy, a node can only use its Wi-Fi interface to deliver activation frames every certain interval, which is called *w-interval*. For the rest time, the ZigBee interface is used. For example, the w-interval of link (v_i, v_{i+1}) for flow f_j, denoted by $w_{i,i+1|j}$, is 3 in Fig. 2.

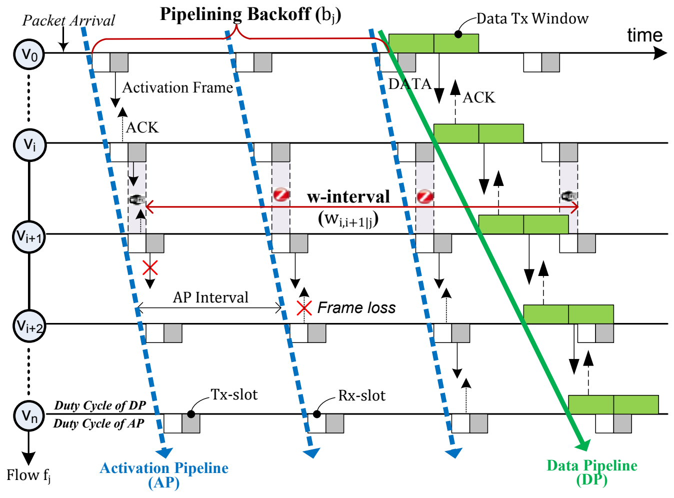


Fig. 2: System framework

The activation process on flow f_j works as follows. When a data packet arrives at source node v_0, v_0 generates an activation frame, which is sent to its next-hop node at the next immediate tx-slot. Upon receiving the activation frame, node v_i forwards it to node v_{i+1} at the next tx-slot. Note that the length of a wake-up slot is set to the time to transmit an activation frame and the corresponding ACK via ZigBee interface. If the transmission fails, the node retries at the next tx-slot. This process continues until the activation frame successfully reaches the destination node. After generating the activation frame, the source node also conducts a certain backoff, which is called *pipelining backoff* and denoted by b_j. After such a backoff period, all nodes on the flow are awake to form a data pipeline, and the source node can transmit the buffered data packets over the DP. The data packets can then be delivered to the destination promptly.

D. Design Principles

To achieve the design objective, our design follows two principles.

Principle I – On-demand pipeline activation with maximized data buffering. With the help of ZigBee interface, APs are formed to dynamically activate asleep Wi-Fi interfaces to form DPs for end-to-end data delivery on demand of current traffic. Besides, to make efficient use of DP, the pipelining backoff should be maximized in order to allow more data packets to be buffered and then delivered through a single DP, as long as the end-to-end delay requirement is satisfied. With the above, energy efficiency of data delivery can be optimized.

Principle II – Adaptive interface scheduling. With the help of Wi-Fi interface, the node in a certain area, where the link quality of ZigBee is low, can temporarily leverage its Wi-Fi interface to take over the activation task. Particularly, by shortening w-interval (i.e., using high-power Wi-Fi interface more often and low-power ZigBee interface less frequently), the end-to-end delay of the activation process can be lowered by sacrificing more energy. With this strategy, the end-to-end delay requirements can be guaranteed.

Therefore, to achieve our design objective, the value of w-interval and pipelining backoff should be appropriately chosen for AP and DP, respectively.

IV. Theoretical Analysis

In this section, we analyze the end-to-end delay and energy consumption given the system model described in Section III.

A. Problem Definition and Assumptions

In the network, there is a set $\mathbf{F}$ of flows. Each flow $f_j \in \mathbf{F}$ is defined as $f_j = \{v_0, v_1, v_2, \cdots, v_i, \cdots, v_{n_j}\}$, where v_i is the i^{th} node following the source node v_0 and n_j is the number of links on flow f_j. The schedule of AP and DP are defined as $\mathbf{S}_{AP} = \{w_{i,i+1|j} | \forall v_i \in f_j, \forall f_j \in \mathbf{F}\}$ and $\mathbf{S}_{DP} = \{b_j | \forall f_j \in \mathbf{F}\}$, respectively. The problem to be solved in our proposed system is how to schedule $\mathbf{S}_{AP}$ and $\mathbf{S}_{DP}$ jointly so as to minimize the network energy consumption while satisfying the end-to-end delay requirements of all flows.

The following assumptions are made in our analysis, though our practical design to be presented later is not restricted to these assumptions.

- Ideal Wi-Fi channel conditions (no data packet loss) are assumed. The packet delay due to contention is either negligible or constant.
- Data packet generation follows the Poisson distribution. The size of all data packets is the same.
- The system is not saturated and no packet is dropped due to overflow of queue. Thus, buffered packets will be eventually transmitted along DP.

The notations frequently used are listed in Table II.

TABLE II: Notation table

$\mathbf{F}$	the set of all flows in the network
$\mathbf{F}_i$	the set of all flows that pass node v_i
$w_{i,i+1\|j}$	the w-interval of link (v_i, v_{i+1}) on flow f_j
$x_{i,i+1\|j}$	the probability that link (v_i, v_{i+1}) uses ZigBee interfaces to transmit activation frames on flow f_j
$p_{i,i+1\|j}$	the probability that node v_i successfully transmits an activation frame to node v_{i+1} on flow f_j via their ZigBee interfaces
$\Omega_{i,i+1\|j}$	the probability that the transmission of activation frame on flow f_j does not conflict with any other flows passing node $v_i \in f_j$
$N_{i,i+1\|j}$	the expected number of transmissions (EXT) for delivering an activation frame over link (v_i, v_{i+1}) on flow f_j
λ_j	the packet arrival rate of the source node of flow f_j
D_j^{e2e}	the required end-to-end delay bound of flow f_j
b_j	the length of pipelining backoff of flow f_j
m_j	the length of DP contention window for flow f_j
n_j	the number of links on flow f_j
I^{AP}	the length of an AP interval
τ	the length of a wake-up slot

B. End-to-end Delay

Once an activation process has been completed, the end-to-end data delivery delay is mainly determined by the pipelining backoff and the actual delivery delay over the DP. We can analyze the end-to-end delay using the model in Fig. 3.

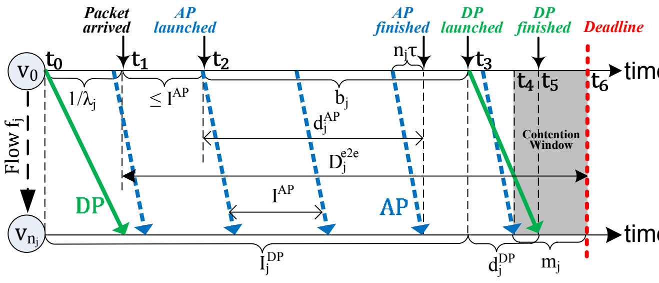


Fig. 3: Analysis model

If we assume the source node should wait at most one AP interval before launching the activation process, and let d_j^{DP} represent the time period to complete the data transmission process over DP, the end-to-end delay on flow f_j (i.e., $t_5 - t_1$), denoted by d_j^{e2e}, should be constrained by $d_j^{e2e} = I^{AP} + b_j + d_j^{DP} \leqslant D_j^{e2e}$. Here, d_j^{DP} is called *DP delay*, which is affected by contentions on using Wi-Fi interfaces when nodes on flow f_j deliver data for different flows. Therefore, each flow f_j shall reserve a window (e.g., the gray area), called *DP contention window* and denoted by m_j, as a temporal buffer to resolve such inter-flow contentions and thus meet the data delivery deadline (e.g., time t_6). This requires $d_j^{e2e} + m_j \leqslant D_j^{e2e}$. Putting the above together, we can get

$$D_j^{e2e} - m_j \leqslant d_j^{e2e} = I^{AP} + b_j + d_j^{DP} \leqslant D_j^{e2e}. \tag{1}$$

From the above, we can see that it is critical to understand d_j^{DP} in order to choose an appropriate value of b_j and set proper contention window m_j to ensure that d_j^{e2e} can satisfy the end-to-end delay requirements.

In the following, we analyze DP delay in details to obtain guidelines for choosing $\mathbf{S}_{DP}$ in practical design of DIPS.

1) DP Delay: In our analysis, data packet arrival is modeled as Poisson process, where λ_j is the packet arrival rate of flow f_j. During a DP interval, which is the time period between any two consecutive DPs on flow f_j (denoted by I_j^{DP}), there are $\lambda_j I_j^{DP}$ data packets buffered at source node v_0. Thus, node v_i takes $\lambda_j I_j^{DP} \cdot T_{D/A}^w$ time to transmit all buffered packets to its next-hop node, where $T_{D/A}^w$ is the turnaround time for delivering a single packet via Wi-Fi. In additional to flow f_j, node v_i may also need to transmit the data packets generated by other passing flows, causing *inter-flow contention* and thus introducing some extra delay. Let $\mathbf{F}_i$ be the set of flows that pass node v_i, the probability that the data delivery on flow f_j occurs concurrently along with any other flow $f_k \in \mathbf{F}_i - \{f_j\}$ can be computed as $(\lambda_k I_k^{DP} \cdot T_{D/A}^w)/I_k^{DP} = \lambda_k T_{D/A}^w$. The expected number of data packets on flow f_k that contend with flow f_j is $\lambda_k I_k^{DP} \cdot \lambda_k T_{D/A}^w = \lambda_k^2 I_k^{DP} \cdot T_{D/A}^w$. Thereby, the *extra contention delay* over link (v_i, v_{i+1}) on flow f_j, denoted by $\hat{d}_{i,i+1|j}$, is constrained by

$$\hat{d}_{i,i+1|j} \leqslant \sum_{f_k \in \mathbf{F}_i - \{f_j\}} (\lambda_k T_{D/A}^w)^2 I_k^{DP}. \tag{2}$$

By accumulating $\hat{d}_{i,i+1|j}$ for all links on flow f_j, we can get

$$d_j^{DP} = n_j \lambda_j I_j^{DP} \cdot T_{D/A}^w + \sum_{i=0}^{n_j - 1} \hat{d}_{i,i+1|j}. \tag{3}$$

2) Guidelines for Choosing $\mathbf{S}_{DP}$*:* In order to buffer more data packets and make efficient use of the pipelined delivery, we always back off b_j period of time to start DP (at time t_3), such that the buffered packets of flow f_j ($\lambda_j I_j^{DP}$ packets in total) can arrive at the destination node right after the beginning of DP contention window (at time t_4), assuming no inter-flow contention (i.e., $d_j^{DP} = n_j \lambda_j I_j^{DP} \cdot T_{D/A}^w$). However, in practice, data packets could arrive at any time within the DP contention window due to inter-flow contentions. Thus, we can have $b_j = D_j^{e2e} - I^{AP} - d_j^{DP} - m_j$. As we model the packet arrival as Poisson process, $I_j^{DP} = 1/\lambda_j + I^{AP} + b_j$. Combining the above, we obtain the following guideline for choosing $\mathbf{S}_{DP}$.

Guideline *1:* Given m_j, we can compute b_j (i.e., $\mathbf{S}_{DP}$) as

$$b_j = \frac{D_j^{e2e} - I^{AP} - m_j - (I^{AP} + \frac{1}{\lambda_j}) n_j \lambda_j T_{D/A}^w}{1 + n_j \lambda_j T_{D/A}^w}. \tag{4}$$

Thus, b_j is determined by m_j.

C. Energy Consumption

As energy overheads for data transmission are mainly from activation processes, we further analyze the energy consumption when transmitting activation frames over APs. Particularly, for any flow f_j, it is desired that the activation process completes right before the DP contention window (at time t_4 in Fig. 3), indicating the maximized utilization of their ZigBee interfaces and thus minimized overall energy consumption for flow f_j. This gives the following optimization condition on energy consumption

$$I^{AP} + d_j^{AP} = D_j^{e2e} - m_j, \tag{5}$$

where d_j^{AP} (called *AP delay*) is the time to complete the activation process over AP. Moreover, by presuming that DP contention window m_j is large enough to handle inter-flow contention among DPs, the above condition can simultaneously ensure end-to-end delay requirements.

In the following, we conduct detailed analysis on AP delay to make a practical transformation of the above dual-purpose condition. Then, constrained by the transformed condition, an optimization problem on energy consumption is then formulated to guide the scheduling of S_{AP} (i.e., $w_{i,i+1|j}$).

1) AP Delay: To complete an activation process, one activation pipeline should be used at least, which takes $n_j \tau$ time. However, in practice, activation frames may get lost due to poor ZigBee link quality or the conflicts with concurrent transmission of activation frames on other flows (called *inter-flow conflicts*). As described before, if an activation frame gets lost, it will be retransmitted in the next AP, thus introducing extra delay. Specifically, the number of extra APs caused by the retransmission of activation frames on flow f_j can be computed as $\lfloor \sum_{i=0}^{n_j-1} (N_{i,i+1|j} - 1) \rfloor$, where $N_{i,i+1|j}$ is the expected number of transmissions (EXT) of activation frame that occur over any link (v_i, v_{i+1}). By modeling frame transmission as Geometric distribution and assuming ideal Wi-Fi channel conditions, we can get

$$N_{i,i+1|j} = \frac{1}{[x_{i,i+1|j} \cdot p_{i,i+1|j} + (1 - x_{i,i+1|j}) \cdot 1] \cdot \Omega_{i,i+1|j}}, \tag{6}$$

where $x_{i,i+1|j}$ is the probability that node v_i and v_{i+1} use their ZigBee interfaces to transmit activation frame for flow f_j. Besides, to characterizing ZigBee link quality and inter-flow conflicts, we introduce $p_{i,i+1|j}$, which is the probability that node v_i successfully transmits an activation frame (i.e., the ACK is successfully received by node v_i) to its next-hop node (i.e., node v_{i+1}) via its ZigBee interface, and $\Omega_{i,i+1|j}$, which is the probability that the transmission of activation frame on flow f_j does *not* conflict with any other flows that pass node v_i, respectively. Then, we can also obtain

$$d_j^{AP} = n_j \tau + I^{AP} \lfloor \sum_{i=0}^{n_j-1} (N_{i,i+1|j} - 1) \rfloor. \tag{7}$$

Rewriting the optimization condition in Eq. (5) as $d_j^{AP} = D_j^{e2e} - m_j - I^{AP}$ and combining it with Eq. (7), we can transform the optimization condition to

$$\sum_{i=0}^{n_j-1} N_{i,i+1|j} = \frac{D_j^{e2e} - m_j - n_j \tau}{I^{AP}} + n_j - 1 = N_j^{opt}, \tag{8}$$

where N_j^{opt} is the optimum total number of EXTs for all links on flow f_j. Practically speaking, it is the total number of EXTs allowed between the beginning of activation (at time t_2) and the beginning of the DP contention window (at time t_4).

2) Guidelines for Choosing $\mathbf{S}_{AP}$*:* The above condition can only reveal the overall optimal behavior of an entire flow by constraining the total number of EXTs of all links on the flow. To obtain deeper insights for optimizing the wake-up behavior of each individual link, we further characterize the overall utilization of the ZigBee interfaces of all nodes on flow f_j, denoted by z_j, as $\sum_{i=0}^{n_j-1} x_{i,i+1|j}$ (i.e., the expected number of nodes using the ZigBee interface in one AP interval). Then, for any flow f_j, we can obtain the optimization problem of maximizing z_j while constrained by (i) Eq. (8) and (ii) $0 \leqslant x_{i,i+1|j} \leqslant 1, \ \forall (v_i, v_{i+1}) \in f_j$.

By removing the latter constraint, the problem can be relaxed and then solved using Lagrange multiplier, from which we can obtain that z_j is maximized when each link (v_i, v_{i+1}) on flow f_j has $N_{i,i+1|j}$ that is proportional to $1/\sqrt{(1 - p_{i,i+1|j})\Omega_{i,i+1|j}}$ and the sum of EXT of all links is N_j^{opt}. Intuitively, given no inter-flow conflicts (with probability of $\Omega_{i,i+1|j}$), the link with lower ZigBee link quality $p_{i,i+1|j}$ is expected to have a smaller expected EXT and thus a shorter w-interval to increase the wake-up frequency of Wi-Fi interface.

Following the above results, we define an expected EXT $\bar{N}_{i,i+1|j}$ for each link (v_i, v_{i+1}) on flow f_j as follows.

$$N_{i,i+1|j} \leqslant \bar{N}_{i,i+1|j} = N_j^{opt} \cdot \frac{\eta_{i,i+1|j}}{\sum_{k=0}^{n_j-1} \eta_{k,k+1|j}}, \tag{9}$$

where $\eta_{i,i+1|j}$, called *link transmission share*, represents link (v_i, v_{i+1})'s share of N_j^{opt} among all links on flow f_j and it is set to $1/\sqrt{(1 - p_{i,i+1|j})\Omega_{i,i+1|j}}$. As long as $N_{i,i+1|j}$ does not exceed $\bar{N}_{i,i+1|j}$ on each link, we can ensure $\sum_{i=0}^{n_j-1} N_{i,i+1|j} \leqslant N_j^{opt}$ and thus the end-to-end delay requirements. Furthermore, according to Eq. (6), $N_{i,i+1|j}$ is a function of $x_{i,i+1|j}$, which can be computed as $1 - \frac{1}{w_{i,i+1|j}}$. Hence, we can obtain the following guideline for choosing $\mathbf{S}_{AP}$.

Guideline *2:* To minimize energy consumption while satisfying end-to-end delay requirements, we should maximize $w_{i,i+1|j}$ (i.e., $\mathbf{S}_{AP}$) as long as the resulting $N_{i,i+1|j}$ is no greater than the allocated $\bar{N}_{i,i+1|j}$. Thus, $w_{i,i+1|j}$ is determined by $\bar{N}_{i,i+1|j}$.

V. Practical Design

Based on Guideline 1 and 2, we can know that $\mathbf{S}_{DP}$ (i.e., b_j) and $\mathbf{S}_{AP}$ (i.e., $w_{i,i+1|j}$) are determined by m_j and $\bar{N}_{i,i+1|j}$, respectively. Thus, DIPS is designed to dynamically adjust m_j and $\bar{N}_{i,i+1|j}$, such that the optimal schedule of $\mathbf{S}_{DP}$ and $\mathbf{S}_{AP}$ can be obtained in a lightweight manner.

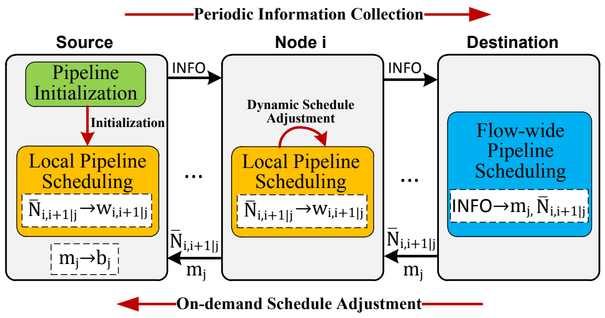

Periodic Information Collection
Source
Node i
Destination
Pipeline Initialization
Initialization
Local Pipeline Scheduling
$\bar{N}_{i,i+1|j} \to w_{i,i+1|j}$
$m_j \to b_j$
INFO
Dynamic Schedule Adjustment
INFO
Flow-wide Pipeline Scheduling
INFO→m_j, $\bar{N}_{i,i+1|j}$
$\bar{N}_{i,i+1|j}$
m_j
On-demand Schedule Adjustment

Fig. 4: Design overview

The adjustment process is depicted in Fig. 4 and can be briefly summarized as follows. Once a flow f_j is set up, the source node launches *pipeline initialization* to help the destination node collect flow information and initialize m_j and $\bar{N}_{i,i+1|j}$ (thus b_j and $w_{i,i+1|j}$ as well). As the network runs, each node v_i on flow f_j executes *local pipeline scheduling* to continuously adjust $w_{i,i+1|j}$ against time-varying ZigBee link quality and inter-flow conflict/contention, according to its allocated $\bar{N}_{i,i+1|j}$. When the destination node learns that the current schedule cannot optimize the performance of flow f_j, it launches *flow-wide pipeline scheduling* to recompute m_j and reallocate $\bar{N}_{i,i+1|j}$, consequently causing the adjustment of b_j and $w_{i,i+1|j}$, respectively.

A. Pipeline Initialization

Initially, all nodes turn on their ZigBee interfaces. When a node needs to communicate with another node, a routing protocol may be employed to find out a route between them. Once a route has been found for flow f_j, the source node sends an initialization request message along the flow. Upon receiving the request, each node v_i piggybacks its estimated ZigBee link quality $p_{i,i+1|j}$ and extra contention delay $\hat{d}_{i,i+1|j}$, and forwards the request. Here, $\hat{d}_{i,i+1|j} = \sum_{f_k \in \mathbf{F}_i - \{f_j\}} (\lambda_k T^w_{D/A})^2 (D^{e2e}_k + 1/\lambda_k)$ according to Eq. (2), where I^{DP}_k is conservatively selected as $D^{e2e}_k + 1/\lambda_k$.

Once the destination node wakes up and receives the initialization request, it can compute m_j and $\bar{N}_{i,i+1|j}$ as follows. To ensure the end-to-end delay bound, m_j is set to its maximum value m^{max}_j, which is computed as $\sum_{i=0}^{n_j-1} \hat{d}_{i,i+1|j}$. Following the guidelines given by Eq. (9), $\bar{N}_{i,i+1|j}$ for each link (v_i, v_{i+1}) on flow f_j can be allocated to node v_i using Alg. 1, which is a flow-wide scheduling algorithm to be elaborated in Section V-C.

Finally, the initial scheduling information (i.e., m_j and $\bar{N}_{i,i+1|j}$) is propagated backwards along the flow. Upon receiving this information, node v_i chooses $w_{i,i+1|j}$ locally according to the local pipeline scheduling algorithm to be presented in Section V-B2. Besides, the source node computes pipeline backoff b_j by Eq. (4).

B. Local Pipeline Scheduling

When network conditions vary, $N_{i,i+1|j}$ may change, causing low energy efficiency or low delay meet ratio. To optimize the system performance, a scheduling scheme is designed to adjust $w_{i,i+1|j}$ and thus $N_{i,i+1|j}$ at each node.

1) Estimating Actual EXT: To estimate $N_{i,i+1|j}$, we need to measure inter-flow conflicts. To do so, each node v_i maintains a conflict table, which records the *conflict rate* (denoted by $\hat{c}^z_{i|j}$) between flow f_j and all other passing flows in $\mathbf{F}_i - \{f_j\}$ on ZigBee channel. Briefly, $\hat{c}^z_{i|j}$ is estimated as the percentage of the active wake-up slots (tx-slots or rx-slots) of flow f_j most-recently occupied by all other flows in $\mathbf{F}_i - \{f_j\}$ for ZigBee transmissions/receptions during a certain time window (e.g., 100 active wake-up slots). Due to the relatively long period of time for contention resolution (e.g., one wake-up slot of 40 ms used in our evaluations), the impact of inter-flow conflicts over Wi-Fi channel can be ignored, which can be also seen from our experiments. Then, by inserting $x_{i,i+1|j} = 1 - \frac{1}{w_{i,i+1|j}}$ into Eq. (6), we can get the following simplified version of $N_{i,i+1|j}$

$$N_{i,i+1|j} = \frac{1}{(1 - \frac{1}{w_{i,i+1|j}}) \cdot p_{i,i+1|j}(1 - \hat{c}^z_{i|j})(1 - \hat{c}^z_{i+1|j}) + \frac{1}{w_{i,i+1|j}} \cdot 1}, \tag{10}$$

in which $(1 - \hat{c}^z_{i|j})(1 - \hat{c}^z_{i+1|j})$, denoted by $\Omega^z_{i,i+1|j}$, is the probability that no conflicts occur at both node v_i and v_{i+1} on ZigBee channel.

2) Adjusting W-interval: To minimize energy consumption while ensuring link delay requirements $N_{i,i+1|j} \leqslant \bar{N}_{i,i+1|j}$, w-interval $w_{i,i+1|j}$ is adjusted as follows.

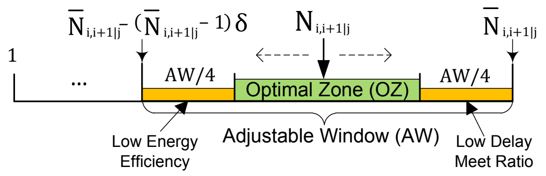


Fig. 5: Local adjustment of w-interval

For any flow f_j, node v_i maintains an adjustable window (AW), as illustrated in Fig. 5. The length of AW is set to $(\bar{N}_{i,i+1|j} - 1) \cdot \delta$, where δ (called *adjustable window ratio*) is a system parameter to control the length of AW and $0 < \delta < 1$. The AW contains an optimal zone in the middle, where $N_{i,i+1|j}$ should be kept to strike a balance between energy and delay. Particularly, when $N_{i,i+1|j}$ moves to the right side of the optimal zone, the risk of failing to ensure $N_{i,i+1|j} \leqslant \bar{N}_{i,i+1|j}$ and thus missing the end-to-end delay bound is high; when $N_{i,i+1|j}$ moves to the left side of the optimal zone, the energy efficiency is low because the node needs to turn on its Wi-Fi interface frequently. In either case, we adjust the w-interval $w_{i,i+1|j}$ by Eq. (10) such that the resulting $N_{i,i+1|j}$ is the middle of the optimal zone (i.e., $\bar{N}_{i,i+1|j} - AW/2$). Otherwise, no adjustment is needed.

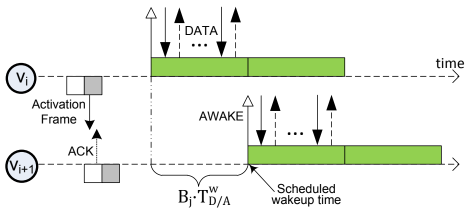
The detailed adjustment of $w_{i,i+1|j}$ works as below.

- Each node v_{i+1} piggybacks its monitored conflict rate (i.e., $\hat{c}^z_{i+1|j}$) in an activation ACK frame.
- Upon receiving the ACK, node v_i computes the new value of $w_{i,i+1|j}$, say $w'_{i,i+1|j}$, as presented earlier. If $w_{i,i+1|j} = w'_{i,i+1|j}$, no adjustment is made. Otherwise, node v_i and v_{i+1} need to synchronize their knowledge about the w-interval shared between them.
- To synchronize the w-interval, node v_i executes the new schedule $w'_{i,i+1|j}$ without canceling the current schedule $w_{i,i+1|j}$, and sends a message to notify node v_{i+1} of $w'_{i,i+1|j}$ in the next immediate tx-slot. Once node v_{i+1} receives the message, it cancels the current w-interval $w_{i,i+1|j}$ and executes $w'_{i,i+1|j}$. Later, node v_i can tell whether the update is successful or not by overhearing the messages from node v_{i+1}. If it succeeds, node v_i cancels $w_{i,i+1|j}$ and the adjustment completes. Otherwise, the adjustment fails and node v_i cancels $w'_{i,i+1|j}$.

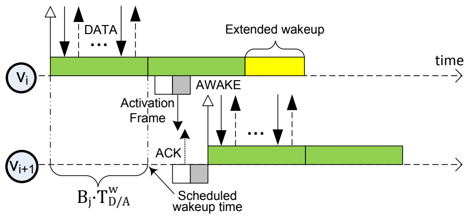
3) Scheduling DP: To schedule the data delivery along a DP, the source node v_0 piggybacks in the activation frame the start time t^{DP}_j of the DP (i.e., b_j time after the activation process begins) and the expected number of buffered data packets B_j for flow f_j when it starts to transmit over the DP (based on current estimation of data traffic at node v_0). Upon receiving the activation frame, each intermediate node v_i wakes up its Wi-Fi interface at time $t^{DP}_j + (i-1) \cdot B_j \cdot T^w_{D/A}$ and sends an AWAKE frame to its previous-hop node v_{i-1} via its Wi-Fi interface to announce that it is awake and ready for data transmission, as illustrated in Fig. 6a. Once node v_{i-1} receives the AWAKE frame, it forwards the data packets to node v_i. Particularly, each data packet has a MORE bit, which is set to 1 when there is no more data packets to transmit. Once node v_i has received a data packet with the MORE bit set to 1, it waits for the AWAKE frame from its next-hop node v_{i+1} to forward the received data packets. When node v_i has completed the data transmission along the passing flows, it goes to sleep. This process continues until the data packets reach the destination node v_n.

Although activation process starts before data delivery, it is likely in some extreme cases that activation frame may arrive at node v_{i+1} after its intended time (i.e., $t^{DP}_j + i \cdot B_j \cdot T^w_{D/A}$) for data delivery over DP, as showed in Fig. 6b. In this case, node v_i can simply keep awake until it receives an AWAKE frame from node v_{i+1}, after it has finished the data transmission with node v_{i-1}. Once node v_{i+1} receives the activation frame, it wakes up immediately to send an AWAKE frame and then retrieve the data packets from node v_i.

Note that it has been found via our evaluations that such case occurs rarely due to the following reason. Moreover, in our system, activation frames are allowed to be forwarded from one node to the next without being ACKed, though the transmission of ACK is designed for reliability purposes.



(a) Case I: Activation frame arriving before DP



(b) Case II: Activation frame arriving after DP

Fig. 6: Data transmission scheduling over DP

Therefore, an activation frame could be many hops ahead of the node that is currently transmitting ACK, which indicates that the actual speed of activation processes in practice could be much faster than our design expects.

C. Flow-wide Pipeline Scheduling

Since $\bar{N}_{i,i+1|j}$ is allocated by the destination node based on flow-wide information, it may need to be reallocated to optimize the system performance when network condition changes. In DIPS, such a flow-wide pipeline scheduling is conducted once a DP is completed on flow f_j, when the destination node runs Alg. 1 using the information (i.e., ZigBee link quality $p_{i-1,i|j}$, conflict rate $\hat{c}^z_{i|j}$ and extra contention delay $\hat{d}_{i,i+1|j}$) piggybacked by each node v_i in activation frames.

1) Scheduling DP Contention Window: For any flow f_j, the destination node records a certain number (e.g., 100 in this paper) of the most recent time difference between the arrival of the *first* data packet over a DP and its expected deadline. Let Δ_{meet} and Δ_{miss} be the sum of the time difference recorded when the first data packet meets and misses its deadline, respectively. Then, the DP contention window is calibrated as $m_j = m^{max}_j \cdot (1 + \frac{\Delta_{miss}}{\Delta_{meet}+\Delta_{miss}})$ in Line 3. On one hand, when the number of packets that miss their deadlines becomes larger, the DP contention window is increased. On the other hand, since $0 \leqslant \frac{\Delta_{miss}}{\Delta_{meet}+\Delta_{miss}} \leqslant 1$, m_j is always no less than m^{max}_j so as to ensure high delay meet ratio.

2) Scheduling Expected EXT: To make use of the guidelines given by Eq. (9) for scheduling expected EXT $\bar{N}_{i,i+1|j}$ in practice, we shall consider the actual bound of $N_{i,i+1|j}$ defined by Eq. (10): $N_{i,i+1|j} = 1$ (i.e., the minimum value of $N_{i,i+1|j}$ with $w_{i,i+1|j} = 1$, indicating using Wi-Fi only) and $N_{i,i+1|j} = \frac{1}{p_{i,i+1|j}\Omega^z_{i,i+1|j}}$ (i.e., the maximum value of $N_{i,i+1|j}$ with $w_{i,i+1|j} = \infty$, indicating using ZigBee only). For the whole flow, we have $\sum_{i=0}^{n_j-1} N_{i,i+1|j} \leqslant \sum_{i=0}^{n_j-1} \frac{1}{p_{i,i+1|j}\Omega^z_{i,i+1|j}} = N^{max}_j$, where N^{max}_j is the maximum total number of EXTs on flow f_j in Line 4. It is likely that $N^{max}_j \leqslant N^{opt}_j$ in

Algorithm 1 Flow-wide pipeline scheduling on flow f_j

```
1:  Let $\mathbf{V}_j = f_j - \{v_{n_j}\}$, where $v_{n_j}$ is the destination node of flow $f_j$
2:  Let $\bar{N}^{new}_{i,i+1|j}$ and $m^{new}_j$ be the new value of $\bar{N}_{i,i+1|j}$ and $m_j$, respectively
3:  Compute $m^{new}_j = m^{max}_j \cdot (1 + \frac{\Delta_{miss}}{\Delta_{meet}+\Delta_{miss}})$
4:  Compute $N^{max}_j = \sum_{i=0}^{n_j-1} \frac{1}{p_{i,i+1|j}\Omega^z_{i,i+1|j}}$
5:  /* All nodes can use ZigBee only */
6:  if $N^{max}_j \leqslant N^{opt}_j$ then
7:      Set $\bar{N}^{new}_{i,i+1|j} = \frac{1}{p_{i,i+1|j}\Omega^z_{i,i+1|j}}$ for all $v_i \in \mathbf{V}_j$
8:      /* Some nodes need to use Wi-Fi interface */
9:  else
10:     repeat
11:         $DONE$ = true
12:         for all $v_i \in \mathbf{V}_j$ do
13:             Set $\bar{N}^{new}_{i,i+1|j} = N^{opt}_j \cdot \frac{\eta_{i,i+1|j}}{\sum_{k=0}^{n_j-1}\eta_{k,k+1|j}}$
14:             if $\bar{N}^{new}_{i,i+1|j} \leqslant 1$ then
15:                 Reset $\bar{N}^{new}_{i,i+1|j} = 1$
16:                 $N^{opt}_j = N^{opt}_j - \bar{N}^{new}_{i,i+1|j}$
17:                 $\mathbf{V}_j = \mathbf{V}_j - \{v_i\}$ and $DONE$ = false
18:             end if
19:             if $\bar{N}^{new}_{i,i+1|j} \geq \frac{1}{p_{i,i+1|j}\Omega^z_{i,i+1|j}}$ then
20:                 Reset $\bar{N}^{new}_{i,i+1|j} = \frac{1}{p_{i,i+1|j}\Omega^z_{i,i+1|j}}$
21:                 $N^{opt}_j = N^{opt}_j - \bar{N}^{new}_{i,i+1|j}$
22:                 $\mathbf{V}_j = \mathbf{V}_j - \{v_i\}$ and $DONE$ = false
23:             end if
24:         end for
25:     until $DONE$
26: end if
27: if $\sqrt{\frac{\sum_{i=0}^{n_j-1}(\bar{N}^{new}_{i,i+1|j} - \bar{N}_{i,i+1|j})^2}{n_j}} > \theta \cdot \frac{\sum_{i=0}^{n_j-1}\bar{N}_{i,i+1|j}}{n_j}$ then
28:     Adjust $\bar{N}_{i,i+1|j}$ and $m_j$ to $\bar{N}^{new}_{i,i+1|j}$ and $m^{new}_j$, respectively
29: end if
```

Line 6, in which case all nodes are simply allowed to use their ZigBee interfaces only in order to optimize the performance. Otherwise, some nodes on flow f_j need to leverage their Wi-Fi interfaces so as to render the resulting value of $\sum_{i=0}^{n_j-1} N_{i,i+1|j}$ to approach N^{opt}_j for optimized performance.

During the allocation of N^{opt}_j to each link (v_i, v_{i+1}) in Line 10–25, there are two special cases: (i) the newly allocated value of $\bar{N}_{i,i+1|j}$ (i.e., $\bar{N}^{new}_{i,i+1|j}$) is smaller than the minimum value of $N_{i,i+1|j}$ in Line 14, and (ii) $\bar{N}^{new}_{i,i+1|j}$ is larger than the maximum value of $N_{i,i+1|j}$ in Line 19. In these two cases, we simply allocate the link with the minimum and the maximum value of $N_{i,i+1|j}$, respectively. Then, link (v_i, v_{i+1}) is done with the allocation and node v_i is removed from $\mathbf{V}_j$. Otherwise, the guideline specified in Eq. (9) is used to allocate $\bar{N}^{new}_{i,i+1|j}$ in Line 13. The above allocation is repeated, until no node in $\mathbf{V}_j$ encounters the above two cases in Line 25.

Besides, we define a metric to measure the difference between the current and the new expected EXT in Line 27. If the difference is greater than a certain ratio of θ, a flow-wide adjustment is performed. Otherwise, no adjustment is conducted to avoid unnecessary adjustments. Here, θ is a system parameter, called *flow adjusting threshold*, to control the sensitivity of flow-wide scheduling to the variation of network conditions.

3) Backward Notification: To notify each node on the flow of $\bar{N}^{new}_{i,i+1|j}$ and m^{new}_j, the destination node sends a NOTIFY frame backwards over AP by leveraging the unused pair of aligned rx-slot (at node v_i) and tx-slot (at node v_{i-1}). The transmission of NOTIFY frame is similar to that of activation frame, expect that it takes at least one AP interval time per hop. Moreover, in order not to interfere with normal data transmission, each node v_i does not update its schedule immediately after it has received the NOTIFY frame. Instead, the source node v_0 indicates the execution of the new schedule in the data packets. Once each node v_i on the flow has completed the data delivery over the current DP, it can then safely update its schedule. Note that pipelining backoff b_j is computed by the source node using Eq. (4).

D. Practical Issues

1) Estimating ZigBee Link Quality: To estimate the ZigBee link quality between two neighboring nodes, each node periodically (e.g., several AP intervals) sends a PROBE frame to its next-hop node in tx-slots via the ZigBee interface. Specifically, node v_{i+1} knows how many PROBE frames its previous node v_i should send in a given time window π (e.g., 2 mins in this paper). ZigBee link quality $p_{i,i+1}$ can be estimated by node v_{i+1} as the number of PROBE frames received by node v_{i+1} divided by the expected number of PROBE frames sent by node v_i during π. Besides, some techniques, such as exponential moving average [12], are used to further improve the estimation accuracy.

2) Synchronizing Time & Supporting Broadcast: Similar to the standard PSM, time synchronization can be achieved through periodic beacon exchanges via ZigBee and/or Wi-Fi interface. Moreover, the periodic beacon frames can also be leveraged to wake up neighboring nodes, which provides an opportunity for message broadcast. Note that in case the slots for beacon exchanges overlap with the tx- or rx-slots used by our scheme, we can simply defer the affected slots by one slot.

3) Handling Collided Activations: In some extreme cases, the schedule of one flow may completely overlap with that of the other. When activation frames on these two flows happen to arrive at the intersection node at the same slot on ZigBee channel, they may always get lost due to collisions. If such collisions occur, ZigBee link quality drops rapidly and thus causes schedule changes (i.e., decreasing w-interval), which can mitigate the conflicts. Further, it is worth pointing out that if such collisions occur on Wi-Fi channel, the impact is insignificant, as one wake-up slot is long enough for the Wi-Fi interface to resolve the collisions following the underlying 802.11 MAC protocols.

VI. Simulation

To evaluate the system performance, we conduct extensive simulations on ns-3.28, where two performance metrics (i.e., network energy consumption and delay meet ratio) are measured. Note that delay meet ratio should be no greater than packet delivery ratio (which is a commonly-used performance metric for multi-hop communications). Hence, delay meet ratio not only reflects the degree to which the end-to-end delay requirements are satisfied, but also provides a referrible lower bound for the corresponding packet delivery ratio. As shown in Table III, the energy consumed by Wi-Fi and ZigBee interfaces is measured according to the specifications of SX-SDWAG Wi-Fi module [27] and ZigBee CC2530 RF transceiver [28], respectively. The data packet size is fixed at 512 bytes [1]. The required end-to-end delay bound of flow f_j is specified as $n_j \times D_{Hop}$, where n_j is the number of links on flow f_j and D_{Hop} is called *per-hop delay bound*.

TABLE III: DEFAULT SETTINGS OF SYSTEM PARAMETERS

Wi-Fi channel bit rate	54 Mbps	ZigBee channel bit rate	250 Kbps
Wi-Fi transmission	1.152 Watt	ZigBee transmission	0.087 Watt
Wi-Fi reception	0.561 Watt	ZigBee reception	0.072 Watt
Wi-Fi idle listening	0.462 Watt	ZigBee idle listening	0.019 Watt
Wake-up slot	20 ms	ZigBee link quality (p)	0.7
Data packet size	512 bytes	Packet arrival rate λ	5 pkt/s
AP Interval	300 ms	Per-hop delay bound	300 ms

In the simulation, 100 nodes are randomly distributed in a 1000 m $\times$ 1000 m area, where 6 flows of 10 to 20 hops are randomly generated. IEEE 802.11g is used for Wi-Fi and the range of both Wi-Fi and ZigBee interfaces are 100 m. The data traffic is generated by the source nodes, based on the source traffic model for wireless applications proposed in [29]. The traffic generation is controlled by the average packet arrival rate λ. The ZigBee link quality of each link is randomly chosen between $p-0.2$ and $p+0.2$ every certain time interval following an Exponential distribution with the mean of 1 min [30], where p is the average ZigBee link quality.

A. Study on System Parameters

To study how system parameters affect the system, we evaluate the system performance by varying one parameter while fixing the other. The results are shown in Fig. 7.

1) Adjustable Window Ratio δ: δ is used to control the size of the adjustable window, in which w-interval is allowed to be adjusted locally. From Fig. 7a, we can observe that the energy consumption is at the minimum when $\delta = 0.12$ in different scenarios. After we investigated the detailed simulation results, the reasons are found to be as follows.

- When δ becomes small, not only the adjustable window but also the optimal zone (which is located in the center of the adjustable window and half of its length) gets small. In this case, the actual EXT (which is desired to vary within the optimal zone) may easily move out of the optimal zone due to network dynamics, triggering frequent local adjustments of w-interval and thus incurring lots of energy overheads.
- When δ becomes large, the adjustable window and the optimal zone get large simultaneously. As a result, the optimal zone (in which the actual EXT varies) is far from the rightmost side of the adjustable window (which corresponds to the expected EXT that is normally fixed). This can indirectly lower the actual EXT and thus results in smaller w-interval, which eventually causes more frequent use of Wi-Fi interface and lower energy efficiency.

2) Flow Adjusting Threshold θ: θ is designed to control the sensitivity of flow-wide pipeline scheduling. Similarly, the energy consumption first decreases and then increases as θ becomes larger, as plotted in Fig. 7b. The minimum energy consumption occurs when $\theta = 0.08$. Particularly, when θ is small, the flow-wide scheduling is conducted frequently, thus consuming lots of energy in transmitting relevant control frames; when θ is large, the flow-wide scheduling cannot be conducted in time to minimize energy consumption.

Also, we measured the delay meet ratio as the above system parameters vary. The resulting delay meet ratio is insensitive to the change of these two parameters and stays above 0.965, due to the effectiveness of local adjustment of w-interval.

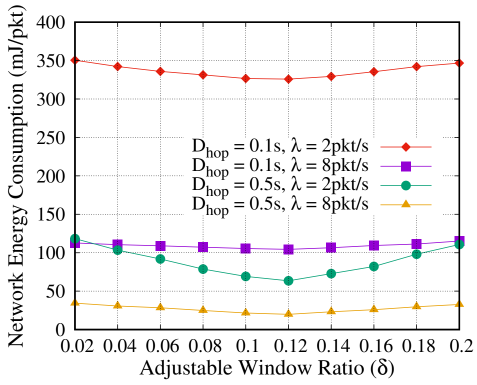
From the above, we can see that the system parameters θ and δ shall be properly set in order to obtain low energy consumption given a certain system setting. However, as the impact of these parameters on system performance is not significant, we leave the study of how to dynamically choose their optimal values under various scenarios in our future work. For the follow-up simulations and experiments, we choose $\delta = 0.12$ and $\theta = 0.08$ by default.

B. Study on System Performance

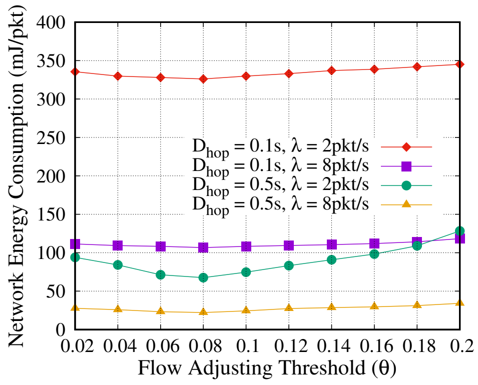
With the above chosen settings, we study the system performance with different settings.

1) ZigBee Link Quality: In our system, ZigBee link quality not only reflects the success rate of ZigBee communications but also determines the usage frequency of Wi-Fi interface. From Fig. 8a, we can see that as ZigBee link quality goes up the network energy consumption decreases significantly, due to less frequent use of high-power Wi-Fi interface for pipeline activation. This demonstrates the advantage of utilizing ZigBee interface for pipeline activation. In contrast, as shown in Fig. 8b, the delay meet ratio drops insignificantly as ZigBee link quality decreases, since high-power Wi-Fi is leveraged to assist activation process in order to ensure end-to-end delay requirements. Moreover, to reveal the detailed delay performance of our system, we plot the CDF of the *normalized end-to-end delay* with the default settings in Fig. 8c, where normalized end-to-end delay is defined as the end-to-end delay measured at all destination nodes divided by the corresponding end-to-end delay bound. From the results, we can observe that the CDF values of the normalized end-to-end delay under different ZigBee link qualities are very similar, which indicates the effectiveness and the responsiveness of DIPS in switching between Wi-Fi and ZigBee interfaces in order to sustain the desired delay performance.

2) AP Interval: In Fig. 9, we plot the energy consumption of Wi-Fi/ZigBee interface and the delay meet ratio with varying AP interval, from which we can observe a tradeoff between energy and delay, i.e., longer AP interval results in lower energy consumption but smaller delay meet ratio. This is because with longer AP interval all nodes wake up less frequently but at the same time fewer transmissions of activation frame are allowed across the flow. From Fig. 9a, we can see that the energy consumption of the ZigBee interface in our system is 74.1% to 83.3% lower than that of the Wi-Fi interface, which demonstrates the energy-saving advantage

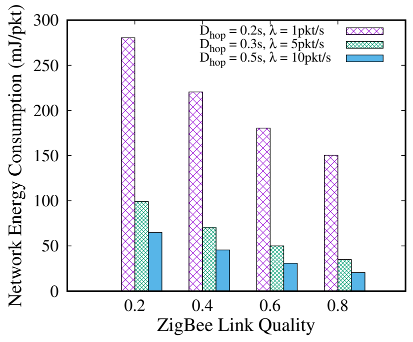


(a) Adjustable window ratio (δ)

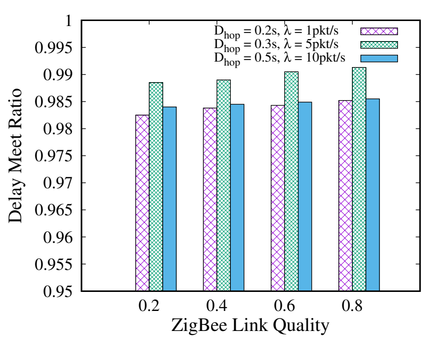


(b) Flow adjusting threshold (θ)

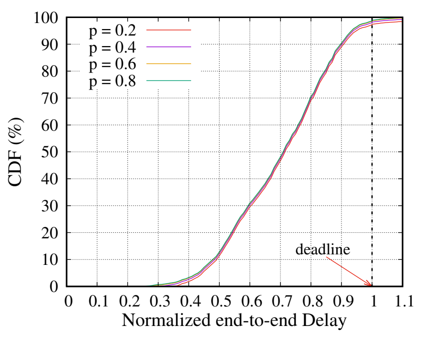
Fig. 7: Performance under different system parameters



(a) Network Energy Consumption



(b) Delay meet ratio



(c) CDF of normalized end-to-end delay ($D_{hop} = 0.3$ s and $\lambda = 5$ pkt/s)

Fig. 8: Performance under different ZigBee link qualities

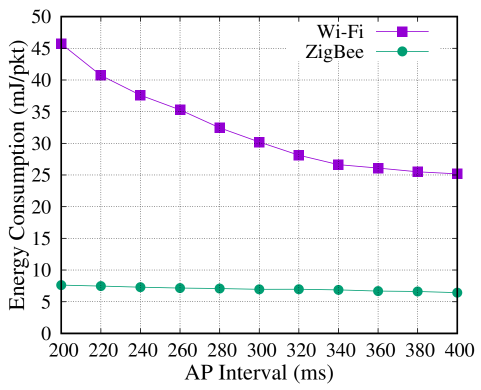
of ZigBee interface over the Wi-Fi interface. Particularly, by taking advantage of the pipelining technique, DIPS can still achieve a delay meet ratio as high as 0.977 with the AP interval of 400 ms, which is even greater than the required per-hop delay bound D_{Hop} (i.e., 300 ms), as shown in Fig. 9b.

3) Link Transmission Share: Link transmission share $\eta_{i,i+1|j}$ is defined for allocating expected EXT to each link (v_i, v_{i+1}) on flow f_j in Eq. (9). In our simulation, we compare DIPS (choosing $\eta_{i,i+1|j} = [(1-p_{i,i+1|j})\Omega_{i,i+1|j}]^{-1/2}$) with three other naive schemes, i.e., Naive-1, Naive-2 and Naive-3, which use $\eta_{i,i+1|j}$ of $[(1-p_{i,i+1|j})\Omega_{i,i+1|j}]^{-1}$, $p_{i,i+1|j}^{-1}$ and 1, respectively. From the results shown in Fig. 10, we can see that DIPS outperforms the other three naive schemes in terms of both energy efficiency and delay meet ratio. Specifically, the energy efficiency of Naive-1 is slightly lower that of DIPS, as it produces an EXT allocation pattern that is closest to DIPS. Naive-2 allocates the link of lower ZigBee link quality with larger expected EXT and thus longer w-interval, which is opposed to DIPS's idea of choosing shorter w-interval to increase the wake-up frequency of Wi-Fi interface when ZigBee link quality gets lower. Moreover, too frequent use of ZigBee interface with low link quality can lead to long delay and thereby decreases the delay meet ratio. Thus, Naive-2 delivers the worst performance. Besides, Naive-3 yields a fair share of expected EXT among all links without considering ZigBee link quality and inter-flow conflicts, thus failing to optimize the system performance.

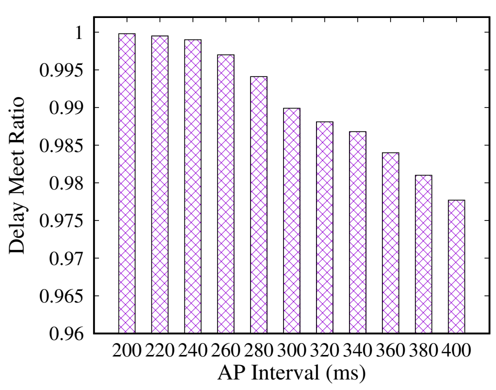
C. Comparison

We compare DIPS with PSM and MH-PSM [1]. MH-PSM is a state-of-the-art pipeline-based scheme built on top of PSM. It allows ATIM frame and thus data packets to travel multiple hops in a single beacon interval (BI) so as to reduce energy consumption and end-to-end delay. Besides, for fairness purpose, the BI of PSM is set to the maximum value (i.e., D_{hop}) such that the end-to-end delay bound is satisfied.

1) Varying Per-hop Delay Bound: The results plotted in Fig. 11a show that DIPS outperforms the other two schemes in terms of energy efficiency, especially when the required per-hop delay bound becomes larger (thus Wi-Fi interface is used less frequently). This is because both schemes suffer from the unnecessary wake-ups for ATIM exchanges even when there is no incoming traffic, which is the major shortcoming that exists inherently in single-interface-based approaches. Since MH-PSM is a best-effort scheme with minimal end-to-end delay, its energy consumption does not react to the change of per-hop delay bound. Although the delay meet ratio of MH-PSM is slightly higher than that of DIPS as shown in Fig. 11b, it consumes much more energy. Moreover, we note that when the BI of PSM (i.e., D_{Hop}) is greater than the inter-

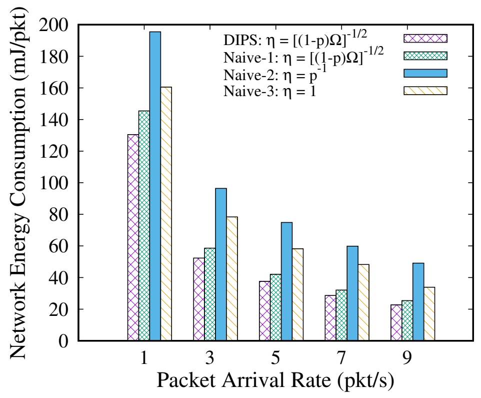


(a) Energy consumption of Wi-Fi and ZigBee

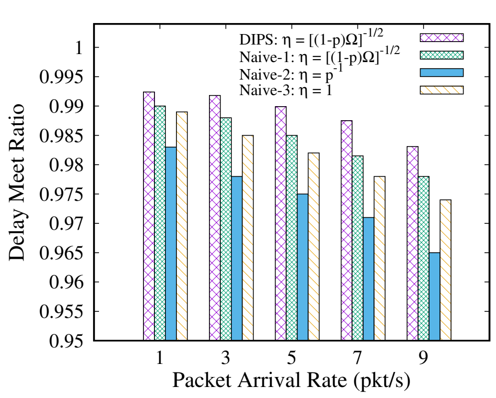


(b) Delay Meet Ratio

Fig. 9: Performance under different AP intervals



(a) Network Energy Consumption



(b) Delay Meet Ratio

Fig. 10: Performance under different link transmission shares

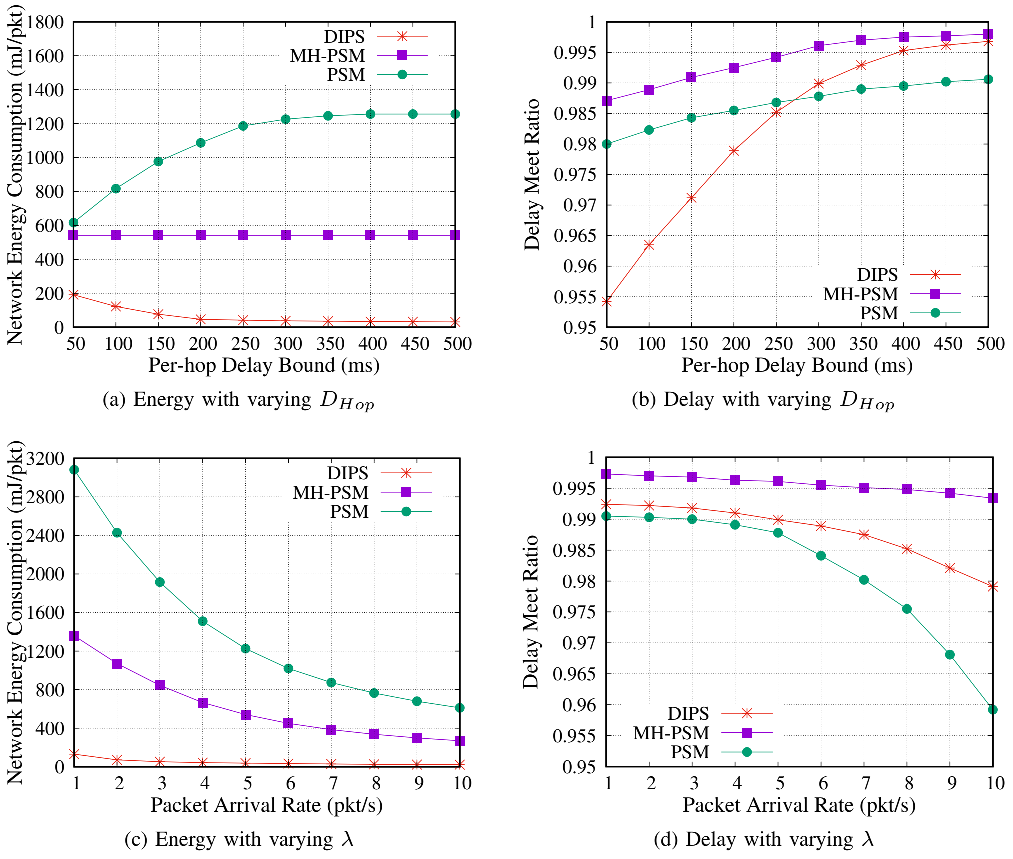
packet arrival time (i.e., 200 ms), PSM nodes need to wake up almost every BI, consuming a large amount of energy.

2) Varying Packet Arrival Rate: From Fig. 11c, we can see that the energy consumption of all three schemes increases as packet arrival rate drops, because fewer data packets can be buffered for each DP, which leads to more energy costs per packet. Due to the adaptability to traffic changes, DIPS can deliver much higher energy efficiency, especially in low traffic scenarios. As for delay meet ratio shown in Fig. 11d, MH-PSM outperforms the other two schemes, since it transmits data packets without buffering and thus yields the shortest end-to-end delay. Nevertheless, DIPS can still achieve a delay meet ratio above 0.979 when the packet arrival rate is 10 pkt/s. The delay meet ratio of PSM dramatically drops when packet arrival rate becomes high. This is because PSM cannot adapt to high traffic scenarios, where the delay caused by intensive contentions gets accumulated hop by hop and eventually leads to the violation of end-to-end delay requirements.

3) Conclusions: With the default settings, MH-PSM reduces the energy consumption of PSM by 51.4% due to the adoption of single-interface single-pipeline technique, while DIPS can further lower the energy consumption of MH-PSM by 92.8% through the introduction of an additional activation pipeline controlled by the additional ZigBee interface (i.e., dual-interface dual-pipeline technique). In sum, with the default settings, DIPS can achieve a delay meet ratio of 0.983, and its energy consumption is 96.5% and 92.8% lower than that of PSM and MH-PSM, respectively.

VII. Implementation

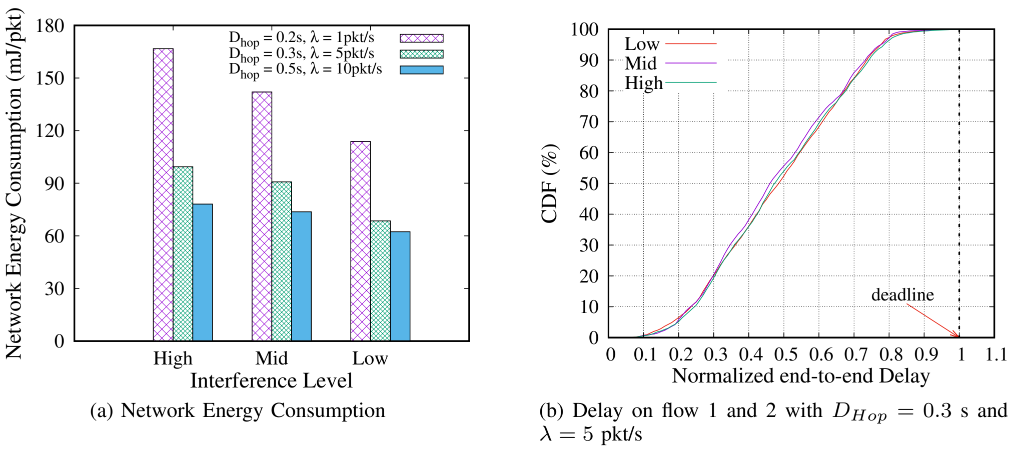
As a proof of concept, we implemented a prototype of our proposed DIPS system in a testbed network of ten IoT devices. Each IoT device consists of a BeagleBone Green Wireless device [31], a Digi XBee S1 Dongle [32] and an Edimax N150 Wi-Fi adapter [33]. Although the BeagleBone Green device has an on-board Wi-Fi chip, its ad hoc function has been disabled in the shipped kernel. Therefore, the external Wi-Fi adapter is added to enable ad hoc Wi-Fi communications. The device software is developed using JAVA to leverage the XBee's JAVA library support. On system start, the Wi-Fi and ZigBee radios are configured to operate on channel 3 and 20, respectively. The purpose is to reduce the uncontrollable interference from surrounding environments (Wi-Fi channel 1, 6, and 11) and the system itself. The prototype IoT devices form up a network as shown in Fig. 12b and three data flows are pre-configured in the network. Unless otherwise specified, the experiments use the same default settings as the simulation.



(a) Energy with varying D_{Hop} (b) Delay with varying D_{Hop}

(c) Energy with varying λ (d) Delay with varying λ

Fig. 11: Comparison with other schemes



(a) Network Energy Consumption

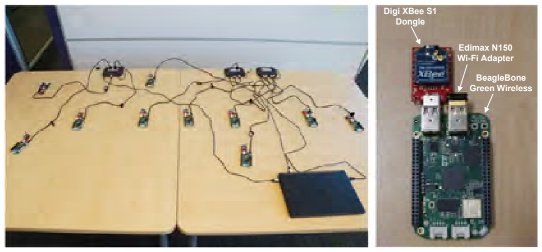
(b) Delay on flow 1 and 2 with $D_{Hop} = 0.3$ s and $\lambda = 5$ pkt/s

Fig. 13: System performance

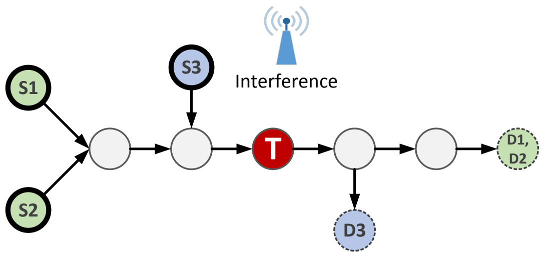
A. Performance Study

To study the impact of interference on our proposed system, we introduced an interference node, which generates valid activation frames on ZigBee and data packets on Wi-Fi at certain intervals to create channel contentions on purpose. Such additional activation traffic can not only intensify the inter-flow contentions/conflicts that inherently exist inside our system, but also emulate the potential background interference from the outside of the system. In particular, the packet generation intervals for creating high, mid and low interference levels are 0.1 s, 0.3 s and 1 s, respectively.

1) Network Energy Consumption: From Fig. 13a, it can be seen that with the decrease of interference level, the energy overheads per packet drop due to less (more) usage of Wi-Fi (ZigBee) for transmitting activation frames. Besides, as the packet arrival rate becomes larger, the energy efficiency of DIPS gets higher, due to pipelined data delivery and usage of ZigBee on transmitting activation frames. These results show a similar trend as we observe from Fig 8a in the simulation. This is because the decrease (increase) of interference level and the increase (decrease) of ZigBee link quality can *equivalently*

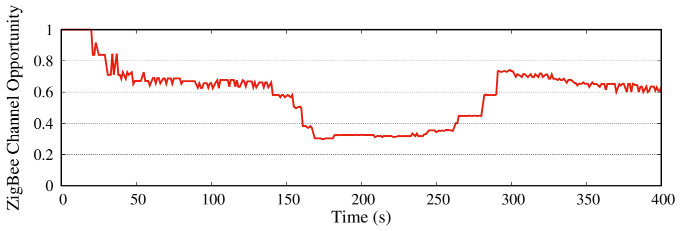


(a) Testbed network and prototype dual-interface devices

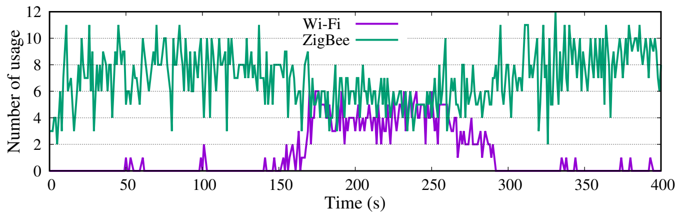


(b) Network topology. Flow 1: $S1 \rightarrow D1$; Flow 2: $S1 \rightarrow D2$; Flow 3: $S3 \rightarrow D3$

Fig. 12: Prototype system



(a) Trace of ZigBee channel opportunity



(b) Trace of number of usage

Fig. 14: Trace of Wi-Fi and ZigBee usage with varying interference when $D_{Hop} = 0.3$ s and $\lambda = 5$ pkt/s

increase (decrease) the success probability of activation frame transmission in our system.

2) Delay Meet Ratio: To compare with the simulation results plotted in Fig. 8c, we also recorded the CDF of the normalized end-to-end delay at all destination nodes. Fig. 13b shows the results collected from the destination nodes of flow 1 and 2, which are the two flows of equal length and thus have the equal end-to-end delay bound of 1.8 s in our experiments. Particularly, any packet received with a normalized end-to-end delay greater than 1 (corresponding to the dotted vertical line) has a delay larger than 1.8 s and thus fails to meet its end-to-end delay bound. From the results, we can see that the CDF of normalized end-to-end delay under different interference levels (equivalently corresponding to different ZigBee link qualities) are very similar, which is consistent with the simulation results plotted in Fig. 8c. This is because more energy are sacrificed (through using Wi-Fi interface more frequently) to ensure end-to-end delay performance when the interference level is high.

B. Trace Study

To illustrate how DIPS uses Wi-Fi and ZigBee interfaces to optimize energy efficiency in detail, we plot in Fig. 14a the traces of the estimated ZigBee channel opportunity (i.e., $\Omega^z_{i,i+1}$ defined in Eq. (10)), which essentially reflects the probability that ZigBee channel is available for transmitting activation frame from one node to another. Optimally, ZigBee (Wi-Fi) interface should be utilized more (less) frequently as ZigBee channel opportunity becomes higher, in order to minimize energy consumption. Besides, the number of Wi-Fi and ZigBee usages on transmitting activation frames with the varying interference levels were also recoded and plotted in Fig. 14b. The results are obtained from the observer node "**T**" in Fig. 12b, which delivers data for three flows and is close to the interference node.

In the experiment, the interference is generated at the low level at first, then increased to the high level after time 150 s; after being kept high for 120 s, it is tuned down to the low level again at time 270 s. Combining Fig. 14a and 14b, we can see that at about the same time when interference is increased (at time 150 s), the observed ZigBee channel opportunity starts dropping due to increased inter-flow contentions/conflicts and interference; correspondingly, the usage of Wi-Fi increases while that of ZigBee decreases. At about time 270 s, the ZigBee channel opportunity increases with the decrease of interference, and so does the usage of ZigBee. This trace well demonstrates how DIPS can work adaptively under varying interference levels by switching between Wi-Fi and ZigBee interfaces, while achieving high energy efficiency and desired end-to-end delay.

VIII. Conclusions and Future Work

Towards enabling Wi-Fi for multi-hop communications in IoT, this paper proposes a dual-interface dual-pipeline scheduling scheme. The core idea is to leverage low-power ZigBee interface to construct pipelines to activate and schedule high-power Wi-Fi interface for fast pipelined data delivery. Extensive simulations and prototype experiments have verified the performance advantages of the proposed DIPS scheme.

To apply DIPS to a wider range of IoT applications, we plan to conduct further study to improve the adaptability and the reliability of our proposed schemes. For example, (i) how to adjust AP interval to allow our system to automatically adapt to IoT applications with varying delay requirements while retaining high energy efficiency, and (ii) how to schedule EXT in order to achieve a fair utilization of different IoT devices and thus improve the sustainability of flow/network would be interesting future directions.

References

[1] V. Vukadinovic, I. Glaropoulos, and S. Mangold, "Enhanced power saving mode for low-latency communication in multi-hop 802.11 networks," *Ad Hoc Networks*, vol. 23, no. December 2014, pp. 18–33, 2014.

[2] B. Ostermaier, M. Kovatsch, and S. Santini, "Connecting things to the web using programmable low-power wifi modules," in *International Workshop on Web of Things*, 2011, p. 2.

[3] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-fi enabled sensors for internet of things: A practical approach," *Communications Magazine IEEE*, vol. 50, no. 6, pp. 134–143, 2012.

[4] E. S. Jung and N. H. Vaidya, "Improving ieee 802.11 power saving mechanism," *Wireless Networks*, vol. 14, no. 3, pp. 375–391, 2008.

[5] Y. Shi and T. A. Gulliver, "An energy-efficient mac protocol for ad hoc networks," *Wireless Sensor Network*, vol. 1, no. 5, pp. 407–416, 2009.

[6] C. Mala, "Modified power save model for better energy efficiency and reduced packet latency," *American Journal of Engineering & Applied Sciences*, vol. 5, no. 3, pp. 237–242, 2012.

[7] K. R. Malekshan, W. Zhuang, and Y. Lostanlen, "An Energy Efficient MAC Protocol for Fully Connected Wireless Ad Hoc Networks," *IEEE Transactions on Wireless Communications*, 2014.

[8] D. N. M. Dang, M. V. Nguyen, C. S. Hong, S. Lee, and K. Chung, "An energy efficient multi-channel mac protocol for wireless ad hoc networks," in *International Conference on Advanced Engineering Theory and Applications*, 2016, pp. 822–830.

[9] X. Zhang and G. S. Kang, "E-mili: Energy-minimizing idle listening in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 9, pp. 1441–1454, 2012.

[10] D. N. M. Dang and C. S. Hong, "H-mmac: A hybrid multi-channel mac protocol for wireless ad hoc networks," in *IEEE International Conference on Communications*, 2012, pp. 6489–6493.

[11] Y. Zhang and Q. Li, "Howies: A holistic approach to zigbee assisted wifi energy savings in mobile devices," in *IEEE INFOCOM*, 2013, pp. 1366–1374.

[12] H. Qin and W. Zhang, "ZigBee-assisted Power Saving Management for Mobile Devices," *Mobile Computing IEEE Transactions on*, vol. 13, no. 12, pp. 2933–2947, 2013.

[13] J. W. Yoo and K. H. Park, "A cooperative clustering protocol for energy saving of mobile devices with wlan and bluetooth interfaces," *IEEE Transactions on Mobile Computing*, vol. 10, no. 4, pp. 491–504, 2010.

[14] T. Jin, G. Noubir, and B. Sheng, "Wizi-cloud: Application-transparent dual zigbee-wifi radios for low power internet access," in *IEEE INFOCOM*, 2011, pp. 1593–1601.

[15] Z-wave alliance. [Online]. Available: http://z-wavealliance.org/

[16] B. Raman, K. Chebrolu, S. Bijwe, and V. Gabale, "Pip:a connection-oriented, multi-hop, multi-channel tdma-based mac for high throughput bulk transfer," in *International Conference on Embedded Networked Sensor Systems, SENSYS 2010, Zurich, Switzerland, November*, 2010, pp. 15–28.

[17] Y. Cao, S. Guo, and T. He, "Robust multi-pipeline scheduling in low-duty-cycle wireless sensor networks," in *INFOCOM, 2012 Proceedings IEEE*, 2012, pp. 361–369.

[18] W. Du, J. C. Liando, H. Zhang, and M. Li, "When pipelines meet fountain: Fast data dissemination in wireless sensor networks," *Vldb Journal*, vol. 7, no. 3, pp. 163–178, 2015.

[19] M. H. Seo, H. J. Yoon, and S. J. Ma, *Fast Flooding in Power Save Mode of IEEE 802.11 DCF Based Mobile Ad Hoc Networks*. Springer Berlin Heidelberg, 2004.

[20] C. M. Chao, J. P. Sheu, and I. C. Chou, "An adaptive quorum-based energy conserving protocol for ieee 802.11 ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 5, pp. 560–570, 2006.

[21] Y. Kondo, H. Yomo, and et al., "Energy-efficient wlan with on-demand ap wake-up using ieee 802.11 frame length modulation," *Computer Communications*, vol. 35, no. 14, pp. 1725–1735, 2012.

[22] S. Tang, H. Yomo, Y. Kondo, and S. Obana, "Wake-up receiver for radio-on-demand wireless lans," in *Global Telecommunications Conference*, 2012, pp. 1–6.

[23] Y. Kondo, H. Yomo, and et al., "Wake-up radio using ieee 802.11 frame length modulation for radio-on-demand wireless lan," in *IEEE International Symposium on Personal Indoor and Mobile Radio Communications*, 2012, pp. 869–873.

[24] Z. Shelby and C. Bormann, "6lowpan: The wireless embedded internet - part 1: Why 6lowpan?" *EE Times Com*, 2011.

[25] G. Ananthanarayanan and I. Stoica, "Blue-fi:enhancing wi-fi performance using bluetooth signals," in *International Conference on Mobile Systems, Applications, and Services*, 2009, pp. 249–262.

[26] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma, "Zifi:wireless lan discovery via zigbee interference signatures," in *International Conference on Mobile Computing and Networking*, 2010, pp. 49–60.

[27] Silex Technology America, Inc., *www.silexamerica.com*.

[28] Texas Instruments, "CC2530 datasheet (Rev. B)," http://www.ti.com/lit/ds/symlink/cc2530.pdf.

[29] P. Tran-Gia, D. Staehle, and K. Leibnitz, "Source traffic modeling of wireless applications," *AEU - International Journal of Electronics and Communications*, vol. 55, no. 1, pp. 27–36, 2001.

[30] M. Zeeshan, A. Ali, A. Naveed, A. X. Liu, A. Wang, and H. K. Qureshi, "Modeling packet loss probability and busy time in multi-hop wireless networks," *Eurasip Journal on Wireless Communications and Networking*, vol. 2016, no. 1, p. 168, 2016.

[31] BeagleBoard.org, "SeeedStudio BeagleBone Green Wireless," https://beagleboard.org/green-wireless.

[32] Digi International, "XBee S1 802.15.4 RF Moudles Datasheet," https://www.digi.com/pdf/ds_xbeemultipointmodules.pdf.

[33] Edimax, "EW-7811Un Datasheet (English)," http://www.edimax.com/edimax/mw/cufiles/files/download/datasheet/EW-7811Un_Datasheet_English.pdf.

Hua Qin received his PhD degree in Computer Science from Iowa State University, USA, in 2013. Since 2015, he has been a faculty member with the College of Information and Electronic Engineering at Hunan City University, Hunan, China. His research interests include Internet of things, vehicular networks and heterogeneous wireless systems.

Weihong Chen received her MS degree in Information Science and Engineering from Hunan University, China, in 2006. Since 1999, she has been a faculty member with the College of Information and Electronic Engineering at Hunan City University, Hunan, China, where she is now a professor. Currently, she is also working on her PhD degree in Information Science and Engineering at Hunan University. Her research interests include wireless networks, cloud computing and big data.

Buwen Cao received his PhD degree in Information Science and Engineering from Hunan University, China, in 2016. Since 2016, he has been a faculty member with the College of Information and Electronic Engineering at Hunan City University, Hunan, China. His research interests include computer networks and bioinformatics.

Min Zeng received her PhD degree in Computer Science from Chosun University, South Korea, in 2011. Since 2013, she has been a faculty member with the College of Information and Electronic Engineering at Hunan City University, Hunan, China. Her research interests include sensor networks, software-defined networks and distributed systems.

Jessica Li received her Bachelor's degree in Computer Science from University of Washington Bothell, WA, USA, in June 2018. She is currently pursuing her Master's degree in Computer Science at Georgia Institute of Technology, GA, USA. Her research interests include Internet of things and wireless networks.

Yang Peng received his PhD degree in Computer Science from Iowa State University, USA, in 2014. He is currently an Assistant Professor in the Division of Computing and Software Systems, University of Washington Bothell, WA, USA. His research interests include Internet of things, wireless sensor networks and cyber-physical systems.