AdaptiveHART: An Adaptive Real-Time MAC Protocol for Industrial Internet-of-Things

Sihoon Moon, Hyungseok Park, Hoon Sung Chwa, *Member, IEEE*, and Kyung-Joon Park, *Senior Member, IEEE*

***Abstract*—Wireless sensor–actuator networks (WSAN) for real-time reliable communications are being embraced in the Industrial Internet of Things. One key property required for WSAN is *adaptability* with respect to dynamic environments. Most existing studies assume that a network manager computes global schedules and disseminates schedule information to every node whenever the network parameters need to be updated. Such a centralized approach may degrade control performance due to nonreal-time response to dynamic environments. In this article, we introduce a novel time division multiple access MAC protocol, AdaptiveHART, which updates the network parameters in real-time without the dissemination intervals. AdaptiveHART achieves a reduction in adaptation latency up to 74%, maximum schedulable ratio, and control performance up to 75% than the conventional WirelessHART. We assess our protocol using a stochastic response time analysis, which provides an accurate end-to-end delay distribution and a schedulability condition that can guarantee real-time performance in WSAN.**

***Index Terms*—Dynamic environments, Industrial Internet of Things (IIoT), real-time adaptability, stochastic response time analysis (SRTA), wireless sensor-actuator networks.**

I. Introduction

Wireless sensor–actuator networks (WSAN) have been increasingly adopted in the Industrial Internet of Things (IIoT) as a communication technology because of the advantages in reducing deployment and maintenance costs [1]–[3]. In the IIoT, WSAN typically connect sensors, actuators, and controllers as a part of feedback-control loops to continuously interact with the physical world in real-time. WSAN require timely data delivery to guarantee timing requirements of feedback control for system stability [4]. In addition, WSAN must deal with unexpected disturbances which can be categorized into *internal* disturbances within the wireless networks (e.g., link failure) and *external* disturbances from the physical environment (e.g., physical noise). Thus, two key requirements for WSAN are 1) providing *reliable* and *real-time* communications and 2) supporting *adaptability* to dynamic disturbances.

Manuscript received 24 March 2021; revised 28 August 2021 and 22 March 2022; accepted 26 April 2022. Date of publication 2 June 2022; date of current version 26 August 2022. This work was supported in part by the National Research Foundation of Korea (NRF), Korea government (MSIT), under Grant NRF-2019R1A2C1088092 and in part by the Ministry of Science and ICT through the DGIST R&D program under Grant 22-DPIC-16. *(Corresponding authors: Hoon Sung Chwa; Kyung-Joon Park.)*

The authors are with the Department of Electrical Engineering and Computer Science, Daegu Gyeongbuk Institute of Science and Technology, Daegu 42988, South Korea (e-mail: msh0576@dgist.ac.kr; hyungseok@dgist.ac.kr; chwahs@dgist.ac.kr; kjp@dgist.ac.kr).

Digital Object Identifier 10.1109/JSYST.2022.3171962

A wireless standard, WirelessHART (WH) [5], has been introduced for IIoT to provide reliable and real-time communications. WH employs a *centralized* approach in network management. A network manager generates a global transmission schedule and disseminates it among all nodes in the network. All nodes are then time synchronized and communicate in their allotted time slots using a time division multiple access (TDMA) protocol. Time slots are grouped together into superframes, which are periodically repeated with schedule dissemination intervals. Such a centralized approach enables collision-free and deterministic communications.

Building upon the centralized approach, a vast amount of work, e.g., [6]–[10], has been developed for handling both internal and external disturbances. Typically, to cope with internal disturbances, a fixed number of retransmission slots for a packet is allocated when the global schedule is generated. Such redundant packet transmissions can provide reliable communications even in the presence of packet losses [6]–[8]. Meanwhile, to adapt to external disturbances, a typical approach is to change transmission periods [9], [10]. For example, for a physical plant in a transient state due to an external disturbance, a shorter period is assigned to its corresponding network flow.

Although the aforementioned centralized approaches allow reliable real-time communications under dynamic disturbances, all of them have at least one of the following limitations. First, they require a schedule dissemination interval for every superframe during which actual data transmission is delayed. Second, they assume that redundant transmission slots are assigned in a static manner. If no packet loss occurs on a particular link, all redundant time slots that were assigned to handle its failure remain unused, resulting in a considerable waste of time and bandwidth (33% more wasted resources than a flexible resource usage [8]). Third, when adjusting network parameters, i.e., *transmission periods* and the *number of retransmission slots*, the global schedule should be reconfigured and redisseminated. Thus, the reflection of the parameter adjustment is delayed until the next dissemination interval. Such limitations make it practically difficult to adapt to dynamic disturbances in a timely manner, resulting in degraded control performance or even system instability.

In this article, to overcome the above-mentioned limitations, we propose a new MAC protocol called AdaptiveHART (AH), which enables adaptive real-time communications in immediate response to both internal and external disturbances and efficient network resource management while guaranteeing stochastic real-time performance. AH is designed to completely change the way packets are scheduled, i.e., a shift from manager-driven centralized scheduling to node-driven distributed approach. Main contributions of this article are as follows.

1) We develop a novel MAC protocol called AH, which enables adaptive real-time communications under dynamic disturbances in WSAN. To the best of our knowledge, AH is the first algorithm that provides on-demand network reconfiguration in a centralized wireless multihop network. In this process, network resource utilization is significantly improved by saving unused retransmission slots.

2) We develop a stochastic end-to-end communication delay analysis of AH, which gives accurate end-to-end communication delay distributions by taking into account the packet loss probability. This is the first try to apply the stochastic response time analysis (SRTA) in real-time system fields to WSAN. With this analysis, we can provide theoretical real-time performance of AH.

3) We empirically validate the effectiveness of AH by in-depth performance evaluation in terms of adaptation latency, real-time performance, and control performance.

AH adopts a piggyback mechanism to distribute the network parameter configuration of a flow over its path during a packet transmission in contrast to the centralized approach in which the global schedule should be broadcast to entire nodes in a dissemination interval. Furthermore, AH provides a node-driven scheduling mechanism under which each node determines its transmission slots and offsets online with no conflict based on radio events. Thus, AH enables prompt parameter adjustment and on-demand retransmission of a lost packet upon disturbances. Hence, AH not only provides high adaptability to disturbances, but also enables efficient network resource utilization, leading to better control and real-time performance.

Furthermore, we perform stochastic end-to-end communication delay analysis and schedulability testing of AH. We extend the traditional SRTA, which is originally designed for uniprocessor platforms, toward wireless multihop networks. In particular, we model the distribution of the number of packet transmissions for a flow by taking the packet loss probability into account. Then, we derive the end-to-end communication delay distribution of a flow and a probabilistic schedulability condition that guarantees the timing requirements of flows under AH, which can be used for admission control and network parameter adjustment at runtime.

AH achieves a significant improvement regarding adaptability. It achieves a reduction in adaptation latency up to 74% than the conventional WH. In a perspective of resource efficiency, AH shows maximum schedulable ratio and 75% improved control performance.

II. Related Work

A. Consideration of Dynamic Disturbances

Substantial studies have been conducted to adapt network parameters in dynamic environments. For centralized networks, optimal network parameter selection algorithms have been proposed in the presence of physical disturbance and wireless interference [6]–[10]. The approaches in [6]–[8] adaptively select the parameter of the retransmission attempts per link. Such adaptation can provide high reliability and resource efficiency than static scheduling. However, these studies do not consider the adaptability of the parameter of the transmission period of each flow, which can enhance the system resiliency. In contrast, the approaches in [9] and [10] can adaptively change only the transmission period parameter in dynamic environments. Thus, previous studies only consider the adaptation of one of two parameters. In addition, they assume the periodic generation and broadcasting of a new global schedule to change the parameters. Such approaches have limited real-time adaptability because the parameters cannot be changed immediately at the requested instant.

To overcome the clear drawbacks of a central network, parameter adaptation in a distributed network has been considered recently in [11] and [12]. Because local nodes determine their own parameters, they have the clear advantage of real-time parameter change in comparison with a central network. However, with these methods, it is difficult to obtain global optimal parameters because of nonreal-time monitoring for all systems and network status. To resolve these limitations, we propose a novel MAC protocol for a broadcasting-eliminated central network.

B. Guarantee of the End-to-End Transmission Time

To analyze the proposed protocols, several approaches have been proposed. A discrete-time Markov chain (DTMC) model was adopted in [13] and [14]. A DTMC can compute transmission delay distribution based on the mean, variance, and worst-case of the delay for flows. However, this analysis only targets single-hop networks and not TDMA-based multihop networks.

In [15], the end-to-end delay of flows in TDMA-based multihop networks was analyzed. This approach can calculate the end-to-end delay, reflecting the preemption delay caused by the conflict between adjacent nodes. However, it only considers the worst-case end-to-end delay, not the probabilistic end-to-end delay distribution. Considering the limitations of these analyses, we calculate the end-to-end delay distribution on TDMA-based multihop networks by taking into account the packet loss probability.

C. Comparison to Multihop Distributed TDMA Scheduling

Recently, there have been substantial research efforts on developing multihop distributed TDMA scheduling to resolve disadvantages of the centralized multihop network such as adaptability. GALLOP [16] proposes bidirectional sequential scheduling in a distributed network for a closed-loop control system. The article argues retransmission is an important factor in the reliability of the control system and proposes some retransmission approaches to improve it. Similarly, Holistic control design [11] also proposes a self-triggered control approach that transmits packets at demand time aperiodically and a flow rate adaptation algorithm to improve closed-loop control performance and network energy efficiency. It uses a static global schedule allocating a fixed transmission window to each node, and then transmits packets reliably based on glossy flooding [19] protocol, which allows nodes to flood their packets irrelevant to transmission conflicts.

Local voting [17] argues that minimizing nodal delay is to minimize an average end-to-end delay in a distributed multihop network. To do this, they balance the traffic load between neighbor nodes through local voting algorithm. Also, a dynamic distributed multichannel TDMA (DDMC-TDMA) [18] protocol proposes a spatial spectrum reuse approach that solves the exposed/hidden node problems, and it can operate independently to

TABLE I
COMPARISON OF DIFFERENT SCHEDULING PROTOCOLS

	Network	Reliability	e2e real-time Tx	e2e sequential Tx	Adaptability	Evaluation
[14] GALLOP	Distributed	Yes (flexible retransmission method)	Yes	Yes	Yes, (signaling overhead)	Analytical Simulation Experimental
[9] Holistic control	Distributed	Yes (flooding approach)	Yes	Yes	Yes, (limited period)	Simulation Experimental
[15] Local Voting	Distributed	No	No	No	Yes, (convergence time)	Analytical Simulation
[16] DDMC-TDMA	Distributed	No	No	No	Yes, (convergence time)	Simulation
[10] DistributedHART	Distributed	Yes (fixed retransmission method)	Yes	No	Yes	Analytical Simulation Experimental
AdaptiveHART	Centralized	Yes (flexible retransmission method)	Yes	Yes	Yes	Analytical Simulation Experimental

network topology. DistributedHART (DH) [12] builds a schedule frame, called epoch, used periodically through a distributed vertex coloring algorithm. Each node can transmit its own packet at the fixed time window. It guarantees reliable transmissions by adopting multirouting paths and a retransmission allocating approach.

In Table I, we compare qualitatively AH and the distributed multihop TDMA scheduling studies in terms of several criteria (reliability, end-to-end real-time and sequential transmission and adaptability, and evaluation). GALLOP can provide reliable transmissions, but it requires a signaling frame overhead to build the retransmission schedule. Additionally, it requires the frame overhead on demand of a traffic pattern change, which results in the convergence time for adaptability. Holistic control can also provide reliable transmission through glossy flooding, but this flooding-based transmission allows only a single flow to take a whole network during a certain time duration, independent of the network topology. Also, in adaptability, the changeable transmission period is limited to an integer multiple. In local voting, DDMC-TDMA, and DH, it is difficult for a node to transmit sequentially according to a routing path because the node tries to schedule slots by considering only neighbor node conflicts. This increases the end-to-end transmission delay and may adversely affect control performance. Local voting and DDMC-TDMA cannot estimate a packet arrival probability within a given deadline because there is no end-to-end delay analysis. It is difficult to say that this guarantees real-time performance. Also, it requires the convergence time for adaptability due to exchanging the local knowledge to neighbor nodes.

On the other hand, AH uses a flexible retransmission approach that retransmits only in the case of actual packet loss (resource-efficient reliability), and transmits packets sequentially according to routing paths (low end-to-end delay). It is adaptable to the changed traffic pattern without the overhead (adaptability). Moreover, we can compute a probability satisfying a given deadline by analyzing end-to-end transmission delay (real-time performance).

III. SYSTEM MODEL

A. Target System

A wireless networked control system (WNCS) consists of a holistic controller (HC), physical plants, and a WSAN, as shown in Fig. 1. The WSAN comprises field devices, such

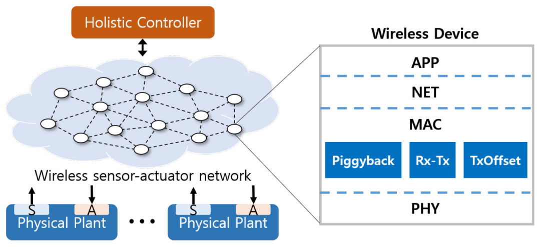


Fig. 1. WNCS architecture.

as sensors and actuators, which are equipped with half-duplex radio transceivers. Sensors are directly attached to plants and deliver their measurements periodically to the corresponding central controller over the multihop network. Control commands generated from the controller are also delivered to the corresponding actuator over the multihop network. Once the actuator receives a command, its plant is controlled by actuator actions based on the command. As a central server, the HC consists of a network manager and controllers for all existing plants [7]. It simultaneously controls all the plants while adjusting the network parameters by measuring the states of both the network and plants. In our system, we assume routing paths are defined statically beforehand.

It is noted that the proposed approach is a node-driven distributed scheduling, but our target system is not a distributed wireless network. Typically, servers in the centralized network can perform expensive computations such as learning-based control with a high-performance processor. However, in IIoT, it is difficult for all servers in the distributed network to have that processor. In this article, since we assume that the central server collects all sensor data and generates control commands by the high-performance processor, our target system is the centralized wireless network.

B. Communication Model

We model a network with a directed graph $G = (V, E)$, where V denotes a set of nodes and E denotes a set of links connected with nodes in V. A link $e \in E$ is a link between its source e_s and destination e_d, $e = (e_s, e_d)$. We support a source routing policy to build routing paths, which has one routing path per flow from its source to destination. The source routing path ϕ_i for flow i consists of n links, $\phi_i = \{e_i^1, \ldots, e_i^n\}$. We adopt a TDMA

MAC protocol on top of the IEEE 802.15.4[1] physical layer. All network nodes are time synchronized. The time slot size is 10 ms, and each slot can accommodate one data transmission and its acknowledgment [5].

In our system, a flow (task) τ_i is characterized by $[T_i, C_i, D_i]$, where T_i represents a period, C_i represents probabilistic execution time (pET), and D_i represents a relative deadline equal to T_i. We design pET considering packet loss probability per link, which will be detailed in Section V-A. We focus on dual-priority systems $\{\tau_L, \tau_H\}$. The network parameters of a flow τ_i are the transmission period T_i and the maximum transmission attempts ε_i. Here, T_i is the period during which the source node generates a packet, and ε_i is the maximum transmission attempts on link $e_i^k \in \phi_i$, where we assume that links $\forall e_i^k \in \phi_i$ have the same value of ε_i.

IV. Proposed AdaptiveHART Protocol

In this section, we provide the details of AH. Our goal is to design a TDMA MAC protocol based on a node-driven scheduling approach. In order to convert from a manager-driven approach to node-driven scheduling, a key issue is how to enable conflict-free transmissions at the node level without using a global schedule. To design a conflict-free node-driven protocol, we propose *Rx-Tx* and *TxOffset* mechanisms. *Rx-Tx* allows each node to select its transmission slots based on radio events. *TxOffset* prevents transmission conflicts by utilizing the hardware characteristics of *capture effects*, which will be described in Section IV-C, even if adjacent transmission instants are unknown. Additionally, to support system adaptability, a *piggyback* mechanism is utilized for parameter delivery to particular nodes on a routing path, rather than entire network nodes, during the packet delivery process.

A. Piggyback Mechanism

Piggyback is a mechanism that stores network parameters in a packet to be sent. This approach is to deliver the changed parameter only to nodes on the flow routing path, in contrast to the typical centralized approach that broadcasts the information to all network nodes. The flow source node piggybacks the parameters, i.e., flow ID τ_i, transmission period T_i, deadline D_i, and the number of retransmission per link ε_i, with an actuation packet when a flow transmits a packet. The payload of the actuation packet is $[\tau_i, T_i, D_i, \varepsilon_i, u_i]$, where D_i means absolute deadline and u_i is a control command for a plant corresponding to the flow. When nodes on ϕ_i receive the packet, they operate according to the piggybacked parameters. For example, let us consider a case in which ε_i is changed from 1 to 3 at time t. Before time t, transmission failure of one of the relay nodes on ϕ_i leads to end-to-end delivery failure. After time t, two consecutive transmission failures on each node are acceptable because of ε_i with 3. Parameter T_i can be adaptive because of instant determination of the source in the *Rx–Tx* mechanism, which will be discussed in Section IV-B.

The piggybacked information is sufficient to be stored in IEEE 802.15.4 MAC payload (102 B), and a time slot of 10 ms is defined to be sufficient to transmit a maximum packet size. Thus, the *piggyback* mechanism has no effect on slot latency, and we validate it by performing experiments and simulations with the maximum packet size in Section VI.

[1]IEEE 802.15.4 supports 250 kb/s data rate and 16 nonoverlapping channels in a 2.4 GHZ ISM band [20]. Standards designed for industry (e.g., WH and ISA100.11a [21]) are based on IEEE 802.15.4 compliant radio.

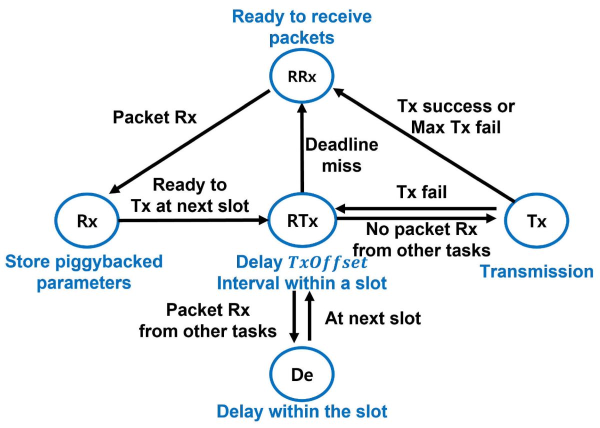

Ready to receive packets
RRx
Packet Rx
Tx success or Max Tx fail
Deadline miss
Ready to Tx at next slot
Rx
RTx
Tx fail
No packet Rx from other tasks
Tx
Store piggybacked parameters
Delay $TxOffset$ Interval within a slot
Transmission
Packet Rx from other tasks
At next slot
De
Delay within the slot

Fig. 2. State transition of the Rx–Tx mechanism.

B. Rx–Tx Mechanism

Rx–Tx allows each node to determine its own transmission time slots based on *radio events*. Basically, transmission of a relay node on link $e_i^k \in \phi_i$ is triggered by completion of packet reception, which is transmitted on $e_i^{k-1} \in \phi_i$.

Specifically, the *Rx–Tx* mechanism is based on the state transition shown in Fig. 2, which can be described as follows.

1) A node u in an idle time is in the ready-to-receive (RRx) state, where u is made ready to receive a packet by turning on its radio at a predetermined channel.

2) If u receives a packet at time slot t, its state changes to receive (Rx); it stores some of the piggybacked parameters $(\tau_i, D_i, \varepsilon_i)$ and places the packet into a queue to be forwarded. The node checks the deadline miss comparing a current time and D_i. If the deadline is missed, it drops the packet and returns to the RRx state.

3) At the next time slot $t + 1$, its state changes to ready-to-transmit (RTx) during an interval of *TxOffset*, which is the time duration from the slot start instant to transmission instant of a packet within a time slot. Note that the value of *TxOffset* should be less than the size of a time slot (10 ms). During the interval, it checks whether packets have been received from other flows. The specific value of *TxOffset* is determined by the *TxOffset* mechanism, as described in Section IV-C.

4) If no packet is received in the RTx state, u enters the transmission (Tx) state, and it transmits the stored packet to its destination. The destination node for τ_i is predetermined during the network initialization phase. If u has multiple types of packets at a certain time instant, it processes the packets in the order of the highest priority. In the Tx state, if transmission fails, it returns to the RTx state and tries to retransmit in the next time slot. Retransmission can be repeated up to $\varepsilon_i - 1$. If transmission is successful in the Tx state, the state changes to RRx to be ready to receive the next packets.

5) In the RTx state, if packet reception from other flows (*flow preemption*) occurs at time slot $t + 1$, the state changes from

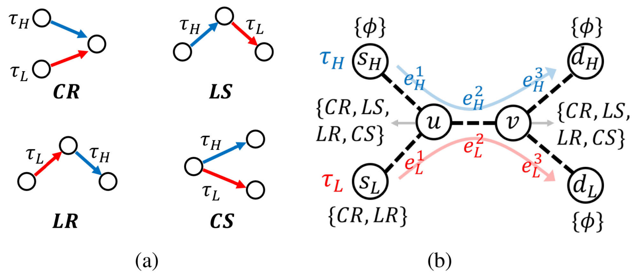


Fig. 3. Definition of transmission conflict types and example of possible conflict types. e_H^i and e_L^i are ith routing link of τ_H and τ_L flows, respectively. (a) Transmission conflict types. (b) Example of network topology and TxConflict types of each node.

RTx to the delay (De) state. In the De state, the transmission of the stored packet is deferred to the next slot because the node is occupied by receiving preemptive packets. At time slot $t+2$, the De state becomes RTx and the node is ready to transmit the stored packet again.

Because relay transmissions are determined by the previous-link transmitter, parameter T_i can be adaptively dependent on the source-select transmission period. In addition, it can save the wasted (unused) retransmission slots on typically fixed retransmission scheduling in a relay process.

C. *TxOffset Mechanism*

TxOffset is a mechanism to avoid possible transmission conflicts due to the absence of global schedules. The key idea of this mechanism is to artificially mismatch the instants of transmission between adjacent nodes utilizing a wireless device hardware characteristic of the capture effect [22]. The capture effect provides low-latency network flooding by eliminating neighborhood contention. It allows a common receiver to receive multiple packets within a time slot if their transmissions have only 3 ms difference. This is experimentally proved in [22]. Therefore, we need to determine how to create the time difference between adjacent transmissions.

TxConflict types: We first define wireless transmission conflict (*TxConflict*) types considering the flow routing path and priority. *TxConflict* is a packet loss and transmission delay generated by the neighbor-simultaneous transmission because a node with half-duplex radio cannot transmit and receive a packet at the same time [23]. We assume that there are two types of flows in the network, high- and low-priority flows (τ_H and τ_L) with overlapped routing paths. There are four types of *TxConflict*, as shown in Fig. 3(a).

1) *Common Receiver (CR) conflict*: Parallel transmissions collide in a common receiver.

2) *Low-flow Receiver (LR) conflict*: The sender of τ_L cannot transmit its packet to the receiver of τ_L because the receiver is also a sender of τ_H.

3) *Low-flow Sender (LS) conflict*: As in (ii), the sender of τ_L cannot receive the packet of τ_H while transmitting the packet to the receiver.

4) *Common Sender (CS) conflict*: A sender cannot transmit multiple flow packets at the same time. In a network topology as Fig. 3(b), each node n can prerecognize its possible transmission conflict set [TxConflict(n)] under fixed routing paths of all flows.

We define the time interval from the start of a slot to the start of transmission as *TxOffset*. Every node adaptively modifies this value for every slot according to the following three events.

1) *Packet priority:* If the priority of packets to be sent is τ_H, then they set the value with Offset$_1$.

2) As a case of $\{\text{LS} \notin \text{TxConflict}(n)\}$ for a node n, when n transmits its own τ_L packet, it does not occur that n receives packets from τ_H, such as the s_L node in Fig. 3(b). In this case, the transmissions of s_L may conflict at its destination u with s_H transmissions on e_H^1 (*CR* conflict type), or u transmissions on e_H^2 (*LR* conflict type). To avoid *CR* and *LR* conflicts, input node n sets the *TxOffset* value with Offset$_2$.

3) As a case of $\{\text{LS} \in \text{TxConflict}(n)\}$, n may receive packets from τ_H at their transmission slot when it transmits τ_L packets, such as u or v nodes on e_L^2 or e_L^3, respectively, in Fig. 3(b). To avoid *LS* conflict, they set the value with Offset$_3$.

Now, we address how to select the specific values of the offsets, i.e., Offset$_{1,2,3}$. We determine the values based on empirical results.

1) Offset$_1$ is 0 ms. In event 1), when node n transmits τ_H packets, transmission conflicts and delay of the packets should not occur. For example, in Fig. 3(b), when transmissions on e_H^1 and e_L^2 simultaneously occur, node u should receive packets of s_H even if it cancels its own transmissions. To achieve this preemption, nodes sending τ_H packets set Offset$_1$ to 0 ms, which is experimentally justified in Fig. 4.

2) Offset$_2$ is 3 ms. In event 2), a destination node (e.g., node u) may simultaneously receive two types of packets, τ_H and τ_L. Based on the capture effect, if the transmission instant of τ_L has a 3 ms difference from that of τ_H, the common receiver can receive the two packets in the same time slot. Because the transmitter of τ_H has transmission with Offset$_1$ (0 ms), the transmitter of τ_L, which satisfies $\{\text{LS} \notin \text{TxConflict}(n)\}$, has transmission with Offset$_2$ (3 ms).

3) Offset$_3$ is 5 ms. In event 3), before the nodes of τ_L transmit their packets, they may receive packets from τ_H. For τ_H packets to preempt τ_L packets, τ_L nodes should wait for a minimum interval (5 ms) during which τ_H packet reception can be recognized. According to the *Rx–Tx* mechanism, if there is no packet reception during that interval, the node of τ_L transmits its packet. Otherwise, it defers τ_L transmissions to complete receiving τ_H packets. We refer to the delayed τ_L transmission due to the preemption of τ_H as the *preemption effect*. The value of 5 ms is derived from experiment results in Fig. 4.

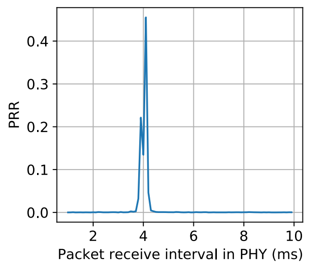


Fig. 4. Rationality for Offset$_3$ selected with 5 ms. Time intervals of a destination node on a link recognizes packet receptions for 1 00 000 packets.

We empirically demonstrate the rationality for Offset$_3$ to be 5 ms. The wireless device used in the experiments is TelosB mote that has a CC2420 radio and a MSP430 microcontroller. We need to verify the packet reception interval, from a slot start time point to a packet reception point, at the physical layer on a destination node. The source and destination nodes are 2 m apart and time synchronized.

Fig. 4 presents a packet reception ratio with respect to the packet reception interval among 1 00 000 packets on a link. From the figure, 85% of the packets arrive at the receiver in a range of [3.9, 4.1] ms, and others are included in [3.7, 3.9] ms and [4.1, 4.3] ms. It shows that the node can recognize the packet reception before 5 ms within a time slot, thus it means that relay nodes can defer sending τ_L packets in the *preemption effect*. We will demonstrate that this effect does not cause a packet collision in a real testbed in Secion VI-B.

V. Performance Analysis

In this section, we propose a SRTA tailored to wireless multihop networks. Using SRTA, we compute end-to-end transmission delay distributions for given network parameters and provide a stochastic real-time schedulability condition. Note that given routing paths and characters of flows, AH is a conflict-free transmission protocol that operates without a specific scheduling algorithm in multihop networks. Thus, our focus in this section is not a formal proof of optimal scheduling, but rather an analysis of the end-to-end delay distribution of flows so that we can estimate the probability of satisfying a given deadline (real-time performance).

A. Probabilistic Transmission Time

To reflect the packet loss probability, we newly model the probabilistic execution time (hereafter probabilistic transmission time). In wireless multihop networks, C_i represents transmission attempts on time slots from source to destination on the routing path ϕ_i. For example, if the number of end-to-end transmissions of τ_i is 4, then $C_i = 4$ because τ_i uses four time slots. We split it by the transmission time C_i^k on each link $e_i^k \in \phi_i$, which represents the number of transmission until the first transmission success on a link. Because the number of transmissions on a link depends on the link quality, $\boldsymbol{C}_i^k$ is a discrete random variable expressed as

$$\boldsymbol{C}_i^k = \begin{pmatrix} C_{\min}^k & \cdots & C_{\max}^k \\ f_{\boldsymbol{C}_i^k}(C_{\min}^k) & \cdots & f_{\boldsymbol{C}_i^k}(C_{\max}^k) \end{pmatrix} \tag{1}$$

where $C_{\min}^k$ is a non-negative integer value. Probability function $f_{\boldsymbol{C}_i^k}(c)$ is the success probability with c attempts to deliver a packet on e_i^k. We calculate $f_{\boldsymbol{C}_i^k}(c)$ as

$$f_{\boldsymbol{C}_i^k}(c) = \begin{cases} (1-q_i^k)^{c-1} \cdot q_i^k & c \in [1, \varepsilon_i^k - 1] \\ 1 - \sum_{n=1}^{\varepsilon_i^k - 1} f_{\boldsymbol{C}_i^k}(n) & c = \varepsilon_i^k \end{cases} \tag{2}$$

where q_i^k is the link quality on e_i^k, which is the success probability of a transmission. Here ε_i^k is the maximum number of transmissions on e_i^k. In this article, we assume that each link $e_i^k \in \phi_i$ has the same ε_i^k ($\varepsilon_i = \varepsilon_i^k \forall e_i^k \in \phi_i$), where ε_i is determined by the required reliability level [6].

In AH, if a transmission success at a time slot, then the next transmission starts immediately at the next slot. Thus, end-to-end transmission time $\boldsymbol{C}_i$ is the sum of link transmission times $\boldsymbol{C}_i^k$, for $\forall e_i^k \in \phi_i$. This means a probability distribution of time duration taken from packet-released time on a source to packet-arrival time on a destination. Because the sum of discrete random variables is computed by convolution $\otimes$, the accumulation of transmission time $\boldsymbol{C}_i^k$ of all links on the routing path ϕ_i is computed as follows

$$\boldsymbol{C}_i = \otimes_{k=1}^{|\phi_i|} \boldsymbol{C}_i^k. \tag{3}$$

B. Stochastic Response Time Analysis (SRTA)

Existing SRTA regards a preemption unit imposed by a high-priority flow τ_{hi} as a job (end-to-end communication delay), which is too pessimistic to apply to multihop networks. To extend it to multihop networks, we newly design the analysis considering the preemption unit as a link communication delay, not end-to-end communication delay.

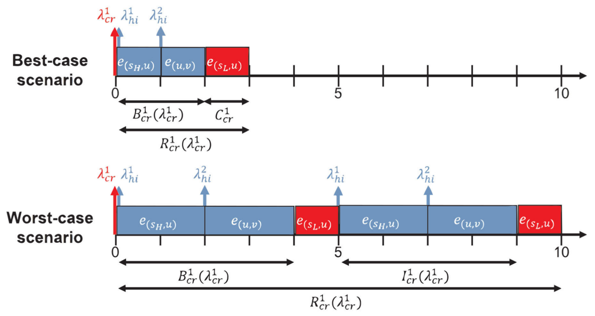
The response time $\boldsymbol{\mathcal{R}}_{\text{cr}}$ (hereafter communication delay) of a flow τ_{cr} indicates the end-to-end packet transmission delay distribution over its routing path ϕ_{cr}. We first define the communication delay distribution on a link $e_{\text{cr}}^k \in \phi_{\text{cr}}$. The link communication delay $\boldsymbol{\mathcal{R}}_{\text{cr}}^k$ denotes the number of time slots from the packet release instant λ_{cr}^k on link e_{cr}^k to the transmission success instant. It is computed as

$$\boldsymbol{\mathcal{R}}_{\text{cr}}^k(\lambda_{\text{cr}}^k) = \boldsymbol{B}_{\text{cr}}^k(\lambda_{\text{cr}}^k) \otimes \boldsymbol{C}_{\text{cr}}^k \otimes \boldsymbol{I}_{\text{cr}}^k(\lambda_{\text{cr}}^k) \tag{4}$$

where k is the hop count on ϕ_{cr}. Here, $\boldsymbol{B}_{\text{cr}}^k(\lambda_{\text{cr}}^k)$ is the *backlog* of high-priority flows that is released before λ_{cr}^k and still not completed yet at λ_{cr}^k, and $\boldsymbol{I}_{\text{cr}}^k(\lambda_{\text{cr}}^k)$ is the transmission time of high-priority flows that are released after λ_{cr}^k. Intuitively, $\boldsymbol{\mathcal{R}}_{\text{cr}}^k(\lambda_{\text{cr}}^k)$ is computed by the sum of the preemption time of high-priority flows that is imposed on e_{cr}^k and its transmission time $\boldsymbol{C}_{\text{cr}}^k$. The equation refers to (3) in [24].

For example, there are two flows $(\tau_{\text{hi}}, \tau_{\text{cr}})$ in a network topology, such as Fig. 3(b). We assume flow characters as follows: τ_{hi} is a higher prioriy flow than τ_{cr}. Routing paths of them are $\phi_{hi} = [e_{(s_H,u)}, e_{(u,v)}]$ and $\phi_{\text{cr}} = [e_{(s_L,u)}]$, respectively. Transmission periods are $T_{\text{hi}} = 5$ and $T_{\text{cr}} = 10$. Packet release times on the first link are the same at slot 0, $\lambda_{\text{hi}}^1 = \lambda_{\text{cr}}^1 = 0$. The link quality of each link is $q_i^k = 0.9 \forall q_i^k \in [\phi_{\text{hi}}, \phi_{\text{cr}}]$, and the maximum transmission opportunity on each link is $\varepsilon_i^k = 2 \forall \varepsilon_i^k \in [\phi_{\text{hi}}, \phi_{\text{cr}}]$. To help to understand (4), we assume node s_H and s_L cannot transmit it to a common destination simultaneously.

Fig. 5 shows the best-case and worst-case communication delay of τ_{cr} on link $e_{(s_L,u)}$. At slot 0, a backlog $\boldsymbol{B}_{\text{cr}}^1(\lambda_{\text{cr}}^1 = 0) = \begin{pmatrix} 2 & 3 & 4 \\ 0.81 & 0.18 & 0.01 \end{pmatrix}$ exists, which is a transmission time distribution on $e_{(s_H,u)}$, because the first link of the flows are released at the same time and flow τ_{hi} has a higher priority, where the detail computation of $\boldsymbol{B}_{\text{cr}}^1$ will be discussed in Section V-B1. In the best scenario, the backlog is $\boldsymbol{B}_{\text{cr}}^1 = (\begin{smallmatrix} 2 \\ 0.81 \end{smallmatrix})$, the transmission time is $\boldsymbol{C}_{\text{cr}}^1 = \begin{pmatrix} 1 \\ 0.9 \end{pmatrix}$, and interference $\boldsymbol{I}_{\text{cr}}^1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ does not exist due to the early completion of the transmission time, thus the best



λ_{cr}^1 λ_{hi}^1 λ_{hi}^2
Best-case scenario
$e_{(s_H,u)}$ $e_{(u,v)}$ $e_{(s_L,u)}$
0
5
10
$B_{cr}^1(\lambda_{cr}^1)$
C_{cr}^1
$R_{cr}^1(\lambda_{cr}^1)$
λ_{cr}^1 λ_{hi}^1 λ_{hi}^2 λ_{hi}^1 λ_{hi}^2
Worst-case scenario
$e_{(s_H,u)}$ $e_{(u,v)}$ $e_{(s_L,u)}$ $e_{(s_H,u)}$ $e_{(u,v)}$ $e_{(s_L,u)}$
0
5
10
$B_{cr}^1(\lambda_{cr}^1)$
$I_{cr}^1(\lambda_{cr}^1)$
$R_{cr}^1(\lambda_{cr}^1)$

Fig. 5. Best- and worst-case communication delay of a flow τ_{cr}, where higher-priority flow τ_{hi} exists.

communication delay is $\boldsymbol{\mathcal{R}}_{\text{cr}}^1 = \begin{pmatrix} 3 \\ 0.729 \end{pmatrix}$ by (4). In the worst scenario, when the transmission time is not completed until slot 5 by $\boldsymbol{B}_{\text{cr}}^1 = \begin{pmatrix} 4 \\ 0.01 \end{pmatrix}$ and $\boldsymbol{C}_{\text{cr}}^1 = \begin{pmatrix} 2 \\ 0.1 \end{pmatrix}$, a newly generated flow τ_{hi} is regarded as interference $\boldsymbol{I}_{\text{cr}}^1(\lambda_{\text{cr}}^1 = 0) = \begin{pmatrix} 2 & 3 & 4 \\ 0.81 & 0.18 & 0.01 \end{pmatrix}$, which is accumulated in $\boldsymbol{\mathcal{R}}_{\text{cr}}^1 = \begin{pmatrix} 100 \\ 0.00001 \end{pmatrix}$, where the interference considers only a worst-case $\boldsymbol{I}_{\text{cr}}^1 = \begin{pmatrix} 4 \\ 0.01 \end{pmatrix}$. Consequently, (4) computes the link communication delay distribution on $e_{(s_L,u)}$ as $\boldsymbol{\mathcal{R}}_{\text{cr}}^1(\lambda_{\text{cr}}^1 = 0) = \begin{pmatrix} 3 & \dots & 10 \\ 0.729 & \dots & 0.00001 \end{pmatrix}$.

We modify (4) considering the fact that preemption from τ_{hi} can occur at different instants with different probabilities. The release instant $\boldsymbol{\lambda}_{\text{hi}}^i$ of a link e_{hi}^i is a random variable because the packet release time on e_{hi}^i is determined by transmission time $\boldsymbol{C}_{\text{hi}}^{i-1}$ of the previous link e_{hi}^{i-1}. Thus, we compute the communication delay $\boldsymbol{\mathcal{R}}_{\text{cr}}^k$ given λ_{cr}^k and $\boldsymbol{\lambda}_{\text{hi}}^i$ as

$$\boldsymbol{\mathcal{R}}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) = \boldsymbol{B}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) \otimes \boldsymbol{C}_{\text{cr}}^k \otimes \boldsymbol{I}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) \tag{5}$$

where the bold text $\boldsymbol{\lambda}_{\text{hi}}^i$ means a random variable and the normal text λ_{cr}^k means a single value.

1) Backlog: To compute $\boldsymbol{\mathcal{R}}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i)$, we first introduce the computation of $\boldsymbol{B}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i)$. For the preemption on e_{cr}^k to occur at λ_{cr}^k, it must satisfy the following two *preemption conditions*.

First, there should be flows with higher-priority τ_{hi} than that of flow τ_{cr}. Second, there should be preemptable candidate links to e_{cr}^k ($Cand(e_{\text{cr}}^k)$). $Cand(e_{\text{cr}}^k)$ denotes link set of τ_{hi} directly connected to e_{cr}^k, e.g., $e_H^{1,2,} \in Cand(e_L^1)$, $e_H^{1,2,3} \in Cand(e_L^2)$ and $e_H^{2,3} \in Cand(e_L^3)$ in Fig. 3(b). Thus, simultaneous transmission of e_{cr}^k and $e_{\text{hi}}^i \in Cand(e_{\text{cr}}^k)$ incurs transmission delay of e_{cr}^k due to the *TxOffset* mechanism.

Given λ_{cr}^k and $\boldsymbol{\lambda}_{\text{hi}}^i$, $\boldsymbol{B}_{\text{cr}}^k$ depends on the accumulated transmission times of candidate links, where link e_{hi}^i satisfies the *preemption conditions*. For example, in Fig. 3(b), when two links $e_{\text{cr}}^2 (= e_{(u,v)})$ and $e_{\text{hi}}^1 (= e_{(s_H,u)})$ are released simultaneously, $\lambda_{\text{cr}}^2 = \lambda_{\text{hi}}^1$, the transmission time on e_{cr}^2 is preempted by that of $\{e_{\text{hi}}^1, e_{\text{hi}}^2 (= e_{(u,v)}), e_{\text{hi}}^3 (= e_{(v,d_H)})\} \in Cand(e_{\text{cr}}^2)$ until the entire transmissions on $Cand(e_{\text{cr}}^2)$ are finished due to *TxOffset* mechanism. However, if λ_{hi}^1 is before λ_{cr}^2 and its transmission is finished before λ_{cr}^2, then the transmission time on e_{hi}^1 is not included in $\boldsymbol{B}_{\text{cr}}^2$. Because the backlog means the residual transmission time after λ_{cr}^2, we need to compute $\boldsymbol{B}_{\text{cr}}^k$ considering both of the amount of preemption on e_{cr}^k from candidate links and its release distributions.

We first propose an equation for the amount of preemption of the candidate links.

Proposition 1: If links e_{cr}^k and $e_{\text{hi}}^i \in Cand(e_{\text{cr}}^k)$ are activated simultaneously at time t, then the preemption imposed to e_{cr}^k is computed as

$$\boldsymbol{C}_{pree,hi}^i = \otimes_{n=i}^j \boldsymbol{C}_{hi}^n \tag{6}$$

where j is the maximum link index in $Cand(e_{\text{cr}}^k)$, where link indexes in the candidate set are sorted in ascending order.

Proof: By the definition of $Cand(e_{\text{cr}}^k)$, links in the set, $\{e_{\text{hi}}^z, \dots, e_{\text{hi}}^j\} \in Cand(e_{\text{cr}}^k)$, can preempt the link e_{cr}^k, where $z \le i \le j$. The candidate links are sorted in ascending order, z (j) is a minimum (maximum) link index in $Cand(e_{\text{cr}}^k)$, respectively. If e_{hi}^i is activated simultaneously with e_{cr}^k, the e_{cr}^k execution is delayed until the completion of e_{hi}^i due to the *TxOffset* mechanism. After the e_{hi}^i completion at a certain time slot, the next link transmissions (higher link index than i) are sequentially executed from the next time slot due to the *Rx-Tx* mechanism. Thus, because e_{cr}^k cannot be executed until the completion of candidate links $\{e_{\text{hi}}^i, \dots, e_{\text{hi}}^j\} \in Cand(e_{\text{cr}}^k)$, the preemption on e_{cr}^k is the sum of the transmission times of the candidate links.■

We introduce a *backlog exist* equation. Given λ_{cr}^k and release distribution $\boldsymbol{\lambda}_{\text{hi}}^i$ of $e_{\text{hi}}^i \in Cand(e_{\text{cr}}^k)$, the presence of a backlog on e_{cr}^k at λ_{cr}^k is computed as

$$\text{BE}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) = \left(\boldsymbol{\lambda}_{\text{hi}}^{i,\le\lambda_{\text{cr}}^k} \otimes \boldsymbol{C}_{\text{pree,hi}}^i\right)^{>\lambda_{\text{cr}}^k}. \tag{7}$$

By the definition of a *backlog*, if $e_{\text{hi}}^i \in Cand(e_{\text{cr}}^k)$ releases before λ_{cr}^k and still executes after λ_{cr}^k, then a *backlog* exists. Here, $\boldsymbol{\lambda}_{\text{hi}}^{i,\le\lambda_{\text{cr}}^k}$ represents values and corresponding probabilities before λ_{cr}^k in a $\boldsymbol{\lambda}_{\text{hi}}^i$ distribution, and $(\boldsymbol{\lambda}_{\text{hi}}^{i,\le\lambda_{\text{cr}}^k} \otimes \boldsymbol{C}_{\text{pree,hi}}^i)$ represents the distribution of finish instants of preemption links by Proposition 1. The notation $(\cdot)^{>\lambda_{\text{cr}}^k}$[2], $(\boldsymbol{\lambda}_{\text{hi}}^{i,\le\lambda_{\text{cr}}^k} \otimes \boldsymbol{C}_{\text{pree,hi}}^i)^{>\lambda_{\text{cr}}^k}$ represents some part of the distribution that still executes after λ_{cr}^k.

Equation (7) expresses the absolute finish instants of preemption links. To express the relative amount of *backlog*, we use the shrink() operation.

$$\boldsymbol{B}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) = \text{shrink}\left(\text{BE}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i), \lambda_{\text{cr}}^k\right) \tag{8}$$

where shrink() shifts BE_{cr}^k to the left λ_{cr}^k units and accumulates the residual probability (1-prob(BE_{cr}^k)) in the origin [26].

2) Interference: Interference $\boldsymbol{I}_{\text{cr}}^k$ is the amount of preemption on e_{cr}^k released after λ_{cr}^k. If there is no interference on e_{cr}^k at λ_{cr}^k, then a delay distribution is derived by accumulating the *backlog* and transmission time, $\boldsymbol{\mathcal{R}}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) = \boldsymbol{B}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i) \otimes \boldsymbol{C}_{\text{cr}}^k$. Otherwise, we accumulate $\boldsymbol{I}_{\text{cr}}^k$ with $\boldsymbol{\mathcal{R}}_{\text{cr}}^k(\lambda_{\text{cr}}^k, \boldsymbol{\lambda}_{\text{hi}}^i)$.

We define an *interference exist* condition. Interference imposed on e_{cr}^k depends on a flow period T_{hi} of τ_{hi}. The condition is that a new job of τ_{hi} is released before the finish of transmission

[2] A partial random variable of $\boldsymbol{X}$ specifies the possible values with respect to a scalar value u. $\boldsymbol{X}^{\le u}$ denotes values less than or equal to u and its corresponding probabilities among the possible values of $\boldsymbol{X}$ [25].

time $\boldsymbol{C}_{\text{cr}}^{k}$. Interference starts from the first link of its candidate links $e_{\text{hi}}^{i=\min} \in Cand(e_{\text{cr}}^{k})$, so we need to identify that release instants $\boldsymbol{\lambda}_{\text{hi}}^{i}$ are earlier than the current communication delay. The release distribution $\boldsymbol{\lambda}_{\text{hi}}^{i}$ is calculated as $\boldsymbol{\lambda}_{\text{hi}}^{i} = \lambda_{\text{hi}}^{1} \otimes_{n=1}^{i-1} \boldsymbol{C}_{\text{hi}}^{n}$. Thus, if $\max(\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}) > \min(\boldsymbol{\lambda}_{\text{hi}}^{i})$ is satisfied, then $\boldsymbol{I}_{\text{cr}}^{k}$ should be accumulated in the current $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}(\lambda_{\text{cr}}^{k}, \boldsymbol{\lambda}_{\text{hi}}^{i})$ [24].

The current delay distribution $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}(\lambda_{\text{cr}}^{k}, \boldsymbol{\lambda}_{\text{hi}}^{i})$ is recomputed including $\boldsymbol{I}_{\text{cr}}^{k}$ as

$$\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}(\lambda_{\text{cr}}^{k}, \boldsymbol{\lambda}_{\text{hi}}^{i}) = \boldsymbol{\mathcal{R}}_{\text{cr}}^{k,\text{head}} \oplus (\boldsymbol{\mathcal{R}}_{\text{cr}}^{k,\text{tail}} \otimes \boldsymbol{I}_{\text{cr}}^{k}) \tag{9}$$

where $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k,\text{head}}$ is the part of the delay distribution that is not affected by the current interference, whereas $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k,\text{tail}}$ is the part of the distribution that is affected by the current interference. Only the distribution of $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k,\text{tail}}$ includes interference $\boldsymbol{I}_{\text{cr}}^{k}$, and it is then combined with $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k,\text{head}}$ using operator $\oplus$[3]. The value of $\boldsymbol{I}_{\text{cr}}^{k}$ is the sum of the transmission times of all candidate links by Proposition 1, $\boldsymbol{I}_{\text{cr}}^{k} = \otimes_{n=i}^{j} \boldsymbol{C}_{\text{hi}}^{n}$, $\{e_{\text{hi}}^{i=\min}, \ldots, e_{\text{hi}}^{j=\max} \in Cand(e_{\text{cr}}^{k})\}$. The specific computation algorithm for $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$ refers to the doPreemption() function in [24], where notations $\mathcal{C}$ and $\mathcal{A}$ are $\boldsymbol{I}_{\text{cr}}^{k}$ and $\boldsymbol{\lambda}_{\text{hi}}^{i}$, respectively.

3) Putting Together: We now explain the overall process to compute communication delay distributions $\boldsymbol{\mathcal{R}}_{\text{cr}}$ of a flow τ_{cr}. We compute $\boldsymbol{\mathcal{R}}_{\text{cr}}$ recursively from the first link $\boldsymbol{\mathcal{R}}_{\text{cr}}^{1}$ to the last one $\boldsymbol{\mathcal{R}}_{\text{cr}}^{|\phi_{\text{cr}}|}$ using the SRTA algorithm (Algorithm 1). Given λ_{cr}^{k} and $\boldsymbol{\lambda}_{\text{hi}}^{i}$ as inputs, the SRTA() algorithm of the current link e_{cr}^{k} computes the communication delay distribution $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$ after λ_{cr}^{k} and triggers the computation for $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k+1}$ of next link e_{cr}^{k+1}. Due to the recursive computation, the SRTA() of the current link e_{cr}^{k} actually returns the accumulated delay distributions from the kth link to the last link.

Specifically, in lines 5–11, it computes the current communication delay distribution $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}(\lambda_{\text{cr}}^{k}, \boldsymbol{\lambda}_{\text{hi}}^{i})$ on link e_{cr}^{k}. Line 5 identifies $\boldsymbol{\lambda}_{\text{hi}}^{i}$ preemtable on e_{cr}^{k}, $\boldsymbol{\lambda}_{\text{hi}}^{i} = \lambda_{\text{hi}}^{1} \otimes_{n=1}^{m-1} \boldsymbol{C}_{\text{hi}}^{n}$ if $i < m$, otherwise, it maintains the input $\boldsymbol{\lambda}_{\text{hi}}^{i}$. $\boldsymbol{C}_{\text{cr}}^{k}$, $\boldsymbol{B}_{\text{cr}}^{k}$, and $\boldsymbol{I}_{\text{cr}}^{k}$ are computed by using (1), (8), and $\boldsymbol{I}_{\text{cr}}^{k} = \otimes_{n=m}^{j} \boldsymbol{C}_{\text{hi}}^{n}$, respectively. In lines 9–11, it computes $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$ that includes $\boldsymbol{I}_{\text{cr}}^{k}$ using the doPreemption() function in [24] if the *interference exist* condition is satisfied. Here, $\boldsymbol{G}_{\text{hi}}^{m}$ in line 8 is the release distribution of e_{hi}^{m} of the newly generated job.

Computation of the next link $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k+1}$ is triggered in lines 12–18. The input $\boldsymbol{\lambda}_{\text{hi}}^{i'}$ on the next SRTA() is $i' = j+1$ due to Proposition 1 if preemption occurs in computing current $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$, otherwise, $i' = i$ because of the possibility for e_{hi}^{i} to preempt on e_{cr}^{k+1}. Another input $\lambda_{\text{cr}}^{k+1}$ is an instant elapsed from the release instant λ_{cr}^{k} to a communication delay $\mathcal{R} \in \boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$, $\lambda_{\text{cr}}^{k+1} = \lambda_{\text{cr}}^{k} \otimes \mathcal{R}$. In lines 16–17, all possible $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k+1}$ are accumulated on $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$.

4) Schedulability Condition: The probability that a flow τ_i misses its deadline, denoted by DMP_i, can be derived from the probability function $f_{\boldsymbol{X}}$ as follows

$$\text{DMP}_i = P(\boldsymbol{\mathcal{R}}_i > D_i) = \sum_{r > D_i} f_{\boldsymbol{\mathcal{R}}_i}(r) \tag{10}$$

where $\boldsymbol{\mathcal{R}}_i$ is a random variable, as computed using Algorithm 1. If the deadline miss probability for a flow is less than or equal to the probability threshold of 1%, and the condition is satisfied for all flows, then the system is considered schedulable.

[3]Coalescion operator ($\oplus$) represents the combination of the two random variables into a single random variable [24].

Algorithm 1: Stochastic response time analysis.

```
 1: Input: $\lambda_{\text{cr}}^{k}, \boldsymbol{\lambda}_{\text{hi}}^{i}$
 2: Output: $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$
 3: Result = []
 4: $m, j$ = min and max link index of $Cand(e_{\text{cr}}^{k})$
 5: Identify $\boldsymbol{\lambda}_{\text{hi}}^{i}$ preemptable on $e_{\text{cr}}^{k}$
 6: Compute $\boldsymbol{C}_{\text{cr}}^{k}$, $\boldsymbol{B}_{\text{cr}}^{k}(\lambda_{\text{cr}}^{k}, \boldsymbol{\lambda}_{\text{hi}}^{i})$, and $\boldsymbol{I}_{\text{cr}}^{k}$
 7: $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k} = \boldsymbol{B}_{\text{cr}}^{k} \otimes \boldsymbol{C}_{\text{cr}}^{k}$
 8:   $\boldsymbol{G}_{\text{hi}}^{m} = T_{\text{hi}} \otimes_{n=1}^{m-1} \boldsymbol{C}_{\text{hi}}^{n}$
 9: if $\max(\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}) > \min(\boldsymbol{G}_{\text{hi}}^{m})$ then
10:     $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$ = doPreemption($\boldsymbol{\mathcal{R}}_{\text{cr}}^{k}, \boldsymbol{G}_{\text{hi}}^{m}, \boldsymbol{I}_{\text{cr}}^{k}$); //in [24]
11: end if
12: for $\mathcal{R} \in \boldsymbol{\mathcal{R}}_{\text{cr}}^{k}$ do
13:     Select next candidate link $i'$
14:     Compute $\boldsymbol{\lambda}_{\text{hi}}^{i'}$ and $\lambda_{\text{cr}}^{k+1}$
15:     $\boldsymbol{\mathcal{R}}_{\text{cr}}^{k+1}$ = SRTA($\lambda_{\text{cr}}^{k+1}, \boldsymbol{\lambda}_{\text{hi}}^{i'}$)
16:     $\mathcal{R}_{\text{intermediary}} = \mathcal{R} \otimes \boldsymbol{\mathcal{R}}_{\text{cr}}^{k+1}$;
17:     Result = Result $\oplus$ $\mathcal{R}_{\text{intermediary}}$
18: end for
```

VI. Performance Evaluation

Based on the proposed three mechanisms (*Piggyback*, *Rx-Tx*, and *TxOffset*), AH delivers packets with minimal end-to-end delay, and provides resource-efficient reliable transmissions by flexible retransmission approach, which only works when an actual packet loss occurs. Although it is a centralized multihop network, it has the adaptability of changing flow characters immediately without global schedule dissemination due to piggybacked information-centric operation. Moreover, using SRTA, we compute end-to-end delay distribution in applying our protocol, which can guarantee real-time performance stochastically.

A. Accuracy of the Stochastic Response Time Analysis

We evaluate the theoretical analysis accuracy by comparing the end-to-end communication delay distribution results of SRTA with those of our protocol implemented on TOSSIM [27], which is a state-of-the-art wireless sensor network simulator. Comparing end-to-end delay distribution results of theoretical SRTA and that of AH implemented in the simulator is intended to verify the accuracy of the SRTA. For validation in dynamic environments, we perform the evaluations both in normal and harsh channel conditions.

Evaluation setting: There are two types of flows (τ_H and τ_L) in the networks, and the evaluations are performed by varying the end-to-end distance of each flow from 2 to 5 hops. A 3-hop network topology is the same as that in Fig. 3(b). A 2-hop topology has no links between node u and v ($u = v$). Likewise, a 5-hop topology has 3 hops between them. We employ packet delivery rates (PDRs) per link of 90% and 52% under the normal and harsh conditions, respectively. To provide a sufficient end-to-end PDR of 99% of each flow in both conditions, the number of transmission attempts per link (#Tx) is 3 under normal

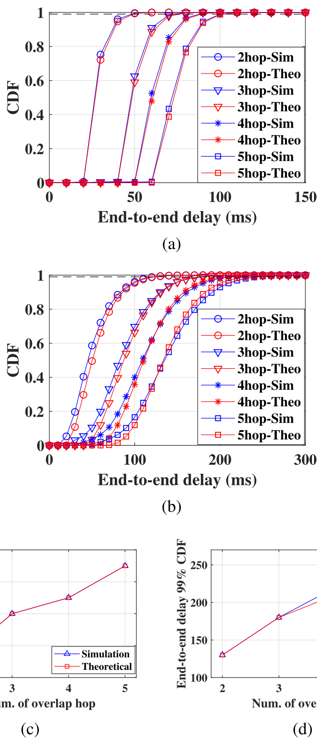


Fig. 6. Comparison between the end-to-end delay distributions of the simulator and SRTA under normal and harsh channel conditions. (a) Delay distribution (normal). (b) Delay distribution (harsh). (c) 99% CDF of the distribution (normal). (d) 99% CDF of the distribution (harsh).

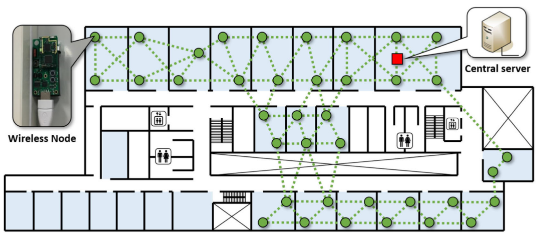


Fig. 7. WNCS testbed.

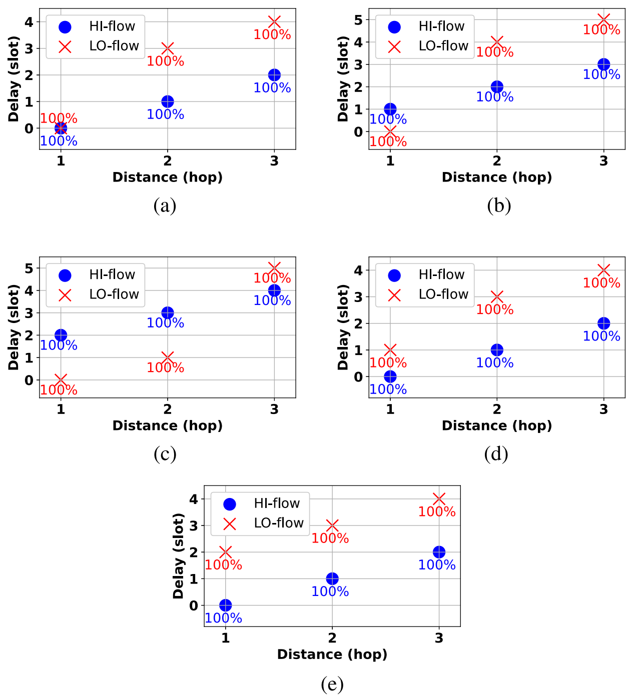


Fig. 8. Packet transmission delays of HI- and LO-flows depending on the difference in the transmission start time between them. (a) Two flows transmit at the same slot. (b) HI-flow transmits one slot later. (c) HI-flow transmits two slots later. (d) LO-flow transmits one slot later. (e) LO-flow transmits two slot later.

and 9 under harsh conditions. These values are determined according to [6]. We employ flow periods of 150 ms for normal conditions and 300 ms for harsh conditions; these values satisfy the schedulability condition (10) sufficiently.

Fig. 6(a) and (b) represent the cumulative density function (CDF) of end-to-end delay distributions for τ_L with variation of the end-to-end distance under normal and harsh channel conditions, respectively. The blue line is delay distributions from the simulator, and the red line is results computed from the proposed theoretical SRTA. For a quantitative comparison, we need to verify 99% instant of each CDF as Fig. 6(c) (normal) and 6(d) (harsh), because we determine stochastic real-time performance baseline with 99%. These results show that the proposed SRTA finds the accurate stochastic real-time deadline under the given parameters, except for 4-hop and 5-hop under harsh condition. One reason for these slight errors is that a real network and wireless simulator are unavailable to have 0% packet loss probability even if a sufficient #Tx is allocated. However, in modeling the link transmission time in (1), we assume the loss probability with zero for simplicity of analysis. Nevertheless, since 10 ms represents only one time slot difference, the error can be regarded as insignificant.

B. Performance of AdaptiveHART

In this section, we evaluate the performance of AH MAC in terms of the following aspects. 1) Protocol operation. 2) Network apatability. 3) Real-time performance. 4) Control performance.

We evaluate 1), 2), and 3) by using our WNCS testbed, and 4) by using a wireless cyber-physical simulator (WCPS) [28] that integrates MATLAB-Simulink with TOSSIM simulator. This WCPS supports cosimulation of control systems implemented in Simulink and the AH protocol implemented in TOSSIM.

Fig. 7 shows the motes deployment of our testbed in our campus building. The system comprises a network manager on a central server and multiple TelosB motes [29] with TinyOS 2.1.2. Each mote can become either an access point or a field device that forms a multihop wireless network. The central server floods synchronization packets periodically, and the nodes that receive them step a unit slot of 10 ms. We allocate each node its destination according to the given network topology. The node operates every time slot based on AH protocol.

1) Protocol Operation: We verify whether the proposed 3 mechanisms of AH cause packet loss and delay with real devices. We place 6 motes within a diameter of 2 m to minimize channel loss and only measure the impact of the protocols. The routing paths of two flows (τ_H and τ_L) overlap on two relay nodes and have their own destinations, as shown in Fig. 3(b). To consider various environments of packet delivery, we intentionally mismatch transmission slots of the sources of the two flows, and verify packet delays and loss probabilities per hop.

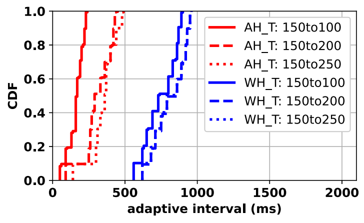
Fig. 8(a) presents the result when the two flows start sending their packets at the same slot. A common destination on the first hop receives all the packets of two flows at the same slot, which validates that Offset$_2$ with 3 ms in the TxOffset mechanism is reasonable. In Fig. 8(b) and (c), HI-flow (τ_H) starts sending one slot and two slots later than LO-flow (τ_L), respectively. Even though the HI-flow τ_H transmits its packets later than LO-flow, they arrive at the destination earlier than those of τ_L and there are no transmission conflicts. This indicates that selecting Offset$_3$ with 5 ms does not cause packet loss from the conflicts and the intended *preemption effect* works properly. On the contrary, in Fig. 8(d) and (e), τ_L begins sending packets at 1 and 2 slots late, respectively. No packet loss occurs in various environment settings, thus AH can provide the proper operation of end-to-end delivery.

Additionally, these experiments show that the piggyback mechanism does not impose an overhead on the protocol operation because we conduct the experiments with maximum payload size packets. Especially, in Fig. 8(b), when the source s_H of τ_H transmits a packet to node u at which u of τ_L is ready to send its packet to node v. Before u transmits τ_L's packet, it can recognize τ_H's packet thanks to *TxOffset* mechanism even though s_H uses the maximum packet size.

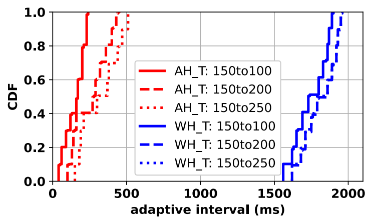
2) Network Adaptability: The advantage of AH is that the network parameters can be updated in real-time, independent of the dissemination interval in global schedule broadcasting. We evaluate the real-time adaptability of the AH protocol in comparison with the WH protocol, where WH uses a deadline monotonic (DM) scheduling policy. Once we change a transmission period parameter at a certain time instant, we evaluate how quickly the changed values can be applied to the system. We use a metric of the adaptation latency, which is the interval between the time instant of the parameter changed and the time instant that a packet reception interval at a destination node follows the changed period for the first time.

We randomly select six devices in our testbed. As an initial setting for two flows, a transmission period is $T_{\{H,L\}} = 150$ ms and a transmission opportunity per link is $\varepsilon_{\{H,L\}} = 2$. The central server changes a period parameter of τ_L to the predetermined values ($T_L = 100$, 200, or 250 ms) at a random time instant. To consider practical environments on the centralized network, we show the adaptation latency of τ_L by changing the superframe length (1000 and 2000 ms). These values can occur in a real network because the length is generally determined by the least common multiple of all flows [8], [10]. The reason that we analyze τ_L's performance instead of τ_H is toF verify the worst-case real-time adaptability because τ_L has a relatively large end-to-end delay.

Fig. 9(a) shows CDF of the adaptation latency with respect to 1000 results with the frame length of 1000 ms. The red and blue lines are about AH and WH, respectively. The solid, dashed, and dotted lines are periods to be changed at a certain time, from 150 to 100 ms and 200 and 250 ms, respectively. From this figure, we can verify that the AH's intervals on all of the changed periods are shorter than that of WH. WH can only adopt the changed period by broadcasting a new global schedule after finishing the superframe. On the other hand, since AH is built to relay packets regardless of the global schedule, it can adopt the changed period at any time, thereby guaranteeing real-time adaptability. In addition, as Fig. 9(b), AH can provide performance independent of the frame length, which shows 74% (from 1950 to 510 ms) improved adaptability performance compared with that of WH at the worst-case results. The finding from this

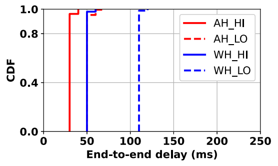


(a)

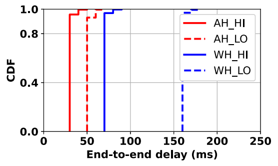


(b)

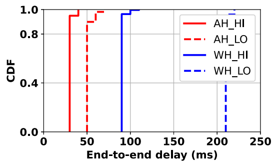
Fig. 9. Real-time adaptability of the AH and WH protocols depending on the superframe length. (a) Frame length 1000 ms. (b) Frame length 2000 ms.



(a)



(b)



(c)

Fig. 10. End-to-end delay CDF of AH and WH protocols with respect to the transmission opportunity. (a) Transmission opportunity of 2. (b) Transmission opportunity of 3. (c) Transmission opportunity of 4.

experiment is that AH can provide improved adaptability for the period parameter change, and it is independent of the frame length change.

3) Real-Time Performance: To compare the real-time performance of two protocols (AH and WH), we consider the end-to-end delay and the schedulable ratio as performance metrics. We assume that packets of flows are generated at the same time and have the same transmission periods on each protocol.

Fig. 10 shows CDF of the end-to-end packet delay of the two protocols with respect to the transmission opportunity. In Fig. 10(a), two flows (τ_H and τ_L) have the transmission opportunity with 2 per link. In the perspective of τ_H, WH has to schedule redundant slots per link in accordance with the transmission opportunity, which may result in a long end-to-end delay.

On the other hand, AH can allocate flexible retransmission slots per link in case of packet loss, providing a relatively short delay. In the perspective of τ_L, WH allows τ_L to transmit its packets after finishing τ_H's delivery due to DM scheduling. This results in a more accumulated delay. However, since τ_L of AH tries to send packets quickly unless it does not cause transmission conflicts with τ_H, the delay difference between the protocols becomes greater.

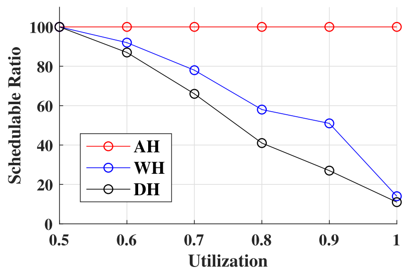


Fig. 11. Schedulable ratio with respect to network utilization.

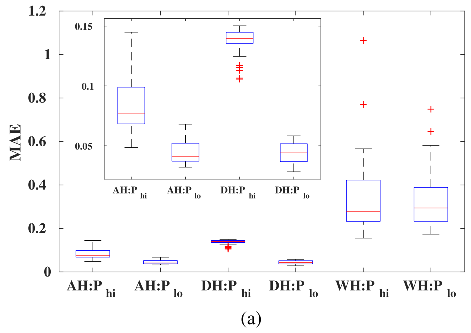
In Fig. 10(b) and (c), the more opportunities are assigned, the longer delay WH experiences due to the increased number of redundant slots. AH can maintain the delay performance unless the channel condition deteriorates. In the results of the transmission opportunity of 2, τ_H and τ_L of AH achieve at most 40% (from 50 ms on WH to 30 ms on AH) and 55% (from 110 ms to 50 ms) reduced end-to-end delay performance compared to those of WH, respectively. Moreover, 67% (from 90 ms to 30 ms) and 76% (from 210 ms to 50 ms) are reduced in the transmission opportunity of 4, respectively.

In a traditional central network, the end-to-end delay results change whenever global scheduling is regenerated (e.g., increasing retransmission parameters). However, we show that the end-to-end delay performance of AH is independent of the retransmission parameter change. This implies that the real-time performance of AH is not significantly affected by parameter changes.

We evaluate the schedulable ratios between AH, WH, and DH [12] by using WCPS, where DH allows each node to select its own time slot by exchanging local knowledge to neighbor nodes in distributed multihop networks. The schedulable ratio is a ratio of schedulable flows for given flow sets. We implement a network topology where the routing paths of two flows (τ_H and τ_L) overlap each other and consist of end-to-end distance with four hops. We identify a set of schedulable flows on a specific network utilization U. Here, U is a sum of the utilization of flows, $U = \sum_{i=\{H,L\}} U_i$, where $U_i = C_i/T_i$. The flow set is a combination of possible periods of flows. We set a fixed transmission opportunity (fixed C_i) and generate 100 random T_i sets, satisfying a given U [4]. An unschedulable flow set is defined as a deadline miss ratio exceeds 1%, where the miss ratio is obtained from an average of 100 simulation runs on a given flow set. The flow deadline is the same as its period.

In Fig. 11, both of them achieve 100% ratios when $U = 0.5$, but when $U = 1$, AH maintains the performance of 100%, while WH and DH decrease to 14% and 11%, respectively. WH results in poor performance because there is no room to schedule the fixed retransmission slots per link within the given deadline. DH makes it difficult to build end-to-end sequential scheduling considering the routing path through only local knowledge exchange. As a result, as utilization increases, the deadline missing more occurs. Meanwhile, AH is capable of sequential delivery to the routing path and flexible retransmissions, so that it tries to save unused time slots, which maintain 100% schedulable ratio even in high utilization. AH can meet the deadlines as long as there is no severe packet loss that causes transmission delays. The finding from this experiment is that AH can have more flexible flow characters even in high-utilization environments due to resource efficiency.

[4]One hundred flow sets on a particular utilization are generated using the UUnifast algorithm [30].



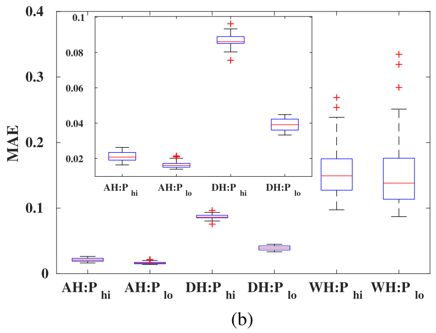


Fig. 12. Comparison of control performance achieved with application of AH and WH protocols. The figures present MAE distributions over 50 simulation runs on each utilization. (a) MAE on the same utilization. (b) MAE on the maximum utilization.

4) Control Performance: The proposed protocol can save wasted time resources in dissemination intervals and retransmission slots. In this experiment, we investigate the impact of resource efficiency on control performance.

We run simulations of a linear time-invariant system. The physical system is a double integrator system, which is a typical example of networked control systems [10]. We simulate two control loops sharing a wireless network. On each loop, a physical system P_H (P_L) is controlled by its corresponding controller via its actuating flow τ_H (τ_L), respectively. The actuating flows have end-to-end overlapped 5-hop distances.

We measure the performance for P_H and P_L applying each of the AH, WH, and DH protocols. The metric of control performance is a mean absolute error (MAE) in our simulations, in which the error is the difference between the optimal state trajectory controlled by wired networks and the trajectory controlled by each protocol. We carry out two experiments as follows: a) two protocols have the same network utilization (SameUtil); b) two protocols use their maximum utilization (MaxUtil) while guaranteeing the schedulability condition.

We adopt a particular utilization adapting flow periods under fixed retransmission parameters ($\varepsilon_{H,L} = 2$). In experiment a), two protocols use the same periods as 300 ms. In experiment b), WH with DM policy sets the minimum periods satisfying the sufficient condition for a schedulability condition from [31]. DH sets minimum periods of two flows as long as end-to-end scheduling is possible within its deadline when it applies vertex coloring algorithm in a given routing path. AH sets the minimum periods satisfying 99% of condition (10).

In Fig. 12(a), P_H and P_L of AH achieve 74% and 87% better control performance than those of WH on SameUtil, respectively. Also, compared with DH, the average control performance of P_H on AH is 38% better than DH, because AH can reduce the end-to-end transmission delay by saving the wasted retransmission slots. When we compare the control performance improvement of MaxUtil, in Fig. 12(b), against SameUtil, AH achieves 75% and 63% improvement for P_H and P_L, respectively, while WH (DH) achieves only 53% (37%) and 52% (12%) improvements, respectively. WH (DH) has a limitation of available minimum flow periods to 250 ms (240 ms), respectively, due to the fixed redundant slots (wasted time resource). On the other hand, since AH does not have the wasted slots, it can select more flexible flow periods of 110 ms, which results in greater improvement of control performance than those achieved with other protocols. Consequently, the proposed protocol can provide more frequent controls in dynamic environments.

VII. Conclusion

In this article, we have proposed the AH MAC protocol, which can adapt network parameters in real-time. Our protocol has significantly improved network resource efficiency by saving the wasted resources of dissemination intervals and unused retransmission slots. We have also proposed a SRTA to guarantee stochastic real-time performance in wireless multihop networks. Our performance evaluation has shown that the proposed SRTA can compute accurate end-to-end delay distributions for given parameters.

Through evaluations, we have found that AH is capable of real-time adaptability to the period parameter change and end-to-end delay performance is independent of the retransmission parameter change. In addition, due to efficient resource consumption, we have found that it is possible to improve the schedulable ratio and control performance in a significant manner.

AH has multiple advantages over existing approaches as described above, but it still has a limitation that only 2 types (high and low priority) of flow can be overlapped on their routing paths. Because transmission conflicts can be avoided by splitting the time slot duration. A possible line of future work is to upgrade it so that more than 2 flows can coexist on the overlapped routing path.

References

[1] X. Tian, Y.-H. Zhu, K. Chi, J. Liu, and D. Zhang, "Reliable and energy-efficient data forwarding in industrial wireless sensor networks," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1424–1434, Sep. 2017.

[2] C. Xu, M. Wu, Y. Xu, and Y. Xu, "Shortest uplink scheduling for NOMA-based industrial wireless networks," *IEEE Syst. J.*, vol. 14, no. 4, pp. 5384–5395, Dec. 2020.

[3] R. Z. K.-J. Park and X. Liu, "Cyber-physical systems: Milestones and research challenges," *Comput. Commun.*, vol. 36, no. 1, pp. 1–7, 2012.

[4] K.-J. Park, J. Kim, H. Lim, and Y. Eun, "Robust path diversity for network quality of service in cyber-physical systems," *IEEE Trans. Ind. Inform.*, vol. 10, no. 4, pp. 2204–2215, Nov. 2014.

[5] F. Group, "WirelessHART specification." [Online]. Available: https://fieldcommgroup.org/technologies/hart/hart-technology

[6] F. Dobslaw, T. Zhang, and M. Gidlund, "End-to-end reliability-aware scheduling for wireless sensor networks," *IEEE Trans. Ind. Inform.*, vol. 12, no. 2, pp. 758–767, Apr. 2016.

[7] Y. Ma, D. Gunatilaka, B. Li, H. Gonzalez, and C. Lu, "Holistic cyber-physical management for dependable wireless control systems," *ACM Trans. Cyber- Phys. Syst.*, vol. 3, no. 1, 2018, Art. no. 3.

[8] R. Brummet, D. Gunatilaka, D. Vyas, O. Chipara, and C. Lu, "A flexible retransmission policy for industrial wireless sensor actuator networks," in *Proc. IEEE Int. Conf. Ind. Internet*, 2018, pp. 79–88.

[9] Y. Chen *et al.*, "Probabilistic per-packet real-time guarantees for wireless networked sensing and control," *IEEE Trans. Ind. Inform.*, vol. 14, no. 5, pp. 2133–2145, May 2018.

[10] P. Park, P. Di Marco, and K. H. Johansson, "Cross-layer optimization for industrial control applications using wireless sensor and actuator mesh networks," *IEEE Trans. Ind. Electron.*, vol. 64, no. 4, pp. 3250–3259, Apr. 2017.

[11] Y. Ma, C. Lu, and Y. Wang, "Efficient holistic control: Self-awareness across controllers and wireless networks," *ACM Trans. Cyber- Phys. Syst.*, vol. 4, no. 4, 2020, Art. no. 41.

[12] V. P. Modekurthy, A. Saifullah, and S. Madria, "DistributedHART: A distributed real-time scheduling system for WirelessHART networks," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp.*, 2019, pp. 216–227.

[13] T. Zheng, M. Gidlund, and J. Åkerberg, "WirArb: A new MAC protocol for time critical industrial wireless sensor network applications," *IEEE Sensors J.*, vol. 16, no. 7, pp. 2127–2139, Apr. 2016.

[14] H. Farag, E. Sisinni, M. Gidlund, and P. Österberg, "Priority-aware wireless fieldbus protocol for mixed-criticality industrial wireless sensor networks," *IEEE Sensors J.*, vol. 19, no. 7, pp. 2767–2780, Apr. 2019.

[15] C. Wu, M. Sha, D. Gunatilaka, A. Saifullah, C. Lu, and Y. Chen, "Analysis of EDF scheduling for wireless sensor-actuator networks," in *Proc. IEEE Int. Symp. Qual. Serv.*, 2014, pp. 31–40.

[16] A. Aijaz and A. Stanoev, "Closing the loop: A high-performance connectivity solution for realizing wireless closed-loop control in industrial IoT applications," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 11860–11876, Aug. 2021.

[17] D. J. Vergados, N. Amelina, Y. Jiang, K. Kralevska, and O. Granichin, "Toward optimal distributed node scheduling in a multihop wireless network through local voting," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 400–414, Jan. 2018.

[18] I. Jabandžić, S. Giannoulis, R. Mennes, F. A. De Figueiredo, M. Claeys, and I. Moerman, "A dynamic distributed multi-channel TDMA slot management protocol for Ad Hoc networks," *IEEE Access*, vol. 9, pp. 61864–61886, 2021.

[19] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," in *Proc. 10th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, 2011, pp. 73–84.

[20] D. Yang, Y. Xu, and M. Gidlund, "Coexistence of IEEE802. 15.4 based networks: A survey," in *Proc. 36th Annu. Conf. IEEE Ind. Electron. Soc.*, 2010, pp. 2107–2113.

[21] ISA100.11a. [Online]. Available: https://www.isa.org/isa100/

[22] J. Lu and K. Whitehouse, "Exploiting the capture effect for low-latency flooding in wireless sensor networks," in *Proc. ACM Conf. Embedded Netw. Sensor Syst.*, 2008, pp. 409–410.

[23] P. Djukic and S. Valaee, "Link scheduling for minimum delay in spatial re-use TDMA," in *Proc. 26th IEEE Int. Conf. Comput. Commun.*, 2007, pp. 28–36.

[24] D. Maxim and L. Cucu-Grosjean, "Response time analysis for fixed-priority tasks with multiple probabilistic parameters," in *Proc. IEEE 34th Real-Time Syst. Symp.*, 2013, pp. 224–235.

[25] Y. Abdeddaïm and D. Maxim, "Probabilistic schedulability analysis for fixed priority mixed criticality real-time systems," in *Proc. Des., Automat. Test Eur. Conf. Exhib.*, 2017, pp. 596–601.

[26] J. L. Díaz *et al.*, "Stochastic analysis of periodic real-time systems," in *Proc. 23rd IEEE Real-Time Syst. Symp.*, 2002, pp. 289–300.

[27] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire Tinyos applications," in *Proc. 1st Int. Conf. Embedded Netw. Sensor Syst.*, 2003, pp. 126–137.

[28] B. Li *et al.*, "Realistic case studies of wireless structural control," in *Proc. ACM/IEEE 4th Int. Conf. Cyber- Phys. Syst.*, 2013, pp. 179–188.

[29] Crossbow, "TelosB mote platform." [Online]. Available: https://www.willow.co.uk/TelosB_Datasheet.pdf

[30] E. Bini and G. C. Buttazzo, "Measuring the performance of schedulability tests," *Real-Time Syst.*, vol. 30, no. 1/2, pp. 129–154, 2005.

[31] E. Bini and G. C. Buttazzo, "Schedulability analysis of periodic fixed priority systems," *IEEE Trans. Comput.*, vol. 53, no. 11, pp. 1462–1473, Nov. 2004.