Towards Mixed Criticality Industrial Wireless Sensor-Actuator Network

Prashant Modekurthy
University of Nevada, Las Vegas, USA

Mahbubur Rahman
City University of New York, NY, USA

Abusayeed Saifullah
Wayne State University, MI, USA

ABSTRACT

In industrial Internet of Things, feedback control loops employed over wireless sensor-actuator network (WSAN) for various process monitoring and control applications require real-time communication for stability. In the real world, most complex control systems are, de facto, Mixed-Criticality (MC) system, meaning that all control loops are not equally critical for the systemâĂŹs correct operation. While the notion of mixed-criticality has been studied widely in CPU scheduling, it still remains largely unexplored for wireless domain. For MC CPU scheduling, the key challenge stems from the uncertainty of worst-case execution times, while the uncertainty in WSAN comes from unpredictable channel conditions and plant dynamics. In this paper, we formulate the MC scheduling problem, formally define the MC semantics for WSAN, and propose MC real-time scheduling in multihop WSAN that allows co-scheduling of the loops for handling dynamic criticality changes. This proposed approach exploits the capture effects of the radios for dynamic resource allocation and reclamation when criticality changes. Then, by exploiting the unused channel capacity of WSAN, we propose a technique to minimize redundancy in high criticality control loop scheduling while preserving the communication reliability and MC constraints, thereby enhancing MC schedulability.

CCS CONCEPTS

• **Networks → Sensor networks**; **Cyber-physical networks**.

KEYWORDS

Wireless Sensor-Actuator Networks, Internet of Things, Mixed-Criticality.

ACM Reference Format:
Prashant Modekurthy, Mahbubur Rahman, and Abusayeed Saifullah. 2023. Towards Mixed Criticality Industrial Wireless Sensor-Actuator Network. In *24th International Conference on Distributed Computing and Networking (ICDCN 2023), January 4–7, 2023, Kharagpur, India.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3571306.3571447

1 INTRODUCTION

Industrial wireless sensor-actuator network (WSAN) powered by Internet of Things (IoT) is evolving as an important class of industrial cyber-physical systems (CPS). Due to cost and limited spectrum, an increasing trend in WSAN is to host many control loops on the same network, enabling diverse monitoring and control applications. In an industrial WSAN based on WirelessHART [9], ISA-100 [6], or WIA-PA [10], the control plane lies at a centralized network manager that manages the network as well as the control loops. For each feedback control loop between sensors and actuators, the sensors measure process variables and deliver them to the controller through the network. The controller sends control commands to the actuators that operate the control and safety components to adjust physical processes. Feedback control loops in a WSAN therefore impose real-time requirements on communication latency between sensors and actuators for stability [21].

A practical approach for dealing with CPS is to treat the system as ***mixed-criticality (MC)*** system where all control loops are not equally critical to its correct operation [7, 8]. For example, the current data of motors have more importance or higher criticality over the pressure data of preheater in a WSAN in cement manufacturing [1]. Motor current must be controlled in real-time to avoid production inefficiency or even explosions. In contrast, the less important preheater pressure data can miss their deadlines occasionally without causing any serious accidents. However, preheater pressure control can have higher priority due to frequent state change. Thus, criticality levels and priorities are different aspects.

MC combined with priority provides a way to guarantee both the importance and real-time requirements of data flows. During normal operations, all data flows are scheduled according to their priorities. During errors or exceptions, the important data are more frequently delivered and more network resources are needed. If there are not enough network resources to serve all flows, the less important data will be sacrificed. ***MC scheduling*** refers to a system where a high-criticality control loop never loses its real-time guarantee before a low-criticality one. In this paper, we address MC in wireless domain and propose a MC scheduling framework for real-time WSAN. Following the seminal notion of Vestal [24] on MC real-time systems, the notion has recently been studied in CPU scheduling [23]. The key challenge in MC scheduling arises from the fact that a task has different worst-case execution times (WCETs) for different criticality levels, of which a scheduling algorithm does not have a priori knowledge. This hinders the conventional real-time scheduling theory to satisfactorily address MC scheduling.

For MC scheduling, uncertainty in CPU scheduling comes from WCETs, while the uncertainty in WSANs comes from unpredictable channel conditions and/or dynamic conditions of the plants. In particular, industrial IoT and CPS applications are characterized by highly dynamic network conditions as well as various activities with different criticalities triggered by asynchronous events. For example, a WSAN in an oil-refinery may suddenly detect the displacement of a safety valve that must be positioned correctly to avoid accidents. In this case, the control loop to operate the valve has higher criticality, while the tank level monitoring receives lower criticality. Once the safety valve is correctly positioned, it is no more critical and the system can enter low-criticality mode. Similarly, if the high-criticality is triggered by channel degradation, the system may come back to low-criticality mode once the channel becomes reliable. Thus, a WSAN may dynamically transition back and forth between the high-criticality and the low-criticality mode. Another challenge stems from the distributed nature of the network which makes it difficult to determine the system's criticality at any time. Till date, there is little progress towards addressing such MC in scheduling transmissions in a WSAN while maintaining the priorities of the control loops.

In this paper, we make the following key contributions. **First**, we formulate the MC scheduling problem, formally define the MC semantics, and propose MC real-time scheduling in WSAN that allows co-scheduling of the loops under different criticalities. In this system, more resources are allocated to high-criticality control loops by sacrificing low criticality ones in high-criticality mode. In low-criticality mode, these resources are dynamically allocated back to low-criticality loops to guarantee their real-time performance. Our approach exploits the capture effects of the radios for dynamic resource allocation and reclamation. **Second**, by exploiting the unused channel capacity without physical layer intervention, we propose a technique to minimize redundancy in high-criticality control loop scheduling while preserving the communication reliability and MC constraints. This technique increases network utilization and significantly enhances MC schedulability.

The rest of the paper is organized as follows. Section 2 and Section 3 describe the related work and system model, respectively. Section 4 presents the proposed MC scheduling framework for wireless network. Section 5 concludes the paper.

2 RELATED WORK

Real-time scheduling for industrial wireless networks has received attention in recent works [15, 17, 22]. But none of these work addresses MC scheduling. The design and analysis for MC real-time scheduling poses fundamentally new challenges. The notion has been widely studied in CPU scheduling (see a recent survey [11]). MC scheduling in wireless poses different challenges since the networked devices are distributed with low capacity and bandwidth, and has to deal with transmission failures and interference which are absent in CPU scheduling. The notion of MC for wireless has been studied recently [12, 18, 19, 26]. However, these works do not propose any technique to handle MC in WSAN. Notably, the work in [12] considers a single-channel wireless network by completely ignoring wireless interference and transmission conflict. Our work is the first systematic study of MC WSAN.

3 SYSTEM MODEL

We consider a general model of WSAN based on existing wireless control standards including WirelessHART [9], ISA-100 [6], and WIA-PA [10]. These low-power multi-hop networks are widely adopted in industrial process monitoring and control due to their feasibility of providing reliable communication in highly unreliable environments. The WSAN forms a multi-hop mesh network consisting of a Gateway, multiple access points, and nodes (sensors and actuators). A centralized network manager and a controller are installed at the *Gateway*. The control plane lies at the centralized *network manager* that is responsible for managing the network, i.e., creating routes and transmission schedules. Multiple *Access points* are wired to the Gateway to avoid bottlenecks between nodes and the Gateway. Each node is equipped with a *half-duplex* omnidirectional radio transceiver, and hence, cannot transmit and receive at the same time, and can receive from at most one sender at a time.

The network adopts a multi-channel TDMA protocol. Time in the network is synchronized and each time slot is of 10 ms. A transmission and its acknowledgement (ACK) need one time slot. The network adopts ***graph routing*** [6, 9]. A *routing graph* is a directed list of paths connecting two devices. Packets from all sensors are routed to the Gateway through the *uplink graph*. For every actuator, there is a *downlink graph* from the Gateway to deliver control messages. The routing graphs can be determined using [21, 25].

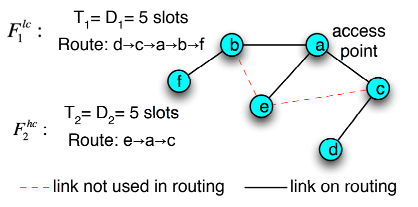
For transmission between a receiver and its sender, a time slot can be either dedicated or shared. In a ***dedicated slot***, only one sender is allowed to transmit to a receiver. In a ***shared slot***, multiple senders can attempt to send to a common receiver using CSMA/CA. Specifically, in a shared slot of a TDMA protocol, a node with a packet to transmit senses the channel using CCA (Clear Channel Assessment) before transmitting, and transmits the packet if the channel is free. Otherwise, it makes a very short random backoff within that slot and retries after that. A packet is scheduled in a shared slot only if its transmission fails in two dedicated slots. Since the probability of scheduling a packet in a shared slot is very low, shared slots are adopted for extreme scenarios. For end-to-end communication between a source (sensor) and destination (actuator), transmissions can be scheduled on multiple paths and multiple slots on routing graphs based on the reliability requirement.

There are n feedback control loops denoted as $F_1, F_2, \cdots, F_n$. Each control loop F_i has a maximum **criticality** C_i based on its importance, which indicates that F_i can run in any criticality between 1 and C_i, depending on the system condition. A higher numerical value of criticality indicates a higher criticality. The period (sampling period of sensors) of loop F_i at the lowest criticality level is denoted by T_i. Its deadline D_i is equal to its period T_i (i.e., $D_i = T_i$). The total time required to deliver a packet from source (sensor) to destination (actuator) of loop F_i is denoted by E_i, and WCET refers to worst-case latency required to deliver a packet. For different criticality levels, the period and the WCET can be different. Specifically, in a higher criticality, the period is $\leq T_i$ and the WCET is $\geq E_i$.

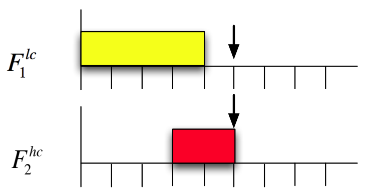
The system is said to be in k-criticality mode at time t, if k is the maximum criticality level among all control loops at that time. The system must transition from criticality i to k (assuming $k > i$) during unexpected interference or operation, where it is no longer able to meet deadlines of all flows with criticality at least i. When the systems is in criticality mode k, the end-to-end delay of each control loop with $C_i \geq k$ must meet its deadline. A system that admits this condition is called **MC schedulable**. The objective of this research is to develop theory and systems for addressing MC scheduling in WSAN.

4 PROPOSED MC SCHEDULING FOR WSAN

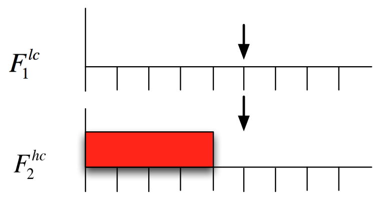
This section presents a framework for MC WSAN that includes MC semantics and MC scheduling.



(a) MC example in wireless: nodes a and c receive on channel 1 and nodes f and b receive on channel 2



(b) If F_2^{hc} does not exceed E_2, then F is feasible



(c) If F_2^{hc} executes for E_2^{ov}, then F^{hc} is feasible

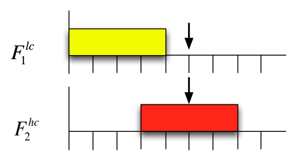
Figure 1: Loop set is MC-schedulable

4.1 Understanding Mixed-Criticality in WSAN

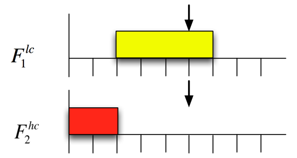
The key challenge in MC scheduling for CPU arises from the fact that a task has different WCETs at different criticality levels. This hinders the conventional real-time scheduling to satisfactorily address MC systems. For MC WSAN, uncertainty can arise from unpredictable communication latencies and channel condition.

Let us consider a system with two criticality levels - **high (HC)** and **low (LC)**. Loops with low criticality, denoted by F^{lc}, have a single WCET estimate E_i, whereas loops with high criticality, denoted by F^{hc}, have a normal WCET estimate E_i and a more conservative one, E_i^{ov} (where $E_i^{ov} \geq E_i$), to take retransmissions into account. Each loop generates an infinite sequence of packets. The set of control loops F is MC schedulable if and only if both of the following conditions are satisfied:

- If each $F_i \in F^{hc}$ uses no more than its optimistic execution time E_i, then there exists a schedule where all loops in F meet their deadlines.
- If one or more loops in F^{hc} exceed their optimistic execution time E_i (but not their conservative estimate E_i^{ov}), then there exists a schedule where all loops in F^{hc} meet their deadlines.



(a) If F_1^{lc} is scheduled at t = 0, F_2^{hc} can miss its deadline



(b) If F_2^{hc} is scheduled at t = 0, F_1^{lc} will miss its deadline

Figure 2: An illustration that no online alg. can guarantee MC-schedulability

Consider a low criticality control loop F_1^{lc} and a high criticality control loop F_2^{hc}, as shown in Fig. 1(a). The sensor and actuator associated with F_1^{lc} are nodes d and f, resp.; and with F_2^{hc} are nodes e and c, resp. Consider the network uses two channels and the dotted lines indicate the links not used in routing (meaning that the transmission of one node can interfere at the node along a dotted link). Consider $T_1 = D_1 = 5$; $T_2 = D_2 = 5$ time slots (where 1 transmission needs 1 slot). From the network, $E_1 = 4$ slots, $E_2 = 2$ slots. Suppose $E_2^{ov} = 4$ slots (when every link along the route will be scheduled on 2 slots). Note that transmissions along $e \to a$ and $d \to c$ cannot happen at the same slot as they use the same channel (channel 1); but those along $b \to f$ and $e \to a$ can happen at the same slot as they use different channels. As illustrated in Figure 1, the loop set is MC-schedulable, since both conditions stated above are met. Nevertheless, Figure 2 illustrates that no online algorithm can guarantee the MC-schedulability of the loops, because for each decision taken at time $t = 0$ (to schedule F_1^{lc} or F_2^{hc}), there exists a situation in which a loop misses its deadline. This example shows that the correct decision that produces an MC-feasible schedule can be taken only by a clairvoyant scheduler that knows (at time $t = 0$) on how many time slots loop F_2^{hc} will make transmissions.

Uncertainty in WSAN scheduling can arise from dynamic behavior of the physical systems. For example, a WSAN in an oil-refinery with a HC loop for operating a safety valve may detect a displacement that must be corrected. In this case, the HC loop can operate at a high sampling rate compared to its low criticality mode.

WSAN may dynamically transition back and forth between different criticalities. In the above example, once the valve is correctly positioned, it is no more critical. Similarly, if the high-criticality is triggered by a channel degradation, the system may be in low-criticality mode once the channel condition improves. A WSAN may enter a high-criticality mode even before a control loop's delay exceeds its worst-case delay estimated in low-criticality level. For example, if the transmission along a link on the path that is used in low-criticality mode fails after multiple retries, the packet will be retransmitted through another link on a different path, and may be considered to be in a higher-criticality mode.

4.2 Defining MC Semantics in WSAN

The MC scheduling semantics states that a high-criticality loop will never lose its real-time guarantee before a low-criticality one. An instance I_i of an MC control loop is characterized by a 4-tuple: $I_i = (r_i, d_i, c_i, e_i)$, where r_i is the release time, d_i is the deadline, c_i denotes the maximum criticality of I_i, and $e_i : \mathbf{N}^+ \mapsto \mathbf{R}^+$ specifies the worst case time requirement of I_i for each criticality level. An MC instance of a system is specified as a finite collection of MC control loop instances: $I = \{I_1, I_2, \cdots, I_n\}$. The actual time requirement θ_i of I_i may vary over runs and are unknown to the scheduler. Let each collection of values $(\theta_1, \theta_2, \cdots, \theta_n)$ be a possible behavior of I. The criticality level of behavior $(\theta_1, \theta_2, \cdots, \theta_n)$ of I is the smallest integer ℓ such that $\theta_i \leq e_i(\ell), \forall i, 1 \leq i \leq n$. If there is no such ℓ then we define that behavior to be erroneous. Our goal is to schedule transmissions of MC loops in the WSAN to ensure correct behavior. A scheduling is correct if it satisfies the following criterion for each $\ell \geq 1$: when scheduling any behavior of criticality level ℓ, it ensures that every loop I_i with $c_i \geq \ell$ receives sufficient time and resource (bandwidth) in interval $[r_i, d_i)$. An MC instance I is called *schedulable* if there exists a correct on-line scheduling.

4.3 MC Co-Scheduling Approach

In low criticality mode, all control loops must meet their deadlines while in high-criticality mode, only the critical loops are required to meet deadlines.Hence, in MC scheduling, the critical control loops need more spatial (such as bandwidth, route redundancy) and temporal resources (such as extra time slots). However, the criticality of a control loop is dynamic, i.e., the network manager is unaware of the criticality level of all control loops durign schedule generation. This raises a key challenge in TDMA scheduling for MC control loops where the time slots need to be scheduled in advance. To manage under limited resources, we propose to co-schedule a control loop under various criticality levels. The extra resources allocated for high-criticality will be used only under high-criticality mode. Thus we co-allocate those resources to low-criticality control loops to allow them to use those in low criticality mode.

We propose MC co-scheduling based on fixed-priority scheduling. Due to its simplicity, fixed-priority scheduling is adopted policy in practice for real-time CPU scheduling, Control-Area Networks, and WSAN [9, 13]. In a *fixed priority scheduling*, each loop has a fixed priority, and the priorities of the loop are usually assigned based on their deadlines, rates, or application demand. Each loop's transmissions are scheduled based on its priority. Starting from the highest priority loop, all transmissions of a control loop are scheduled on one or more routing paths in its routing graph. Before scheduling, the network manager first performs a receiver based channel allocation based on an approach similar to [22]. Each node is assigned a fixed channel to receive message; its neighbors use this channel to send to it.

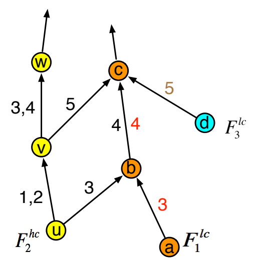
Based on the semantics of MC scheduling in WSAN, when a loop switches to HC mode due to plant dynamics, its sampling rate (i.e., $1/period$) can be higher. On the other hand, when a loop switches to HC mode due to network dynamics, its WCET can be higher. Following this semantic, we assign bandwidth based on criticality, i.e., the HC loops are assigned more bandwidth. The additional bandwidth will be used by a HC loop during HC mode. However, when a HC loop runs in LC mode, the unused additional bandwidth will be exploited by other LC loops. This will allow LC loops to achieve real-time performance in LC mode while guaranteeing real-time performance for HC loops in HC mode.

We explain MC co-scheduling considering two criticalities - HC and LC. Following the order of priorities, we schedule LC and HC loops on their routing graphs. LC loops are scheduled without redundancy or with less redundancy while HC loops are scheduled with high sampling rate and on redundant paths and redundant slots. We maintain strong **temporal isolation** and **spatial isolation** between HC loops so that no transmissions between HC loops are interfered in any mode. In contrast, such isolations are not strictly followed between HC and LC loops. Thus, on the redundant paths and slots of HC loops, we also schedule LC loops, if needed. This is because in LC mode the HC loops do not use these redundant paths and slots. Thus LC loops will use those in LC mode. We try to maximize isolation between HC and LC loops to maximize LC loop schedulability in HC mode for better system performance.

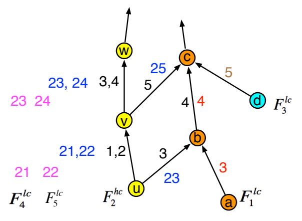
If the system enters HC mode, some loop will use its redundant paths/slots. On that redundant path, a sending node will always choose the HC transmission if there are LC packets at it. However, many receiving nodes are scheduled to receive from multiple nodes in shared slots. In such scenarios, to avoid collision between HC and LC packets in HC mode, we will enable **capture effect** [20] of the radio at the receiver so that HC packet can be received successfully. We can also schedule HC loops assuming a high sampling rate (considering the criticality due to plant dynamics), thereby scheduling its redundant instances which will not be released in LC mode. We will allow LC loops to be scheduled on those redundant slots (of HC loops) in the same way. Here also these extra slots will be used by LC loops in LC mode, and by HC loops in HC mode. Thus criticality mode changes and bandwidth redistribution among LC and HC loops are handled locally and dynamically. be received successfully.

Enabling capture effect. WSAN based on WirelessHART, ISA100, WIA-PA [10] uses IEEE 802.15.4 compliant radios [9]. During the header decoding mode while receiving a packet, a node searches for preambles and start frame delimiter with the strongest Received Signal Strength (RSS) [2, 3, 9]. After this, the radio generates an interrupt and locks to payload reception mode. Therefore, ***capture effect*** [20] can recover the stronger RSS packet (> 3db) if it comes before the radio locks to a colliding packet's payload reception mode, requiring no physical layer modification. Hence, our objective is to ensure that HC transmission in HC mode will always be received before an LC transmission at the same receiver node. Thus for successful transmission (Tx) of HC packet, we adopt the following technique. In HC mode, a node will transmit an HC packet on redundant links immediately after a slot starts and will use the highest Tx power. On the other hand, a node will always transmit LC packets at a moderate Tx power to make the required RSS difference at the receiver. Also, a node will always transmit LC packets after θ time in a slot (while HC packets are transmitted in the very beginning of a slot). This will ensure that the receiver's radio gets locked to an HC packets's payload reception mode and will enable capture effect in case of collisions in HC mode. We can determine the values of θ and Tx power difference through experiments.

Figure 3(a) shows a simple example where a high criticality control loop F_2^{hc} is scheduled along path $u \rightarrow v \rightarrow w \cdots$ on slots 1,2 (on link $u \rightarrow v$) and on slots 3,4 (on link $v \rightarrow w$). This path will be used in its LC mode. To handle failures of both slots on a link, it is also scheduled on redundant paths. For example, it is scheduled along path $u \rightarrow b \rightarrow c \cdots$ on slots 3,4, $\cdots$. But in LC mode, it will not use these slots. So, another LC loop F_1^{lc} has been scheduled along path $a \rightarrow b \rightarrow c \cdots$ on slots 3,4, $\cdots$. In HC mode, if F_2^{hc} follows path $u \rightarrow b \rightarrow c \cdots$ on slots 3,4, $\cdots$, then enabling capture effect will ensure that HC packet will be received at node b at slot 3. Thus it will be guaranteed that HC loop will occupy the redundant path in HC mode. For the same example, Figure 3(b) shows that F_2^{hc} is scheduled for one more instance on slots 21,22 (on link $u \rightarrow v$) and on



(a) Scheduling LC loop on redundant paths



(b) Scheduling LC loop on redundant slots

Figure 3: MC scheduling example

slots 23,24 (on link $v \rightarrow w$) to handle the case when plant dynamic will require its sampling rate to be high in HC mode. These slots are also assigned to LC loops F_4^{lc} (slots 21 and 23) and F_5^{lc} (slots 22 and 24). In LC mode these redundant slots will be used by LC loops.

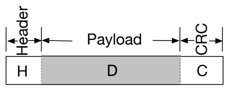
Based on the above idea for MC scheduling under two criticality levels, we shall generalize our approach to any number of criticalities. One potential idea for more than two criticality levels is to consider degree of redundancy in scheduling transmissions of a control loop proportionally with the level of criticality.

4.4 Enhancing MC Schedulability by Reducing Redundancy

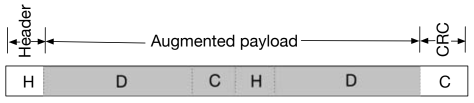
In TDMA scheduling with redundant paths and time slots, many time slots and paths can remain unused in network operation purporting the network to be underutilized. This can reduce the MC schedulability of a set of loops. Since TDMA scheduling for MC WSAN also needs to allocate redundant spatial and temporal resources for HC control loops, we propose a novel technique to reduce such redundancy while maintaining equivalent reliability and respecting MC scheduling constraints. Minimizing redundancy naturally enhances the schedulability of the control loops. This also allows us to maximize isolation between LC and HC loops (while we maintain complete isolation between HC loops).

We propose to reduce temporal redundancy through bit redundancy by exploiting unused channel capacity for HC control loops. Specifically, most WSANs including WirelessHART, ISA100, WIA-PA, and IEEE 802.15.4 based networks have a maximum allowable packet size of roughly 128 bytes of which 104 bytes (roughly) is payload. In practice, their payload is very short, i.e., few tens of bytes only [14]. Thus, it is both feasible and reasonable to increase the payload size if needed. We thus propose to double a packet (by appending a packet at its end) and send in one slot when a control loop is in HC mode instead of scheduling the same packet on two consecutive dedicated time slots on its primary path. Sending a doubled packet in one slot is more efficient than sending two single packets in two separate slots. While the energy consumption of a double-sized packet is higher than a single packet, it can be lower than that consumed when the single packet is transmitted in two slots. Besides, a doubled packet can still be accommodated easily in a 10-ms time slot (which can comfortably accommodate up to 128 bytes). Although a larger (doubled) packet can have less reliability in transmission. While using a high Tx power in HC mode and lower Tx power in LC mode can already compensate this, we argue that reliability is not sacrificed even if different Tx powers are not adopted. Note that only HC packets in HC mode are doubled .

Typically, two consecutive time slots can be highly positively correlated, i.e., a failure in slot i implies a high probability of a failure in slot $i + 1$. Thus the probability of success using two consecutive slots may not dramatically increase from that using just one slot. Besides, in commercial devices, a packet is considered to be lost even if a single bit is lost (after error correction) as the error correction code (such as CRC) calculated does not match. This means that when a packet is considered lost, it does not necessarily mean that all of its bits were lost. That is, in a scenario of packet loss, the receiver receives many bits of the packet which, however, are not sufficient for decoding the packet correctly. All these things happen



(a) An original packet

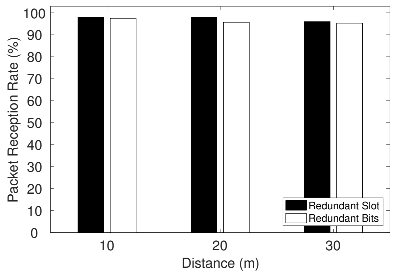


(b) The same packet with augmented payload

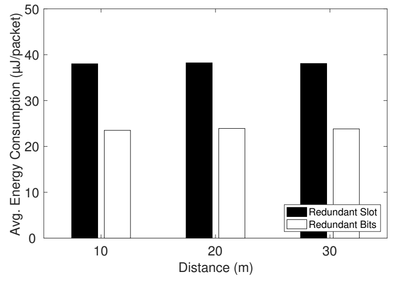
Figure 5: Payload augmentation of HC packet in HC mode

at the physical layer and the link layer is unaware of that. When we double the packet size by appending the same packet at the end, the probability of receiving the bits sufficient to decode a packet becomes much higher. When a (doubled) packet is considered lost at the link layer, it is high probability that the bits received at its physical layer are still sufficient to recover the packet. Thus the transmission reliability after doubling the packet is not sacrificed due to the usage of redundant bits (data).

We have performed some experiments using WirelessHART link layer implementation in GNU Radio [5] on USRP devices [4] to compare the two scenarios. Considering 40-byte packet (which is more than sufficient to carry WSAN message) under **slot redundancy** (in two 10-ms slot) and the scenario under **bit redundancy** that sends a doubled (i.e. 80-byte) packet in one slot, for 1000 packet transmissions, Figure 4(a) shows that the reliability in the two cases are competitive against each other while the average energy consumption under bit redundancy (sending 80-byte packet in one slot) is only 61% of that under slot redundancy (sending one 40-byte packet in two slots). Besides, the time needed for 80-byte packet transmission was less than 4ms (within one time slot).



(a) Reliability



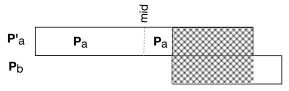
(b) Average energy consumption

Figure 4: Feasibility of exploiting unused channel capacity

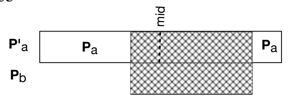
4.5 Handling Dynamic Criticality Changes

Instead of scheduling on two dedicated slots on each link on the primary path, an HC packet is scheduled only in one slot for each link on its primary path like a LC packet. On shared slots, they are scheduled with other LC packets where a receiver is scheduled to receive from multiple transmitters (who contend to transmit using a CSMA/CA approach in that slot). Whenever the criticality changes and a control loop becomes HC, its packet is doubled. We schedule a HC packet by doubling its size where an HC packet is appended to itself. This is done only through payload augmentation without changing the packet structure that has 3 parts: header (H), followed by payload (D), followed by checkcum/CRC (C) as shown in Figure 5(a). As shown in Figure 5(b) for the augmented structure, we retain the original structure of the packet by only augmenting the payload. At the receiver, we perform CRC check after decoding and retrieving payload from the augmented payload section. Since we use higher Tx power in HC mode, sending a doubled packet remains well feasible. Along a dedicated link its transmission thus achieves reliability equivalent to that in transmitting in two different slots (for original packet). High criticality in dedicated slot is thus respected at much lower cost (i.e. with lower redundancy). Note that a packet is doubled only once in HC mode. It is halved to the original packet once its criticality changes back to LC.

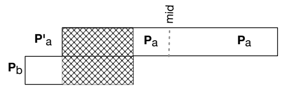
As we have discussed above, in HC mode, high criticality of a packet in dedicated slot is respected through bit redundancy (by doubling the packet). However, in a shared slot an HC packet and an LC packet can collide. In HC mode, the HC packet must be received while we can sacrifice the LC packet. Now we describe how this is achieved in our approach. We show all types of collisions for 2 packets in Figure 6 where P'_a represents the augmented packet of HC packet P_a. The other packet P_b is an LC packet. For any arrival time offsets of the two packets, the HC packet P_a is always decodable. For example, in Figures 6(a) and 6(c), the first half and the second half (each representing P_a), respectively, of P'_a is collision-free and hence can be retrieved. For the case in Figures 6(b), adding 2 collision-free chunks will provide P_a. In the future, we shall extend the above idea of two criticality levels to more than two criticality levels. Our objective is to augment the payload at most to double. We will adopt the scheduling accordingly. One potential direction is to allow conflict between packets of criticality k and $k-1$ in the scheduling.



(a) At least first half of P'_a is collision-free



(b) One chunk in the beginning and one at the end of P'_a are collision-free



(c) At least last half of P'_a is collision-free

Figure 6: HC and LC packet collision types

Implementation Issues. Note that our approach of retrieving HC packet in HC mode is different from existing packet collision resolution techniques, for example that in [16], which need physical layer re-design. Our above proposed decoding can be done without changing the physical layer of off-the-shelf WSAN devices. It can be implemented by working at the driver level. We will first disable the feature so that that radio does not consider packet loss if some bits are lost. We will exploit all correctly received bits that are decoded by the device's existing decoder. If some bits are missing in the first half of the (augmented) packet, we may find those from the second half. Thus an ACK will be sent after a packet is reconstructed from the received bits. A packet will be considered lost only when the received bits are not sufficient to reconstruct the packet.

5 CONCLUSION AND FUTURE WORK

In this paper, we have studied MC real-time scheduling for WSAN. We have formulated the MC scheduling problem, and defined the MC semantics for WSAN, and proposed MC real-time scheduling that allows co-scheduling of the feedback control loops for handling dynamic criticality changes in industrial IoT. Then, by exploiting the unused channel capacity of WSAN, we have proposed a technique to minimize redundancy in high criticality control loop scheduling while preserving the communication reliability and MC constraints, thereby enhancing MC schedulability. We have presented our approaches considering two criticality levels. In the future, we shall extend our approach to consider more criticality levels. We shall also implement and evaluate our approach on our physical testbed.

ACKNOWLEDGMENTS

This work was supported by NSF through grants CNS-2301757, CAREER- 2211523, CCF-2118202, CNS-2211642, and by ONR through grant N00014-22-1-2155.

REFERENCES

[1] [n. d.]. http://www.climatetechwiki.org/technology/energy-saving-cement.
[2] [n. d.]. http://www.atmel.com/images/doc8111.pdf.
[3] [n. d.]. http://www.ti.com/product/cc2420.
[4] [n. d.]. http://www.ettus.com/product/details/UB210-KIT.
[5] [n. d.]. GNU Radio. http://gnuradio.org.
[6] [n. d.]. ISA100. http://www.isa.org/MSTemplate.cfm?MicrositeID=1134&CommitteeID=6891.
[7] [n. d.]. Mixed Critical Systems. http://www.nitrd.gov/about/blog/white_papers/20-Mixed_Criticality_Systems.pdf.
[8] [n. d.]. Resilient Mixed Criticality Systems. http://www.crosstalkonline.org/storage/issue-archives/2009/200909/200909-Sha.pdf.
[9] 2007. WirelessHART Specification. http://www.hartcomm2.org.
[10] 2011. I.E.C.C. IEC/PAS 62601: IndustriaL Communication Networks – Fieldbus Specifications – WIA-PA Communication Network and Communication Profile. IEC: Worcester, MA, USA.
[11] A. Burns and R. Davis. 2018. A Survey of Research into Mixed Criticality Systems. *ACM Comput. Surv.* 50, 6 (Nov. 2018), 1–37.
[12] A. Burns, J. Harbin, L. Indrusiak, I. Bate, R. Davis, and D. Griffin. 2018. AirTight: A Resilient Wireless Communication Protocol for Mixed-Criticality Systems. In *IEEE RTCSA*. 65–75.
[13] G. Butazzo. 2005. *Hard Real-Time Computing Systems*. Springer.
[14] E. Callaway. 2003. *Wireless Sensor Networks: Architectures and Protocols*. Auerbach Publications. Page. 206.
[15] S. Fahmida, P. Modekurthy, D. Ismail, A. Jain, and A. Saifullah. 2022. Real-Time Communication over LoRa Networks. In *2022 IEEE/ACM IoTDI*. 14–27.
[16] S.h Gollakota and D. Katabi. 2008. Zigzag Decoding: Combating Hidden Terminals in Wireless Networks. In *ACM SIGCOMM*. 159–170.
[17] X. Jin, A. Saifullah, C. Lu, and P. Zeng. 2019. Real-time scheduling for event-triggered and time-triggered flows in industrial wireless sensor-actuator networks. In *IEEE INFOCOM*. IEEE, 1684–1692.
[18] X. Jin, J. Wang, and P. Zeng. 2015. End-to-end delay analysis for mixed-criticality WirelessHART networks. *IEEE/CAA Journal of Auto. Sinica* 2, 3 (2015), 282–289.
[19] X. Jin, C. Xia, H. Xu, J. Wang, and P. Zeng. 2016. Mixed Criticality Scheduling for Industrial Wireless Sensor Networks. *Sensors* 16, 9 (2016).
[20] J. Lu and K. Whitehouse. 2008. Exploiting the Capture Effect for Low-latency Flooding in Wireless Sensor Networks. In *ACM SenSys*.
[21] P. Modekurthy, A. Saifullah, and S. Madria. 2018. Distributed Graph Routing for WirelessHART Networks. In *ICDCN*. 24:1–24:10.
[22] P. Modekurthy, A. Saifullah, and S. Madria. 2021. A Distributed Real-time Scheduling System for Industrial Wireless Networks. *ACM TECS* 20, 5 (2021), 1–28.
[23] J. Ren and booktitle=ECRTS title=Mixed-Criticality Scheduling on Multiprocessors Using Task Grouping year=2015 pages=25-34 Phan, L. [n. d.].
[24] S. Vestal. 2007. Preemptive Scheduling of Multi-criticality Systems with Varying Degrees of Execution Time Assurance. In *IEEE RTSS*. 239–243.
[25] C. Wu, D. Gunatilaka, A. Saifullah, M. Sha, P. B. Tiwari, C. Lu, and Y. Chen. 2016. Maximizing Network Lifetime of WirelessHART Networks under Graph Routing. In *IoTDI*. 176–186.
[26] C. Xia, X. Jin, L. Kong, and P. Zeng. 2017. Bounding the Demand of Mixed-Criticality Industrial Wireless Sensor Networks. *IEEE Access* 99 (2017).