Chapter 2
Schedulability Analysis of Mixed-Criticality Data Under Fixed-Priority Scheduling

Abstract WirelessHART, as a robust and reliable wireless protocol, has been widely-used in industrial systems. Its real-time performance has been extensively studied, but limited to the single-criticality case. Many advanced applications have mixed-criticality communications, where different data flows come with different criticality levels. Hence, in this chapter, we study the real-time mixed-criticality communication based on WirelessHART networks, and present an end-to-end delay analysis method under fixed priority scheduling.

2.1 Background

WirelessHART is based on a centralized network management and multi-channel Time Division Multiple Access (TDMA). These special features have attracted researchers' attentions, and they have done some studies to improve the real-time performance of WirelessHART networks, e.g. [1–4]. However, all these studies focus on the single-criticality case. Advanced real applications come with mixed-criticality data communications, such as the case of the cement factory in Sect. 1.2.2.

The key difference between mixed- and single-criticality systems is that the criticality of data in mixed-criticality systems must be considered together with real-time performance [5]. This leads to the problem that directly using traditional priority-based scheduling algorithms of single-criticality systems to mixed-criticality systems is infeasible due to independence between criticality and traditional priorities [6–12]. Therefore, the traditional real-time theory needs a revision to support mixed-criticality networks.

There are a few related studies on mixed-criticality networks. The work in [13–15] designs Network-on-Chips for mixed-criticality multiprocessor systems. The work in [16] proposes mixed-criticality protocols for the Controller Area Network (CAN), and then a response-time analysis method and an optimal priority assignment scheme are provided. The work in [17] designs a virtual CAN controller

X. Jin et al., *Mixed-criticality Industrial Wireless Networks*, Wireless Networks,
https://doi.org/10.1007/978-981-19-8922-3_2

to provide differentiated services for different criticality levels. The work in [18–20] focuses on the wired network—TTEthernet. They propose some scheduling algorithms to guarantee the performance of messages under real-time constraints. The work in [21] is about wireless networks. It introduces a mixed-criticality scheduling method to JPEG2000 video systems based on the IEEE 802.11 standard. However, WirelessHART networks are based on the IEEE 802.15.4 standard and are quite different from the wireless video system. Therefore, existing system models and solutions cannot be used in the WirelessHART model.

Since the end-to-end delay analysis is the foundation of the real-time theory, in this chapter, we present an end-to-end delay analysis method for fixed priority scheduling in mixed-criticality real-time WirelessHART networks. The analysis can be used to test whether the data flows can meet their special requirements when designing a WirelessHART network.

In the following, first, we introduce the concept of mixed criticality into real-time wireless sensor-actuator networks and propose a formulated system model; second, we propose an end-to-end delay analysis method, which is a fast feasible method to test the reliability of mixed-criticality systems; third, evaluation results show that the proposed method is very effective and only incurs little pessimism comparing with simulation results and a real testbed.

2.2 System Model

2.2.1 Mixed-Criticality Wireless Network Model

We consider a WirelessHART network characterized by $G =< V, E, m >$:

- A WirelessHART network consists of sensor/actuator nodes and a gateway with a centralized network manager. We use n nodes $V = \{v_1, v_2, \ldots, v_n\}$ to denote these devices, and the gateway is v_1. Each node is equipped with a transmitter, so it cannot send and receive in the same time slot.
- $E : V \times V$ is the set of links. Each element e_{ij} in E represents existing reliable communication between v_i and v_j. Transmitting a packet through one link is called *transmission*.
- We use m to denote the number of available channels. WirelessHART networks support 16 non-overlapping channels. However, since these channels may suffer from persistent external interference, not all of them can always be accessed. Hence, $0 < m \leq 16$. Each channel supports only one transmission in one time slot.

The data flow set is denoted by $\mathbb{F} = \{F_1, F_2, \ldots\}$. Each flow F_i is characterized by $F_i =< \chi_i, c_i, t_i(x), p_i, \phi_i >$. p_i denotes the distinct fixed priority. ϕ_i ($\phi_i \subseteq E$) is an ordered sequence of links and denotes the routing path of the flow F_i. The centralized manager of the WirelessHART network collects sensing data and distributes actuator data, so the gateway is the source or destination for each flow. In the TDMA policy, each time slot allows a one-hop data transmission and its acknowledgement to be transmitted. We use c_i to denote the number of time slots required to deliver a packet from the source to the destination, i.e., c_i is equal to the number of hops of the flow F_i.

χ_i denotes the criticality level of the flow F_i. For ease of presentation, we only focus on a dual-criticality system, in which there are only two criticality levels L (low) and H (high). However, it can be easily extended to systems with an arbitrary number of criticality levels. Correspondingly, the network also has dual-criticality modes $\{H, L\}$. If the criticality level of the flow F_i is not less than the current network mode x, it can be delivered; otherwise, the flow is discarded. The network starts in the low-criticality mode ($x = L$), in which all flows are served. When an error or exception occurs in a node, the node will trigger the changing of the network mode from low criticality to high criticality ($x = H$). Then only the flows with high-criticality level can be delivered, and the low-criticality flows are discarded. Note that the mode change will introduce additional time to the delay of the high-criticality flow and the message of mode change should be broadcast to the entire network as soon as possible. There are some methods used to solve this problem. For example, one channel of each node can be reserved to serve the message. Therefore, we only model the duration of the mode change as C, which is used to calculate the delay of the packet delivered during the mode change.

When errors and exceptions occur, workers will handle problems and change the mode from high criticality to low criticality. We do not consider this process due to the unpredictability of workers' behavior, i.e., we do not study the mode change from high criticality to low criticality. The assumption is also widely adopted in existing works (such as [22–24] etc.).

In mixed-criticality uniprocessor/multiprocessor systems, the execution time of a task is a function of the system mode. In wireless networks, the number of time slots required to deliver a packet is equal to the number of hops and fixed. However, the period t_i is dependent on the network criticality mode. Since the important flow is more frequently delivered when the network mode is changed to high criticality, hence, $t_i(H) < t_i(L)$.

According to the period $t_i(x)$, the flow F_i periodically releases a packet, which is assigned the parameters specified in F_i. Our system adopts the implicit-deadline, i.e., the packet's relative deadline is equal to the flow's period corresponding with the network mode of generating the packet. For example, if the packet is released in the network mode L, then its relative deadline is $t_i(L)$. Therefore, in a stable network, at most one packet of each flow is active at any time. However, when the network mode is changed from L to H, there may exist two active packets belonging to one flow because of the change of the flow's period. In this case, the packet released in the network mode H has higher priority than another.

A packet is released at time slot s_1, and arrives at its destination at time slot s_2, then its end-to-end delay is $(s_2 - s_1 + 1)$. The end-to-end delay of a flow is the maximum delay among all its packets. If a scheduling algorithm can schedule all flows such that all packets' end-to-end delays are less than or equal to their deadlines, the flow set is called *schedulable* under the scheduling algorithm.

Note that not all of the above assumptions are supported by the original WirelessHART protocol. However, they can be implemented in the application layer [1–3]. In Sect. 2.4, our real testbed is introduced.

2.2.2 *Fixed Priority Scheduling*

We focus on the end-to-end delay analysis for fixed priority scheduling, which is the most commonly used real-time scheduling in real systems. In fixed priority scheduling, transmissions are scheduled within a hyper-period T, which is equal to the least common multiple of periods of all flows, since after that all schedules are cyclically repeated. The period supported by the WirelessHART protocol is 2^i, where i is an integer greater than or equal to 0. Therefore, the hyper-period T is equal to the maximum period among all flows. At each time slot of T, if there exist unused channels, the transmission with the highest priority is scheduled. However, if the released transmission shares nodes with the transmissions that have been scheduled at this time slot, it cannot be scheduled since one node cannot serve more than one transmission at one time slot (as shown in Fig. 2.1). Therefore, there are two factors influencing the transmission scheduling: *channel contention* (there are no unused channels assigned to the transmission) and *transmission conflicts* (a transmission cannot be scheduled, if it shares a node with a transmission that has been scheduled in this time slot). In other words, the two factors introduce extra delays. In the following, we analyze delays introduced by two factors separately.

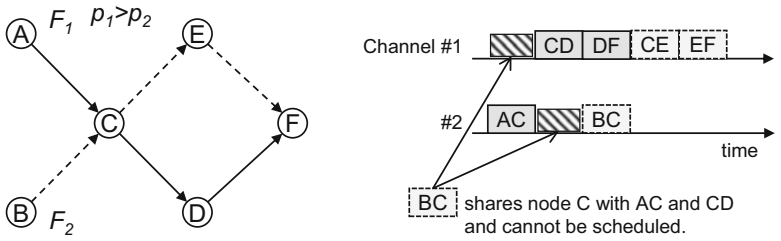

Fig. 2.1 Fixed priority scheduling

Table 2.1 Key notations

Symbol	Definition
G	A WirelessHART network
V	Set of all devices in the network G
E	Set of links in the network G
m	Number of available channels
F_i	A data flow, $F_i \in \mathbb{F}$
χ_i	Criticality level of the flow F_i
c_i	Number of hops of the flow F_i
x	Criticality mode of the current network
$t_i(x)$	Period of the flow F_i at the current mode
p_i	Priority of the flow F_i
ϕ_i	Routing path of the flow F_i
C	Duration of the mode change
s_i	The ith time slot
T	Hyper-period
R_k^{ch}	Pseudo worst case delay of the flow F_k for single networks
$hp(F_k)$	Set of flows whose priorities are higher than F_k
$R_k(x)$	Worst case delay of the packet that belongs to the flow F_k at the network mode x
r	Number of hops that the packet has passed before the mode change

Problem Statement Given the mixed-criticality WirelessHART network G, the flow set $\mathbb{F}$ and the fixed priority scheduling algorithm, our objective is to analyze the end-to-end delay for each flow, such that the schedulability of the flow set can be determined.

Table 2.1 summarizes the key notations used in this chapter.

2.3 End-to-End Delay Analysis

Our analysis is based on the *EDA* (End-to-end Delay Analysis) method [1], which is the state-of-the-art end-to-end delay analysis for fixed priority scheduling in single-criticality real-time WirelessHART networks. To make this chapter self-contained, we first introduce EDA. Then we present our end-to-end delay analysis for mixed-criticality WirelessHART Networks.

2.3.1 Analysis for Single-Criticality Networks

The EDA analysis contains two steps. The first step calculates the delay due to channel contention, which is called *pseudo upper bound* of the worst case end-to-

end delay and denoted by R_k^{ch}. Then the second step incorporates the delay due to transmission conflicts into the result of the first step.

2.3.1.1 Pseudo Delay

The flow F_k experiences the worst case delay when the level-k busy period occurs. The *level-k busy period* is the maximum continuous time interval during which all channels are occupied by flows of priority higher than the priority of F_k, until F_k finishes its active packet transmitting. The notation $hp(F_k)$ is used to denote the set of flows whose priorities are higher than F_k. If the flow F_i ($F_i \in hp(F_k)$) has a packet with release time earlier than the level-k busy period and deadline in the level-k busy period, it is said to have *carry-in* workload in the busy period. Then two types of workload are presented as follows:

- $W_k^{NC}(F_i, \alpha)$ denotes the workload upper bound in the level-k busy period of α slots, if F_i has no carry-in workload:

$$W_k^{NC}(F_i, \alpha) = \left\lfloor \frac{\alpha}{t_i} \right\rfloor \cdot c_i + \min(\alpha \bmod t_i, c_i)$$

  where t_i denotes the period of F_i in single-criticality networks.
- $W_k^{CI}(F_i, \alpha)$ denotes the workload upper bound in the level-k busy period of α slots, if F_i has a carry-in workload:

$$W_k^{CI}(F_i, \alpha) = \left\lfloor \frac{\max(\alpha - c_i, 0)}{t_i} \right\rfloor \cdot c_i + c_i + \mu_i$$

  where $\mu_i = \min(\max(\max(\alpha - c_i, 0) - (t_i - R_i), 0), c_i - 1)$ and R_i denotes the worst case end-to-end delay of F_i in single-criticality networks.

  Similarly, there are two types of interference between F_i and F_k during α slots:

$$I_k^{NC}(F_i, \alpha) = \min(W_k^{NC}(F_i, \alpha), \alpha - c_k + 1) \tag{2.1}$$

$$I_k^{CI}(F_i, \alpha) = \min(W_k^{CI}(F_i, \alpha), \alpha - c_k + 1) \tag{2.2}$$

At most $m - 1$ higher priority flows have carry-in workload in the network with m channels. Therefore, F_k's total delay due to channel contention is

$$\Omega_k(\alpha) = \sum_{F_i \in hp(F_k)} I_k^{NC}(F_i, \alpha) + U_k(\alpha)$$

where $U_k(\alpha)$ is the sum of the $\min(|hp(F_k)|, m-1)$ largest values of the differences $I_k^{CI}(F_i, \alpha) - I_k^{NC}(F_i, \alpha)$ among all $F_i \in hp(F_k)$.

The WirelessHART network contains m channels, so Eq. (2.3) shows the delay due to channel contention. And the pseudo upper bound R_k^{ch} is the minimal value of α that solves Eq. (2.3). α can be found using the iterative fixed-point algorithm [25], which is widely used in the delay analysis of real time systems. The iterative calculation of α starts at $\alpha = c_k$. During the iterations, if α is larger than the deadline of the flow F_k, the algorithm terminates and the flow set is unschedulable; if the value of α is fixed and less than the deadline, the fixed-point is R_k^{ch}.

$$\alpha = \left\lfloor \frac{\Omega_k(\alpha)}{m} \right\rfloor + c_k \tag{2.3}$$

2.3.1.2 Worst Case Delay

This step incorporates the delay due to transmission conflicts into R_k^{ch} to calculate the actual end-to-end delay R_k. First, we introduce some definitions.

- $Q(k, i)$: the total number of F_i's transmissions that share nodes with F_k's transmissions.
- $\delta_j(k, i)$: the number of nodes along the jth maximal common path between F_k and F_i. $\delta'_j(k, i)$ is the length of the maximal common path with a length of at least 4. The delay caused by a maximal common path is at most 3, so the extra length is specially marked using $\delta'_j(k, i)$.
- $\Delta(k, i)$: the upper bound of end-to-end delay due to transmission conflicts that F_i contributes to F_k,

$$\Delta(k, i) = Q(k, i) - \sum_{j=1}^{\sigma} (\delta'_j(k, i) - 3)$$

where σ is the number of maximal common paths between F_k and F_i.

Thus the upper bound of the actual delay R_k is the minimal solution of Eq. (2.4) by running the iterative fixed point algorithm starting at $\beta = R_k^{ch}$.

$$\beta = R_k^{ch} + \sum_{F_i \in hp(F_k)} \left\lceil \frac{\beta}{t_i} \right\rceil \cdot \Delta(k, i) \tag{2.4}$$

2.3.2 Analysis for Mixed-Criticality Networks

Mixed-criticality networks dynamically change the network mode, which results in three types of packets transmitted in the network:

- The release time and deadline of a packet are all in the network mode L. The end-to-end delay of this packet is denoted by $R_k(L)$.
- The release time and deadline of a packet are all in the network mode H. The notation $R_k(H)$ is used to present the upper bound of its delay.
- When the network mode is changed, the packet, which is released by high-criticality flow in the network mode L, cannot be dropped. In this situation, the packet's release time is in the network mode L, but its deadline is in the H mode. Flows formed by these packets are delivered only once. The notation $\mathbb{F}'$ presents the set of these flows and $R_k(L2H)$ denotes the upper bound of the delay.

$R_k(L)$ is unaffected by the mode change and is equal to the delay R_k calculated in single-criticality networks. Therefore, we only analyze $R_k(H)$ and $R_k(L2H)$.

2.3.2.1 Analyzing $R_k(H)$

In the network mode H, packets belonging to the high-criticality flow are delivered, no matter when they are released. Therefore, $R_k(H)$ is interfered by the following two flow sets

$$hpL(F_k) = \{F_i | F_i \in \mathbb{F}', p_i < p_k, \chi_i = H\},$$

$$hpH(F_k) = \{F_i | F_i \in \mathbb{F}, p_i > p_k, \chi_i = H\}.$$

From these, we can derive that the delay due to channel contention is

$$\Omega_k(\alpha) = \sum_{F_i \in hpH(F_k) \cap hpL(F_k)} I_k^{NC}(F_i, \alpha) + U_k(\alpha)$$

where $U_k(\alpha)$ is also for the interferences of $hpH(F_k)$ and $hpL(F_k)$. Note that the interferences of flows in $hpH(F_k)$ are the same with Eqs. (2.1) and (2.2), since the flows release packets periodically. However, the flow F_i ($F_i \in hpL(F_k)$) is delivered only once. In the worst case, its workload in the level-k busy period is $\min\{\alpha, c_i\}$. Therefore,

$$\forall F_i \in hpL(F_k) : I_k^{CI}(F_i, \alpha) = I_k^{NC}(F_i, \alpha) = \min(\min(\alpha, c_i), \alpha - c_k + 1)$$

Then the pseudo upper bound $R_k^{ch}(H)$ can be derived based on Eq. (2.3).

For the delay due to transmission conflicts, similarly, besides the periodic flows in $hpH(F_k)$, the flows in $hpL(F_k)$ delivered only once will introduce $\Delta(k, i)$ to

$R_k(H)$. Therefore,

$$\beta = R_k^{ch}(H) + \sum_{F_i \in hpH(F_k)} \left\lceil \frac{\beta}{t_i(H)} \right\rceil \cdot \Delta(k,i) + \sum_{F_i \in hpL(F_k)} \Delta(k,i) \tag{2.5}$$

According to Eq. (2.5), the iterative algorithm can be used to find the fixed β, i.e., $R_k(H)$.

2.3.2.2 Analyzing $R_k(L2H)$

The flow F_k ($F_k \in \mathbb{F}'$) is divided into two flows. The first flow F_{kL} is delivered in the L mode, and the second flow F_{kH} is in the H mode. We use $R_k^r(L)$ and $R_k^r(H)$ to denote the delays of F_{kL} and F_{kH}, respectively, where r means that the packet has passed through r hops before the mode change, and $r \in [0, c_k - 1]$. Correspondingly, $c_{kL} = r$ and $c_{kH} = c_k - r$. And priorities of F_{kL} and F_{kH} are assigned as p_k.

The calculation of $R_k^r(L)$ is the same as that of $R_k(L)$, since they are all in the stable network. However, $R_k^r(H)$ is different from $R_k(H)$. According to our system model, packets released by F_k in the network mode H have higher priority than the packets of F_{kH}. Therefore, the delay contributed by these higher priority packets must be added to $R_k^r(H)$, i.e., F_{kH} is interfered by $hpL(F_k)$, $hpH(F_k)$ and $\{F_k\}$ in the network mode H. From these, we can derive

$$\Omega_{kH}(\alpha) = \sum_{F_i \in hpH(F_k) \cap hpL(F_k) \cap \{F_k\}} I_{kH}^{NC}(F_i, \alpha) + U_{kH}(\alpha)$$

where U_{kH} is for $hpL(F_k)$, $hpH(F_k)$ and $\{F_k\}$. For the flow F_{kH}, F_k releases higher priority packets periodically. The interference introduced by F_k is the same as that by $hpH(F_k)$. Therefore, Eqs. (2.1) and (2.2) also can be used to calculate it.

For the delay due to transmission conflicts, the packets released by $\{F_k\}$ must be considered. Then the actual end-to-end delay is shown as follows:

$$\beta = R_{kH}^{ch}(H) + \sum_{F_i \in hpH(F_k) \cap \{F_k\}} \left\lceil \frac{\beta}{t_i(H)} \right\rceil \cdot \Delta(kH,i) + \sum_{F_i \in hpL(F_k)} \Delta(kH,i)$$

And $R_k^r(H)$ is also solved by the iterative algorithm.

The range of r is from 0 to $c_k - 1$. If $r = 0$, it means that the packet has been released but not been delivered before the network mode is changed to H. Thus, $c_{kL} = 0$. This will cause the failure of the iterative algorithm. Therefore, if $\exists F_i$ and $p_i > p_k$, α starts with 1; otherwise, there is no interference for F_k and $R_k^r(L) = 0$. If $r = c_k$, it means that the packet has been delivered to its destination in the L network mode. Hence, the delay of the packet is $R_k(L)$.

The delay of F_k is the sum of $R_k^r(L)$ and $R_k^r(H)$. However, different values of r lead to different $R_k^r(L)$ and $R_k^r(H)$. Therefore, the upper bound of F_k's delay is

$$R_k(L2H) = \max_{r \in [0, c_k - 1]} \{R_k^r(L) + R_k^r(H)\} + C$$

where C is the additional time introduced by the mode change (shown in Sect. 2.2).

To sum up, if the flow F_k satisfies $R_k(L) \leq t_k(L)$, $R_k(H) \leq t_k(H)$ and $R_k(L2H) \leq t_k(L)$, then it is schedulable. In a flow set, if all the flows are schedulable, the flow set is schedulable. The calculation of our analysis is in pseudo polynomial time because our analysis is based on the iterative fixed-point algorithm.

2.4 Performance Evaluations

In this section, we will compare our analysis method with simulations and a real testbed.

2.4.1 Simulations

In order to illustrate the applicability of our method, for each parameter configuration, 100 test cases are generated randomly. For each test case, the gateway is placed at the center and other nodes are placed randomly in the playground area A. According to the suggestion in [26], the number of nodes n and the playground area A should satisfy

$$\frac{n}{A} = \frac{2\pi}{d^2\sqrt{27}}$$

where the transmitting range d is set as 40 m. Then, each node connects to the nearest node, which must be in its transmitting range and has been connected to the gateway. If some nodes cannot connect to the gateway, their locations are generated randomly again.

The flow set $\mathbb{F}$ contains $0.8 \cdot n$ flows. Other parameters are set as follows. We use the utilization u ($u = \sum_{\forall F_i \in \mathbb{F}} c_i/t_i$) to control the workload of the entire network, and *UUniFast algorithm* [27] is used to generate each flow's utilization u_i ($u_i = c_i/t_i$). The result generated by UUniFast algorithm follows a uniform distribution and is neither pessimistic, nor optimistic for the analysis. For the flow F_i, its criticality

level is assigned randomly. If $\chi_i = H$, then $t_i(L) = 2^{\lceil \log_2^{ti} \rceil}$ and $t_i(H) = 2^{\lfloor \log_2^{ti} \rfloor}$; otherwise, $t_i(L) = 2^{\lceil \log_2^{ti} \rceil}$ and $t_i(H) = +\infty$. The mode change duration C is set as the maximum number of hops between any two nodes of the network. If one channel and one transmitter of each node are reserved to serve the mode change, the change command can be broadcast to all the nodes in the duration C. The fixed priority assignment follows the two classical algorithms [28]: (1) Deadline Monotonic (DM), in which the flow with the shorter deadline is assigned the higher priority; (2) Proportional Deadline monotonic (PD), in which the flow with the shorter subdeadline is assigned the higher priority. *Subdeadline* is defined for its deadline divided by the total number of its transmissions.

The mode change can occur at any time slot. Hence, the simulation should list all cases. However, for the complex state space, the execution time of simulations is unacceptable. Therefore, if the execution time exceeds 30 minutes, the simulation is suspended and the maximum delay is chosen as the worst case end-to-end delay. We use *pessimism ratio* (the proportion of our analyzed delay to the maximum delay observed in simulations) and *acceptance ratio* (the percentage of flow sets that are schedulable) as the performance metrics.

Figure 2.2 plots the pessimism ratios with different numbers of nodes. We set that $m = 12$ and $u = 1$. In order to make test cases simulated in an acceptable time, the number of nodes is only up to 110. From the figures, we can see that the 75th percentile of the pessimism ratios is less than 2.1 and 2.2 for DM and PD, respectively. In [1], the result of the state-of-the-art analysis EDA for single-criticality networks is 1.5 and 1.6, respectively. Compared with them, our analysis only introduces a small degree of pessimism, even though the mode change increases the complexity of the end-to-end delay analysis. Therefore, our analysis is highly effective.

In order to evaluate the performance of our analysis method for the larger scale networks in an acceptable time, we set $m = 6$ and $u = 1$. Figure 2.3 shows the boxplots of the pessimism ratios under varying network sizes. From the evaluation results, we know that our analysis method is stable under different network sizes. Comparing with Fig. 2.2, Fig. 2.3 is more pessimistic. The less number of channels introduces more contentions, and the delay analysis is to consider the worst case scenario. Thus, all of additional contentions are considered in the delay analysis, but not all of them appear in simulations. Therefore, the analysis with fewer channels is more pessimistic.

We compare the acceptance ratios of our analysis and simulations, and the utilization u is increased to 3.2. Figure 2.4 shows the comparison, in which AMC is our Analysis for Mixed-Criticality networks and SIM is the result of simulations. We observe that our results are close to those of simulations. Therefore, our analysis method can be used to verify whether flows can meet their deadlines or not before implementing the real system.

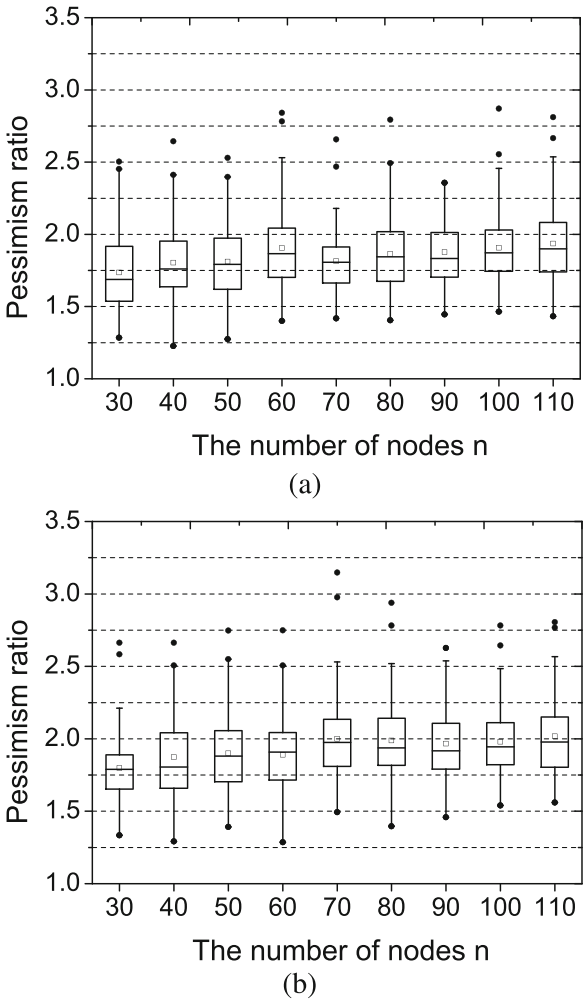

Fig. 2.2 Pessimism ratio under varying network sizes with $m = 12$. (**a**) The priority assignment policy DM. (**b**) The priority assignment policy PD

Comparing Fig. 2.4a and b, we observe that the acceptance ratio of the policy PD is less than that of the policy DM. It is because that, compared with the policy DM, the policy PD introduces more interferences to the flows with short paths, which leads to a longer delay. Similarly, all the interferences are considered in the delay analysis, but not all of them appear in simulations. Therefore, in Figs. 2.2 and 2.3, the result of PD is more pessimistic than that of DM.

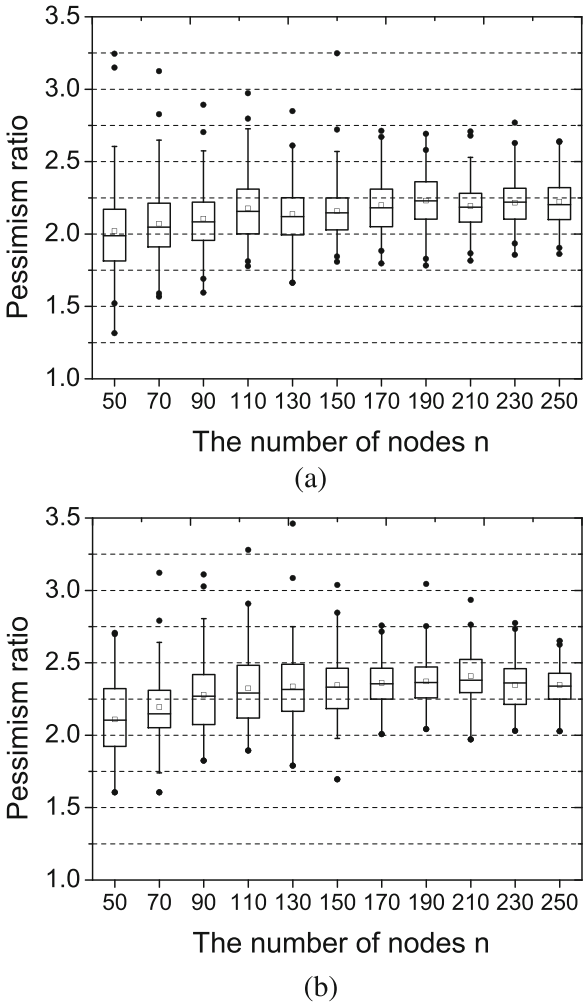

Fig. 2.3 Pessimism ratio under varying network sizes with $m = 6$. (**a**) The priority assignment policy DM. (**b**) The priority assignment policy PD

2.4.2 Real Testbed

We implement a real testbed that contains three types of physical devices: the gateway device, routing devices and field devices. The gateway device manages the network and adopts a low power SoC (System of Chip) AT91RM9200 and a CC2420 transceiver chip. The routing device is implemented on an MSP430 and a CC2420. The field device is equipped with a temperature and humidity sensor SHT15 besides an MSP430 and a CC2420. Our testbed supports the IEEE 802.15.4 protocol, which is the physical and MAC (Medium Access Control) layers of WirelessHART networks, and an improved WirelessHART network according to our requirements. The improved WirelessHART implements the specific requirements

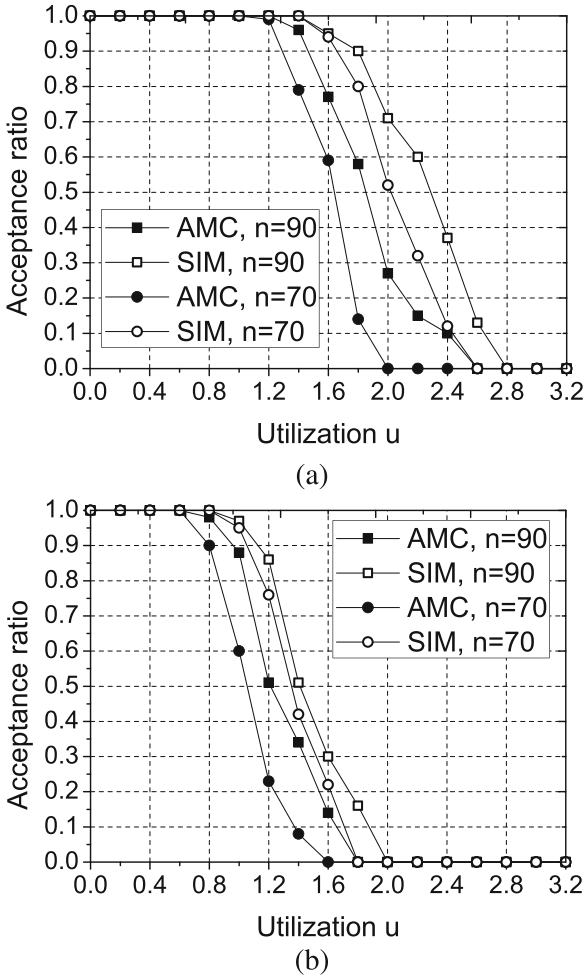

Fig. 2.4 Acceptance ratio under varying utilizations with $m = 6$. (**a**) The priority assignment policy DM. (**b**) The priority assignment policy PD

in the application layer, and it is compatible with the original WirelessHART. Channel 23 is used to broadcast mode change messages and configuration messages. Six schedulable channels are 15–20. Additionally, six devices are configured as sniffers to monitor packets transmitted on the six channels. Then the sniffed packet with a timestamp is sent to a PC via an 8-port RS-232 PCI Express serial board.

Figure 2.5 shows our testbed. The network is deployed in a building. For each parameter configuration, 100 test cases are implemented. The generation of configurations is the same as in simulations. The configuration message is sent to devices via the gateway. Figure 2.6 shows pessimism ratios in a certain scope under different parameters. The point of the pessimism ratio 1.4 reports the number of test cases, whose pessimism ratios are between 1.4 and 1.6. When the utilization is set as 1 and the number of channels is 12, compared with PD and DM, our average

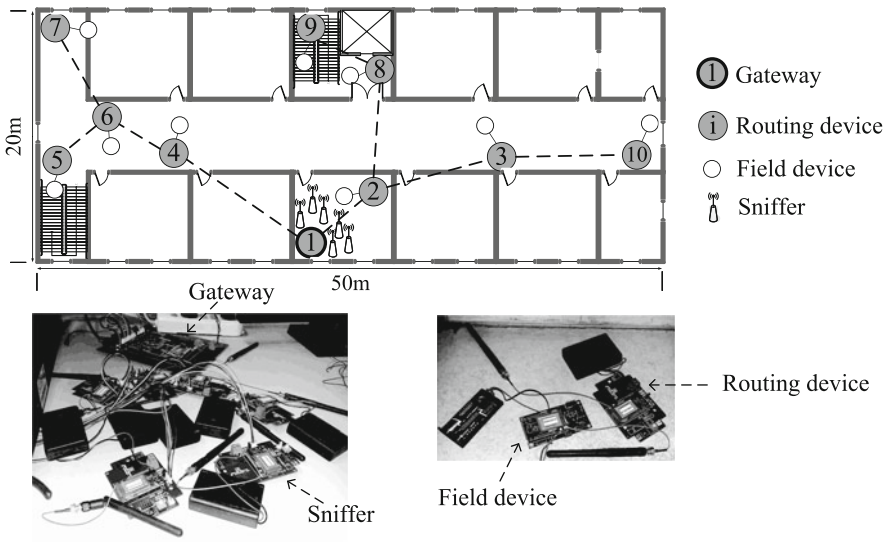

Fig. 2.5 Our testbed

pessimism ratio of 100 test cases is about 2.5 and 2.4, respectively. The result is more pessimistic than the simulations. It is because that real cases only cover a little state space. The delay observed in the real testbed is not the worst case delay, while our analysis focuses on the worst case. Therefore, for the end-to-end delay, our analysis method is more reliable than real tests.

2.5 Summary

Multiple criticality levels co-exist in real-life wireless networks. However, previous works only focus on the single-criticality network. We present an end-to-end delay analysis method for fixed priority scheduling in mixed-criticality WirelessHART networks, which can be used to determine whether all flows can be delivered to their destinations within their deadlines. In evaluations, we compare our analysis results with simulations and a testbed. The results show that the pessimism of our analysis is acceptable and reliable.

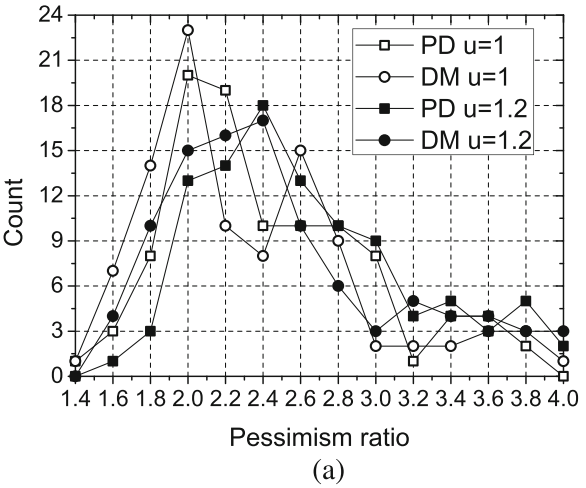

Fig. 2.6 Pessimism ratio on the testbed. (**a**) The number of channel $m = 12$. (**b**) The number of channel $m = 6$

References

1. Saifullah A, Xu Y, Lu CY, Chen YX (2011) End-to-end delay analysis for fixed priority scheduling in WirelessHART networks. In: Real-time and embedded technology and applications symposium (RTAS). IEEE, pp 13–22
2. Saifullah A, Xu Y, Lu CY, Chen YX (2010) Real-time scheduling for WirelessHART networks. In: Real-time systems symposium (RTSS). IEEE, pp 150–159
3. Saifullah A, Xu Y, Lu CY, Chen YX (2011) Priority assignment for real-time flows in WirelessHART networks. In: Euromicro conference on real-time systems (ECRTS). IEEE, pp 35–44
4. Soldati P, Zhang HB, Johansson M (2009) Deadline-constrained transmission scheduling and data evacuation in WirelessHART networks. In: The European control conference. IEEE, pp 1–7

5. Vestal S (2007) Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In: The IEEE real-time system symposium (RTSS). IEEE, pp 239–243
6. Niz D, Lakshmanan K, Rajkumar R (2009) On the scheduling of mixed criticality real-time task sets. In: The IEEE real time systems symposium (RTSS). IEEE, pp 291–300
7. Baruah S, Li HH, Stougie L (2010) Towards the design of certifiable mixed-criticality systems. In: The IEEE real-time and embedded technology and applications symposium. IEEE, pp 555–556
8. Baruah S, Bonifaci V, D'Angelo G, Li HH, Marchietti-Spaccamela A, Megow N, Stougie L (2012) Scheduling real-time mixed-criticality jobs. Trans Comput 61(8): 1140–1152
9. Li HH, Baruah S (2012) Global mixed-criticality scheduling on multiprocessors. In: The Euromicro conference on real-time systems (ECRTS). IEEE, pp 166–175
10. Baruah SK, Bonifaci V, D'Angelo G, Marchietti-Spaccamela A, Van Der Ster S, Stougie L (2011) Mixed-criticality scheduling of sporadic task systems. In: The 2011 European conference on algorithms. Springer, pp 555–566
11. Guan N, Ekberg P, Stigge M, Wang Y (2011) Effective and efficient scheduling of certifiable mixed-criticality sporadic task systems. In: The IEEE real-time systems symposium (RTSS). IEEE, pp 13–23
12. Huang HM, Gill C, Lu CY (2014) Implementation and evaluation of mixed criticality scheduling approaches for sporadic tasks. Trans Embed Comput Syst 13(4): 1–25
13. Tobuschat S, Axer P, Ernst R, Diemer J (2013) IDAMC: a NoC for mixed criticality systems. In: The IEEE international conference on embedded and real-time computing systems and applications. IEEE, pp 149–156
14. Diemer J, Ernst R (2010) Back suction: service guarantees for latency-sensitive on-chip networks. In: The ACM/IEEE international symposium on networks-on-chip. IEEE, pp 155–162
15. Audsley N (2013) Memory architectures for NoC-based real-time mixed criticality systems. In: The 1st workshop on mixed criticality systems. IEEE, pp 37–42
16. Burns A, Davis RI (2013) Mixed criticality on controller area network. In: The Euromicro conference on real-time systems (ECRTS). IEEE, pp 125–134
17. Herber C, Richter A, Rauchfuss H, Herkersdorf A (2013) Spatial and temporal isolation of virtual CAN controllers. In: The IEEE international conference on embedded and real-time computing systems and applications. IEEE, pp 1–7
18. Tamas-Selicean D, Pop P, Steiner W (2011) Synthesis of communication schedulers for TTEthernet-based mixed-criticality systems. In: The international conference on hardware/software codesign and system synthesis. IEEE, pp 473–482
19. Suethanuwong E (2012) Scheduling time-triggered traffic in TTEthernet systems. In: The conference on emerging technologies and factory automation. IEEE, pp 1–4
20. Steiner W (2011) Synthesis of static communication schedules for mixed-criticality systems. In: The 14th IEEE international symposium on object/component/service-oriented real-time distributed computing workshop. IEEE, pp 11–18
21. Addisu A, George L, Sciandra V, Agueh M (2013) Mixed criticality scheduling applied to JPEG2000 video streaming over wireless multimedia sensor networks. In: The 1st workshop on mixed criticality systems. IEEE, pp 1–6
22. Baruah S, Bonifaci V, D'Angelo G, Li HH, Marchietti-Spaccamela A, Megow N, Stougie L (2012) Scheduling real-time mixed-criticality jobs. Trans Electron Comput 61(8):1140–1152
23. Li HH, Baruah S (2012) Global mixed-criticality scheduling on multiprocessors. In: The 24th Euromicro conference on real-time systems. IEEE, pp 166–175
24. Guan N, Ekberg P, Stigge M, Wang Y (2011) Effective and efficient scheduling of certifiable mixed-criticality sporadic task systems. In: The 32nd IEEE real-time systems symposium. IEEE, pp 13–23
25. Joseph M, Pandya P (1986) Finding response times in a real-time system. Comput J 29(5):390–395

26. Camilo T, Silva JS, Rodrigues A, Boavida F (2007) Gensen: a topology generator for real wireless sensor networks deployment. In: The international conference on software technologies for embedded and ubiquitous systems. Springer, pp 436–445
27. Bini E, Buttazzo CC (2005) Measuring the performance of schedulability tests. Real-Time Syst 30(1):129–154
28. Liu J (2000) Real-time systems. Prentice Hall, Upper Saddle River