End-to-End Communication Delay Analysis in Industrial Wireless Networks

Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen

Abstract—WirelessHART is a new standard specifically designed for real-time and reliable communication between sensor and actuator devices for industrial process monitoring and control applications. End-to-end communication delay analysis for WirelessHART networks is required to determine the schedulability of real-time data flows from sensors to actuators for the purpose of acceptance test or workload adjustment in response to network dynamics. In this paper, we consider a network model based on WirelessHART, and map the scheduling of real-time periodic data flows in the network to real-time multiprocessor scheduling. We then exploit the response time analysis for multiprocessor scheduling and propose a novel method for the delay analysis that establishes an upper bound of the end-to-end communication delay of each real-time flow in the network. Simulation studies based on both random topologies and real network topologies of a 74-node physical wireless sensor network testbed demonstrate that our analysis provides safe and reasonably tight upper bounds of the end-to-end delays of real-time flows, and hence enables effective schedulability tests for WirelessHART networks.

Index Terms—Wireless sensor networks, scheduling, Real-time and embedded systems.

1 Introduction

Wireless Sensor-Actuator Networks (WSANs) are an emerging communication infrastructure for monitoring and control applications in process industries. In a feedback control system where the networked control loops are closed through a WSAN, the sensor devices periodically send data to the controllers, and the control input data are then delivered to the actuators through the network. To maintain the stability and control performance, industrial monitoring and control applications impose stringent end-to-end delay requirements on data communication between sensors and actuators [1]. Real-time communication is critical for process monitoring and control since missing a deadline may lead to production inefficiency, equipment destruction, and severe economic and/or environmental threats. For example, in oil refineries, spilling of oil tanks is avoided by monitoring and control of level measurement in real-time.

WirelessHART [2] has been designed as an open WSAN standard to address the challenges in industrial monitoring and control. To meet the stringent real-time and reliability requirements in harsh and unfriendly industrial environments, the standard features a centralized network management architecture, multi-channel Time Division Multiple Access (TDMA), redundant routes, and channel hopping [1]. These unique characteristics introduce unique challenges in end-to-end delay analysis for process monitoring and control in WirelessHART networks.

• *The authors are with the Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130. E-mail: {saifullaha, yx2, lu, chen}@cse.wustl.edu*

In this paper, we address the problem of end-to-end delay analysis for periodic real-time flows from sensors to actuators in a network that is modeled based on WirelessHART (simply named WirelessHART network throughout the paper). We derive upper bounds of the end-to-end delays of the flows under fixed priority scheduling where the transmissions associated with each flow are scheduled based on the fixed priority of the flow. Fixed priority scheduling is a common class of real-time scheduling policies in practice.

Analytical delay bounds can be used to test, both at design time and for online admission control, whether a set of real-time flows can meet all their deadlines. Compared to extensive testing and simulations, an end-to-end delay analysis is highly desirable in process monitoring and control applications that require real-time performance guarantees. It can also be used for adjusting the workload in response to network dynamics. For example, when a channel is blacklisted or some routes are recalculated, the delay analysis can be used to promptly decide whether some flow has to be removed or some rate has to be updated.

A key insight underlying our analysis is to map the real-time transmission scheduling in WirelessHART networks to real-time multiprocessor scheduling. This mapping allows us to provide a delay analysis of the real-time flows in WirelessHART networks by taking an analysis approach similar to that for multiprocessor scheduling. By incorporating the unique characteristics of WirelessHART networks into the state-of-the-art worst case response time analysis for multiprocessor scheduling [3], we propose a novel end-to-end delay analysis for fixed priority transmission scheduling in WirelessHART networks. The proposed analysis calculates a safe and tight upper bound of the

end-to-end delay of every real-time periodic data flow in pseudo polynomial time. Furthermore, we extend the pseudo polynomial time analysis to a polynomial time method that provides slightly looser bounds but can calculate the bounds more quickly.

We evaluate our analysis through simulations based on both random network topologies and the real network topologies of a wireless sensor network testbed consisting of 74 TelosB motes. The simulation results show that our delay bounds are safe and reasonably tight. The proposed analysis, hence, enables an effective schedulability test for WirelessHART networks.

In the rest of the paper, Section 2 reviews related works. Section 3 presents the network model. Section 4 defines the scheduling problem. Section 5 presents the end-to-end delay analysis. Section 6 extends our delay analysis to a polynomial time method. Section 7 presents evaluation results. Section 8 concludes the paper.

2 Related Work

Real-time transmission scheduling in wireless networks has been widely studied in previous works [4]. However, very few of those are applicable to WirelessHART networks. Scheduling based on CSMA/CA protocols has been studied in [5]–[10]. In contrast, WirelessHART adopts a TDMA-based protocol to achieve predictable latency bounds. Although TDMA-based scheduling has been studied in [11]–[13], these works do not focus on schedulability or delay analysis. The real-time schedulability analysis has been studied in [14], [15] for single-hop wireless network where communications happen between an access point and a set of clients on a single active channel at a time. In contrast, our work focuses on multi-channel and multi-hop wireless mesh network. The authors in [16] propose a schedulability analysis for multi-hop wireless sensor networks (WSNs) by upper bounding the real-time capacity of the network. However, in their model, taking the advantage of TDMA or frequency division has no effect. The schedulability analysis for WSNs has also been pursued in [17], [18]. But these are designed only for data collection through a routing tree using single channel, and do not address multi-channel communication or multi-path routing supported by WirelessHART.

For WirelessHART networks, routing [19], schedule modeling [20], real-time transmission scheduling [21], [22], and rate selection [23] have been studied recently. Our work in [21] proves the NP-hardness of the optimal real-time transmission scheduling in a WirelessHART network. It also presents an optimal scheduling algorithm based on branch-and-bound and a heuristic policy. Neither algorithm employed fixed priority. Moreover, no efficient worst-case delay analysis was provided for either algorithm. We studied priority assignment in [22] and rate selection methods in [23] for real-time flows in WirelessHART networks, both of which leverages worst-case delay analysis which is the focus of this paper. To summarize, none of our previous works addresses worst-case delay analysis. In contrast, this paper presents an end-to-end delay analysis that is suitable for any fixed priority scheduling policy. An efficient delay analysis is particularly useful for online admission control and adaptation (e.g., when network route or topology changes) so that the network manager is able to quickly reassess the schedulability of the flows.

3 Network Model

We consider a network model inspired by WirelessHART. A WirelessHART network consisting of a set of field devices and one gateway. These devices form a mesh network that can be modeled as a graph $G = (V, E)$, where V is the set of nodes (i.e., field devices and the gateway), and E is the set of communication links between the nodes. A *field device* is either a sensor node, an actuator or both, and is usually connected to process or plant equipment. The *gateway* connects the WirelessHART network to the plant automation system, and provides the host system with access to the network devices. For any link $e = (u, v)$ in E, devices $u \in V$ and $v \in V$ can communicate with each other. For a transmission, denoted by $\vec{uv}$, that happens along link (u, v), device u is designated as the *sender* and device v the *receiver*. All network devices (i.e., field devices and the gateway) are able to send, receive, and route packets.

For process control, the controllers are installed in control hosts connected to the gateway through the plant automation network. The sensor devices deliver their sensor data to the gateway. The control messages from the gateway are then delivered to the actuators through the wireless mesh network. The unique features that make WirelessHART particularly suitable for industrial process control are as follows [1], [2].

Centralized Management. A WirelessHART network is managed by a *centralized network manager* installed in the gateway. The network manager collects the network topology information, and determines the routes. It then creates the schedule of transmissions, and distributes the schedules among the devices. The centralized management limits the number of nodes under a gateway [24] that makes the centralized management practical and desirable, and enhances the reliability and real-time performance.

Time Division Multiple Access (TDMA). In WirelessHART networks, time is synchronized, and communication is TDMA-based. A time slot is 10ms long, and allows exactly one transmission and its associated acknowledgement between a device pair. For transmission between a receiver and its senders, a time slot can be either dedicated or shared. In a *dedicated time slot*, only one sender is allowed to transmit to the receiver. In a *shared slot*, more than one sender can attempt to transmit to the same receiver. Since

collisions may occur within a shared slot, a transmission within a shared slot may be successful only when other senders do not need to send.

Route Diversity. To enhance the end-to-end reliability, both upstream and downstream communications are scheduled based on graph routing. A *routing graph* between two devices is a directed list of paths that connect two devices, thereby providing redundant paths between them. On one path from the source to the destination, the scheduler allocates a dedicated slot for each en-route device starting from the source, followed by allocating a second dedicated slot on the same path to handle a retransmission. Then, to offset failure of both transmissions along a primary link, the scheduler again allocates a third shared slot on a separate path to handle another retry.

Spectrum Diversity. Spectrum diversity gives the network access to all 16 channels defined in IEEE 802.15.4 and allows per time slot channel hopping in order to avoid jamming and mitigate interference from coexisting wireless systems. Besides, any channel that suffers from persistent external interference is *blacklisted* and not used. Due to difficulty in detecting interference between nodes and the variability of interference patterns, WirelessHART networks typically avoid spatial reuse of a channel within the same time slot. Thus all transmissions in a time slot use different channels. This strategy effectively avoids transmission failure due to interference between concurrent transmissions, thereby providing a high degree of reliability for critical process monitoring and control applications. Henceforth we assume there is no spatial reuse of channels in this work.

Each device is equipped with a half-duplex omnidirectional radio transceiver and, hence, cannot both transmit and receive in the same time slot. In addition, two transmissions that have the same intended receiver interfere each other. Therefore, two transmissions $\vec{uv}$ and $\vec{ab}$ are *conflicting* and, hence, are not scheduled in the same slot if $(u = a) \vee (u = b) \vee (v = a) \vee (v = b)$. Since different nodes experience different degrees of conflict during communication, transmission conflicts play a major role in analyzing the end-to-end delays in the network.

Simplifying assumptions. As the first step toward a real-time schedulability analysis for WirelessHART networks, we make some simplifying assumptions on routing. Instead of a general graph routing, we assume a multi-path routing between every source and destination pair. To simplify the analysis further, we also assume that the packets are scheduled using dedicated slots only. The simplifying assumption facilitates the development of the first end-to-end delay analysis based on real-time scheduling theory. While our analysis leverages these simplified assumptions, it provides fundamental building blocks for the analysis based on general graph routing.

4 End-to-End Scheduling Problem

We consider a WirelessHART network $G = (V, E)$ with a set of end-to-end flows denoted by $\mathbb{F}$. Each flow $\mathbb{F}_j \in \mathbb{F}$ is characterized by a period P_j, a deadline D_j where $D_j \leq P_j$, and a set of one or more routes Φ_j. Each $\phi \in \Phi_j$ is a route from a network device $Source_j \in V$, called the *source of* $\mathbb{F}_j$, to another network device $Destination_j \in V$, called the *destination of* $\mathbb{F}_j$, through the gateway. Each flow $\mathbb{F}_j$ periodically generates a packet at period P_j which originates at $Source_j$ and has to be delivered to $Destination_j$ within deadline D_j. For flow $\mathbb{F}_j$, if a packet generated at slot r is delivered to $Destination_j$ at slot f through a route $\phi \in \Phi_j$, its *end-to-end delay* through ϕ is defined as $L_j(\phi) = f - r + 1$.

A flow $\mathbb{F}_j$ may need to deliver its packet through more than one route in Φ_j. If the delivery through a route fails, the packet can still be delivered through another route in Φ_j. Therefore, in a TDMA schedule, for a flow $\mathbb{F}_j$, time slots must be reserved for transmissions through each route in Φ_j for redundancy. Hence, for end-to-end delay analysis purpose, through each of its routes flow $\mathbb{F}_j$ is treated as an individual flow F_i with deadline and period equal to $\mathbb{F}_j$'s deadline and period, respectively. That is, $\mathbb{F}_j$ is now considered $|\Phi_j|$ individual flows, each with a single route. Therefore, from now onward the term 'flow' will refer to an individual flow through a route. We denote this set of flows by $F = \{F_1, F_2, \cdots, F_N\}$. Thus, associated with each flow $F_i, 1 \leq i \leq N$, are a period P_i, a deadline D_i, a source node $Source_i$, a destination node $Destination_i$, and a route ϕ_i from $Source_i$ to $Destination_i$. For each flow F_i, if every transmission is repeated χ times to handle retransmission on a single route, then the number of transmissions required to deliver a packet from $Source_i$ to $Destination_i$ through its route ϕ_i is $C_i = length(\phi_i) * \chi$, where $length(\phi_i)$ is the number of links on ϕ_i. Thus, C_i is the number of time slots required by flow F_i.

Fixed priority scheduling. For fixed priority scheduling, each flow F_i has a fixed priority. We assume that all flows are ordered by priorities. Flow F_i has higher priority than flow F_j if and only if $i < j$. We use $hp(F_i)$ to denote the set of flows whose priorities are higher than that of flow F_i. That is, $hp(F_i) = \{F_1, F_2, \cdots, F_{i-1}\}$. In practice, priorities may be assigned based on deadlines, rates, or the criticality of the real-time flows. Priority assignment policies are not the focus of this paper, and our delay analysis can be applied to any fixed priority assignment. Under a *fixed priority scheduling policy*, the transmissions of the flows are scheduled in the following way. Starting from the highest priority flow F_1, the following procedure is repeated for every flow F_i in decreasing order of priority. For current priority flow F_i, the network manager schedules its transmissions along its route (starting from the source) on earliest available time

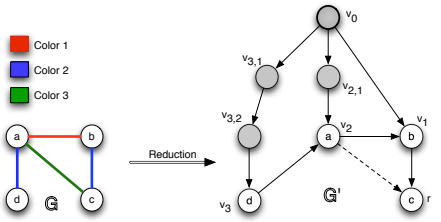


Fig. 1. Reduction from edge-coloring

slots and on available channels. A time slot is *available* if no conflicting transmission is already scheduled in that slot. In a WirelessHART network, the complete schedule is divided into superframes. A *superframe* represents transmissions in a series of time slots that repeat infinitely and represent the communication pattern of a group of devices.

Problem formulation. Transmissions are scheduled using m channels. The set of flows F is called *schedulable* under a scheduling algorithm $\mathbb{A}$, if $\mathbb{A}$ is able to schedule all transmissions in m channels such that no deadline is missed, i.e., $L_i \leq D_i$, $\forall F_i \in F$, with L_i being the end-to-end delay of F_i. For $\mathbb{A}$, a schedulability test $\mathbb{S}$ is *sufficient* if any set of flows deemed schedulable by $\mathbb{S}$ is indeed schedulable by $\mathbb{A}$. To determine schedulability of a set of flows, it is sufficient to show that, for every flow, an upper bound of its worst case end-to-end delay is no greater than its deadline. Thus, given the flows F and a fixed priority algorithm $\mathbb{A}$, our objective is to decide schedulability of F based on an end-to-end delay analysis. Theorem 1 proves that an exact schedulability analysis (i.e., both sufficient and necessary) problem for fixed priority scheduling in a WirelessHART network is NP-hard by proving that its decision version is NP-complete. Note that this proof is based on the reduction used in Theorem 1 of [21] where the NP-completeness of deciding the schedulability of a set of periodic real-time flows in a WirelessHART network was proven under dynamic scheduling.

Theorem 1: Given a real-time scheduling problem for a WirelessHART network under a fixed priority scheduling policy, it is NP-complete to decide whether the problem is schedulable or not.

Proof: The problem belongs to NP as, for any instance of the fixed priority real-time scheduling problem for a WirelessHART network with N flows, we can verify in $O(N)$ time whether all the flows meet their deadlines. To prove NP-hardness, we reduce an arbitrary instance $< \mathbb{G}, k >$ of the *graph edge-coloring* problem to an instance $\mathbb{S}$ of the fixed priority real-time scheduling for a WirelessHART network and show that graph $\mathbb{G}$ is k edge-colorable if and only if $\mathbb{S}$ is schedulable (Figure 1).

The reduction is as follows. Let $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ has n nodes. We create a depth-first search tree of $\mathbb{G}$ rooted at an arbitrary node $r \in \mathbb{V}$. For every $u \in \mathbb{V} - \{r\}$, a tree edge is directed from u to its *parent*; and zero or more ancestors connected by a non-tree edge directed from u are its *virtual parents*. Every node in $\mathbb{V} - \{r\}$ is given a unique label v_i, where $1 \leq i \leq n-1$. Create a node v_0. For every node v_i, $1 \leq i \leq n-1$, add $i-1$ additional nodes $v_{i,1}, v_{i,2}, \cdots, v_{i,i-1}$ and connect v_0 to v_i through these nodes (i.e., create $v_0 - v_{i,1} - v_{i,2} - \cdots - v_{i,i-1} - v_i$ path). Now, following is an instance $\mathbb{S}$ of the fixed priority real-time scheduling for a WirelessHART network. The reduced graph $\mathbb{G}' = (\mathbb{V}', \mathbb{E}')$ is a network with v_0 being the gateway. The parent and the virtual parents of every node v_i, $1 \leq i \leq n-1$, are the destination nodes, and v_0 is a source node. For every v_i, $1 \leq i \leq n-1$, a flow F_i periodically generates a packet starting at $(n-i)$-th slot at v_0 and follows the route $v_0 - v_{i,1} - v_{i,2} - \cdots - v_i$ and is, then, forwarded by v_i to its parent and every virtual parent. Each flow F_i is assigned fixed priority i. For simplicity, we consider only the first packet of every flow F_i. For F_i, the release time and the absolute deadline of this packet are $n-i$ and $n-1+k$, respectively. All flows have the same period $\geq n-1+k$. The number of channels is $n-1$. This reduction runs in $O(n^2)$ time.

Let $\mathbb{G}$ is edge-colorable using k colors. Let Q be the set of all last one-hop transmissions in $\mathbb{G}'$. These transmissions involve edges $\mathbb{E} \subset \mathbb{E}'$, one transmission per edge. Using all $n-1$ channels, we can complete all transmissions in $\mathbb{G}'$ except those in Q in first $n-1$ slots, when the transmissions of each flow are scheduled based on its (fixed) priority. Since the transmissions along the edges having the same color can be scheduled on the same slot, all transmissions in Q can be scheduled in next k slots. Hence, all packets meet the deadline. Now, let $\mathbb{S}$ is schedulable based on the fixed priorities of the flows. If all channels are used in scheduling, then all but the transmissions in Q are completed in first $n-1$ slots in the fixed priority scheduling. Hence, all transmissions in Q are schedulable using next k slots. For transmissions that happen on the same slot, the corresponding edges can be given the same color. Hence, graph $\mathbb{G}$ is k edge-colorable. If $\mathbb{A}$ does not use all channels, then no transmission in Q can happen in first $n-1$ slots. Let there are t slots starting from the earliest slot at which some transmission in Q can be scheduled to the latest slot by which all transmissions in Q must be scheduled. Since all packets meet the deadline, $t \leq k$. The value of t is the smallest when we can schedule all non-conflicting transmissions in Q on the same slot. That is, the smallest value of t is the edge chromatic number χ of $\mathbb{G}$. Thus, $\chi \leq t \leq k$. Since $\mathbb{G}$ is χ edge-colorable, it is k edge-colorable also. □

Uses of a sufficient analysis. Since an exact analysis is NP-hard, we pursue an end-to-end delay analysis which serves as a sufficient condition for schedulability. For real-time flows in industrial process control applications that require hard real-time guarantees, this analysis can thus be used for online admission control and to adjust workload in response to network

dynamics. For example, when a channel is blacklisted or some routes are recalculated, the network manager can execute our sufficient analysis to verify whether the current set of flows remain schedulable. If the analysis cannot guarantee the schedulability of all the flows, the network manager may remove a subset of the flows (e.g., based on criticality) or reduce the data rates of some of the flows so that the new set of flows becomes schedulable under our analysis.

5 End-to-end Delay Analysis

In this section, we present an end-to-end delay analysis for the real-time flows in a WirelessHART network. An efficient end-to-end delay analysis is particularly useful for online admission control and adaptation to network dynamics so that the network manager is able to quickly reassess the schedulability of the flows (e.g., when network route or topology changes, or some channel is blacklisted). In analyzing the end-to-end delays, we observe two reasons that contribute to the delay of a flow. A lower priority flow can be delayed by higher priority flows (a) due to *channel contention* (when all channels are assigned to transmissions of higher priority flows in a time slot), and (b) due to *transmission conflicts* (when a transmission of the flow and a transmission of a higher priority flow involve a common node). At first, we analyze each delay separately. We, then, incorporate both types of delays into our analysis and end up with an upper bound of the end-to-end delay for every flow. A holistic approach that can analyze two types of delays combining into a single step might lead to tighter delay bound, but we opt for the divide-and-conquer approach to simplify the theoretical analysis of the safety of the bound. If every transmission is repeated χ times to handle retransmission on a single route, then every time slot is simply multiplied by χ in delay calculation. For simplicity of presentation we use retransmission parameter $\chi = 1$.

5.1 Delay due to Channel Contention

5.1.1 Observations between Transmission Scheduling and Multiprocessor CPU Scheduling

A key insight in this work is that we can map the multi-channel fixed priority transmission scheduling problem for WirelessHART networks to the fixed priority real-time CPU scheduling on a global multiprocessor platform. Towards this direction, we make the following observations between these two domains.

In a WirelessHART network, each channel can accommodate one transmission in a time slot across the entire network. Thus, a flow executing for one time unit on a CPU of a multiprocessor system is equivalent to a packet transmission on a channel which takes exactly one time slot in a WirelessHART network. That one flow cannot be scheduled on different processors at the same time is similar to the fact that one flow cannot be scheduled on different channels at the same time. In addition, flows executing on multiprocessor platform are considered independent while the flows being scheduled in a WirelessHART network are also independent. Again, execution of flows on a global multiprocessor platform is equivalent to switching of a packet to different channels at different time slots due to channel hopping. Finally, completing the execution of a flow on a CPU is equivalent to completing all transmissions of a packet from the source to the destination of the flow.

Thus, in absence of conflicts, the worst case response time of a flow in a multiprocessor platform is equivalent to the upper bound of its end-to-end delay in a WirelessHART network. Therefore, to analyze the delay due to channel contention, we can map the transmission scheduling in a WirelessHART network to global multiprocessor CPU scheduling.

5.1.2 Mapping to Multiprocessor CPU Scheduling

Based on the observations discussed above, the mapping from multi-channel transmission scheduling in a WirelessHART network to multiprocessor CPU scheduling is as follows.

- Each channel is mapped to a *processor*. Thus, m channels correspond to m processors.
- Each flow $F_i \in F$, is mapped to a *task* that executes on multiprocessor with period P_i, deadline D_i, execution time C_i, and priority equal to the priority of flow F_i.

While the proposed mapping allows us to potentially leverage the rich body of literature on real-time CPU scheduling, the end-to-end delay analysis for WirelessHART networks remains an open problem. An important observation is that we must consider transmission conflicts in the delay analysis. Note that transmission conflict is a distinct feature of wireless networks that does not exist in traditional real-time CPU scheduling problems. A key contribution of our work, therefore, is to incorporate the delays caused by transmission conflicts into the end-to-end delay analysis. By incorporating the delay due to these conflicts into the multiprocessor real-time schedulability analysis, we establish an upper bound of the end-to-end delay of every flow in a WirelessHART network.

In the proposed end-to-end delay analysis, we first analyze the delay due to channel contention between the flows. Whenever there is a channel contention between two flows, the lower priority flow is delayed by the higher priority one. Based on the above mapping, the analysis for the worst case delay that a lower priority flow experiences from the higher priority flows due to channel contention in a WirelessHART network is similar to that when the flows are scheduled on a multiprocessor platform. Therefore, instead of establishing a completely new analysis for the delay due to channel contention, the proposed mapping allows us to exploit the results of the state-of-the-art response time analysis for multiprocessor scheduling [3].

5.1.3 Response Time Analysis for Multiprocessor

To make our paper self-contained, here we present the results of the state-of-the-art response time analysis for multiprocessor scheduling proposed by Guan et al. [3]. Assuming that the flows are executed on a multiprocessor platform, they have observed that a flow experiences the worst case delay when the earliest time instant after which all processors are occupied by the higher priority flows occurs just before its release time. Therefore, for flow F_k, a *level-k busy period* is defined as the maximum continuous time interval during which all processors are occupied by flows of priority higher than or equal to F_k's priority, until F_k finishes its active instance. We use the notation BP(k,t) to denote a level-k **b**usy **p**eriod of t slots. The delay that some higher priority flow $F_i \in hp(F_k)$ will cause to F_k depends on the workload of all instances of F_i during a BP(k,t). Flow F_i has *carry-in* workload in a BP(k,t), if it has one instance with release time earlier than the BP(k,t) and deadline in the BP(k,t). When F_i has no carry-in, an upper bound $W_k^{\text{nc}}(F_i,t)$ of its workload in a BP(k,t), and an upper bound $I_k^{\text{nc}}(F_i,t)$ of the delay it can cause to F_k are as follows.

$$W_k^{\text{nc}}(F_i,t) = \left\lfloor \frac{t}{P_i} \right\rfloor . \ C_i + \min(t \bmod P_i,\ C_i) \quad (1)$$

$$I_k^{\text{nc}}(F_i,t) = \min\Big(W_k^{\text{nc}}(F_i,t),\ t - C_k + 1\Big) \quad (2)$$

When F_i has carry-in μ_i and the worst case response time R_i, an upper bound $W_k^{\text{ci}}(F_i,t)$ of its workload in a BP(k,t), and an upper bound $I_k^{\text{ci}}(F_i,t)$ of the delay that it can cause to F_k are as follows.

$$W_k^{\text{ci}}(F_i,t) = \left\lfloor \frac{\max(t - C_i, 0)}{P_i} \right\rfloor . \ C_i + C_i + \mu_i \quad (3)$$

$$I_k^{\text{ci}}(F_i,t) = \min\Big(W_k^{\text{ci}}(F_i,t), t - C_k + 1\Big) \quad (4)$$

where

$$\mu_i = \min\Big(\max\Big(\lambda - (P_i - R_i), 0\Big), C_i - 1\Big)$$

$$\lambda = \max(t - C_i, 0) \bmod P_i$$

With the observation that at most $m-1$ higher priority flows can have carry-in, an upper bound $\Omega_k(t)$ of the total delay caused by all higher priority flows to an instance of F_k during a BP(k,t) is

$$\Omega_k(t) = X_k(t) + \sum_{F_i \in hp(F_k)} I_k^{\text{nc}}(F_i,t) \quad (5)$$

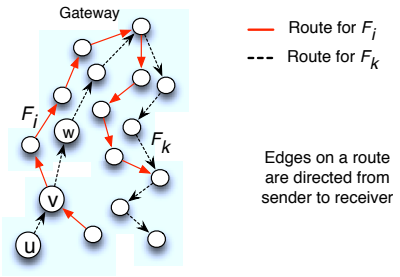
with $X_k(t)$ being the sum of the $\min(|hp(F_k)|, m-1)$ largest values of the differences $I_k^{\text{ci}}(F_i,t) - I_k^{\text{nc}}(F_i,t)$ among all $F_i \in hp(F_k)$.

5.2 Delay due to Transmission Conflicts

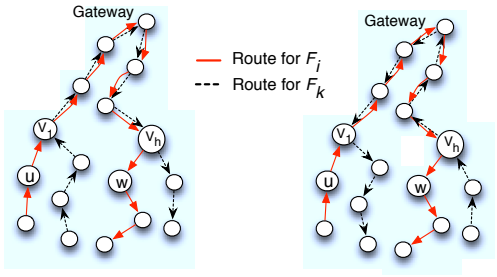
Now we analyze the delay that a flow can experience due to transmission conflicts. Whenever two transmissions conflict, the transmission that belongs to the lower priority flow must be delayed, no matter how many channels are available. Since different transmissions experience different degrees of conflict during communication, these conflicts play a major role in analyzing the end-to-end delays in the network. In the following discussion, we derive an upper bound of the delay that a lower priority flow can experience from the higher priority ones due to conflicts.

Two flows F_k and F_i are said to be *conflicting* when a transmission of F_k conflicts with a transmission of F_i, i.e., their transmissions involve a common node. When F_k and $F_i \in hp(F_k)$ conflict, F_k has to be delayed due to having lower priority. Intuitively, the amount of delay depends on how their routes intersect. A transmission $\overrightarrow{uv}$ of F_k is delayed at most by ω slots by an instance of F_i, if F_i has ω transmissions that involve node u or v. For example, in Figure 2(a), a transmission $\overrightarrow{uv}$ or $\overrightarrow{vw}$ of F_k has to be delayed at most by 2 slots by an instance of F_i. Let $Q(k,i)$ be the total number of F_i's transmissions that share nodes on F_k's route. Since two routes can intersect arbitrarily, in the worst case, flow F_k may conflict with each of these $Q(k,i)$ transmissions of F_i. As a result, $Q(k,i)$ represents an upper bound of the delay that F_k can experience from an instance of F_i due to conflicts.

$Q(k,i)$ often overestimates the delay because when there is "too much" overlap between the routes of F_i and F_k, F_i will not necessarily cause "too much" delay to F_k. We define $\Delta(k,i)$ as a more precise upper bound of the delay that F_k can experience from an instance of F_i due to transmission conflicts. In Figure 2(a), an instance of F_k can be delayed by an instance of F_i at most by 5 slots since $Q(k,i) = 5$, but in Figure 2(b), F_k can be delayed by an instance of F_i at most by 3 slots while $Q(k,i) = 8$. To obtain a value of $\Delta(k,i)$, we introduce the concept of a *maximal common path (MCP)* between F_k and F_i defined as a path $v_1 \to v_2 \to \cdots \to v_h$, where $v_l \neq v_q$ for $l \neq q$ (where $1 \leq l, q \leq h$), on F_i's route such that $v_1 \to v_2 \to \cdots \to v_h$ or $v_h \to v_{h-1} \to \cdots \to v_1$ is a path on F_k's route and it is maximal, i.e., no such longer path contains it (Figure 2(b)). On an MCP between F_k and F_i, denoted by $M_j(k,i)$, F_k can be directly delayed by F_i at most by 3 slots, no matter how long the MCP is. For $M_j(k,i)$, we define its *length* $\beta_j(k,i)$ as the total number of F_i's transmissions along it. That is, for $M_j(k,i) = v_1 \to \cdots \to v_h$, if there exist $u, w \in V$ such that $u \to v_1 \to \cdots \to v_h \to w$ is also on F_i's route, then $\beta_j(k,i) = h+1$. If only u or only w exists, then $\beta_j(k,i) = h$. If neither u nor v does exist, then $\beta_j(k,i) = h-1$. During the time when F_i executes these transmissions (i.e., $\overrightarrow{uv_1}$, $\overrightarrow{v_1v_2}, \cdots, \overrightarrow{v_hw}$), it can cause delay to F_k at most by 3 of these transmissions.



(a) $Q(k,i) = 5$ and $\Delta(k,i) = 5$



(b) Two scenarios where routes of F_k and F_i overlap with $Q(k,i) = 8$, but $\Delta(k,i) \leq 3$

Fig. 2. An example when F_k can be delayed by F_i

Thus, Lemma 2 establishes a value of $\Delta(k,i)$.

Lemma 2: Let $\beta'_j(k,i)$ denote the length of an MCP $M'_j(k,i)$ between F_k and $F_i \in hp(F_k)$ with length at least 4. If there are total $\sigma(k,i)$ MCPs between F_k and F_i each with length at least 4, then

$$\Delta(k,i) = Q(k,i) - \sum_{j=1}^{\sigma(k,i)} \Big(\beta'_j(k,i) - 3\Big) \tag{6}$$

Proof: Let an MCP $M'_j(k,i)$ be $v_1 \rightarrow \cdots \rightarrow v_h$. Let there exist u and w such that the path $u \rightarrow v_1 \rightarrow \cdots \rightarrow v_h \rightarrow w$ is on F_i's route. Now, either $v_1 \rightarrow \cdots \rightarrow v_h$ or $v_h \rightarrow \cdots \rightarrow v_1$ must lie on F_k's route (Figure 2(b)). If $v_1 \rightarrow \cdots \rightarrow v_h$ is on F_k's route, then a transmission $v_l\overrightarrow{v_{l+1}}$, $1 \leq l < h$, of F_k on this path shares node with at most 3 transmissions of F_i on $u \rightarrow v_1 \rightarrow \cdots \rightarrow v_h \rightarrow w$. Similarly, if $v_h \rightarrow \cdots \rightarrow v_1$ is on F_k's route, then a transmission $v_l\overrightarrow{v_{l-1}}$, $1 < l \leq h$, of F_k on this path shares node with at most 3 transmissions of F_i on $u \rightarrow v_1 \rightarrow \cdots \rightarrow v_h \rightarrow w$. Therefore, in either case, a transmission of F_k on $M'_j(k,i)$ can be delayed by the transmissions of F_i on $M'_j(k,i)$ at most by 3 slots. Again, in either case, once the delayed transmission of F_k is scheduled, the subsequent transmissions of F_k and F_i on $M'_j(k,i)$ do not conflict and can happen in parallel. That is, for any $M'_j(k,i)$ with length at least 4, at least $\beta'_j(k,i) - 3$ transmissions will not cause delay to F_k. But $Q(k,i)$ counts every transmission of F_i on $M'_j(k,i)$. Therefore, $Q(k,i) - \sum_{j=1}^{\sigma(k,i)} \big(\beta'_j(k,i) - 3\big)$ represents the bound $\Delta(k,i)$. □

According to Lemma 2, we need to look for an MCP only if $Q(k,i) \geq 4$ and at least 4 consecutive transmissions of F_i share nodes on F_k's route. This is because in such cases looking for an MCP will no longer reduce the bound as the delay is (already) at most 3 (as $Q(k,i)$ is at most 3). Again, when $\beta'_j(k,i)$ is calculated for an $M'_j(k,i)$, we look for the next MCP only if $Q(k,i) - \beta'_j(k,i) \geq 4$.

The number of instances of flow $F_i \in hp(F_k)$ that contribute to the delay of an instance of flow F_k during a time interval of t slots is upper bounded by $\lceil \frac{t}{P_i} \rceil$. Hence, the total delay that an instance of F_k can experience from flow F_i is at most $\left\lceil \frac{t}{P_i} \right\rceil \Delta(k,i)$. An upper bound of the total delay that flow F_k can experience from all higher priority flows due to transmission conflicts during a time interval of t slots is denoted by $\Theta_k(t)$ and can thus be expressed as

$$\Theta_k(t) = \sum_{F_i \in hp(F_k)} \left\lceil \frac{t}{P_i} \right\rceil . \Delta(k,i) \tag{7}$$

5.3 A Tighter Bound on Conflict Delay

The upper bound derived in Equation 7 for the transmission conflict delay experienced by a flow is based on pessimistic assumptions that will result in overestimate of the end-to-end delay of the flow. In this subsection, we avoid the pessimistic assumptions, and establish a tighter bound on the delay of a flow that occurs due to transmission conflict.

Since a flow is a chain of transmissions from a source to a destination, in considering the conflict delay caused by multiple instances of F_i on flow F_k, we observe that at the time when a transmission of F_k conflicts with some transmission of F_i, the preceding transmissions on F_k are already scheduled. These *already scheduled* transmissions of F_k are no longer subject to delay by the subsequent instances of F_i. For example, in Figure 2(a) let us consider that one instance of F_i is conflicting and causing delay on F_k's transmission $\vec{vw}$. This implies that F_k's transmission $\vec{uv}$ is already scheduled (since transmission $\vec{vw}$ can be ready only after transmission $\vec{uv}$ is scheduled). Hence, the next instance of F_i must not cause delay on transmission $\vec{uv}$ (since this transmission is already scheduled). That is, in calculating $\Theta_k(t)$ for F_k, only the transmissions that have not yet been scheduled should be considered for conflict delay by the subsequent instances of F_i (that will be released in future in the considered time interval). These observations lead to Lemma 3, and then to Theorem 4 to upperbound the total delay (due to transmission conflict) caused on F_k by all instances of F_i.

Lemma 3: Let us consider any two instances of a higher priority flow F_i such that each causes conflict delay on a lower priority flow F_k in a time interval. Then, there is at most one common transmission on F_k that can be delayed by both instances.

Proof: Let these two instances of F_i be denoted by $F_{i,1}$ and $F_{i,2}$, where $F_{i,1}$ is released before $F_{i,2}$. Suppose to the contrary, both of these instances cause delay on two transmissions, say τ_j and τ_r, of the

lower priority flow F_k. Without loss of generality, we assume that τ_j precedes τ_r on the route of flow F_k. $F_{i,1}$ causes delay on τ_r because τ_r is ready to be scheduled. This implies that τ_j has already been scheduled. Hence, $F_{i,2}$ which releases after $F_{i,1}$ cannot cause any delay on τ_j, thereby contradicting our assumption. □

Based on Lemma 3, we can now determine a tight upper bound of the conflict delay caused by multiple instances of F_i on F_k in any case. To do so, we introduce the notion of a *bottleneck transmission* (of F_k with respect to F_i) which is the transmission of F_k that may face the maximum conflict delay from F_i. An upper bound of the conflict delay caused by one instance of F_i on F_k's bottleneck transmission is denoted by $\delta(k,i)$, and is determined in the following way. For every transmission τ of F_k, we count the total number of F_i's transmissions that share a node with τ. Then, the maximum of these values (among all transmissions of F_k) is determined as $\delta(k,i)$. In other words, there are at most $\delta(k,i)$ transmissions of (one instance of) F_i such that each of them share a node (and hence may conflict) with the same transmission of F_k. By Lemma 3, for any two instances of F_i, F_k has at most one transmission on which both instances can cause delay. In the worst case, the bottleneck transmission of F_k can be delayed by multiple instances of F_i. Hence, the value of $\delta(k,i)$ plays a major role in determining the delay caused by F_i on F_k as shown in Theorem 4.

Theorem 4: In a time interval of t slots, the worst case conflict delay caused by a higher priority flow F_i on a lower priority flow F_k is upper bounded by

$$\Delta(k,i) + \Big(\Big\lfloor \frac{t}{P_i} \Big\rfloor - 1 \Big).\delta(k,i) + \min\Big(\delta(k,i), t \bmod P_i\Big)$$

Proof: For the case when $t < P_i$, there is at most one instance of F_i in a time interval of t slots. Hence, the total conflict delay caused by F_i on F_k is at most $\Delta(k,i)$ which clearly follows the theorem. We consider the case with $t \geq P_i$ for the rest of the proof.

There are at most $\lceil \frac{t}{P_i} \rceil$ instances of F_i in a time interval of t slots. Let the set of transmissions of F_i which cause conflict delay on F_k be denoted by Γ. When one instance $F_{i,1}$ of F_i causes conflict delay on F_k, a subset Γ_1 of Γ causes the delay. Now consider a second instance $F_{i,2}$ of F_i. For $F_{i,2}$, another subset Γ_2 of Γ causes delay on F_k. When all subsets $\Gamma_1, \Gamma_2, \cdots, \Gamma_{\lceil \frac{t}{P_i} \rceil}$ are mutually disjoint, by the definition of $\Delta(k,i)$, the conflict delay caused by Γ on F_k is at most $\Delta(k,i)$. Hence, the total conflict delay caused by all $\Gamma_1, \Gamma_2, \cdots, \Gamma_{\lceil \frac{t}{P_i} \rceil}$ in this case is at most $\Delta(k,i)$. That is, the total conflict delay on F_k caused by F_i is at most $\Delta(k,i)$.

Now let us consider the case when the subsets $\Gamma_1, \Gamma_2, \cdots, \Gamma_{\lceil \frac{t}{P_i} \rceil}$ are not mutually disjoint, i.e., there is at least one pair Γ_j, Γ_h such that $\Gamma_j \cap \Gamma_h \neq \emptyset$, where $1 \leq j, h \leq \lceil \frac{t}{P_i} \rceil$. Let the total delay caused by all instances of F_i on F_k is $\Delta(k,i) + Z(k,i)$, i.e., the delay is higher than $\Delta(k,i)$ by $Z(k,i)$ time slots. The additional delay (beyond $\Delta(k,i)$) happens because the transmissions that are common between Γ_j and Γ_h cause both instances of F_i to create delay on F_i. By Lemma 3, for any two instances of F_i, F_k has at most one transmission on which both instances can cause delay. If there is no transmission of F_k that is delayed by both the p-th instance and the $p+1$-th instance of F_i, then no transmission of F_k is delayed by both the p-th instance and the q-th instance of F_i, for any $q > p+1$, where $1 \leq p < \lceil \frac{t}{P_i} \rceil$. Thus, $Z(k,i)$ is maximum when for each pair of consecutive instances (say, the p-th instance and $p+1$-th instance, for each p, $1 \leq p < \lceil \frac{t}{P_i} \rceil$) of F_i, there is a transmission of F_k that is delayed by both instances. Hence, at most $\lceil \frac{t}{P_i} \rceil - 1$ instances contribute to this additional delay $Z(k,i)$, each instance causing some additional delay on a transmission. Since one instance of F_i can cause delay on a transmission of F_k at most by $\delta(k,i)$ slots, $Z(k,i) \leq (\lceil \frac{t}{P_i} \rceil - 1)\delta(k,i)$. Since the last instance may finish after the considered time window of t slots, the delay caused by it is at most $\min(\delta(k,i), t \bmod P_i)$ slots. Taking this into consideration, $Z(k,i) \leq (\lfloor \frac{t}{P_i} \rfloor - 1)\delta(k,i) + \min(\delta(k,i), t \bmod P_i)$. Thus, the total delay caused on F_k by all instances of F_i is at most

$\Delta(k,i) + Z(k,i)$

$\leq \Delta(k,i) + (\lfloor \frac{t}{P_i} \rfloor - 1).\delta(k,i) + \min(\delta(k,i), t \bmod P_i)$ □

From Theorem 4, now $\Theta_k(t)$ (i.e., an upper bound of the total delay flow F_k can experience from all higher priority flows due to transmission conflicts during a time interval of t slots) is calculated as follows.

$$\Theta_k(t) = \sum_{F_i \in hp(F_k)} \Bigg(\Delta(k,i) + \Big(\Big\lfloor \frac{t}{P_i} \Big\rfloor - 1 \Big).\delta(k,i) + \min\Big(\delta(k,i), t \bmod P_i\Big) \Bigg) \tag{8}$$

Since usually $\delta(k,i) \ll \Delta(k,i)$, the above value of $\Theta_k(t)$ is significantly smaller than that derived in Equation 7. Our simulation results (in Section 7) also demonstrate that the above bound is a significant improvement over the bound derived in Equation 7.

5.4 End-to-End Delay Bound

Now we consider both types of delays together to develop an upper bound of the end-to-end delay of every flow. For a flow, we first derive an upper bound of its end-to-end delay assuming that it does not conflict with any higher priority flow. We then incorporate its worst case delay due to conflict into this upper bound. This is done for every flow in decreasing order of priority starting with the highest priority flow as explained below.

For F_k, we use $R_k^{\text{ch,con}}$ to denote an upper bound of the worst case end-to-end delay considering delays due to both **ch**annel contention and **con**flicts between flows. We use the following two steps to estimate

$R_k^{\text{ch,con}}$ for every flow $F_k \in F$ in decreasing order of priority starting with the highest priority flow.

5.4.1 Step 1

First, we calculate a pseudo upper bound (i.e., not an actual upper bound), denoted by R_k^{ch}, of the worst case end-to-end delay of F_k assuming that F_k is delayed by the higher priority flows due to channel contention only. That is, we assume that F_k does not conflict with any higher priority flow. This calculation is based on the upper bounds $R^{\text{ch,con}}$ of the worst case end-to-end delays of the higher priority flows which are already calculated considering both types of delay. Based on our discussion in Subsection 5.1, to determine R_k^{ch}, the worst case delay that flow F_k will experience from the higher priority flows can be calculated using Equation 5. The amount of delay that a higher priority flow F_i will cause to F_k depends on F_i's workload during a BP(k, x) (i.e., a level-k busy period of x slots). Note that, in Equations 1 and 3, the workload bound of F_i was derived in absence of conflict between the flows. Now we first analyze the workload bound of $F_i \in hp(F_k)$ in the network where both channel contention and transmission conflicts contributed to the worst case end-to-end delay of F_i.

From Equation 1, if flow F_i does not have carry-in, its workload $W_k^{\text{nc}}(F_i, x)$ during a BP(k, x) does not depend on its worst case end-to-end delay. Therefore, if F_i has no carry-in, $W_k^{\text{nc}}(F_i, x)$ during a BP(k, x) still can be calculated using Equation 1, no matter what the worst case end-to-end delay of F_i is. That is,

$$W_k^{\text{nc}}(F_i, x) = \left\lfloor \frac{x}{P_i} \right\rfloor . C_i + \min(x \bmod P_i, C_i) \qquad (9)$$

Now $I_k^{\text{nc}}(F_i, x)$ is calculated using Equation 2 and is guaranteed to be an upper bound of the delay that $F_i \in hp(F_k)$ can cause to F_k due to channel contention.

From Equation 3, when flow F_i has carry-in, its workload $W_k^{\text{ci}}(F_i, x)$ during a BP(k, x) depends on its worst case response time R_i. Equation 3 also indicates that $W_k^{\text{ci}}(F_i, x)$ is monotonically nondecreasing in R_i. Now, in the WirelessHART network, an upper bound of the end-to-end delay of F_i must be no less than R_i since both channel contention and transmission conflicts contribute to its end-to-end delay. That is, $R_i^{\text{ch,con}} \geq R_i$. Therefore, if we replace R_i with $R_i^{\text{ch,con}}$ in Equation 3, $W_k^{\text{ci}}(F_i, x)$ is guaranteed to be an upper bound of F_i's workload during a BP(k, x). Thus,

$$W_k^{\text{ci}}(F_i, x) = \left\lfloor \frac{\max(x - C_i, 0)}{P_i} \right\rfloor . C_i + C_i + \mu_i \qquad (10)$$

where $\mu_i = \min\Big(\max\Big(\lambda - (P_i - R_i^{\text{ch,con}}), 0\Big), C_i - 1\Big)$ and $\lambda = \max(x - C_i, 0) \bmod P_i$. Similarly, $I_k^{\text{ci}}(F_i, x)$ calculated using Equation 4 is guaranteed to be an upper bound of the delay that F_i can cause to F_k due to channel contention.

Once the bounds $I_k^{\text{nc}}(F_i, x)$ and $I_k^{\text{ci}}(F_i, x)$ of the delay from every higher priority flow $F_i \in hp(F_k)$ are calculated, the total delay $\Omega_k(x)$ that an instance of F_k experiences from all higher priority flows during a BP(k, x) due to channel contention is calculated using Equation 5. Now assuming that F_k does not conflict with any higher priority flow, an upper bound of its end-to-end delay can be found using the same iterative method that is used for multiprocessor scheduling [3]. Since there are m channels, the pseudo upper bound R_k^{ch} of the worst case end-to-end delay of F_k can be obtained by finding the minimal value of x that solves Equation 11.

$$x = \left\lfloor \frac{\Omega_k(x)}{m} \right\rfloor + C_k \qquad (11)$$

Equation 11 is solved using an iterative fixed-point algorithm starting with $x = C_k$. This algorithm either terminates at some fixed-point $x^* \leq D_k$ that represents the bound R_k^{ch} or x will exceed D_k eventually. In the latter case, this algorithm terminates and reports the instance as *"unschedulable"*.

Effect of Channel Hopping. To every transmission, the scheduler assigns a channel offset between 0 and $m - 1$ instead of an actual channel, where m is the total number of channels. Any channel offset c (i.e., $1, 2, \cdots, m-1$) is mapped to different channels at different time slots s as follows.

$$channel = (c + s) \bmod m$$

That is, although the physical channels used along a link changes (hops) in every time slot, the total number m of available channels is fixed. The scheduler only assigns a fixed channel index to a transmission which maps to different physical channels in different time slots, keeping the total number of available channels at m always, and scheduling each flow on at most one channel at any time. Hence, channel hopping does not have effect on channel contention delay.

5.4.2 Step 2

Once the value of R_k^{ch} is computed, we incorporate the transmission conflict delay into it to obtain the bound $R_k^{\text{ch,con}}$. Namely, for flow F_k, the bound R_k^{ch} has been derived in Step 1 by assuming that F_k does not conflict with any higher priority flow. Therefore, in this step, we take into account that F_k may conflict with the higher priority flows and, hence, can experience further delay from them. An upper bound $\Theta_k(y)$ of the total delay that an instance of F_k can experience due to conflicts with the higher priority flows during a time interval of y slots is calculated using Equation 8. Note that when F_k conflicts with some higher priority flow it must be delayed, no matter how many channels are available. Therefore, we add the delay $\Theta_k(y)$ to the pseudo upper bound R_k^{ch} to derive an upper bound of F_k's worst case end-to-end delay. Thus, the minimal value of y that solves the following equation gives the bound $R_k^{\text{ch,con}}$ for F_k

that includes both types of delay:

$$y = R_k^{\text{ch}} + \Theta_k(y) \tag{12}$$

Equation 12 is solved using an iterative fixed-point algorithm starting with $y = R_k^{\text{ch}}$. Like Step 1, this algorithm also either terminates at some fixed-point $y^* \leq D_k$ that is considered as the bound $R_k^{\text{ch,con}}$ or terminates with an *"unschedulable"* decision when $y > D_k$. Thus, termination of the algorithm is guaranteed.

Theorem 5: For every flow $F_k \in F$, let R_k^{ch} be the minimal value of $x \geq C_k$ that solves Equation 11, and $R_k^{\text{ch,con}}$ be the minimal value of $y \geq R_k^{\text{ch}}$ that solves Equation 12. Then $R_k^{\text{ch,con}}$ is an upper bound of the worst case end-to-end delay of F_k.

Proof: Flows are ordered according to their priorities as $F_1, F_2, \cdots, F_N$ with F_1 being the highest priority flow. We use mathematical induction on priority level k, $1 \leq k \leq N$. When $k = 1$, i.e., for the highest priority flow F_1, Equations 11 and 12 yield $R_1^{\text{ch,con}} = C_1$, where C_1 is the number of transmissions along F_1's route. Since no flow can delay the highest priority flow F_1, the end-to-end delay of F_1 is always C_1. Hence, the upper bound calculated using Equation 12 holds for $k = 1$. Now let the upper bound calculated using Equation 12 holds for flow F_k, for any k, $1 \leq k < N$. We have to prove that the upper bound calculated using it also holds for flow F_{k+1}.

To calculate $R_{k+1}^{\text{ch,con}}$ in Step 2, we initialize y (in Equation 12) to R_{k+1}^{ch}. Note that R_{k+1}^{ch} is computed in Step 1 for flow F_{k+1}. In Step 1, R_{k+1}^{ch} is computed considering upper bounds $R_h^{\text{ch,con}}$ of the worst case end-to-end delays of all F_h with $h < k+1$ which are already computed considering both types of delay. Equation 11 assumes that F_{k+1} does not conflict with any higher priority flow. This implies that the minimal solution of x, i.e., R_{k+1}^{ch} is an upper bound of the worst case end-to-end delay of F_{k+1}, if F_{k+1} is delayed by the higher priority flows due to channel contention only. If F_{k+1} conflicts with some higher priority flow, then it can be further delayed by the higher priority flows at most by $\sum_{F_h \in hp(F_{k+1})} \lceil \frac{y}{P_h} \rceil \Delta(k+1, h)$ slots during any time interval of length y. Equation 12 adds this delay to R_{k+1}^{ch} and establishes the recursive equation for y. Therefore, the minimal solution of y, i.e., $R_{k+1}^{\text{ch,con}}$ is guaranteed to be an upper bound of the worst case end-to-end delay of F_{k+1} that includes the worst case delays both due to channel contention and due to conflicts between flows. □

The end-to-end delay analysis procedure calculates $R_i^{\text{ch,con}}$, for $i = 1, 2, \cdots, N$ (in decreasing order of priority level), and decides the flow set to be schedulable if, for every $F_i \in F$, $R_i^{\text{ch,con}} \leq D_i$. According to Equations 11 and 12, each $R_i^{\text{ch,con}}$ can be calculated in pseudo polynomial time for every F_i. The correctness of this upper bound of the worst case end-to-end delay follows from Theorem 5.

Note that our above analysis has been derived considering the retransmission parameter $\chi = 1$. If every transmission is repeated χ times to handle retransmission on a single route, then every time slot is simply multiplied by χ in delay calculation. Hence, to adopt the above delay analysis for any general value of χ, we simply replace the values of $C_i, \Delta(k,i)$, and $\delta(k,i)$ with $C_i.\chi$, $\Delta(k,i).\chi$, and $\delta(k,i).\chi$, respectively. Our model is motivated by WirelessHART [2] that uses a fixed number of retransmissions for all links. It is trivial from the above analysis to handle varying the number of transmissions for different links based on link qualities. Specifically, instead of multiplying the above values by a uniform value of χ, we have to consider different values for different links.

6 Delay Analysis in Polynomial Time

We now extend the pseudo polynomial time analysis to a polynomial time method. While this may provide comparatively looser bounds, it can calculate the bounds more quickly, and hence is more suitable for online use when time efficiency is critical.

Exploiting the same mapping presented in Section 5, we can use the polynomial time response time analysis for global multiprocessor scheduling proposed in [25] to calculate the channel contention delays. In particular, using this analysis, the maximum channel contention delay, denoted by $\Omega_k(D_k)$, that a flow F_k can experience during its lifetime from the higher priority flows can be expressed as follows.

$$\Omega_k(D_k) = \sum_{F_i \in hp(F_k)} \min(W_k(i), D_k - C_k + 1), \tag{13}$$

where $W_k(i) = \left\lfloor \frac{D_k + D_i - C_i}{P_i} \right\rfloor .C_i \; + \; \min\left(C_i, D_k + D_i - C_i - \left\lfloor \frac{D_k + D_i - C_i}{P_i} \right\rfloor .P_i\right)$

Therefore, similar to Equation 11, R_k^{ch} of F_k (i.e., the worst case end-to-end delay of F_k assuming that it is delayed by the higher priority flows due to channel contention only) can be calculated as follows.

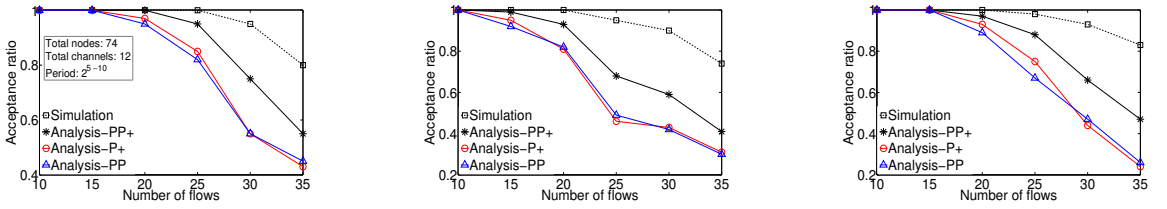
$$R_k^{\text{ch}} = \left\lfloor \frac{\Omega_k(D_k)}{m} \right\rfloor + C_k \tag{14}$$

To calculate the conflict delay of F_k in polynomial time, we can estimate the maximum delay in an interval of D_k slots from Equation 8 as follows.

$$\Theta_k(D_k) = \sum_{F_i \in hp(F_k)} \left(\Delta(k,i) + \left(\left\lfloor \frac{D_k}{P_i} \right\rfloor - 1 \right).\delta(k,i) + \min\left(\delta(k,i), D_k \bmod P_i\right) \right) \tag{15}$$

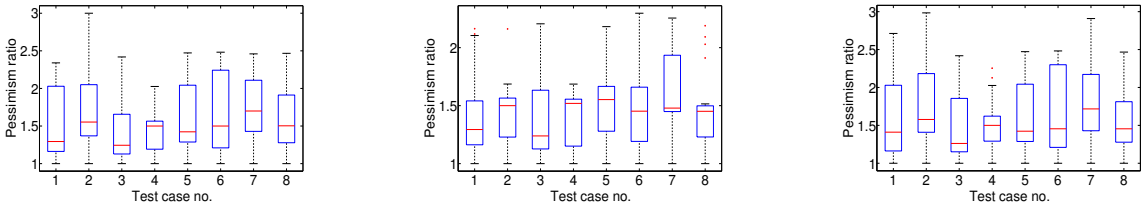
Like Equation 12, the worst case end-to-end delay $R_k^{\text{ch,con}}$ of flow F_k considering both channel contention delay and transmission conflict delay is calculated as

$$R_k^{\text{ch,con}} = R_k^{\text{ch}} + \Theta_k(D_k) \tag{16}$$



(a) Topology at -1 dBm Tx power (b) Topology at -3 dBm Tx power (c) Topology at -5 dBm Tx power

Fig. 3. Schedulability without retransmission on testbed topology



(a) Analysis-PP (b) Analysis-PP+ (c) Analysis-P+

Fig. 4. Pessimism ratio without retransmission on testbed topology

7 Evaluation

We evaluate our end-to-end delay analysis through simulations based on both random topologies and a real wireless sensor network testbed topologies. Evaluations are performed in terms of acceptance ratio and pessimism ratio. *Acceptance ratio* is the proportion of the number of test cases deemed schedulable by the delay analysis method to the total number of test cases. For each flow, *pessimism ratio* is quantified as the proportion of the analyzed theoretical bound to its maximum end-to-end delay observed in simulation. In particular, *pessimism ratio* quantifies our overestimate in the analytical delay bounds. Due to this overestimate in the delay bounds, some test case that is schedulable may be determined as unschedulable by our conservative delay analysis, and hence is rejected by admission control based on our analysis. The impact of a sufficient delay analysis on the pessimism of admission control is quantified by the *acceptance ratio* metric. The higher the acceptance ratio, the less pessimistic (i.e., more effective) the delay analysis.

There is no baseline to compare the performance of our analysis which, to our knowledge, is the first delay analysis for real-time flows in WirelessHART networks. Hence, we evaluate the performance of our delay analysis by observing the delays through simulations of the complete schedule of all flows released within the hyper-period. In the figures in this section, **"Simulation"** denotes the fraction of test cases that have no deadline misses in the simulations. This fraction indicates an upper bound of acceptance ratio for any delay analysis method. The analyses evaluated in this section are named as follows.

Analysis-PP is the pseudo polynomial time analysis without considering the improved conflict delay bound of Section 5.3. Namely, it calculates the end-to-end delay bound using Equation 12 where the conflict delay is calculated based on Equation 7.

Analysis-PP+ is the pseudo polynomial time analysis by considering the tighter conflict delay bound of Section 5.3. That is, Analysis-PP+ calculates the end-to-end delay bound using Equation 12 where the conflict delay is calculated based on Equation 8.

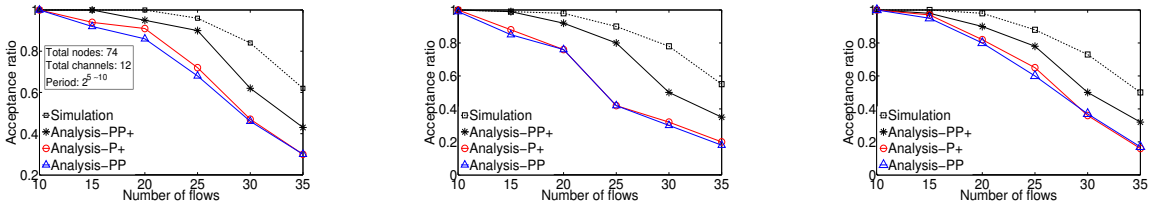
Analysis-P+ is the polynomial time analysis derived in Section 6. It calculates the delay bounds using Equation 16 based on the tighter conflict delay bound.

7.1 Simulation Setup

A fraction of nodes is considered as sources and destinations. The sets of sources and destinations are disjoint. The *reliability* of a link is represented by the *packet reception ratio (PRR)* along it. The node with the highest number of neighbors is designated as the gateway. Since all flows pass through the gateway, we determine routes between the sources and destinations that include the gateway. Routes are determined based on link reliabilities. The most reliable route connecting a source to a destination is determined as the primary route. For additional routes, we choose the next most reliable route that excludes the links of any existing route between the same source and destination. Each flow is assigned a harmonic period of the form 2^a time slots, where $a > 1$. The deadline of each flow is set equal to its period. The priorities of the flows are assigned based on *deadline monotonic* policy that assigns priorities according to relative deadlines. The bandwidth is assumed to be sufficient to accommodate a transmission within a time slot.

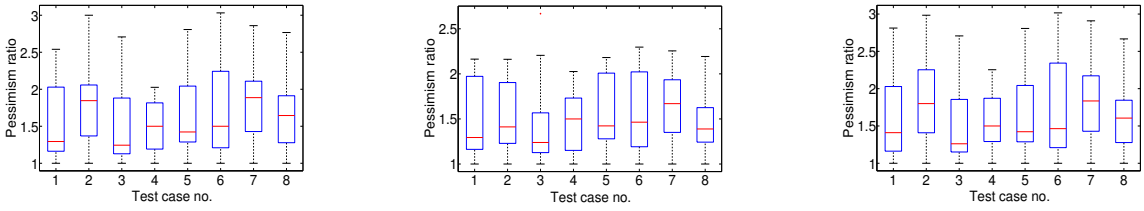
7.2 Simulations with Testbed Topologies

Due to large impact of transmission conflicts on the end-to-end delay of a flow, the delay analysis largely depends on the topology of the network since transmission conflicts depend on how the links or routes intersect (as seen in Sections 5.2 and 5.3). Therefore, first we conduct simulation results based on real network topologies. These are the topologies of a wireless sensor network testbed, and are generated using various transmission power levels of its nodes



(a) Topology at -1 dBm Tx power (b) Topology at -3 dBm Tx power (c) Topology at -5 dBm Tx power

Fig. 5. Schedulability with retransmission on testbed topology



(a) Analysis-PP (b) Analysis-PP+ (c) Analysis-P+

Fig. 6. Pessimism ratio with retransmission on testbed topology

since the network connectivity (hence the topology) varies as we vary transmission powers. Our testbed consists of 74 TelosB motes each equipped with Chipcon CC2420 radios which are compliant with IEEE 802.15.4 (WirelessHART's physical layer is also based on IEEE 802.15.4). It is deployed in two buildings of Washington University [26]. Setting the same transmission (Tx) power at every node, each node (in a round-robin fashion) broadcasts 50 packets while its neighbors record the sequence numbers of the packets they receive. This cycle is repeated giving each node 5 rounds to transmit 50 packets in each round. Every link with a higher than 80% PRR is considered a reliable link to derive the topology of the testbed. We collected topologies at 3 different Tx power levels (-1 dBm, -3 dBm, -5 dBm). We generate different flows in these topologies by randomly selecting the sources and destinations. Their periods are randomly generated in the range $2^{5\sim10}$ time slots. We generate 100 test cases considering these topologies.

Figure 3 shows the acceptance ratios of our delay analysis methods without considering retransmission and without redundant routes. According to Figure 3(a), when the number of flows $N < 25$ in the topology with Tx power of -1 dBm, Analysis-PP+ has an acceptance ratio of 1.0, which means that all test cases that are indeed schedulable are also deemed schedulable by our analysis. When $N = 30$, the value of "Simulation" is 0.99 while the acceptance ratio of Analysis-PP+ is 0.95 which indicates that the analysis is highly efficient. After that, the acceptance ratios of our analysis decreases with the increase in N. However, the difference between its acceptance ratio and the value of "Simulation" always remains strictly less than 0.24. Therefore, the acceptance ratios are always tight for any (moderate or severe) overload in the testbed topology. Besides, the acceptance ratio of Analysis-PP+ is always a lot higher than that of Analysis-PP. This happens because the delay bounds calculated in Analysis-PP+ are significantly tighter than those in Analysis-PP. Analysis-P+ which determines looser bounds in polynomial time is highly competitive against Analysis-PP. This happens because Analysis-P+ determines the conflict delay based on the improvement made in Equation 8. Figures 3(b) and 3(c) show the similar results for the topology with Tx power of -3 dBm and -5 dBm, respectively.

Now we analyze our results to evaluate the tightness of the delay bounds in terms of pessimism ratios. Among 100 test cases, each consisting of 25 flows from the above experiment we randomly select 8 test cases that are schedulable under all 3 analyses, and plot the distributions of pessimism ratios as box plots in Figure 4. The figures indicate that the end-to-end delay bounds calculated in Analysis-PP+ are tighter than those calculated in Analysis-PP since the former uses a tighter bound of conflict delay. Specifically, the results show that the 75th percentiles of the pessimism ratios are no greater than 1.75, 2.2, and 2.25 for Analysis-PP+, Analysis-PP, and Analysis-P+, respectively. This indicates that the delay bounds derived in our new analysis (Analysis-PP+) are much tighter than those in the original analysis (Analysis-PP). Even the polynomial time analysis Analysis-P+ that uses our improved conflict delay analysis is highly competitive against Analysis-PP that is a pseudo polynomial-time analysis. These results thus indicate that incorporating our improved conflict delay analysis into the original analysis significantly tightens the delay bounds. In addition, if we look back to Figure 3 for acceptance ratio, our algorithms are effective for admission control (in term of acceptance ratio) despite the (high) pessimism ratio.

Figure 5 shows the acceptance ratios by considering retransmissions but without redundant routes. In this case, the acceptance ratios are a lot lower than those in Figure 3. This is reasonable because we have to schedule each transmission in two time slots, and due to limited bandwidth the schedulable cases (observed in simulation) are also lower than those in Figure 3

(wthout retransmission). However, these results also indicate that the acceptance ratio of Analysis-PP+ is always higher than that of Analysis-PP. For the same 8 test cases selected in Figure 4, we now draw the pessimism ratios in Figure 6 considering retransmissions. Figure 6 indicates that the pessimism ratios increase in some cases but do not vary a lot compared to the case without retransmission. Since both the analytical delay (x) and the delay observed in simulations (y) increase under retransmissions, the pessimism ratios ($\frac{x}{y}$) do not vary significantly compared to the case without retransmission.

We now determine the schedulability considering both retransmissions and redundant routes. That is, for each transmission along the primary route between a sources and destination is scheduled on 2 time slots. In addition, each packet is also scheduled along each redundant route. Figure 7 shows how the schedulability changes with the increase of number of routes considering 25 flows in the topology with -1 dBm Tx power. When there is no redundant route, the value of "Simulation" is 0.96 while the acceptance ratio under Analysis-PP+ is 0.9. As the number of redundant routes increases, the schedulable cases as well as acceptance ratios decrease sharply. However, at least 50% of the total schedulable cases are determined as schedulable by Analysis-PP+ as long as the number of redundant routes is no greater than 2. When there are 3 redundant routes, the value of "Simulation" is 0.15 and the acceptance ratio under Analysis-PP+ is 0.05. This decrease in acceptance ratio is because many redundant links need to be scheduled.

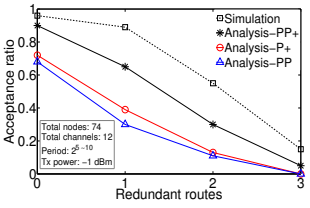


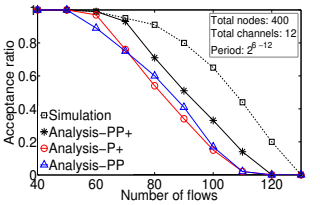
Fig. 7. Schedulability with retransmission and redundant routes on testbed topology

These results demonstrate that the improved analysis (derived in Subsection 5.3) of transmission conflict delay is highly effective in reducing the pessimism of the analysis. It also shows that the polynomial-time analysis is reasonably tight when compared against the original pseudo polynomial time analysis.

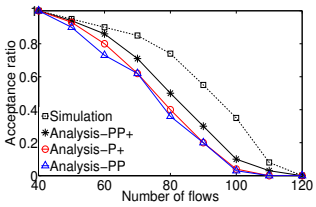
7.3 Simulations with Random Topologies

We test the scalability of our algorithms in terms of number of flows on random topologies of larger number of nodes. Given the number of nodes and edge-density, we generate random networks. A network with n nodes and $\rho\%$ edge-density has a total of $(n(n-1)*\rho)/(2*100)$ bidirectional edges. The edges are chosen randomly and assigned PRR randomly in the range $[0.80, 1.0]$. Then we generate different number of flows in 400-node networks of 40% edge-density. For every different number of flows, we generate 100 test cases. The periods are considered harmonic and are randomly generated in the range $2^{6\sim12}$ time slots. Larger periods (compared to the case with testbed topologies) are used to accommodate large networks and a large number of flows.

The acceptance ratios of our analyses in 400-node network are shown in Figure 8. Figure 8(a) shows that without retransmission the acceptance ratio of Analysis-PP+ is equal to the value of "Simulation" as long as the number of flows is no greater than 60. As the number of flows increases, the difference between the acceptance ratios of Analysis-PP+ and the value of "Simulation" increases but always remains less than 0.33. Figure 8(b) shows the results considering retransmissions along the primary route but no redundant routes. In this case acceptance ratios in all methods are lower since the total number of actual schedulable cases are lower. However, the acceptance ratio of Analysis-PP+ is always higher than that of Analysis-PP, and Analysis-P+ is competitive against Analysis-PP.



(a) Without retransmission



(b) With retransmission

Fig. 8. Schedulability on random topology

Figure 9 shows the results for 80 flows in the 400-node network under retransmissions and varying number of redundant routes. Similar to our results with testbed topology here also we observe that both the value of "Simulation" and the acceptance ratios of our analyses decrease sharply with the increase in the number of redundant routes.

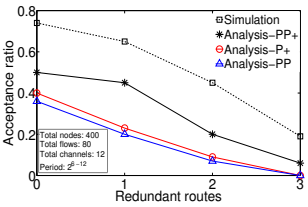


Fig. 9. Schedulability with retransmission and redundant routes on random topology

In every setup, we have observed that the acceptance ratios of our analysis are close to those of simulation which indicates that not many schedulable cases are rejected by our analysis. All test cases accepted by our analysis meet their deadlines in the simulations which demonstrates that the estimated bounds are safe. The results demonstrate that our analysis can be used as an acceptance test for real-time flows under various network configurations.

8 Conclusion

In this paper, we have mapped the transmission scheduling of real-time data flows between sensors

and actuators in a WirelessHART network to real-time multiprocessor scheduling. Based on the mapping, we have presented an end-to-end delay analysis to determine the schedulability of real-time data flows in WirelessHART networks. Through simulation studies, we have demonstrated that our analysis enables effective schedulability tests for WirelessHART networks.

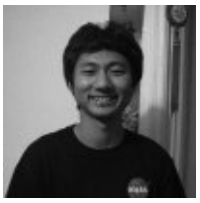
ACKNOWLEDGEMENTS

This work is supported by NSF through grants CNS-1035773 (CPS), CNS-1320921 (NeTS), CNS-1017701 (NeTS) and CNS-1144552 (NeTS).

REFERENCES

[1] D. Chen, M. Nixon, and A. Mok, *WirelessHART^{TM}Real-Time Mesh Network for Industrial Automation*. Springer, 2010.
[2] "WirelessHART," 2007, http://www.hartcomm2.org.
[3] N. Guan, M. Stigge, W. Yi, and G. Yu, "New response time bounds for fixed priority multiprocessor scheduling," in *RTSS '09*.
[4] J. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou, "Real-time communication and coordination in embedded sensor networks," *Proceedings of the IEEE*, vol. 91, no. 7, pp. 1002–1022, 2003.
[5] H. Li, P. Shenoy, and K. Ramamritham, "Scheduling messages with deadlines in multi-hop real-time sensor networks," in *RTAS '05*.
[6] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha, "Flow-based real-time communication in multi-channel wireless sensor networks," in *EWSN '09*.
[7] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly, "Distributed multi-hop scheduling and medium access with delay and throughput constraints," in *MobiCom '01*.
[8] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He, "RAP: A real-time communication architecture for large-scale wireless sensor networks," in *RTAS '02*.
[9] K. Karenos and V. Kalogeraki, "Real-time traffic management in sensor networks," in *RTSS '06*.
[10] T. He, B. M. Blum, Q. Cao, J. A. Stankovic, S. H. Son, and T. F. Abdelzaher, "Robust and timely communication over highly dynamic sensor networks," *Real-Time Sys.*, vol. 37, no. 3, 2007.
[11] T. W. Carley, M. A. Ba, R. Barua, and D. B. Stewart, "Contention-free periodic message scheduler medium access control in wireless sensor/actuator networks," in *RTSS '03*.
[12] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "JiTS: Just-in-time scheduling for real-time sensor data dissemination," in *PERCOM '06*.
[13] Y. Gu, T. He, M. Lin, and J. Xu, "Spatiotemporal delay control for low-duty-cycle sensor networks," in *RTSS '09*.
[14] I.-H. Hou, V. Borkar, and P. R. Kumar, "A theory of QoS for wireless," in *INFOCOM '09*.
[15] I.-H. Hou, A. Truong, S. Chakraborty, and P. R. Kumar, "Optimality of periodwise static priority policies in real-time communications," in *CDC '11*.
[16] T. F. Abdelzaher, S. Prabh, and R. Kiran, "On real-time capacity limits of multihop wireless sensor networks," in *RTSS '04*.
[17] O. Chipara, C. Lu, and G.-C. Roman, "Real-time query scheduling for wireless sensor networks," in *RTSS '07*.
[18] J. B. Schmitt and U. Roedig, "Sensor network calculus –A framework for worst case analysis," in *DCOSS '05*.
[19] S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon, "Reliable and real-time communication in industrial wireless mesh networks," in *RTAS '11*.
[20] R. Alur, A. D'Innocenzo, K. Johansson, G. Pappas, and G. Weiss, "Modeling and analysis of multi-hop control network," in *RTAS '09*.
[21] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for WirelessHART networks," in *RTSS '10*.
[22] ——, "Priority assignment for real-time flows in WirelessHART networks," in *ECRTS '11*.
[23] A. Saifullah, C. Wu, P. Tiwari, Y. Xu, Y. Fu, C. Lu, and Y. Chen, "Near optimal rate selection for wireless control systems," in *RTAS '12*.
[24] http://www2.emersonprocess.com/siteadmincenter/PM%20Rosemount%20Documents/00813-0200-4420.pdf.
[25] M. Bertogna, M. Cirinei, and G. Lipari, "Schedulability analysis of global scheduling algorithms on multiprocessor platforms," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 4, pp. 553–566, 2009.
[26] WSN testbed, http://mobilab.wustl.edu/testbed.

Abusayeed Saifullah is a Ph.D. candidate in the Department of Computer Science and Engineering at Washington University in St Louis. Advised by Chenyang Lu, he is a member of the Cyber-Physical Systems Laboratory at Washington University. Abu's research focuses on real-time wireless sensor-actuator networks used in emerging cyber-physical systems, and spans a broad range of topics in wireless sensor networks, embedded systems, real-time systems, and multi-core parallel computing. He has received the best student paper awards at the 32nd IEEE Real-Time Systems Symposium (RTSS 2011) and at the 5th International Symposium on Parallel and Distributed Processing and Applications (ISPA 2007), and best paper nomination at the 18th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2012).



You Xu is a Ph.D. candidate of Computer Science at the Washington University in St. Louis. He received the M.Sc degree in computer science from the Washington University in St. Louis in 2009. His research interests include nonlinear optimization, constrained search, planning and scheduling. He is now working for Google.

Chenyang Lu is a Professor of Computer Science and Engineering at Washington University in St. Louis. Professor Lu is Editor-in-Chief of ACM Transactions on Sensor Networks, Area Editor of IEEE Internet of Things Journal and Associate Editor of Real-Time Systems. He also serves as Program Chair of premier conferences such as IEEE Real-Time Systems Symposium (RTSS 2012), ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS 2012) and ACM Conference on Embedded Networked Sensor Systems (SenSys 2014). Professor Lu is the author and co-author of over 100 research papers with over 10000 citations and an h-index of 47. He received the Ph.D. degree from University of Virginia in 2001, the M.S. degree from Chinese Academy of Sciences in 1997, and the B.S. degree from University of Science and Technology of China in 1995, all in computer science. His research interests include real-time systems, wireless sensor networks and cyber-physical systems.

Yixin Chen is an Associate Professor of Computer Science at the Washington University in St Louis. His research interests include data mining, machine learning, artificial intelligence, optimization, and cyber-physical systems. He received a Ph.D. in Computing Science from University of Illinois at Urbana-Champaign in 2005. He received the Best Paper Award at the AAAI Conference on Artificial Intelligence (2010) and International Conference on Tools for AI (2005), and best paper nomination at the ACM KDD Conference (2009). His work on planning has won First Prizes in the International Planning Competitions (2004 & 2006). He has received an Early Career Principal Investigator Award from the Department of Energy (2006) and a Microsoft Research New Faculty Fellowship (2007). He is an Associate Editor for ACM Transactions of Intelligent Systems and Technology and IEEE Transactions on Knowledge and Data Engineering, and serves on the Editorial Board of Journal of Artificial Intelligence Research.