View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Washington University St. Louis: Open Scholarship

Washington University in St. Louis

Washington University Open Scholarship

All Computer Science and Engineering Research

Computer Science and Engineering

Report Number: WUCSE-2011-86

2011

End-to-End Communication Delay Analysis in WirelessHART Networks

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

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 map the scheduling of real-time periodic data flows in a WirelessHART 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... **Read complete abstract on page 2.**

Follow this and additional works at: https://openscholarship.wustl.edu/cse_research

 Part of the Computer Engineering Commons, and the Computer Sciences Commons

Recommended Citation

Saifullah, Abusayeed; Xu, You; Lu, Chenyang; and Chen, Yixin, "End-to-End Communication Delay Analysis in WirelessHART Networks" Report Number: WUCSE-2011-86 (2011). *All Computer Science and Engineering Research.*
https://openscholarship.wustl.edu/cse_research/68

Department of Computer Science & Engineering - Washington University in St. Louis
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160.

This technical report is available at Washington University Open Scholarship: https://openscholarship.wustl.edu/cse_research/68

End-to-End Communication Delay Analysis in WirelessHART Networks

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

Complete 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 map the scheduling of real-time periodic data flows in a WirelessHART 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 a WirelessHART 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.

Department of Computer Science & Engineering

Washington University in St. Louis
School of Engineering & Applied Science

2011-86

End-to-End Communication Delay Analysis in WirelessHART Networks

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

Corresponding Author: saifullaha@cse.wustl.edu

Web Page: http://www.cse.wustl.edu/~saifullaha/

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 map the scheduling of real-time periodic data flows in a WirelessHART 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 a WirelessHART 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.

Type of Report: Other

Department of Computer Science & Engineering - Washington University in St. Louis
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160

End-to-End Communication Delay Analysis in WirelessHART 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 map the scheduling of real-time periodic data flows in a WirelessHART 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 a WirelessHART 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. Similarly, many parts of a plant area are equipped with safety valves; failure in real-time monitoring and control of these valves may lead to accidents and even serious explosions in the plant area.

To address the challenges in industrial monitoring and control, WirelessHART [2] has been designed as an open WSAN standard specifically for process industries. 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, avoidance of spatial reuse of channels, channel blacklisting, and channel hopping [3]. These unique characteristics introduce unique challenges in end-to-end delay analysis for process monitoring and control in WirelessHART networks.

In this paper, we address the open problem of end-to-end delay analysis for periodic real-time flows from sensors to actuators in a WirelessHART network. Specifically, we focus on the delay analysis for fixed priority scheduling where transmissions associated with each flow are scheduled based on the fixed priority of the flow. Fixed priority scheduling is the most commonly adopted real-time scheduling strategy in practice, e.g., in CPU scheduling and wired real-time networks such as Control-Area Networks (CANs). Our objective is to derive an upper bound of the end-to-end delay for each periodic flow. The end-to-end delay analysis 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, analytical delay bounds are highly desirable in process monitoring and control applications that require real-time performance guarantees. The end-to-end delay analysis 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, end-to-end delay analysis can be used to promptly decide whether some flow has to be removed or some rate has to be updated to meet deadlines.

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

• *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*

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 [4], 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.

The rest of the paper is organized as follows. Section 2 reviews related works. Section 3 presents the WirelessHART network model. Section 4 defines the scheduling problem. Section 5 presents the mapping and the end-to-end delay analysis. Section 6 shows how the delay bounds can be extended to a polynomial time method. Section 7 presents evaluation results. Section 8 concludes the paper.

2 Related Works

Real-time transmission scheduling in wireless networks has been widely studied in previous works [5]. However, very few of those are applicable to WirelessHART networks. Scheduling based on CSMA/CA protocols has been studied in [6]–[12]. In contrast, WirelessHART adopts a TDMA-based protocol to achieve predictable latency bounds. Although TDMA-based scheduling has been studied in [13]–[15], these works do not address multi-channel communication or multi-path routing.

The authors in [16] propose a schedulability analysis for 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]. But it is designed only for data collection through a routing tree using single channel. End-to-end delay bounds have been derived in [18] for real-time flows in WSNs. But this approach works only for cluster-tree model, and is based on single channel and time division cluster schedule. Considering the routing structure as a tree, the worst case delay of messages has been derived in [19] using sensor network calculus. It considers traffic only from the sensor nodes to the base station and there is no priority among the messages. The MAC protocol proposed in [20] assigns fixed priorities to messages and provides an upper bound on the queuing times of messages. However, this bound can help only to derive a necessary condition for schedulability. Thus, the afore-mentioned existing works are not applicable for sufficient schedulability analysis of the fixed priority real-time flows in a WirelessHART network that exploits the advantages of TDMA, multi-channel, and multi-path routing.

Since the standard was ratified in September 2007, transmission scheduling for WirelessHART networks has been investigated in recent works. Several papers have proposed scheduling algorithms for convergecast assuming simplified network models such as linear [21] and tree networks [22], [23]. For tree topology, they further assume that the depth of the tree is no greater than the number of channels. In contrast, we consider arbitrary network topologies without any constraint on route length. Moreover, we consider bidirectional real-time flows from sensors to the gateway and then to actuators, whereas these works only consider data collection at the gateway. Finally, these works do not consider real-time flows with different priorities and priority-based transmission scheduling, which are the focus of this paper.

Some recent works have considered WirelessHART networks of arbitrary topologies. These works focus on real-time scheduling [24], [25], routing [26], and rate selection [27] algorithms, or framework for schedule modeling [28]. Thus, none of these works addresses the end-to-end delay analysis. In contrast, we present an end-to-end delay analysis that is suitable for any fixed priority scheduling policy. Fixed priority scheduling is a widely adopted real-time scheduling policy in practice for both real-time CPU scheduling and wired real-time networks such as CANs. Instead of devising a new real-time transmission scheduling algorithm, the key contribution of our work is an efficient analysis for deriving the worst case delay bounds for real-time flows that are scheduled based on fixed priority. 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 WirelessHART Network Model

We consider 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 are able to send, receive, and route packets.

For process monitoring and control applications, the controllers are installed in the gateway. The sensor devices deliver their sensor data to the controllers at the gateway, and the control messages are then delivered to the actuators through the network. The unique features that make WirelessHART particularly suitable for industrial process monitoring and control are as follows [2], [3].

Centralized Management. The WirelessHART network is managed by a *centralized network manager* installed in the gateway. The network manager uses the network topology information to determine the routes. It creates the schedule of transmissions, and distributes the schedules among the devices. Experiences in industrial environments have shown daunting challenges in deploying large-scale WSANs. The limit on the network size for a WSAN makes the centralized management practical and desirable, and enhances the reliability and real-time performance. Large-scale networks can be organized using multiple gateways or as hierarchical networks that connect small WSANs through traditional resource-rich networks such as Ethernet and 802.11 networks.

Time Division Multiple Access (TDMA). Compared to CSMA/CA mechanism, TDMA protocols can provide predictable communication latencies making them an attractive approach for real-time communication. In WirelessHART networks, time is synchronized, and communication is TDMA-based. A time slot is 10 ms 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 (using the same channel) 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*, a subgraph of G, is a directed list of paths that connect two devices. In *graph routing*, the routers determine a packet's next hop following the route designated by the packet's graph ID. For end-to-end communication between a source and destination pair, the convention is to allocate one link for each en-route device starting from the source, followed by allocating a second dedicated slot on the same path to handle a retransmission, and then to allocate a third shared slot on a separate path to handle another retry. Doing so for every link and for each of its subsequent links on the routing graph requires allocation of a huge number of time slots for one packet since there are multiple paths between a source and destination pair. As a result, a huge number of allocated time slots will remain unused since only one route will be chosen based on the network condition. Such a wastage of bandwidth severely degrades the real-time schedulability of packets. To address this problem, and for the sake of real-time communication we rather assume a simplified version of this routing. We assume that the number of routes between every pair is a small constant (typically 1 or 2). Again, using shared slots makes the real-time schedulability analysis highly complicated due to the presence of collisions. In our model, we are concerned about collision-free schedule, and hence only consider dedicated time slots.

Spectrum Diversity. Spectrum diversity gives the network access to all 16 channels defined in IEEE $802.15.4$ physical layer 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, every transmission in a time slot happens on a different channel across the entire network to avoid transmission failure due to interference between concurrent transmissions. Thus, the maximum number of concurrent transmissions in the entire network at any slot cannot exceed the number of available channels. This design decision improves the reliability at the potential cost of reduced throughput. The potential loss in throughput is also mitigated due to the small size of network. The combination of spectrum and route diversity allows to handle the challenges of network dynamics in harsh and variable environments at the cost of redundant transmissions and scheduling complexity.

Each device is equipped with a half-duplex omni-directional 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.

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. The source and destination are characterized to be a sensor node and an actuator, respectively. 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 or some link on the route is broken, the packet can still be delivered through another route in Φ_j. Therefore, in a predetermined schedule, for a flow $\mathbb{F}_j$, time slots must be reserved for transmissions through each route in Φ_j for redundancy. That is, the schedule must be created such that a flow $\mathbb{F}_j$ can meet deadline through each route in Φ_j. 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, 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.

Each flow F_i, $1 \leq i \leq N$, 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 end-to-end delay analysis can be applied to any fixed priority assignment. In a *fixed priority scheduling policy*, at any time slot, among all ready transmissions and those not conflicting with the scheduled ones, the transmission that belongs to the highest priority flow is scheduled on an available channel.

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. In fixed priority scheduling, the created schedule can be mapped to superframes as follows. For any i and j such that $1 \leq i < j \leq n$, the schedule for flows $F_1, F_2, \cdots, F_i$ is repeated after their hyper-period. Therefore, the schedule for flows $F_1, F_2, \cdots, F_i$ can be assigned to a superframe of length (i.e., total time slots in the superframe) equal to their hyper-period. Similarly, the schedule for flows $F_i,\ F_{i+1}, \cdots, F_j$ is repeated after the hyper-period of first j flows (i.e., flows $F_1,\ F_2, \cdots, F_j$), and hence can be assigned to a superframe of length equal to that hyper-period. For example, when $D_i = P_i$, for each F_i, using rate monotonic scheduling, flows having the same period are assigned in the superframe of length equal to their period.

Transmissions are scheduled using m channels. The set of periodic 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 to be schedulable by $\mathbb{S}$ is indeed schedulable by $\mathbb{A}$. To determine the 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 set of real-time flows F and a global fixed priority algorithm $\mathbb{A}$, our objective is to decide the schedulability of F based on end-to-end delay analysis.

5 End-to-end Delay Analysis

In this section, we present an efficient 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. If every transmission is repeated χ times to handle retransmission, then every time slot is simply multiplied by χ in delay calculation. For simplicity of presentation we use the 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 important observations between these two domains.

Since spatial reuse of channels is avoided 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 and non-trivial problem. An important observation is that we must consider transmission conflicts in the delay analysis. Note that transmission conflict is a distinguishing feature of transmission scheduling in WirelessHART 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 a safe 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 [4].

5.1.3 Response Time Analysis for Multiprocessor CPU Scheduling

To make our paper self-contained, here we present the results of the state-of-the-art response time analysis for multiprocessor scheduling which was proposed by Guan et al. [4]. 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. Now, 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 is said to have *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.

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

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

When F_i has carry-in, 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 carry-in $\mu_i = \min\Big(\max\Big(\lambda - (P_i - R_i), 0\Big), C_i - 1\Big)$; $\lambda = \max(t - C_i, 0) \bmod P_i$; with R_i being the worst case response time of F_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 derived as follows.

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

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 WirelessHART 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 χ slots by an instance of F_i, if F_i has χ transmissions that involve node u or v. For example, in Figure 1(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. For example, in Figure 1(a), an instance of F_k has to be delayed at most by 5 slots since $Q(k,i) = 5$.

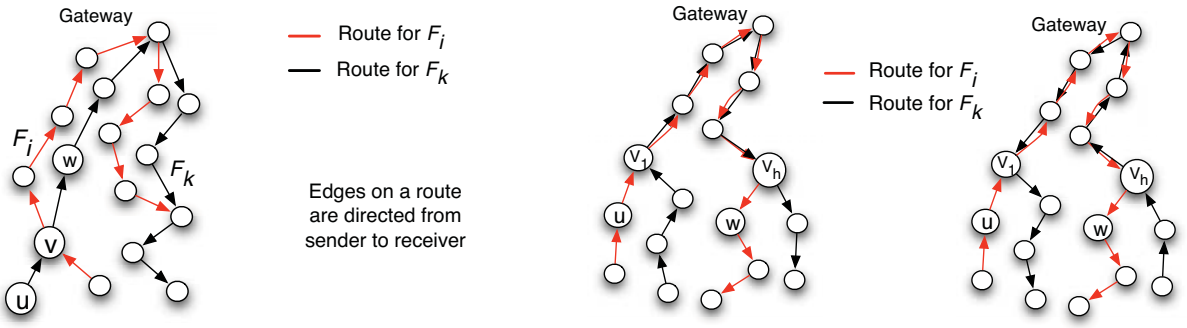
$Q(k,i)$ often overestimates the delay because when there is "too much" overlap between the routes of F_i and F_k, F_i will not necessarily cause "too much" delay to F_k. For example, in Figure 1(b), F_k can be delayed by an instance of F_i at most by 3 slots while $Q(k,i) = 8$. To obtain a more precise upper bound of the delay due to transmission conflicts, 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 1(b)). On an MCP between F_k and F_i, denoted by $M_j(k,i)$, F_k can be 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* $\delta_j(k,i)$ 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 $\delta_j(k,i) = h+1$. If only u or only w exists, then $\delta_j(k,i) = h$. If neither u nor v does exist, then $\delta_j(k,i) = h-1$. 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. Thus, Lemma 1 establishes a more precise upper bound $\Delta(k,i)$ of the delay that F_k can experience from an instance of F_i.

Lemma 1: Let $\delta'_j(k,i)$ 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

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

Proof: Let an MCP $M'_j(k,i)$ be $v_1 \to \cdots \to v_h$. Let there exist u and w such that the path $u \to v_1 \to \cdots \to v_h \to w$ is on F_i's route. Now, either $v_1 \to \cdots \to v_h$ or $v_h \to \cdots \to v_1$ must lie on F_k's route (Figure 1(b)). If $v_1 \to \cdots \to v_h$ 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 $u \to v_1 \to \cdots \to v_h \to w$. Similarly, if $v_h \to \cdots \to v_1$ 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 $u \to v_1 \to \cdots \to v_h \to w$. 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 $\delta'_j(k,i) - 3$ transmissions will not cause delay to F_k. But $Q(k,i)$ counts every transmission of F_i on $M'_j(k,i)$. Therefore, $Q(k,i) - \sum_{j=1}^{\sigma(k,i)} \big(\delta'_j(k,i) - 3\big)$ represents the bound $\Delta(k,i)$. □

According to Lemma 1, 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. Again, when $\delta'_j(k,i)$ is calculated for an $M'_j(k,i)$, we look for the next MCP only if $Q(k,i) - \delta'_j(k,i) \geq 4$.



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

(b) $Q(k,i) = 8$ but $\Delta(k,i) = 3$

Fig. 1. An example when F_k can be delayed by $F_i \in hp(F_k)$

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, an upper bound of the total delay that an instance of F_k can experience from flow F_i is

$$\left\lceil \frac{t}{P_i} \right\rceil \Delta(k,i)$$

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 follows.

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

5.3 Tighter Bound for Delay due to Transmission Conflicts

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 present how to avoid the pessimistic assumptions, and establish a tighter bound on the delay of a flow that occurs due to transmission conflict.

According to Equation 7, the upper bound $\Theta_k(t)$ of the delay due to transmission conflicts in a time interval of t slots assumes that

1) The lower priority flow F_k is delayed by every instance of the higher priority flow F_i that is released within the time interval of t slots, and
2) The lower priority flow F_k is delayed by $\Delta(k,i)$ time slots by every instance of the higher priority flow F_i.

In a real scheduling sequence, as we present in the next discussion, not every instance of a higher priority flow F_i can cause delay by $\Delta(k,i)$ time slots on F_k, thereby making the above assumptions highly pessimistic. The delay due to transmission conflicts plays a major role in the end-to-end delay of a flow. Overestimate in conflict delay may result in significant pessimism in the end-to-end delay analysis. In the rest of this subsection, we provide critical observations to avoid these pessimistic assumptions, and establish a more precise bound on conflict delay, that results in an improved schedulability test.

The pessimistic assumptions are due to the fact that the analysis for determining $\Theta_k(t)$ in the previous subsection does not exclude F_k's transmissions that have already been scheduled into the consideration for calculating the future delay on F_k. In other words, some transmissions of F_k that have already been scheduled are still considered to be subject to delay by F_i, which clearly should not be the case.

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 more subject to delay by the subsequent instances of F_i. For example, in Figure 1(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 2 and Lemma 3.

Lemma 2: 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. □

Lemma 3 now determines an upper bound of the total conflict delay caused by multiple instances of F_i when each transmission of F_k is delayed by at most one instance of F_i.

Lemma 3: Let $p \geq 1$ instances of a higher priority flow F_i cause conflict delay on F_k such that each transmission of F_k is delayed by at most one instance of F_i. Then the total delay caused on F_k by these p instances is at most $\Delta(k,i)$.

Proof: The proof follows directly when $p = 1$. Let us consider the case when $p \geq 2$. 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 delay on F_k. 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. Given each transmission of F_k is delayed by at most one instance of F_i, subsets Γ_1 and Γ_2 must be disjoint. Similarly, for any p, the subsets $\Gamma_1, \Gamma_2, \cdots, \Gamma_p$ are 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 $\Gamma_1, \Gamma_2, \cdots, \Gamma_p$ is at most $\Delta(k,i)$. That is, the total conflict delay on F_k caused by p, $p \geq 2$, instances of F_i is at most $\Delta(k,i)$. □

Based on Lemma 2 and 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 2, 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

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

Proof: There are at most $\lceil \frac{t}{P_i} \rceil$ instances of F_i in a time interval of t slots. If no transmission of F_k is delayed by more than one instance of F_i, by Lemma 3, the total delay caused on F_k by all instances of F_i is at most $\Delta(k,i)$.

When some transmission of F_k is delayed by more than one instance of F_i, 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. By Lemma 2, 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

$$\begin{aligned}&\Delta(k,i) + Z(k,i)\\ &\leq \Delta(k,i) + \Big(\left\lfloor \frac{t}{P_i} \right\rfloor - 1 \Big).\delta(k,i) + \min\Big(\delta(k,i), t \bmod P_i\Big)\end{aligned}$$

□

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.

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

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, thereby establishing an upper bound of its worst case end-to-end delay due to both channel contention and transmission conflicts. This is done for every flow in decreasing order of priority starting with the highest priority flow as explained below.

For flow F_k, we use $R_k^{\text{ch,con}}$ to denote an upper bound of the worst case end-to-end delay considering delays both due to **ch**annel contention and due to **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 WirelessHART 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 flow F_i has no carry-in, its workload $W_k^{\text{nc}}(F_i, x)$ 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 [4]. 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 pseudo upper bound R_k^{ch} is computed, we incorporate the upper bound of the delay due to conflicts 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 recursive equation will give us 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 that solves Equation 11 starting with $x = C_k$. Let $R_k^{\text{ch,con}}$ be the minimal value of y that solves Equation 12 starting with $y = R_k^{\text{ch}}$. 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.

6 Polynomial-Time End-to-End Delay Analysis

The end-to-end delay analysis presented in the previous section calculates the end-to-end delay bound of each flow in pseudo polynomial time. In this section, we extend the analysis to a polynomial time method. While the polynomial time method 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.

To derive a polynomial-time analysis, we have to calculate both the channel contention delay and the transmission conflict delay in polynomial time. Using the same mapping presented in Section 5 of transmission scheduling in a WirelessHART network to the global multiprocessor scheduling, we can also use the polynomial time response time analysis for global multiprocessor scheduling proposed in [29] 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.

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

Thus, similar to 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 follows.

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

The above analysis indicates that it does not require calculating the worst case end-to-end delays of the flows in order of their priorities. Since the lower priority flows have the higher chances of missing deadlines, the above analysis, unlike the pseudo polynomial time one, allows us to calculate the end-to-end delays of the lower priority flows first, thereby getting a quicker decision on the schedulability of the flows.

7 Evaluation

We evaluate our end-to-end delay analysis through simulations based on both random topologies and the real topologies of a wireless sensor network testbed. There is no baseline to compare the performance of our analysis which, to the best of our knowledge, is the first end-to-end 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.

Metrics. We evaluate our analysis in terms of acceptance ratio and pessimism ratio. *Acceptance ratio* is defined as the proportion of the number of test cases deemed to be 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 upper bound to its maximum end-to-end delay observed in simulation.

We implement both the pseudo polynomial time analysis and the polynomial time analysis. In evaluating the pseudo polynomial time analysis, we first evaluate it without considering the improved conflict delay bound derived in Subsection 5.3. Then we evaluate it by considering the improved conflict delay bound derived in Subsection 5.3. This helps us observe that the conflict delay bound derived in Equation 8 is significantly tighter than that derived in Equation 7. Specifically, in the figures in this section, the analyses are marked 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. Specifically, it calculates the delay bounds using Equation 16.

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. The number of routes between every source and destination is set to 1, and this is the most reliable route connecting a source to a 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 flow with the shortest deadline being assigned the highest priority. If there is a tie, then the flow with the smallest ID is assigned the highest priority. The algorithms have been implemented in C and the tests have been performed on a MacBook Pro laptop.

7.2 Simulations with Testbed Topologies

Our wireless sensor network testbed is deployed in two buildings (Bryan Hall and Jolley Hall) of Washington University in St Louis [30]. The testbed consists of 74 TelosB motes each equipped with Chipcon CC2420 radios which are compliant with the IEEE 802.15.4 standard. Note that the physical layer in WirelessHART is also based on IEEE 802.15.4. Setting the same transmission power at every node, every node broadcasts 50 packets while its neighbors record the sequence numbers of the packets they receive. After a node completes sending its 50 packets, the next sending node is selected in a round-robin fashion. 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. Figure 2 shows the network topology with transmission power of -1 dBm (embedded on the floor plans of two buildings).

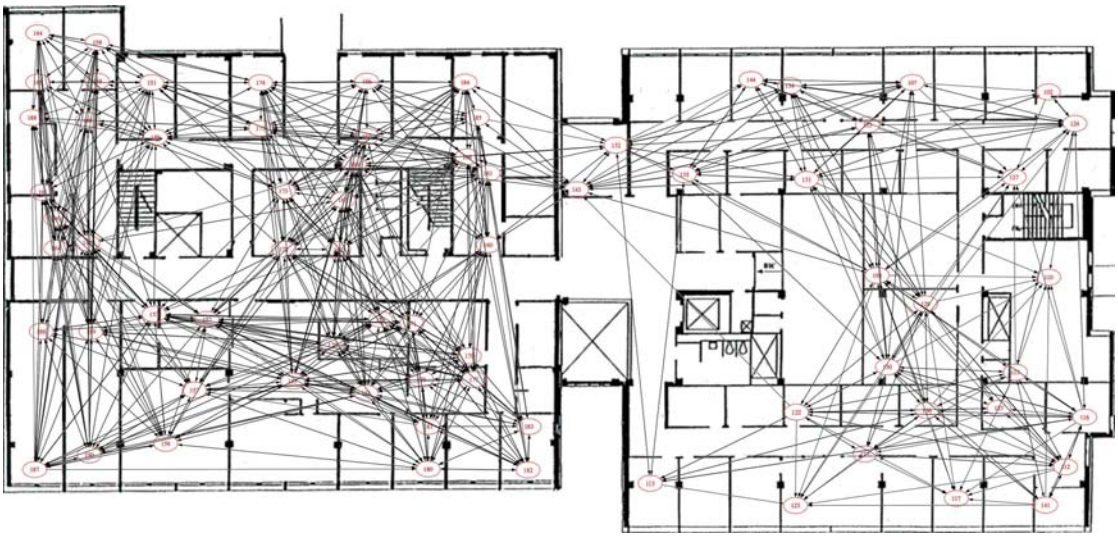
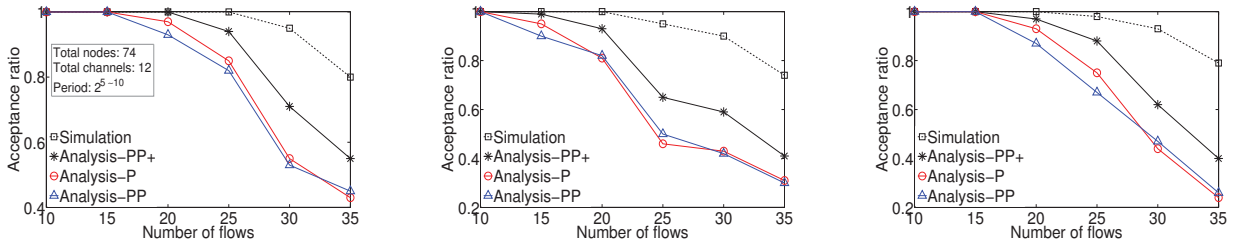


Fig. 2. Testbed topology (at transmission power of -1 dBm)



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

Fig. 3. Schedulability on testbed topology

We collected the topologies at 3 different transmission power levels (-1 dBm, -3 dBm, -5 dBm).

We generate different number of flows by randomly selecting the sources and destinations considering the topologies. The periods of the flows are randomly generated in the range $2^{5\sim10}$ time slots. We generate 100 test cases and simulate them on topologies at different transmission power levels. The acceptance ratios under Analysis-PP, Analysis-PP+, and Analysis-P are shown in Figure 3 as interpreted below. Every test case is simulated by scheduling all the instances of the flows released within their hyper-period. In the figure, "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.

Figure 3(a) shows the acceptance ratios of our delay analysis methods using the topology with transmission power of -1 dBm. According to the figure, when the number of flows is less than 25, Analysis-PP+ has an acceptance ratio of 1.0, which means that all tests cases that are indeed schedulable are also determined as schedulable by our analysis. When the number of flows is 30, the value of "Simulation" is 0.99 meaning that 99% test cases are indeed schedulable, while the acceptance ratio of Analysis-PP+ is 0.93. Thus, approximately 94% of schedulable cases are deemed schedulable by Analysis-PP+ which indicates that the analysis is highly efficient. After that, the acceptance ratios of our analysis decreases with the increase in the total number of flows. However, the difference between its acceptance ratio and the value of "Simulation" always remains strictly less than 0.25.

Besides Analysis-PP+, Figure 3(a) also plots the acceptance ratios under Analysis-PP and Analysis-P. As the figure indicates, the acceptance ratio of Analysis-PP+ is always a lot higher than that of Analysis-PP when the number of flows is greater than 15. For example, for 30 flows where 95% test cases are actually schedulable (as tested through simulations), Analysis-PP+ can determine 71% test cases as schedulable while Analysis-PP determines only 53% test cases as schedulable. This happens because the delay bounds calculated in Analysis-PP+ are significantly tighter than those calculated in Analysis-PP. Analysis-P which determines looser (compared to Analysis-PP+) delay bounds but calculates the bounds in polynomial time is highly competitive against Analysis-PP. Specifically, except the case when the number of flows is 35, the acceptance ratio of Analysis-P is always no less than that of Analysis-PP. This happens because Analysis-P determines the conflict delay based on the

improvement made in Equation 8 (by extending it to a polynomial time method).

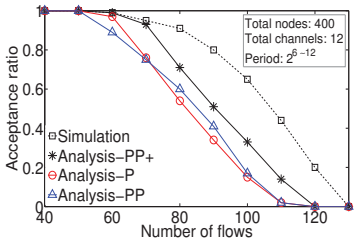


Fig. 4. Schedulability on random topology

Figure 3(b) shows the similar results for the topology with transmission power of -3 dBm. The figure indicates that when the number of flows is no greater than 20, the value of "Simulation" is larger than the acceptance ratio of Analysis-PP+ at most by 0.16. After that, the difference between the acceptance ratio of Analysis-PP+ and the value of "Simulation" increases, but always remains less than 0.38. The acceptance ratio of Analysis-PP+ is always higher than that of Analysis-PP, and Analysis-P is also highly competitive against Analysis-PP. For example, for 30 flows in this topology, the acceptance ratios of our Analysis-PP+, Analysis-PP, and Analysis-P are 0.59, 0.40, and 0.43, respectively. For the topology with transmission power of -5dBm, Figure 3(c) also shows that Analysis-PP+ is a highly effective delay analysis method.

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 (i.e., without considering the tighter conflict delay bound).

7.3 Simulations with Random Topologies

We test the scalability of our algorithms 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 of the flows are considered harmonic and are randomly generated in the range $2^{6\sim12}$ time slots.

The acceptance ratios of our analyses in 400-node network are shown in Figure 4. According to the figure, 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. The figure also indicates that the acceptance ratio of Analysis-PP+ is always higher than that of Analysis-PP, and Analysis-P is highly competitive against Analysis-PP. For example, for 100 flows in this topology, the acceptance ratios of our Analysis-PP+, Analysis-PP, and Analysis-P are 0.33, 0.17, and 0.15, respectively.

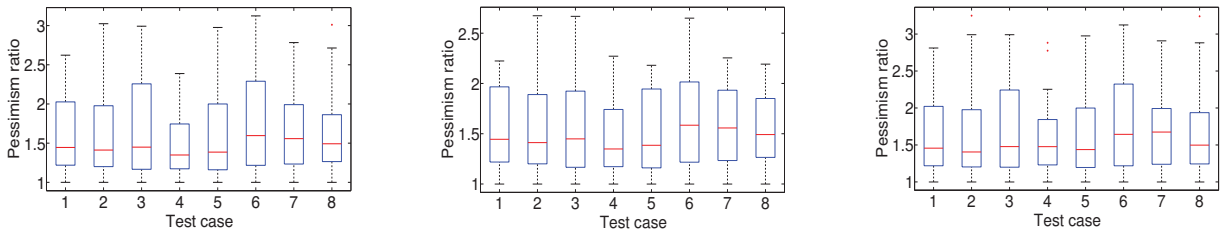
Among 100 test cases, each consisting of 70 flows in the 400-node network, we randomly select 8 test cases that are schedulable under all 3 analyses, and Figure 5 plots the pessimism ratios in Analysis-PP, Analysis-PP+, and Analysis-P. Figures 5(a) and 5(b) indicate that the 75th percentile of the pessimism ratios is less than 2.0 in all 8 test cases under Analysis-PP+, while those under Analysis-PP are greater than 2 for test case 3 and 6. Figures 5(a) and 5(c) indicate that the statistics in pessimism ratios under Analysis-PP and Analysis-P do not vary a lot. The pessimism ratios indicate that the end-to-end delay bounds calculated in Analysis-PP+ are overestimated by a factor of at most 2 in most cases. They also indicate that that the end-to-end delay bounds calculated in Analysis-PP+ are smaller than those calculated in Analysis-PP since the latter uses a pessimistic bound of conflict delay.

The results indicate that our analysis is effective even for very large networks with large number of flows. The pessimism ratios under different sized networks indicate that our estimated bounds are reasonably tight. 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

An efficient end-to-end delay analysis is required, both at design time and for online admission control, to decide the schedulability of real-time data flows in a WirelessHART network. Compared to extensive testing and simulations, analytical delay bounds are highly desirable in process monitoring and control applications that require real-time performance guarantees. A delay analysis can also be used for adjusting workload in response to network dynamics.

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 a pseudo polynomial time end-to-end delay analysis to determine the schedulability of real-time data flows in WirelessHART networks. Furthermore, we have extended the analysis to a polynomial time method that can be used to compute a looser delay bound more efficiently. Simulation studies based



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

Fig. 5. Pessimism ratio

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.

ACKNOWLEDGEMENTS

This research was supported by NSF under grants CNS-1144552 (NeTS), CNS-1035773 (CPS), CNS-1017701 (NeTS), CNS-0708460 (CRI), a Microsoft Research New Faculty Fellowship, and a Sloan-Kettering Center grant.

REFERENCES

[1] J. Song, A. K. Mok, D. Chen, and M. Nixon, "Challenges of wireless control in process industry," in *Workshop on Research Directions for Security and Networking in Critical Real-Time and Embedded Systems*, 2006.

[2] "WirelessHART," 2007, http://www.hartcomm2.org.

[3] D. Chen, M. Nixon, and A. Mok, *WirelessHART^{TM}Real-Time Mesh Network for Industrial Automation*. Springer, 2010.

[4] N. Guan, M. Stigge, W. Yi, and G. Yu, "New response time bounds for fixed priority multiprocessor scheduling," in *RTSS '09: Proceedings of the 30th IEEE International Real-Time Systems Symposium*, 2009, pp. 387–397.

[5] 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, July 2003.

[6] H. Li, P. Shenoy, and K. Ramamritham, "Scheduling messages with deadlines in multi-hop real-time sensor networks," in *RTAS '05: Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2005, pp. 415–425.

[7] 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: the 6th European Conference on Wireless Sensor Networks*, 2009.

[8] 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: Proceedings of the 7th ACM international conference on Mobile computing and networking*, 2001, pp. 200–209.

[9] 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: Proceedings of the 8th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2002, pp. 55–66.

[10] K. Karenos, V. Kalogeraki, and S. V. Krishnamurthy, "A rate control framework for supporting multiple classes of traffic in sensor networks," in *RTSS '05: Proceedings of the 26th IEEE International Real-Time Systems Symposium*, 2005, pp. 287–297.

[11] K. Karenos and V. Kalogeraki, "Real-time traffic management in sensor networks," in *RTSS '06: Proceedings of the 27th IEEE International Real-Time Systems Symposium*, 2006, pp. 422–434.

[12] 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 Systems*, vol. 37, no. 3, 2007.

[13] 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: Proceedings of the 24th IEEE International Real-Time Systems Symposium*, 2003, pp. 298 – 307.

[14] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "JiTS: Just-in-time scheduling for real-time sensor data dissemination," in *PERCOM '06: Proceedings of the 4th IEEE International Conference on Pervasive Computing and Communications*, 2006, pp. 42–46.

[15] Y. Gu, T. He, M. Lin, and J. Xu, "Spatiotemporal delay control for low-duty-cycle sensor networks," in *RTSS '09: Proceedings of the 30th IEEE International Real-Time Systems Symposium*, 2009, pp. 127–137.

[16] T. F. Abdelzaher, S. Prabh, and R. Kiran, "On real-time capacity limits of multihop wireless sensor networks," in *RTSS '04: Proceedings of the 25th IEEE International Real-Time Systems Symposium*, 2004, pp. 359–370.

[17] O. Chipara, C. Lu, and G.-C. Roman, "Real-time query scheduling for wireless sensor networks," in *RTSS '07: Proceedings of the 28th IEEE International Real-Time Systems Symposium*, 2007, pp. 389–399.

[18] P. Jurcík, R. Severino, A. Koubâa, M. Alves, and E. Tovar, "Real-time communications over cluster-tree sensor networks with mobile sink behaviour," in *RTCSA '08: Proceedings of the 14th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2008.

[19] J. B. Schmitt and U. Roedig, "Sensor network calculus A framework for worst case analysis," in *DCOSS '05: Proceedings of the IEEE international Conference on Distributed Computing in Sensor Systems*, 2005.

[20] N. Pereira, B. Andersson, E. Tovar, and A. Rowe, "Static-priority scheduling over wireless networks with multiple broadcast domains," in *RTSS '07: Proceedings of the 28th IEEE International Real-Time Systems Symposium*, 2007, pp. 447–458.

[21] H. Zhang, P. Soldati, and M. Johansson, "Optimal link scheduling and channel assignment for convergecast in linear WirelessHART networks," in *WIOPT '09: Proceedings of the 7th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, 2009.

[22] P. Soldati, H. Zhang, and M. Johansson, "Deadline-constrained transmission scheduling and data evacuation in WirelessHART networks," in *ECC '09: Proceedings of the European Control Conference*, 2009.

[23] H. Zhang, F. Osterlind, P. Soldati, T. Voigt, and M. Johansson, "Rapid convergecast on commodity hardware: Performance limits and optimal policies," in *SECON '10: Proceedings of the 7th IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks*, 2010, pp. 1–9.

[24] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for WirelessHART networks," in *RTSS '10: Proceedings of the 31st IEEE International Real-Time Systems Symposium*, 2010, pp. 150 – 159.

[25] ——, "Priority assignment for real-time flows in WirelessHART networks," in *ECRTS '11: Proceedings of the 23rd Euromicro Conference on Real-Time Systems*, 2011, pp. 35–44.

[26] 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: Proceedings of the 17th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2011, pp. 3–12.

[27] 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: Proceedings of the 18th IEEE Real-Time and Embedded Technology and Applications Symposium (to appear)*, 2012, pp. 231–240.

[28] R. Alur, A. D'Innocenzo, K. Johansson, G. Pappas, and G. Weiss, "Modeling and analysis of multi-hop control network," in *RTAS '09: Proceedings of the 15th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2009, pp. 223–232.

[29] 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.

[30] WSN testbed, http://mobilab.wustl.edu/testbed.