

End-to-End Delay Analysis for Fixed Priority Scheduling in WirelessHART Networks

Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen
Department of Computer Science and Engineering
Washington University in St. Louis
{saifullaha, yx2, lu, chen}@cse.wustl.edu

Abstract—The WirelessHART standard has been specifically designed for real-time communication between sensor and actuator devices for industrial process monitoring and control. End-to-end communication delay analysis for WirelessHART networks is required for acceptance test of real-time data flows from sensors to actuators and for 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 end-to-end delay analysis of the real-time flows that are scheduled using a fixed priority scheduling policy in a WirelessHART network. Simulations based on both random topologies and real network topologies of a physical testbed demonstrate the efficacy of our end-to-end delay analysis in terms of acceptance ratio under various fixed priority scheduling policies.

I. INTRODUCTION

Wireless Sensor-Actuator Networks (WSANs) impose stringent end-to-end delay requirements on data communication for industrial process monitoring and control [1]. A feedback control loop implemented in a WSAN periodically sends sensor data from the sensor devices to a controller and, then, delivers the control input data to the actuators within an end-to-end deadline. 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.

WirelessHART [2] is an open WSAN standard which has been specifically designed for monitoring and control applications in 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 tackle 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 real-time flow are scheduled based on fixed priorities assigned to these flows. 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 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 specifically for fixed priority transmission scheduling in WirelessHART networks. Our analysis establishes a safe and tight upper bound of the end-to-end delay of every real-time periodic data flow.

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 48 TelosB motes. The simulation results show that our estimated delay bounds are reasonably tight and that our end-to-end delay analysis is highly effective in terms of acceptance ratio of real-time flows under various fixed priority scheduling policies.

The rest of the paper is organized as follows. We review the related works in Section II. Section III presents the

WirelessHART network model. The scheduling problem is defined in Section IV. Section V presents the mapping and the end-to-end delay analysis. The simulation results are presented in Section VI. Finally, we conclude in Section VII.

II. 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, we can conclude that the afore-mentioned 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, the transmission scheduling for WirelessHART networks has been investigated in some 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 to the gateway. Finally, these previous works do not consider real-time flows with different priorities and priority-based transmission scheduling, which are the focus of this paper.

Transmission scheduling of real-time flows for arbitrary WirelessHART network topologies was addressed in [24]. It presents real-time scheduling algorithms based on branch-and-bound and heuristics that do not support fixed priority scheduling. In contrast, we focus on fixed priority scheduling and 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.

III. WIRELESSHART NETWORK MODEL

We consider the WirelessHART network model followed in [24]. A *WirelessHART network* consists of a set of field devices, a gateway, and a centralized network manager. 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. Schedulability analysis and transmission scheduling of the network are performed centrally at the *network manager* connected to the gateway which uses the network topology information in combination with the communication requirements of the devices and applications. The network manager, then, distributes the schedules among the devices. The unique features that make WirelessHART particularly suitable for industrial process monitoring and control are as follows.

Limiting Network Size. Experiences in process industries have shown the daunting challenges in deploying large-scale WSNs. Typically, 80-100 field devices comprise a WirelessHART network with one gateway. The limit on the network size for a WSN makes the centralized management practical and desirable, and enhances the reliability and real-time performance. Large-scale networks can be organized by using multiple gateways or as hierarchical networks that connect small WSNs through traditional resource-rich networks such as Ethernet and 802.11 networks.

Time Division Multiple Access (TDMA). In contrast with CSMA/CA MAC protocols, TDMA protocols provide predictable communication latencies, thereby making themselves an attractive approach for real-time communication. In WirelessHART networks, time is synchronized and slotted. The length of a time slot allows exactly one transmission and its associated acknowledgement between a device pair.

Route and Spectrum Diversity. Spatial diversity of routes allows messages to be routed through multiple paths in order to mitigate physical obstacles, broken links, and interference. 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. The combination of spectrum and

route diversity allows a packet to be transmitted multiple times, over different channels over different paths, thereby handling the challenges of network dynamics in harsh and variable environments at the cost of redundant transmissions and scheduling complexity.

Handling Internal Interference. Due to difficulty in detecting interference between nodes and the variability of interference patterns, WirelessHART allows only one transmission in each channel in a time slot across the entire network, thereby avoiding the spatial reuse of channels [3]. Thus, the total number of concurrent transmissions in the entire network at any slot is no greater than the number of available channels [3]. This design decision effectively avoids transmission failure due to interference between concurrent transmissions, and 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.

With the above features, WirelessHART forms a mesh network that can be modeled as a graph $G = (V, E)$, where the node set V represents the network devices and E is the set of edges between these devices. That is, the set V consists of the gateway and the field devices. An edge $e = (u, v)$ is in E if and only if devices $u \in V$ and $v \in V$ can reliably communicate with each other. A transmission involves exactly one pair of devices connected by an edge. For a transmission, denoted by \vec{uv} , that happens along edge (u, v) , device u is designated as the *sender* and device v the *receiver*. All network devices are able to send and receive packets and to route packets on behalf of others.

A device 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, cannot be 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.

IV. 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. Therefore, from now onward the term ‘flow’ will refer to an individual flow through a route. We denote this set of flows by $F = \{F_1, F_2, \dots, F_N\}$. 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 , the number of transmissions required to deliver a packet from $Source_i$ to $Destination_i$ through its route ϕ_i is denoted by C_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, $hp(F_i) = \{F_1, F_2, \dots, F_{i-1}\}$. In practice, priorities may be assigned based on deadlines, rates, or the criticality of the real-time flows. 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. 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.

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.

V. 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.

A. Analysis of Delays due to Channel Contention

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.

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].

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 is due to 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 busy period 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 $W_k^{nc}(F_i, t)$ of its workload in a $BP(k, t)$, and an upper bound $I_k^{nc}(F_i, t)$ of the delay it can cause to F_k are as follows:

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

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

When F_i has carry-in, an upper bound $W_k^{ci}(F_i, t)$ of its workload in a $BP(k, t)$, and an upper bound $I_k^{ci}(F_i, t)$ of the delay that it can cause to F_k are as follows:

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

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

where carry-in $\mu_i = \mathbf{min}\left(\mathbf{max}\left(\lambda - (P_i - R_i), 0\right), C_i - 1\right)$; $\lambda = \mathbf{max}(t - C_i, 0) \bmod P_i$; 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.

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

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

B. Analysis of Delays due to 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 \vec{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 \vec{uv} or \vec{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 $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_h$, where $v_l \neq v_q$ for $l \neq q$ (where $1 \leq l, q \leq h$), on F_i 's route such that $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_h$ or $v_h \rightarrow v_{h-1} \rightarrow \dots \rightarrow v_1$ 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 $M_j'(k, i) = v_1 \rightarrow \dots \rightarrow v_h$, if there exist $u, w \in V$ such that $u \rightarrow v_1 \rightarrow \dots \rightarrow v_h \rightarrow w$ is also

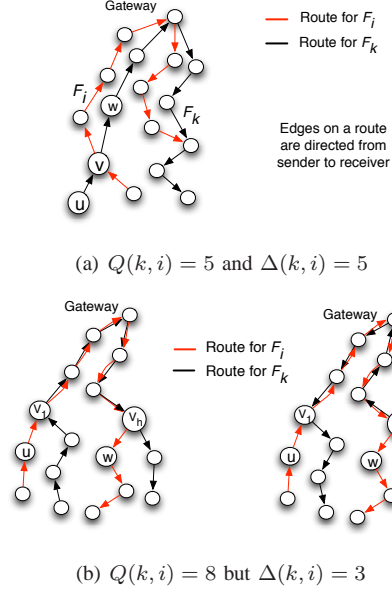


Fig. 1. An example when F_k can be delayed by $F_i \in hp(F_k)$

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., $\vec{uv}_1, \vec{v_1v_2}, \dots, \vec{v_hw}$), 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 σ MCPs between F_k and F_i each with length at least 4, then

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

Proof: Let an MCP $M_j'(k, i)$ be $v_1 \rightarrow \dots \rightarrow v_h$. Let there exist u and w such that the path $u \rightarrow v_1 \rightarrow \dots \rightarrow v_h \rightarrow w$ is on F_k 's route. Now, either $v_1 \rightarrow \dots \rightarrow v_h$ or $v_h \rightarrow \dots \rightarrow v_1$ must lie on F_k 's route (Figure 1(b)). If $v_1 \rightarrow \dots \rightarrow v_h$ is on F_k 's route, then a transmission $v_l v_{l+1}$, $1 \leq l < h$, of F_k on this path shares node with at most 3 transmissions of F_i on $u \rightarrow v_1 \rightarrow \dots \rightarrow v_h \rightarrow w$. Similarly, if $v_h \rightarrow \dots \rightarrow v_1$ is on F_k 's route, then a transmission $v_l v_{l-1}$, $1 < l \leq h$, of F_k on this path shares node with at most 3 transmissions of F_i on $u \rightarrow v_1 \rightarrow \dots \rightarrow v_h \rightarrow w$. 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 $\delta_j'(k, i) - 3$ transmissions will not cause delay to F_k . But $Q(k, i)$ counts every transmission of F_i on $M_j'(k, i)$. Therefore, $Q(k, i) - \sum_{j=1}^{\sigma} (\delta_j'(k, i) - 3)$ 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 $Q(k, i) - \delta'_j(k, i) \geq 4$.

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, $\lceil \frac{t}{P_i} \rceil \Delta(k, i)$ is an upper bound of the total delay that an instance of F_k can experience from flow F_i . Let $\Theta_k(t)$ be an upper bound of the total delay an instance of F_k can experience from all higher priority flows during a time interval of t slots. The bound $\Theta_k(t)$ is calculated as follows.

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

C. Analysis of End-to-End Delays

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 $R_k^{ch,con}$ to denote an upper bound of the worst case end-to-end delay considering delays both due to **channel contention** and due to **conflicts** between flows. We use the following two steps to estimate $R_k^{ch,con}$ for every flow $F_k \in F$ in decreasing order of priority starting with the highest priority flow.

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 $R_k^{ch,con}$ 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 V-A, 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 $W_k^{nc}(F_i, x)$ 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 $W_k^{nc}(F_i, x)$ 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,

$$W_k^{nc}(F_i, x) = \left\lfloor \frac{x}{P_i} \right\rfloor \cdot C_i + \mathbf{min}(x \bmod P_i, C_i) \quad (8)$$

Now $I_k^{nc}(F_i, x)$ 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 $W_k^{ci}(F_i, x)$ during a BP(k, x) depends on its worst case response time R_i . Equation 3 also indicates that $W_k^{ci}(F_i, x)$ 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, $R_i^{ch,con} \geq R_i$. Therefore, if we replace R_i with $R_i^{ch,con}$ in Equation 3, $W_k^{ci}(F_i, x)$ is guaranteed to be an upper bound of F_i 's workload during a BP(k, x). Thus,

$$W_k^{ci}(F_i, x) = \left\lfloor \frac{\mathbf{max}(x - C_i, 0)}{P_i} \right\rfloor \cdot C_i + C_i + \mu_i \quad (9)$$

where $\mu_i = \mathbf{min}\left(\mathbf{max}\left(\lambda - (P_i - R_i^{ch,con}), 0\right), C_i - 1\right)$ and $\lambda = \mathbf{max}(x - C_i, 0) \bmod P_i$. Similarly, $I_k^{ci}(F_i, x)$ 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 $I_k^{nc}(F_i, x)$ and $I_k^{ci}(F_i, x)$ 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 10.

$$x = \left\lfloor \frac{\Omega_k(x)}{m} \right\rfloor + C_k \quad (10)$$

Equation 10 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*”.

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 $R_k^{ch,con}$. 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 7. 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 $R_k^{ch,con}$ for F_k that includes both types of delay:

$$y = R_k^{ch} + \Theta_k(y) \quad (11)$$

Equation 11 is solved using an iterative fixed-point algorithm starting with $y = R_k^{ch}$. Like Step 1, this algorithm also either terminates at some fixed-point $y^* \leq D_k$ that is considered as the bound $R_k^{ch,con}$ or terminates with an “*unschedulable*” decision when $y > D_k$. Thus, termination of the algorithm is guaranteed.

Theorem 2: For every flow $F_k \in F$, let R_k^{ch} be the minimal value of x that solves Equation 10 starting with $x = C_k$. Let $R_k^{ch,con}$ be the minimal value of y that solves Equation 11 starting with $y = R_k^{ch}$. Then $R_k^{ch,con}$ 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, \dots, 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 10 and 11 yield $R_1^{ch,con} = C_1$, 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 11 holds for $k = 1$.

Now let the upper bound calculated using Equation 11 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 $R_{k+1}^{ch,con}$ in Step 2, we initialize y (in Equation 11) to R_{k+1}^{ch} . Note that R_{k+1}^{ch} is computed in Step 1 (before Step 2) for flow F_{k+1} . In Step 1, R_{k+1}^{ch} is computed considering upper bounds $R_h^{ch,con}$ 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 10 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 11 adds this delay to R_{k+1}^{ch} and establishes the recursive equation for y . Therefore, the minimal solution of y , i.e., $R_{k+1}^{ch,con}$ 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 $R_i^{ch,con}$, for $i = 1, 2, \dots, N$ (in decreasing order of priority level), and decides the flow set to be schedulable if, for every $F_i \in F$, $R_i^{ch,con} \leq D_i$. According to Equations 10 and 11, $R_i^{ch,con}$ is 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 2.

VI. EVALUATION

We evaluate our end-to-end delay analysis through simulations based on both random topologies and the real topology 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 fixed-priority scheduling in WirelessHART networks. We compare the performance of our analysis with that of simulations.

Priority assignment policies. The effectiveness of the proposed end-to-end delay analysis has been demonstrated with the following fixed priority assignment policies. (a) *Deadline Monotonic (DM)*: DM assigns priorities to flows according to their relative deadlines; the flow with the shortest deadline being assigned the highest priority. (b) *Proportional Deadline monotonic (PD)*: PD assigns priorities to flows based on *relative subdeadline* defined for a flow as its relative deadline divided by the total number of transmissions along its route. In each of these policies, if there is a tie, then the flow with the smallest ID is assigned the highest priority among those requesting the same priority.

Metrics. We evaluate our analysis in terms of the following metrics. (a) *Acceptance ratio*: This is defined as the proportion of the number of test cases deemed to be schedulable to the total number of test cases. (b) *Pessimism ratio*: For a flow, this metric is defined as the proportion of the analyzed theoretical upper bound to its maximum end-to-end delay observed in simulations.

A. Simulation Setup

A fraction (θ) of field devices is considered as sources and destinations. The sets of sources and destinations are disjoint. The node with the highest number of neighbors is designated as the gateway. The *reliability* of a link is represented by the *packet reception ratio (PRR)* along it. The most reliable route connecting a source to a destination is determined. For additional routes, we choose the next most reliable route that excludes the links of any existing route between the same source and destination.

n :	Number of nodes in the network
m :	Number of channels
ρ :	Edge-density of the network
θ :	Fraction of total nodes which are sources and destinations
γ :	Number of routes between every source and destination
P_{\sim} :	Period range
α :	Deadline parameter (i.e., route length \leq deadline \leq α *period)
β :	Rate factor (i.e., new rate = β *old rate)

TABLE I
NOTATIONS

The period P_i of a flow F_i is generated randomly in a given range denoted by $P_{\sim} = 2^{a \sim b}$ time slots, $a \leq b$. A parameter called *rate factor* (β) is used to tune the rate (i.e. $1/P_i$) of every flow F_i as follows: *new rate* = β **rate*. The value of the relative deadline D_i of every flow F_i having period P_i is randomly generated in the range between C_i and $\alpha * P_i$ slots, for $0 < \alpha \leq 1$, with C_i being the number of transmissions along its route. In every figure, we show the parameter setups

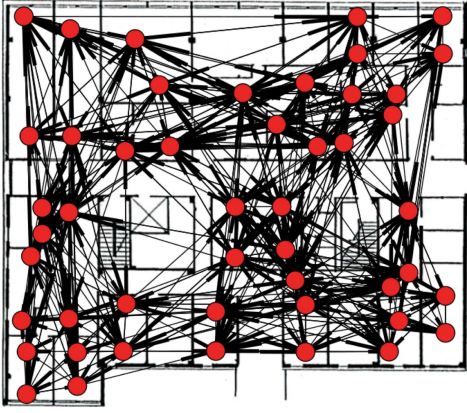


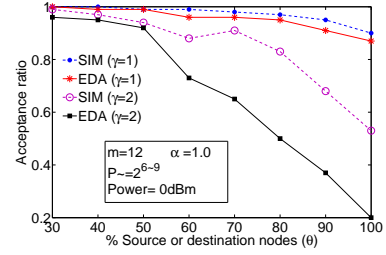
Fig. 2. The testbed topology with a transmission power of 0 dBm

of the corresponding experiment. The algorithms have been implemented in C and the tests have been performed on a MacBook Pro laptop. The notations used in this section are summarized in Table I.

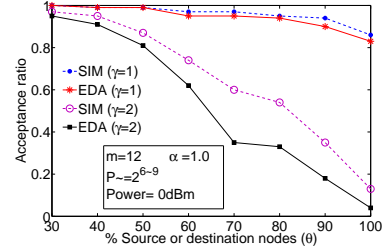
B. Simulations with Testbed Topologies

Our wireless sensor network testbed is deployed in Bryan Hall of Washington University in St Louis [25]. The testbed consists of 48 TelosB motes each equipped with Chipcon CC2420 radios which are compliant with the IEEE 802.15.4 standard. At the default transmission power level (0 dBm), 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. Figure 2 shows the network topology with transmission power of 0 dBm. Every link with a higher than 80% PRR is considered a reliable link and drawn in Figure 2 (embedded on the floor plan of the building). Considering this topology, we have tested our analysis as explained below.

Varying sources and destinations. The flows are generated in the network by randomly selecting the sources and destinations. In the first set of simulations, we generate different number of flows by varying θ where $\frac{\theta}{2}\%$ of the total nodes are sources while another $\frac{\theta}{2}\%$ are destinations. The periods of the flows are randomly generated in the range $P_{\sim} = 2^{6-9}$ time slots. The number of channels m is set to 12. The deadlines of the flows are assigned by setting $\alpha = 1.0$. For every θ , we generate 100 test cases and show the performances under varying θ in Figure 3. In the figures, “EDA” indicates the acceptance ratio of our analysis. To compare the acceptance ratio of our analysis, every test case is simulated by scheduling all the instances of the flows released within their hyperperiod. “SIM” denotes the fraction of test cases that have no deadline misses in the simulations. For DM priority assignment, Figure 3(a) indicates that our analysis can determine 87% test cases as schedulable while 90% test cases are actually schedulable as tested through simulations when $\theta = 100\%$. Thus, in this case, the difference between the acceptance ratio

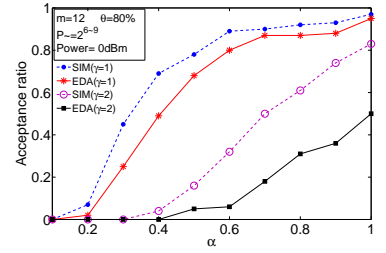


(a) Priority assignment: DM

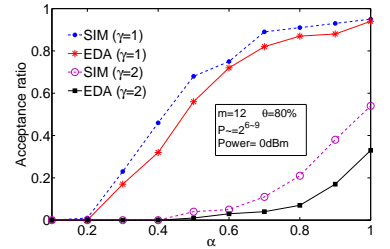


(b) Priority assignment: PD

Fig. 3. Acceptance ratio under varying number of sources and destinations



(a) Priority assignment: DM



(b) Priority assignment: PD

Fig. 4. Acceptance ratio under varying deadlines

(EDA) of our analysis and the fraction (SIM) of test cases that have no deadline misses in the simulations is 0.03. That is, there are 3% test cases that are actually schedulable but have been rejected by our analysis. The figure also indicates that the difference between EDA and SIM is always less than 0.06 when $\gamma = 1$. When $\gamma = 2$, the difference is less than 0.04 as long as $\theta \leq 50\%$. When $\theta > 50\%$, this difference increases sharply with the increase of θ but always remains less than 0.34. For PD priority assignment, Figure 3(b) shows that the difference between SIM and EDA is less than 0.05 when $\gamma = 1$ and is less than 0.25 when $\gamma = 2$.

Varying deadlines. Now we evaluate the performance by varying the deadlines of the flows. For the test cases with $\theta = 80\%$, we vary the deadlines of the flows by changing α

and the results are shown in Figure 4. Note that the deadlines of the flows become longer as α increases. Therefore, the number of schedulable test cases increases with the increase of α . For DM priority assignment, Figure 4(a) shows that the difference between *SIM* and *EDA* is at most 0.20 when $\gamma = 1$. The results indicate that most of the schedulable cases are accepted by our analysis when $\gamma = 1$ under DM priority assignment policy. When $\gamma = 2$, the difference is less than 0.12 as long as $\alpha \leq 0.5$. The difference is larger for $\alpha > 0.5$. We can see the maximum difference at $\alpha = 0.9$ where *SIM* is 0.74 while *EDA* being 0.34 only. That is, the number of schedulable test cases that are rejected by our analysis is comparatively less if the deadlines are tighter when $\gamma = 2$. The difference between *EDA* and *SIM* is much smaller when the priorities are assigned using PD as shown in Figure 4(b). Here, in most cases, *EDA* is very close to *SIM* when $\gamma = 1$. The maximum difference is 0.15. *EDA* remains very close to *SIM* when $\gamma = 2$ too. Their difference always remains less than 0.22. This indicates that the number of schedulable test cases that are rejected by our analysis is much less under PD priority assignment than that under DM priority assignment.

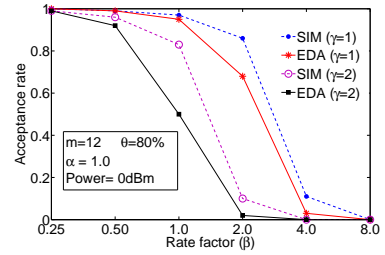
Varying rates. In all previous tests, periods were in the range $2^{6\sim 9}$ slots. For the test cases with $\theta = 80\%$, we now tune the rate of every flow by changing β . For example, setting $\beta = 0.50$ doubles the period of every flow. The acceptance ratios under varying rates are shown in Figure 5. For DM priority assignment, Figure 5(a) shows that the maximum difference between *SIM* and *EDA* is 0.18 which happens when $\beta = 2$ for $\gamma = 1$. No test cases are schedulable if $\beta > 4$. For $\gamma = 2$, the maximum difference between *SIM* and *EDA* is 0.33. According to Figure 5(b), the difference between *SIM* and *EDA* is always less than 0.21 both when $\gamma = 1$ and when $\gamma = 2$ under PD priority assignment. The results indicate that the acceptance ratios of our analysis decrease sharply with the increase of rates but remain close to the fractions of test cases that meet deadlines in simulations.

C. Simulations with Random Topologies

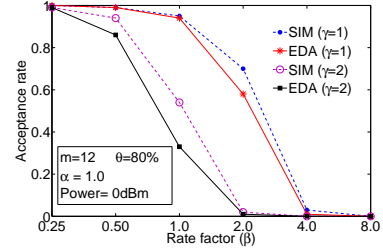
Generating networks. We test the scalability of our algorithms on random topologies of different number of nodes. Given the number of nodes (n) and edge-density (ρ), we generate random networks. A network with n nodes and $i\%$ edge-density has a total of $(n(n-1)*i)/(2*100)$ bidirectional edges. The edges are chosen randomly and assigned PRR randomly in the range $[0.80, 1.0]$. We keep regenerating a network until the required number of routes (γ) between every source and destination pair are found.

We vary the total number of nodes (n) in the network while the other parameters are chosen as follows: (i) $\theta = 80\%$ meaning that 40% of the total nodes are sources while another 40% are destinations; (ii) number of channels $m = 12$; (iii) edge-density $\rho = 40\%$; (iv) $\alpha = 1.0$; (v) periods are chosen randomly in the range $P_{\sim} = 2^{7\sim 10}$ slots.

Performance under DM. Figure 6 plots the performance of our analysis in networks with different number of nodes when the



(a) Priority assignment: DM



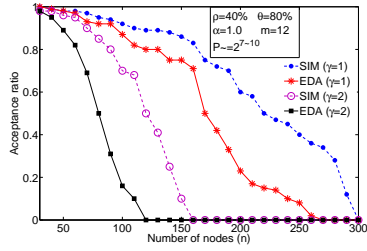
(b) Priority assignment: PD

Fig. 5. Acceptance ratio under varying rates

priority is assigned using DM. As shown in Figure 6(a), when $\gamma = 1$, the acceptance ratio of our analysis (*EDA*) is equal to *SIM* up to 50 nodes. Up to 140 nodes, the difference between *EDA* and *SIM* is less than 0.1. For 270 nodes, *EDA* is 0 while *SIM* is still 0.34. *SIM* becomes 0 when the number of nodes is 300. Compared to the cases with $\gamma = 1$, the performance is worse when $\gamma = 2$. For 120 nodes, *EDA* is 0 while *SIM* is still 0.50. *SIM* becomes 0 when the number of nodes is increased beyond 150. Figures 6(b) and 6(c) plot the pessimism ratios of the flows in a randomly selected run for every network size. Figure 6(b) indicates that the 75th percentile of the pessimism ratios is less than 1.6 for all cases up to 260 nodes except the case with 210 nodes when $\gamma = 1$. Figure 6(c) indicates that the 75th percentile is less than 1.8 for all test cases up to 110 nodes when $\gamma = 2$.

Performance under PD. Figure 7 shows the performance under PD. Figure 7(a) indicates that the difference between *EDA* and *SIM* remains less than 0.13 up to 160 nodes when $\gamma = 1$ and up to 60 nodes when $\gamma = 2$. Figure 7(b) indicates that the 75th percentile of the pessimism ratios is less than 1.61 for all cases up to 180 nodes. Figure 7(c) indicates that the 75th percentile is less than 1.74 for all test cases up to 100 nodes when $\gamma = 2$.

The results indicate that our analysis is effective even for very large networks under various fixed priority assignment policies. 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. We have also observed that the performance of our analysis is much better when $\gamma = 1$ compared to the cases when $\gamma = 2$. 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



(a) Acceptance ratio

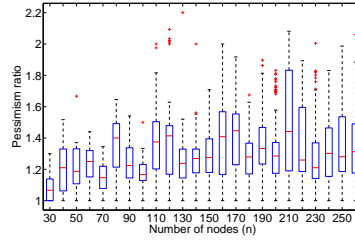
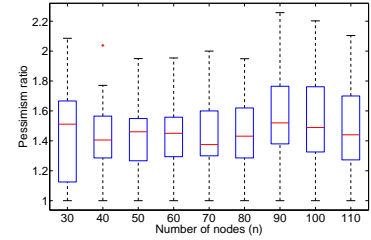
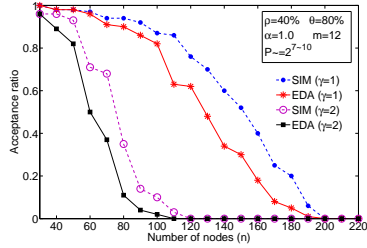
(b) Pessimism ratio ($\gamma = 1$)(c) Pessimism ratio ($\gamma = 2$)

Fig. 6. Schedulability considering DM fixed priority scheduling under varying network sizes



(a) Acceptance ratio

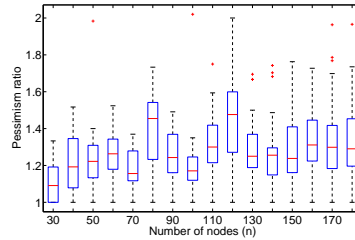
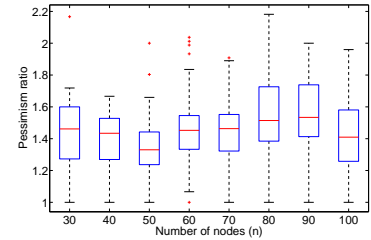
(b) Pessimism ratio ($\gamma = 1$)(c) Pessimism ratio ($\gamma = 2$)

Fig. 7. Schedulability considering PD fixed priority scheduling under varying network sizes

as an acceptance test for real-time flows under various network configurations.

VII. CONCLUSION

In this paper, we have mapped the scheduling of real-time data flows between sensors and actuators in a WirelessHART network to the real-time multiprocessor scheduling. Based on the mapping, we have presented a novel end-to-end delay analysis to determine the schedulability of feedback control loops in WirelessHART networks which is as yet unaddressed in the literature. Simulation results on both random and real network topologies demonstrate that our estimated delay bounds are reasonably tight and that our end-to-end delay analysis is very effective in terms of acceptance ratio under various fixed priority scheduling policies.

ACKNOWLEDGEMENT

This research was supported by NSF under grants CNS-0448554 (CAREER), CNS-0627126 (NeTS-NOSS), CNS-1017701 (NeTS), and CNS-0708460 (CRI).

REFERENCES

- [1] J. Song, A. K. Mok, D. Chen, and M. Nixon, "Challenges of wireless control in process industry," in *Workshop on Research Dir. for Security and Net. in Critical Real-Time & Embedded Sys.*, 2006.
- [2] "WirelessHART specification," 2007, <http://www.hartcomm2.org>.
- [3] D. Chen, M. Nixon, and A. Mok, *WirelessHARTTM 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*.
- [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*.
- [7] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha, "Flow-based real-time communication in multi-channel wireless sensor networks," in *6th European Conference on Wireless Sensor Net. 2009 (EWSN '09)*.
- [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*.
- [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*.
- [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*.
- [11] K. Karenos and V. Kalogeraki, "Real-time traffic management in sensor networks," in *RTSS '06*.
- [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 Syst.*, 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*.
- [14] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "JiTTS: Just-in-time scheduling for real-time sensor data dissemination," in *PERCOM '06*.
- [15] Y. Gu, T. He, M. Lin, and J. Xu, "Spatiotemporal delay control for low-duty-cycle sensor networks," in *RTSS '09*.
- [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] P. Jurcik, R. Severino, A. Koubãa, M. Alves, and E. Tovar, "Real-time communications over cluster-tree sensor networks with mobile sink behaviour," in *RTCSA '08*.
- [19] J. B. Schmitt and U. Roedig, "Sensor network calculus A framework for worst case analysis," in *DCOSS '05*.
- [20] N. Pereira, B. Andersson, E. Tovar, and A. Rowe, "Static-priority scheduling over wireless networks with multiple broadcast domains," in *RTSS '07*.
- [21] H. Zhang, P. Soldati, and M. Johansson, "Optimal link scheduling and channel assignment for convergecast in linear WirelessHART networks," in *IEEE WiOpt*, Seoul, Korea, Jun 2009.
- [22] P. Soldati, H. Zhang, and M. Johansson, "Deadline-constrained transmission scheduling and data evacuation in WirelessHART networks," in *The European Control Conference 2009 (ECC '09)*.
- [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*.
- [24] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for WirelessHART networks," in *RTSS '10*.
- [25] Wireless sensor network testbed, <http://mobilab.wustl.edu/testbed>.