A Utilization-Based Approach for Schedulability Analysis in Wireless Control Systems

Venkata P. Modekurthy, Dali Ismail, Mahbubur Rahman, Abusayeed Saifullah
Department of Computer Science, Wayne State University, Detroit, MI, USA
{modekurthy, dali.ismail, r.mahbub, saifullah}@wayne.edu

Abstract*—Recent advancements in industrial Internet-of-Things (IoT), more specifically, the development of industrial wireless standards such as WirelessHART and ISA100, are paving the way for the fourth industrial revolution, Industry 4.0. These wireless standards specify highly reliable and real-time communications as key requirements in industrial wireless sensor-actuator networks. Schedulability analysis remains the cornerstone for analyzing the real-time performance of these networks. While it is well-explored in the domain of CPU scheduling, schedulability analysis for multi-hop wireless networks has seen little progress till date. Existing work mostly focuses on worst-case delay analysis that runs in pseudo-polynomial time, making it is less suitable under frequent network dynamics which are quite common in industrial IoT. To address this, in this paper, we develop a schedulability analysis based on utilization bound for multi-hop, multi-channel industrial wireless sensor-actuator networks. Because of its extremely low runtime overhead, utilization-based schedulability test is considered to be one of the highly efficient and effective schedulability analyses. However, no work has been done yet on utilization-based analysis for multi-hop wireless network. The key challenge for a utilization-based test for multi-hop wireless network arises from the fact that wireless network is subject to transmission conflict and network dynamics which are not present in CPU scheduling. We address this challenge by bridging the gap between wireless domain and CPU task scheduling. We have evaluated our result through simulations using TOSSIM that shows that our schedulability analysis is safe and effective in practice.

***Index Terms*—Industrial Internet of Things, Wireless Sensor-Actuator Networks, WirelessHART, Schedulability Analysis.**

I. Introduction

Recent advancements in industrial Internet-of-Things (IoT), more specifically the development of industrial wireless standards such as WirelessHART [1] and ISA100 [2], are paving the way for the fourth industrial revolution, Industry 4.0 [3]. These wireless standards offer a closed loop communication between the sensors and actuators, where sensors measure process variables and deliver to a controller. The controller generates control commands based on the measured process variables and then sends the control commands to the actuators through the network. In order to ensure the stability of the industry, these wireless standards should offer reliable and real-time communication, i.e., a control command should reach the actuator before a given deadline. For example, in oil refineries, the spilling of oil tanks has to be avoided by controlling the level measurement in real-time. However, industry settings pose a harsh environment for wireless communication causing frequent transmission failures due to noisy channels, limited bandwidth, obstacles, multi-path fading, and interference that make it difficult to meet these requirements [4].

Industrial wireless standards such as WirelessHART mitigate frequent transmission failures through channel hopping and multi-channel communication. These networks, therefore, provide the feasibility of achieving reliable and real-time communication over wireless for critical process control applications. Nevertheless, unlike the wired counterpart, real-time scheduling theory for the wireless network is still not well-developed.

Schedulability analysis remains the cornerstone of real-time scheduling theory for industrial IoT [3]. Schedulability analysis is used to determine, both at design time and for online admission control, whether a set of real-time control loops/flows (i.e., end-to-end communication between a sensor and an actuator) can meet their deadlines. It thus helps the network manager to plan in advance and adjust workloads in response to network dynamics for real-time process control applications. For example, during channel blacklisting or a route change, the analysis is used to promptly decide the rate of a control loop/s to maintain real-time guarantee. In addition to design time and online admission control, a schedulability analysis is used in scheduling-control codesign [5], real-time routing, and priority assignment [6]. However, existing work on schedulability analysis focuses on worst-case delay analysis [7], [8] which runs in exponential time. Hence, these techniques are less suitable for Industry 4.0 architectures, using industrial IoT, which require frequent checking due to channel/link/node failures and changes to plant operating conditions.

In this paper, we develop a schedulability analysis based on *utilization bound* which is yet an unexplored problem for multi-hop wireless networks. In this approach, we ascertain the maximum possible utilization of all flows in the network and determine the flows as *schedulable* if the total utilization does not exceed the maximum possible utilization in the network. Because of its extremely low runtime overhead, a utilization-bound based schedulability test is considered one of the most efficient and effective schedulability tests. Therefore, it was extensively studied in CPU scheduling [9]. However, no work has been done yet on utilization-based analysis of multi-hop wireless network. The key challenge arises from the fact that wireless networks are subject to transmission conflict and dynamics which are not present in CPU scheduling. We address this challenge by bridging between wireless domain and CPU task scheduling. We characterize simultaneous transmission on multiple channels as processors in a multi-processor environment and transmission conflict as task blocking in traditional non-preemptive scheduling.

We evaluated our schedulability analysis in simulations using TOSSIM [36] for the earliest deadline first (EDF) and the deadline-monotonic (DM) scheduling algorithms. Simulations results show that our schedulability analysis is safe and effective in practice. Our analysis hence can be used as an effective schedulability test for admission control of real-time flows.

The rest of the paper is organized as follows. Section II describes the background on schedulability analysis. Section III reviews related work on schedulability analysis for industrial IoT. Section IV describes the system model and Section V presents the problem formulation. Section VI presents the schedulability analysis. Section VII extends the proposed schedulability analysis to a general industrial IoT environment. Section VIII presents the simulation results. Section IX concludes the paper.

II. Background

In general, *end-to-end delay bound analysis* and *utilization bound analysis* are the two broad approaches for schedulability analysis. A utilization bound analysis specifies the maximum possible utilization of all flows in the network and determines the flows as *schedulable* if the total utilization does not exceed the maximum possible utilization in the network. Because of its extremely low runtime overhead, a utilization-based schedulability test is considered one of the most efficient and effective schedulability tests. The end-to-end delay bound based analysis [7], [10], [8] requires a separate schedulability test for each flow, which runs in *pseudo-polynomial* time (i.e., *exponential* in the length of the input). However, utilization bound based analysis can provide a single closed-form expression that can run in polynomial time (usually in *linear* time). It thus greatly simplifies various scheduling-control optimization problems, for which pseudo-polynomial time delays bounds is a major hurdle due to its non-linearity, non-convexity, non-differentiability, long execution time, and a large number of constraints (at least n constraints for n flows) [5]. In this research, we want to develop a schedulability analysis based on *utilization bound* which is a yet unexplored problem for multi-hop wireless networks.

III. Related Work

Real-time scheduling for wireless networks was explored in many early [11] and recent works [12], [13], [14], [15], [16], [17], [18], [19], [20], [21]. However, these works do not focus on schedulability analysis in the network. The works in [22], [23], [24], [21], [25], [26] discuss schedulability analysis for wireless sensor networks using end-to-end delay bound and they focus on data collection through a routing tree [24], [22] and/or do not consider multiple channels [22], [23]. In contrast, we consider an industrial IoT based on multiple channels and our analysis is not limited to data collection towards a sink. Furthermore, our analysis is targeted for real-time flows between sensors and actuators for process control purposes and applies to multi-path routing with minimal changes.

Real-time scheduling for industrial IoT based on WirelessHART has received considerable attention in the recent past [27], [7], [6], [28], [5], [29], [30], [31]. The works in [28], [29] focus on data collection in a tree topology. The works in [30], [32] address graph routing algorithms for WirelessHART networks and that in [31] propose a localization system using WirelessHART. Priority assignment policies for WirelessHART are studied in [6] and rate selection algorithms are studied in [5]. The work in [27] considers dynamic priority scheduling and does not address any schedulability analysis. To summarize, none of these works focus on schedulability analysis. The work in [7] presents the first step in establishing a rigorous delay analysis for industrial IoT. Nevertheless, this work does not consider the critical fault-tolerant mechanisms, for achieving reliable communication, like retransmissions and reliable graph routing. The work in [8] provides a suite of end-to-end delay analysis techniques for schedulability analysis under fixed priority scheduling in WirelessHART networks. In summary, these papers use the worst-case delay that can be obtained by a flow to determine its schedulability. Consequently, these algorithms are less suitable for admission control as many control optimization algorithms execute during run-time and processor cannot be overloaded with other jobs.

In this paper, we propose a utilization-based approach for schedulability analysis which provides the simplicity and efficiency in application. A utilization-based analysis was studied in [33] for a single-hop wireless network. In contrast, we focus on multi-hop industrial IoT in which scheduling and analysis are significantly different and challenging as it has to deal with multiple concurrent transmissions on different channels, interferences, and transmission conflicts. Efficient schedulability analysis is particularly useful for online admission control and adaptation (e.g., when network route, topology, or channel condition change) so that the network manager can quickly reassess the schedulability of the flows.

IV. System Model

Because of the worldwide adoption of WirelessHART for process control in challenging industrial environments, we consider an industrial IoT based on the WirelessHART standard [1]. WirelessHART networks operate on the 2.4GHz band and are build on the physical layer of IEEE 802.15.4. They form a multi-hop mesh topology of nodes consisting of multiple field devices, multiple access points, and a gateway. The *network manager* creates routes and transmission schedules. *Access points* provide redundant paths between the wireless network and the gateway. The *field devices* are wirelessly networked sensors and actuators. The sensors periodically deliver sample data to the controller through the access points. The controller generates control commands based on the measured process variables and then sends the control commands to the actuators. Each node has a *half-duplex* omnidirectional radio transceiver, and hence cannot both transmit and receive at the same time and can receive from at most one sender at a time.

Transmissions in a WirelessHART network are scheduled based on a multi-channel TDMA (Time Division Multiple Access) protocol. The network employs global time synchronization protocols to synchronize time at all nodes in the network. Each time slot is of $10ms$ and each transmission needs one time slot. A receiver transmits an acknowledgment to notify the sender about a successful reception of a packet. Note that, both the transmission and acknowledgment happen in one 10ms time slot. The network uses the channels defined in IEEE 802.15.4. It adopts *channel hopping* in every time slot to achieve high reliability. An excessively noisy channel is assumed to be *blacklisted* and not used for communication. We also assume that the network does not allow spatial reuse of channels, i.e., a channel can be used by only one node to transmit packet in a time slot. We assume the network adopts a tree routing, where all nodes in the network form a tree rooted at the gateway. Sensor nodes forward data to the controller (located at the gateway) through the upward links. The controller sends control commands to the actuators through the downward links. Note that, we make this assumption only to provide a tight bound on the schedulability analysis and our method can be extended to graph routing, as will be described in Section VII-A, that is typically adopted in industrial IoT and that provides redundant paths for packet delivery for enhanced reliability.

V. Problem Formulation

Each control loop, also called a *flow*, involves one or more sensors and one or more actuators. Transmissions between access points, sensors, and actuators are scheduled on m ($m \geq 1$) channels. We assume, there are n control loops denoted as $F_1, F_2, \cdots, F_n$. The worst-case execution time of a flow, period (sampling rate of sensors), and the deadline of F_i are denoted by C_i, T_i, and D_i, respectively. Note that, real-time wireless standards like WirelessHART and ISA-100 reserve a fixed ω number of slots for each link to handle both transmissions and re-transmissions. WirelessHART uses $\omega = 2$ for each link to successfully transmit a packet. Thus, if there are a total of ℓ_i links on flow F_i's route, then C_i can be computed as $C_i = \ell_i \times \omega$. Note that, C_i can change due to network dynamics and route change. In these situations, we need to run the schedulability test again to make sure the new routes are schedulable on the network.

The set of periodic flows F is called *schedulable* if there is a transmission schedule such that no deadline is missed. A schedulability test $\mathbb{S}$ is *sufficient* if any set of flows deemed to be schedulable by $\mathbb{S}$ is indeed schedulable. If flow F_i involves a maximum of C_i transmissions, then its utilization u_i is defined as $\frac{C_i}{T_i}$ and the total utilization of all n flows is defined as $\sum_{i=1}^{n} \frac{C_i}{T_i}$. In this paper, our objective is to determine a sufficient schedulability analysis for EDF and DM schedulers based on utilization bound. Note that, utilization-based schedulability test for other schedulers is out of the scope of this paper.

VI. Utilization-Based Schedulability Analysis

Here, we first propose an approach for determining a schedulability analysis by bridging the gap between processor and network scheduling. We then discuss transmission conflict delay computation for a flow. Table I summarizes the notations used in this section.

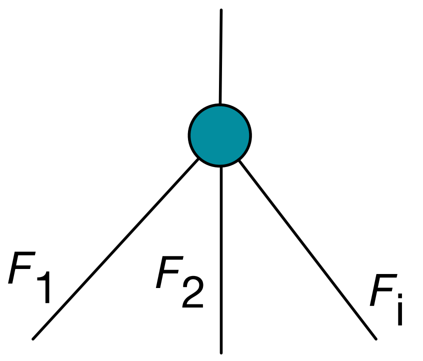
A. Establishing a Utilization Bound Analysis

Channel contention and transmission conflict are the two sources of delays in multi-hop wireless networks. ***Channel contention*** is defined as the delay caused when all available channels in a time slot are assigned to higher priority flows. ***Transmission conflict*** is the delay when two flows share a common node, and the higher priority flow delays the low priority flow at the common node since the common node can transmit/receive a packet for only one flow in one time slot (due to single half-duplex radio). In this section, we establish the utilization-based analysis assuming we know the transmission conflict delay. In the following section, we discuss the transmission conflict delay computation.

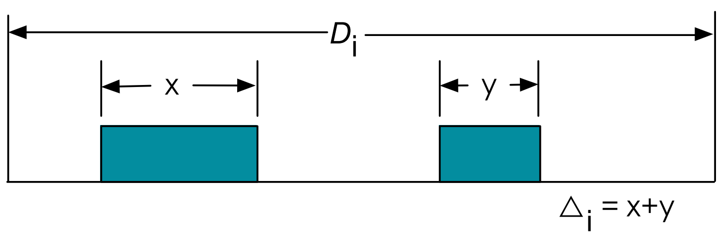
Channel contention delay in industrial IoT can be considered similar to execution delay experienced by a task running on a multi-processor platform when (i) the number of channels in industrial IoT is equal to the number of processors (we use m interchangeably to denote both channels and processors); (ii) length of one time slot (10ms) is equal to the length of one time unit in process scheduling (where a task is non-preemptable); and (iii) period, deadline and worst-case execution time (WCET) of each flow is equal to its analogous task (in the analogous processor task set). This is because a flow (in industrial IoT) and its equivalent task (in processor) are contending for m shared resources. For the same values of period, deadline, and execution time both a flow and its equivalent task generate the same access patterns. Thus, the delay due to channel contention observed by a flow in the network is the same as the delay observed by its equivalent task on a processor. Our technical approach leverages this bridge between multiprocessor scheduling and wireless transmission scheduling.

We first review the results on preemptive and non-preemptive scheduling on multiprocessors. In *preemptive scheduling*, a task upon start can be preempted by any higher priority task any time. In *non-preemptive scheduling*, a task once started can never be preempted by any other task. In non-preemptive scheduling, a higher priority task thus experiences priority inversion where a high priority task is *blocked* by a lower priority task (as it cannot preempt if the lower priority task has already started). EDF is a dynamic priority scheduling policy where, at any time, the task having the shortest absolute deadline is scheduled first. A set of n real-time tasks with a constrained deadline (i.e. $D_i \leq T_i$) is schedulable using preemptive EDF scheduling on m processors [34] if

$$\sum_{i=1}^{n} \frac{C_i}{D_i} \leq m - (m-1) \left(\max \left\{ \frac{C_i}{D_i} | 1 \leq i \leq n \right\} \right). \quad (1)$$



(a) Transmission conflict



(b) Time loss (cyan color) due to conflict

Fig. 1. Time loss due to transmission conflict

DM scheduling is a fixed priority scheduling policy where tasks are prioritized based on their relative deadlines. A set of n real-time tasks with a constrained deadline (i.e. $D_i \leq T_i$) is schedulable using preemptive DM scheduling on m processors [34] if

$$\sum_{i=1}^{n} \frac{C_i}{D_i} \leq \frac{m}{2}\left(1 - \max\left\{\frac{C_i}{D_i} | 1 \leq i \leq n\right\}\right) + \max\left\{\frac{C_i}{D_i} | 1 \leq i \leq n\right\}. \quad (2)$$

For non-preemptive scheduling the corresponding conditions are derived by taking into account the maximum blocking time.

To adopt the similar results for industrial IoT, we present our technique as follows. We can use Equation (1) and Equation (2) for industrial IoT in the absence of transmission conflict, and when every transmission happens on a separate channel in each time slot (allowing at most m concurrent transmission per time slot). However, transmission conflict poses an additional challenge in the wireless domain. We can model the transmission conflict delays as blocking time in non-preemptive scheduling. Assuming Δ_i (as computed in Section VI-B) denotes the transmission conflict delay caused on flow F_i by all higher priority flows. Then, $D_i - \Delta_i$ represents the utilization loss due to transmission conflict, i.e., F_i can use at most $D_i - \Delta_i$ time slots to complete the end-to-end communication. For example, in Fig. 1(a), the receiver can receive from at most one transmitter in a time slot, and hence only one flow is assigned a time slot to transmit and the other two flows are blocked during this period. In this example, F_1 has the highest priority and hence is assigned time slots t and $t+1$ (let t be the current system time). Similarly, F_2 is assigned time slots $t+2$ and $t+3$. During time slots t, $t+1$, $t+2$, and $t+3$, flow F_i is blocked and waits for an idle transmission slot regardless of channel availability. Thus, we can consider a loss of four time slots from the relative deadline of F_i (Fig. 1(b)). Namely, from deadline D_i, the flow F_i loses at most Δ_i slots, and hence its effective utilization (as shown in Fig. 1(b)) μ_i becomes $\mu_i = \frac{C_i}{D_i - \Delta_i}$. Let us define

$$\mu_{\max} = \max\{\mu_i | 1 \leq i \leq n\};$$
$$\mu_{\text{sum}} = \sum_{i=1}^{n} \mu_i.$$

Therefore, from Equations (1) and (2), any constrained deadline set of real-time flows is schedulable on m channels in an industrial IoT that allows at most m concurrent transmissions under EDF scheduling if

$$\begin{aligned} &\mu_{\text{sum}} \leq m - (m-1)\mu_{\max} \\ &0 < \mu_i \leq 1 \qquad \forall 1 \leq i \leq n \end{aligned} \quad (3)$$

and under DM scheduling if

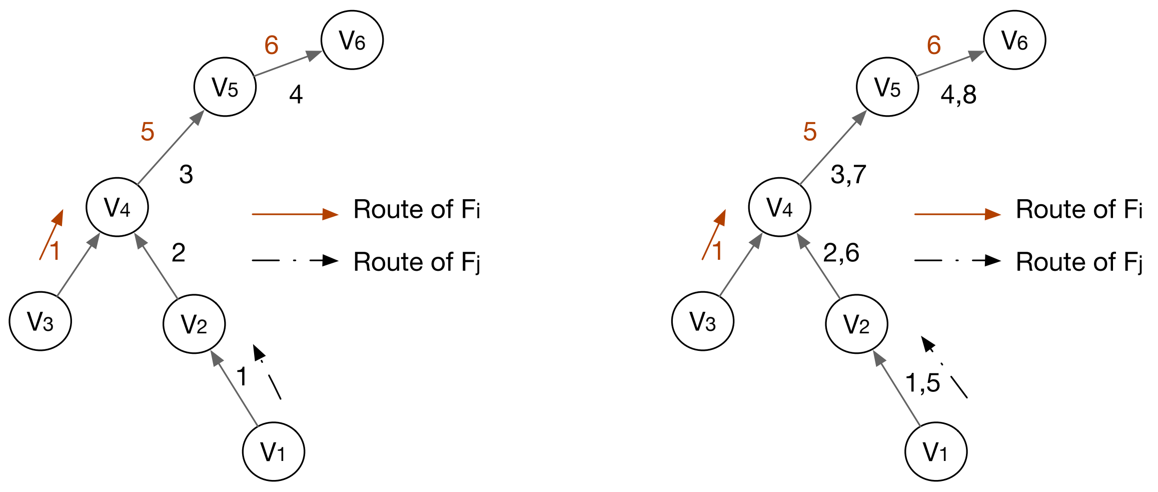
$$\begin{aligned} &\mu_{\text{sum}} \leq \frac{m}{2}(1 - \mu_{\max}) + \mu_{\max} \\ &0 < \mu_i \leq 1 \qquad \forall 1 \leq i \leq n. \end{aligned} \quad (4)$$

Note that, Equations (3) and (4) consider a preemptive scheduler where preemptions are allowed at the start of a time slot. For non-preemptive schedulers, Δ_i also includes the maximum blocking time caused due to priority inversions.

B. Transmission Conflict Delay Computation

In this section, we first obtain a bound on the number of shared paths between two routes on a tree routing. We then obtain a bound on the number of time slots a packet of flow F_j can delay a packet of flow F_i on a shared path. We then use these two bounds to derive an upper bound on the transmission conflict delay that a flow can experience under the assumption that network routing happens on a bi-directional tree.

We define a *common path* as a set of nodes shared between routes of two flows. A flow F_i experiences a transmission conflict delay by another flow F_j on a common path between the routes of F_i and F_j. To estimate the transmission conflict delay, we first estimate the number of common paths between F_i and F_j. Specifically, we first bound the number of common paths on the uplink route (which connects a sensor to the



(a) $T_i = T_j = 8; \delta(i,j) = 3$

(b) $T_i = 8, T_j = 4; \delta(i,j) = 3$

Fig. 2. An example of F_j delaying F_i on a tree routing

Symbol	Description
F_i	Flow i
T_i	Period of F_i
D_i	Deadline of F_i
C_i	worst-case execution time of F_i
Δ_i	Transmission conflict delay on F_i
$\delta(i,j)$	Transmission conflict delay caused on F_i by high priority flow F_j
$hp(F_i)i$	Set of flows that are higher priority than F_i
$\alpha(i,j)$	# of common paths between routes of F_i and F_j
$\alpha_1(i,j)$	# of common paths with path length as one between routes of F_i and F_j
$\beta(\rho_i,j)$	# of common paths between ρ_i-th path of flow F_i and all paths of F_j
$\beta_1(\rho_i,j)$	# of common paths between ρ_i-th path of flow F_i and all paths of F_j with path length 1
ω	# of transmission slots assigned for each link in a flow

TABLE I
NOTATIONS

controller) and the downlink route (which connects the controller to an actuator). In the uplink route, F_i and F_j transmit messages to the same destination, i.e., the controller, which resides at the root of the tree. In a tree, there can exist only one parent for every node. Therefore, a common path which starts at some node V_k (where V_k can be an intermediate node or the controller), only ends at the controller. Thus, we can conclude that in an uplink route there exists only one common path which starts at some node V_k and ends at the controller. We can use similar reasoning for the downlink route. In a downlink route, there exists only one common path which starts at the controller and ends at an intermediate node V_l. Therefore, in a tree route, the number of common paths is limited to 1 where the common path is said to start at V_k and end at V_l. For example, Fig 2(a) shows that two flows F_i and F_j share a path from V_4 to V_6.

We now compute the maximum delay caused by one packet of a flow F_j on a packet of flow F_i considering tree routing. At a common node, V_a, on the common path, the packet of F_i is delayed at most 3ω times by a packet of F_j [8]. For example, as shown in Fig. 2(a) flow F_i's transmission from $V_4 \rightarrow V_5$ conflicts with three transmissions of F_j, $V_2 \rightarrow V_4, V_4 \rightarrow V_5$, and $V_5 \rightarrow V_6$. At time $3\omega + 1$ (time slot after the blocking duration), packets of F_i and F_j have different destination nodes, and they can use different channels to make concurrent transmissions. Note that, if only one channel is available for transmission, then the packet of F_i is blocked due to channel contention and not due to transmission conflict. Thus, we can say that on a common path between two flows, a low priority packet can be delayed due to transmission conflict by a high priority packet on one node. For example, in a common path from $V_a, V_b, V_c, V_d, \cdots V_k$, if at time τ a packet of F_j delays a packet of F_i by $\delta(i,j)$, then at time $\tau + 3\omega$ the packet of F_j is at node V_c and packet of F_i is at node V_a. At time $\tau + 3\omega + 1$, F_i and F_j can make simultaneous transmission thereby keeping a non-decreasing distance between F_i and F_j flows. In summary, we can say that a packet of F_j blocks a packet of F_i by at most 3ω time slots.

We now compute the maximum delay caused by a flow F_j on a flow F_i. On a common path, a flow F_i can be delayed by at most $\lceil \frac{T_i}{T_j} \rceil$ times by a high priority flow F_j. Note that, we use a pessimistic scenario in which all packets of flow F_j that spawned in the interval $[\alpha T_i, (\alpha+1)T_i]$ interfere α^{th} packet of F_i. A more accurate bound can be obtained by using response time analysis which is complicated and takes exponential time to find a solution (which we are trying to avoid in this paper). Considering a pessimistic value on the number of interfering packets, the total delay caused by flow F_j on a flow F_i is expressed as

$$\delta(i,j) = 3\omega \left\lceil \frac{T_i}{T_j} \right\rceil.$$

For a DM scheduler, a flow F_i can only be delayed by a high priority flow. Therefore, the total delay experienced by flow F_i under a DM scheduler considering all high priority flows (given by $hp(F_i)$) is shown by Equation (5).

$$\Delta_i^{DM} = \sum_{F_j \in hp(F_i)} \delta(i,j) \tag{5}$$

In case of an EDF scheduler, the flows have dynamic priority based on their absolute deadlines. Therefore, a flow F_i can be delayed by every other flow F_j where $j \neq i$. Note that, some flows interfere at most once due to very large periods and some flows interfere multiple times due to very short periods. To incorporate these additional delays, we can extend the total delay computation for DM schedulers to EDF schedulers by considering all flows interfere F_i. Under an EDF scheduler, an upper bound of the total delay experienced by a flow F_i is given by Equation (6).

$$\Delta_i^{EDF} = \sum_{j \in [1,n] \text{ and } \neq i} \delta(i,j) \tag{6}$$

VII. Extending the Utilization Bound Analysis to Graph Routing and Hierarchical Networking

In this paper, we consider a network model that supports only tree routing and does not consider the spatial reuse of channels. In this section, we present an approach to handle graph routing algorithms and spatial reuse of channels.

A. Transmission Conflict Delay Computation for Graph Routing Algorithms

Routing in industrial IoT is broadly classified into two types: source routing and graph routing. Source routing provides a single route for each flow in the network. We define, a *routing graph* as a directed list of loop-free paths between a source and a destination. Each node in a routing graph must have a minimum of two unique outgoing paths from itself to the destination. Graph routing allows to schedule a packet on multiple links using multiple channels on multiple time slots through multiple paths to deliver a packet to a destination, thereby ensuring high reliability in highly unreliable environments. A routing graph consists of an uplink graph and multiple downlink graphs. An uplink graph connects all sensors to controllers while each downlink graph connects a controller to an actuator. Note that, some recent work on industrial wireless network focuses on route-less approach by relying on a network-wide Glossy flooding [35]. However, such approaches consider very sparse traffic and rely on single channel. Such a model does not have to handle transmission conflict or channel contention. Therefore, our model and approach are significantly different, more challenging, and more general.

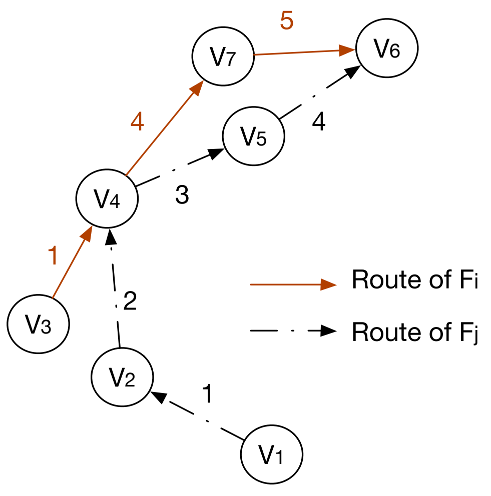
In a source or graph routing algorithm, packets can experience different degrees of conflict during communication. For example, a flow can experience a delay at multiple nodes, as shown in Fig. 3(a). These different degrees of conflict cause different transmission conflict delay on flows. In the following discussion, we derive an upper bound on the delay that a lower priority flow can experience from the higher priority ones due to conflicts for a network with source routing. Here, we first discuss the transmission conflict delay caused by a packet of F_j on a packet of F_i. We then extend the analysis to consider transmission conflict delay caused by all packets of F_j on F_i under a source routing algorithm. We then extend this result to graph routing algorithms.

In a generalized routing, routes of two flows can intersect each other more than once and can generate multiple common paths since there is no limitation on the number of parent nodes. One packet of flow F_i can be delayed at all common path by the same packet of flow F_j since F_j and F_i experience a different delay pattern on the non-common paths. However, on a common path, a packet of F_j can delay a packet of F_i only for 3ω time slots (the same reasoning as discussed in the previous section applies here). Therefore, if routes of two flow F_i and F_j intersect each other $\alpha(i,j)$ times then maximum delay a low priority packet experiences due to a high priority packet is expressed as $3\omega \times \alpha(i,j)$. However, as shown in Fig. 3(a) routes that intersect on only one node can have a maximum delay of 2ω since the destination node is not the same. Therefore, a packet of flow F_j delays a packet of flow F_i by $3\omega \times \alpha(i,j) - \omega \times \alpha_1(i,j)$ where $\alpha_1(i,j)$ represents the number of intersection in two routes with only one node common to both of them.

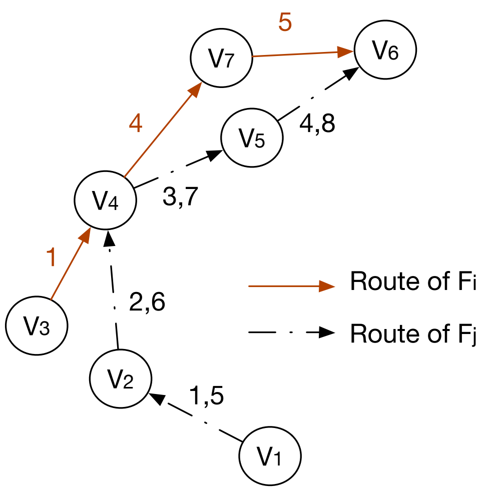
We now compute the maximum delay caused by a flow F_j on flow F_i. The delay caused by a flow F_j on a flow F_i is given by the Equation (7).

$$\delta(i,j) = (\alpha(i,j) + \left\lceil \frac{T_i}{T_j} \right\rceil - 1)3\omega - \omega \times \alpha_1(i,j) \tag{7}$$

To prove this, let us assume that there are $\alpha(i,j)$ common paths between the routes of F_i and F_j, of these common paths $\alpha_1(i,j)$ common paths consist of only one node, and $\lceil \frac{T_i}{T_j} \rceil$ packets of F_j interfere one packet of F_i. A low priority flow experiences a maximum delay when the first packet of the F_j interferes with the packet of F_i on the first $\alpha(i,j)-1$ common paths and $\lceil \frac{T_i}{T_j} \rceil$ packets of F_j interfere F_i on last common path (and the last common path has more than 2 nodes). In this scenario, the delay experienced by F_i at the first $\alpha(i,j)-1$ nodes is given by $(\alpha(i,j)-1)3\omega - \omega \times \alpha_1(i,j)$ and the delay experienced at the last common path is given by $\lceil \frac{T_i}{T_j} \rceil 3\omega$. Now combining the delays experienced at each node, we can compute the total delay as given in Equation (7). Note that, this is one scenario that leads to the maximum transmission conflict delay. However, no other scenario will result in a greater delay. To prove this, let us assume, without loss of generality, the first packet of F_j interferes the packet from F_i on 1-st to η-th common path, where $0 < \eta < \alpha(i,j)$, and $\lceil \frac{T_i}{T_j} \rceil$ packets of F_j interferes at the η-th common path and last packet interferes from the η-th common path to the last common path. In this scenario, first packet of F_j cannot interfere after η-th and second packet of F_j cannot interfere before η-th. Therefore, applying the same reasoning, a packet of F_i experiences a transmission conflict delay of $\lceil \frac{T_i}{T_j} \rceil \times 3\omega$ time slot only at the η-



(a) $T_i = T_j = 8; \delta(i,j) = 2$



(b) $T_i = 8, T_j = 4; \delta(i,j) = 2$

Fig. 3. An example of F_j delaying F_i when any routing algorithms is used

th common path and at all other common paths it experiences a transmission conflict delay of $(\alpha(i,j)-1)3\omega - \omega \times \alpha_1(i,j)$ time slots. This delay does not change with the value of η. Consequently, the delay caused by a flow F_j on a flow F_i is given by the Equation (7). For a DM scheduler only high priority flows interfere a low priority flow. Therefore, the total transmission conflict delay for a low priority flow F_i is given as

$$\Delta_i^{DM} = \sum_{F_j \in hp(F_i)} \delta(i,j).$$

For an EDF scheduler, all flows interfere all other flows. Therefore, the total transmission conflict delay for a flow F_i is given as

$$\Delta_i^{EDF} = \sum_{j \in [1,n] \text{ and } j \neq i} \delta(i,j).$$

We now use the transmission conflict delay result of source routing algorithm and extend it to compute the transmission conflict delay for a graph routing algorithm. In graph routing, we need to consider common paths generated due to multiple paths of the high priority flow. Let us assume the flow F_i has ϵ_i number of paths in the graph route, ρ_i denotes one path of the graph route, $\beta(\rho_i, j)$ denotes the number common paths between ρ_i-th path of F_i all paths of flow F_j and $\beta_1(\rho_i, j)$ denotes the number of common paths between ρ_i-th path of F_i all paths of flow F_j with only one common node. On a path ρ_i of flow F_i, the worst-case transmission conflict delay can be experienced when a packet of F_j interferes on $\beta_1(\rho_i, j)$ common paths between path ρ_i and all paths of F_j. Considering that on a path, a packet is at most delayed for 3ω (or 2ω depending on common path length), we can extend transmission conflict delay computation from source routing to transmission conflict delay experienced by a packet on the path ρ_ith to be

$$\zeta(\rho_i, j) = (\beta(\rho_i, j) + \left\lceil \frac{T_i}{T_j} \right\rceil - 1)3\omega - \beta_1(\rho_i, j)\omega.$$

The total transmission conflict delay caused by F_j on a packet of F_i (considering all paths of F_i) is given by

$$\delta(i,j) = \sum_{\rho_i=1}^{\epsilon_i} \zeta(\rho_i, j).$$

For a DM scheduler, the total transmission conflict delay experienced by a flow F_i is given by

$$\Delta_i^{DM} = \sum_{F_j \in hp(F_i)} \delta(i,j).$$

Similarly, for an EDF scheduler, the total transmission conflict delay experienced by a flow F_i is given by

$$\Delta_i^{EDF} = \sum_{j \in [1,n] \text{ and } j \neq i} \delta(i,j).$$

B. Adopting the Utilization Based Analysis through Hierarchical Networking

Because we derived the above results considering at most m concurrent transmissions in the network, we now propose a hierarchical network-based analysis where this constraint is relaxed for the global network. Specifically, we consider the network as a collection of subnetworks, where each subnetwork has its own subnetwork manager. Each manager adopts the above result at the subnetwork level. A global network manager coordinates with the subnetwork managers to manage the entire network in a hierarchical fashion. Every subnetwork will involve a unique channel for every transmission in a time slot. Thus if there are $m'(\leq m)$ channels used in a subnetwork, then there will be at most m' concurrent transmissions in the subnetwork. Therefore, we can use the results of Eq. 3 and Eq. 4 in each subnetwork directly.

An important technical challenge in our proposed hierarchical architecture is to deal with the interdependencies among the subnetworks. For example, if the subnetwork

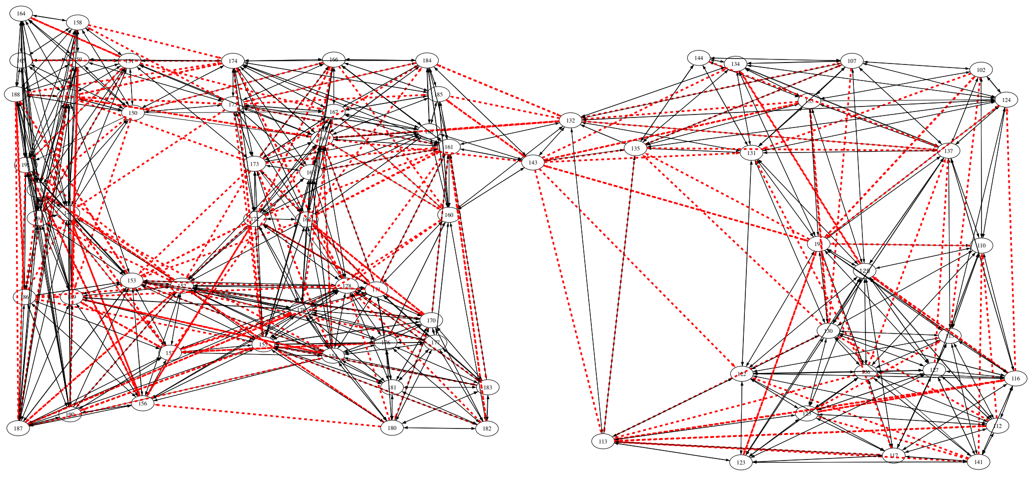


Fig. 4. Network topology used in simulation

manager of a subnetwork needs to create a local TDMA schedule (i.e., for the links inside the subnetwork), it may need to wait for its neighboring subnetworks (or some of the neighboring subnetworks) to finish their schedule, this is because of the dependency created by a packet. A packet routed through multiple subnetworks should be scheduled in the earlier subnetworks first. Because feedback flows involve both upwards and downward communication in the WSAN, such dependencies can be cyclic. For example, let us consider 2 packets p and q such that p needs to be scheduled first in subnetwork C_1 and then in subnetwork C_2, and that q needs to be scheduled first in subnetwork C_2 and then in subnetwork C_1. In such a scenario, C_1 needs to create a schedule after C_2 creates, and C_2 needs to create a schedule after C_1 creates, thereby creating a cyclic dependency.

Our proposed method to remove these dependencies is to assign sub-deadlines and release offsets for each flow among the subnetworks. Specifically, for every flow F_i that passes through a subnetwork C_j, we assign a release offset $r_{i,j}$ and a sub-deadline $d_{i,j}$ in the subnetwork. The deadline of flow F_i is equally divided into sub-deadlines $d_{i,j}$ for each subnetwork a flow passes through. The release offset $r_{i,j}$ is equal to the sub-deadline of F_i in the subnetwork where it needs to be scheduled immediately before C_j. Thus, subnetwork C_j needs to schedule F_i within the time window $[r_{i,j}, d_{i,j}]$, thereby requiring no knowledge of the schedule (for F_i) in other subnetworks.

VIII. Evaluation

A. Simulation Setup

We evaluated our utilization-based schedulability analysis results through simulations on TOSSIM [36]. We used the topology collected from a wireless sensor network testbed [37] of 74 nodes. Fig. 4 shows the collected topology where black lines are transmission links and red dotted lines show interference. Each node is equipped with Chipcon CC2420 radio which is compliant with the IEEE 802.15.4 standard. We implemented a multi-channel TDMA medium access control (MAC) protocol with channel hopping and a network layer to support tree routing. Time in the network was divided into 10ms slots, and clocks were synchronized across the entire network using the Flooding Time Synchronization Protocol (FTSP) [38]. For the sake of simplicity, we used Dijkstra's shortest path algorithm to generate a directional tree with root as the base station. We assumed all links to be bi-directional. We used packet reception ratio (PRR) as a metric for generating the routes. PRR values used in the simulation are obtained from real experiments. Links with a PRR higher than 90% are used to determine the topology of the network.

We used 5 IEEE 802.15.4 channels 12, 15, 17, 20, 21 for our simulations and the rest are assumed to be blacklisted. For the sake of simplicity, we assumed that spatial diversity of channels is not allowed, and at most one transmission is allowed on a channel. We used an optimal channel assignment algorithm proposed in [39] for channel assignment. We used either DM or EDF scheduling algorithm to allocate transmission time slots to each flow. We assigned 2 transmission time slots for each link on a flow. Second transmission slot is provided for redundancy and to account for transmission failures occurring due to channel noise.

We evaluated our analysis in terms of *Schedulability ratio* defined as the fraction of the test cases that are deemed schedulable. We used 100 random test cases to obtain the schedulability ratio. We used the number of flows in the network as a parameter for comparison. We generate flows by randomly selecting sources and destinations, and simulate their schedules. One node with the highest degree of connectivity in the topology was selected as an access point. We assigned a random harmonic period in the range $2^{10\sim15}$ms. The deadlines are equal to periods. Priorities of the flows are assigned based on the DM policy.

B. Results

We analyze the effectiveness of our analysis by simulating the complete schedule of transmissions of all flows released

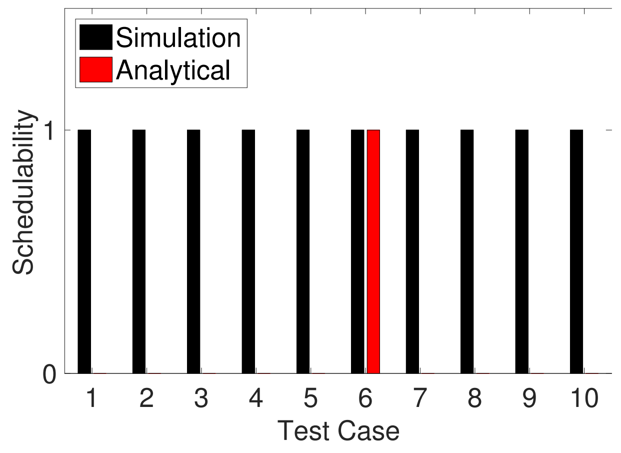


Fig. 5. Schedulability for 10 test cases

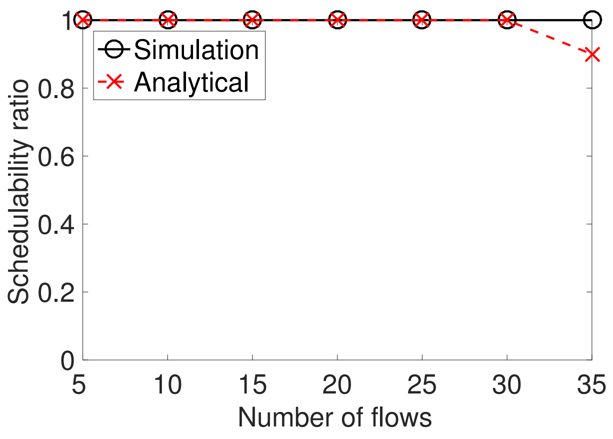


Fig. 7. Schedulability ratio under EDF scheduling

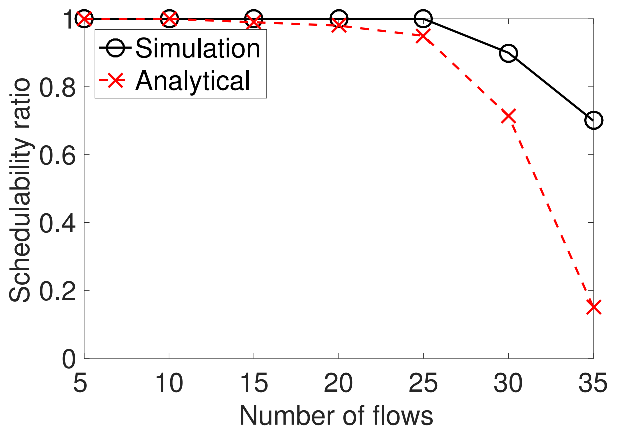


Fig. 6. Schedulability ratio under DM scheduling

within the hyper-period. In Fig. 6 and Fig. 7, "Simulation" indicates the fraction of test cases that have no deadline misses in the simulations, and represents conservative upper bounds of schedulability ratios; "Analytical" indicates the schedulability ratio based on our utilization-based schedulability analysis.

Fig. 6 shows the schedulability ratio for 100 test cases under varying number of flows for deadline monotonic scheduling algorithm. In our results, we observed that the analytical result shows a small decrease in schedulability ratio when compared to the simulation result. For 30 control loops, the simulation result shows that 73 test cases were schedulable and analytical results show that only 14 test cases were schedulable. Fig. 5 shows 10 such test cases. We observed that every test case that was said to be schedulable by the analytical result was, in fact, schedulable in simulation. We observed that the early decrease in schedulability ratio for analytical analysis for deadline monotonic scheduling is due to the loose upper bound for schedulability ratio in a multi-processor environment.

Fig. 7 shows the schedulability ratio for 100 test cases with the earliest deadline first scheduling algorithm. For 30 control loops, we observed that the simulation results show a 100% schedulability and analytical results show 90% schedulability. Similar to DM scheduling, all test cases that were deemed schedulable under analytical analysis were schedulable under simulation. We observed that delay computation for tree routing gives a tight upper bound and the gap between the analytical schedulability ratio and that based on simulations stems from the pessimism in the multi-processor utilization bounds. We observed a slow decrease in schedulability ratio for analytical analysis for the EDF scheduler. This phenomenon was due to the tighter bounds for schedulability ratio in a multi-processor environment. We also observed that for some cases, the scheduling algorithm also improved the schedulability ratio.

IX. Conclusion and Future Work

We have developed a schedulability analysis based on *utilization bound* which is a yet unexplored problem for multi-hop wireless networks. This approach determines the maximum total utilization of all flows in the network and determines those as *schedulable* if the total utilization does not exceed the maximum possible utilization in the network. Because of its extremely low runtime overhead, utilization-based schedulability test is considered one of the most efficient and effective schedulability tests. In this paper, we show the utilization bound for a WirelessHART application with tree routing. We have also discussed the computation of a utilization bound for WirelessHART applications with source and graph routing.

This work is the inception of a new horizon on utilization-based analysis for industrial IoT which can direct the wireless community in the same way the real-time systems research today evolved from Liu and Layland's utilization bound. Our result can trigger many research directions in the line of real-time scheduling, scheduling-control codesign, control performance optimization, routing, priority assignment, and mixed-criticality real-time wireless sensor and actuator networks. Our future work involves analyzing the effects of assigning sub-deadlines for large networks, packet loss, and the trade-offs among various control performance metrics.

References

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

[2] "ISA100: Wireless systems for automation," http://www.isa.org/MSTemplate.cfm?MicrositeID=1134&CommitteeID=6891.

[3] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, 2018.

[4] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-time wireless sensor-actuator networks for industrial cyber-physical systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1013–1024, 2016.

[5] A. Saifullah, C. Wu, P. Tiwari, Y. Xu, Y. Fu, C. Lu, and Y. Chen, "Near optimal rate selection for wireless control systems," *ACM Transactions on Embedded Computing Systems*, vol. 13, no. 4s, pp. 1–25, 2013.

[6] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Priority assignment for real-time flows in wirelesshart networks," in *the 23rd Euromicro Conference on Real-Time Systems (ECRTS)*. IEEE, 2011, pp. 35–44.

[7] ——, "End-to-end delay analysis for fixed priority scheduling in wirelesshart networks," in *the 17th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2011, pp. 13–22.

[8] A. Saifullah, D. Gunatilaka, P. Tiwari, M. Sha, C. Lu, B. Li, C. Wu, and Y. Chen, "Schedulability analysis under graph routing in WirelessHART networks," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2015, pp. 165–174.

[9] G. C. Buttazzo, *Hard real-time computing systems: predictable scheduling algorithms and applications*. Springer, 2005, vol. 24, 2nd edition.

[10] C. Wu, M. Sha, D. Gunatilaka, A. Saifullah, C. Lu, and Y. Chen, "Analysis of EDF scheduling for wireless sensor-actuator networks," in *the 22nd International Symposium of Quality of Service (IWQoS)*. IEEE, 2014, pp. 31–40.

[11] 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.

[12] H. Li, P. Shenoy, and K. Ramamritham, "Scheduling messages with deadlines in multi-hop real-time sensor networks," in *the 11th Real time and embedded technology and applications symposium (RTAS)*. IEEE, 2005, pp. 415–425.

[13] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha, "Flow-based real-time communication in multi-channel wireless sensor networks," in *European Conference on Wireless Sensor Networks (EWSN)*. Springer, 2009, pp. 33–52.

[14] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly, "Distributed multi-hop scheduling and medium access with delay and throughput constraints," in *the 7th annual international conference on Mobile computing and networking (MobiCom)*. ACM, 2001, pp. 200–209.

[15] 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 *the 8th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2002, pp. 55–66.

[16] K. Karenos and V. Kalogeraki, "Real-time traffic management in sensor networks," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2006, pp. 422–434.

[17] 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, pp. 261–289, 2007.

[18] 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 *Real-Time Systems Symposium (RTSS)*. IEEE, 2003, pp. 298–307.

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

[20] Y. Gu, T. He, M. Lin, and J. Xu, "Spatiotemporal delay control for low-duty-cycle sensor networks," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2009, pp. 127–137.

[21] N. Pereira, B. Andersson, E. Tovar, and A. Rowe, "Static-priority scheduling over wireless networks with multiple broadcast domains," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2007, pp. 447–458.

[22] O. Chipara, C. Lu, and G.-C. Roman, "Real-time query scheduling for wireless sensor networks," *IEEE transactions on computers*, vol. 62, no. 9, pp. 1850–1865, 2013.

[23] T. Abdelzaher, S. Prabh, and R. Kiran, "On real-time capacity limits of multihop wireless sensor networks," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2004, pp. 359–370.

[24] J. B. Schmitt and U. Roedig, "Sensor network calculus–a framework for worst case analysis," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*. Springer, 2005, pp. 141–154.

[25] A. Saifullah, S. Sankar, J. Liu, C. Lu, B. Priyantha, and R. Chandra, "CapNet: Exploiting wireless sensor networks for data center power capping," *ACM Transactions on Sensor Networks*, 2018.

[26] A. Saifullah, S. Sankar, J. Liu, C. Lu, R. Chandra, and B. Priyantha, "CapNet: A real-time wireless management network for data center power capping," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2014, pp. 334–345.

[27] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-time scheduling for WirelessHART networks," in *Real-Time Systems Symposium (RTSS)*. IEEE, 2010, pp. 150–159.

[28] P. Soldati, H. Zhang, and M. Johansson, "Deadline-constrained transmission scheduling and data evacuation in wirelesshart networks," in *European Control Conference (ECC)*. IEEE, 2009, pp. 4320–4325.

[29] H. Zhang, F. Osterlind, P. Soldati, T. Voigt, and M. Johansson, "Rapid convergecast on commodity hardware: Performance limits and optimal policies," in *the 7th Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON)*. IEEE, 2010, pp. 1–9.

[30] S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon, "Reliable and real-time communication in industrial wireless mesh networks," in *the 17th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2011, pp. 3–12.

[31] X. Zhu, P.-C. Huang, S. Han, A. K. Mok, D. Chen, and M. Nixon, "Roaminghart: A collaborative localization system on wirelesshart," in *the 18th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2012, pp. 241–250.

[32] V. P. Modekurthy, A. Saifullah, and S. Madria, "Distributed graph routing for wirelesshart networks," in *the 19th International Conference on Distributed Computing and Networking (ICDCN)*. ACM, 2018, pp. 24:1–24:10.

[33] T. F. Abdelzaher, V. Sharma, and C. Lu, "A utilization bound for aperiodic tasks and priority driven scheduling," *IEEE Transactions on Computers*, vol. 53, no. 3, pp. 334–350, 2004.

[34] S. Baruah, "Techniques for multiprocessor global schedulability analysis," in *the Real-Time Systems Symposium (RTSS)*. IEEE, 2007, pp. 119–128.

[35] M. Zimmerling, L. Mottola, P. Kumar, F. Ferrari, and L. Thiele, "Adaptive real-time communication for wireless cyber-physical systems," *ACM Transactions on Cyber-Physical Systems*, vol. 1, no. 2, p. 8, 2017.

[36] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *the 1st international conference on Embedded networked sensor systems (SenSys)*. ACM, 2003, pp. 126–137.

[37] M. Sha, D. Gunatilaka, C. Wu, and C. Lu, "Implementation and experimentation of industrial wireless sensor-actuator network protocols," in *the European Conference on Wireless Sensor Networks (EWSN)*. Springer, 2015, pp. 234–241.

[38] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *the 2nd international conference on Embedded networked sensor systems (SenSys)*. ACM, 2004, pp. 39–49.

[39] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Distributed channel allocation protocols for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, pp. 2264–2274, 2014.