Real-Time Scheduling for 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

WirelessHART is an open wireless sensor-actuator network standard for industrial process monitoring and control that requires real-time data communication between sensor and actuator devices. Salient features of a WirelessHART network include a centralized network management architecture, multi-channel TDMA transmission, redundant routes, and avoidance of spatial reuse of channels for enhanced reliability and real-time performance. This paper makes several key contributions to real-time transmission scheduling in WirelessHART networks: (1) formulation of the end-to-end real-time transmission scheduling problem based on the characteristics of WirelessHART; (2) proof of NP-hardness of the problem; (3) an optimal branch-and-bound scheduling algorithm based on a necessary condition for schedulability; and (4) an efficient and practical heuristic-based scheduling algorithm called Conflict-aware Least Laxity First (C-LLF). Extensive simulations based on both random topologies and real network topologies of a physical testbed demonstrate that C-LLF is highly effective in meeting end-to-end deadlines in WirelessHART networks, and significantly outperforms common real-time scheduling policies.

1 Introduction

Wireless Sensor-Actuator Networks (WSANs) are emerging as a new generation of communication infrastructure for industrial process monitoring and control [5]. Feedback control loops in industrial environments impose stringent end-to-end latency requirements on data communication. To support a feedback control loop, the network periodically delivers data from sensors to a controller and then delivers its control input data to the actuators within an end-to-end deadline. The direct effects of deadline misses in data communication may range from production inefficiency, equipment destruction to irreparable financial and environmental impacts. For instance, real-time monitoring of level measurement and control are required to avoid overfilling of oil tanks that may lead to serious economic loss and environmental threats. Moreover, stringent regulations for Health, Safety, and the Environment (HSE) are now being enforced in many countries [1]. HSE regulations require continuous monitoring of safety (shower, corrosive chemicals, and safety instrumentation) for workers around the plant so that help can be dispatched on time.

WirelessHART [3] has recently been developed as an open standard for WSANs for process industries. The standard has been instrumental in the adoption and deployment of wireless network technology in the field of process monitoring and control [14]. Drawing upon the insights and lessons learned from real-world industrial applications, WirelessHART has the following salient features specifically designed to meet the stringent real-time and reliability requirements of process monitoring and control: centralized network management architecture, multi-channel Time Division Multiple Access (TDMA), avoidance of spatial reuse of channels [5], and redundant routes. The unique characteristics of WirelessHART introduce a challenging real-time transmission scheduling problem.

In this paper, we study the real-time transmission scheduling problem of a set of periodic data flows with end-to-end deadlines from sensors to actuators in a WirelessHART network. This paper makes the following key contributions to address this problem:

- We formulate the real-time transmission scheduling problem based on the characteristics of WirelessHART networks and prove that it is NP-hard.
- We derive a necessary condition for schedulability in WirelessHART networks which can be used to effectively prune the search space for an optimal solution as well as to provide the insight for an efficient heuristic-based solution.
- We propose an optimal scheduling algorithm based on a branch-and-bound technique.

- We design a practical heuristic-based algorithm called **C**onflict-aware **L**east **L**axity **F**irst (C-LLF) that is efficient and, hence, can be used to handle dynamic changes of network topology and workloads.

The algorithms are evaluated using extensive simulations based on both random network topologies and real network topologies of a physical indoor testbed. Our results demonstrate that C-LLF is highly effective in meeting end-to-end communication deadlines in WirelessHART networks, while significantly outperforming the existing real-time scheduling policies. Moreover, it incurs minimal computational overhead and buffer space in the field devices, thereby making it a practical and effective solution for real-time transmission scheduling in WirelessHART networks.

The rest of the paper is structured as follows. The WirelessHART network model is presented in Section 2. Section 3 presents the problem formulation and the proof of NP-hardness. We derive the necessary condition for schedulability in Section 4. Section 5 presents the optimal scheduling based on branch-and-bound. C-LLF scheduling algorithm is presented in Section 6. Section 7 shows the simulation results. Section 9 is the conclusion.

2 WirelessHART Network Model

We consider a WirelessHART network consisting of field devices, one gateway, and a centralized network manager. The gateway provides the host system with access to network devices. Scheduling of transmissions is performed centrally at the network manager connected to the gateway which uses the network routing information in combination with communication requirements of the devices and applications. The network manager, then, distributes the schedules among the devices. The salient features of WirelessHART which make it particularly suitable for process industries are as follows:

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

Time Division Multiple Access (TDMA). Compared to CSMA/CA mechanism, TDMA protocols can provide predictable communication latencies making them an attractive approach for real-time communication. In WirelessHART networks, time is synchronized and slotted, and 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, effectively avoiding the spatial reuse of channels [5] to avoid transmission failure due to interference between concurrent transmissions. Thus, the maximum number of concurrent transmissions in the entire network at any slot cannot exceed the number of available channels [5]. This design decision improves the reliability at the potential cost of reduced throughput. The potential loss in throughput is also mitigated due to the small size of network.

Based on the above features, a WirelessHART network forms a mesh network modeled as a graph $G = (V, E)$, where the nodes V represent the network devices and E is the set of edges between the devices. That is, the set V consists of the gateway and the field devices. Every field device is either a sensor node, or an actuator, or both. An edge $e = (u, v)$ exists in the graph if and only if nodes u and v can communicate reliably with each other. Each $u \in V$ is able to send and receive packets, and to route packets for other network devices. For a transmission $\tau_i = \overrightarrow{uv}$ happening along an edge (u, v), device u is designated as the *sender* and device v the *receiver*.

A device cannot both transmit and receive at the same time slot. Two transmissions with the same intended receiver at a slot interfere each other. Hence, two transmissions $\tau_i = \overrightarrow{uv}$ and $\tau_j = \overrightarrow{wz}$ are *conflicting* and cannot be scheduled in the same slot, if $(u = w) \vee (u = z) \vee (v = w) \vee (v = z)$. A set of transmissions is *mutually exclusive* if every pair of transmissions in the set is conflicting.

3 Problem Formulation

In the *real-time scheduling for a WirelessHART network* $G = (V, E)$, we consider N end-to-end flows $F = \{F_1, F_2, \cdots, F_N\}$. Each flow $F_i \in F$ periodically generates a packet that originates at a network device $u \in V$, called the *source* of the packet, passes through the gateway,

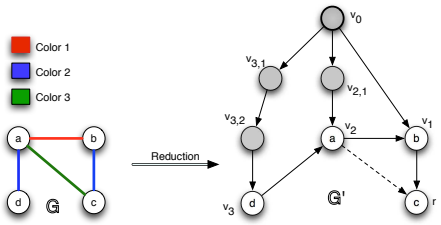


Figure 1. Reduction from edge-coloring

and ends at a network device $v \in V - \{u\}$, called the *destination* of the packet, within a deadline. The source and destination are characterized to be a sensor node and an actuator, respectively. From a source to a destination, there may exist more than one route, and the packet is delivered to the destination through each of these routes. The *release time* of a packet is the earliest time slot when it is ready to be scheduled. For a packet released at slot k and delivered to a destination at slot j through a route, its *end-to-end latency* through this route is $j - k + 1$. For flow F_i, its *end-to-end latency* L_i is the maximum end-to-end latency among all the packets generated by F_i.

Each flow F_i is, thus, characterized by a period P_i, a deadline D_i where $D_i \leq P_i$, and a set of routes Φ_i. A packet generated by F_i is routed through each $\phi \in \Phi_i$ that connects the source node to a destination node through the gateway. Thus, given the set of flows F, our objective is to schedule all transmissions in m channels such that $L_i \leq D_i, \forall F_i \in F$.

For the above problem, a scheduling algorithm $\mathbb{A}$ is called *optimal*, if $\mathbb{A}$ can schedule all transmissions whenever a feasible schedule (where no deadline is missed) exists. In the following, we prove that the problem is NP-hard by proving that its decision version is NP-complete.

Theorem 1. *Given a real-time scheduling problem for a WirelessHART network, it is NP-complete to decide whether it is schedulable or not.*

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

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

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

4 Necessary Condition for Schedulability

In this section, we establish a necessary condition for schedulability. This condition can be used to effectively prune the search space in a branch-and-bound algorithm. It also provides the key insights for efficient heuristic scheduling policies. In a WirelessHART network, the conflicting transmissions play a major role in the communication delays and the schedulability of the flows. The delays caused by conflicting transmissions are especially high near the gateway where all the flows converge creating a hot spot.

We first define some terminologies used in the necessary condition analysis. For the given set of flows, let T be the

hyper-period, i.e., the *least common multiple* of the periods of flows. It is sufficient to find a schedule for transmissions of packets generated no later than slot T. We use $p_{i,j}$ to denote the j-th packet, $0 \leq j < T/P_i$, generated by flow F_i. For packet $p_{i,j}$, its *release time* $R_{i,j} = P_i * j + 1$, and the *absolute deadline* $D_{i,j} = R_{i,j} + D_i - 1$.

From the release time and deadline of a packet, we can also derive a deadline and an anticipated release time for every transmission of the packet. For packet $p_{i,j}$, let $\tau_k = \vec{uv}$ be a transmission of $p_{i,j}$ through a route connecting its source to a destination such that the destination is $post_k$ hops away from node v. Since $D_{i,j}$ is the deadline of packet $p_{i,j}$, transmission τ_k needs to happen no later than slot $D_{i,j} - post_k$. Therefore, we can define the *deadline* of transmission τ_k as $d_k = D_{i,j} - post_k$. At a time slot s, let packet $p_{i,j}$ requires $pre_{k,s}$ transmissions before transmission $\tau_k = \vec{uv}$ can happen. That is, packet $p_{i,j}$ is $pre_{k,s}$ hops away from node u on its route at slot s. At slot s, a transmission is said to be *released* and, hence, is ready to be scheduled, if its preceding transmission is already scheduled before slot s. Therefore, unlike a packet, the exact release time of a transmission cannot be determined in advance (except for a packet's first hop transmission). Instead, at time slot s, we define the *anticipated release time* of transmission τ_k as $r_k = R_{i,j} + pre_{k,s} + max(s - R_{i,j}, 0)$.

Now we analyze the time demand of a packet for scheduling its transmissions in different time windows. For transmission τ_k of packet $p_{i,j}$, we call the time window $[r_k, d_k]$ the *lifetime* of transmission τ_k meaning that τ_k can happen no earlier than slot r_k and no later than slot d_k. Therefore, in window $[r_k, d_k]$, packet $p_{i,j}$ must need at least one time slot. If $pre_{k,s} > 0$ at slot s, then the lifetime of $p_{i,j}$'s transmission that must precede τ_k is $[r_k - 1, d_k - 1]$. Similarly, if $post_k > 0$, then the lifetime of $p_{i,j}$'s transmission that is preceded by τ_k is $[r_k + 1, d_k + 1]$. Thus, at any time slot s, we can conclude that packet $p_{i,j}$ needs at least: *(a)* 1 slot in window $[r_k, d_k]$; *(b)* 2 slots in window $[r_k - 1, d_k]$, if $pre_{k,s} > 0$; *(c)* 2 slots in window $[r_k, d_k + 1]$, if $post_k > 0$; *(d)* 3 slots in window $[r_k - 1, d_k + 1]$, if $pre_{k,s} > 0$ and $post_k > 0$. For τ_k, these time windows are denoted by:

$$\Omega(\tau_k) = \{[r_k - \beta_1, d_k + \beta_2] \mid 0 \leq \beta_1, \beta_2 \leq 1\} \qquad (1)$$

Let Γ be the set of transmissions of all the packets released no later than slot T (hyper-period). At slot s, $\Gamma_s \subseteq \Gamma$ denotes the set of unscheduled transmissions. Considering a transmission $\tau_k \in \Gamma_s$ of packet $p_{i,j}$, we know a lower bound of the time demand of $p_{i,j}$ in every window in $\Omega(\tau_k)$ from the above analysis. Again, a window $[a, b] \in \Omega(\tau_k)$ may contain another window $[a', b'] \in \Omega(\tau_{k'})$ (i.e., $a \leq a'$ and $b' \leq b$) of another transmission $\tau_{k'}$. Taking into account the lower bounds of time demand of every packet in window $[a, b] \in \Omega(\tau_k)$ yields a tighter lower bound of the number of transmissions by the packets in window $[a, b]$. Since, the total number of transmissions that can be accommodated in a time window is limited by the conflicting transmissions as well as the number of available channels, this time window analysis leads to a necessary condition for the schedulability. Intuitively, analyzing the lower bound of time demands in all windows $[x, y],\ 1 \leq x \leq y \leq T$ will lead to a strong necessary condition. But there are $O(T^2)$ such windows, thereby making the analysis computationally very expensive. However, a time window that does not contain any transmission's lifetime is useless in the analysis. Besides, the number of transmissions in a window is finite. As a result, the larger the numbers β_1 and β_2 are, the less effective the window $[r_k - \beta_1, d_k + \beta_2]$ is for necessary condition analysis. To balance between the complexity and effectiveness, we limit our analysis to $\beta_1 \leq 1$ and $\beta_2 \leq 1$ (in Equation 1).

We now derive the necessary condition for schedulability. Let $\psi^k_{a,b}$ be the number of transmissions in the largest set of mutually exclusive transmissions containing transmission τ_k such that the lifetime of each of these transmissions is contained in window $[a, b]$. Let $q_{a,b}$ be the total number of transmissions whose lifetimes are contained in $[a, b]$. For window$[a, b]$ and a transmission τ_k whose lifetime is contained in $[a, b]$, we define $\Delta^k_{a,b}$ as follows:

$$\Delta^k_{a,b} = (b - a + 1) - max(\psi^k_{a,b}, \left\lceil \frac{q_{a,b}}{m} \right\rceil) \qquad (2)$$

Let $\mu(\tau_k)$ be the minimum $\Delta^k_{a,b}$ among all $[a, b] \in \Omega(\tau_k)$.

$$\mu(\tau_k) = min(\{\Delta^k_{a,b} \mid [a, b] \in \Omega(\tau_k)\}) \qquad (3)$$

Based on above time window analysis, Theorem 2 establishes a strong necessary condition for schedulability.

Theorem 2. *For a set of flows F, let Γ_s be the set of unscheduled transmissions at slot s. If these transmissions are schedulable, then* $min(\{\mu(\tau_k) | \tau_k \in \Gamma_s\}) \geq 0$.

Proof. Let $\mathbb{S}$ be a feasible schedule of these transmissions where all the flows meet their deadlines. Time window $[a, b]$ can accommodate at most $b - a + 1$ mutually exclusive transmissions, irrespective of how many channels are available. Again, time window $[a, b]$ can accommodate at most $m * (b - a + 1)$ transmissions in total. But, for $q_{a,b}$ transmissions that must happen in window $[a, b]$, at least $\lceil \frac{q_{a,b}}{m} \rceil$ time slots are required. Again, for every transmission τ_k among these, there are $\psi^k_{a,b}$ transmissions each of which must be scheduled on a different time slot. That is, at least $max(\psi^k_{a,b}, \lceil \frac{q_{a,b}}{m} \rceil)$ time slots are required to accommodate the transmissions in window $[a, b]$. At any time slot s, the *laxity* of a packet $p_{i,j}$ can be defined as $(D_{i,j} - s + 1) - h_{i,j}$, where $h_{i,j}$ is the remaining number of transmissions of $p_{i,j}$ through its route. The *Laxity of schedule* $\mathbb{S}$ is the minimum laxity among all packets. The value

$\mu(\tau_k) = min(\{\Delta^k_{a,b} | \ [a,b] \in \Omega(\tau_k)\})$ is an upper bound of the schedule laxity of $\mathbb{S}$. Thus, $min(\{\mu(\tau_k)|\tau_k \in \Gamma_s\})$ indicates a tighter upper bound. Since $\mathbb{S}$ is a feasible schedule, $min(\{\mu(\tau_k)|\tau_k \in \Gamma_s\}) \geq 0$. □

5 Optimal Branch-and-Bound Scheduling

In this section, we present a scheduling algorithm based on branch-and-bound (B&B). Our B&B scheduling algorithm exploits the necessary condition established in Theorem 2 to effectively discard infeasible branches in the search space. It is optimal and complete in that it guarantees to find a schedule whenever a feasible one exists. The optimal B&B uses a search tree, where every node corresponds to a partial schedule that may or may not lead to a complete feasible schedule. For decision making at every node, the algorithm estimates an upper bound of the laxity of the schedule that the node may lead to. The *laxity of a packet* is its remaining time slots minus its remaining number of transmissions, and the *laxity of a schedule* is the minimum laxity among all packets. According to Theorem 2, for transmissions Γ_s to be scheduled on or after slot s, following is an upper bound (UB) of its schedule's laxity:

$$UB = min(\{\mu(\tau_k)|\tau_k \in \Gamma_s\}) \quad (4)$$

Step 0. Γ= set of all transmissions. $LB \leftarrow 0$;

Step 1. Compute *UB* for Γ. If *UB* < *LB* then stop since the given instance is unschedulable. Otherwise, create an empty schedule. Call this node the parent node. Find the released transmissions.

Step 2. For every valid subschedule of released transmissions, create a new child node. For each node, append the subschedule to the parent schedule and create new set of released transmissions. Compute *UB* for this node.

Step 3. If steps 4 and 5 have been performed for all childnodes then close the parent and go to step 6, otherwise, select the next child.

Step 4. If no unscheduled transmission is left, then stop, a feasible solution has been found.

Step 5. If *UB* < *LB*, then close this child node. Otherwise, create next released set of transmissions. Go to Step 2.

Step 6. Select a node among the open nodes. Call this node the parent node and go to step 2.

Algorithm 1: Optimal B&B Scheduling Algorithm

The search globally maintains a lower bound (LB) of schedule laxity as 0. Computing *UB* at a node (using Equation 4) gives one of these two decisions: *unschedulable* or *may be schedulable*. Specifically, if *UB* <*LB* at a node, it is guaranteed that this node will not lead to any feasible schedule and, hence, is discarded without further consideration. In contrast, if *UB* ≥*LB*, then this node may lead to a feasible solution and, hence, is expanded further. The algorithm terminates as soon as it finds a feasible complete schedule that meets all deadlines. If the original problem is infeasible, the algorithm will also terminate as soon as it determines that this is the case.

The search tree has as its root node an empty schedule along with all unscheduled transmissions. If it turns out that *UB* < *LB* at the root, then we terminate immediately with *unschedulable* decision. Otherwise, we determine the released transmissions at the first slot. For every valid subschedule of the released transmissions, we create a successor node that appends its subschedule to the schedule determined at the parent. By a *valid subschedule* we mean a subset of released transmissions that can be scheduled in current slot. Considering all unscheduled transmissions, the algorithm computes *UB* at this node to decide whether they are unschedulable or may be schedulable. If *UB* < *LB*, then this child is closed. Otherwise, we calculate the transmissions that are to be released for the next slot and the node is expanded further. We continue to create new nodes in the search tree until we either find a feasible solution, or until there exists no unexpanded node for which *UB* ≥ *LB*. In the latter case, no feasible valid solution exists. The steps of our optimal B&B are presented as Algorithm 1.

6 Conflict-aware Least Laxity First

While the B&B algorithm presented in Section 5 is optimal, its execution time may limit its applicability to dynamic environments where network topology changes frequently requiring the schedule to be recomputed quickly. In this section, we present a simple and efficient scheduling policy that is suitable for dynamic environments.

While the traditional real-time scheduling policies such as Least Laxity First (LLF) have been effective in end-to-end real-time scheduling over wired networks, such traditional policies do not deal with conflicts between transmissions in wireless networks. Since conflicting transmissions must be scheduled in different time slots, transmission conflicts contribute significantly to the communication delays in wireless networks. In WirelessHART networks, transmission conflicts can play a major role in schedulability even for moderate workloads due to the high degree of conflicts near the gateway. Moreover, different nodes experience different degree of conflicts as different nodes have different number of neighbors in a routing graph. The gateway and the nodes with high connectivity in the routing graph tend to experience significantly higher degrees of conflicts. *Hence, scheduling algorithms for WirelessHART networks must be cognizant of conflicts between transmissions*.

Based on this key insight into the WirelessHART networks, we present an efficient scheduling policy called *Conflict-aware Least Laxity First* (C-LLF). It uses *conflict-aware laxity* of every released transmission as the decision variable. The *conflict-aware laxity* of a transmission

is determined by considering the length of time windows in which the transmission must be scheduled as well as the potential conflicts that the transmission may experience in these windows. That is, the approach combines LLF and the degree of conflicts associated with a transmission. Thus, it can schedule a transmission while the remaining ones are likely to retain the necessary condition established in Theorem 2. Specifically, the algorithm identifies some critical time windows in which too many conflicting transmissions have to be scheduled, thereby determining the criticality of each released transmission. Criticality of a transmission is quantified by its conflict-aware laxity. Transmissions exhibiting lower conflict-aware laxity are assessed to be more critical. C-LLF gives the highest priority to the transmissions exhibiting lower conflict-aware laxity.

Now, for a transmission $\tau_k = \vec{uv}$ at slot s, we derive an expression (Equation 7) to compute its conflict-aware laxity denoted by λ_k^s. The transmissions whose lifetimes intersect with τ_k's lifetime are the potential sources of conflict while trying to schedule τ_k. We consider a subset of these transmissions to compute λ_k^s efficiently and effectively for transmission $\tau_k = \vec{uv}$. The subset consists of the transmissions that involve node u. Since we have to consider these transmissions until they are scheduled or until their deadlines are past, we consider the time windows that start at current slot and ends at their deadlines. For transmission $\tau_k = \vec{uv}$, let Λ_k^u be the set of deadlines of transmissions that involve node u and whose lifetimes intersect with the lifetime of τ_k, i.e.,

$$\Lambda_k^u = \{d_j | \tau_j = \vec{uz} \text{ or } \vec{zu},\ r_k \leq r_j \leq d_k\} \tag{5}$$

Since conflicting transmissions have to be scheduled on different slots, at slot s, we assess the criticality of window $[s, b]$, for every $b \in \Lambda_k^u$, by the difference $\delta_{s,b}$ between its length and the considered number of conflicts in it. That is,

$$\delta_{s,b} = (b - s + 1) - \sigma_{s,b}^u,\ \text{for } b \in \Lambda_k^u \tag{6}$$

with $\sigma_{s,b}^u$ being the number of transmissions that involve node u in time window $[s, b]$. That is, $\sigma_{s,b}^u$ counts every transmission $\tau_j = \vec{uz}$ or $\vec{zu}$ with $s \leq r_j$ and $d_j \leq b$, where $[r_j, d_j]$ is the lifetime of τ_j.

Now according to Equation 6, a smaller value of $\delta_{s,b}$ indicates that transmission τ_k will be conflicting with too many transmissions in a short time window. Therefore, the *conflict-aware laxity* λ_k^s of transmission $\tau_k = \vec{uv}$ at slot s is defined as the minimum $\delta_{s,b}$ over all $b \in \Lambda_k^u$ as follows:

$$\lambda_k^s = min(\{\delta_{s,b} | b \in \Lambda_k^u\}) \tag{7}$$

Therefore, the smaller the value of λ_k^s is, the more critical transmission τ_k is.

At every time slot s, C-LLF computes the conflict-aware laxity λ_k^s for every released transmission τ_k. Once it is calculated for every released transmission, the transmission with the smallest conflict-aware laxity is scheduled first. If there is a tie, then the transmission (among those having the smallest conflict-aware laxity) that has the earliest deadline is selected to schedule on an available channel. Any further tie is broken arbitrarily. Every released transmission that conflicts with the scheduled one cannot be scheduled in this slot. If there remains an unassigned channel, then the transmission with the next smallest conflict-aware laxity among the remaining released transmissions is picked. Similarly, the tie is broken first by the earliest deadline and then arbitrarily, and the transmissions conflicting with the scheduled one are no more considered for current slot. For the current slot, the same thing is repeated until no free channel is available, or no ready transmission is conflict-free with the scheduled ones in this slot, or there is no released transmission unscheduled. Then the schedule is performed for the next slot in the same way.

Input: $\Gamma \leftarrow$ transmissions of packets released no later than hyper-period T ; $m \leftarrow$ total channels;
Output: $S[1 \cdots T][0 \cdots m-1]$; `/* schedule */`
$s \leftarrow 1$; `/* initialize time slot */`
$\Gamma_s \leftarrow \Gamma$; `/* Unscheduled transmissions */`
while $(\Gamma_s \neq \emptyset)$ **do**
  $Released(s) \leftarrow$ set of released transmissions at slot s;
  $ch \leftarrow 0$; `/* initialize channel offset */`
  for (*each* $\tau_k \in Released(s)$) **do** Compute λ_k^s ;
  while $(ch < m)$ **do**
    $B \leftarrow$ transmissions with the smallest λ in $Released(s)$;
    $\tau^* \leftarrow$ a transmission with the shortest deadline among B;
    if *(τ^* misses deadline)* **then return** *unschedulable;*
    $S[s][ch] \leftarrow \tau^*$; $\Gamma_s = \Gamma_s - \{\tau^*\}$; $ch \leftarrow ch + 1$;
    Remove from $Released(s)$ every transmission conflicting with τ^*;
  end
  $s \leftarrow s + 1$; `/* go to next slot */`
end

Algorithm 2: C-LLF Scheduling Algorithm

As shown in the pseudo code (Algorithm 2), C-LLF outputs the schedule as a 2-dimensional array $S[1 \cdots T][0 \cdots m-1]$. The algorithm terminates with the *unschedulable* decision if, for a transmission τ_k with deadline d_k, it determines that $s > d_k$ at any slot s. When a transmission τ^* is assigned slot s and a channel offset ch, $0 \leq ch < m$, the schedule is recorded as $S[s][ch] = \tau^*$. In WirelessHART, the channel offset is then mapped to a physical channel for slot s.

Complexity analysis. C-LLF is a pseudo-polynomial time algorithm as analyzed below. There can be at most $O(N)$ released transmissions at a time slot considering a constant number of routes for every flow. At slot s, to calculate the conflict-aware laxity λ_k^s for a transmission τ_k, we need to consider all packets released within window $[r_k, d_k]$ (lifetime of τ_k). Therefore, the number of transmissions that

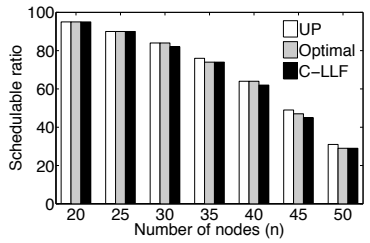
should be considered for calculating λ_k^s is upper bounded by $O(N.H.D/P)$, where H is the maximum length among all routes, D is the maximum relative deadline and P is the minimum period among all flows. Since at most two transmissions of a packet can involve a node in any direction (sensor to gateway or gateway to actuator) on a route, calculating λ_k^s takes $O(N.H.D/P)$ time. The time for calculating the conflict-aware laxities for all released transmissions and sorting them is $O(N^2.H.D/P)$. Since the schedule has to be calculated only up to the hyper-period T, the complexity of C-LLF is thus $O(N^2.T.H.D/P)$.

7 Evaluation

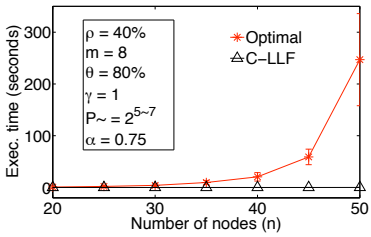
Baselines. We compare our algorithms against several well-known real-time scheduling policies: *(a) Deadline Monotonic (DM)* schedules using the relative deadline of the flows; transmission that belongs to the flow with the smallest relative deadline being scheduled first; *(b) Earliest Deadline First (EDF)* schedules a transmission based on its packet's absolute deadline; *(c) Proportional Deadline monotonic (PD)* schedules a transmission based on its packet's *relative subdeadline* defined as the relative deadline (of its flow) divided by the total number of transmissions along its route; *(d) Earliest Proportional Deadline first (EPD)* schedules a transmissiont based on its packet's *absolute subdeadline* defined, at every slot, as its remaining time divided by the remaining number of transmissions; *(e) Least Laxity First (LLF)* schedules a transmission based on its packet's *laxity* defined as its remaining time minus its remaining number of transmissions.

Metrics. Following metrics are used for performance analysis. *(a) Schedulable ratio* is measured as the percentage of test cases for which an algorithm is able to find a feasible schedule. *(b) Buffer size* is the maximum number of packets buffered at a node when transmissions are scheduled. *(c) Execution time* is the total time required to create a schedule for packets generated within the hyper-period. We plot the average execution time (along with the 95% confidence interval) of the schedulable cases out of 100 runs.

Simulation Setup. A fraction (θ) of nodes is used as sources and destinations of the flows. The sets of sources and destinations are disjoint. The node with the highest number of neighbors is 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. The periods of flows are harmonic and are generated randomly in a given range denoted by $P_\sim = 2^{i\sim j}, i \leq j$. The relative deadline of a flow F_i with period P_i is generated randomly



(a) Schedulable ratio



(b) Execution time

Figure 2. Scheduling with the B&B and C-LLF under varying network sizes

in the range between H_i and $\alpha * P$, for $0 < \alpha \leq 1$, with H_i being the maximum length among the routes associated with F_i. In every figure, we show the parameter setups of the corresponding experiment. The algorithms have been written in C and the tests have been performed on a Mac OS X machine with 2.4 GHz Intel Core 2 Duo processor. The notations used in this section are summarized in Table 1.

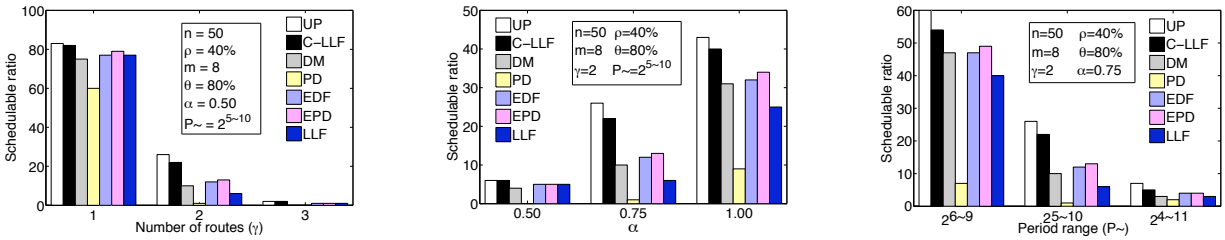
n :	Number of nodes of a network
m :	Number of channels
ρ :	Edge-density of a network
θ :	Fraction of sources and destinations
γ :	Number of routes between every source and destination
$P_\sim$:	Period range
α :	Maximum route length$\leq$deadline$\leq \alpha*$period

Table 1. Notations

7.1 Simulations with Random Topologies

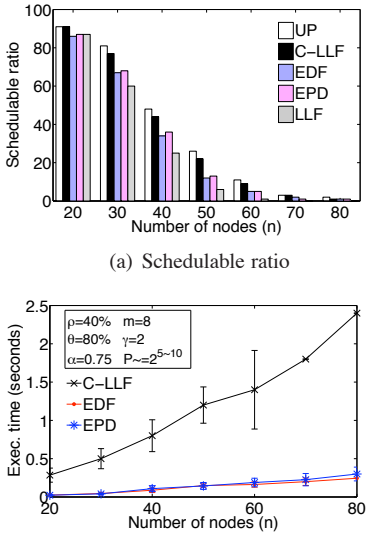
Generating networks. Given the number of nodes (n) and edge-density (ρ), we generate random networks. A network with n nodes and $i\%$ edge-density has a total of $(n(n-1)*i)/(2*100)$ bidirectional edges. The edges are chosen randomly and assigned PRR randomly in the range $0.80 \sim 1.0$. We keep regenerating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

Optimal B&B and C-LLF. We evaluate the tightness of our necessary condition based on the percentage of test cases that pass the necessary condition but are found to be



(a) Varying number of routes (γ) (b) Varying α (c) Varying periods ($P_\sim$)

Figure 3. Schedulable ratio of C-LLF and baselines



(a) Schedulable ratio

(b) Execution time

Figure 4. Comparison under varying network sizes

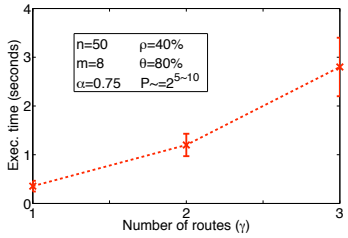


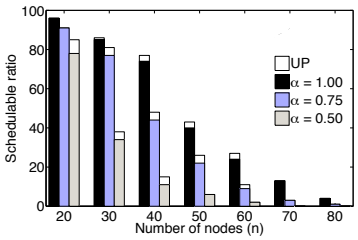
Figure 5. Execution time of C-LLF under varying number of routes (γ)

unschedulable under the B&B. The percentage of test cases that pass the condition is denoted by UP and it indicates an upper bound of schedulable ratio. Figure 2 shows the performance of the B&B, C-LLF, and the tightness of our necessary condition using 8 channels in different sized networks of 40% edge-density with $\gamma = 1$ and θ=80%. θ=80% implies that 40% of the total nodes are sources while another 40% are destinations of flows. Figure 2(a) shows that the number of test cases that satisfy necessary condition but are not schedulable by the B&B is less than 3%. It indicates that the necessary condition in Theorem 2 is highly effective for pruning the search space in the B&B. Figure 2(a) also shows that C-LLF is highly competitive against the B&B in terms of schedulable ratio. As shown in Figure 2(b), while the B&B incurs reasonable execution times for 20 and 30 nodes, its execution time increases dramatically as the number of nodes increases. Its average execution time is 247 seconds for 50 nodes, making it less desirable for relatively larger networks with frequent topology changes. In contrast, C-LLF remains highly efficient for varying network sizes and maintains an average execution time of 0.06 seconds for 50-node networks. This result indicates that C-LLF can be used as an effective online scheduling algorithm in face of dynamic network topologies.

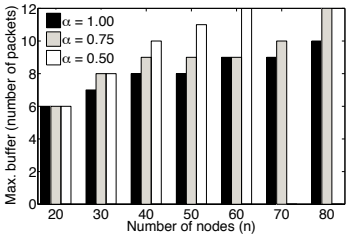
Comparison with real-time heuristics. Now we compare our algorithm with the baselines. As the B&B has significantly longer execution time, we exclude the B&B in this set of simulations. As the last set of simulations showed that the necessary condition is fairly tight in practice, we plot UP as a conservative upper bound for the schedulable ratio under any scheduling algorithm. Figures 3(a), 3(b), and 3(c) show the schedulable ratios achieved by C-LLF and the baselines under varying γ, α, and periods, respectively, in 50-node networks. C-LLF consistently outperforms all baselines under all tested configurations. Moreover, its schedulable ratio remains close to UP.

Since the performances of PD and DM are less competitive, we no more present them. Figure 4 shows the performance of C-LLF against the baselines under varying number of nodes in the network. Figure 4(a) indicates that the schedulable ratio of C-LLF is higher than those of the baselines and is close to UP even when the number of nodes is 80. Figure 4(b) shows that the baselines are much faster than C-LLF. However, for 80 nodes, the average time of C-LLF is less than 2.5 seconds which is a reasonable time for computing schedules for WirelessHART networks.

Scalability of C-LLF. Figure 5 shows that the execution time of C-LLF increases sharply with the increase of γ. This is reasonable since increasing γ increases the workload significantly. However, in less than 3 seconds, can it complete scheduling when $\gamma = 3$. No feasible solution is found if we further increase γ. Figure 6 shows the scalability of C-



(a) Schedulable ratio



(b) Maximum buffer

Figure 6. Scheduling by C-LLF under varying network sizes

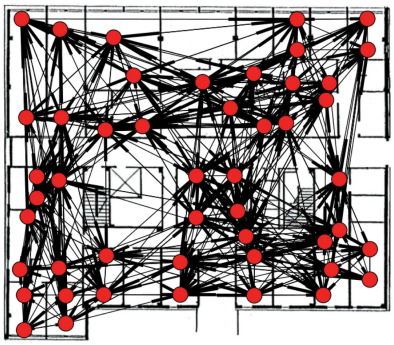
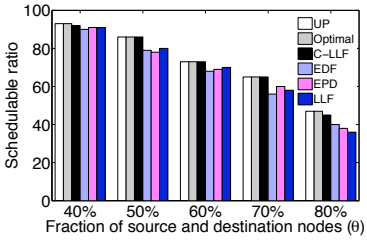


Figure 7. The testbed topology with a transmission power of 0 dBm

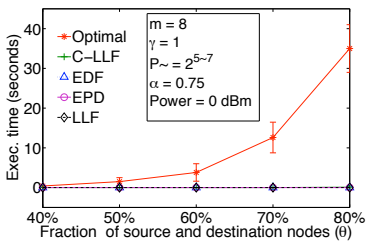
LLF under increasing number of nodes and with different values of α when $\gamma = 2$. The Figure 6(a) shows that the schedulable ratio of C-LLF remains close to UP even with tighter deadlines (i.e., with lower α) in different sized networks. For 20, 70, and 80 nodes, C-LLF performs like an optimal algorithm as it achieves schedulable ratio equal to UP when $\alpha \geq 0.75$. Figure 6(b) shows the maximum buffer size required at a node when transmissions are scheduled under C-LLF. The maximum number packets buffered at a node is 12 for a 80-node network. It indicate that buffer size required at a node does not dramatically increase with the increase of network size.

7.2 Simulations with Testbed Topologies

Network Topology. We evaluate our algorithms on the network topologies of a physical indoor testbed [2] (in Bryan



(a) Schedulable ratio



(b) Execution time

Figure 8. Scheduling with the B&B, C-LLF, and baselines under varying number of sources and destinations

Hall of Washington University in St Louis) consisting of 45 TelosB motes equipped with Chipcon CC2420 radios which are compliant with the IEEE 802.15.4 standard. At each transmission power level, every node broadcasts 50 packets while its neighbors record the sequence numbers of the packets they receive. After a node completes sending its 400 packets, the next sending node is selected in a round-robin fashion. This cycle is repeated giving each node 5 rounds to transmit 400 packets in each round. Figure 7 shows the network topology with transmission power of 0 dBm. Every link with a higher than 80% packet reception ratio is considered a reliable link and drawn in Figure 7. We test the algorithms on the topologies at 4 different power levels.

Scheduling performance. For the network topology at 0 dBm, Figure 8 shows the performances of C-LLF, the B&B, and the baseline heuristics under varying θ. In Figure 8(a), we see that the schedulable ratio of C-LLF is close to that of the B&B and is better than those of the baselines. It also shows that UP and the schedulable ratio of B&B are very close. Figure 8(b) shows that the average execution time of the B&B is 33 seconds while it is close to 0 seconds for C-LLF and the baselines when $\theta = 80\%$. Figure 9 shows the performance comparison under different power levels. As expected, the schedulable ratios of all algorithms slightly decrease when the transmission power level decreases. However, at every power level, we can see that the schedulable ratio of C-LLF is close to that of the B&B as well as UP which demonstrates the effectiveness of C-LLF in meeting deadlines in WirelessHART networks.

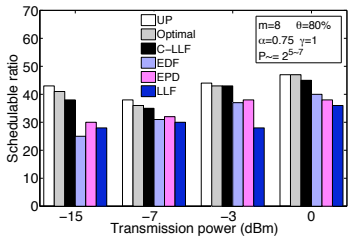


Figure 9. Schedulable ratio under different power level

8 Related Works

Although real-time transmission scheduling in wireless networks has been studied in the literature [15], few of previous works are applicable to WirelessHART networks. Several papers [8–10, 12, 16] proposed scheduling based on CSMA/CA MAC protocols. In contrast, WirelessHART adopts a TDMA-based approach to achieve predictable latency bounds. Several others developed TDMA scheduling algorithms [4, 6, 7, 11], but did not consider multi-channel communication supported by WirelessHART.

Scheduling for WirelessHART networks has been investigated recently since the standard was ratified in 2007. Convergecast scheduling has been studied for simplified network models such as linear [17] and tree networks [13] with depth no greater than the number of channels. There are several fundamental differences between our work and these previous studies. First, we consider the general WirelessHART network model. Our algorithms support multi-path routing, whereas the previous research only considered a single route for each node. Besides, our algorithms can deal with arbitrary network topologies without any constraint on the length of routes. Second, our scheduling algorithms support real-time flows for feedback control loops, whereas previous research only considered data collection to the gateway. Finally and importantly, our algorithms aim to meet end-to-end deadlines which may differ based on the requirements of flows, while previous research focused on minimizing data collection latencies. Our research, therefore, addresses a more complicated scheduling problem suitable for process control in WirelessHART networks.

9 Conclusion

In this paper, we make key contributions to real-time transmission scheduling in WirelessHART networks: (1) formulation of the end-to-end real-time transmission scheduling problem based on the characteristics of WirelessHART, (2) proof of NP-hardness of the problem, (3) an optimal branch-and-bound scheduling algorithm based on a strong necessary condition, and (4) an efficient and practical heuristic-based algorithm called Conflict-aware Least Laxity First (C-LLF). The key insight underlying C-LLF is that it is important to incorporate transmission conflicts in scheduling policies for WirelessHART networks. Simulations based on both random and real network topologies demonstrate that C-LLF significantly outperforms the traditional real-time scheduling policies and that it is highly effective in meeting end-to-end communication deadlines.

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] `www.hse.gov.uk/pubns/regindex.htm`.
[2] `http://mobilab.wustl.edu/testbed`.
[3] WirelessHART specification, 2007. `http://www.hartcomm2.org`.
[4] 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*.
[5] D. Chen, M. Nixon, and A. Mok. *WirelessHART™ Real-Time Mesh Net. for Industrial Automation*. Springer, 2010.
[6] O. Chipara, C. Lu, and G.-C. Roman. Real-time query scheduling for wireless sensor networks. In *RTSS '07*.
[7] Y. Gu, T. He, M. Lin, and J. Xu. Spatiotemporal delay control for low-duty-cycle sensor networks. In *RTSS '09*.
[8] K. Karenos and V. Kalogeraki. Real-time traffic management in sensor networks. In *RTSS '06*.
[9] K. Karenos, V. Kalogeraki, and S. V. Krishnamurthy. A rate control framework for supporting multiple classes of traffic in sensor networks. In *RTSS '05*.
[10] H. Li, P. Shenoy, and K. Ramamritham. Scheduling messages with deadlines in multi-hop real-time sensor networks. In *RTAS '05*.
[11] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang. JiTS: Just-in-time scheduling for real-time sensor data dissemination. In *PERCOM '06*.
[12] 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*.
[13] P. Soldati, H. Zhang, and M. Johansson. Deadline-constrained transmission scheduling and data evacuation in WirelessHART networks. In *ECC '09*.
[14] J. Song, S. Han, A. Mok, D. Chen, M. Lucas, and M. Nixon. WirelessHART: Applying wireless technology in real-time industrial process control. In *RTAS '08*.
[15] J. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91(7):1002–1022, 2003.
[16] X. Wang, X. Wang, X. Fu, G. Xing, and N. Jha. Flow-based real-time communication in multi-channel wireless sensor networks. In *EWSN '09*.
[17] H. Zhang, P. Soldati, and M. Johansson. Optimal link scheduling and channel assignment for convergecast in linear WirelessHART networks. In *IEEE WiOpt '09*.