Summer 8-1-2013

End-to-End Delay Analysis for Wireless Control Networks under EDF Scheduling

Chengjie Wu
Washington University in St. Louis

Washington University in St. Louis

School of Engineering and Applied Science

Department of Computer Science and Engineering

Thesis Examination Committee:
Chenyang Lu, Chair
Yixin Chen
Christopher D. Gill

End-to-End Delay Analysis for Wireless Control Networks under EDF Scheduling

by

Chengjie Wu

A thesis presented to the Graduate School of Arts and Sciences
of Washington University in partial fulfillment of the
requirements for the degree of

Master of Science

August 2013
Saint Louis, Missouri

Contents

List of Tables

List of Figures

Acknowledgments

First, I would like to thank my advisor Professor Chenyang Lu for his continuous guidance and advice both in research and my personal growth. He has introduced me to important research areas, taught me how to find real-world high-impact questions and how to appreciate the beauty and simplicity in real research.

My sincere gratitude goes to my committee members Professor Yixin Chen and Professor Christopher D. Gill. Thanks for Professor Yixin Chen's patient advice on my research projects. Thanks for Professor Christopher D. Gill for serving as my committee member.

I am grateful to my current and previous intelligent colleagues in the our research group. My gratitude goes to Dr. Greg Hackmann, Professor Octav Chipara, Yong Fu, Abu Sayeed Saifullah, Mo Sha, Sisu Xi, Bo Li, Rahav Dor and Jing Li.

Finally, I would like to give my deepest gratitude to my parents and my wife for their endless love and support.

Chengjie Wu

Washington University in Saint Louis
August 2013

Dedicated to my parents and my wife

ABSTRACT OF THE THESIS

End-to-End Delay Analysis for Wireless Control Networks under EDF Scheduling

by

Chengjie Wu

Master of Science in Computer Science

Washington University in St. Louis, August 2013

Research Advisor: Professor Chenyang Lu

Process industries are starting to adopt multi-hop and multi-channel wireless control networks (WCNs) for process control applications. To meet the stringent real-time performance requirements of control systems, there is a critical need for fast end-to-end delay analysis to support online admission control of periodic real-time flows. While recent results on delay analysis for WCNs have focused on fixed-priority scheduling, this thesis presents the first end-to-end delay analysis for real-time flows in WCNs that schedule transmissions based on the Earliest Deadline First (EDF) policy, a widely used dynamic scheduling policy in real-time systems. This analysis provides safe end-to-end delay bounds for real-time flows and can be used for efficient admission control at run time. Simulations based on both random topologies and the topology of a wireless testbed demonstrate the effectiveness of our analysis for online admission control of real-time flows.

Chapter 1

Introduction

With the emergence of industrial standards such as WirelessHART [2] and ISA100 [12], process control industries are now moving towards Wireless Control Networks (WCNs). In a WCN, feedback control loops periodically deliver sensor data from sensors to controllers, and then deliver control messages from controllers to actuators through a wireless mesh network. Since excessive communication delay may lead to severe degradation of control performance or even instability of the control system, it is critical to estimate worst-case end-to-end communication delays for real-time flows in WCNs. Moreover, fast delay analysis is needed for online admission control and network reconfiguration in response to dynamic changes of channel conditions in industrial environments.

Recently, real-time transmission scheduling for WCNs has received attention [15–17, 20, 23, 24]. However, existing end-to-end delay analysis [15] for WCNs focuses on fixed priority transmission scheduling. While dynamic scheduling has been studied [17], to date there is no fast delay analysis for WCNs scheduled based on dynamic priority scheduling. Earliest Deadline First (EDF) policy is a widely adopted dynamic priority scheduling strategy for real-time systems [22] and has been shown to be an effective scheduling strategy for WCNs [17]. We focus our analysis on EDF because we can leverage existing schedulability analysis of EDF for CPU scheduling. Moreover, our simulation study shows EDF outperforms fixed priority scheduling with near optimal priority assignment while only slightly underperform state-of-the-art dynamic priority scheduling with no efficient schedulability analysis.

In this paper, we address the open problem of delay analysis for periodic flows in WCNs scheduled by EDF scheduling policy. In this problem, real-time flows periodically generate packets at sources which needed to be delivered to destinations within their respective

deadlines. Under the EDF policy, transmissions are scheduled based on the deadlines of the packets. Packets with earlier deadlines are assigned with higher priorities. A key feature of our analysis lies in a novel approach to combine two types of delays in a wireless control network: *contention delay* due to limited number of wireless channels, and the *conflict* delay caused by conflicts between concurrent wireless transmissions. Specifically, this paper (1) leverages real-time multiprocessor scheduling analysis for global EDF to derive contention delays, (2) integrates both conflict and contention delays in a holistic end-to-end delay analysis, and (3) reduces the pessimisms in admission control through tighter delay bounds on flows with tight deadlines.

We evaluate our delay analysis through simulations based on both random network topologies and topologies of a 69-node wireless sensor network testbed. The simulation results show that our sufficient schedulability tests are effective in terms of the acceptance ratio while providing safe end-to-end communication delays. We also provide a comprehensive simulation study that compares different existing real-time scheduling analyses in terms of both schedulability and the computation time.

Chapter 2

Wireless Control Network Model

We consider a Wireless Control Network (WCN) model based on the WirelessHART standard [2] with simplifications discussed in the end of this chapter. A WCN consists of a set of field devices and a gateway. A field device could be a sensor, an actuator, or both. Each device (field device or gateway) is equipped with a half-duplex omnidirectional radio transceiver, and cannot transmit or receive simultaneously. All devices and the gateway form a mesh network. The gateway is the bridge between the mesh network and the process control system. The WCN has a centralized architecture. All devices are managed by a centralized network manager connected to the gateway through wired connection. For industrial process control applications, controllers are also installed in a host that is wired connected to the gateway. All sensing data packets are delivered from sensors to the controllers. Then, the controllers send control packets to actuators. The network manager determines the routing based on topology information of the network. Scheduling of transmissions across the network is also generated by the network manager in a centralized fashion.

The WCN model adopts a Time Division Multiple Access (TDMA) MAC. All devices across the network are synchronized. The time is divided into 10 ms slots. Each time slot can accommodate one data packet transmission and its associated acknowledgment. The WCN employs multi-channel communication using the channels defined in IEEE 802.15.4 standard. To avoid internal interference, channel reuse is prevented. Each channel can only accommodate one transmission across the entire network in any time slot. As a result, the total number of concurrent transmissions in the network can not exceed the number of channels. While this conservative design adopted by WirelessHART reduces network throughput

and scalability, it helps enhance reliability and predictability that is important for industrial control applications.

As the first step toward real-time EDF scheduling analysis for WCNs, we make simplifying assumptions about routing. Instead of the graph routing approach employed by WirelessHART, we assume there exist one or more routes between every source and destination and a flow with N redundant routes can be treated as N separate flows for the purpose of transmission scheduling. Henceforth each flow refers to a flow over a single route. Under this simplified routing approach our delay analysis does not need to consider redundant routes made available by graph routing. By establishing the first delay analysis for EDF scheduling in WCNs, this work provides a foundation towards a practical analysis for WirelessHART networks with graph routing and EDF scheduling.

Chapter 3

Related Works

Real-time transmission scheduling in wireless networks has been well studied in the literature [21]. However, early research on real-time scheduling is not applicable to recent industrial WCN standards such as WirelessHART with multi-channel TDMA scheduling and a centralized architecture. For example, [9, 10, 13, 14] proposed real-time transmission scheduling algorithms for wireless sensor networks. Real-time capacity and communication delay of wireless sensor networks have been studied in [3, 19]. These works are targeted at data collection in wireless sensor networks instead of real-time flows in wireless control networks.

Some recent works [20, 23, 24] have studied the transmission scheduling for WCNs with linear or tree topologies. Transmission scheduling of real-time flows for arbitrary WCN topologies was studied in [17]. It presented a real-time scheduling algorithm based on branch-and-bound and a dynamic priority scheduling algorithm called C-LLF, but it did not present any delay analysis.

End-to-end delay analysis for fixed priority scheduling in WCN has been proposed in [15, 18]. The performance of fixed priority scheduling highly depends on the priority assignment, which is proven to be a difficult problem [16] and near-optimal priority assignment algorithms incurs significant delay when used online (e.g., for admission control or network reconfiguration). While dynamic priority scheduling represents an attractive alternative to fixed priority scheduling, end-to-end delay analysis for dynamic priority scheduling has not been studied for WCNs. EDF is a commonly used dynamic priority scheduling algorithm in real-time systems and has also been found to outperform common fixed priority scheduling algorithms in WCNs in previous studies [17].

EDF schedulability test for multiprocessor scheduling has been studied in several works [4–8, 11]. Goossens et al. [11] proposed a sufficient schedulability test which can be summarized as one inequation. Baker [4, 5] proposed a schedulability test by identifying the necessary conditions that a task job will miss its deadline. Bertogna et al. [7] proposed schedulability tests by bounding interferences a task job may suffer. They improved their analysis in [8] by an iterative algorithm. Baruah [6] claimed to improve the schedulability test by proposing a pseudo-polynomial analysis. However, none of them works for our problem, since transmission conflict is a unique property of real-time flow scheduling problem. In our approach, we will incorporate conflict delay into our schedulability analysis. We provide the first end-to-end delay analysis of EDF scheduling (and dynamic scheduling in general) for WCNs. Moreover, we provide a comprehensive simulation study that compares different existing real-time scheduling analyses in terms of both schedulability and the computation time.

Chapter 4

Problem Formulation

A WCN is modeled as a graph $G = (V, E)$, where the node set V represents the network devices and E is the set of links between these devices. The set V consists of the gateway and field devices. A device cannot both send and receive a packet in the same time slot. A transmission $u\vec{v}$ is associated with a link (u, v), a time slot and a channel. Device u is designated as the sender and device v as the receiver. Note that channel reuse is avoided in WCN, only one transmission can be scheduled on a channel in each time slot. If all available channels are assigned to transmissions, remaining transmissions have to be postponed to later slots. Because channel reuse is avoided, only one transmission will be scheduled on any channel in one time slot, there is no interference between transmissions in a time slot. However, two transmissions *conflict* with each other only if they share at least a node, because a radio interface can transmit or receive only one packet in a time slot. Specifically, two transmissions $u\vec{v}$ and $p\vec{q}$ are conflicting if $(u = p) \vee (u = q) \vee (v = p) \vee (v = q)$. Two conflicting transmissions cannot be scheduled in the same time slot.

A set of real-time periodic flows $F = \{F_1, F_2, \cdots, F_N\}$ need to be scheduled. Each flow F_i is associated with a period T_i, a relative deadline D_i, a source device s_i, a destination device d_i, a route ϕ_i. ϕ_i is composed by a sequence of links in the network. To ensure reliability, each transmission is scheduled κ times to overcome link failure. We define C_i as the *execution time* of F_i, which equals to the total number of transmissions scheduled for one packet of this flow. In this case, $C_i = |\phi_i|\kappa$, where $|\phi_i|$ is the length of ϕ_i. A constrained deadline model $D_i \leq T_i$ is followed here, hence different packets of the same flow can not exist in the network in the same time slot.

At the beginning of jth period, flow F_i releases a packet P_{ij} at source node s_i. Each packet P_{ij} needs to be delivered to the destination d_i through a sequence of transmissions along ϕ_i that are scheduled according to the EDF policy. Each packet is assigned with a priority based on its absolute deadline. The packet with earlier absolute deadline is assigned with higher priority. Transmissions of all packets are scheduled to m channels. At any time slot, among all ready transmissions which do not conflict with the scheduled transmissions, the transmission that belongs to the highest priority packet is scheduled on a channel among all available channels.

For a packet P_{ij}, if it is released from the source at time slot t_r and is delivered to the destination at slot t_d through its route, its *end-to-end delay* is defined as $r_{ij} = t_d - t_r + 1$. Here we define the *worst-case end-to-end delay* of flow F_i as R_i, which is the maximum end-to-end delay among all its packets.

A flow set $F = \{F_1, F_2, \cdots, F_N\}$ is *schedulable* under a scheduling algorithm $\mathcal{A}$, if $\mathcal{A}$ can schedule all transmissions that belong to F using m channels such that no deadline is missed. A schedulability test $\mathcal{S}$ for $\mathcal{A}$ is *sufficient* if any flow set which is tested to be schedulable by $\mathcal{S}$ is indeed schedulable by $\mathcal{A}$. Given the set of real-time flows F, our goal is to derive an upper bound on worst-case end-to-end delay of every flow. The delay analysis can then be used as a sufficient schedulability test $\mathcal{S}$ that can predict the schedulability of F under EDF.

All notations used in our analysis are summarized in Table 4.1.

F_k	A flow with index k
T_k	Period of flow F_k
D_k	Deadline of flow F_k
C_k	Total number of transmissions for one packet of F_k
R_k	Worst-case end-to-end delay of flow F_k
P_{kj}	jth packet of flow F_k
$I(k,i)$	Number of transmissions that belong to F_i and are scheduled in lifetime of a packet of F_k
$I^{conf}(k,i)$	Number of transmissions of F_i that introduce conflict delay to a packet of F_k
$I^{cont}(k,i)$	Number of transmissions of F_i that introduce contention delay to a packet of F_k
W_{ki}	Maximum conflict delay that a single packet of F_i could introduce to a single packet of F_k
$W_{ki}(\nu)$	Maximum conflict delay that the last ν transmissions of a single packet of F_i could introduce to the first ν transmissions of a single packet of F_k

Table 4.1: Notations

Chapter 5

Worst-Case End-to-End Delay Analysis

In this chapter, we present our worst-case end-to-end delay analysis for real-time flows under the EDF policy. A set of real-time flows are schedulable if every flow has a worst-case end-to-end delay that is less than or equals to its deadline. A packet P_{kj} of flow F_k is delayed if it has a transmission $\vec{uv}$ that is ready at a time slot t but not scheduled at t. As observed in [15], we categorize the delays that a packet may experience in a WSN into two types: *contention delay* and *conflict delay*.

- **Contention Delay** Since channel reuse is forbidden in a WCN, each channel can only accommodate one transmission across the network in each time slot. If all channels are assigned to transmissions of other packets, a packet suffers one time slot of contention delay because its ready transmission can not be scheduled onto any channel at this time slot.
- **Conflict Delay** Because of the half-duplex radio, two transmissions conflict with each other if they share a node (sender or receiver). Then only one of them can be scheduled at one time slot. Therefore, if a transmission conflicts with another transmission that has already been scheduled in current time slot, it has to be postponed to a later time slot, resulting in one time slot of conflict delay.

Previous work [15] has studied worst-case end-to-end delay of flows under fixed priority scheduling policy, but their analysis can not be applied to this work. As proven in [15],

the worst-case delay of a real-flow is the sum of the worst-case conflict delay and the worst-case contention under *fixed priority* scheduling. This property allows a divide and conquer approach that derives the upper bound of each type of delay. Unfortunately, this property does *not* hold under dynamic priority scheduling such as EDF. A key challenge tackled in our analysis is to find the worst-case scenario for a packet of a flow. Then we propose a worst-case end-to-end delay analysis based on our worst-case scenario analysis.

Consider a packet P_{kj} of flow F_k released at t_r with absolute deadline an $t_d = t_r - 1 + D_k$. We want to analyze the delay P_{kj} suffers from packets of all the other flows. We start with analyzing the delay caused by the packets of an arbitrary flow F_i, and then we include all flows into our analysis.

Given EDF is used to schedule transmissions of packets, we ignore packets of F_i that have absolute deadlines which are earlier than t_r or later than t_d. Packets with absolute deadlines earlier than t_r will finish their transmissions before P_{kj}'s release. Packets with absolute deadlines later than t_d have lower priorities than P_{kj}, hence they will not delay P_{kj}.

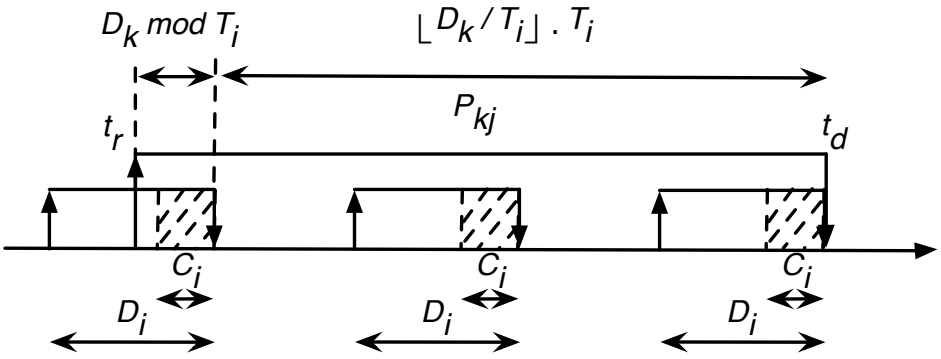


Figure 5.1: Worst-case scenario for packet P_{kj}

We first consider the scenario illustrated in Fig. 5.1. In the figure, the release time of a packet is indicated as an ascending arrow, and the deadline is indicated as a descending arrow. The dashed area is used to denote the time slots where transmissions are scheduled. In the worst-case scenario, the absolute deadline of one packet of F_i aligns with t_d. In this case, the workload of packets in flow F_i within P_{kj}'s lifetime is maximized. Because if we slide P_{kj}'s absolute deadline earlier, then the last packet of F_i in the figure would have an absolute deadline later than t_d, which makes it has a lower priority than P_{kj}. Also, if we slide P_{kj}'s

absolute deadline later, the workload of packets of F_i would decrease as well. We also assume all transmissions of a packet P_{ih}, whose absolute deadline is later than t_r and earlier than t_d, are scheduled in latest time slots. The third assumption is among these transmissions, those have potential to conflict with P_{kj}'s transmissions indeed introduce conflict delays.

Observation 1. *For a packet P_{kj} of flow F_k, P_{kj} meets the worst-case delay when the following conditions are true for every flow $F_i, i \neq k$:*

1. *The absolute deadline of a packet of F_i aligns with the absolute deadline of P_{kj}.*
2. *For any packet P_{ih} in flow F_i that has an absolute deadline later than t_r and no later than t_d, all its transmissions are scheduled at the very end of its scheduling window. In other words all transmissions are scheduled at the latest time slots before the absolute deadlines.*
3. *For any packet P_{ih} in flow F_i that has an absolute deadline later than t_r and no later than t_d, all its transmissions that may conflict with transmissions of P_{kj} indeed introduce conflict delay to P_{kj}.*

We use $I(k,i)$ to denote the number of transmissions of flow F_i scheduled within P_{kj}'s lifetime. Given the first condition in Observation 1, since the absolute deadline of one packet of F_i aligns with the absolute deadline of P_{kj}, we upper bound the workload of packets in F_i that are within P_{kj}'s lifetime as:

$$I(k,i) = \lfloor D_k/T_i \rfloor C_i + \min(C_i, D_k \ mod \ T_i)$$

Within $I(k,i)$, there are two types of transmissions: (1) transmissions that bring conflict delay to P_{kj} and (2) transmissions that bring contention delay to P_{kj}. We name the first type of transmissions as *conflict transmissions*, and denote the total number of conflict transmissions as $I^{conf}(k,i)$. Meanwhile we name the second type of transmissions as *contention transmissions* and denote the total number of contention transmissions as $I^{cont}(k,i)$.

Note that one conflict transmission introduces much more delay than one contention transmission since it will directly delay a packet P_{kj} for one time slot regardless of the number

of available channels. A contention transmission will occupy one channel for its time slot. The packet P_{kj} can be delayed by contention transmissions only if all channels are occupied. Also the number of conflict transmissions equals to the conflict delay that P_{kj} suffers, since one conflict transmission will delay P_{kj} for exactly one time slot.

The necessary condition of a transmission $\vec{uv}$ of flow F_i to become conflict transmission is that it should at least share one node with one transmission $\vec{pq}$ of flow F_k, i.e. $u = p$ or $v = p$ or $v = q$ or $u = q$. However, this is not a sufficient condition, since only when $\vec{pq}$ is ready at time slot t and $\vec{uv}$ is scheduled at time slot t, $\vec{uv}$ becomes a conflict transmission. Otherwise, it is a contention transmission. We name all transmissions of F_i that share at least one node with one transmission of F_k as *potential transmissions*. Here we use the number of potential transmissions as the upper bound of number of conflict transmissions.

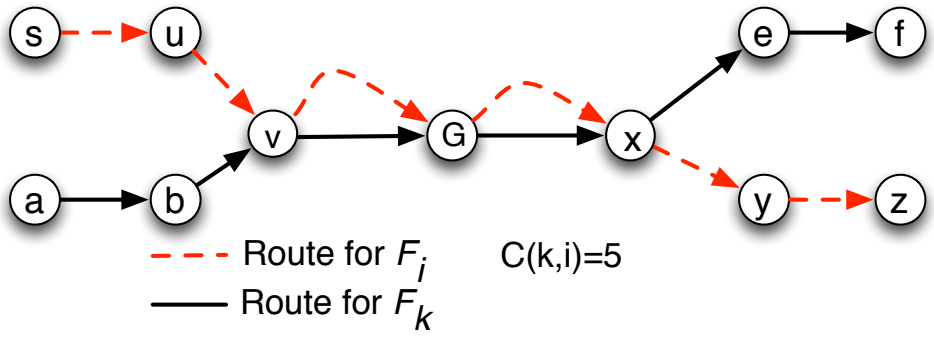


Figure 5.2: An example to show conflict delay

As shown in Figure 5.2, suppose F_k and F_i are two flows that share a part of their routes. Now we analyze the maximum number of potential transmissions within one packet of F_i that may introduce conflict delay to P_{kj}. Suppose absolute deadline of P_{ih} is earlier than P_{kj}, and P_{ih} has a higher priority than P_{kj}. A transmission of P_{kj} conflicts with a transmission of P_{ih} when they involve a common node. Whenever two transmissions conflict, the transmission that belongs to the lower priority packet must be delayed, no matter how many channels are available. The number of potential transmissions equals to the number of F_i's transmissions that share nodes with F_k's route. Let W_{ki} be the total number of F_i's transmissions that share nodes with F_k's route, then W_{ki} is the number of potential transmissions.

For example, in Fig. 5.2, the set of transmissions of flow F_i that share common nodes with F_k is $\{\vec{uv}, \vec{vG}, \vec{Gx}, \vec{xy}\}$. Here W_{ki} equals to 4. Then the maximum conflict delay a packet of F_k can suffer from one packet of flow F_i is no more than the number of potential transmissions W_{ki}, which equals 4 in this case.

Following the same reasoning of analyzing the maximum workload, the worst-case conflict delay P_{kj} suffers from packets of flow F_i is:

$$I^{conf}(k,i) = \lfloor D_k/T_i \rfloor W_{ki} + \min(W_{ki}, D_k \ mod \ T_i) \tag{5.1}$$

After bounding the maximum conflict delay that single packet P_{kj} suffers from flow F_i, we analyze the maximum contention delay that packets of flow F_i could introduce. As discussed before, the number of conflict transmissions of flow F_i equals to the conflict delay that P_{kj} suffers from flow F_i. Besides conflict transmissions, the remaining transmissions in $I(k,i)$ are grouped into contention transmissions:

$$I^{cont}(k,i) = I(k,i) - I^{conf}(k,i). \tag{5.2}$$

Theorem 1. *The maximum delay a single packet of flow F_k can suffer is:*

$$\theta_k = \lfloor \frac{1}{m} \sum_{i \neq k} I^{cont}(k,i) \rfloor + \sum_{i \neq k} I^{conf}(k,i)$$

Proof. The maximum conflict delay that a packet P_{kj} can suffer from flow F_i is $I^{conf}(k,i)$ as equation 5.1 showed. By adding up all flows except flow F_k (a constrained deadline model is considered here, so packet can not be delayed by packets from the same flow), we get the overall conflict delay as $\sum_{i \neq k} I^{conf}(k,i)$. Then we consider contention delay. By removing transmissions that bring conflict delay from $I(k,i)$, $I^{cont}(k,i)$ is the number of transmissions from flow F_i that will consume channel resource. By adding $I^{cont}(k,i)$ from all flows, then divided by number of channels m, we get the the worst-case contention delay $\lfloor \frac{1}{m} \sum_{i \neq k} I^{cont}(k,i) \rfloor$. This follows from the observation that EDF is work conserving, thus when a packet of F_k is ready but not executing, if it does not experience a conflict delay, each channel must be occupied by a packet of other flows. □

Recall *worst-case end-to-end delay* of flow F_i is the largest end-to-end delay among all its packets. Since end-to-end delay of a packet is the delay it suffers plus its execution time, we have following corollary:

Corollary 1. *The worst-case end-to-end delay of flow F_k is:*

$$R_k = \lfloor \frac{1}{m} \sum_{i \neq k} I^{cont}(k,i) \rfloor + \sum_{i \neq k} I^{conf}(k,i) + C_k \tag{5.3}$$

5.1 Improved Worst-case End-to-End Delay Analysis

We proposed a worst-case end-to-end delay analysis in previous section. Inspired by the multiprocessor scheduling techniques proposed in [8], we propose an improved worst-case end-to-end delay analysis based on a new observation. From now on, we will call the worst-case end-to-end delay analysis in Corollary 1 as the *basic delay analysis (BDA)*.

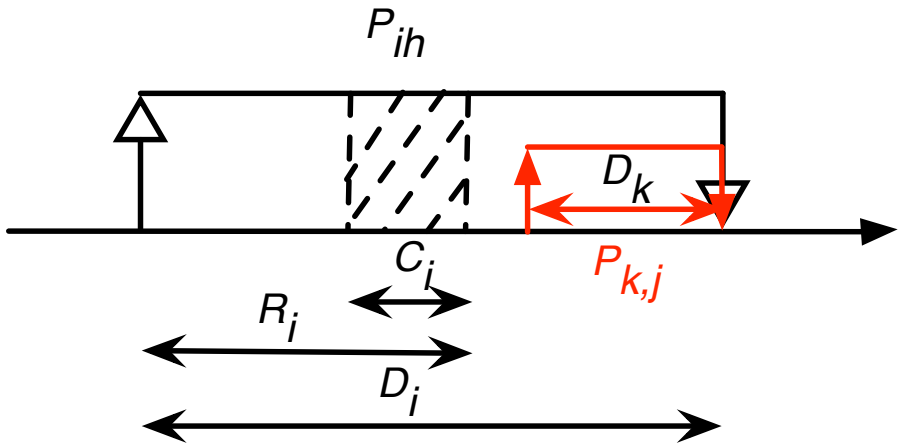


Figure 5.3: An example for Observation 2

We start from the Observation 1 for worst-case scenario. Remind the second point of the observation is that all transmissions of packets are scheduled at the very end of their scheduling windows. However, this observation can be broken in some cases. For example, if the end-to-end delay of a packet is far smaller than its absolute deadline, all transmissions should be scheduled far before its absolute deadline, which makes this point invalid. From a flow

perspective, if the worst-case end-to-end delay of a flow is far smaller than its relative deadline, then any packet of this flow will schedule all its transmissions far before its absolute deadline. We summarize it with the following observation.

Observation 2. *Transmissions of a packet P_{ih} cannot be scheduled later than the worst-case end-to-end delay R_i of flow F_i.*

As shown in Figure 5.3, transmissions of P_{ih} can only be scheduled before the worst-case end-to-end delay, which makes the assumption that all transmissions are scheduled at the very end of its scheduling window unreasonable. Actually, this observation will greatly improve the effectiveness of our end-to-end delay analysis. In BDA, the most difficult part is about flows with very short deadlines. A flow with short deadline is sensitive to delay especially conflict delay. When a flow F_k has a very short deadline D_k, it is critical to derive a precise delay analysis. Because even a small pessimistic delay analysis for each F_i will turn out to give a large amount of pessimism when we add delay from all flows together. Specifically, when D_k is small, with previous analysis, each flow F_i will introduce delay on F_k. However, with improved analysis, F_i will not even introduce any delay on F_k if the gap between F_i's worst-case end-to-end delay R_i and deadline D_i is larger than D_k. For example, in Figure 5.3, we discuss the delay that P_{kj} suffers from flow F_i. Deadline of packet P_{ih} is aligned with deadline of packet P_{kj}. Worst-case end-to-end delay of F_i is shown in the figure with R_i. From the figure, end-to-end delay of P_{ih} is before absolute deadline of P_{ih} as well as release time of P_{kj}. Then all transmissions of $P_{i,l}$ are scheduled before release of P_{kj}. This makes the delay that P_{kj} suffers from P_{ih} zero. However, based on BDA, conflict delay that P_{kj} suffers is W_{ki}, besides the channel contention delay it suffers.

By incorporating this observation, we propose an *improved delay analysis (IDA)*. In the IDA, we use * to denote the new result of any terminology we already analyzed in BDA. The general worst-case scenario is shown in Figure 5.4. We discuss the worst-case delay packet P_{kj} suffered from flow F_i. The lifetime of P_{kj} is $[t_r, t_d)$. Note that we also show the worst-case end-to-end delay R_i in the figure. From the figure, we can see the leftmost packet of F_i has at most $\min(C_i, (D_k \ mod \ T_i) - (D_i - R_i))$ transmissions scheduled within time window $[t_r, t_d)$. This is different from our previous analysis $\min(C_i, D_k \ mod \ T_i)$, where worst case end-to-end delay of F_i is not considered. Based on this new observation, a new upper

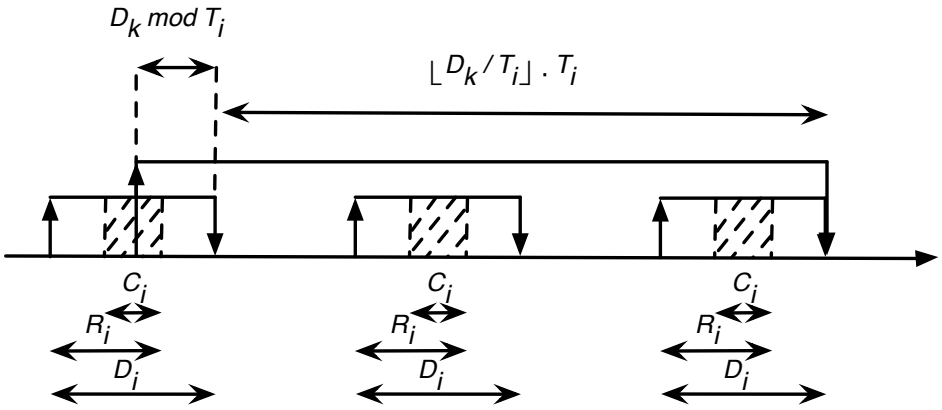


Figure 5.4: Worst-case scenario under Observation 2

bound of workload that F_i finishes within the time window $[t_r, t_d)$ is

$$I(k,i)^* = \lfloor D_k/T_i \rfloor C_i + \min(C_i, (D_k \ mod \ T_i) - (D_i - R_i)) \tag{5.4}$$

It becomes more interesting when we derive an upper bound on the transmission conflict delay a packet of F_k suffers from F_i.

Theorem 2. *An upper bound on the conflict delay that any packet of F_k experiences from F_i is:*

$$I^{conf}(k,i)^* = \begin{cases} 0, & \text{if } D_k \le D_i - R_i \\ W_{ki}(D_k - (D_i - R_i)), & \text{if } D_i - R_i < D_k \le D_i \\ \lfloor D_k/T_i \rfloor W_{ki} + W_{ki}(\max\{0, D_k \ mod \ T_i - (D_i - R_i)\}), & \text{if } D_k > D_i \end{cases} \tag{5.5}$$

Proof. We discuss the transmission conflict delay in three cases.

- $D_k \le D_i - R_i$
- $D_i - R_i < D_k \le D_i$
- $D_k > D_i$

Suppose P_{kj} is the packet we are discussing for F_k. In the first case, deadline of F_k is smaller than gap between F_i's worst-case end-to-end delay and its deadline. All transmissions of F_i are scheduled before P_{kj}'s release time. Then there is no transmission conflict delay in this case.

In the second case, release time of P_{kj} is before F_i's worst-case end-to-end delay. Part of F_i's transmissions are scheduled after P_{kj}'s release time. The first $D_k - (D_i - R_i)$ transmissions of P_{kj} could potentially conflict with F_i's last $D_k - (D_i - R_i)$ transmissions. We introduce a new terminology $W_{ki}(\nu)$, which denote the maximum conflict delay that last ν transmissions of F_i could introduce to the first ν transmissions of F_k. Intuitively, we can treat the last ν transmissions of F_i as a new flow F_i' and treat the first ν transmissions of F_k as a new flow F_k'. We calculate $W_{ki}(\nu)$ by counting the number of transmissions of F_i' that share nodes with F_k'. Note that if ν is larger than both C_i and C_k, then $W_{ki}(\nu) = W_{ki}$. Then the maximum possible transmission conflict delay is $W_{ki}(D_k - (D_i - R_i))$.

In the third condition, there is potentially more than one packet of F_i that brings transmission conflict delay into P_{kj}. The number of packets of F_i that fully coincide with P_{kj}'s lifetime is $\lfloor D_k/T_i \rfloor$, and their worst-case conflict delay to P_{kj} is $\lfloor D_k/T_i \rfloor W_{ki}$. The conflict delay that the leftmost packet of F_i in Figure 5.4 brings to P_{kj} depends on the length of $D_k \ mod \ T_i$. If $D_k \ mod \ T_i \leq Di - R_i$, then the leftmost packet will not bring conflict delay. If $D_k \ mod \ T_i > Di - R_i$, the leftmost packet will bring at most $W_{ki}(\min\{0, D_k \ mod \ T_i - (D_i - R_i)\})$ to P_{kj}.

□

Given the new analysis on workload and transmission conflict delay of flow F_i on flow F_k, the number of channel transmissions is:

$$I^{cont}(k,i)^* = I(k,i)^* - I^{conf}(k,i)^*. \tag{5.6}$$

Similar to Theorem 1, we give an upper bound on the maximum delay that any packet of flow F_k can suffer and the worst-case end-to-end delay of F_k here.

Corollary 2. *The maximum delay a single packet of flow F_k can suffer is:*

$$\theta_k^* = \lfloor \frac{1}{m} \sum_{i \neq k} I^{cont}(k,i)^* \rfloor + \sum_{i \neq k} I^{conf}(k,i)^* \tag{5.7}$$

Corollary 3. *The worst-case end-to-end delay of flow F_k is:*

$$R_k^* = \lfloor \frac{1}{m} \sum_{i \neq k} I^{cont}(k,i)^* \rfloor + \sum_{i \neq k} I^{conf}(k,i)^* + C_k \tag{5.8}$$

The flow set $\{F_1, F_2, \cdots, F_n\}$ is schedulable if following is true for each flow F_k:

$$R_k^* \leq D_k \tag{5.9}$$

We use an iterative algorithm to derive the worst-case end-to-end delay R_k^*. In the initialization, R_k is set to D_k for all flows. At the beginning of each iteration, for each flow F_k, R_k is set to R_k^* from last iteration. And R_k^* is calculated based on equation (5.4)-(5.8). The algorithm will enter a new iteration if the flow set is unschedulable and at least one flow has a worst-case end-to-end delay R_k^* updated, otherwise it terminates. We show the pseudo-code in Algorithm 1.

Algorithm 1: Iterative algorithm

```
Set R_k^* = D_k, ∀k ≤ N;
repeat
    Set R_k = R_k^*, ∀k ≤ N;
    for k ≤ N do
        Calculate R_k^* based on (5.4)-(5.8);
    end
until ∀k ≤ N, R_k^* ≤ D_k or ∀k ≤ N, R_k^* = R_k;
```

5.2 Complexity Analysis

Now we want to estimate the complexity of our delay analysis. The BDA in Corollary 1 is polynomial. The calculation of worst-case delay of flow F_k is in $O(n)$ since we have n flows.

The complexity of BDA is $O(n^2)$ since we need to calculate worst-case delay for every flow. The total time complexity is $O(n^2)$.

The improved delay analysis (IDA) in Corollary 3 is pseudo-polynomial. The analysis in each iteration is $O(n^2)$ as discussed above. Since there are n flows, and each one's worst-case end-to-end delay can range from C_k to D_k, the number of iterations is bounded as $O(n \max(D_k - C_k))$. Thus, the overall complexity is $O(n^3 \max(D_k - C_k))$.

Chapter 6

Evaluation

We evaluate our end-to-end delay analysis through simulations based on both the real topologies of a wireless sensor network testbed and random topologies. We evaluate our improved delay analysis (IDA) in Corollary 3 by comparing it with the basic delay analysis (BDA) in Corollary 1 and the simulation (SIM).

Our delay analysis is evaluated in terms of *pessimism ratio* and *acceptance ratio*. The former one is used to assess the tightness of the delay analysis, and the latter one is used to evaluate the effectiveness of our analysis for online admission control. For each flow, the *pessimism ratio* is defined as the ratio of its theoretical worst-case end-to-end delay given by our analysis to its maximum end-to-end delay observed in simulation. The *acceptance ratio* is defined as the ratio of the number of test cases deemed schedulable by our analysis (or simulation) to the total number of test cases. A test case is schedulable in simulation if all flow instances released in the hyper-period meet their deadlines. The simulator is written in C++ and all tests are performed on a MacBook Pro laptop with 2.4 GHz Intel Core 2 Duo processor.

6.1 Evaluation on Testbed Topologies

In this part of evaluation, we use the topologies of a real sensor network testbed [1] deployed on the fifth floor in Bryan Hall and Jolley Hall of Washington University in St. Louis, as shown in Figure 6.1. The testbed consists of 69 TelosB motes. The TelosB mote's radio is compliant with the IEEE 802.15.4 standard. For each link in the testbed, we measured its *packet reception ratio (PRR)* by counting the number of received packets among 250 packets

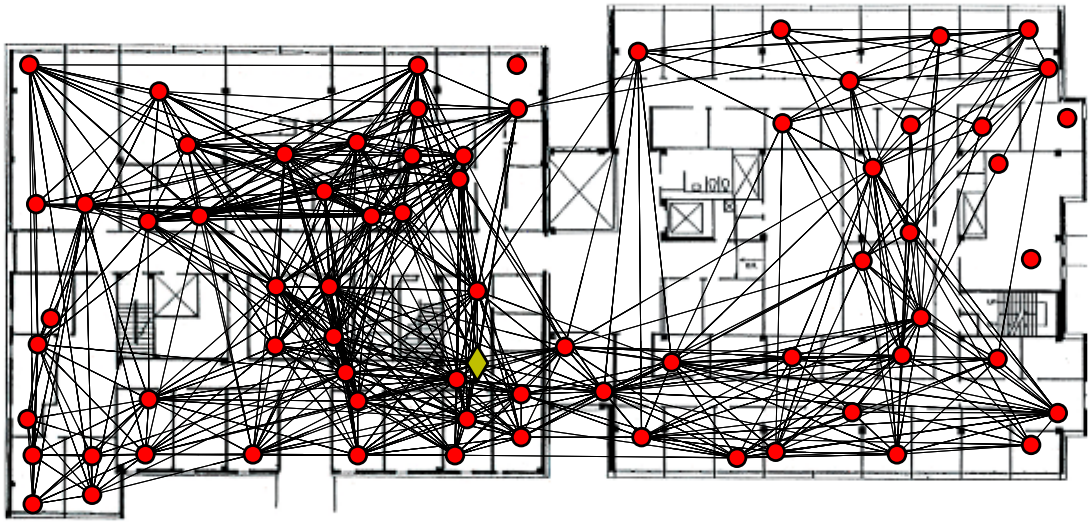
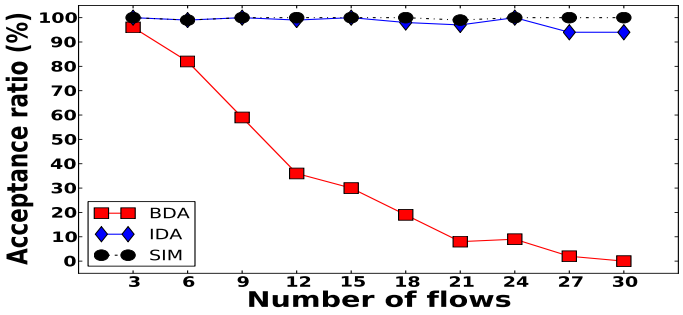


Figure 6.1: Testbed topology (at transmission power of 0 dBm)

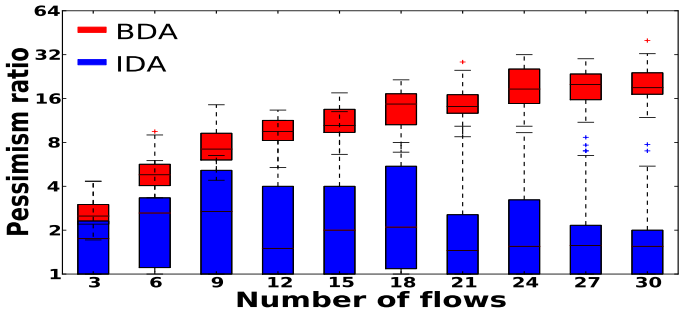
transmitted on the link. We only add links with PRR higher than 80% to the topology of the testbed. Topologies at 3 different transmission power levels (0 dBm, -1 dBm, -3 dBm) are collected for our simulations. The node with the highest number of neighbors is designated as the gateway (yellow diamond in Figure 6.1). A fraction of the remaining nodes are used as sources and destinations. The sets of sources and destinations are disjoint.

Different numbers of flows are generated by increasing the number of source and destination pairs. The period T_i of the each flow F_i is randomly generated in the range of $2^{6\sim11}$ time slots. The relative deadline D_i of every flow F_i is randomly generated in the range of $(C_i, \beta * P_i)$ slots, here β is a randomly generated number in range of $(0, 1)$. C_i is the required time slots needed to deliver a packet from the source to the destination. For each flow set, we generate 100 test cases and simulate them on topologies at different transmission power levels.

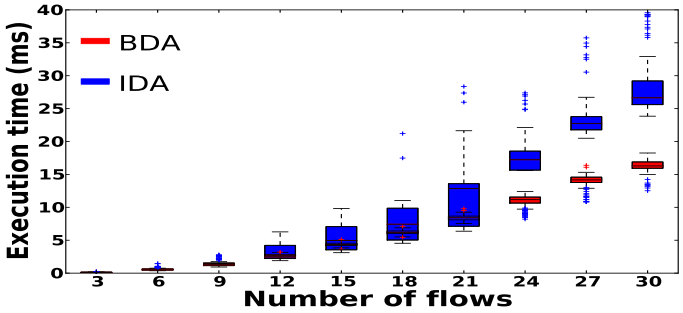
The acceptance ratios of IDA, BDA and simulation (SIM) using topology with transmission power of 0 dBm are shown in Figure 6.2(a). The acceptance ratio of IDA remain close to simulations. The gap between IDA and SIM widens as the number of flows increases, but remains within 10%. This result indicates the effectiveness of IDA for admission control. The acceptance ratio of IDA is much higher than BDA, which shows the IDA highly outperforms BDA in terms of acceptance ratio.



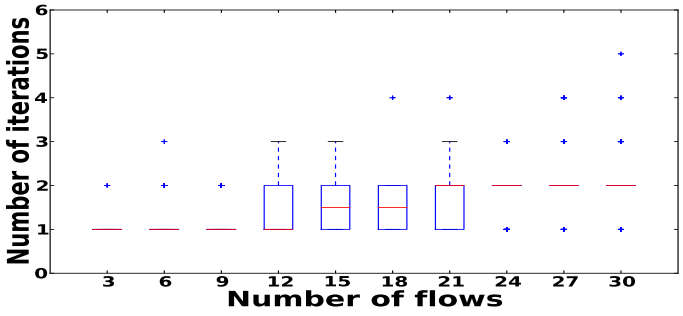
(a) Acceptance ratio



(b) Boxplot of pessimism ratio



(c) Boxplot of execution time



(d) Boxplot of number of iterations

Figure 6.2: Schedulability analysis on testbed topology

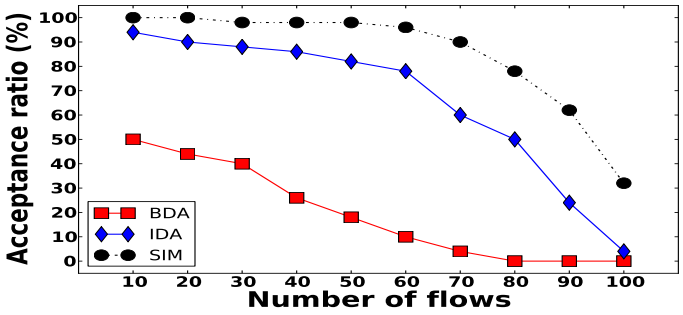
Figure 6.2(b) shows the boxplots of the pessimism ratios of IDA and BDA. Note that the y-axis is plotted in a log scale. Since if a test case is not schedulable under simulation, the simulator could not lay out the schedule of all flows, then we could not get the actual maximum end-to-end delay. So all pessimism ratios here are from test cases that are schedulable under simulation. This result confirms IDA greatly improves the tightness of the delay bound compared to BDA. The pessimism ratio of BDA increases as the number of flows increases. However, the pessimism ratio of IDA remain low despite the increase of number of flows. This figure shows our IDA is scalable to large number of flows.

The time complexity of our algorithms are shown in Figure 6.2(c). The execution time of IDA grows faster than BDA as the number of flows grows. Figure 6.2(c) and Figure 6.2(d) shows the boxplot of number of iterations in IDA. The median value is at most 2 and the 75% percentile is at most 3. The figure shows that when number of flows is small, our algorithm IDA converges fast. Figures 6.2(a)-6.2(c) show the tradeoff between accuracy and time complexity. While IDA runs slower than BDA, it gives a much more precise estimation of end-to-end delay, which leads to a higher acceptance ratio.

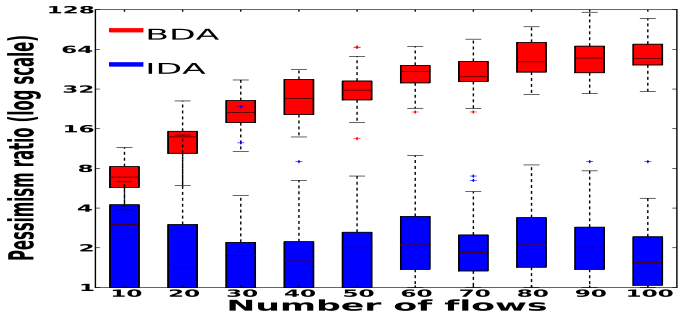
6.2 Evaluation on Random Topologies

Besides real testbed topologies, we also test our analysis on larger random topologies with larger number of flows. We generate random networks with 400 nodes and 800 links. Links are chosen randomly and assigned PRR randomly in the range of $[0.80, 1.0]$. We test our delay analysis on different number of flows. The rest of the simulation setup is same as our simulations on testbed topologies.

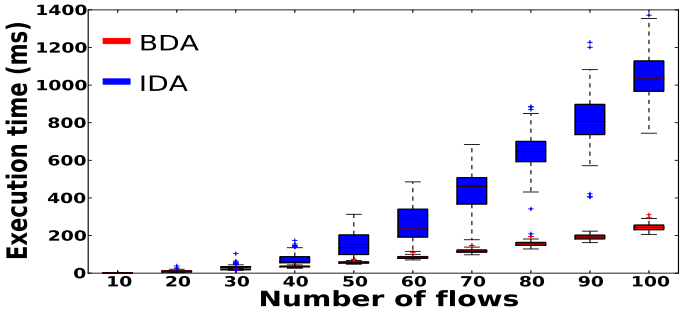
The results under large random topologies have similar trends as those under the testbed topologies. IDA is effective for admission control with the acceptance ratio within 30% to simulations even when the network is heavily loaded. IDA significantly outperforms BDA in admission control thanks to its tighter delay bounds as shown in Figure 6.3(b). Even when the number of flows increases 100, the pessimism ratio of IDA still remains in the same range as 10 flows, which shows IDA's scalability in terms of pessimism ratio. Figure 6.3(d) shows the median number of iterations in IDA increase from 2 to 5 on random topologies. As shown in Figure 6.3(c), although the number of flows increases to 100, the maximum



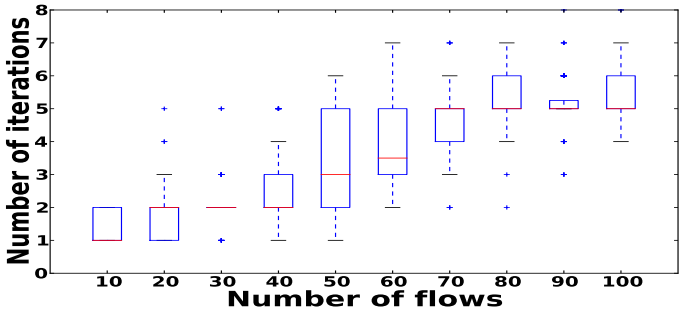
(a) Acceptance ratio



(b) Boxplot of pessimism ratio



(c) Boxplot of execution time



(d) Boxplot of number of iterations

Figure 6.3: Schedulability analysis on random topology

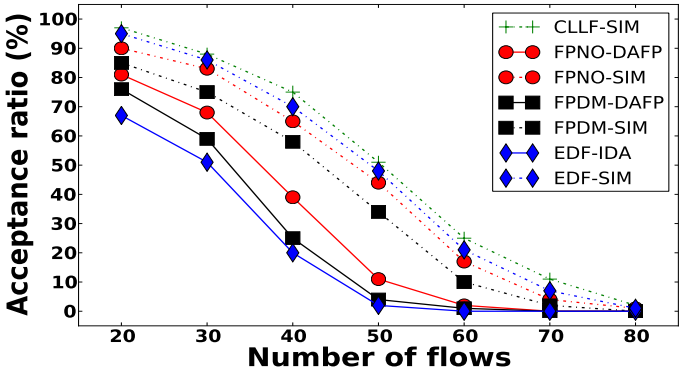
execution time is less than 1.5 seconds in worst case. This figure shows IDA is scale to large number of flows in terms of computational cost.

6.3 Comparative Study of Scheduling Policies

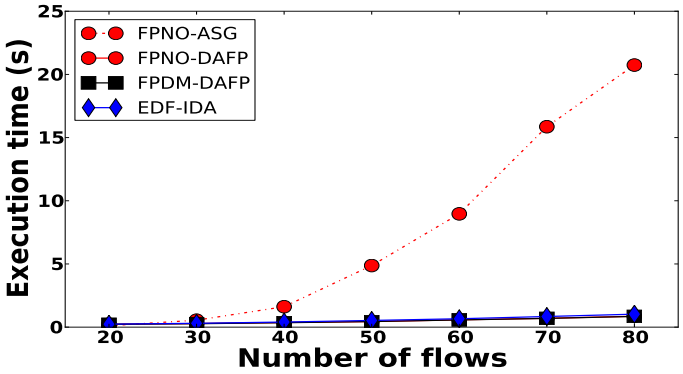
In this section, we compare EDF with some representative scheduling policies: (a) Fixed Priority scheduling with Deadline Monotonic priority assignment (FPDM), which assigns priorities to flows using the relative deadline and schedules transmissions based on flows' priorities; (b) Fixed Priority scheduling with Near Optimal priority assignment (FPNO), which assigns priorities of flows using a heuristic search algorithm [16]; (c) Conflict-aware Least Laxity First (CLLF) [17], which incorporates conflict delay into traditional Least Laxity First scheduling policy. We also compare our IDA delay analysis against Delay Analysis for Fixed Priority scheduling policies (DAFP) [15]. All tests in this subsection are based on random topologies. We increase the number of flows in the network from 20 to 80 and show simulation results in Figure 6.4.

We compare scheduling policies through simulation in Figure 6.4(a). Dash lines show the percentage of task sets that different scheduling policies can schedule in simulation. Solid lines show acceptance ratios of different schedulability analysis techniques. Results show EDF can schedule more task sets than FPNO and FPDM, which indicates that EDF is indeed an effective scheduling policy. Although CLLF can schedule more task sets than EDF, there is no schedulability analysis for CLLF. We also compare IDA with DAFP in Figure 6.4(a). Given the complexity that EDF brings to schedulability analysis, the acceptance ratio of IDA is slightly lower than DAFP. Overall IDA is shown to be competitive to the state-of-the-art delay analysis technique DAFP.

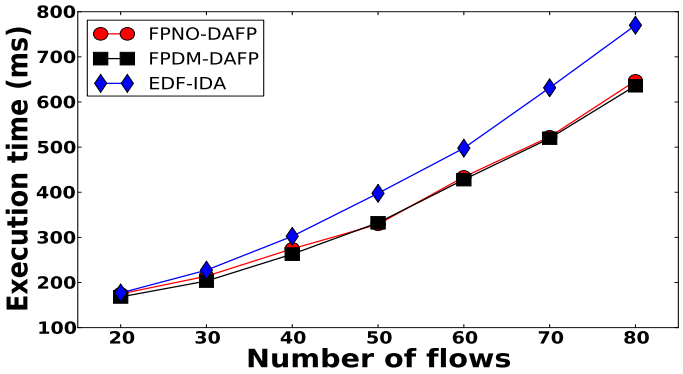
Figure 6.4(b) shows execution time of two schedulability analysis techniques as well as priority assignment algorithm used in FPNO. The execution time of priority assignment algorithm used in FPNO (FPNO-ASG) is much higher than execution time of DAFP and IDA. We further compare execution time of DAFP and IDA in Figure 6.4(c). Given EDF is a more complicated scheduling policy compared to fixed priority scheduling policies, the execution time of IDA is slightly higher than DAFP. It indicates that IDA is an efficient schedulability analysis approach.



(a) Acceptance ratio



(b) Execution time



(c) Execution time

Figure 6.4: Comparison of different scheduling policies

Chapter 7

Conclusions

With the emergence of industrial standards such as WirelessHART, industrial process control is moving towards wireless control networks as the network technology for real-time communication between sensors, controllers and actuators. To meet the stringent real-time performance requirements of control systems, there is a critical need for fast end-to-end delay analysis to support online admission control of periodic real-time flows in WCNs. This thesis presents the first end-to-end delay analysis for WCNs under Earliest Deadline First (EDF) transmission scheduling, a widely used dynamic priority scheduling policy in real-time systems. Our analysis considers delays caused by both channel contentions and conflicts between concurrent transmissions and provides safe upper bounds for the end-to-end delays of real-time flows. Specifically, this thesis (1) leverages real-time multiprocessor scheduling analysis to derive contention delays, (2) integrates both conflict and contention delays in a holistic end-to-end delay analysis, and (3) reduces the pessimisms in admission control through tighter delay bounds on flows with tight deadlines. Simulations based on both random topologies and the topology of a wireless testbed demonstrate the effectiveness of our analysis for online admission control of real-time flows.

References

[1] `http://mobilab.wustl.edu/testbed`.

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

[3] Tarek F. Abdelzaher, Shashi Prabh, and Raghu Kiran. On real-time capacity limits of multihop wireless sensor networks. In *RTSS '04*.

[4] T.P. Baker. Multiprocessor edf and deadline monotonic schedulability analysis. In *RTSS'03*.

[5] T.P. Baker. An analysis of edf schedulability on a multiprocessor. *IEEE Transactions on Parallel and Distributed Systems*, 16(8):760 – 768, aug. 2005.

[6] Sanjoy Baruah. Techniques for multiprocessor global schedulability analysis. In *RTSS'07*.

[7] Marko Bertogna, Michele Cirinei, and Giuseppe Lipari. Improved schedulability analysis of edf on multiprocessor platforms. In *ECRTS'05*.

[8] Marko Bertogna, Michele Cirinei, and Giuseppe Lipari. Schedulability analysis of global scheduling algorithms on multiprocessor platforms. *IEEE TPDS*, 20(4):553 – 566, April 2009.

[9] Octav Chipara, Chenyang Lu, and Gruia-Catalin Roman. Real-time query scheduling for wireless sensor networks. In *RTSS '07*.

[10] Octav Chipara, Chengjie Wu, Chenyang Lu, and William Griswold. Interference-aware real-time flow scheduling for wireless sensor networks. In *The 23rd Euromicro Conference on Real-Time Systems (ECRTS 2011)*.

[11] Joël Goossens, Shelby Funk, and Sanjoy Baruah. Priority-driven scheduling of periodic task systems on multiprocessors. *Real Time Systems*, 25(2-3):187 – 205, 2003.

[12] Isa 100. http://www.isa.org/isa100.

[13] Petr Jurcík, Ricardo Severino, Anis Koubâa, Mário Alves, and Eduardo Tovar. Real-time communications over cluster-tree sensor networks with mobile sink behaviour. In *RTCSA '08*.

[14] N. Pereira, B. Andersson, E. Tovar, and A. Rowe. Static-priority scheduling over wireless networks with multiple broadcast domains. In *RTSS '07*.

[15] Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen. End-to-end delay analysis for fixed priority scheduling in WirelessHART networks. In *RTAS'11*.

[16] Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen. Priority assignment for real-time flows in WirelessHART networks. In *ECRTS'11*.

[17] Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen. Real-time scheduling for WirelessHART networks. In *RTSS'10*.

[18] Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen. End-to-end communication delay analysis in wirelesshart networks. Tech. Rep. WUCSE-2011-86, Washington University in St Louis, 2011.

[19] Jens B. Schmitt and Utz Roedig. Sensor network calculus - A framework for worst case analysis. In *DCOSS '05*.

[20] Pablo Soldati, Haibo Zhang, and Mikael Johansson. Deadline-constrained transmission scheduling and data evacuation in WirelessHART networks. In *ECC '09*.

[21] J.A. Stankovic, T.E. Abdelzaher, Chenyang Lu, Lui Sha, and J.C. Hou. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91(7):1002–1022, July 2003.

[22] John A. Stankovic, Krithi Ramamritham, and Marco Spuri. *Deadline Scheduling for Real-Time Systems: EDF and Related Algorithms*. Kluwer Academic Publishers, 1998.

[23] Haibo Zhang, Fredrik Osterlind, Pablo Soldati, Thiemo Voigt, and Mikael Johansson. Rapid convergecast on commodity hardware: Performance limits and optimal policies. In *SECON '10*.

[24] Haibo Zhang, Pablo Soldati, and Mikael Johansson. Optimal link scheduling and channel assignment for convergecast in linear WirelessHART networks. In *WiOpt'09*.

Vita

Chengjie Wu

Degrees

B.S. Math and Physics, May 2006
M.S. Control Science and Engineering, May 2008

Professional Societies

Institute of Electrical and Electronics Engineers

Publications

Chengjie Wu, You Xu, Yixin Chen and Chenyang Lu (2012). Submodular Game for Distributed Application Allocation in Shared Sensor Networks, *TIEEE International Conference on Computer Communications (INFOCOM'12)* March 2012.

Abusayeed Saifullah, Chengjie Wu, Paras Tiwari, You Xu, Yong Fu, Chenyang Lu and Yixin Chen (2012). Near Optimal Rate Selection for Wireless Control Systems, *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'12)*, April 2012.

Octav Chipara, Chengjie Wu, Chenyang Lu and William Griswold (2011). Interference-Aware Real-Time Flow Scheduling for Wireless Sensor Networks, *Euromicro Conference on Real-Time Systems (ECRTS'11)*, July 2011.

August 2013

Delay Analysis for WCNs under EDF Scheduling, Wu, M.S. 2013