Fault-tolerant and Real-time Wireless Sensor Network for Control System

by

Wenchen Wang

Bachelor of Engineering, Northeastern University, China 2013

M.S. in Computer Science, University of Pittsburgh, 2017

Submitted to the Graduate Faculty of

the Kenneth P. Dietrich School

of Arts and Sciences in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2019

UNIVERSITY OF PITTSBURGH

DIETRICH SCHOOL OF ARTS AND SCIENCES

This dissertation was presented

by

Wenchen Wang

It was defended on

July 27, 2018

and approved by

Dr. Daniel Mosse, Department of Computer Science, University of Pittsburgh

Dr. Rami Melhem, Department of Computer Science, University of Pittsburgh

Dr. Youtao Zhang, Department of Computer Science, University of Pittsburgh

Dr. Daniel Cole, Department of Mechanical Engineering and Materials Science, University of Pittsburgh

Dissertation Director: Dr. Daniel Mosse, Department of Computer Science, University of Pittsburgh

Fault-tolerant and Real-time Wireless Sensor Network for Control System

Wenchen Wang, PhD

University of Pittsburgh, 2019

Wireless control systems (WCSs) enable several advantages over traditional wired industrial monitoring and control systems, including self-organization, flexibility, rapid deployment, and lower maintenance. However, wireless network delay and packet loss can result in two main challenges for the control system: instability and performance degradation. This dissertation aims at solving the instability and performance degradation challenges by developing fault-tolerance and real-time approaches for a WCS.

For the instability challenge, we first developed a fault-tolerant network design and a novel model to meet the control system stability requirement for one-way wireless transmission. The evaluation results showed that our model was accurate with average 4.1% difference from the simulation result. We scaled the work to two-way wireless transmission to meet the control system stability requirement by analyzing the worst-case end-to-end delay. We carried out an analysis to calculate the maximum number of conflicts that could happen during one message transmission, and then derived the worst-case end-to-end delay. The simulation results showed that our end-to-end delay analysis was accurate within 4.2% of realistic simulation results.

For the performance degradation challenge, we explored a hybrid offline-online network reconfiguration framework with time-varying link failures to improve control system performance for the WCS with a single physical system. Accordingly, a precise network imperfection model and six reconfiguration algorithms had been developed to quantify and improve the performance, respectively. The case study results showed that our network imperfection model was accurate with Pearson correlation 0.993 and our network reconfiguration approach performed better than the state-of-the-art static scheme. To improve the overall control system performance for the WCS with multiple physical systems, we studied a dynamic packet assignment approach. The case study results demonstrated that our approach was effective in improving the overall control system performance.

Table of Contents

List of Tables

List of Figures

List of Equations

List of Algorithms

Preface

In loving memory of my grandparents, two most important person in my life, Mr. Keming Wang and Mrs. Renjuan Yu.

1.0 Introduction

1.1 Background

A control system is a system, which provides the desired input by controlling the output [con, 2018] as shown in Figure 1. The desired input is called *reference function*. For example, cruise control in a car controls the speed with the reference function of a constant function of a certain speed setting by users. A *stable control system* produces a bounded output for a given bounded input. Control systems can be classified as *open loop control systems* and *closed loop control systems*. In open loop control systems, output is not fed-back to the input. Whereas, in closed loop control systems, output is fed back to the input. In this dissertation, we focus on the closed loop control systems.

Traditional control systems rely on wires to connect controller, sensors, and actuators, where the controller is the central point that can control one or more physical systems (we call them wired control systems in this dissertation). Typically, the controller is physically separated from the physical plant, in a remote location. Figure 2 shows a general wired control system with a single physical system. The sensors attached to the physical system send measurement packets to the remote controller periodically and the remote controller calculates the control signal and sends back the control signal to the actuator to actuate the physical system using the received control signal. However, when a wired control system with a single physical system is scaled to control a large number of physical systems, it will bring the deployment and maintenance problems. Motivated by these problems, wireless control

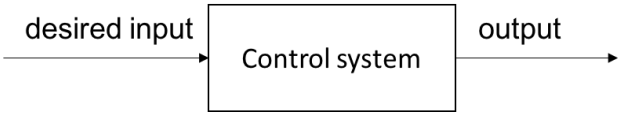


Figure 1: Control system

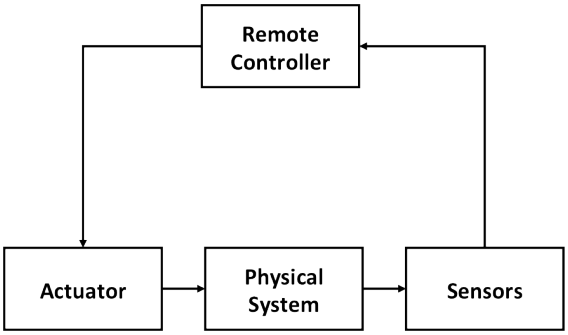


Figure 2: Wired control system

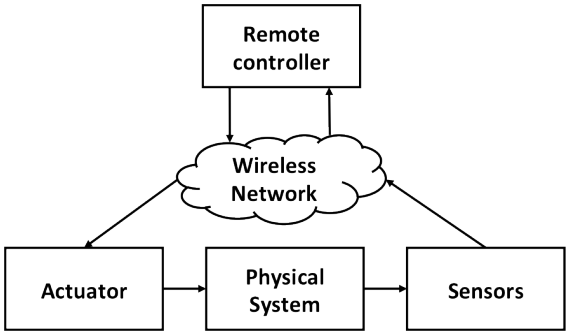


Figure 3: Wireless control system

systems (WCSs) are gaining rapid adoption in the industrial process, because WCSs can overcome the problems of wired control systems and have the advantages of self-organization and flexibility [Gungor and Hancke, 2009]. WCSs have been widely applied to domains of transportation, health-care, manufacturing, agriculture, energy, aerospace and building. WCSs controlled over multi-hop wireless sensor networks (WSNs) have especially received significant attention in recent years [Han et al., 2011; Li et al., 2015, 2016; Pajic et al., 2011b,a; Wang et al., 2016; Kim and Kumar, 2010]. As shown in Figure 3, the wireless transmission has two directions: (1) *up*, sensors sending measurement messages to the remote controller; (2) *down*, the remote controller transmitting the messages with control signals back to the actuators. In this dissertation, we say a message is sent "up" to the remote controller and "down" to the actuator. We focus on two wireless communication scenarios in a WCS: (1) one-way wireless transmission: transmitting messages up to the remote controller by assuming the messages are sent down on another wireless channel with a different radio frequency; (2) two-way wireless transmission: transmitting messages up to the controller and down to the actuator sharing a wireless network with the same radio frequency.

1.2 Problem Description

While early success of WSNs has been recognized, significant potential remains in exploring WSNs as fault-tolerance and real-time networks for industrial plants. Even though

WSNs are good for deployment, wireless network communications are imperfect in terms of packet loss and network delay. Most WSNs embedded in WCSs are deployed and applied in industrial environments, such as smart grids [Gungor et al., 2010], water tanks [Li et al., 2015] and even nuclear power plants (NPPs) [Wang et al., 2016]. Harsh and complex electric-power-system environments pose great challenges in the reliability of WSN communications. Interference is the main factor of packet losses in WSN [Li, 2015], where wireless links exhibit widely varying characteristics over time due to moving people/obstacles and electromagnetic and radio frequency interference (EMI/RFI) [Baccour et al., 2012; Kar and Moura, 2009; Baccour et al., 2012; Gungor et al., 2010; Ganesan et al., 2001]. The interference can make some links/nodes inaccessible and disconnected for a limited amount of time (e.g., if an obstacle, like a factory robot transporting materials, blocks the wireless transmission). Moreover, the time delay is another issue of WSN due to retransmissions and multi-hop characteristic. Real-time scheduling has been studied in WSN to constraint network delays in [Saifullah et al., 2010; Gobriel et al., 2009b; Stankovic et al., 2003].

A WCS is a system with two subsystems, the wireless sensor network and the control system. The performance of one subsystem will affect the other. Network-induced imperfections [Zhang et al., 2013; Gupta and Chow, 2010], that is, packet loss and time delay (discussed above) can result in two main problems for the control system: instability [Zhang et al., 2001; Zhang and Yu, 2008; Jusuf and Joelianto] and performance degradation [Pant et al., 2015; Li et al., 2016]. When the control system is unstable, the plant (i.e., the physical system) or part thereof can be damaged and lead to serious safety issues and financial loss. On the other hand, even if the control system is stable under network-induced imperfections, WSN can introduce unreliable/non-deterministic levels of service in terms of delays and losses and induce undesirable additional errors, that is, *network-induced error*. The smaller the network-induced error, the closer to the wired control system performance.

The control system application desired requirements can be categorized as hard and soft requirements, which are stability and performance, respectively. The performance requirement comes after the stability requirements are met. This dissertation studies different fault tolerance and real-time techniques in WSN to solve the following two problems:

- *P1*: control system stability guarantee;

- *P2*: network-induced error reduction.

In a WCS, the *control sampling period* is the interval where the control loop makes decisions. In practice, the control sampling period in a WCS and cyber-physical system (CPS) is 2^n seconds, where $-2 \leq n \leq 9$, that is, from 250 ms to approximately 8 minutes [Han et al., 2011] and it depends on the plant being controlled. After one knows the control sampling period, there are two cases for the network delay: (1) worst-case network delay is less than or equal to the control sampling period; (2) worst-case network delay is more than the control sampling period. For the first case, the network reliability is the key effect on control system performance. The higher the reliability, the better the control system performance. To achieve high reliability, the network can be designed to be "as reliable as possible," that is, a high level of redundancy, which requires more backup nodes. A higher number of backup nodes typically induces more delay for messages to be delivered (more traffic on the network), but all messages still arrive within the control sampling period and, thus the delay has little (if any) effect on the control system performance. Recent research works mainly focus on this case [Saifullah et al., 2011; Li et al., 2015; Saifullah et al., 2015; Li et al., 2016]. However, for the second case, there is a trade-off between network delay and packet losses for the control system stability and performance, which is a more complex case that there is limited research insight on.

Our dissertation focuses on addressing *P1* and *P2* covering the two cases above, that is, it is possible that the worst-case network delay is more than the control sampling period, which is more general and complicated than previous research.

1.3 Research Overview

In this dissertation, there is a delicate interplay between network reliability and network delay for designing a WSN in the WCS. Redundancy requires an additional delay to achieve network reliability. Conversely, it is easy to see that reducing the redundancy (e.g., backup rely nodes) in a WSN to reduce end-to-end network delay increases the probability of packet loss. The trade-off between fault tolerance and real-time network for the industrial wireless

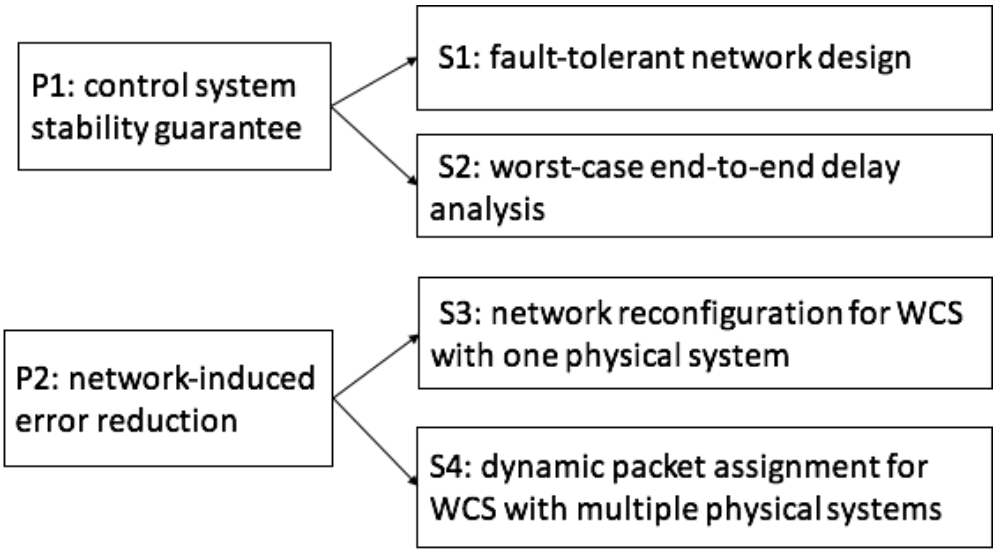


Figure 4: The relationship between the problems and solutions of this dissertation

network has been explored in the literature [Han et al., 2011; Yu et al., 2014]. Limited insights have emerged with respect to how the wireless control system application's desired requirements affect and are affected by the fault tolerance and real-time communication.

To this end, this dissertation aims at developing fault-tolerance and real-time approaches to address *P1* and *P2*. Our goal is to study the viability of and provide the justification for the following **dissertation statement**:

"It is possible to achieve stability and reduce network-induced error for control systems, while operating under packet losses and real-time constraints in a wireless network."

We seek to achieve this objective by studying the two problems (P1 and P2).

For P1, the control system stability requirement in this dissertation is given by the control engineering researchers (see Equation (4.1)), that is, an inequality constraint of network delay and packet loss. Given the control system stability requirement, we propose a fault-tolerant network node placement design and a model to estimate the minimum number of active nodes in the network, by network delay and packet loss analysis, to meet the stability requirement for the one-way transmission (solution 1: *S1*). To meet the stability requirement for two-way

transmission, the network delay analysis is more complicated than the one-way transmission (the packet loss analysis in *S1* can be applied to two-way transmission). So we proposed a worst-case end-to-end delay analysis (solution 2: *S2*).

For P2, under the assumption that the control system stability requirement is met, we studied reducing the network-induced error in the WCS with a single physical system and with multiple physical systems. Specifically, we explored a network reconfiguration framework with offline and online parts to reduce the network-induced error for a WCS with a single physical system (solution 3: *S3*). We then studied a dynamic packet assignment approach to reduce the network-induced error for a WCS with multiple physical systems (solution 4: *S4*).

Figure 4 shows the relationship between the problems and solutions.

1.4 Contributions

This dissertation consists of the following main contributions.

Fault-tolerant network design. Control system stability is critical for physical plants, since system instability can result in plant damage and severe safety issues [Zhang et al., 2001; Zhang and Yu, 2008; Jusuf and Joelianto]. In WCSs, network delay and packet loss are the potential threats to control system stability. Given a control system stability requirement in terms of network delay and packet loss, we first propose a flexible fault-tolerant node placement design. We then develop a model to meet the requirement for one-way wireless transmission and to determine the initial network topology with the minimum number of active nodes [Wang et al., 2016]. This contribution is *S1* to solve *P1*, with the detail presenting in Chapter 4.

Worst-case end-to-end delay analysis. *S1* cannot be scaled to two-way wireless transmission for meeting the control system stability requirement is because the difference in network delay analysis. Specifically, the two-way wireless communication will incur communication conflicts. Therefore, we propose a worst-case end-to-end delay analysis to meet control system stability requirement for two-way wireless transmission. We first carried out

the conflict analysis to get a message schedulability condition. Based on the condition, we then calculate the maximum number of conflicts that will happen during one message transmission and derive the worst-case end-to-end delay [Wang et al., 2018a]. This contribution is *S2* to solve *P1*, with the detail explaining in Chapter 5.

Offline-online network reconfiguration framework for a WCS with a single physical system. The trade-off of network delays and packet losses affecting the WCS performance motivates us to find the optimal network configuration to minimize the network-induced error. Another main difficulty of having wireless networks for the control systems is caused by interference and noise that produce time-varying fault patterns [Cerpa et al., 2005; Srinivasan et al., 2010], which motivates us to find a fast and effective way to carry out network reconfiguration at run time. We design and implement a new framework with offline and online components to do network reconfiguration for the control system with time-varying link failures [Wang et al., 2017a, 2018b]. We propose a network imperfection model in the offline part and six network reconfiguration algorithms in the online part. This contribution is *S3* to solve *P2*, with the detail showing in Chapter 6.

Dynamic packet assignment for WCS with multiple physical systems. Wireless control system with multiple physical systems will be increasingly common due to the development of IoT (Internet of Things) systems and IIoT (Industrial IoT). Network delay and packet loss will impact each control system performance differently due to different application demand. For the same delay and packet loss, we found that the physical system with more urgent application demand will have more network-induced error than the one with less urgent demand. In addition, the network paths within the wireless network can have different network characteristic in terms of delay and packet loss (e.g., the source routing and the graph routing in WirelessHart[wir, 2007]). The facts above motivate us to assign urgent demand packets to network path with low delay and high reliability. We propose a dynamic packet assignment approach to assign the packets from the WCS with multiple physical systems to the network paths, in order to reduce the overall network-induced error [Wang et al., 2018a, 2017b]. This contribution is *S4* to solve *P2*, with the detail introducing in Chapter 7.

Nuclear power plant case study. In order to evaluate the performance of our pro-

posed methods and models for WCSs, we conducted case studies of one or more primary heat exchangers (PHXs) in nuclear power plants (NPP) with a wireless sensor network. We combined a start-of-the-art cyber-physical system simulator (WCPS 2.0 [Li et al., 2015]) with an NPP simulator to mimic our wireless control system. For the wireless network, we use the TOSSIM network simulator (embedded in WCPS) with wireless noise traces from a 21-node subset of WUSTL Testbed [tes, 2017] under a wide range of wireless conditions (e.g., different levels of noise/interference). For each case study, we evaluate the network and control system performance, that is, the network delay + reliability and network-induced error, respectively. This contribution is spread from Chapter 4 to Chapter 7.

1.5 Dissertation Outline

The rest of this dissertation is organized as follows: Chapter 2 reviews existing fault tolerance and real-time techniques in WCSs. Chapter 3 introduces the background and assumptions of this dissertation. In Chapter 4, we present the energy-aware fault-tolerant network design approach and results. In Chapter 5, we do a worst-case end-to-end delay analysis for two-way wireless transmission. In Chapter 6, we build a network reconfiguration framework for link failures varying over time for a WCS with a single physical system. Chapter 7 presents the dynamic packet assignment approach for a WCS with multiple physical systems. Finally, Chapter 8 concludes the dissertation.

2.0 Related Work

The solutions for network delay and packet losses in WCS are typically divided into three categories: control only, network only, and control and network co-design solutions. In this dissertation, we only review the last two categories of fault tolerance and real-time techniques. This chapter first reviews the literature of fault tolerance and real-time solution from the WSN perspective only, then presents the recent fault tolerance and real-time research works in WCSs considering the interaction between network and control system, respectively.

2.1 Fault Tolerance Technique

2.1.1 Failures in WCSs

The failures in WCSs have hierarchical characteristics due to the joint of two subsystems, control system and wireless network. The highest level of failures of WCS is control system instability and performance degradation. One of the main causes of the instability and performance degradation of WCSs is the unreliability of the wireless network, that is, packet loss. Packet loss in WSNs is caused by two categories of failures, link and node failures due to various factors such as power depletion, environmental impact, radio interference, asymmetric communication links, dislocation of the sensor node and collision [Kakamanshadi et al., 2015]. We mainly focus on link failures in this dissertation and the literature review in the following subsections.

2.1.2 Network only Solutions

Radio link quality estimation (LQE) is the first step to tolerate the link failures, which has fundamental impact on the network performance and network protocol design [Baccour et al., 2012]. LQE is the statistical characterization of wireless links through estimation theory. PRR (packet reception ratio)-based passive LQE algorithms are presented in [Woo

and Culler, 2003; Cerpa et al., 2005]. In [Gobriel et al., 2009a], the authors show how different fault-tolerant, duplicate-sensitive, aggregation schemes for WSNs can take advantage of link quality information by an extensive simulation study. The fault tolerance techniques for the link failures are typically divided as static and dynamic solutions.

For the static fault tolerance solutions, fault tolerance in-network aggregation protocols in WSNs have been studied on the tree-based [Gobriel et al., 2006], cluster-based [Zhou et al., 2004; Mahimkar and Rappaport, 2004], multipath [Nath et al., 2008], hybrid [Manjhi et al., 2005], and gossip-based [Boyd et al., 2006; Aysal et al., 2009] approaches. Their objectives are to extract useful global information by collecting individual sensor readings and sending the aggregated information to the sink node. They are applied to monitor a specific environment, which is different from the communication in a WCS with no need of information aggregation. On the other hand, fault-tolerant node placement algorithms for link failures, k edge-disjoint algorithms with the minimum number of nodes in the network to save network energy consumption have been investigated in [Frank and Tardos, 1989; Han et al., 2010]. As the adoption of the WSN in process control system, reliable routing algorithms are proposed from wireless sensor network perspective for the WCS in [Heo et al., 2009; Han et al., 2011]. Specifically, EARQ [Heo et al., 2009] provides real-time, reliable delivery of a packet considering network energy consumption. It calculates the probability of selecting a path, using the estimates of the energy cost, delay and reliability of a path to the sink node. In [Han et al., 2011], the authors propose three routing graphs for different communication ways of transmitting messages up (sensing), down (actuation) and broadcasting messages. Based on the graphs, data link layer communication schedules are generated. However, all the aforementioned works focus solely on the network without considering the control aspect of a WCS. We solve control system instability issue by a flexible fault-tolerant node placement design and a model to quantify the network-induced imperfections (see Chapter 4).

Since the network interference is unpredictable and varies with time, the link quality fluctuates over time [Cerpa et al., 2005; Srinivasan et al., 2010]. It is necessary to tolerate the network link failures in a dynamic way. Interference can make the network disconnected and becomes inaccessible for a certain amount of time and will degrade the control system

performance. Network reconfiguration is an essential part of the network fault-tolerance technique. Based on the LQE algorithms, network reconfiguration schemes are explored in dynamic routing algorithms [Zhang et al., 2015]. In addition, several algorithms [Li et al., 2003; Li and Hou, 2004] mitigate the impact of lossy links by maintaining k-connectivity of the network. Topology control is another active research area to dynamically tolerant link failures [Ramanathan and Rosales-Hain, 2000; Santi, 2005]. Topology control is achieved by adjusting the transmit powers of nodes, which brings the positive effect of reducing contention when accessing the wireless channel and making the network more reliable. Unfortunately, these works do not consider control system performance. We design a network reconfiguration framework with a network imperfection model, indicating the impact of network delay and packet loss on control system performance (see Chapter 6).

2.1.3 Control and Network Co-design Solutions

Fault-tolerant co-design of the network and control system is effective for WCSs. Most solutions of recent research works either extract the condition/requirements of the control system or design a wireless network based on a control requirement or both. A set of topological conditions is extracted for the controller, distributed over the nodes in the network that allows the control system to be stabilized in [Pajic et al., 2011a]. A reliability analysis that evaluates a given configuration of an actively replicated networked control system and quantifies its resiliency to electromagnetic interference-induced transient faults is presented in [Gujarati et al., 2018]. In [Mouradian and Augé-Blum, 2013], the authors propose a formal verification method to derive the property of correctness probability of a given network topology based on the WSN radio links probability. This probability must meet the requirements of the control application; otherwise, the system must be changed to increase the probability. Other co-design solutions are case studies to observe the interaction between the network dynamics and control system performance. For example, a case study is conducted to see the interaction between the model predictive control and network routing schemes in [Li et al., 2016] with the observation of control system having different levels of resilience to packet loss for sensing and actuation. However, none of these works address the tradeoff

between network delay and packet loss in WCSs, nor present the interaction between the network reconfiguration and control. We conduct a case study to show how the network reconfiguration affects the control system performance (see Chapter 6).

2.2 Real-time Technique

2.2.1 Network Delay in WCS

There are mainly two kinds of message delays in WCSs: sensor-controller delay and controller-actuator delay. The sensor-controller delay represents the time interval from the instant when the physical plant is sampled to the instant when the controller receives the sampled message; and the controller-actuator delay indicates the time duration from the generation of the control message at the controller until its reception at the actuator. In control theory, these delays cause phase shifts that limit the control bandwidth and affect closed-loop stability and performance [Park et al., 2018].

2.2.2 Network only Solutions

In order to guarantee the application deadline of a WSN, worst-case end-to-end delay analysis is an important research area to study. In recent research works [Saifullah et al., 2010, 2011, 2015], the worst-case end-to-end delay analysis for source and graph routing based on wirelessHart standard to guarantee the real-time communication in WCSs are discussed. However, they all consider the network flow deadlines are smaller than their periods. We focus on a general case when it is possible that the transmission deadlines are greater than their periods. To the best of our knowledge, no other works are studying this case before, but it is common in real-time WCSs. We came up with a worst-case end-to-end delay analysis, which helps a network design to guarantee the control system deadline (see Chapter 5).

Dynamic real-time network scheduling is an effective solution to constrain network delays. Real-time TDMA scheduling algorithms in WSNs are studied from many aspects to reduce the end-to-end network delays: parallel transmission design [Gobriel et al., 2009b];

packet prioritization [Liu et al., 2006; Zhang et al., 2015]; and optimization with other constrains, such as energy consumption [Gu et al., 2009] and the number of packet drops [Hong et al., 2015]. However, the above works do not consider control system application demands. We propose a dynamic packet assignment approach considering dynamic control system application demands and network reliability + delay impact on the control system performance (see Chapter 7).

2.2.3 Control and Network Co-design Solutions

For the co-design of real-time network and control system, [Li et al., 2015] explores a data link layer real-time communication protocol with slot stealing algorithm [Gobriel et al., 2009b] and event-based communication on top of the WirelessHART protocol [wir, 2007] to reserve time slots for emergency packets. However, neither network-induced error minimization is considered, nor multiple control systems are involved. Online dynamic link layer scheduling algorithms have been proposed in [Hong et al., 2015; Zhang et al., 2017] to meet the deadline of a rhythmic flow and minimize the number of dropped regular packets in a centralized and distributed way, respectively, based on a rhythmic task model proposed in [Kim et al., 2012]. While the impact of network dynamics on existing network flows is minimized, overall control system performance (different control system application demands) is not considered ,and there is no case study for real-world applications. In [Gatsis et al., 2014], the authors first abstract the control performance requirements as desired decrease rates of Lyapunov functions. Since the channel conditions on wireless medium not only change unpredictably overtime but also differ among users, they present a framework for designing opportunistic channel-aware centralized schedulers for a WCS of multiple control tasks over a shared wireless medium. In their later work [Gatsis et al., 2016], they derive a sufficient decoupling condition for the random access policy employed by each node in the wireless network given control dynamics. They then design a random access policy that can adapt to the dynamic of the physical system online. The end-to-end real-time guarantee between interfaces of distributed components in WCS besides the wireless network is proposed in [Jacob et al., 2016]. A real-time high-speed wireless protocol is explored in [Wei et al., 2013].

However, the above works only consider the control stability and not the overall control system performance. To the best of our knowledge, a cross-layer dynamic packet assignment has not been studied in a WCS with multiple physical systems, which is our last work in this dissertation (see Chapter 7).

2.3 Summary

This chapter reviews related work in the fields of fault tolerance and real-time techniques in WSNs and WCSs. We introduce the failures in WCSs and state-of-the-art fault tolerance techniques from only the WSN aspect and control and network interaction aspect. On the other hand, we discuss the network delay in WCSs and the real-time solutions of network only and control and network co-design.

Motivated by the WCS challenges and shortcomings of existing work, this dissertation targets at the work that lies at the intersection of fault tolerance and real-time scheduling for WCSs.

3.0 Background and Assumptions

In this chapter, we first introduce the background for this dissertation with (a) the control system we use to do our performance evaluation and (b) the network protocol we apply to do wireless transmission. Then we discuss the definitions and assumptions we made for this dissertation before we go into more details in the following chapters.

3.1 Background

3.1.1 Primary Heat Exchanger System

The control system we study in this dissertation is a remote controller controlling one or more primary heat exchanger systems in a nuclear power plant (NPP). A new trend in nuclear power plants is to use several Small Modular Reactors (SMRs) rather than a single large reactor [Greene et al., 2010], due to the flexibility and cost-benefit of starting and stopping SMRs. Given the large number of SMRs in a modern NPP, the cost and difficulty of cabling all sensors and actuators would be prohibitive. Typically there is one primary heat exchanger system (PHX) and two secondary heat exchangers in each SMR. Note that we only model the PHX in this dissertation, since the two secondary heat exchangers are backups for safety and would follow a similar approach with a different network. A PHX in an NPP is modeled as a nonlinear system and has as its main function the exchange of heat from inside of the reactor to the outside, which controls the pressure and the temperature of the reactor. A PHX makes many measurements, three of which are the focus of this dissertation, given its importance to the NPP control, namely the outlet hot leg temperature, the inlet hot leg temperature, and the mass flow rate. In our NPP, these measurements are periodically sent via a wireless network to the remote controller locating in the operator control room. The remote controller will compute the control signal using the received measurements and then send back the control signal to the actuators to actuate the PHX. When there are

multiple PHXs, they all share one wireless network to transmit the measurement packets to the centralized remote controller and send the control signals back to each PHX.

We use the control system we discussed above to do case studies in this dissertation to evaluate our models and approaches. We conducted wireless control case studies for one PHX in Chapter 4 and Chapter 6, and the case study for three PHXs in Chapter 7. Note that although important for NPPs, safety issues are beyond the scope of this dissertation and we focus on the feasibility of making control system stable and reducing network-induced error.

3.1.2 Ridesharing Protocol

The network protocol we use in this dissertation is based on a TDMA protocol [1], ridesharing [Gobriel et al., 2006] to tolerate link failures. Ridesharing is an in-network aggregation protocol, which is used to monitor a certain environment. We define *relay nodes* as the sensor nodes passing messages in between the source and destination. In ridesharing (shown in Figure 5), relay nodes are organized in a children-parent relationship. There is a direct wireless link between the child and parent node, where a *child node* (e.g., C_1) is a transmitter, and a *parent node* (e.g., P_1) is a receiver of the message from its child node. To make the network more reliable, besides a relay node which is called *primary node*, we place one or more relay nodes as backups which are called *backup nodes*. For example, P_2 and P_3 in Figure 5 are the backup nodes of the primary node P_1. The primary node and its backup nodes are called *sibling nodes* (e.g., P_1, P_2 and P_3 are sibling nodes). Thus, a node has one primary parent and zero, one or more backup parents, and also has zero, one or more siblings. If the backup parent finds out (while overhearing) that the primary parent did not send out the values it should receive from their children, the backup parent will compensate for the primary parent. For example, in Figure 5, C_1 has one primary parent P_1, two backup parents, P_2 and P_3. The link between C_1 and P_1 fails. When C_1 broadcasts its message in

[1] TDMA scheduling can achieve the real-time transmission, since the transmission slots are scheduled ahead of time. The non-TDMA scheduling protocols can also achieve real-time transmission purpose, such as ad-hoc protocols. The network analysis in terms of network delay and message losses is the same as TDMA protocols.

time slot 0, P_2 and P_3 receive the message. When P_1 broadcasts its message in time slot 1, P_2 overhears P_1's message and knows that P_1 did not receive C_1's message. At time slot 2, P_2 aggregates C_1's measurement with the measurement it senses itself and sends a message with an added field indicating that it received C_1's message. P_3 overhears P_2's message knowing that P_2 already handled C_1's message and P_3 discards C_1's message. In time slot 3, P_3 sends out a message with its own sensed measurement. In this way, the backup parent P_2 tolerates the link failure between C_1 and P_1 and improves the network reliability. If P_2 fails to handle the fault (i.e., the link between C_1 and P_2 fails), P_3 does so in a similar manner. The more backup nodes, the more reliable the network. In this dissertation, we modified the ridesharing protocol, by concatenating measurements into one message, instead of aggregating the measurements. We remove the measurement aggregation is because there are different types of measurements that cannot be aggregated in WCS applications. We choose not to send each measurement with one message is to save network energy and reduce the network contention, since more messages will bring more chances of transmission conflicts. But we only concatenate the measurements sensed at the same control sampling period due to the limited size of the message. Therefore, when a relay node receives multiple messages before it sends out its own message, it will first discard the duplicate messages, which were already received by its siblings via overhearing its siblings' messages; then it concatenates the measurements of the remaining received messages of the same control sampling period to the message it is going to send (note that the concatenated measurements are distinct, since the duplicate messages are discarded). For the communication frequency, we use single-channel communication in Chapter 4, 5 and 6, and use multi-channel communication in Chapter 7.

3.2 Assumptions and Definitions

In this dissertation, we make definitions as follows.

- **Link success ratio and delivery ratio**. Links in wireless network fail independently with a certain probability and we define link success ratio (LSR) as the probability a message can be sent out successfully on that link. We use average LSR over all the

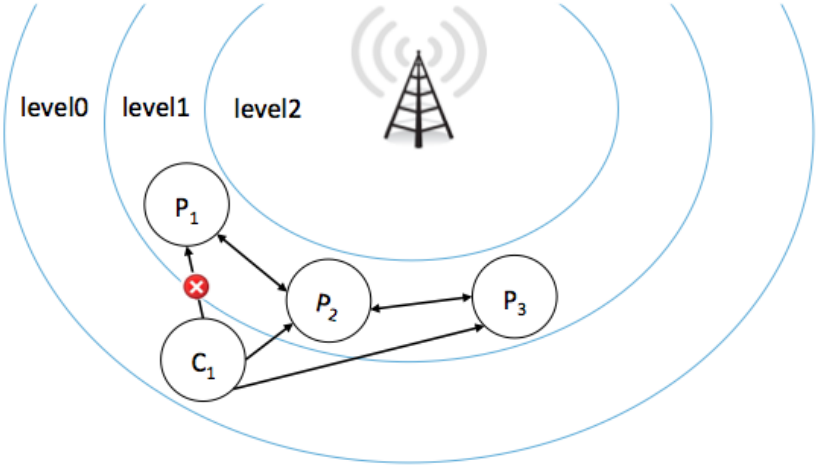


Figure 5: Ridesharing protocol example illustration

network links as the indication of the average network interference. We use network delivery ratio (DR) as the network reliability indicator, that is, the ratio of arrived measurements or signals or messages in the destination ($DR \in [0, 1]$). Typically, the bigger the LSR value, the bigger the DR.

- **Sleep nodes and active nodes**. Sleep nodes are the relay nodes in sleep mode that cannot receive and transmit messages, while active nodes are the relay nodes that can receive and transmit messages. To save energy, we make relay nodes to sleep for a while. We also wake up the sleep nodes to be active as needed by sending reconfiguration messages periodically. Thus, in this dissertation, the sleep nodes will wake up periodically to listen to the reconfiguration messages to see whether they need to wake up. The reconfiguration messages are sent in reserved slots, as in many proposed TDMA algorithms [Yackovich et al., 2011]. Thus, active nodes in this dissertation listen to the messages of their children, overhear the messages of their siblings, listen to the reconfiguration messages and send out its own messages.

- **Neighbors of a relay node**. Neighbors of a relay node A are defined as the network nodes that are one hop from A.

We make assumptions as follows.

- There is no message retransmission in our wireless transmissions. Similar to [Li et al., 2015], when a message gets lost during a transmission, the controller/actuator uses the last received measurements/control signal to carry out the control algorithm/actuation, respectively.
- All sensors attached to one physical system send out the measurements at the same period as the control period (i.e., the sensing sampling period is the same as the control sampling period).
- In a traditional wired control system, the packet loss and delay can be ignored.
- The reconfiguration messages to activate (i.e., wake up sleep nodes) or deactivate (i.e, put active nodes to sleep) relay nodes never get lost.
- The links between parent nodes and children nodes can fail, but the links between sibling nodes never fail.
- An active node only overhears its sibling's messages if the sibling node is scheduled to transmit messages before it.
- We ignore the time delay of the control signal computation in the remote controller.

4.0 Fault-tolerant Network Design

Control system stability is critical for physical plants, since system instability can result in plant damage and severe safety issues [Zhang et al., 2001; Zhang and Yu, 2008; Jusuf and Joelianto]. In WCS, network delay and packet loss are the potential threats to control system stability. Given a control system stability requirement in terms of network delay and packet loss, we first propose a fault-tolerant node placement design. We then develop a model quantifying the stability requirement for different network topologies. We can determine the initial network topologies with the minimum number of active nodes to meet the requirement for different average LSR values. Finally, we evaluate our model by simulating a WCS with one PHX and a 12-hop wireless network.

4.1 Introduction

In [Wang et al., 2016], the control engineering researchers use system damping ratio to quantify the system stability. System damping ratio is a dimensionless measure describing how oscillations in a system decay after a disturbance. In other words, the damping ratio is a measure of describing how rapidly the oscillations decay from one bounce to the next. Different system has their own system damping ratio threshold to bound the system stability (e.g., the damping ratio threshold of a PHX is 0.2 [Wang et al., 2016]). The less the damping ratio, the more time system need to come to the rest, the less stable. Network health (NH) is derived by the damping ratio (the derivation process details can be found in [Wang et al., 2016]). NH is shown in Equation 4.1 in terms of end-to-end network delay $D_{network}$ and delivery ratio (DR), where p_1, p_2, p_3 are constants (they are real numbers and could be positive or negative) for a specific controller [Wang et al., 2016]. The end-to-end delay $D_{network}$ is the time delay of any one message from the time it is sent out to the time it is received by the controller (for one-way wireless transmission)/actuator (for two-way wireless transmission). We will consider the $D_{network}$ of one-way wireless transmission in this chapter

$$NH = p_1 D_{network}^2 + p_2 D_{network} + p_3 - (1 - DR) \tag{4.1}$$

and two-way wireless transmission in Chapter 5.

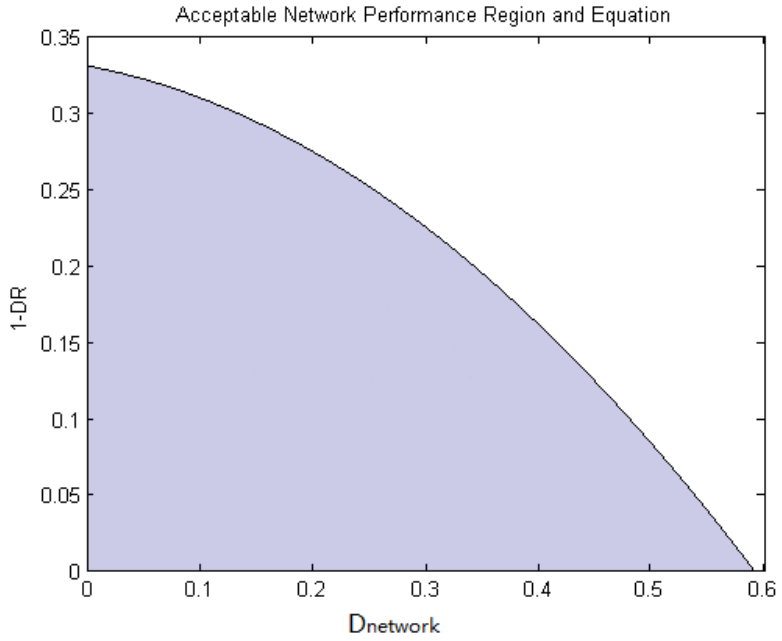


Figure 6: Example of network performance region of a PHX with damping ratio threshold 0.2

When $NH = 0$, the system damping ratio equals to the damping ratio threshold (system is stable) and p_1, p_2, and p_3 can be calculated accordingly (NH is correlated with system damping ratio, which equals to the damping ratio threshold). When $NH > 0$, the system damping ratio is greater than the damping ratio threshold (system is stable). When $NH < 0$, the system damping ratio is less than the damping ratio threshold (system is unstable). In other words, the control engineers have came up with a general control system stability requirement that when $NH \geq 0$, the control system is stable. Figure 6 shows an example of acceptable network performance region in terms of network delay $D_{network}$ and network

reliability $1 - DR$ with damping ratio threshold of 0.2 of one PHX. Specifically, the black curve is when $NH = 0$; the blue region is the acceptable region when $NH > 0$ and damping ratio is greater then 0.2; and the white region is when $NH < 0$ and damping ratio is less than 0.2 and system is unstable. We can get the maximum network loss rate $(1 - DR)^{max}$ by setting $D_{network} = 0$, which means the maximum network loss rate the control system can tolerate to remain stable. Similarly, we can get the maximum network delay $D^{max}_{network}$ the control system can tolerate by setting $1 - DR = 0$. Both $(1 - DR)^{max}$ and $D^{max}_{network}$ can be used in the network design phase.

Given the NH in Equation (4.1), our goal is to design a fault-tolerant wireless network to meet the requirement ($NH \geq 0$) with the minimum active relay nodes to save network energy for one-way wireless transmission (messages sending up to the remote controller). In this chapter, we first design a fault-tolerant network node placement for WCS and propose a model to quantify NH and determine the minimum number of active nodes to have in the network, to meet $NH \geq 0$ requirement. We will scale our approach to two-way wireless transmission in Chapter 5.

4.2 Network Node Placement Design

Since the remote controller is not on the same site (physically separated) as the sensors and actuators, the periodic measurement messages will be transmitted through relay nodes to the remote controller. In this chapter, we assume each wireless network link fails independently with a the same probability. As the first network design step, we place relay nodes in between the measurement sensors and the remote controller. We divide the network area into two regions, namely *k-connected region* and *relay region* (see Figure 7). A *virtual root* demarcates the connection between the two regions. The reason why we have two regions is that a physical system could be very large and the sensors sensing different types of measurements could be dispersed attached to it and we want to have the measurements sent to one location (virtual root) first, then pass them together to the remote controller. After we place the relay nodes in the network area, we describe how we generate a network topology

set that will be used to determine the initial topology to meet the control system stability requirement. Note that *topology* in this dissertation is the set of nodes that are active and that are not (i.e., nodes that are asleep), assuming that there are a certain number of nodes already physically placed (Section 4.2.1 and 4.2.2 discusses in detail how nodes are placed). Every time a topology change occurs, it means that some nodes are activated and some are put to sleep. Finally, we introduce the TDMA scheduling for each region based on the node placement design.

4.2.1 k-connected Region

Optimal node placement in a wireless sensor network has been shown to be NP-hard [Han et al., 2010]. Given that we only consider link failures, we apply k edge-disjoint algorithms [Frank and Tardos, 1989; Han et al., 2010], instead of k node-disjoint algorithms [Zhang et al.; Bredin et al.]. We define a *k-connected* region as the nodes and links having k edge-disjoint paths from each measurement sensor to the virtual root. In the k-connected region, we apply Han's algorithm to place relay nodes [Han et al., 2010]. To get k edge-disjoint paths, we solve an optimization problem for finding a minimum-cost subgraph H of a digraph $G = (V, E)$ such that H contains k edge-disjoint paths from a fixed node of G to any other node, which can be reduced to a weighted matroid intersection problem [Frank and Tardos, 1989]. A common basis of these matroids corresponds to a subgraph, that is, the union of k disjoint spanning trees. Therefore, finding k edge-disjoint paths of a graph is equivalent to finding the subgraph of k disjoint spanning trees with minimum cost (e.g., [Yang, 2005]). We also add $(k-1)$ backup nodes to the virtual root to improve reliability. Note that we call the virtual root and its backup nodes as virtual roots and treat them as one node when generating the k edge-disjoint paths. The last relay node of each path can reach all the virtual roots, as we place virtual roots as close as possible together (will explain the reason in Theorem 4.2.1). In k-connected region, for k edge-disjoint paths of each measurement sensor, the path with the minimum number of relay nodes is called the *primary path*, the path with the second smallest number of relay nodes is called the first *backup path*, and so on.

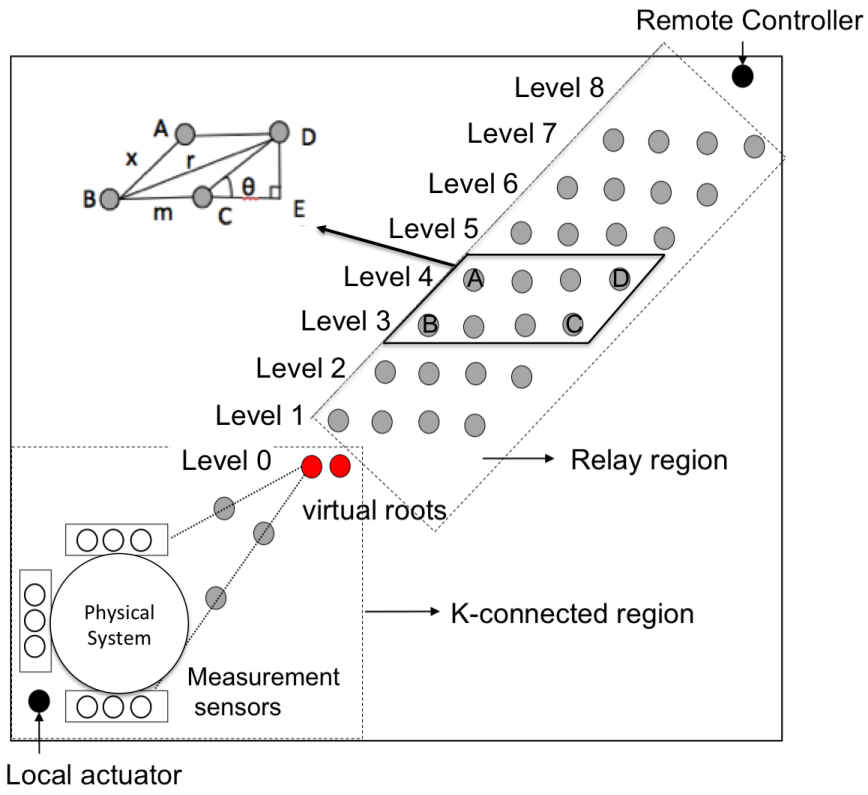


Figure 7: Fault-tolerant relay nodes placement design for a single control system (2-connected region and 3 lines of backup nodes in relay region)

4.2.2 Relay Region

In the relay region, primary relay nodes are placed in a "straight line" between the virtual roots and the remote controller, called the *line of primary relay nodes*. The distance between two consecutive primary nodes is the same. In addition, we place b *lines of backup nodes* ($b \geq 1$), that is, each primary relay node have b backup nodes. For example, in Figure 7, A and B are the primary nodes, and D and C are the backup nodes. Horizontally, the level where one primary node and its backup nodes are located is called one *level*. Each node in level h is able to listen to all the nodes in its lower level $h-1$ (the level closer to the physical system) and its upper level $h+1$ (the level closer to the remote controller). To ensure each node can hear all the nodes one hop from it with the minimum number of relay nodes, we place the nodes in each level as close as possible ("horizontally" right next to each other), as proved below. We will also place a virtual root and its backups as close as possible for the same reason. Thus, we can assume the side link between any two nodes (siblings) in the same level (e.g., $A \rightarrow D$) never fails, given they are so close (it is one of the assumptions mentioned in Section 3.2).

Theorem 4.2.1. *Assuming faults are independent events, adding backup nodes as close as possible to each primary relay node, minimizes the number of relay nodes in the relay region.*

Proof. Referring to the inset in the upper left corner of Figure 7, note that the distance between two consecutive primary nodes (e.g., A and B) is x, which is a function the radio technology used and the power level each node transmits. The maximum distance between a primary node and its furthest backup node (e.g., B-C) is m. The maximum distance between the sender and any backup receivers (e.g., B-D) is r. We use an auxiliary imaginary point, E, that forms a right angle between D and the primary node B. θ is the angle DCE. Therefore, we can get $(m+\cos\theta x)^2+(\sin\theta)^2x^2 = r^2$. Solving the equation, $x = \frac{\sqrt{4r^2-4(\sin\theta)^2m^2}-2\cos\theta m}{2}, r \geq \sin\theta m$. To minimize the number of primary nodes, we need to maximize x, the distance between two consecutive primary nodes. Therefore, m should be as small as possible (note the negative sign of factors containing m), which means primary and backup nodes in the same level should be placed as close as possible, since θ and r are constant.

□

4.2.3 A Network Topology Set Generation

After placing the relay nodes in the network area, backup paths in the k-connected region and backup nodes in the relay region can be put to sleep to save energy when the network condition is bad (LSR is low), and some of the backup paths in the k-connected region or backup nodes in the relay region need to be activated when the network condition is good (the LSR is high). Thus, we generate a network topology set with the different number of active nodes based on the node placement design. The reason why we need different typologies with different levels of redundancy is different typologies have different packet loss, delay and energy consumption that could fit to different physical system needs at different time. For example, the more redundancy, the less packet loss, but more network energy consumption and network delay, that is suitable to the physical system with high delay but low measurement loss tolerance. Specifically, we activate the primary paths for each measurement sensor in the k-connected region and the primary line of relay nodes in the relay region to make sure the network is connected. Then, for each measurement sensor, we activate the backup paths in k-connected region from the first backup path (the shortest) to the last backup path (the longest) one path at a time. Meanwhile, we also activate one node at a time in the first line of backup nodes from the virtual roots to the remote controller, then add one node at a time in the second line of backup nodes, etc. Thus, we can generate a network topology set with different number of active nodes (e.g., a topology with a 2-connected region and one and a half lines of backup nodes in the relay region or a topology with a 1-connected region and three lines of backup nodes in the relay region).

4.2.4 TDMA Scheduling

The TDMA scheduling in k-connected region is done first and then, after synchronizing at the virtual roots, the TDMA scheduling for the relay region. This is due to different node placement design in the two regions.

In the k-connected region, one relay node could be shared by multiple paths of different measurement sensors. Thus, we design a TDMA scheduling to make sure each relay node receives all the messages from its children before transmitting its own message. For example,

$$p_c = \begin{cases} LSR^{l_c}_{l_1}, & \text{if } k = 1 \\ k(LSR^{l_c}_{l_1} + \sum_{i=2}^{k} LSR^{l_c}_{l_i}(\prod_{j=1}^{i-1}(1 - LSR^{l_c}_{l_j}))), & k > 1 \end{cases} \tag{4.2}$$

in a 2-connected region, two measurement sensors (numbered as 0 and 1) send messages to a virtual root (numbered as 3) by 5 relay nodes (numbered as 2, 4, 5, 6, and 7). The two paths of sensor 0 are $0 \to 7 \to 6 \to 3$ and $0 \to 2 \to 5 \to 4 \to 3$ and the two paths of sensor 1 are $1 \to 7 \to 5 \to 4 \to 3$ and $1 \to 2 \to 5 \to 6 \to 3$. One of the feasible TDMA schedulings is 0-1-2-7-5-4-6-3. The relay nodes in k-connected region only listen to its children's messages and do not overhear the messages from other relay nodes. Thus, the virtual roots can (and probably will) receive duplicate messages, discard them and only send out messages with the measurements of the remaining received messages.

In the relay region, the relay nodes broadcast messages from the lowest level (where the virtual roots are located) to the highest level (where the remote controller is located). Within each level, the primary node will broadcast first, then the first, second, and third active backup node (if any), in order. Each relay node will overhear the messages of some siblings before it sends out its own message. Therefore, the more active backup nodes in the network, the more messages are sent, and thus the higher network delay.

4.3 A Model for Quantifying NH

We propose a model to quantify the DR in terms of average LSR and the worst-case end-to-end network delay, $D^{worst}_{network}$, in order to quantify NH for the network topology set with different number of active nodes. We can determine the network topology with the minimum number of active nodes to meet the requirement $NH \geq 0$, for different LSR values.

4.3.1 Delivery Ratio Calculation

We calculate the DR at the remote controller as the ratio of the expected number of measurements received at the remote controller and the total number of measurements it should receive for the different number of active nodes in the network given an average LSR value. Specifically, we calculate the probability of measurement reception at the virtual roots in k-connected region from the measurement sensors. We then design a dynamic programming algorithm to calculate the probability of different measurement receiving situations level by level from the virtual roots to the remote controller for the relay region.

For the k-connected region, we assume there are m measurement sensors and each measurement sensor sends out one message with one measurement value in every control sampling period. We denote the number of hops in the i^{th} path from measurement sensor c to virtual roots is l_i^c. The probability of the measurement sent from measurement sensor c and received by at least one of the virtual roots is shown in Equation 4.2, where $LSR^{l_1^c}$ is the probability the primary path transmits the measurement sent from measurement sensor c successfully over l_1^c hops to one of the virtual roots; $LSR^{l_i^c}(\prod_{j=1}^{i-1}(1 - LSR^{l_j^c}))$ is the probability the i^{th} backup path transmits the measurement sent from measurement sensor c successfully over l_j^c hops and the primary path and $i-1$ previous backup paths fail. As mentioned in Section 4.2.1, if we have k edge-disjoint paths in the k-connected region, we have k virtual roots, each one can receive the last nodes in all the paths. Thus, the probability that at least one of the virtual roots receives the measurement c should have the factor k as a multiplier.

For the relay region, each node sends one message with one or more measurements that it has received to its parent nodes in the next level. According to the ridesharing protocol in Section 3.1.2, the number of measurements received by each node varies (due to network errors) from 0 to the number of sensed values, but the total number of measurements of all the active nodes in current level that are going to send out to the next level cannot exceed the total number of measurements (or sensed values), due to the overhearing mechanism.

We introduce the concept of *state*, which represents the measurement receiving situation of a level. A state of level h, s_h is $[c_{h,1}, c_{h,2}, ..., c_{h,n_h}, m_h, p_h]$, where n_h is the total number of active nodes in level h, $c_{h,i}$ is the number of measurements that are going to send to level

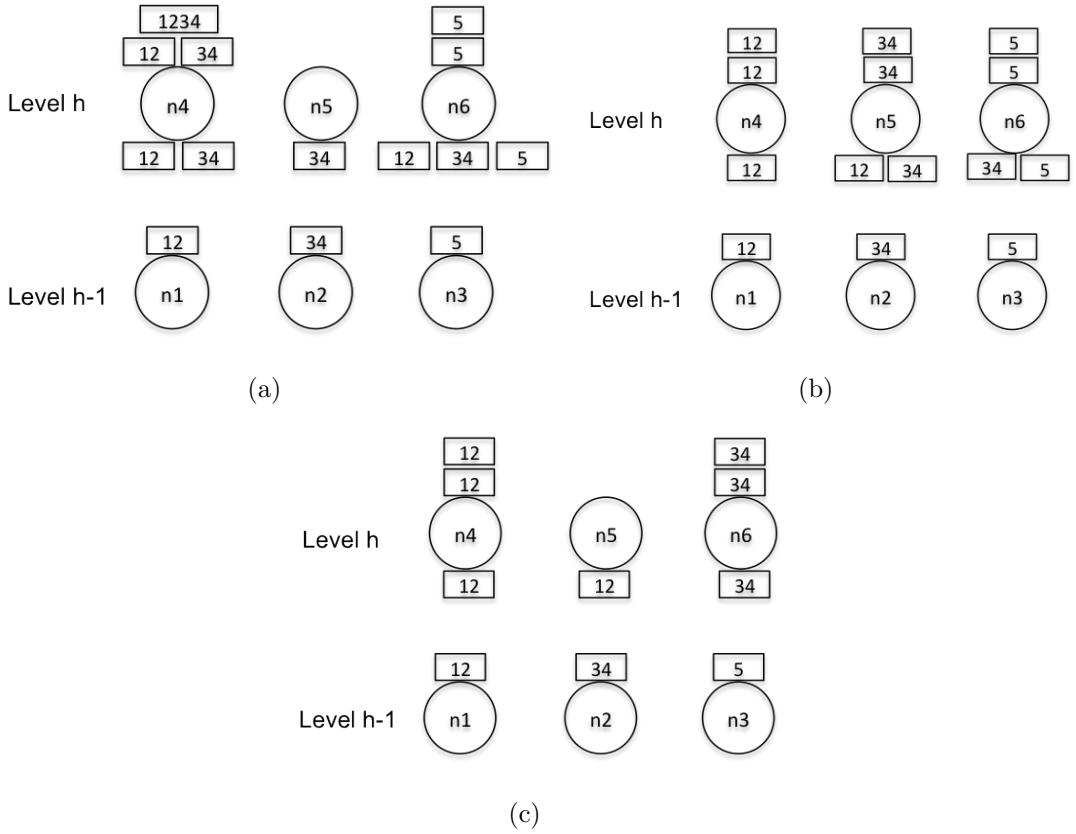


Figure 8: Three example states of level h generated from one of the states of level $h-1$

$h+1$ by the i^{th} node in level h; we define the array $[c_{h,1}, c_{h,2}, ..., c_{h,n_h}]$ as the measurement-sending array of level h; $m_h = \sum_{i=1}^{n_h} c_{h,i}$ $(m_h \leq m)$, which is the total number of measurements that are going to pass to the next level; p_h is the probability of state s_h occurring. We define $x_{h,i}$ as the number of received messages from level $h-1$ that are going to be concatenated into one message by the i^{th} node in level h (i.e., the number of remaining messages, since the duplicated messages were already discarded via overhearing mechanism), so $x_{h,i}$ messages of the i^{th} node contain $c_{h,i}$ measurements. We define y_h as the total number of messages that are going to be used to do the measurement concatenation, $y_h = \sum_{i=1}^{n_h} x_{h,i}$. For example in Figure 8(a), there are three nodes in level $h-1$ and h, respectively; 5 measurements are sent from level $h-1$ to level h; the rectangle represents a message with one or more

$$p_h = p_{h-1} \prod_{i=1}^{n_h} ((1 - LSR)^{i-1} LSR)^{x_{h,i}} \times ((1 - LSR)^{n_h})^{y_{h-1} - y_h} \tag{4.3}$$

measurements and the circle represents a relay node; the rectangles below the nodes represent the messages received from the previous level; the rectangles above the nodes represent the messages received from previous level that are used to do measurement concatenation; and the top rectangles represent the messages are going to send to the next level. Figure 8(a) shows that n1 sends a message with two measurements, 1 and 2; n2 sends a message with two measurements, 3 and 4; and n3 sends a message with one measurement 5; n4 receives the messages from n1 and n2; n5 receives the message from n2; and n6 receives all the message from level $h-1$. n4 concatenates the four measurements to its message from the received two messages ($x_{h,1} = 2$) and sends out the message. n5 and n6 overhear n4's message, knowing that n4 already got messages from n1 and n2, and discard the messages received from n1 and n2 (if any). n5 does not send a message ($x_{h,2} = 0$). n6 sends a message only including measurement 5 from one received message ($x_{h,3} = 1$). The measurement-sending array of level h is [4, 0, 1].

s_h is one of the states calculated recursively from a state in the previous (lower) level $h - 1$, s_{h-1}. For example, let the previous level $h - 1$ have n_{h-1} nodes, each node in level $h - 1$ sends one message to the upper level h. Note that $m_h \leq m_{h-1}$ and $y_h \leq y_{h-1}$ (the reason is discussed above), where m_{h-1} is the total number of measurements sent from lower level $h - 1$, and y_{h-1} is the total number messages used for measurement concatenation of level $h - 1$. The probability of state s_h can be computed recursively as shown in Equation 4.3.

The above notation is a simplification of the problem, because there are many possible states at level h that can be derived from many possible states at level $h - 1$. For example, Figure 8 shows three example states of level h that generated from one state of level $h - 1$:

$$DR = \frac{\sum_{i=1}^{m}(p_{RC}(i) \times i)}{m} \tag{4.4}$$

the measurement-sending array of the state of level $h-1$ in the example is $[2, 2, 1]$. From Figure 8(a) and 8(b), the measurement-sending arrays of level h are $[4, 0, 1]$ and $[2, 2, 1]$, respectively. Figure 8(c) shows that the measurement 5 is lost during the transmission and the measurement-sending array of level h is $[2, 0, 2]$. Therefore, strictly speaking, we should treat each element of the state representation with another superscript, as follows. A state k of level h is represented as $s_h^k = [c_{h,1}^k, c_{h,2}^k, ..., c_{h,n_h}^k, m_h^k, p_h^k]$ computed from a state j of level $h-1$, s_{h-1}^j similarly defined. Note that the states of level 0 (where virtual roots locate) are computed by enumerating all possible values in the measurement-sending array and calculating the corresponding occuring probability from Equation (4.2). To calculate the probability of all possible number of measurements received by the remote controller, we need to enumerate all possible states each level could have. For each level, we carry out the calculation with two phases, namely, a states-generating phase and states-combining phase. For the former, one or more states are generated by one of the states of the previous level (like the examples shown in Figure 8). Formally, the new states, s_h^k of level h that are generated by one state, state s_{h-1}^j in level $h-1$, are the combinations of all possible values of $c_{h,i}^k$ with the following conditions: $m_h^k \leq m_{h-1}^j$, $0 \leq c_{h,i}^k \leq m_h^j$ $(1 \leq i \leq n_h)$. For the states-combining phase, the probability of states with the same measurement-sending array (generated from different states of level $h-1$), $[c_{h,1}^k, c_{h,2}^k, ..., c_{h,n_h}^k]$ are summed up and combined into one state. Since we compute each state's probability iteratively from the measurement sensors to the remote controller, all possible number of measurements will be accounted for at the remote controller. The delivery ratio at the remote controller is calculated in Equation 4.4, where m is number of measurements sent from the sensors, and $p_{RC}(i)$ is the combined state probability that the remote controller receives exactly i measurements. $p_{RC}(i)$ is calculated

$$D_{network}^{worst} = n_{active}\Delta t \tag{4.5}$$

recursively from the level of measurement sensors to the level of the remote controller, where the states of each level in relay region are calculated during the phases of states-generating (using Equation (4.3) for each state probability calculation) and states-combining.

4.3.2 Worst-case End-to-end Delay and NH Calculation

Based on the ridesharing protocol [Gobriel et al., 2006], which is a modification of TDMA scheduling, each active node is reserved one time slot for transmitting the measurement message. We focus on one-way wireless transmission in a WCS in this chapter and there is a pipeline of messages one after another (periodic messages). If the level difference between the two messages from consecutive control sampling periods is less than 3, there will be message transmission conflict. The messages are unschedulable (messages cannot be delivered to the destination) and will be stuck at a certain level forever. We will explain later in the Lemma 5.2.1 in Chapter 5. Once the messages are schedulable (messages can be delivered to the destination), it is not possible that the messages of previous control sampling period are conflicted by the messages of the current period, because previous messages are always sent and arrive at the remote controller before the current message. Therefore, the worst-case network delay is calculated in Equation 4.5 where n_{active} is the number of the current active nodes and Δt is the time slot of TDMA scheduling.

Therefore, for a given LSR, we can calculate NH using Equation (4.1) for the topology set (generated in Section 4.2.3) with the different number of active nodes in the network, by calculating DR from Equation (4.4) and $D_{network}^{worst}$ from Equation (4.5). Then, we select the topology with the minimum number of active nodes that meets the constraint $NH \geq 0$ for different LSR values.

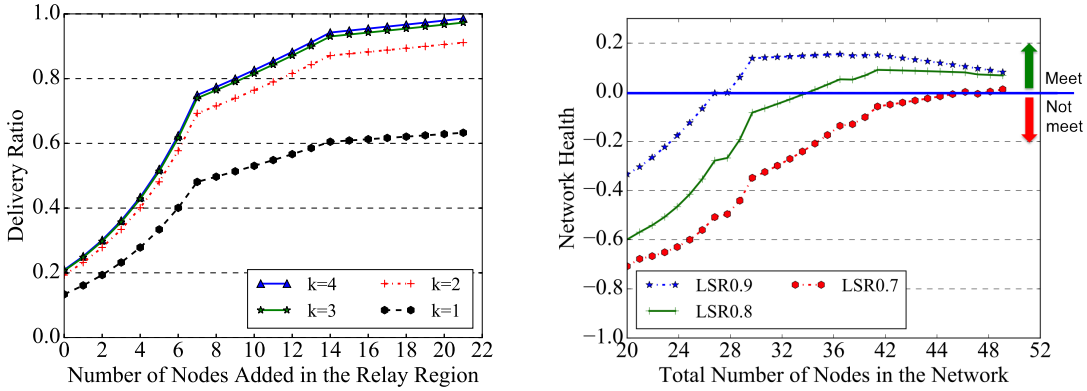
4.4 Performance Evaluation

We carried out a case study of a WCS with one PHX in an NPP to evaluate our model, with the control system stability requirement as $p_1 = -0.714$, $p_2 = -0.138$, and $p_3 = 0.330$ [Wang et al., 2016] (Equation (4.1)). We use the TOSSIM network simulator [Levis et al., 2003] with wireless noise traces from a 21-node subset of the WUSTL Testbed [tes, 2017]. Similar to [Li et al., 2015], we use controlled Received Signal Strength (RSSI) [Lee et al., 2007] with uniform gaps to simulate various LSR values. For the wireless network, we evaluate a 12-hop (5 hops in the k-connected region and 7 levels in the relay region) network of both the model results and simulation results. For our model analysis, we place 3 lines of backup nodes in the relay region ($b = 3$) to avoid long-running computations. For the simulation analysis, we place 7 lines of backup nodes in the relay region ($b = 7$) for sensitivity analysis purposes. We set the maximum connectivity degree of k-connected region as 4 (i.e., $k \leq 4$). Starting with one line of primary nodes and fixed k-connected region, we activate lines of backup nodes from virtual roots to the remote controller (as discussed in Section 4.2.3). We assume the same time slot duration, $\Delta t = 0.01s$, of WirelessHart [wir, 2007]. We evaluate three metrics: DR, NH, and the minimum number of active nodes in the network for different LSRs with $NH \geq 0$.

4.4.1 The Model for Quantifying NH Result

The calculated DR from Equation (4.4) at the remote controller is shown in Figure 9(a) for the average LSR 0.8 (other values of LSR show the same trend). Figure 9(a) represents the calculated DR as a function of the number of added backup nodes in the relay region for different values of k-connected region. Three interesting observations are as follows. First, the inflection points happen when all primary relay nodes have the same number of backup nodes (every 7 nodes or a complete line in this case). Second, while adding the first line of backup nodes, the DR exponentially increases because the probability of sending a message successfully from virtual roots to the remote controller is $(2LSR(1 - LSR) + 2LSR)^b \times LSR^{7-b} = 2^b LSR^7(2 - LSR)^b$, where b is the number of backup nodes added in the first line

of backup nodes; $(2LSR(1-LSR)+2LSR)^b$ means the probability of sending a message successfully from the level of virtual roots to the level of the last backup nodes added in the first line of backup nodes. As b increases, the probability increases exponentially. Third, the slope decreases when adding more lines of backup nodes. Figure 10 demonstrates the reason: the probability of using the last active node in one level handling a message decreases as the number of backup nodes in each level increases, which explains why the slope of the first line of backup nodes is the steepest.



(a) Delivery ratio at remote controller; average $LSR = 0.8$

(b) NH with different LSRs: 0.7, 0.8 and 0.9.

Figure 9: The model of quantifying NH results

Figure 9(b) shows the average NH for different LSRs. The system should operate above $NH \geq 0$, when the network meets the control system requirements (above the horizontal line in Figure 9(b)). From this result, we are able to select topologies with the minimum number of active nodes that meet $NH \geq 0$ for different LSR values (more results are shown in Table 1). For example, given LSR value of 0.8, the minimum number of active nodes in a topology meeting the requirement is 34 (see Figure 9(b)). As the number of backup nodes increases, the NH increases, but the slope of the improvement decreases. When the LSR is 0.9 and 0.8, the NH starts to decrease after more than 40 nodes in the network. It is because the network delay increases faster than the network reliability and starts to have bad effect on NH.

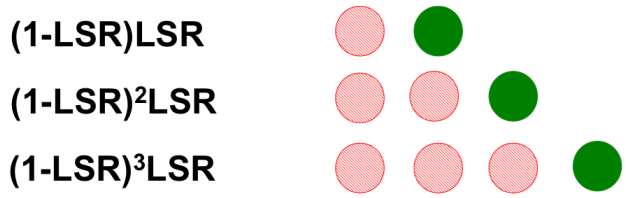


Figure 10: Illustration of the probability of a message sent from previous level is handled by the last node of a level. Red nodes do not receive messages and green nodes handle messages

4.4.2 Simulation Results

In the simulation, we adjust the RSSI value to simulate various LSR values. Figure 11 shows how to correlate RSSI and LSR based on 12,000 transmissions, where RSSI is between -64 dBm and -84dBm. As the RSSI increases, the average LSR increases, which indicates the network condition becomes better. The simulated average DR is shown in Figure 12(a) for different RSSI values as a function of the number of active nodes in the network. The DR increases when the number of active nodes increases, showing network gains in reliability. Obviously, the higher the RSSI, the higher the DR; however, the difference in DR decreases as a function of the number of active nodes and RSSI becomes irrelevant for networks with many nodes (backup nodes dominate then and network is very reliable). Simulated values of NH are shown in Figure 12(b). From this result, we can also select topologies with the minimum number of active nodes that meet $NH \geq 0$ (above the horizontal line in Figure 12(b)) for different LSR values (we can correlate the RSSI values with the LSR values from Figure 11). It is interesting to see that the NH increases at first as the number of active nodes in the network increases, because the DR increases faster than the network delay. But the NH then decreases as the number of active nodes increases, because the network delay increases faster than the DR.

We compare our model results (MR) with the simulation results (SR). Table 1 is the comparison of the minimum number of active nodes of MR (MinMR) and the minimum number of active nodes of SR (MinSR) when satisfying $NH \geq 0$ for various values of RSSI. *Diff* is defined as the percentage difference between MinMR and MinSR, $Diff = (MinMR -$

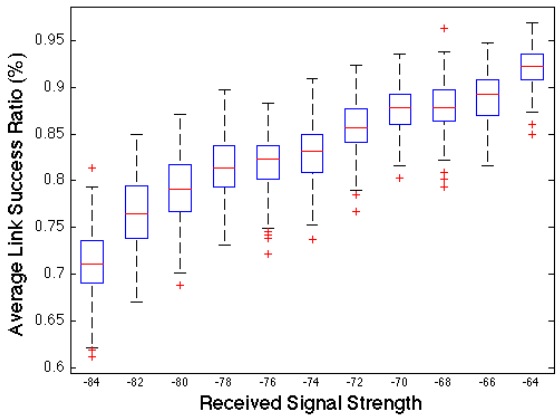
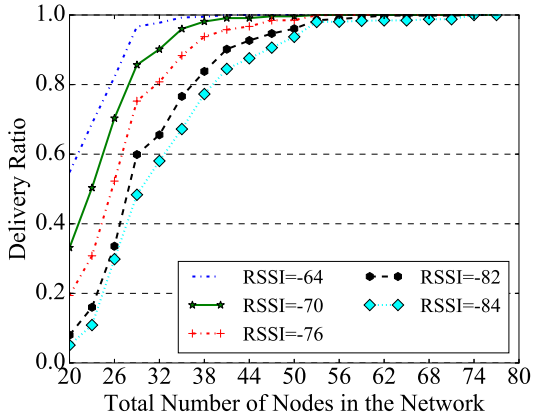
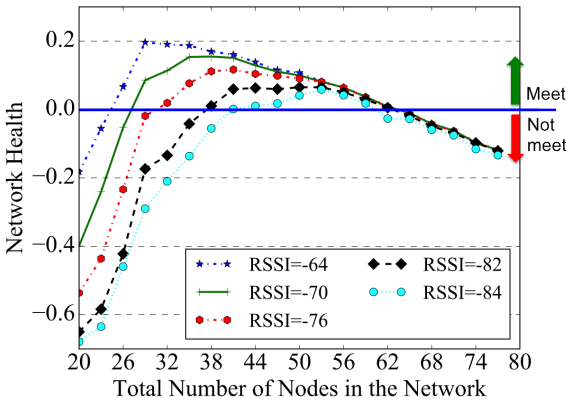


Figure 11: The relationship between RSSI and average LSR



(a) Delivery ratio for different number of active nodes.



(b) NH distribution for different number of active nodes.

Figure 12: Simulation results

MinSR)/MinMR, quantifying the difference between our model and the simulation. The simulation result demonstrates that our model is accurate with average 4.1% difference from the model result (minimum and maximum difference is 0% and 8.7%, respectively).

The MinMR is different from the MinSR due to the following reason. MR uses a constant value of LSR to do the calculation of the minimum number of active nodes in the network that would meet $NH \geq 0$, while the distribution of LSR in our simulation follows the CPM model [Lee et al., 2007] in TOSSIM. We conduct a simulation of 12,000 transmissions in a network and calculate the average LSRs every 60 transmissions. In order to see the LSR distribution clearly, we normalize the LSRs by subtracting with the average LSR over the 12,000 transmissions. Figure 13 shows the histograms of the LSR difference distribution, for $RSSI = -64$, $RSSI = -76$ and $RSSI = -84$. They all have LSRs that are different from the overall average LSR, but the more concentrated around 0, the more LSRs close to the overall average LSR, which matches the LSR standard deviation in Table 1. The LSR dispersion degree (Figure 13) is the highest for poor network ($RSSI = -84$), indicating that the LSR has the most variation and is the most different from the constant LSR of our model, which explains $RSSI = -84$ shows the highest *Diff*.

Table 1: Comparison of model and simulation results

RSSI (dBm)	average LSR	LSR stdv	MinMR	MinSR	Diff
-64	0.93	0.020	26	26	0%
-70	0.88	0.024	29	30	-3.4%
-76	0.82	0.031	33	32	3.0%
-82	0.77	0.035	37	39	-5.4%
-84	0.71	0.037	46	42	8.7%

4.5 Summary

In this chapter, we focused on designing a fault-tolerant wireless network to meet control system requirement with the minimum number of active nodes. We first propose a fault-

tolerant network node placement design. We then present our model to calculate the control system stability requirement, NH, and determine the minimum number of active nodes in the network to meet $NH \geq 0$. For validation, we simulate a 12-hop wireless network on TOSSIM simulator, transmitting messages for a PHX in an NPP. The simulation result demonstrates the correctness of our model with average 4.1% difference from the model result. Furthermore, we find that the redundancy in the wireless network is not always good for the control system stability, which could induce more network delay in the WCS.

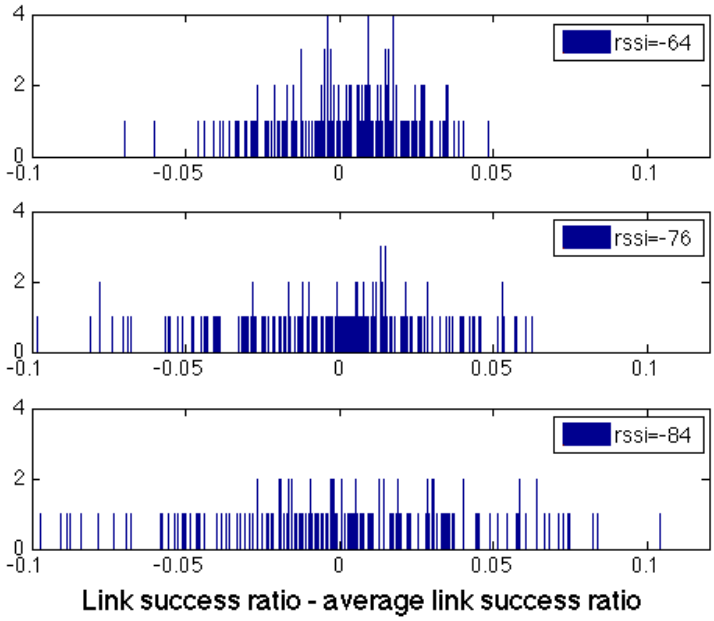


Figure 13: Histogram of LSR difference distribution for $RSSI = -64$, $RSSI = -76$ and $RSSI = -84$

5.0 Worst-case End-to-end Delay Analysis

End-to-end network delay is critical to control system stability in WCS according to our control system stability requirement (Equation (4.1)) and recent research works [Yu et al., 2014; Saifullah et al., 2011, 2015], which require network packets transmitting within a certain control system deadline to make the control system stable. Thus, it is necessary to quantify the worst-case end-to-end network delay, which can be used to test, both at design time and for online admission control, whether a set of real-time transmissions can meet all their deadlines. Compared to extensive testing and simulations, analytical delay bounds are highly desirable in wireless control system applications that require real-time performance guarantees [Saifullah et al., 2011]. As mentioned in Chapter 1 and Chapter 4, there is no transmission conflict during the one-way wireless transmission in WCS if the messages are schedulable. We quantify the associated worst-case end-to-end delay in Equation (4.5). However, for the two-way wireless transmission, the messages going up will conflict with the messages going down, which induce more delay than the one-way wireless transmission. In order to scale our model of control system stability requirement estimation (in terms of network delay and delivery ratio estimation [1]) in Chapter 1 to two-way wireless communication, this chapter focuses on analyzing the worst-case end-to-end delay for the two-way wireless transmission in a general WCS.

In this chapter, we first introduce the network model we study. We then analyze the conflicts that could happen during the message transmission and get the schedulability condition (the condition that messages can be delivered to the destination within the bounded amount of time) from the analysis. Based on the schedualability condition, we then compute the worst-case end-to-end delay by calculating the delay without conflict, the maximum number of conflicts during one message transmission and maximum time to resolve the conflicts. To the best of our knowledge, this is the first work that discusses the end-to-end delay analysis

[1] The procedure of delivery ratio estimation in two-way wireless communication is the same as the procedure in Chapter 4.3.1. Although the algorithm complexity increase for two-way communication, it is acceptable since our analysis is offline.

for the network deadline greater than the control sampling period in the real-time WCS with traffic in both directions.

5.1 Network Model

In this chapter, the network model we study is the relay region of the network we proposed in Chapter 4 (for the network topology please see Figure 7). Our network model is shown in Figure 14: there is one primary line of relay nodes (marked as black) and zero or more lines of backup relay nodes (marked as grey). Different from the relay region in Chapter 4, each line of nodes in our network model are complete. The relay nodes broadcast messages level by level towards the controller, then back to the actuator. Within each level, the primary node will broadcast first, then the first, second, and third backup nodes, in order. Therefore, the more relay nodes in the network, the more messages are sent, and thus the higher network delay. Note that the node radio frequency in this chapter is the same for all nodes (i.e., no multi-channel transmission). Every control sampling period, we assume there is one message containing all the measurement data sent out via the wireless network to the remote controller, which runs the control algorithm and then sends a message back via the same network again to the actuator. The worst-case end-to-end delay analysis is the worst-case time delay of any one message from the time it is sent out to the time it is received by the actuator. We also assume there is no measurement concatenation for measurements sensed from different time steps.

We assume that there are n hops from the sensors to the remote controller and l lines of relay nodes, that is, it takes l time slots at each level to transmit messages (one slot per node). To be reliable, the controller will send out l duplicate messages to the relay nodes (i.e. takes l time slots). We denote the current time slot as t ($t = 0, 1, 2, ...$), the current level as h ($h = 0, 1, ..., n$), and control sampling period as p. The number of time slots during one sampling period is $p_s = \frac{p}{\Delta t}$, where Δt is the duration of the time slot. Thus, with the time delay (i.e., stall time) caused by conflicting with other messages, d_0 (in terms of the number of time slots), message m_0 sent at time $t = 0$ up to the controller is at level

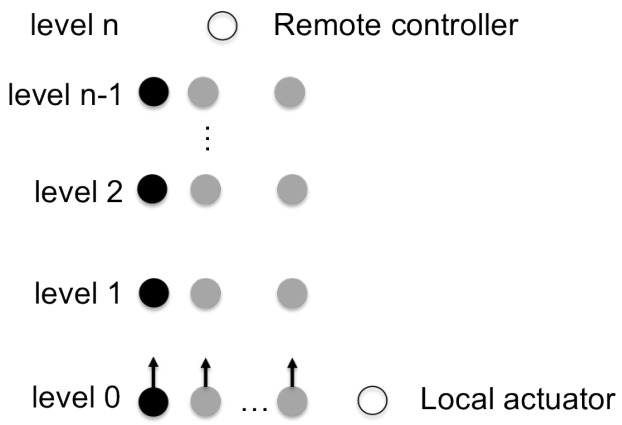


Figure 14: Network model with one or more lines of relay nodes

$h(m_0) = \lfloor \frac{t-d_0}{l} \rfloor$ $(0 \leq \lfloor \frac{t-d_0}{l} \rfloor < n)$ and the same message on its way down to the actuator is at level $h(m_0) = 2n - \lfloor \frac{t-d_0}{l} \rfloor$ $(n \leq \lfloor \frac{t-d_0}{l} \rfloor \leq 2n-1)$. More generally, with the time delay caused by conflicting with other messages, d_i, a message m_i sent out at time $t = ip_s$, $(i = 0, 1, ...)$ traveling up is at level $h(m_i) = \lfloor \frac{t-d_i-ip_s}{l} \rfloor$ $(0 \leq \lfloor \frac{t-d_i-ip_s}{l} \rfloor < n)$ and traveling down is at level $h(m_i) = 2n - \lfloor \frac{t-d_i-ip_s}{l} \rfloor$ $(n \leq \lfloor \frac{t-d_i-ip_s}{l} \rfloor \leq 2n-1)$. We also use $t_c(m_i, m_j)$ to denote the time message m_i starts conflicting with m_j.

5.2 Conflict Analysis

We want to determine the worst-case end-to-end delay for periodic messages in a general case, when the network delay is greater than the control sampling period (as discussed in Chapter 1). We focus on the delay analysis for fixed priority scheduling where message transmissions are scheduled based on the most recent message first and the oldest message first schemes. We only do our proof based on the most recent message first scheme, given that the derivation for the oldest message first scheme first is symmetric, which will be discussed in Section 5.4. We denote the priority of a message m_i as $pri(m_i)$. The delay without conflicts for transmitting one message up to the remote controller is nl and the same amount

of delay for going down. Thus, the delay without conflict is $2nl$. When $2nl \leq p_s$, there will be no conflcts, given that the messages will go up and down before the next message is sent out. When $2nl > p_s$, the current message m_i will conflict with the message m_j with higher priority ($pri(m_i) < pri(m_j)$) and induce more network delay. In this section, we will do the conflict analysis of the message transmission.

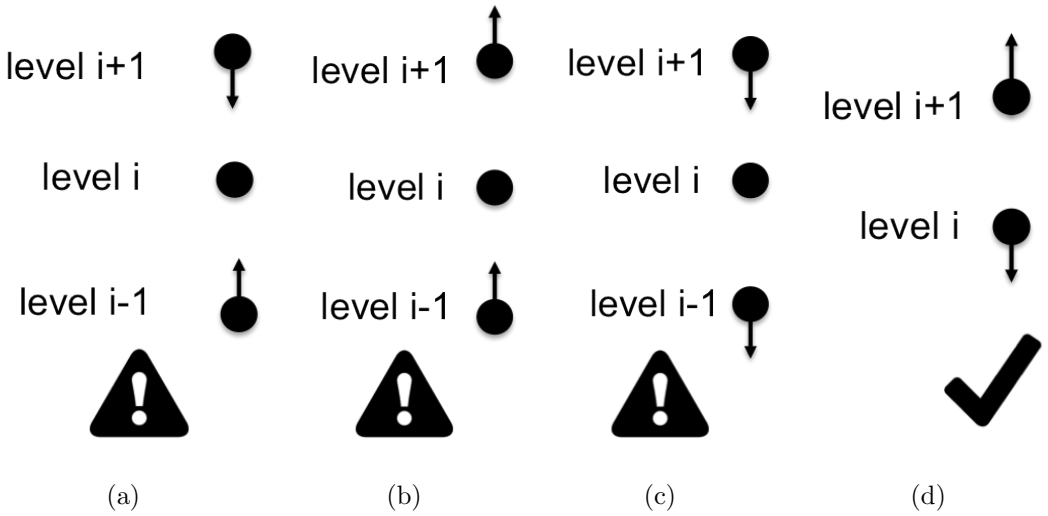


Figure 15: Conflict situation (a) (b) (c) and no conflict situation (d)

A node cannot both transmit and receive in the same time slot and two transmissions that have the same intended receiver interfere each other. If two transmissions are conflicting, they cannot be scheduled in the same slot, which induces more time delay to the lower-priority transmission. Given a set of n, l and p_s, where $2nl > p_s$, there are three canonical situations that two messages will conflict with each other. As usual in wireless networks, conflicts arise when simultaneous transmissions arrive at the same node. The three scenarios are shown as conflict situations 1, 2 and 3 in Figure 15(a), 15(b) and 15(c), respectively, for a single line of relay nodes (no backups). Conflict situation 1 shows the scenario when a message going up is at a lower level than the other message going down. Conflict situation 2 and 3 show the scenarios when two messages are going in the same direction but very close together. But for the situation shown in 15(d), when the message going up is at a higher level than the message going down, there is no conflict (even though the level difference is 1), since their

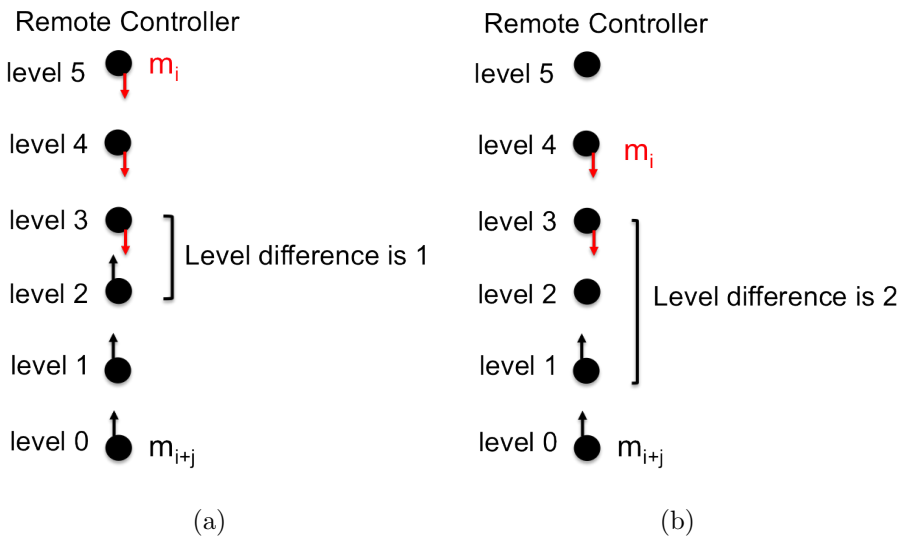


Figure 16: The conflicts of m_i when the level difference with m_{i+j} is 5 (a) and 4 (b)

receivers are separated apart. Thus, the two messages start to conflict at the level difference, Δh is 1 or 2, only under the three conflict situations. When the $\Delta h \geq 3$, the two messages will not conflict with each other, since it is not possible that one receiver listening to the messages coming from more than one transmitters at the same time. For conflict situation 1, when the $\Delta h = 1$ and the level of two messages is less or equal to $n-2$, it will take $2l$ time slots to resolve the conflict, given that the high-priority message will go up two levels while the low-priority message waits. At this time the conflict is resolved. Similarly, when the $\Delta h = 2$ and the level of two messages is less or equal to $n-2$, the conflict will be resolved in $3l$ time slots. In general, when message m_i starts going down, the level difference between m_i and m_{i+j}, $\Delta h(m_i, m_{i+j})$ can be odd or even. When $\Delta h(m_i, m_{i+j}) = |h(m_i) - h(m_{i+j})|$ is odd (as shown in Figure 16(a)), each of the two messages will make progress on one level at a time, until they are separated by exactly 1 level, at which time the conflict happens and will be resolved in $2l$ time slots. Similarly, when $\Delta h(m_i, m_{i+j})$ is even (as shown in Figure 16(b)), they will make progress until they are separated by exactly 2 levels, at which time the conflict happens and will be resolved in $3l$ time slots. For conflict situations 2 and 3,

it will take $4l$ or $5l$ time slots to resolve the conflict, when the level difference is 1 or 2, respectively.

Let us consider consecutive messages, m_0, m_1, m_2, ..., m_i that are sent at $t = 0$, $t = p_s$, $t = 2p_s$, ..., $t = ip_s$, respectively. Since we apply the most recent message first message priority scheduling scheme, where $pri(m_0) < pri(m_1) < ... < pri(m_i)$. We define level separation of two messages m_i and m_{i+j}, $ls(m_i, m_{i+j})$ as the number of levels m_{i+j} needs to go through to be at the same level and in the same direction of m_i if m_i stays still. The level separation of two consecutive messages before any conflicts is $ls(m_i, m_{i+1}) = \lfloor \frac{p_s}{l} \rfloor$, which describes how separated the two consecutive messages are (in other words, how long in terms of levels to wait to transmit the next sensor value). Intuitively, the more $\lfloor \frac{p_s}{l} \rfloor$, the fewer conflicts will happen. Note that level separation is different from level difference of two messages, when two messages go in opposite directions (e.g., if there are 5 hops in the network and m_i is going down at level 4 and m_j is going up at level 0, $\Delta h(m_i, m_j) = 4$ and $ls(m_i, m_j) = 6$). Recall that if $p_s \leq 2nl$, there will be no conflicts, given that the message will go up and down before the next message is sent out. Thus, three cases under the condition of $2nl > p_s$ are discussed in the following sections: (1) $\lfloor \frac{p_s}{l} \rfloor \leq 2$, (2) $3 \leq \lfloor \frac{p_s}{l} \rfloor \leq 4$ and (3) $\lfloor \frac{p_s}{l} \rfloor \geq 5$.

5.2.1 Conflict Analysis for Case $\lfloor \frac{p_s}{l} \rfloor \leq 2$

Lemma 5.2.1. *When $\lfloor \frac{p_s}{l} \rfloor \leq 2$, no message can be delivered to the destination.*

Proof. For the base case of m_0 and m_1, when both m_0 and m_1 go up, their levels are $h(m_0) = \lfloor \frac{t}{l} \rfloor$ and $h(m_1) = \lfloor \frac{t-p_s}{l} \rfloor$ ($\lfloor \frac{t}{l} \rfloor < n$), respectively. The level difference of m_0 and m_1 is $\Delta h(m_0, m_1) = \lfloor \frac{p_s}{l} \rfloor \leq 2$. Conflict situation 2 happens, since m_0 and m_1 are separated by less than 3 levels. At time $t = p_s, h(m_0) = \lfloor \frac{p_s}{l} \rfloor$, m_1 is sent out, and m_0 needs to wait until m_1 is at level $h(m_1) = \lfloor \frac{p_s}{l} \rfloor + 3$ at time $t = p_s + 3l$. However, at time $t = 2p_s < p_s + 3l$ (i.e., before the conflict of m_0 and m_1 is resolved), m_2 will be transmitted and also block m_1. Since the conflict of m_0 and m_1 cannot be resolved, m_0 will never move past level $\lfloor \frac{p_s}{l} \rfloor$.

In general, the situation of any two consecutive messages m_i and m_{i+1} is similar to the situation of m_0 and m_1, where at time $t = (i + 1)p_s$, m_{i+1} will start transmission and

interrupt m_i at level $\lfloor \frac{p_s}{l} \rfloor$, creating a chain reaction. Therefore, all messages will be blocked by messages with higher priority and no message can be delivered to the destination. Since all messages are blocked at level $\lfloor \frac{p_s}{l} \rfloor$ when going up, we do not need to consider conflicts situation 1 and 3 because they will never occur if $\lfloor \frac{p_s}{l} \rfloor \leq 2$. □

5.2.2 Conflict Analysis for Case $3 \leq \lfloor \frac{p_s}{l} \rfloor \leq 4$

Lemma 5.2.2. *When $3 \leq \lfloor \frac{p_s}{l} \rfloor \leq 4$, no message can be delivered to the destination.*

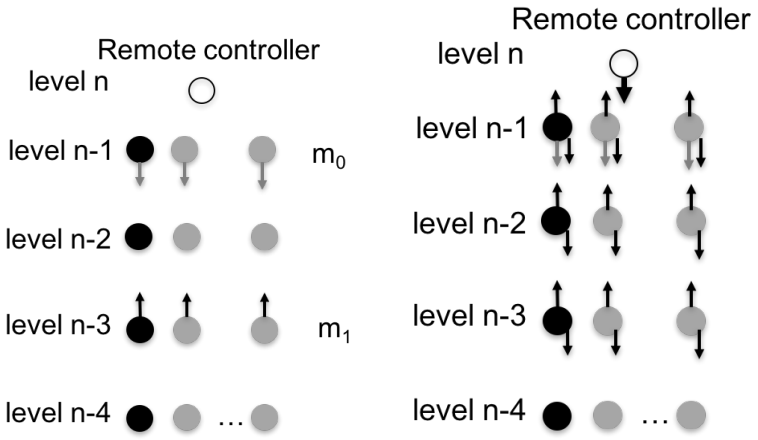


Figure 17: Conflict situation when $\lfloor \frac{p_s}{l} \rfloor = 4$: (a) m_0 starts conflicting with m_1 and (b) the conflict is resolved in $7l$ time slots if the subsequent messages do not exist

Proof. Let us first consider the best case (the largest separation between two consecutive messages): $\lfloor \frac{p_s}{l} \rfloor = 4$.

For the base case, when both m_0 and m_1 go up ($\lfloor \frac{t}{l} \rfloor < n$), $\Delta h(m_0, m_1) = \lfloor \frac{p_s}{l} \rfloor \geq 3$ with no conflict. When m_0 is already going down ($\lfloor \frac{t}{l} \rfloor \geq n$) and m_1 is still going up ($\lfloor \frac{t-p_s}{l} \rfloor < n$), $\Delta h(m_0, m_1) = 2n - \lfloor \frac{t}{l} \rfloor - \lfloor \frac{t-p_s}{l} \rfloor \leq 2n - 2\lfloor \frac{t}{l} \rfloor + \lfloor \frac{p_s}{l} \rfloor \leq \lfloor \frac{p_s}{l} \rfloor = 4$. Let us consider the best case (the largest separation of m_0 and m_1) with $\Delta h(m_0, m_1) = 4$. As shown in Figure 17(a), the conflict happens when $h(m_0) = n - 1$ on the way down (grey arrow represents m_0) and

$h(m_1) = n - 3$ on the way up (black arrow represents m_1). As shown in Figure 17(b), the conflict involves conflict situations 1 and 3: (1) during the conflict situation 1, m_0 is blocked by m_1, while m_1 goes up to the remote controller; (2) when m_1 reaches remote controller, the conflict becomes conflict situation 3 and is resolved until m_1 reaches level $n - 3$. So the conflict is resolved in $7l$ time slots if m_2 and the following messages do not exist. However, after $4l$ slots of the conflict of m_0 and m_1, where m_1 is on the way down at level $n - 1$, m_1 will conflict with m_2 (like the situation in Figure 17(a)) and the previous conflict of m_0 and m_1 will never be resolved. m_0 will be blocked at level $n - 1$ forever.

For general case of m_i and m_{i+1}, when m_i goes down, $h(m_i) = 2n - \lfloor \frac{t-ip_s}{l} \rfloor$ ($\lfloor \frac{t-ip_s}{l} \rfloor \geq n$); and when m_{i+1} goes up, $h(m_{i+1}) = \left\lfloor \frac{t-(i+1)p_s}{l} \right\rfloor$ ($\left\lfloor \frac{t-(i+1)p_s}{l} \right\rfloor < n$). Since $\Delta h(m_i, m_{i+1}) = 2n - \lfloor \frac{t-ip_s}{l} \rfloor - \left\lfloor \frac{t-(i+1)p_s}{l} \right\rfloor \leq 2n - 2\lfloor \frac{t-ip_s}{l} \rfloor + \lfloor \frac{p_s}{l} \rfloor \leq 4$, with the best case of the largest level difference of 4, m_i will conflict with m_{i+1} as the same situation of m_0 and m_1 above. After $4l$ of the conflict of m_i and m_{i+1} (the conflict takes $7l$ to resolve), m_{i+1} conflicts with m_{i+2}, and the conflict of m_i and m_{i+1} cannot be resolved. Therefore, all the messages will be blocked by higher priority messages at level $n - 1$ with $\lfloor \frac{p_s}{l} \rfloor = 4$.

Clearly, if the best case of $\lfloor \frac{p_s}{l} \rfloor = 4$ causes indefinite blocking, the case of $\lfloor \frac{p_s}{l} \rfloor = 3$ will come to the same conclusion.

□

5.2.3 Conflict Analysis for Case $\lfloor \frac{p_s}{l} \rfloor \geq 5$

We consider two cases for $\lfloor \frac{p_s}{l} \rfloor \geq 5$: the case of odd value of $\lfloor \frac{p_s}{l} \rfloor$ in Lemma 5.2.3 and the case of even value of $\lfloor \frac{p_s}{l} \rfloor$ in Lemma 5.2.4.

Lemma 5.2.3. *When $\lfloor \frac{p_s}{l} \rfloor \geq 5$, all messages will be delivered, if $\lfloor \frac{p_s}{l} \rfloor$ is odd.*

Proof. We prove this Lemma by showing that it is true for the worst case (smallest separation of two consecutive messages) when $\lfloor \frac{p_s}{l} \rfloor$ is odd, that is, $\frac{p_s}{l} = 5$. We show the Lemma is true for the base case of m_0 and m_1, and then generalize to any two consecutive messages, m_i and m_{i+1}. There are three cases:

(1) When both m_0 and m_1 go up ($\lfloor \frac{t}{l} \rfloor < n$), $\Delta h(m_0, m_1) = \lfloor \frac{t}{l} \rfloor - \lfloor \frac{t-p_s}{l} \rfloor = \frac{p_s}{l} = 5 \geq 3$, there is no conflict.

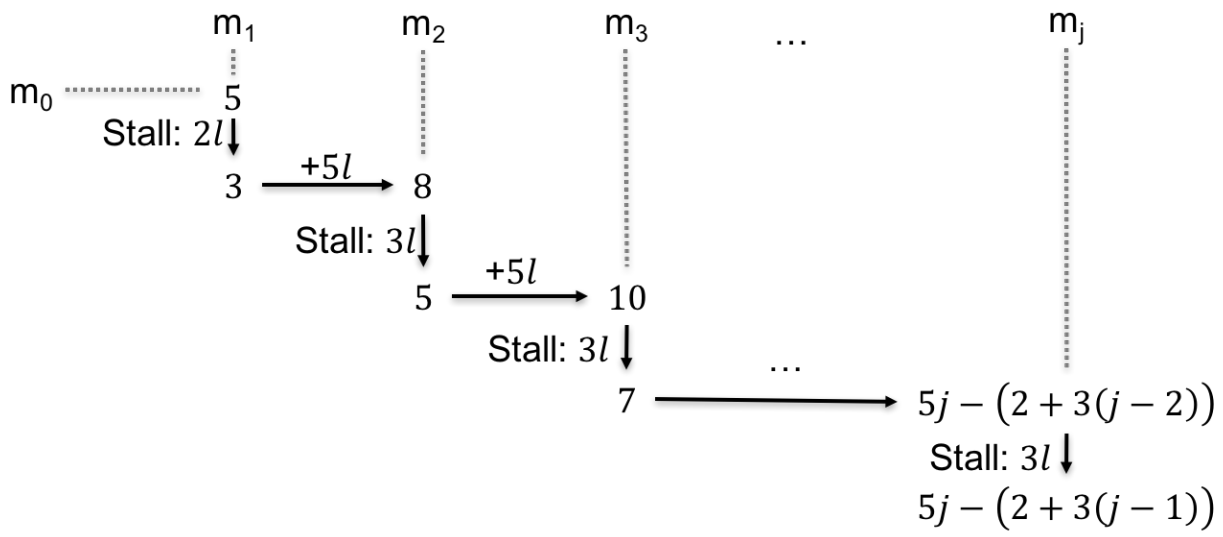


Figure 18: The calculation process of level separations with higher priority messages of m_0 and m_1, when $\frac{p_s}{l} = 5$

(2) When m_0 goes down ($\lfloor \frac{t}{l} \rfloor \geq n$) and m_1 goes up ($\lfloor \frac{t-p_s}{l} \rfloor < n$), $\Delta h(m_0, m_1) = 2n - \lfloor \frac{t}{l} \rfloor - \lfloor \frac{t-p_s}{l} \rfloor = 2n - 2\lfloor \frac{t}{l} \rfloor + \frac{p_s}{l}$. The conflict only involves the conflict situation 1. Since we are dealing with the case of $\frac{p_s}{l} = 5$, which means level separation is odd, so is $\Delta h(m_0, m_1)$, the conflict happens with $\Delta h(m_0, m_1) = 1$ and can be resolved in $2l$ time slots. By solving $\Delta h(m_0, m_1) = 2n - 2\lfloor \frac{t}{l} \rfloor + \frac{p_s}{l} = 1$, we get $\lfloor \frac{t}{l} \rfloor = n + 2$. After this conflict, m_0 stays at the same level as the conflict before (stalled), $h(m_0) = 2n - \lfloor \frac{t}{l} \rfloor = n - 2$; m_1 goes up 2 levels and $h(m_1) = \lfloor \frac{t-p_s}{l} \rfloor + 2 = n - 1$. Although the level difference is 1, m_0 and m_1 are in the situation shown in Figure 15(d), there is no more conflict between m_0 and m_1. Figure 18 shows the level separation of m_0 and m_1 is 5 to start with (before conflict), going down to 3, after the conflict (because m_1 advances 2 levels while m_0 stalls).

(3) When both m_0 and m_1 go down, m_0 and m_1 will conflict with higher priority messages, m_2, m_3, ... m_j. These conflicts involve the conflict situation 1, given that m_2, m_3, ..., m_j are going up. For both m_0 and m_1, only the first conflict starts with an odd level separation (for m_0 see case (2) above) and the rest of conflicts are all even. Therefore, as shown in Figure 18, conflicts after the first conflict are resolved in $3l$ time slots. A similar process can

Table 2: The total stalls of m_0 and m_1 (i.e., d_0 and d_1) when m_0 and m_1 conflict with higher priority messages ($\frac{p_s}{l} = 5$)

	m_1	m_2	m_3	...	m_j
m_0	$2l$	$(2+3)l$	$(2+2*3)l$		$2l+3(j-1)l$
m_1	-	$2l$	$(2+3)l$		$2l+3(j-2)l$

be followed for m_1. Table 2 shows the total stalls in terms of the number of time slots when m_0 and m_1 conflict with m_2, m_3, ..., m_j under the condition $\frac{p_s}{l} = 5$.

In addition to conflict situation 1, we also need to consider conflict situation 3, given that when both m_0 and m_1 go down and m_0 is ahead of m_1, m_0 will stall first given the conflict, causing m_1 to approach m_0, further causing situation 3 conflict. Below, we discuss three subcases to show how these conflicts are resolved: (3A) m_0 and m_1 conflicting with m_2, (3B) m_0 and m_1 conflicting with m_3 and (3C) m_0 and m_1 conflicting with m_j ($j \geq 2$).

Case 3A: m_0 and m_1 conflict with m_2. During the conflict of m_0 with m_2, m_0 will go down 1 level, and during the conflict of m_1 with m_2, m_1 will go down 2 levels, as follows. The level of m_0, m_1 and m_2 is $h(m_0) = 2n - \lfloor \frac{t-2l}{l} \rfloor$ (as shown in Table 2, $d_0 = 2l$ due to the conflict with m_1), $h(m_1) = 2n - \lfloor \frac{t-p_s}{l} \rfloor$ and $h(m_2) = \lfloor \frac{t-2p_s}{l} \rfloor$, respectively. When m_0 starts conflicting with m_2, $\Delta h(m_0, m_2) = 2n - \lfloor \frac{t-2l}{l} \rfloor - \lfloor \frac{t-2p_s}{l} \rfloor = 2$, and we get $\lfloor \frac{t}{l} \rfloor = n + \frac{p_s}{l}$, so $t_c(m_0, m_2) = nl + p_s$ (as mentioned earlier, it is the time m_0 and m_2 starts conflicting) and $h(m_0) = 2n - \lfloor \frac{t-2l}{l} \rfloor = 2n - \lfloor \frac{t}{l} \rfloor + 2 = n - \frac{p_s}{l} + 2$. When m_1 starts conflicting with m_2, $\Delta h(m_1, m_2) = 2n - \lfloor \frac{t-p_s}{l} \rfloor - \lfloor \frac{t-2p_s}{l} \rfloor = 1$, and we get $\lfloor \frac{t-p_s}{l} \rfloor = n + \frac{1}{2}\frac{p_s}{l} - \frac{1}{2}$, so $t_c(m_1, m_2) = nl + \frac{3}{2}p_s - \frac{1}{2}l$ and $h(m_1) = 2n - \lfloor \frac{t-p_s}{l} \rfloor = n - \frac{1}{2}\frac{p_s}{l} + \frac{1}{2}$. Given that $\Delta h(m_1, m_0) = n - \frac{1}{2}\frac{p_s}{l} + \frac{1}{2} - (n - \frac{p_s}{l} + 2) = \frac{1}{2}\frac{p_s}{l} - \frac{3}{2} = 1 < 3$ (i.e., the level difference between m_0 and m_1 when m_0 and m_1 start conflicting with m_2), m_0 and m_1 will conflict again with each other (this time under conflict situation 3).

To explain how long m_0 gets stalled before m_1 starts its conflict with m_2, we turn to Figure 19, which shows the stall time of m_0 from I_0 to I_2 and m_1 from I_2 to I_3. The length of

I_0, I_1, I_2 and I_3 is l, the time to transmit a message for one level. Since m_0 stalls for $3l$ and $t_c(m_1, m_2) - t_c(m_0, m_2) = \frac{1}{2}p_s - \frac{1}{2}l = 2l$, the overlap of m_0 and m_2 is l, that is, I_2. During I_0 to I_1, m_0 conflicts with m_2 (and stalls), while m_1 keeps going down 2 levels and m_2 goes up 2 levels. During I_2, both m_0 and m_1 conflict with m_2 and only m_2 (highest priority) goes up 1 level. During I_3, m_1 conflicts with m_2, allowing m_0 to go down 1 level and m_2 to go up 1 level.

m_0 and m_1 will not conflict with m_3, since during I_3 ($\lfloor \frac{t}{l} \rfloor = n + \frac{p_s}{l} + 3$), the level of m_0, m_1 and m_3 is $h(m_0) = n - \frac{p_s}{l} + 2$, $h(m_1) = n - \frac{1}{2}\frac{p_s}{l} + \frac{1}{2}$ and $h(m_3) = \lfloor \frac{t-3p_s}{l} \rfloor = \lfloor \frac{t}{l} \rfloor - \frac{3p_s}{l} = n - \frac{2p_s}{l} + 3$, respectively, with $\Delta h(m_0, m_3) = 4$ and $\Delta h(m_1, m_3) = 5$ both greater than 3. From I_0 to I_2, the level of m_0 and m_1 are both higher than their levels during I_3 and the level of m_3 is lower than its level of I_3. Since there is no conflict with m_3 during I_3, there is no conflict from I_0 to I_2. Thus, m_0 and m_1 will not conflict with m_3 and will not conflict with other messages (i.e., higher priority messages of m_3) either during I_0 to I_3.

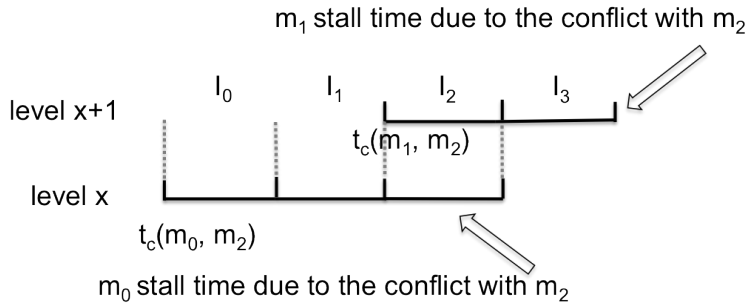


Figure 19: The stall time for m_0 (lower segments) and m_1 (upper segments), when conflicting with m_2

Case 3B: m_0 and m_1 conflict with m_3. m_0 and m_1 will not be completely blocked during the conflicts with m_3: m_0 and m_1 will both go down for 1 level. The level of m_0, m_1 and m_3 is $h(m_0) = 2n - \lfloor \frac{t-5l}{l} \rfloor$ (as shown in Table 2, $d_0 = 5l$ due to the conflicts with m_1 and m_2), $h(m_1) = 2n - \lfloor \frac{t-p_s-2l}{l} \rfloor$ (as shown in Table 2, $d_1 = 2l$ due to the conflicts with m_2) and $h(m_3) = \lfloor \frac{t-3p_s}{l} \rfloor$, respectively. When m_0 starts conflicting with m_3, $\Delta h(m_0, m_3) = 2n - \lfloor \frac{t-5l}{l} \rfloor - \lfloor \frac{t-3p_s}{l} \rfloor = 2$, and we get $\lfloor \frac{t}{l} \rfloor = n + \frac{3}{2}\frac{p_s}{l} + \frac{3}{2}$, so $t_c(m_0, m_3) = nl + \frac{3}{2}p_s + \frac{3}{2}l$ and $h(m_0) = 2n - \lfloor \frac{t-5l}{l} \rfloor = 2n - \lfloor \frac{t}{l} \rfloor + 5 = n - \frac{3}{2}\frac{p_s}{l} + \frac{7}{2}$. When m_1 starts conflicting

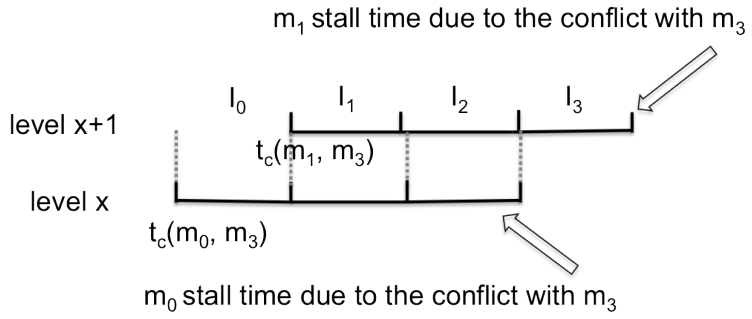


Figure 20: The stall time for m_0 (lower segments) and m_1 (upper segments), when conflicting with m_3

with m_3, $\Delta h(m_1, m_3) = 2n - \left\lfloor \frac{t-p_s-2l}{l} \right\rfloor - \left\lfloor \frac{t-3p_s}{l} \right\rfloor = 2$, and we get $\left\lfloor \frac{t-p_s}{l} \right\rfloor = n + \left\lfloor \frac{p_s}{l} \right\rfloor$, so $t_c(m_1, m_3) = nl + 2p_s$ and $h(m_1) = 2n - \left\lfloor \frac{t-p_s-2l}{l} \right\rfloor = 2n - \left\lfloor \frac{t-p_s}{l} \right\rfloor + 2 = n - \frac{p_s}{l} + 2$. Thus, $\Delta h(m_1, m_0) = \frac{1}{2}\frac{p_s}{l} - \frac{3}{2} = 1 > 3$, which means m_0 and m_1 have conflict (conflict situation 3). The start conflicting time difference is $t_c(m_1, m_3) - t_c(m_0, m_3) = \frac{1}{2}p_s - \frac{3}{2}l = l$. Figure 20 illustrates the stall intervals for m_0 conflicting with m_1 and m_3. During I_0, m_0 conflicts with m_1 and m_3, allowing both m_1 and m_3 to go down and up for 1 level, respectively. During I_1 to I_2, m_0 conflicts with m_1 and m_3, and m_1 conflicts with m_0 and m_3, allowing only m_3 to go up 2 levels. During I_3, m_1 conflicts with m_0 and m_3, allowing m_0 to go down for 1 level and m_3 to go up 1 level. Even though m_0 and m_1 conflict, each gets a chance to move further by 1 level when the other one is stalled with m_3.

Similar to case 3A, m_4 cannot conflict with m_0 and m_1 during the conflict from I_0 to I_3. Since during I_3 ($\left\lfloor \frac{t}{l} \right\rfloor = n + \frac{3}{2}\frac{p_s}{l} + \frac{9}{2}$), the level of m_0, m_1 and m_4 is $h(m_0) = n - \frac{3}{2}\frac{p_s}{l} + \frac{7}{2}$, $h(m_1) = n - \frac{p_s}{l} + 2$ and $h(m_4) = n - \frac{5}{2}\frac{p_s}{l} + \frac{9}{2}$, respectively, with $\Delta h(m_0, m_4) = 4$ and $\Delta h(m_1, m_4) = 5$ both greater than 3, m_4 will not conflict with m_0 and m_1 during I_3. Therefore, m_4 will not conflict with any messages from I_0 to I_3 and thus no conflict of m_1, m_2 with other messages (i.e., the higher priority messages of m_4) also.

Case 3C: m_0 and m_1 conflict with m_j ($j \geq 2$). m_0 and m_1 will not be completely blocked during the conflict and can both go down 1 level. The level of m_0, m_1 and m_j is $h(m_0) = 2n - \left\lfloor \frac{t-(2+3(j-2))l}{l} \right\rfloor$ (as shown in Table 2, $d_0 = (2+3(j-2))l$ due to the conflicts with

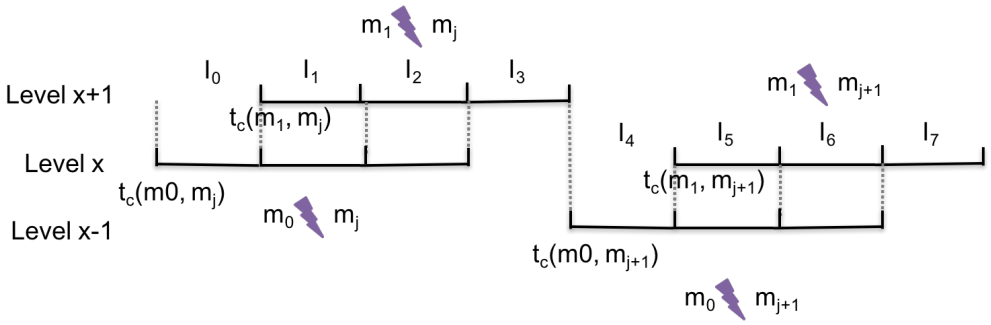


Figure 21: The stall time for m_0 (lower segments) and m_1 (upper segments), when conflicting with m_j and m_{j+1}

m_0, m_1, ..., m_{j-1}), $h(m_1) = 2n - \left\lfloor \frac{t-p_s-(2+3(j-3))l}{l} \right\rfloor$ (as shown in Table 2, $d_1 = (2+3(j-3))l$ due to the conflicts with m_1, ..., m_{j-1}) and $h(m_j) = \lfloor \frac{t-jp_s}{l} \rfloor$, respectively. In general, when m_0 starts conflicting with m_j, $\Delta h(m_0, m_j) = 2n - \left\lfloor \frac{t-(2+3(j-2))l}{l} \right\rfloor - \lfloor \frac{t-jp_s}{l} \rfloor = 2$, and we get $\lfloor \frac{t}{l} \rfloor = n + \frac{3}{2}(j-2) + \frac{j}{2}\frac{p_s}{l}$, so $t_c(m_0, m_j) = nl + \frac{3}{2}(j-2)l + \frac{j}{2}p_s$ and $h(m_0) = 2n - \left\lfloor \frac{t-(2+3(j-2))l}{l} \right\rfloor = 2n - \lfloor \frac{t}{l} \rfloor + (2+3(j-2)) = n - \frac{j}{2}\frac{p_s}{l} + \frac{3}{2}(j-2) + 2$. When m_1 starts conflicting with m_j, $\Delta h(m_1, m_j) = 2n - \left\lfloor \frac{t-p_s-(2+3(j-3))l}{l} \right\rfloor - \lfloor \frac{t-jp}{l} \rfloor = 2$, and we get $\lfloor \frac{t-p_s}{l} \rfloor = n + \frac{3}{2}(j-3) + \frac{1}{2}\left\lfloor \frac{(j-1)p_s}{l} \right\rfloor$, so $t_c(m_1, m_j) = nl + \frac{3}{2}(j-3)l + \frac{j}{2}p_s + \frac{1}{2}p_s$ and $h(m_1) = 2n - \left\lfloor \frac{t-p_s-(2+3(j-3))l}{l} \right\rfloor = 2n - \lfloor \frac{t-p_s}{l} \rfloor + (2+3(j-3)) = n - \frac{1}{2}(j-1)\frac{p_s}{l} + 2 + \frac{3}{2}(j-3)$. Thus, $\Delta h(m_1, m_0) = \frac{1}{2}\frac{p_s}{l} - \frac{3}{2} = 1$ and m_0 and m_1 are still conflicting with each other. The start conflict time difference is $t_c(m_1, m_j) - t_c(m_0, m_j) = \frac{1}{2}p_s - \frac{3}{2}l = l$. The stall time for both m_0 and m_1 is the same as Figure 20: during the conflict, m_1 can go down 1 level during I_0; and m_0 can go down 1 level during I_3. Also, m_{j+1} will not conflict with conflict with m_0 and m_1 from I_0 to I_3. Since during I_3 ($\lfloor \frac{t}{l} \rfloor = n + \frac{3}{2}j + \frac{j}{2}\frac{p_s}{l}$), the level of m_0, m_1 and m_4 is $h(m_0) = n - \frac{j}{2}\frac{p_s}{l} + \frac{3}{2}(j-2) + 2$, $h(m_1) = n - \frac{1}{2}(j-1)\frac{p_s}{l} + 2 + \frac{3}{2}(j-3)$ and $h(m_{j+1}) = n + \frac{3}{2}j - (\frac{j}{2}+1)\frac{p_s}{l}$, respectively. With $\Delta h(m_0, m_{j+1}) = 4$ and $\Delta h(m_1, m_{j+1}) = 5$, m_{j+1} and other higher priority messages will not conflict with m_0 and m_1 from I_0 to I_3. This pattern will repeat itself indefinitely in the worst case.

No delay caused by the conflict of m_0 and m_1 for Case 3A, 3B and 3C According to the Case 3A, Case 3B and Case 3C, m_0 and m_1 always conflict with each other. However,

the conflict does not induce more delay is because the duration between the start time of the conflict of m_0 with m_j ($j \geq 2$) and the start time of the conflict of m_0 with m_{j+1} is $t_c(m_0, m_{j+1}) - t_c(m_0, m_j) = nl + \frac{3}{2}(j-1)l + \frac{j+1}{2}p_s - (nl + \frac{3}{2}(j-2)l + \frac{j}{2}p_s) = \frac{3}{2}l + \frac{1}{2}p_s = 4l$. As shown in Figure 21, the duration between the start time of the two conflicts equals to the duration of the conflicts among m_0, m_1 and m_j, which means that the new conflict of m_0 and m_{j+1} starts when the conflicts among m_0, m_1 and m_j finishes. There is no "rest time" between the conflicts among m_0, m_1 and m_j and the conflicts among m_0, m_1 and m_{j+1}. So, the conflict of m_0 and m_1 always happen during the conflicts with other higher priority messages and will not induce more stall time alone.

For any two consecutive messages, m_i and m_{i+1}, we can show the message progress, similar to the process above. Conflicts always happen when the lower priority messages are going down (conflict situation 1). Even though the two messages going down conflict with each other (conflict situation 3), each gets a chance to make progress when the other one is stalled due to the conflicts with higher priority messages (the newer message m_{i+1} will never get ahead of the older message m_i); both messages finally can reach to the destination.

The proof above is for the worst case for odd separation ($\frac{p_s}{l} = 5$). Outside the worst case, the message density is lower, and therefore fewer conflicts and stalls will happen, which comes to the same conclusion.

□

Lemma 5.2.4. *When $\lfloor \frac{p_s}{l} \rfloor$, all messages will be delivered, if $\lfloor \frac{p_s}{l} \rfloor$ is even*

Proof. Similar to odd value of $\lfloor \frac{p_s}{l} \rfloor \geq 5$, we first consider the worst case of the smallest separation of two consecutive messages when $\lfloor \frac{p_s}{l} \rfloor$ is even, $\frac{p_s}{l} = 6$. We show the lemma is true for the base case of m_0 and m_1, and then generalize to any two consecutive messages, m_i and m_{i+1}. There are three cases:

(1) When both m_0 and m_1 go up ($\lfloor \frac{t}{l} \rfloor < n$), $\Delta h(m_0, m_1) = \lfloor \frac{t}{l} \rfloor - \lfloor \frac{t-p_s}{l} \rfloor = \frac{p_s}{l} = 6 > 3$, there is no conflict.

(2) When m_0 goes down ($\lfloor \frac{t}{l} \rfloor \geq n$) and m_1 goes up ($\lfloor \frac{t-p_s}{l} \rfloor < n$), $\Delta h(m_0, m_1) = 2n - \lfloor \frac{t}{l} \rfloor - \lfloor \frac{t-p_s}{l} \rfloor = 2n - 2\lfloor \frac{t}{l} \rfloor + \frac{p_s}{l}$. The conflict only involves the conflict situation 1. Since $\frac{p_s}{l} = 6$ is even ($\Delta h(m_0, m_1)$ is even), the conflict happens with $\Delta h(m_0, m_1) = 2$ and can be resolved

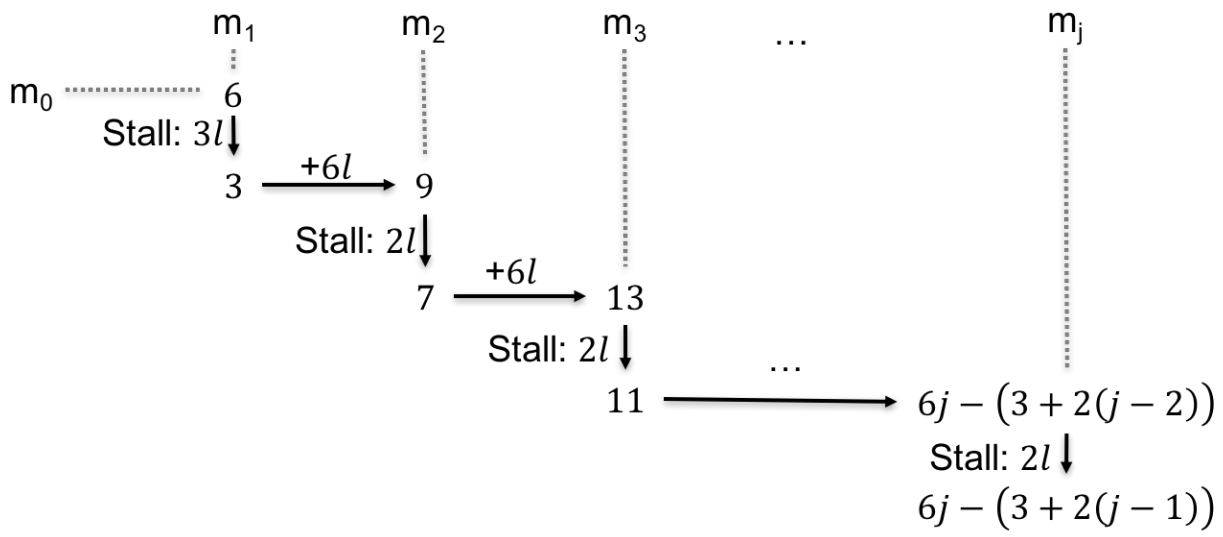


Figure 22: The calculation process of level separations with higher priority messages for m_0 and m_1, when $\frac{p_s}{l} = 6$

in $3l$ time slots. By solving $\Delta h(m_0, m_1) = 2n - 2\lfloor \frac{t}{l} \rfloor + \frac{p_s}{l} = 2$, we get $\lfloor \frac{t}{l} \rfloor = n + 2$. After this conflict, m_0 stays at the same level as the conflict before (stalled), $h(m_0) = 2n - \lfloor \frac{t}{l} \rfloor = n - 2$; m_1 goes up 2 levels and $h(m_1) = \lfloor \frac{t - p_s}{l} \rfloor + 3 = n - 1$. Although the level difference is 1, m_0 and m_1 are in the situation shown in Figure 15(d), there is no more conflict between m_0 and m_1. Figure 22 shows that the level separation of m_0 and m_1 is 6 to start with (before conflict), going down to 3, after the conflict (because m_1 advances 3 levels while m_0 stalls).

(3) Similar to the case of $\frac{p_s}{l} = 5$, when both m_0 and m_1 go down, both m_0 and m_1 will conflict with higher priority messages, m_2, m_3, ..., m_j. These conflicts involve the conflict situation 1, given that m_2, m_3, ..., m_j go up. For both m_0 and m_1, only the first conflict starts with an even level separation (for m_0 see case (2) above) and the rest of conflicts are all odd. Therefore, as shown in Figure 22, conflicts after the first conflict are resolved in $2l$ time slots. A similar process can be followed for m_1. Table 3 shows the total stalls in terms of the number of time slots when m_0 and m_1 conflict with m_2, m_3, ..., m_j under the condition $\frac{p_s}{l} = 6$. Below, we separate this into three subcases to show how these conflicts are resolved: (3A) m_0 and m_1 conflicting with m_2, (3B) m_0 and m_1 conflicting with m_3 and

Table 3: The total stalls of m_0 and m_1 (i.e., d_0 and d_1) when m_0 and m_1 conflict with higher priority messages ($\frac{p_s}{l} = 6$)

	m_1	m_2	m_3	...	m_j
m_0	$3l$	$(3+2)l$	$(3+2*2)l$		$3l+2(j-1)l$
m_1	-	$3l$	$(3+2)l$		$3l+2(j-2)l$

(3C) m_0 and m_1 conflicting with m_j.

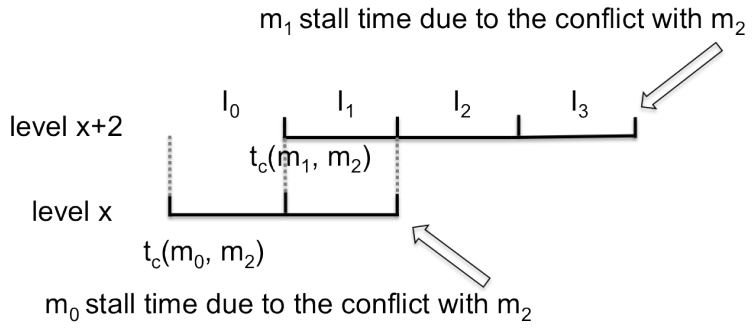


Figure 23: The stall time for m_0 (lower segments) and m_1 (upper segments), when conflicting with m_2

Case 3A: m_0 and m_1 conflict with m_2. m_0 and m_1 will not be completely blocked during the conflicts with m_2: m_0 will go down for 2 levels, and m_1 will go down for 1 level. The level of m_0, m_1 and m_2 is $h(m_0) = 2n - \lfloor \frac{t-3l}{l} \rfloor$ (as shown in Table 3, $d_0 = 3l$ due to the conflict with m_1), $h(m_1) = 2n - \lfloor \frac{t-p_s}{l} \rfloor$ and $h(m_2) = \lfloor \frac{t-2p_s}{l} \rfloor$, respectively. When m_0 starts conflicting with m_2, $\Delta h(m_0, m_2) = 2n - \lfloor \frac{t-3l}{l} \rfloor - \lfloor \frac{t-2p_s}{l} \rfloor = 1$, and we get $\lfloor \frac{t}{l} \rfloor = n + \lfloor \frac{p_s}{l} \rfloor + 1$, so $t_c(m_0, m_2) = nl + p_s + l$ and $h(m_0) = 2n - \lfloor \frac{t-3l}{l} \rfloor = 2n - \lfloor \frac{t}{l} \rfloor + 3 = n - \frac{p_s}{l} + 2$. When m_1 starts conflicting with m_2, $\Delta h(m_1, m_2) = 2n - \lfloor \frac{t-p_s}{l} \rfloor - \lfloor \frac{t-2p_s}{l} \rfloor = 2$, and we get $\lfloor \frac{t-p_s}{l} \rfloor = n + \frac{1}{2} \lfloor \frac{p_s}{l} \rfloor - 1$, so $t_c(m_1, m_2) = nl + \frac{3}{2}p - l$ and $h(m_1) = 2n - \lfloor \frac{t-p_s}{l} \rfloor = n - \frac{1}{2}\frac{p_s}{l} + 1$. $\Delta h(m_1, m_0) = \frac{1}{2}\frac{p_s}{l} - 1 = 2$, which means m_0 and m_1 will conflict again (conflict situation 3) with each other given that m_0 got stalled before m_1 conflicts with m_2. $t_c(m_1, m_2) - t_c(m_0, m_2) = \frac{1}{2}p_s - 2l = l$. Figure

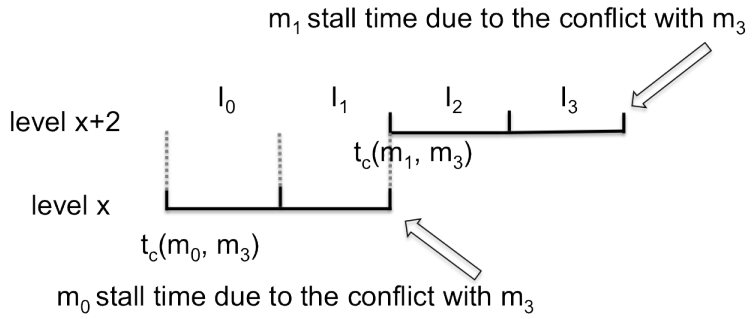


Figure 24: The stall time for m_0 (lower segments) and m_1 (upper segments), when conflicting with m_3

23 represents the stall time for m_0 and m_1. During I_0, m_0 conflicts with m_2 (and stalls), while m_1 keeps going down for 1 level and m_2 goes up for 1 level. During I_1, m_0 conflicts with both m_1 and m_2; m_1 conflicts with m_2; only m_2 (the highest priority message) goes up 1 level. During I_2 to I_3, m_1 conflicts with m_2, allowing m_0 to go down 2 levels and m_2 to go up 2 levels.

m_0 and m_1 will not conflict with m_3, since during I_3 ($\lfloor \frac{t}{l} \rfloor = n + \lfloor \frac{p_s}{l} \rfloor + 4$), the level of m_0, m_1 and m_3 is $h(m_0) = n - \frac{p_s}{l} + 2$, $h(m_1) = n - \frac{1}{2}\frac{p_s}{l} + 1$ and $h(m_3) = \lfloor \frac{t-3p_s}{l} \rfloor = \lfloor \frac{t}{l} \rfloor - \frac{3p_s}{l} = n - \frac{2p_s}{l} + 4$, respectively, with $\Delta h(m_0, m_3) = 4$ and $\Delta h(m_1, m_3) = 6$ both greater than 3, m_3 will not conflict with m_0 and m_1. Thus, m_3 and its other higher priority messages will not conflict with m_0 and m_1 during I_0 to I_3 (see Figure 23).

Case 3B: m_0 and m_1 conflict with m_3. m_0 and m_1 will not be blocked during the conflicts with m_3: m_0 will go down for 2 levels, and m_1 will go down for 2 levels. The level of m_0, m_1 and m_3 is $h(m_0) = 2n - \lfloor \frac{t-5l}{l} \rfloor$ (as shown in Table 3, $d_0 = 5l$ due to the conflicts with m_1 and m_2), $h(m_1) = 2n - \lfloor \frac{t-p_s-3l}{l} \rfloor$ (as shown in Table 3, $d_1 = 3l$ due to the conflict with m_2) and $h(m_3) = \lfloor \frac{t-3p_s}{l} \rfloor$, respectively. When m_0 starts conflicting with m_3, $\Delta h(m_0, m_3) = 2n - \lfloor \frac{t-5l}{l} \rfloor - \lfloor \frac{t-3p_s}{l} \rfloor = 1$, and we get $\lfloor \frac{t}{l} \rfloor = n + \frac{3}{2}\lfloor \frac{p_s}{l} \rfloor + 2$, so $t_c(m_0, m_3) = nl + \frac{3}{2}p_s + 2l$ and $h(m_0) = 2n - \lfloor \frac{t-5l}{l} \rfloor = 2n - \lfloor \frac{t}{l} \rfloor + 5 = n - \frac{3}{2}\frac{p_s}{l} + 3$. When m_1 starts conflicting with m_3, $\Delta h(m_1, m_3) = 2n - \lfloor \frac{t-p_s-3l}{l} \rfloor - \lfloor \frac{t-3p_s}{l} \rfloor = 1$ and get $\lfloor \frac{t-p_s}{l} \rfloor = n + \lfloor \frac{p_s}{l} \rfloor + 1$, so $t_c(m_1, m_3) = nl + 2p_s + l$ and $h(m_1) = 2n - \lfloor \frac{t-p_s-3l}{l} \rfloor = 2n - \lfloor \frac{t-p_s}{l} \rfloor + 3 = n - \frac{p_s}{l} + 2$. The level

difference between m_1 and m_0 is $\Delta h(m_1, m_0) = \frac{1}{2}\frac{p_s}{l} - 1 = 2$, which means m_0 and m_1 conflict with each other again. The start conflicting time difference is $t_c(m_1, m_3) - t_c(m_0, m_3) = \frac{1}{2}p_s - l = 2l$. Figure 24 illustrates stall intervals for m_0 and m_1. During I_0 to I_1, m_0 conflicts with m_3, allowing both m_1 and m_3 to go down and up for 2 levels, respectively. During I_2 to I_3, m_1 conflicts with m_0 and m_3, allowing both m_0 and m_3 to go down and up for 2 levels, respectively. Even though m_0 and m_1 conflict, each can move further by 2 levels when the other one conflicts with m_3.

Similar to case 3A, m_4 cannot conflict with m_0 and m_1 during the conflict from I_0 to I_3 in Figure 24. Since during I_3 ($\lfloor \frac{t}{l} \rfloor = n + \frac{3}{2}\lfloor \frac{p_s}{l} \rfloor + 5$), the level of m_0, m_1 and m_4 is $h(m_0) = n - \frac{3}{2}\frac{p_s}{l} + 3$, $h(m_1) = n - \frac{p_s}{l} + 2$ and $h(m_4) = n - \frac{5}{2}\frac{p_s}{l} + 5$, respectively, with $\Delta h(m_0, m_4) = 4$ and $\Delta h(m_1, m_4) = 6$ both greater than 3, m_4 will not conflict with m_0 and m_1. Therefore, m_4 and its other higher priority messages will not conflict with any messages from I_0 to I_3.

Case 3C: m_0 and m_1 conflict with m_j ($j \geq 2$). m_0 and m_1 will not be blocked during the conflict and can go down by 2 levels. The level of m_0, m_1 and m_j is $h(m_0) = 2n - \left\lfloor \frac{t-(3+2(j-2))l}{l} \right\rfloor$ (as shown in Table 3, $d_0 = (3 + 2(j-2))l$ due to the conflicts with m_1, m_2, ..., m_{j-1}), $h(m_1) = 2n - \left\lfloor \frac{t-p_s-(3+2(j-3))l}{l} \right\rfloor$ (as shown in Table 3, $d_1 = (3 + 2(j-3))l$ due to the conflicts with m_2, ..., m_{j-1}) and $h(m_j) = \lfloor \frac{t-jp_s}{l} \rfloor$, respectively. In general, when m_0 starts conflicting with m_j, $\Delta h(m_0, m_j) = 2n - \left\lfloor \frac{t-(3+2(j-2))l}{l} \right\rfloor - \lfloor \frac{t-jp_s}{l} \rfloor = 1$, and we get $\lfloor \frac{t}{l} \rfloor = n + \frac{j}{2}\lfloor \frac{p_s}{l} \rfloor + j - 1$, so $t_c(m_0, m_j) = nl + \frac{j}{2}p_s + (j-1)l$ and $h(m_0) = 2n - \left\lfloor \frac{t-(3+2(j-2))l}{l} \right\rfloor = 2n - \lfloor \frac{t}{l} \rfloor + (3 + 2(j-2)) = n - \frac{j}{2}\frac{p_s}{l} + j$. When m_1 starts conflicting with m_j, $\Delta h(m_1, m_j) = 2n - \left\lfloor \frac{t-p_s-(3+2(j-3))l}{l} \right\rfloor - \lfloor \frac{t-jp}{l} \rfloor = 1$, and we get $\lfloor \frac{t-p_s}{l} \rfloor = n + \frac{j-1}{2}\lfloor \frac{p_s}{l} \rfloor + j - 2$, so $t_c(m_1, m_j) = nl + \frac{j+1}{2}p_s + (j-2)l$ and $h(m_1) = 2n - \left\lfloor \frac{t-p_s-(3+2(j-3))l}{l} \right\rfloor = 2n - \lfloor \frac{t-p_s}{l} \rfloor + (3 + 2(j-3)) = n - \frac{1}{2}(j-1)\frac{p_s}{l} + j - 1$. The level difference between m_1 and m_0 is $\Delta h(m_1, m_0) = \frac{1}{2}\frac{p_s}{l} - 1 = 2$, which means m_0 and m_1 will conflict again (conflict situation 3). The start conflict time difference is $t_c(m_1, m_j) - t_c(m_0, m_j) = \frac{1}{2}p_s - l = 2l$. The stall time for both m_0 and m_j is the same as Figure 24: during the conflict, m_1 can go down for 2 levels during I_0 to I_1; and m_0 can go down for 2 levels during I_2 and I_3.

Also, m_{j+1} will not conflict with m_0 and m_1 from I_0 to I_3. Since during I_3 ($\lfloor \frac{t}{l} \rfloor = n + \frac{j}{2}\lfloor \frac{p_s}{l} \rfloor + j + 2$), the level of m_0, m_1 and m_4 is $h(m_0) = n - \frac{j}{2}\frac{p_s}{l} + j$, $h(m_1) = n - \frac{1}{2}(j-1)\frac{p_s}{l} + j - 1$

and $h(m_{j+1}) = n - (\frac{j}{2}+1)\frac{p_s}{l} + j + 2$, with $\Delta h(m_0, m_{j+1}) = 4$ and $\Delta h(m_1, m_{j+1}) = 6$, m_{j+1} and its higher priority messages will not conflict with m_0 and m_1 from I_0 to I_3. This pattern will repeat itself indefinitely in the worst case.

Similar to the reason of the general case of $\frac{p_s}{l} = 5$, there is no delay caused by the conflict of m_0 and m_1 for Case 3A, 3B and 3C above. For any two consecutive messages, m_i and m_{i+1}, even though they conflict with each other during the downside transmission, each gets a chance to make progress and finally reaches to the destination.

The proof above is for the worst case of $\frac{p_s}{l} = 6$. For the other even values of $\lfloor \frac{p_s}{l} \rfloor$ will obviously come to the same conclusion. □

5.3 Worst-case End-to-end Delay Determination

Based on Lemmas 5.2.1, 5.2.2, 5.2.3 and 5.2.4, we have the message schedulability condition: $\lfloor \frac{p_s}{l} \rfloor \geq 5$, which is independent of the number of hops n. We assume that a message already conflicted with $(Q-1)$ higher priority messages. The upper-bound of the total stalls is $3l(Q-1)$ (given that each conflict can be resolved in at most $3l$ time slots for conflict situation 1). The following formula shows the difference in levels between m_i and m_{Q+i}, which is $\Delta h(m_i, m_{Q+i})$; if that value is 1 or 2, the Q^{th} conflict will happen: $1 \leq \Delta h(m_i, m_{Q+i}) = 2n - \left\lfloor \frac{t - ip_s - 3(Q-1)l}{l} \right\rfloor - \lfloor \frac{t - ip_s - Qp_s}{l} \rfloor \leq 2$. Based on properties of the floor operation, we can get:

$$
\begin{aligned}
1 \leq 2n - \left\lfloor \frac{t - ip_s}{l} \right\rfloor + 3(Q-1) - \left\lfloor \frac{t - ip_s}{l} \right\rfloor + \left\lfloor \frac{Qp_s}{l} \right\rfloor - 1 \leq \Delta h(m_i, m_{Q+i}) \\
\leq 2n - \left\lfloor \frac{t - ip}{l} \right\rfloor + 3(Q-1) - \left\lfloor \frac{t - ip_s}{l} \right\rfloor + \left\lfloor \frac{Qp_s}{l} \right\rfloor \leq 2
\end{aligned}
$$

Then, we get $2n - \lfloor \frac{t - ip_s}{l} \rfloor + 3(Q-1) - \lfloor \frac{t - ip_s}{l} \rfloor + \lfloor \frac{Qp_s}{l} \rfloor = 2$. Therefore, we can get Equation 5.1.

Since the conflicts happen only when a message is transmitted down, the following condition holds about the level of message m_i: $n \leq \left\lfloor \frac{t - 3(Q-1) - ip_s}{l} \right\rfloor \leq 2n - 1$, so $n + 3(Q-1) \leq$

$$\left\lfloor \frac{t - ip_s}{l} \right\rfloor = n + \frac{3}{2}(Q-1) + \frac{1}{2}\left\lfloor \frac{Qp}{l} \right\rfloor - 1 \tag{5.1}$$

Table 4: Simulation parameters and values

parameters	values
p	0.05s, 0.1s, 0.15s, 0.2s, 0.25s, 0.3s
p_s	5, 10, 15, 20, 25, 30
l	1, 2, 3, 4
n	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

$\left\lfloor \frac{t-ip_s}{l} \right\rfloor \leq 2n - 1 + 3(Q-1)$. Put the Equation (5.1) into the above condition, we get $\frac{l}{3l-p_s} \leq Q \leq \frac{2nl-3l}{p_s-3l}$ and derive the maximum Q as $\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor$. After calculating the maximum number of conflicts, we can estimate the worst-case stalls in terms of the number of time slots caused by conflicts, $D_{conflict} = 3lQ = 3l\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor$. Recalling that the delay without conflict, $D_{pure} = 2nl$, the worst-case end-to-end delay in terms of the number of time slots is $D_{slots} = D_{pure} + D_{conflict} = 2nl + 3l\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor$.

To determine the worst-case end-to-end delay, we multiply D_{slots} by Δt, and obtain $D^{worst}_{network} = (2nl + 3l\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor)\Delta t$. Note that our derivation is general and scalable to any network topology with n hops and l lines of nodes.

5.4 Worst-case End-to-end Delay Analysis Validation

To validate our worst-case end-to-end delay analysis, we implement a simulation to simulate the process of the dynamic message transmission. Recall that the schedulability con-

dition ($\lfloor \frac{p_s}{l} \rfloor \geq 5$) is to determine if a message can be delivered to the destination within a limited amount of time. We carried out a set of tests on our simulation with different values of p, l and n, as shown in Table 4, where each test corresponds to a value of p, l and n.

Our test set can be divided into two test sets, *test set 1*, where all the tests meet the condition and *test set 2* where all the tests do not meet the condition. We test the schedulability condition on the test set and calculate the test accuracy by summing the percentage of the tests that can deliver the messages within a limited amount of time for test set 1 and the percentage of the tests that cannot deliver the messages within a limited amount of time for test set 2. We get 100% accuracy for the schedulability condition test, demonstrating the correctness of our schedulability condition. Under the test set 2, the worst-case delay analysis overestimates the delay by 4.2% compared with the realistic simulation results (always pessimistic, but a very tight pessimism).

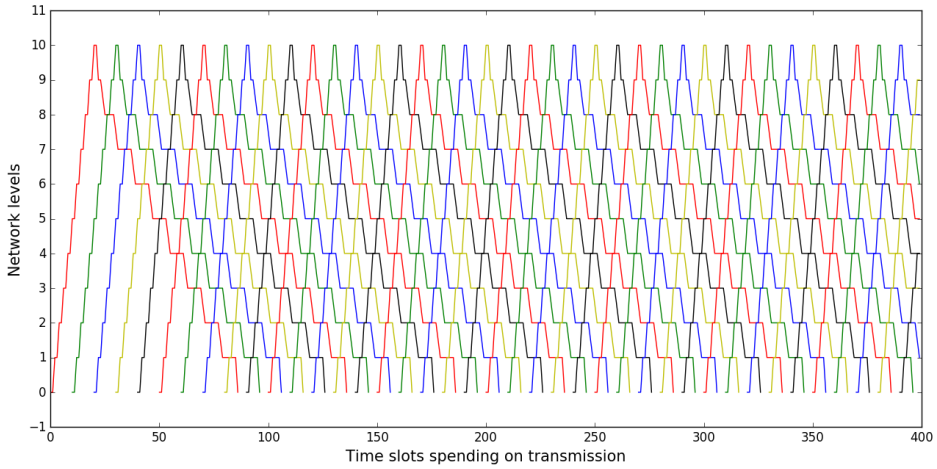
Figure 25 shows the examples of message transmission process of the most recent message first (Figure 25(a)) and the oldest message first scheduling schemes (Figure 25(b)) with $p = 0.1s$, $p_s = 10$, $l = 2$ and $n = 10$. For the most recent message first scheme, as discussed above and shown in Figure 25(a), the lower priority messages conflict with higher priority messages and are delayed when traveling "down". As analyzed in Section 5.2, when a message traveling down, it is delayed at every level starting with level $n - 2$ (level 8 in this example) but can still move down by 1 level until it reaches to the destination. Regardless, they still arrive at the controller within the deadlines, because the condition $\lfloor \frac{p_s}{l} \rfloor \geq 5$ is satisfied.

For the oldest message first, as shown in Figure 25(b), the conflicts happen when lower priority messages (later messages) traveling up. The message transmissions are *unsteady* (i.e., the end-to-end delays of the messages are not the same) at first, given that there are not many higher priority messages ahead, so the delay is less for the earlier than for the later (lower priority) messages. The transmissions get *steady* (The steady state of message delays, that is, the end-to-end delays of the messages are the same.) after the 275 time slots and the transmission process is symmetric with the most recent message first scheme. The proof process for the oldest message first scheme is exactly the same as the most recent message first scheme, that is, starting with the first two lowest priority messages, which are the last two messages for the oldest message first scheme. Note that if the schedulability condition

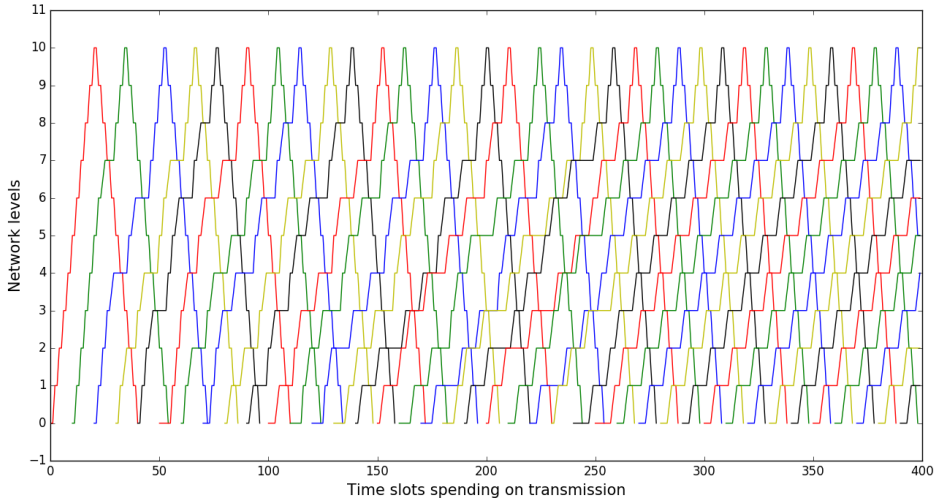
is not met, the oldest message first scheme will be always unsteady, but can still delivery messages to the destination; this is a significant difference from the most recent message first scheme. However, the end-to-end network delay is unbounded (becomes larger and larger), since more and more conflicts are accumulated and are unresolved.

5.5 Summary

In this chapter, we carried out the worst-case end-to-end delay analysis for the two-way wireless transmission in a WCS with one single physical system. From the transmission conflict analysis, we get the schedulability condition, $\left\lfloor \frac{p_s}{l} \right\rfloor \geq 5$. Based on the condition, we calculate the maximum number of conflicts during one message transmission as $\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor$. With the maximum number of conflicts, we derive the worst-case end-to-end delay as $D_{network}^{worst} = (2nl + 3l\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor)\Delta t$. The simulation results show 100% accuracy for the schedulability condition test. With the schedulablity condition satisfied, the simulation results show that our end-to-end delay analysis is accurate within 4.2% of the realistic simulation results (always pessimistic, but a very tight pessimism).



(a)



(b)

Figure 25: Examples of (a) the most recent message first scheme and (b) the oldest message first scheme transmission process with $p = 0.1s$, $p_s = 10$, $l = 2$ and $n = 10$. Note that the symmetry of the oldest message first scheduling scheme with the most recent message first scheduling scheme begins at the 275^{th} time slots.

6.0 Dynamic Network Reconfiguration for WCS with One Physical System

In Chapter 4 and 5 we achieve the control system stability requirement with minimum number of nodes in the network. However, only making sure the system stability is not enough, since network-induced imperfections can still degrade control quality comparing with the wired control system and result in equipment damage and serious economic losses [Yu et al., 2014]. So it is necessary to reduce the network-induced error to make the control system performance as close as the performance of the wired control system.

In this chapter, under the assumption that the control system stability is satisfied, we discuss reducing network-induced error for a WCS with one single physical system for one-way wireless transmission. The trade-off between network delay and packet loss motivates us to find the estimated optimal network configuration (will explain in Section 6.2.2) to minimize the network-induced error for the control system under different LSR values. Another main difficulty of having wireless networks for the control systems is caused by the interference and noise that produce time-varying fault patterns [Cerpa et al., 2005; Srinivasan et al., 2010], which motivates us to find a fast and effective way to carry out network reconfiguration at run time. Our goal is to reduce the network-induced error for a WCS with a single physical system under time-varying network link failures. We design and implement a framework with offline and online components to do network reconfiguration for the control system to tolerate LSR changing over time (caused by the time-varying link failures). To evaluate the control system performance with our network reconfiguration framework, we conduct a systematic case study with a WCS for a single PHX in an NPP.

6.1 Network Reconfiguration Framework

We propose a network reconfiguration framework that has as input a network configuration set, that is, a network topology set generated in Section 4.2.3 based on the node placement design discussed in Section 4.2. Different topologies correspond to different num-

ber of active nodes in the network.

Our framework contains two parts, offline and online, as shown in Figure 26. For the offline part, to quantify the network-induced imperfection impact on control system performance, we propose a network imperfection model to transform network delay and DR to the *total induced delay* on the control system. We estimate the total induced delay for each topology in the network topology set. We find an estimated optimal (will be explained in more details in Section 6.2.2) network topology set for each LSR offline, and store them at the controller node. For the online part, at run time, the network notifies the controller what the estimated LSR is and the controller selects a network topology from the estimated optimal topology set computed offline. The controller then broadcasts the new network topology to all the nodes in the network to carry out a reconfiguration. Therefore, the remote controller acts as a centralized network manager and decides which network configuration should choose.

6.2 Offline Optimal Network Configuration

We first introduce a model describing the network-induced imperfection impact on the control system performance, which quantifies the trade-off between network delay and DR. We then show how to find estimated optimal network topology set by using this model.

6.2.1 Network Imperfection Model

Although previous research discussed how the network reliability and network delay affect the control system performance [Wang et al., 2016; Li et al., 2015], to the best of our knowledge there is still no model that builds the relationship between network performance (i.e. network delay and message loss) and control system performance (i.e. network-induced error). We define the delay induced into the control system by the wireless network as $T_{used} - T_{sensed}$, where T_{used} is the time the measurement signal is used by the controller and T_{sensed} is the time the sensor sends out the sensor measurement. The total delay induced into

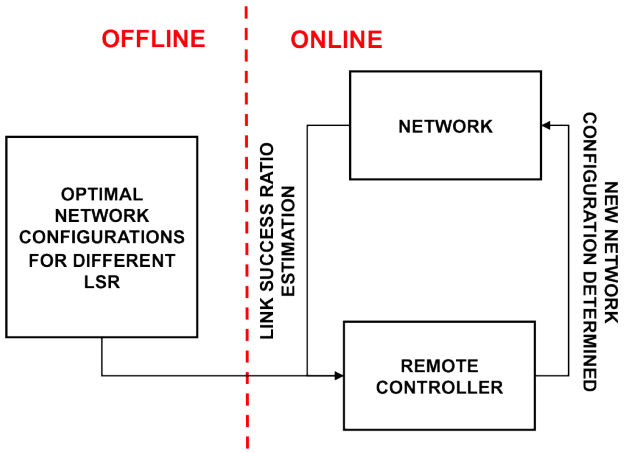


Figure 26: Network reconfiguration framework for the control system with dynamic network interference

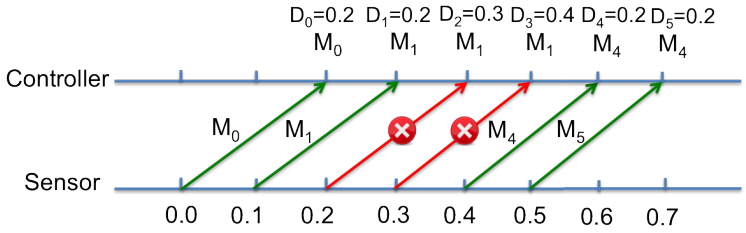


Figure 27: Network delay and delivery ratio trade-off illustration, when network delay is greater than control sampling period ($p = 0.1s$ and $D_{network} = 0.2s$)

$$D = (\left\lceil \frac{D_{network}}{p} \right\rceil + n_{loss})p \tag{6.1}$$

the control system is calculated dynamically every time it is needed, shown in Equation 6.1, where $D_{network}$ is a variable of current end-to-end network delay, p is the control sampling period, n_{loss} is the number of consecutive packet losses. For example, as shown in Figure 27, the control sampling period is 0.1s, but when the network delay is 0.2s and measurement M_2 gets lost, the induced delay D_2 is 0.3s and the controller will use measurement M_1 instead. When the measurement M_3 also gets lost, the induced delay D_3 is 0.4s and the controller will (re-)use measurement M_1. Note that n_{loss} is computed from the control system perspective, that is, if a message is received by the controller every control sampling period, $n_{loss} = 0$.

D is related to both network delay and the number of consecutive packet losses. n_{loss} is estimated by the expected value of the network loss ratio (1-DR), as shown in Equation 6.2 where $(1 - DR)^i DR$ is the probability of i consecutive losses. We assume message losses follow the uniform distribution, since DR can be viewed as the probability a message received by the controller. When the probability is less than a threshold (c), we assume that the probability can be ignored to avoid long-running computations. For example, for $DR = 0.9$ and $c = 0.0009$, the probability of getting 1, 2 and 3 consecutive losses are 0.09, 0.009, and 0.0009, respectively. Therefore, the expected number of consecutive losses is $1 \times (1 - 0.9) \times 0.9 + 2 \times (1 - 0.9)^2 \times 0.9 + 3 \times (1 - 0.9)^3 \times 0.9 = 0.1107$. The situations of 4 or more consecutive losses are ignored, since the probability of 4 consecutive message losses is $(1 - 0.9)^4 \times 0.9 = 0.00009 < c$.

6.2.2 Estimated Optimal Network Configuration Determination

The goal of the offline algorithm is to minimize the network impact (network delay and packet loss) on the control system, using the network imperfection model above. By applying

$$n_{loss} = \sum_{i=1}^{n} i(1-DR)^i DR, \quad (1-DR)^i DR \geq c \tag{6.2}$$

the worst-case network delay ($D_{network}^{worst}$) calculation in Equation (4.5) and DR calculation in Equation (4.4) (so that n_{loss} can be estimated from Equation (6.2)), we are able to estimate the total induced delay D for the network topology set for different LSR values. For each LSR value, our algorithm searches all the network topologies and finds the one with minimum estimated induced delay, D (the topology is called estimated optimal topology given a LSR value. We define the estimated optimal network topology/configuration as the optimal topology/configuration based on the estimated value (i.e., estimated induced delay D)). Thus, we find a set of estimated optimal network topologies with minimum total induced delay D for each LSR value and store them in a look-up table indexed by LSR value. Therefore, our offline algorithm discovers the set of estimated optimal network topologies that will be saved and used later during the online portion.

6.3 Online Network Reconfiguration

Wireless networks, especially in radiation-prone locations, suffer from varying electromagnetic interference, which causes some links, some of the time, to fail. Clearly, static configurations do not adapt to the time-varying noise and interference that can cause **time-varying link failures**, that is, the average LSR remains constant over a period of time (i.e., each link fails independently from other links but with the same probability during that period. We do not consider space-correlated link failures [1]). We devised an online dynamic

[1] Space correlated failures is where one link fails, the links physically close to it have high probability to fail, typically due to a common source of interference.

network reconfiguration to improve the control system performance by reducing the total induced delay D.

The controller carries out both the control and network reconfiguration algorithms, given that it has all the information needed to decide the new configuration (i.e., network topology). The controller detects a reconfiguration is needed (for example, due to interference/noise), and computes and propagates a new network configuration to all the nodes by broadcasting reconfiguration messages. To deal with packet re-ordering, we discard old messages (i.e., the time stamps of the messages are older than the received messages) at the remote controller. In this dissertation, we restrict network configuration to the network topology, even though our offline algorithm (Section 6.2) is general and the offline algorithm can be designed to deal with network configuration variables like data link layer schedule, network routing, etc. We first introduce the network reconfiguration process. Since our online network reconfiguration is based on the offline look-up table given the current LSR, we then propose an algorithm to estimate the average LSR at run time. Finally, we propose six online network reconfiguration algorithms, that is, three original algorithms with two variations each, namely considering or not consecutive packet losses.

6.3.1 Network Reconfiguration Process

Recall that the network topology set is generated following the process discussed in Section 4.2.3. When we do network reconfiguration, that is, changing the network topology from topology A to topology B, there are two cases: (a) topology B has more active nodes than topology A; and (b) topology A has more active nodes than topology B.

For the first case, we need to activate the sleep nodes to be active to get topology B by activating backup nodes in the relay region or/and the backup paths in the k-connected region. For the relay region, we activate the backup nodes from the lowest level to the highest level starting with the line of inactive backup nodes that is close to the primary line of relay nodes, then from the lowest level to the highest level in the second line of inactive backup nodes that is close to the primary line of relay nodes and so on. For the k-connected region, we activate the backup paths in the order of the shortest inactive path, then the second

shortest inactive path and so on. For the second case, opposite behaviors occur to put the active nodes to sleep to get topology B: deactivating the active nodes from the highest level to the lowest level in the relay region starting with the last line of active backup nodes; or/and deactivating the backup paths from the longest active backup path. The speed and how many nodes to activate/deactivate at a time will be explained later in Section 6.3.3 and Section 6.3.4.

6.3.2 Network Average Link Success Ratio Estimation

Since our online network reconfiguration is based on the offline look-up table for each LSR value, we need a way to estimate the average LSR when the reconfiguration algorithm is executed. To estimate the LSR at run time, we propose a jumping window in-network aggregation method. During a certain amount of time, that is, the LSR estimation interval (LSRI), each node counts the number of messages it receives (NMR), n_{rev} and the number of messages it should receive (NMS), n_{should} (knowing the period of message arrival). At the end of the LSRI, each node concatenates the two numbers, n_{rev} and n_{should} to its message, and sends the message to its parent nodes. Then, the parent nodes will sum their own NMR and NMS with their children's NMRs and NMSs, respectively; this repeats until getting to the controller. Eventually, the remote controller will compute the final overall network average LSR by its received n_{rev} and n_{should} as $\frac{n_{rev}}{n_{should}}$. Algorithm 1 shows the LSR estimation algorithm running on one relay node in more detail.

6.3.3 Reconfiguration Not Considering Consecutive Losses

The intuition behind the online algorithm is to find the estimated optimal topology according to the current estimated LSR calculated by the remote controller, and then adjust the current network topology to the estimated optimal topology. We explore three options to reach the estimated optimal topology, given that the reconfiguration depends on the LSR, which cannot be computed instantaneously. We first discuss three algorithms not considering consecutive message losses, which are DirectJump to Optimal (DO), Multiplicative Increase and Conservative Decrease (MICD), and Adaptive Control (AC). The inputs to these algo-

Initialization: $n_{should} = 0$, $n_{rev} = 0$, LSRCounter=0;

while *true* **do**

  if $LSRCounter == LSRI$ **then**

    if *it is time to receive a message* **then**

      get the n^i_{rev} and n^i_{should} from the received message m_i ;

      $n_{rev} = n_{rev} + n^i_{rev}$;

      $n_{should} = n_{should} + n^i_{should}$;

    end

    if *it is time to transmit a message* **then**

      Attach n_{should} and n_{rev} to its message that is going to send;

      Send out the message;

      $n_{rev} = 0$;

      $n_{should} = 0$;

    end

    $LSRCounter$=0;

  else

    if *it is time to receive a message* **then**

      n_{should}++;

      if *receive a message* **then**

        n_{rev}++;

      end

    end

    $LSRCounter$++;

  end

end

Algorithm 1: LSR estimation algorithm running on one active relay node

rithms are the offline look-up table and the LSR computed in Section 6.2 and Section 6.3.2, respectively.

DO (Direct Jump) The controller sends out a message to all participating nodes to adjust the network topology to instantaneously have the estimated network topology whenever the LSR estimation value changes according to the offline look-up table. Algorithm 2 shows the detail.

```
Initialization;
LSRCounter=0;
while true do
    if LSRCounter == LSRI then
        estimate current LSR, curr_LSR;
        look up the offline table and get the new network topology, T based on curr_LSR;
        change the current network topology to T;
        LSRCounter=0;
    end
    LSRCounter++;
end
```

Algorithm 2: Direct jump to optimum (DO)

MICD (Multiplicative Increase Conservative Decrease) Inspired by [Sankarasubramaniam et al., 2003], with a focus on network reliability, the number of sleep nodes is multiplicatively (i.e., very quickly) activated when the number of active nodes in current topology is less than the number of active nodes in the estimated topology based on a changed LSR value (converse of TCP/IP protocols window reduction [Chiu and Jain, 1989]). When the number of active nodes in current topology ($curr_{node}$) is more than the number of active nodes in the estimated topology (est_{node}), the number of active nodes is conservatively deactivated (in our case, we deactivate one active node at a time). Algorithm 3 shows more detail: every LSRI, a certain number of nodes ($change_{nodes}$) is activated or deactivated on top of the current topology to achieve reconfiguration.

AC (Adaptive Control) Inspired by adaptive control theory [Hovakimyan and Cao, 2010], the larger the difference between the number of active nodes in estimated topology

```
Initialization;
LSRCounter=0, est_node=0, increment = 1;
while true do
    if LSRCounter == LSRI then
        get the number of active nodes in current topology, curr_node;
        estimate current LSR, curr_LSR;
        look up the offline table and get the new network topology, T based on curr_LSR;
        est_node=T.node;
        if curr_node < est_node then
            change_node = increment;
            increment = increment×2;
        else if curr_node > est_node then
            change_node = 1;
        else
            increment=0;
        end
        Activate or deactivate change_node nodes on the current network topology to get
         a new topology;
        LSRCounter=0;
    end
    LSRCounter++;
end
```

Algorithm 3: Multiplicative increase and conservative decrease (MICD)

(getting from the offline table based on the estimated LSR) and the number of active nodes in current topology, the faster we activate or deactivate number of nodes in the network. That is, $curr_{node} = \alpha \times curr_{node} + (1-\alpha) \times est_{node}$. Parameter α guides the speed of activation or deactivation of the relay nodes in the network ($0 < \alpha < 1$). When $\alpha = 0$, AC behaves like DO and the speed of activating or deactivating nodes is maximum. When $\alpha = 1$, the current number of nodes does not change, that is, it is a static network. In essence, smaller α implies higher speed to change the current number of active nodes. Algorithm 4 shows more detail.

Initialization;
$LSRCounter$=0, est_{node}=0, $curr_{node} = min_{node}$;
while *true* **do**
  if $LSRCounter == LSRI$ **then**
    get the number of active nodes in current topology, $curr_{node}$;
    estimate current LSR, $curr_{LSR}$;
    look up the offline table and get the new network topology, T based on $curr_{LSR}$;
    est_{node}=$T.node$;
    $new_{node} = \alpha \times curr_{node} + (1 - \alpha) \times est_{node}$;
    Activate or deactivate $|curr_{node}$-$new_{node}|$ nodes on the current network topology to get a new topology;
    $LSRCounter$=0;
  end
  $LSRCounter$++;
end

Algorithm 4: Adaptive control (AC)

6.3.4 Reconfiguration Considering Consecutive Losses

From Equation (6.1), the total induced delay is proportional to the number of consecutive losses n_{loss}. However, in the algorithms above, we did not consider n_{loss}. Since the LSR estimation is not completely accurate, it takes time to reconfigure and the network conditions

vary over time, there could be consecutive message losses, which will degrade the control system performance. In other words, when there are consecutive message losses, we need to make the network more robust (we choose to activate more nodes). As a first experimental step, whenever $n_{loss} \geq q$, we add g more nodes in the network. g and q are user-selected parameters.

Considering consecutive losses, we devise three more online algorithms: CL-DO, CL-MICD, and CL-AC.

6.4 Case Study

We conducted a case study to show and experiment with our wireless network reconfiguration framework for one PHX of one-way wireless transmission. We deploy the same network in the case study as the one in Chapter 4, that is, a network with 12-hop and up to 50 nodes. We use a state-of-the-art cyber-physical system simulator (WCPS 2.0 [Li et al., 2015]) to combine TOSSIM network simulator [Levis et al., 2003] and the PHX simulink model together. The ridesharing protocol is implemented in the TOSSIM simulator with wireless noise traces from a 21-node subset of the WUSTL Testbed [tes, 2017] to more realistically simulate real-life scenario. The online reconfiguration algorithms mentioned in Section 6.3 are implemented on the controller. Note that our simulation of online LSR is done by CPM model [Lee et al., 2007] (embedded in the TOSSIM simulator to simulate realistic scenarios, involving the space correlated link failures), which has the discrepancy with the offline LSR estimation.

To simulate time-varying link failures, we propose a network fault model as follows. We hold RSSI constant for a period of time and change RSSI to another value for the next period of time (the duration is based on the noise traces mentioned above). We adjust each relay node's RSSI to change LSR within the range (0.5, 1.0), depending on the following three quantities: *RSSI duration*: the time interval at which the RSSI is fixed (after that, the RSSI may be changed); *RSSI range* and *time range*: the value and time range the RSSI duration is chosen from. We randomly choose RSSI from *RSSI range* with a uniform distribution and

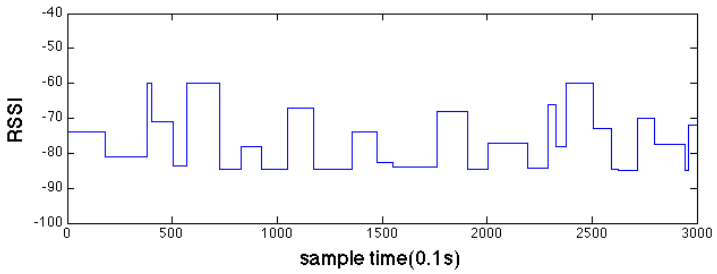


Figure 28: Time-varying RSSI variation example

randomly choose *RSSI duration* from *time range* also with a uniform distribution. Figure 28 shows an example of the generated RSSI over time using our fault model with RSSI range of (-60 dBm, -85 dBm) and time range of (0, 20s).

To evaluate the performance of the control system (in this case the PHX), we adopted two metrics: Integral Absolute Error (IAE) [Li et al., 2016] and Root Mean Square Error (RMSE), which is the RMS error measured between the closed-loop responses using wired control and wireless control under consideration. These metrics quantify the quality of the WCS: the less (error), the better. We also measure three more metrics, the total induced delay D (to analyze IAE and RMSE) of the network imperfection model, the number of active nodes that are used in the network, and the network lifetime. Table 5 shows our simulation parameters and values.

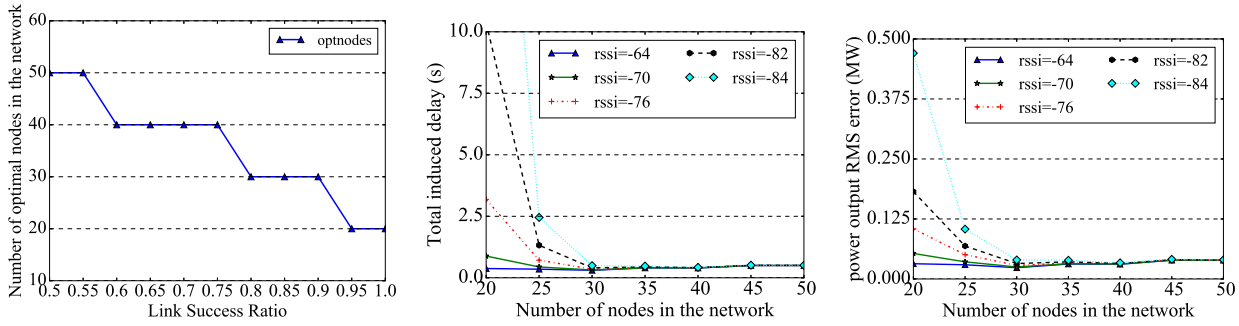
6.5 Case Study Results

6.5.1 Offline Optimal Network Configuration Results

By applying the algorithm in Section 6.2.2, we can get the look-up table containing the estimated optimal topology for each LSR value. Figure 29(a) shows the number of active nodes of the estimated optimal topology for different LSR values. The higher the LSR, the higher the percentage of packets that get delivered, and the more robust the

Table 5: Parameters and values

Parameters	Values
Control sampling period (p)	0.1s
TDMA time slot (Δt)	0.01s
Simulation time	300s
RSSI range	(-60 dBm, -85 dBm)
time range	(0, 20s)
LSRI values	2s, 4s, 8s, 12, 16s, 20s
Reference functions	ramp30, ramp60, ramp90, ramp120
α value	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9



(a) (b) (c)

Figure 29: (a) The number of active nodes of the offline estimated optimal topology with different LSR values; (b) total induced delay result for RSSI values of -64, -70, -76, -82 and -84 that correspond to average LSR values of 0.93, 0.88, 0.82, 0.77, and 0.72, respectively; (c) power output RMSE for different number of active nodes in the network.

network will be, and therefore the fewer relay nodes needed. The optimal number is always a multiple of 10 due to the ceiling operation in our network imperfection model (see Equation (6.1)). In details, $D^{worst}_{network} = n_{active}\Delta t$ (from Equation (4.5)) is multiple of time slot duration $\Delta t = 0.01s$; when $n_{active} = 10x$ ($x \geq 1$), where x is an integer, the network is more reliable (less n_{loss}) than the network of $n_{active} = 10(x-1)+y$ ($y = 1, .., 9$) but with the same value of the term $\left\lceil \frac{D^{worst}_{network}}{p} \right\rceil$ ($p = 0.1$). For example, a network with 25 active nodes (network delay is 0.25s, but is considered as 0.3s due to the ceiling operation) will definitely have more total delay D than the network with 30 active nodes (network delay is also 0.3s), since 30 nodes is more reliable and has fewer n_{loss} than 25.

To see the correlation of the network imperfection model from Section 6.2.1 and the control system performance, we run the simulation with static RSSI values. Figure 29(b) shows the total induced delay for different number of nodes and different RSSI values. Note that when the number of nodes is 20, the network is not robust (delivery ratio is less than 0.6 when $RSSI = -64$ and less than 0.1 when $RSSI = -84$ according to the results in Figure 12(a)) and has more consecutive message losses, thus has more induced delay even though the actual network delay is the lowest (for the messages that are actually delivered), since it has the smallest number of nodes. When the number of nodes is 50, the network is the most robust and almost does not have consecutive message losses (DR is above 0.9 when $RSSI = -84$ according to Figure 12(a)), but still has induced delay due to the highest network delay.

Figure 29(c) shows the power output RMSE of the PHX. Comparing Figure 29(b) and Figure 29(c), we can see that our network imperfection model is accurate visually and statistically (Pearson correlation $r = 0.993$, $p < 0.001$) correlating well to the power output RMSE.

6.5.2 Online Network Reconfiguration Results

To simulate time-varying link quality models, we varied the RSSI range and time range to get different representative network fault models (Section 6.4) with different average RSSI values over the simulation time. We simulate our system on five fault models with average

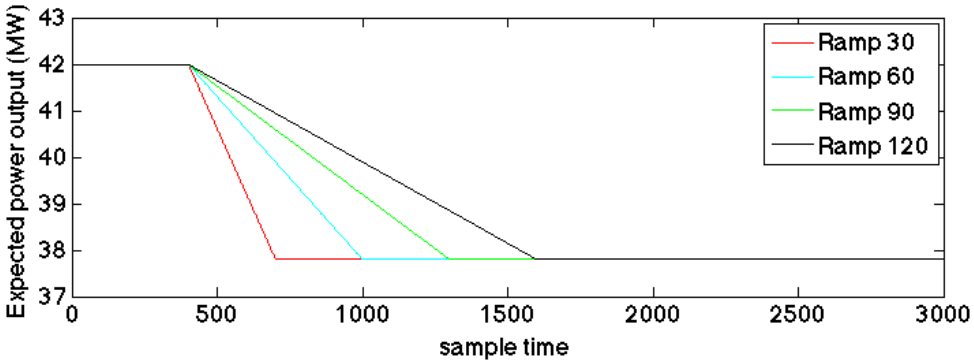


Figure 30: Control system power reference functions

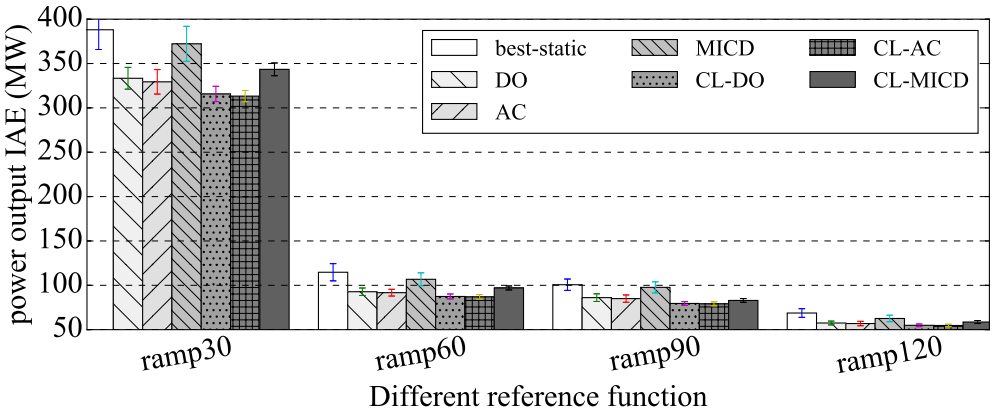
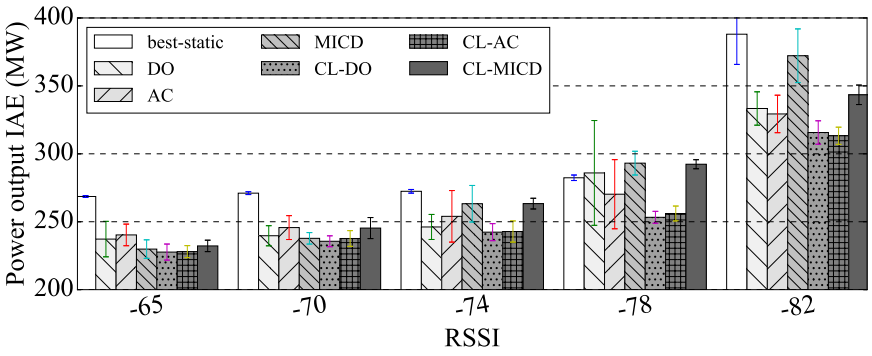


Figure 31: Power output IAE for different reference functions (average RSSI: -82dBm; LSRI: 2s (20 samples))

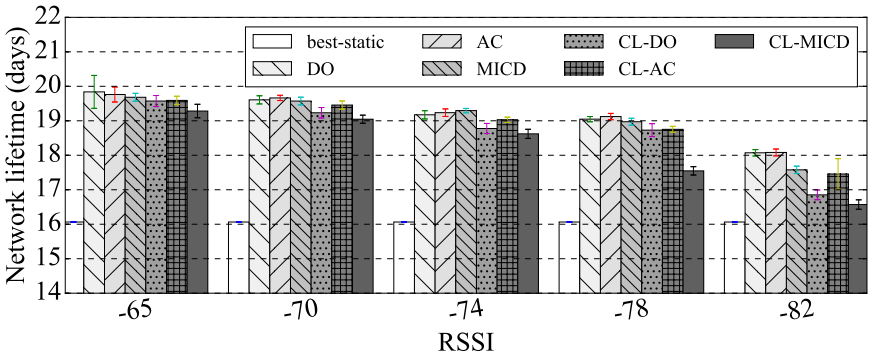
RSSI values (in dBm) of -65, -70, -74, -78 and -82. In this section, we present results of control system performance (RMSE and IAE) and network lifetime; and the number of nodes, total induced delay and RMSE changing over time for different online reconfiguration algorithms.

Heat exchanger system power reference function To study the behavior of the PHX, we consider the case when an operator changes the power output for the reactor. We set the power reference function (i.e., the required power output of a nuclear reactor), to reduce power from 42 MW to 37.8 MW (typical values for NPPs) as shown in Figure 30 (sample time is time/p). Ramp30 means that it takes 30 seconds to reduce the power from 42 MW to 37.8 MW. We consider four reference functions in this chapter: ramp30, ramp60, ramp90 and ramp120. As shown in Figure 31, the steeper the reference function, the larger the IAE. This is because when the reference function is steep, it requires the control system to reduce its power output more aggressively, and thus it will have more transient response, causing larger IAE. In the figure, "best-static" is the best fixed number of active nodes in the network, as follows. For each reference function, we tested the number of nodes 20 to 50 for each fault model and chose the static scheme with minimum average IAE over all the fault models (in our case it is the static scheme with 40 active nodes). As shown in Figure 31, online algorithms all perform better than the best-static with the average improvement of 16%, 23%, 19% and 20% for ramp30, ramp60, ramp90 and ramp120, respectively. Note that the online network reconfiguration algorithms have similar trends for all reference functions as shown in Figure 31. We only present the results for ramp30 in the following sections.

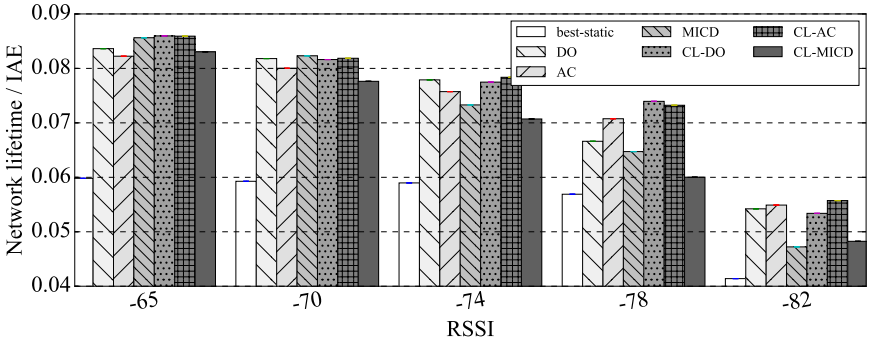
Comparison of Online Reconfiguration Algorithms Figure 32(a) shows the power output IAE of the PHX for different average values of RSSI and different online network reconfiguration algorithms; RMSE results are similar to the IAE results and are thus omitted. For the first study, in this case study, we add 3 more nodes ($g = 3$) in the CL-* algorithms when 3 consecutive message losses ($q = 3$) happen. As the average RSSI value decreases, the power output IAE increases, since the network has more interference. As expected, our dynamic algorithms perform typically better than the static scheme for all fault models. CL-DO and CL-AC algorithms perform better than the other dynamic algorithms, because they add more nodes only when needed, that is, when the network has consecutive message



(a)



(b)



(c)

Figure 32: (a) Power output IAE and (b) network lifetime (c) network lifetime / IAE results for different RSSI values (LSRI: 2s; α: 0.1)

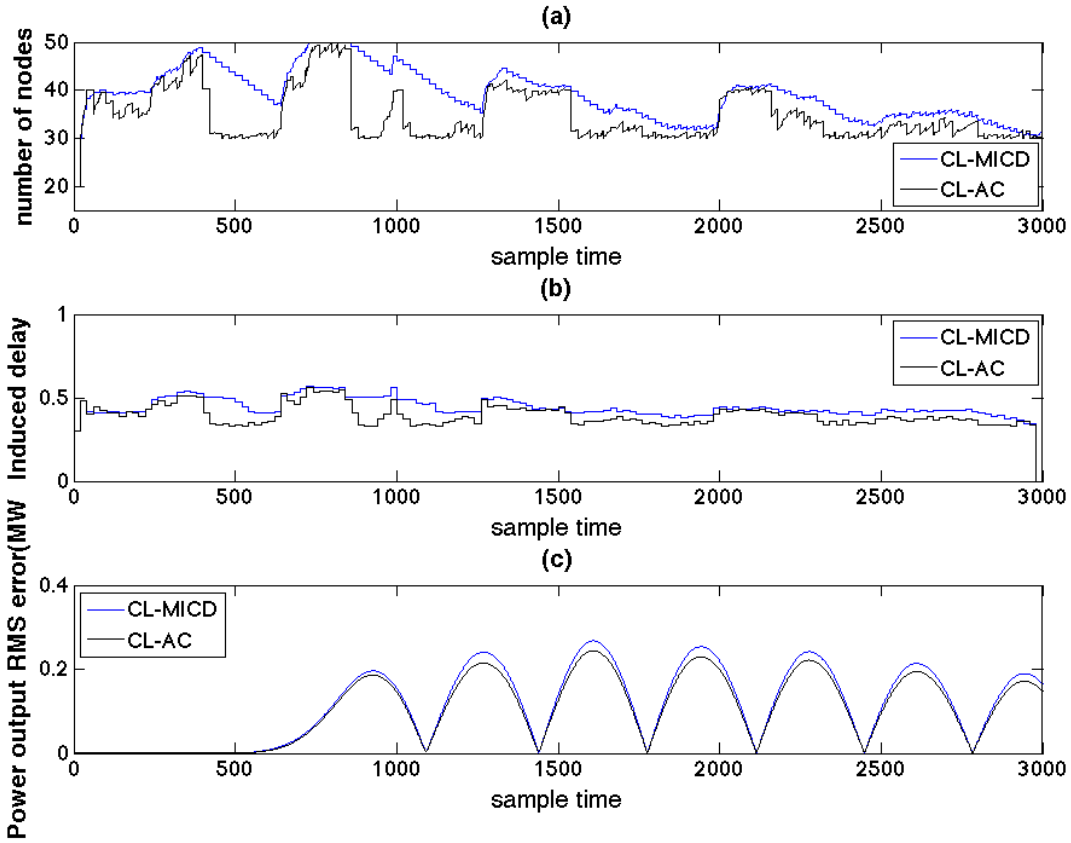
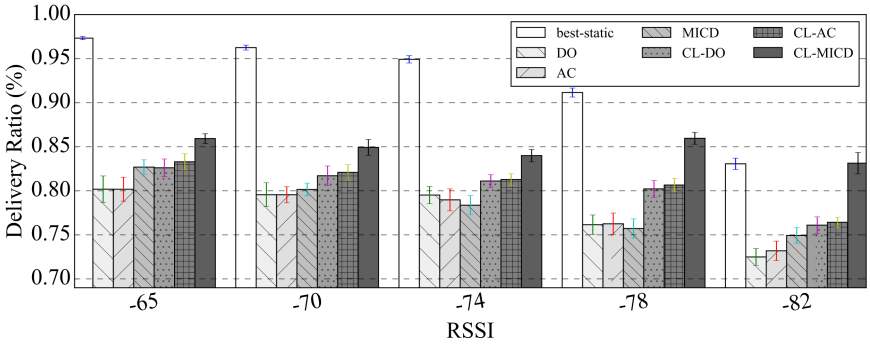
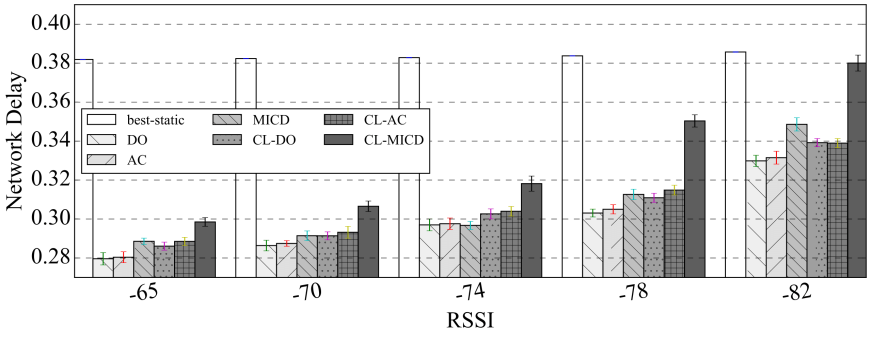


Figure 33: (a) Average number of nodes in the network, (b) average induced delay and (c) average RMSE over 20 experiments changing over time (LSRI: 2s; average RSSI: -82dBm; α: 0.1)



(a)



(b)

Figure 34: (a) Network delivery ratio; (b) network delay for different average RSSI values (LSRI: 2s; α: 0.1)

losses. But CL-MICD always perform worse than CL-DO and CL-AC on average by 7.8% and 7.4% over five fault models, respectively. Figure 33 shows the comparison of CL-MICD and CL-AC on 20 experiments over 3,000 samples. The reason CL-MICD always performs the worst among CL-* algorithms is because the speed of reducing the number of nodes is slow (reduce one at a time) and the speed of adding nodes is fast (exponential increase), which often overshoots the number of actually needed nodes and thus causes more induced delay (induced delay of CL-MICD is always higher than CL-AC) and degrades the control system performance.

We calculate the average network lifetime to measure the network energy consumption. We define our network lifetime as the average relay node lifetime, and calculate the average network energy consumption over different topologies experiencing in the online schemes for one round of measurement transmissions from the measurement sensors to the controller. For simplicity, we assume a general battery capacity is 8640J, which is the typical capacity of two AA batteries. Figure 32(b) shows the network lifetime for different reconfiguration algorithms. Algorithms considering consecutive message losses (CL-DO, CL-AC and CL-MICD) consume more energy than their counterparts not considering consecutive message losses (DO, AC and MICD). This is because CL-* algorithms are more aggressive activating additional nodes when there are consecutive losses. In addition, from Figure 32(b), we found that when there is more interference in the network, the network consumes more energy, since the network needs more backup nodes to handle link failures. For network performance results, see Figure 34(a) and Figure 34(b). As the average RSSI value decreases, indicating more interference in the network environment, the DR decreases, but the network delay increases since more active nodes are reconfigured participating.

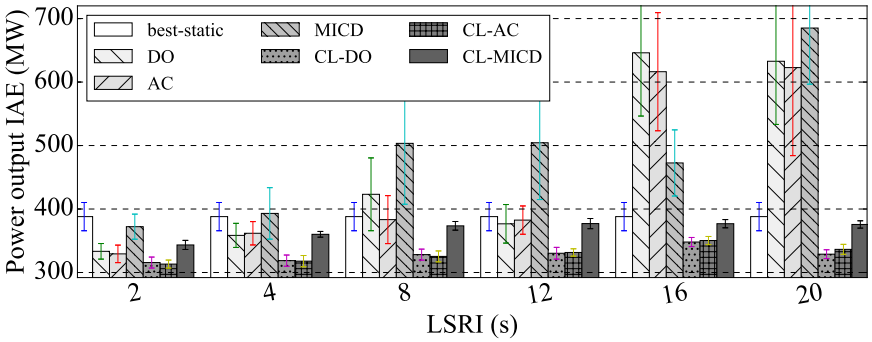
To consider both control system performance and network energy consumption together, we normalize network lifetime by IAE (i.e., network lifetime / IAE; the more normalized value, the better) in Figure 32(c) for different average RSSI values. The static scheme is significantly worse than the dynamic algorithms, because it consumes the most network energy consumption, and it causes the most power output IAE, demonstrating that our reconfiguration algorithms are necessary and work well. Note that we selected the best-static scheme to be conservative in our evaluation, but in reality it would be hard to select

a good static configuration *a priori*, since the network interference is unpredictable.

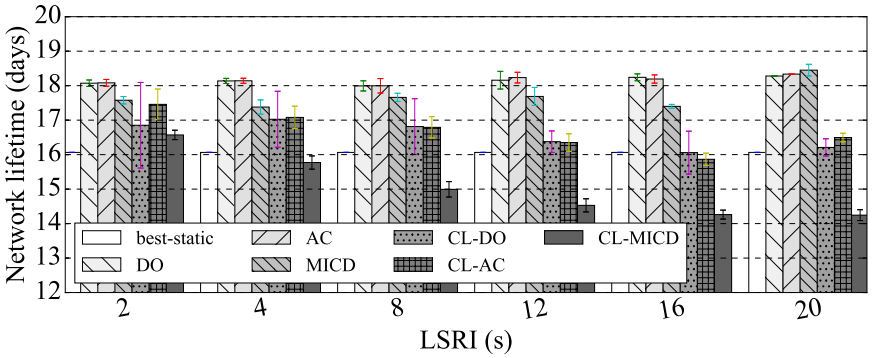
Sensitivity Analysis of LSR Estimation Interval Since LSR is estimated periodically, the length of the LSR estimation interval (LSRI) will affect the control system performance. Figure 35(a) shows the results of the power output IAE for different LSRI values. When the LSRI increases, the IAE of algorithms DO, MICD and AC increases because our estimation is less accurate at high LSRI values. Figure 35(b) and Figure 35(c) show the network lifetime and network lifetime normalized by IAE, respectively. In Figure 36, the green line is the real LSR; the black line (LSRI of 2s) in Figure 36(a) tracks the real LSR better than the LSRI of 8s in Figure 36(b) and the LSRI of 16s in Figure 36(c). Therefore, the control system performance has less error when the LSR estimation is accurate. Figure 37 shows the comparison of the DO with LSRI of 2s and 20s (AC and MICD have similar results). From sample 600 to 800, the DO with LSRI of 20s runs with 30 nodes in the network because the LSR is estimated high averaged over the last 200 samples (from 400 to 600). However, from sample 600 to 800, the LSR is low (network has more interference) and 30 nodes cannot handle the link failures, which makes the consecutive message losses happen (induced delay D is high) and negatively affects the control system performance. The IAEs of the CL-* algorithms are not affected by the LSRI values because, even though the LSR estimation may not be accurate, CL-* algorithms activate additional nodes to compensate to make the network robust. However, the side-effect is that CL-* algorithms consume more energy (see Figure 35(b)). For network performance (DR and delay), see Figure 38(a) and Figure 38(b).

Adaptive control algorithm with different α values

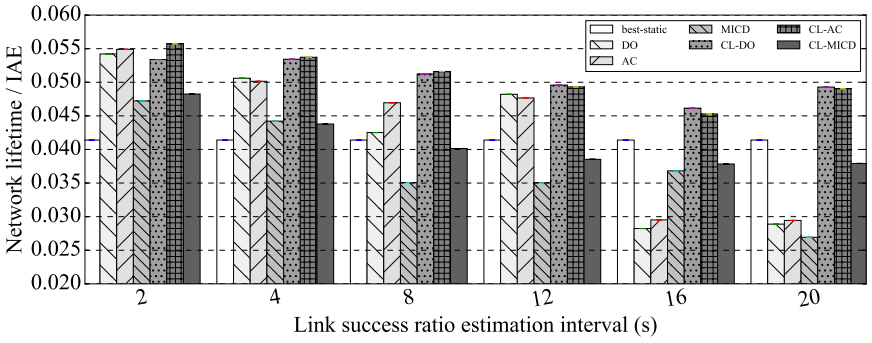
Recall that the AC algorithm has a variable α ($0 < \alpha < 1$), which determines the speed to activate or deactivate nodes in the network (small α, fast node activating/deactivating). The α value can also affect the control system performance. Figure 39 shows the IAE of AC and CL-AC algorithms for different α values. First, looking at the results without consecutive losses, when $\alpha > 0.5$, the control system performs worse. This is because the speed of adding or removing nodes is slow that it cannot react to the LSR variation on time. Figure 40 shows the reason more clearly. From sample 600 to 800, when the network has more interference, the speed of activating nodes in AC with $\alpha = 0.9$ is slower than α=0.1, causing consecutive



(a)

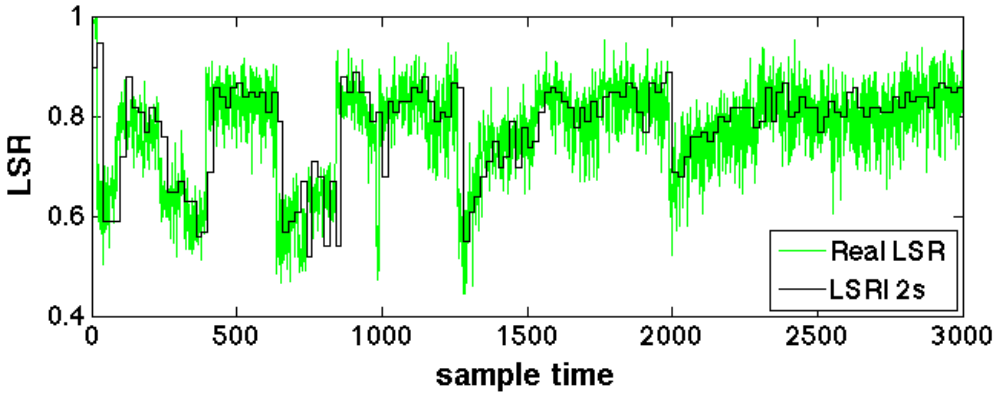


(b)

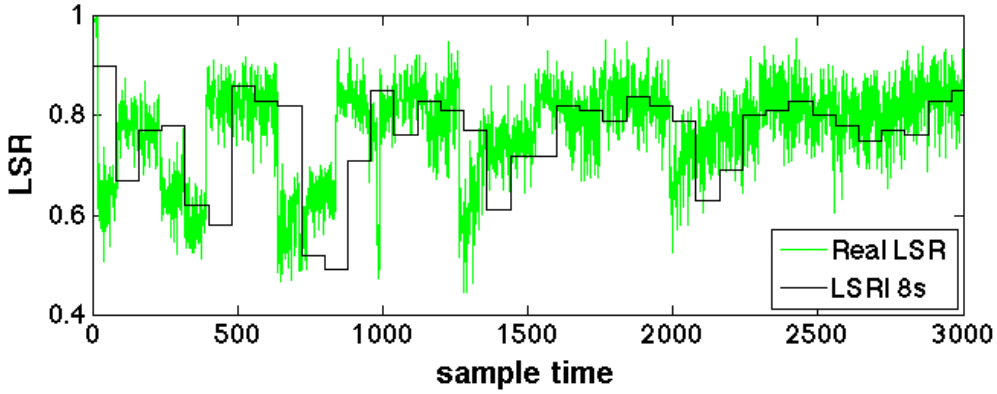


(c)

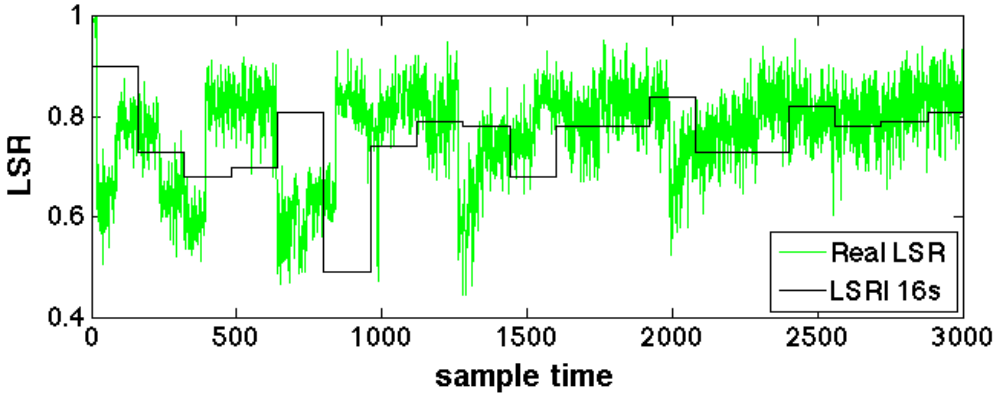
Figure 35: (a) Power output IAE and (b) network lifetime (c) network lifetime / IAE results for different LSRIs (average RSSI: -82dBm; α: 0.1)



(a)



(b)



(c)

Figure 36: Comparison of estimated and real LSRs (a) LSRI is 2s (b) LSRI is 8s (c) LSRI is 16s (average RSSI: -82dBm)

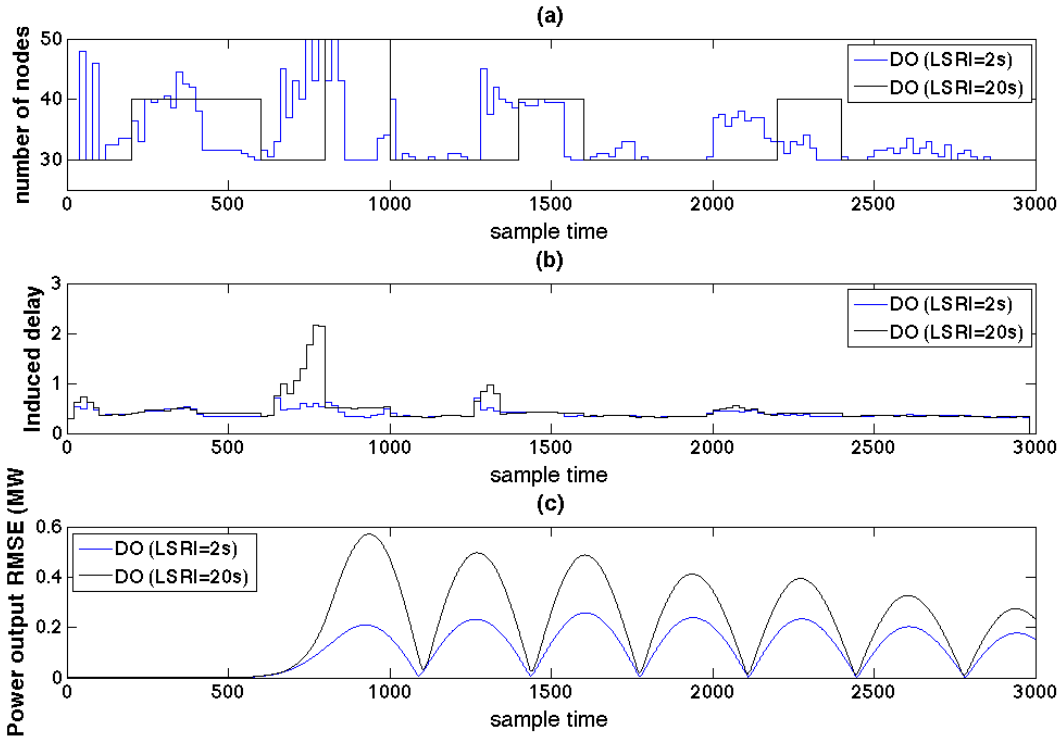
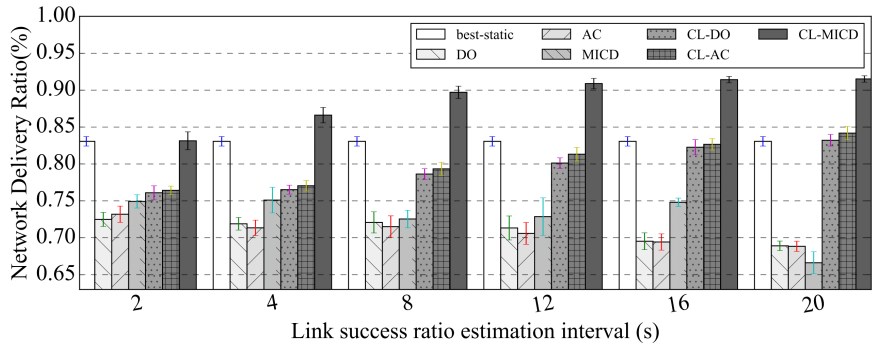
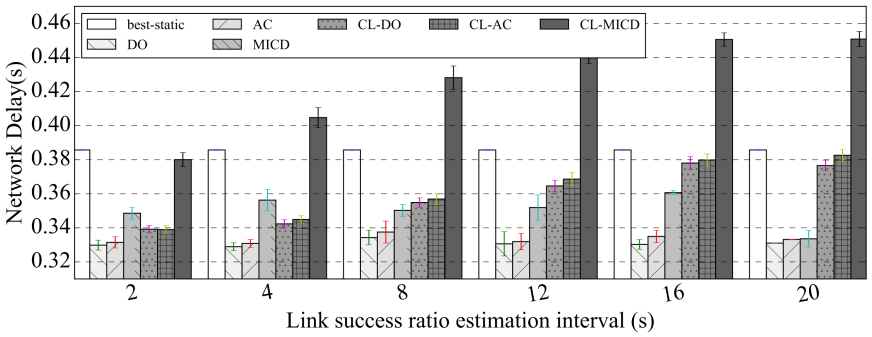


Figure 37: (a) Average number of nodes in the network, (b) average induced delay and (c) average RMSE over 20 experiments changing over time (average RSSI: -82dBm)



(a)



(b)

Figure 38: (a) Network delivery ratio; (b) network delay for different LSRIs (the average RSSI value: 82dBm; α: 0.1)

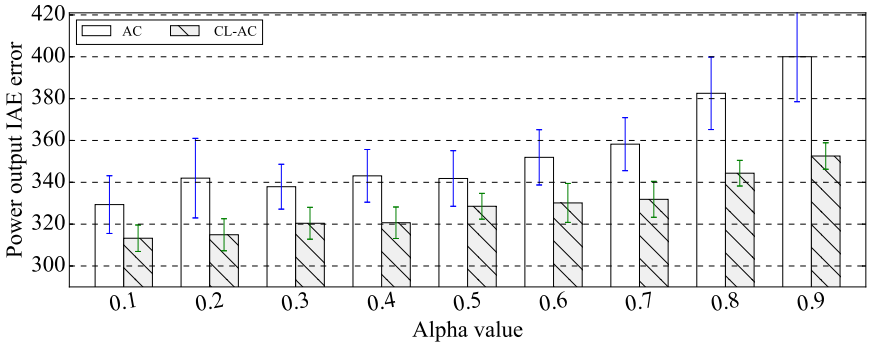


Figure 39: Power output IAE result comparison of AC and CL-AC for different alpha values (average RSSI: -82dBm; LSRI: 2s)

message losses and more induced delay. From sample 800 to 1300, when the network has less interference, the speed of AC (α=0.9) of deactivating nodes is also slow and induce more delay (network delay is high) into the control system.

When considering consecutive losses, from Figure 39, we find that CL-AC always performs better than AC. Although the speed to activate or deactivate nodes is slow for AC with α=0.9, considering consecutive losses can compensate with 13.5% IAE reduction. Figure 40 shows more details. From sample 600 to 800, when the network has more interference (consecutive message losses happen), CL-AC (α=0.9) activates more nodes in the network than AC (α=0.9), which improves the control system performance. But CL-AC consumes more network energy than AC (see Figure 39), due to activating additional nodes when there are consecutive losses.

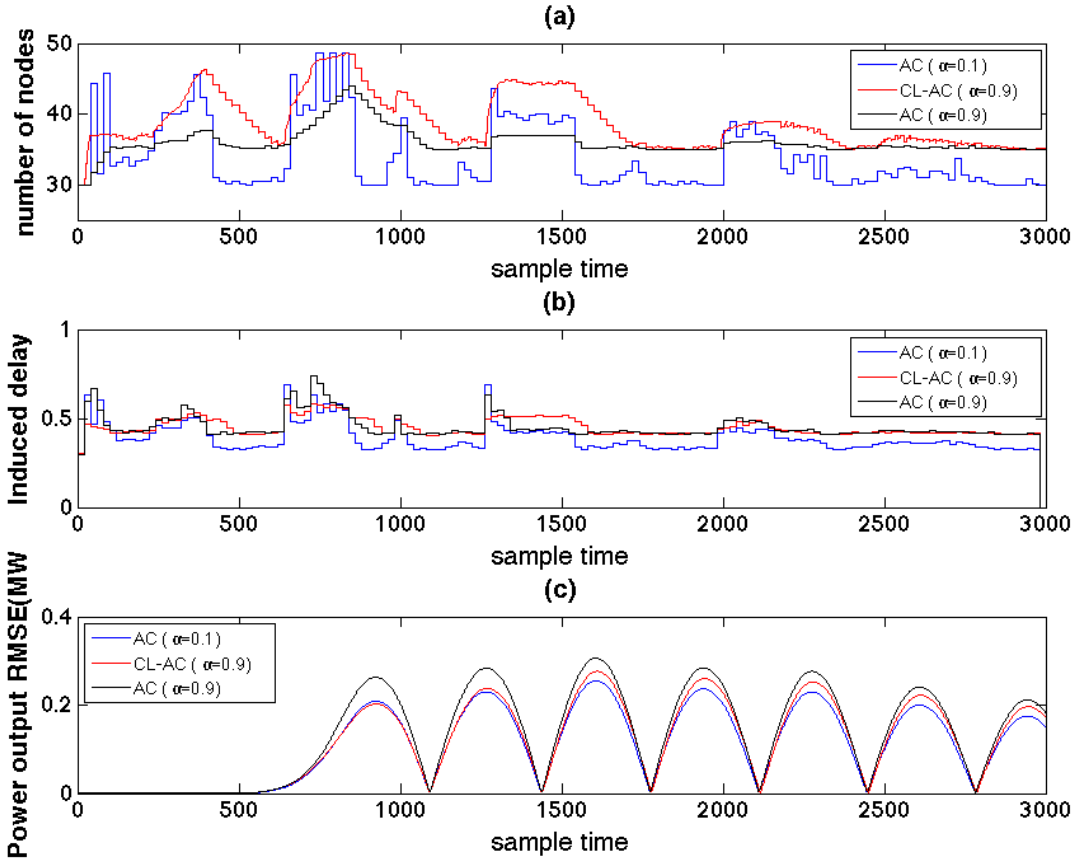


Figure 40: (a) Average number of nodes in the network, (b) average induced delay and (c) average RMSE over time for AC (α=0.1), CL-AC (α=0.9) and AC (α=0.9) (average RSSI: -82dBm; LSRI: 2s)

6.6 Summary

In this chapter, we focus on the objective of reducing the network-induced error (i.e., RMSE and IAE) of a WCS with one physical system, as the LSR changes over time. We demonstrate that network reconfiguration is capable of achieving this goal. To assess the performance of our proposed network reconfiguration framework with offline and online parts, a systematic case study is conducted to see the in-depth interaction between the network reconfiguration and the control. The simulation results show that our network imperfection model is accurate with Pearson correlation 0.993, that network reconfiguration works better than the static scheme showing low error and longer network lifetime. Furthermore, we find that consecutive message losses can degrade the control system performance, but online algorithms can compensate for them dynamically.

7.0 Dynamic Packet Assignment for WCS with Multiple Physical Systems

In Chapter 6, we introduce the network-induced error reduction for a WCS with one single physical system. However, the situation of multiple physical systems utilizing one shared wireless network will be common, especially in IoT (Internet of Things) systems and IIoT (Industrial IoT). It is necessary to study the control system performance improvement for a WCS with multiple physical systems, which is the focus of this chapter. To the best of our knowledge, it is the first study on dynamic packet assignment for a WCS with multiple physical systems.

7.1 Introduction

We consider a WCS of multiple control systems with one shared wireless network, controlled by a centralized remote controller by assuming all the control systems are stable (i.e., the stability requirement is satisfied). In a shared network, a real-time wireless network typically has multiple network paths to transmit messages in parallel (some paths may have redundancy). A network path is defined as the routing path, that is, the nodes along the path transmitting the messages of a physical system from the source to the destination. Two different paths can share common nodes to do message transmission. The network paths are frequency separated (transmit messages in different channels). Note that in this chapter, the number of nodes per path is fixed. Each path may have different characteristic in terms of delay and reliability (e.g., in WirelessHart Protocol [wir, 2007], one can choose between more reliable and higher delay and lower delay but less reliable paths, which refer to graph routing and source routing, respectively). Also, different control systems may have different application demands. For example, one control system has urgent demand, such as reducing temperature by 10 °C within 10 minutes in a room while another system has less urgent demand, such as increasing the temperature by 2 °C within one hour.

Our solution follows our intuition: to get better overall control system performance, we

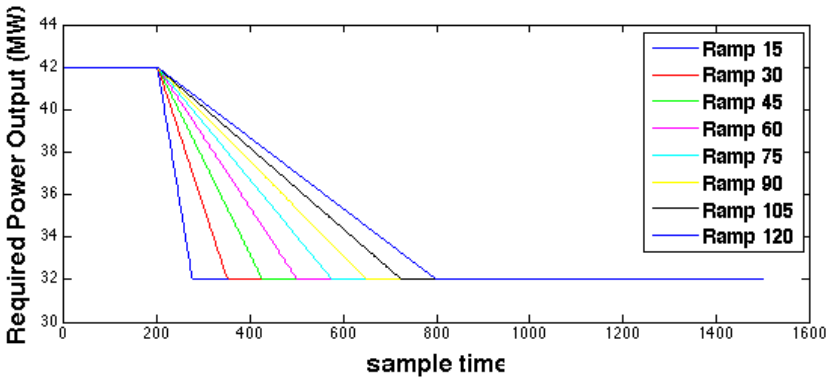


Figure 41: Control system power reference functions with control sampling period of 0.2s

should assign the messages of the control system with the urgent demand to fast and reliable paths and assign the messages with the less urgent demand to slower or less reliable paths. To test our intuition, we simulate a PHX system in Simulink. Figure 41 shows 8 different reference functions (ramp functions) of a PHX when the controller decides to reduce the output power from 42MW to 32MW within different amount of time. For example, ramp30 means to reduce the power from 42MW to 32MW within 30s. The control system application demand urgency order of the 8 reference functions is ramp15 > ramp30 > ramp45 > ramp60 > ramp75 > ramp90 > ramp105 > ramp120.

To motivate how important packet loss and delay are to different control system application demands, we inject packet losses and time delay into the PHX system. We assume in this chapter the DR from measurement sensors to the remote controller (i.e., sensing) is the same as the DR from the remote controller to the actuator (i.e., actuation), since we apply the same network routing scheme for both sensing and actuation in this dissertation. We inject random packet drop with the same probability of (1-DR) and inject the same delay for both sensing and actuation. Note that the DR in this chapter refers to the half-way network reliability (i.e., sensing or actuation), but the network delay refers to the total delay for both sensing and actuation. We measure system performance through power RMSE (the same metric used in the case study in Chapter 6).

Figure 42 shows the effect of network delays and power output reference functions (from

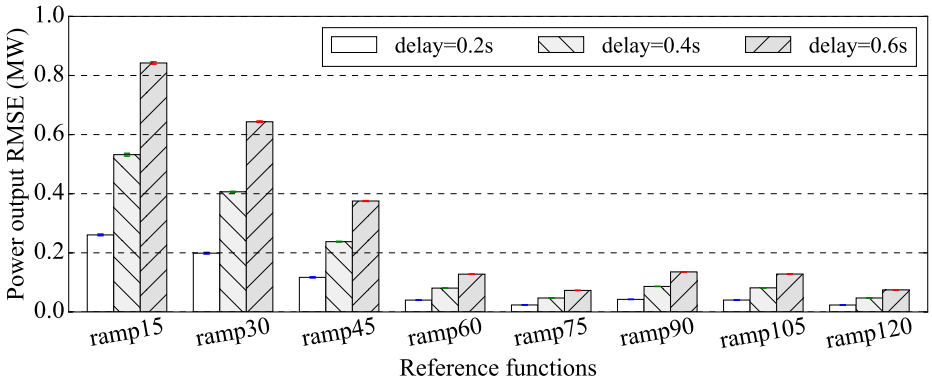


Figure 42: Power output RMSE for different reference functions with different network delays for a single PHX (DR=0.9; random packet drop with probability of 0.1)

Figure 41) with DR=0.9 (other values of DR show similar trends of the RMSE). Figure 43 shows the power output RMSE with different network delays and DRs when the reference function is ramp30 (similar trends for the other reference functions). We have two observations: (1) As shown in Figure 42, for the same network delay and DR, the steeper the reference function, the larger the RMSE. This is because when the reference function is steep, it requires the control system to reduce its power output aggressively (in much less time), and thus it will have a more transient response, causing larger RMSE. However, if the time required to change the power output is longer than 60 seconds (i.e., ramp60), the control system has approximately the same error due to the slow reaction required by the NPP. (2) As shown in Figure 43, for the same reference function, the higher the network delay and lower DR, the larger the RMSE. For the same control system application demand, different network delay and delivery ratio can lead to different control system performance.

Based on the two observations above, network imperfections will impact each control system differently, depending on the control system's application demand (e.g., a reference function in case of a PHX). Thus, our goal is to reduce the overall control system RMSE caused by network-induced imperfections. We propose an approach to dynamically assign packets of different physical systems to the appropriate network paths (with redundancy or not). Our approach has two parts: (1) priority determination of the packets of different

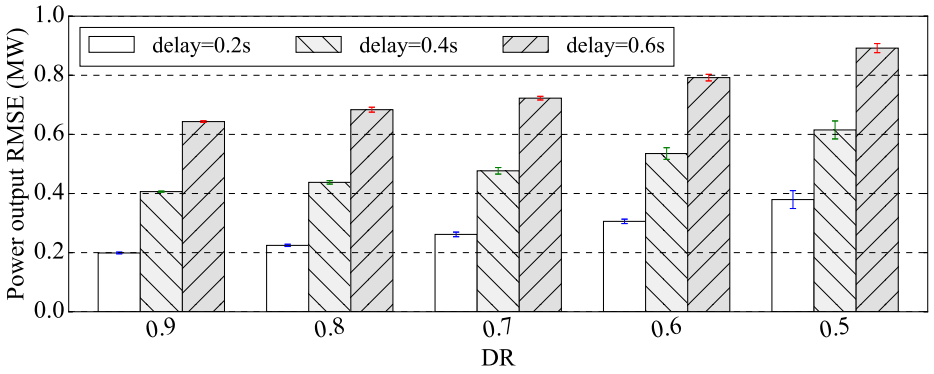


Figure 43: Power output RMSE with different network delays and DRs for a single PHX (reference function: ramp30)

physical systems (highest priority for the most urgent physical plant and the priorities over the physical plants could change dynamically). Each packet of the physical system transmit packets periodically every control sampling period (2) network path selection. For the second part, we study two cases: (2a) **consider network delay only** Based on the worst-case end-to-end delay analysis in Chapter 5, we assign the highest priority packets to the fastest network path. (2b) **consider both network delay and packet loss** We propose a more general model to describe the network path quality combining the impact of network delay and packet loss on the control systems together, based on the network imperfection model proposed in Section 6.2.1. Quality here is from the perspective of the control system: higher quality brings higher performance (smaller RMSE) to the control system. After all, the highest priority packet is assigned to the highest quality path. To evaluate our approach, we first carried out a case study on three PHXs in a modern, SMR (Small Modular Reactor)-based NPP. Note that our approach is general and can be applied to other WCSs. The results demonstrate that our packet assignment approach is effective and able to compensate for delay and packet loss incurred by the network during the transition between steady states of multiple physical systems when they vary their demands simultaneously. This approach is able to create a WCS with the performance close to a wired network.

$$RMSE_i = \sqrt{\frac{1}{w}\sum_{j=0}^{w}(wired_i(t_j) - wireless_i(t_j))^2} \tag{7.1}$$

7.2 Problem Formulation and Solution

7.2.1 Problem Formulation

There are N physical systems that share one wireless network. We define a series of time steps $T = \{t_0, t_1, ..., t_w\}$, where T is the interval of w time steps and the time between two consecutive time steps is the control sampling period. We assume all physical systems have the same control sampling period in our dissertation. During T any physical system is in transition (the system is in non-steady state). We have a set of N reference functions $R = \{r_1(T), r_2(T), .., r_N(T)\}$ that define different physical system application demands. Similar to [Saifullah et al., 2010], there are e choices of network paths $P = \{path_1, path_2, ..., path_e\}$ $(e \geq N)$, each path associated with a different delay and delivery ratio, which depends on the redundancy in the path as well as the scheduling and routing scheme. In this chapter, each network path delivers one message with the measurements of one physical system to the remote controller and delivers back one message with the associated control signal to the actuator periodically (for the situation of physical systems sharing one network path is out of scope of this work). In order to quantify the network-induced error, for each physical system i, we compute $RMSE_i$, defined in Equation 7.1, where $wired_i(t_j)$ and $wireless_i(t_j)$ are the wired (no losses, no delay) and proposed wireless control system power output of physical system i at time step t_j, respectively. Note that $RMSE_i$ is the root mean square of power output of physical system i over w time steps. Our objective is to minimize the $RMSE_{avg}$, defined in Equation 7.2. Our scheme produces the network path selection for physical systems over all time steps. $PS = \{[ps_1(t_0), ps_2(t_0), ..., ps_N(t_0)],$

$$RMSE_{avg} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} RMSE_i^2} \tag{7.2}$$

$[ps_1(t_1), ps_2(t_1), ..., ps_N(t_1)]$, ..., $[ps_1(t_w), ps_2(t_w), ..., ps_N(t_w)]\}$, where $ps_i(t_j)$ is the selected network path number for the i^{th} physical system transmission at time t_j. Note that RMSE of each physical system can change from one time step to the next, thus necessitates recalculation of the path selection.

7.2.2 Solution Overview

In essence, our solution is to determine which network path to transfer which physical system's measurement over T (i.e., PS) to achieve the objective of $RMSE_{avg}$ minimization, at the side of the centralized remote controller. Let us first assume that the packet loss and network delay on all network paths are predictable. We consider a brute-force way to solve the problem. At each time step, we try all possible combinations of network paths $C(e, N)$ and choose the best path selection, that is, PS that has minimum $RMSE_{avg}$ over w time steps. The complexity of our problem is $O(C(e, N)^w)$, which is exponential. Even if we assume the network is predictable, it is impractical due to its high computation time and storage costs. When we consider the realistic case that network delay and loss are unpredictable, the optimal solution does not exist. Therefore, we need solutions to make decisions at run time.

We propose to solve the problem in two steps. We first propose three heuristic methods to determine which physical system has the most urgent application demand and impose a priority order for the packets (Section 7.3). We then study two cases: (1) considering network delay only: based on the analysis of the worst-case end-to-end network delay, we assign the most urgent packet to the network path with the shortest delay (Section 7.4); (2) considering both network delay and packet losses: we propose a network path quality model

to consider both the end-to-end delay and reliability of a network path. We assign the most urgent packet to the network path that can deliver the measurement as high reliability and as short delay as possible (the highest quality path) to result in small $RMSE_{avg}$ (Section 7.4). Note that both the packet priority and the path quality calculation are done in the remote controller.

7.3 Packet Priority Determination

The basic idea of priority determination of packets is to give high priority to the packet of the system that would yield low performance, to avoid increasing RMSE and thus $RMSE_{avg}$. To determine the packet priority, we propose three heuristic methods with different perspectives. For each heuristic method, we propose a metric to calculate the urgency of the packets, we then sort the urgency for each packet and get the packet priority (low urgency, low priority).

7.3.1 Static RMSE

Similar to the analysis in Figure 42, we carried out a thorough offline analysis for each physical system for all possible reference functions (e.g., different slopes of the ramp functions) by injecting the same amount of time delay but no packet loss into the control system. Thus, for each control system, we can get a list of reference functions, each with a RMSE result over the same period of time (when the system is in non-steady state). According to the offline analysis, we can estimate each physical system performance (meaning the RMSE result with the same reference setting, e.g., the same ramp function). This heuristic gives the highest priority to the packets of physical system with the highest estimated RMSE obtained from the offline analysis. But the priority determination of packets is fixed and is not dynamically changed at run time. Note that static RMSE is the baseline of our packet priority determination methods.

$$rRMSE_i(t_x) = \sqrt{\frac{1}{x}\sum_{j=0}^{x}(r_i(t_j) - wireless_i(t_j))^2} \tag{7.3}$$

7.3.2 Dynamic RMSE

Since our objective is to minimize the control system $RMSE_{avg}$, the heuristic is based on the following: the higher RMSE, the more necessary to transmit its message as soon and reliably as possible (thus reducing the RMSE). Since we cannot get the RMSE comparing with the wired control system output at run time, we track each system's rRMSE (for presentation purposes), that is, RMSE comparing with its reference function at run time. Equation 7.3 shows the rRMSE of i^{th} physical system, $rRMSE_i(t_x)$ comparing with its reference function r_i from time step t_0 to t_x. At current time step t_x, we calculate rRMSE of each physical system at the remote controller (the system output $wireless_i(t_x)$ needs to sent with the measurements to the remote controller), sort the rRMSEs of N physical systems and assign the highest priority to the packet of the system with the highest current rRMSE, $\max_{1 \le i \le N} rRMSE_i(t_x)$.

7.3.3 PID

Our third heuristic method is inspired by the PID feedback control loop mechanism. We determine a proportional term (P-term) as $K_p e_i(t_x)$, where K_p is a constant and $e_i(t_x)$ is the difference between the i^{th} physical system output at time t_x and the desired setpoint $r_i(t_x), i \in N$; in our case, the setpoint is determined by the reference function. The P-term describes how far the current system performance is from what it should be (i.e., the reference function). We define the integral term (I-term) as $K_i \sum_{j=0}^{x} e_i(t_j)$, where K_i is a constant and $\sum_{j=0}^{x} e_i(t_j)$ is the integral error from time t_0 to t_x. The I-term denotes the overall system performance from the beginning. The D-term is defined as $K_d(e_i(t_x) - e_i(t_x - 1))$, where

$$pid_i(t_x) = K_p\ e_i(t_x) + K_i \sum_{j=1}^{x} e_i(t_j) + K_d\ (e_i(t_x) - e_i(t_x - 1)) \tag{7.4}$$

K_d is a constant. This term approximates the trend of error in the future (e.g., if this term is negative, it means the system error tends to reduce). The $pid_i(t_x)$ function is shown in Equation 7.4. We use $pid_i(t_x)$ to describe the i^{th} system performance and track $pid_i(t_x), i \in N$ for each physical system at run time. We assign the highest priority to the measurement of the physical system with highest $pid_i(t_x)$ value at time t_x. As usual in control systems, since K_p, K_i and K_d are constants, we tune these constants by manual tuning in Section 7.6.

7.4 Network Path Selection

After we determined the priority of the packets, we need to determine which path to transmit the message of which control system. We focus on a wireless network with multiple network paths that can transmit messages in parallel. Each network path has one line of primary nodes and zero or more lines of backup nodes, as shown in Figure 14 in Chapter 5. Thus, each path has different lines of relay nodes and has the different characteristic in terms of network delay and DR. Every control sampling period, each path transmits one message with all the measurements of one physical system up to the controller, and transmits one message with one or more control signals associated to the same physical system back to the actuators. We first consider network delay only for path selection. Based on the worst-case end-to-end delay analysis of the network path in Chapter 5, we assign the highest priority packet to the network path with the smallest worst-case delay.

Now we determine which network path to transmit messages when considering packet losses. Based on the network imperfection model we proposed in Section 6.2.1, we propose a

$$PQ = (\left\lceil \frac{D_{network}}{p} \right\rceil + \beta n_{loss})p \tag{7.5}$$

more general network path quality model, the *PQmodel*, as described by Equation 7.5 where β is a constant, that quantifies how much the network affects the control system. The value β is static during the path selection in this dissertation. Since our PQmodel quantifies the network imperfection impact to the control system, a smaller PQ value means better quality of the network path. Thus, the PQ value is used to dynamically monitor the network path quality. We use β to adjust the importance between network delay and network reliability. When $\beta = 1$, network delay and network reliability have the same importance to the control system performance. β is set according to different WCSs we are dealing with. When the worst-case network delay is smaller than the control system sampling period (e.g., like the water tank system in [Li et al., 2015]), β is set to a very large number since network reliability is the only factor that affects the control system performance. When the control sampling period is smaller than the network delay, a more common scenario, β is a number closer to 1. For instance, when the control system uses kalman filter or any other technique to compensate for message losses, we can reduce the network reliability importance and set β to be small. β also can be adjusted under different network situations for the same control system. We will discuss the value of β under different network situations in Section 7.6.

7.5 Case Study

As shown in Figure 44, we conduct a case study of an NPP with three SMRs (three PHXs, each of which transmits and receives messages via a shared wireless network). Given that there are several SMRs in an NPP, the power output of each SMR may differ and the controller may decide to change the power output of each SMR dynamically, based on

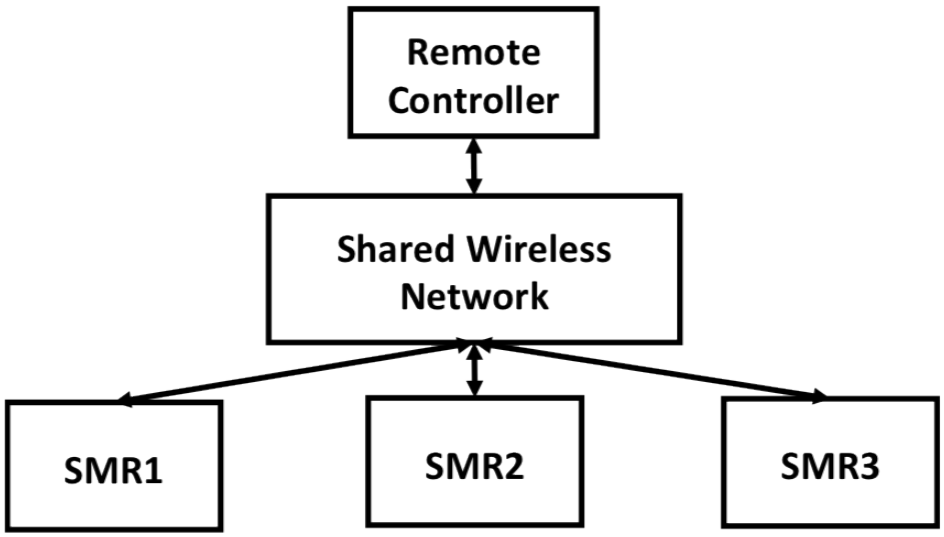


Figure 44: System overview: three SMRs transmit measurement messages via shared wireless network to the remote controller, and the remote controller transmits back control signals backup via the same network

energy requirements, efficiency, and power balance that is required to achieve a certain level of power output. The PHXs in SMRs are identical systems except for the reference functions, which are set by the nuclear engineer/operator based on the NPP requirement. In our case study, a reference function is a ramp function, defined (1) power change amount (PCA) as the amount of power required to change; (2) power change duration (PCD) as the interval of time the power finishes changing; (3) start interval (SI) as the time duration from time 0 to the time the power starts to change. For example, ramp30 in Figure 41 is with PCA=10MW, PCD=30s and SI=40s. The parameters in *a set of reference functions* are 3 PCAs, 3 PCDs and 3 SIs. Each reference function is randomly chosen from the range of values of PCA, PCD and SI listed in Table 6. In order to include all the PCDs, we choose simulation time as 300s, taking into account the system settling time (even after the PCD, the system still needs sometime to settle down to the setpoint). Each PHX will generate one packet (with its three measurements) and send out the packet by wireless network periodically at the sampling period 0.2s.

Table 6: Parameters and values of the simulation of SMR-based NPP

Parameters	Values
Control sampling period	0.2s
Simulation time	300s
TDMA time slot duration	0.01s
PCA values	2MW, 4MW, 6MW, 8MW, 10MW
PCD values	15s, 30s, 45s, 60, 75s, 90s, 105s, 120s
SI	range: [20s, 300s]
β value	range: [0.0 2.0]

Based on the deadline of one PHX system (0.586s [Wang et al., 2016]), we design a wireless network with three paths, each of 6 hops: path 1 ($path_1$) has no backups (worst-case round-trip delay: 0.12s); path 2 ($path_2$) has 1 line of backup nodes (worst-case round-trip delay: 0.3s); path 3 ($path_3$) has 2 lines of backup nodes (worst-case round-trip delay: 0.54s). Each path satisfies the schedulability condition, $\lfloor \frac{p_s}{l} \rfloor \geq 5$ (p_s=20 with control sampling period 0.2 as shown in Table 6). The reliability relationship of the three paths is $path_1 < path_2 < path_3$. Each network path can transmit messages independently from the others, that is, all 3 paths can transmit messages in parallel, without interfering with each other. We apply the most recent message first scheme. We combined a state-of-the-art cyber-physical system simulator (WCPS 2.0 [Li et al., 2015]) with an NPP simulator to mimic the WCS we consider. Our simulator allows multiple wireless network paths running together with multiple PHXs. We implement the heuristic methods proposed in Section 7.3 and the network quality model from Section 7.4 at the remote controller. We use the TOSSIM network simulator in WCPS with wireless noise traces from a 21-node subset of the WUSTL Testbed [tes, 2017]. We controlled the Received Signal Strength with uniform gaps to simulate various wireless signal strength (RSSI) values to change the LSR. As shown in Figure 11, we adjust the RSSI values for the average LSR to be in the range (0.71, 1.0).

7.6 Case Study Results

Based on the wireless control system for the NPP introduced above, we first compare the reliability of the three network paths for different network conditions (Section 7.6.1). We then evaluate our network path quality model (Section 7.6.2). Specifically, we did a sensitivity analysis of β values and analyze the network path selection for different network conditions. Additionally, we compare $RMSE_{avg}$ for both end-to-end delay approach and PQmodel approach (Section 7.6.3). Finally, we compare the $RMSE_{avg}$ among the three heuristic methods of packet priority determination (Section 7.6.4).

In order to determine the constants K_p, K_i and K_d of the PID heuristic method, we ran 100 experiments (each experiment corresponds to one set of reference functions) for each set of constant values of K_p, K_i and K_d with no message loss (to make sure the path quality order is fixed by the network delay. We choose the set of constant values (K_p=1, $K_i = 2\ K_p/t$ and $K_d = 0$) that has the average minimum $RMSE_{avg}$ over all the experiments. Note that K_p, K_i and K_d value may be different with different types of control systems (e.g., K_d=0 in our case, but can be nonzero in other types of WCS).

7.6.1 Network Reliability Results

Figure 45 shows the DR of three network paths under different RSSI values. The DR increases as the number of backup paths increases. Since $path_1$ has no backup path, the DR is only about 0.6 when the RSSI value is -64 (good network condition). At the other extreme, the DR of $path_3$ (two backup paths) is above 0.8 with the RSSI value -84 (poor network conditions).

The percentage of the number of consecutive packet losses for paths $path_1$, $path_2$ and $path_3$ are presented in Figures 46, 47 and 48 respectively. As expected, for the same network interference condition, the more backup paths in the network, the fewer number of consecutive losses. For each network path, as interference in the network increases (less RSSI value), the percentage of massive consecutive loss ($nloss \geq 6$) increases (e.g., the topmost region is much larger for -84 than for -60 in Figure 46).

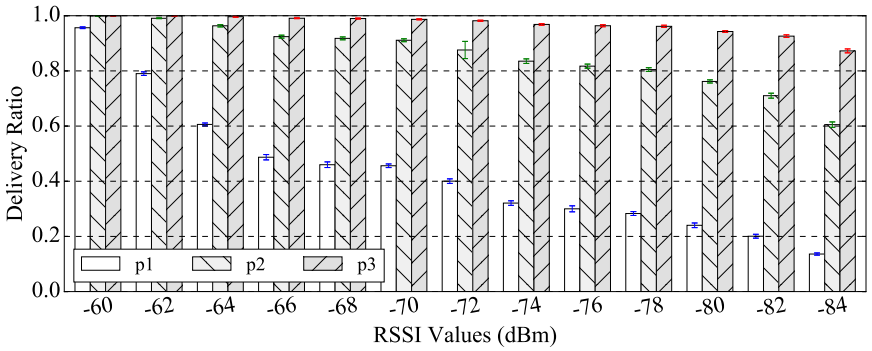


Figure 45: Delivery ratio of three network paths under different RSSI values

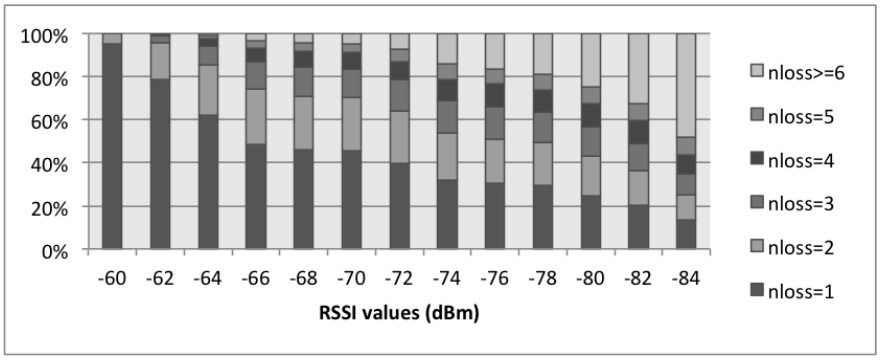


Figure 46: Percentage of consecutive losses (nloss) for network path 1

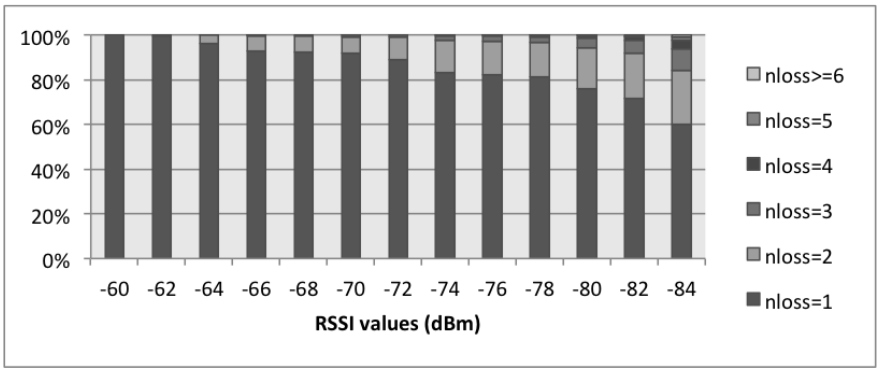


Figure 47: Percentage of consecutive losses (nloss) for network path 2

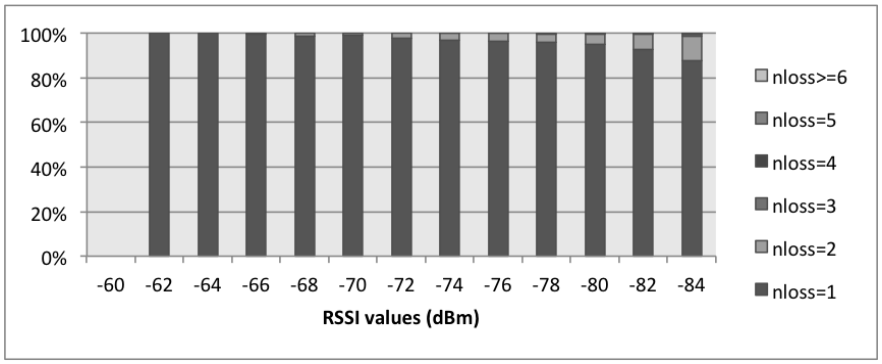


Figure 48: Percentage of consecutive losses (nloss) for network path 3

7.6.2 PQmodel Approach Results

Sensitivity analysis of β value for network path quality To evaluate the network quality model proposed in Section 7.4, we experiment with different β values from 0.1 to 2.0 for the three heuristic methods proposed in Section 7.3 over different RSSI values on 20 sets of reference functions. We ran each set of reference function 20 times on the network paths given the RSSI value. Figure 49 shows the value of β (i.e., best β) that makes the average minimum $RMSE_{avg}$ over 20 sets of reference functions for the three heuristic methods. For each heuristic method, the value of the best β increases first, then decreases as the interference in the network increases. It is because that when the network has less interference ($RSSI = -60$), all three network paths are very reliable and the n_{loss} is less important than the network delay. When the network has a lot interference ($RSSI = -84$), all paths lose many messages and no path is reliable, so n_{loss} is also not as important as the delay.

Path quality order selection Figure 50 shows the number of selected path orders when the best β is applied in the network quality model and dynamic RMSE heuristic method is applied for different RSSI values (we only show parts of the RSSI values for ease of the presentation, and the trend can be easily seen from the figure; the other heuristic methods have the similar trend). The y-axis shows 6 combinations of path orders, each one corresponding to the descending path quality order. For example, 132 means $path_1$ is the highest quality path; $path_3$ is the mid-quality path and $path_2$ is the lowest quality path.

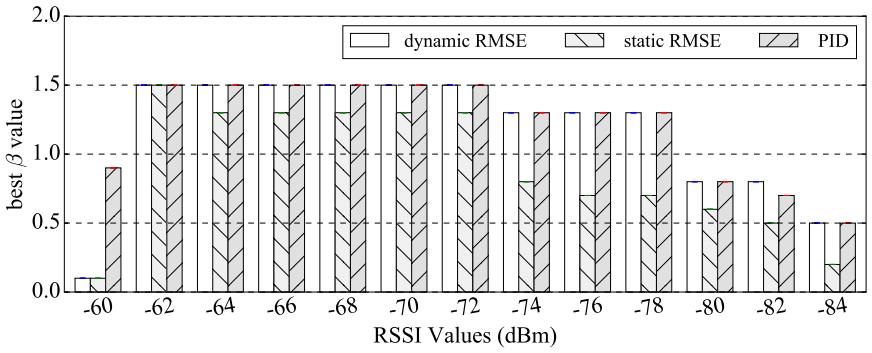


Figure 49: The best β value over different RSSI values for three huristic methods

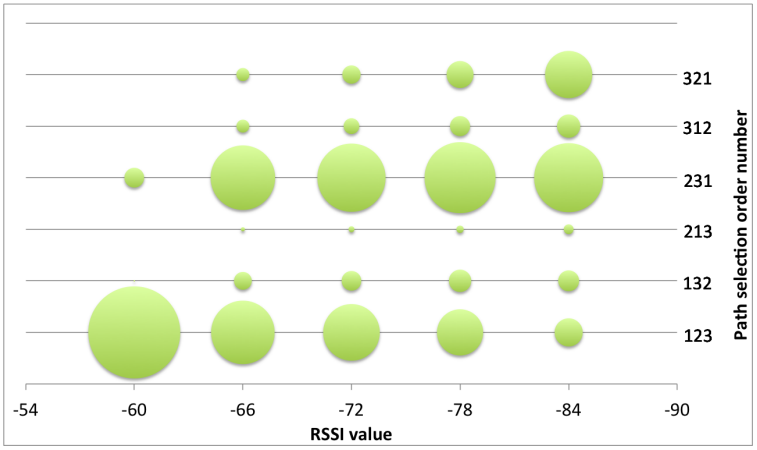


Figure 50: Path quality order selections for different RSSI values (the size of the bubble means the number of time steps a certain path order is selected)

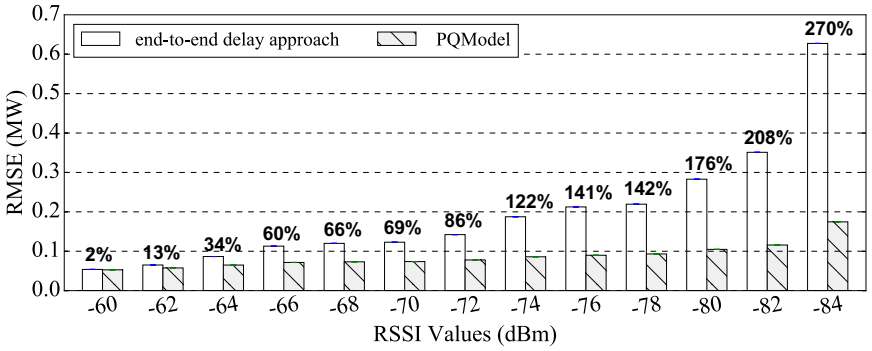


Figure 51: $RMSE_{avg}$ comparison of end-to-end delay approach and PQmodel (best β values) over different network conditions with dynamic RMSE heuristic method

Thus, the highest priority packet goes to path $path_1$; mid-priority packet goes to path $path_3$; and the lowest priority packet goes to path $path_2$. The size of the bubble shows the average number of time steps that the path quality order is selected over the experiments of 20 sets of reference functions. As the RSSI value decreases, the number of path quality order 123 decreases and path quality order 231 increases, since $path_1$ has more packet losses and the quality of $path_1$ decreases. The path quality order 321 and 312 are not high, since $path_3$ has the highest network delay and it will only be selected when the other two paths have too many message losses (i.e., low values of RSSI). Moreover, quality order 213 is also small, because when $path_2$ has the highest quality, it implies the network condition is not good; since $path_3$ has higher reliability than $path_1$, the chance that the quality of $path_1$ is higher than $path_3$ is low.

7.6.3 End-to-end Worst-case Delay Approach and PQmodel Approach Comparison

We evaluate the $RMSE_{avg}$ (defined in Equation 7.2) for end-to-end delay approach and PQmodel approach over 100 different sets of the reference functions for three heuristic methods. For each set of reference functions, we run 20 times on the three wireless network paths for each RSSI value. The average $RMSE_{avg}$ of the dynamic RMSE heuristic method is shown

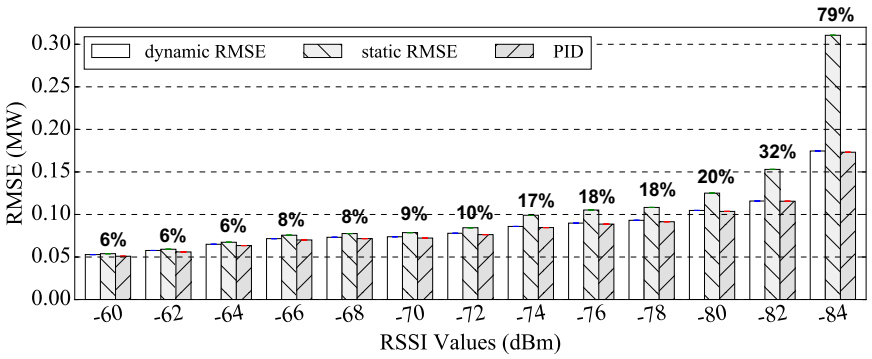


Figure 52: $RMSE_{avg}$ comparison for three heuristic methods with best β value

in Figure 51 (other heuristic methods show the similar results). The PQmodel with the best β value performs better than only considering end-to-end worst-case delay in all network conditions by 2% ($RSSI = -60$) to 259% ($RSSI = -84$). The more interference in the network, the more improvement we can get from the PQmodel because message losses affect more on the control system performance and the PQmodel appropriately characterizes the relationship between network delay and message loss under different network conditions, and thus can more effectively assign the priority of the network paths. The results demonstrate that both network delay and packet loss are key factors for the overall control system performance for less than great networks. Our two-step approach (heuristic method + PQmodel) is effective showing low $RMSE_{avg}$.

7.6.4 Packet Priority Determination Method Comparison

We compare the average $RMSE_{avg}$ for the three heuristic methods on the best β values, as presented in Figure 52. We did pairwise z-test on the results of the dynamic RMSE, static RMSE and PID schemes, since the results are not obvious to compare. We found that PID schemes always perform statistically significantly better than static RMSE scheme (with p value < 0.001) by 5.92% ($RSSI = -60$) to 79.4% ($RSSI = -84$). This is because the packet priority of static RMSE scheme is fixed during the simulation based on offline analysis, which does not consider the cases of multiple systems changing power simultaneously during

a period of time. The static RMSE scheme is not flexible enough to handle the system dynamics, which demonstrates that dynamic packet priority determination is necessary to reduce the overall system error. Moreover, the PID scheme performs statistically significantly better than dynamic RMSE scheme over all network conditions when the RSSI value is greater than -74. Specifically, PID scheme performs better than dynamic RMSE scheme by 2.0% ($RSSI = -72$) to 3.8% ($RSSI = -60$). It is because PID scheme acts like another controller to track the errors generated by three PHXs at run time and can more precisely reflect the priority of the physical systems' packets.

7.7 Summary

In this chapter, we explore the interaction between dynamic packet assignment and the control system performance in a WCS with one shared wireless network and multiple physical systems. Motivated by the observation that network delay and packet loss have different effects on control system performance depending on the system application demand, we propose a dynamic packet assignment solution with the goal of minimizing the overall RMSE (i.e., $RMSE_{avg}$) caused by the network imperfections. Specifically, our solution has two steps: packet priority determination and network path quality determination, which takes account only the network delay first, then proposes a PQmodel considering both network delay and message losses. To evaluate our solution, we carried out a case study on three PHXs in an NPP with one shared wireless network. Our proposed PQmodel performs better than only considering network delay by 2% (for good network conditions) to 259% (for really bad network conditions), which demonstrates that both network delay and reliability play an essential role in control system performance. The results also show that our two-step solution is effective in lowering the total power output error of the nuclear power plant. We also find that dynamic packet priority determination is necessary to reduce the overall system error from the results that PID heuristic method performs statistically significantly better than static RMSE.

8.0 Summary, Lessons Learned and Future Work

8.1 Summary

Wireless control systems are gaining rapid adoption in industries because of its advantages in lowering deployment and maintenance cost in challenging environments. While early success of industrial WSN has been recognized, significant potentials remain in exploring WCS as a unified system to address control system instability and performance issues. We address the issues with fault-tolerance and real-time techniques by meeting the stability requirement first, then reducing the network-induced error.

For the system instability challenge, given the control system stability requirement in terms of network delay and packet loss, we first propose a fault-tolerant network design and a novel model to meet the requirement with the minimum number of active nodes for one-way wireless transmission. The evaluation results show that our model is accurate with average 4.1% difference from the simulation result. We then scale the work mentioned above to two-way wireless transmission to meet the control system stability requirement. We derive the worst-case end-to-end delay based on the maximum number of conflicts, which is calculated by the conflict analysis. The simulation results show that our end-to-end delay analysis is accurate within 4.2% of a realistic simulation result.

For the performance degradation challenge, we first propose a network reconfiguration framework with online and offline parts to tolerate the time-varying link failure for WCS with one physical system. In the offline part, we studied a network imperfection model to quantify the impact of packet loss and network delay on the control system performance. In the online part, we came up with six reconfiguration algorithms. The case study results show that our network imperfection model is accurate with Pearson correlation 0.993 and our network reconfiguration approach performs better than the state-of-the-art static scheme with low network-induced error and low network energy consumption. We then studied a dynamic packet assignment approach to reduce the overall network-induced error for the WCS with multiple physical systems. To achieve the dynamic packet assignment, we first

propose three heuristic methods to assign the priority of network packets; we then study two ways to quantify the quality of network paths; finally, we assign the highest priority packet to the network path with the highest quality. The case study results present our approach is effective in reducing the overall network-induced error.

8.2 Lessons Learned

From this dissertation, we have the listed major lessons learned as follows.

- Our proposed network model is accurate to estimate the network delay and packet loss, which achieves the objective of meeting the control system stability requirement.
- For two-way wireless communication, when the round trip network delay is greater than the control sampling period (assuming control sampling period is the same as network transmission period), message schedulability condition is $\left\lfloor \frac{p_s}{l} \right\rfloor \geq 5$, which is independent of the number of hops n.
- Under the schedulability condition, the worst-case end-to-end delay is derived as $D_{network} = (2nl + 3l\left\lfloor \frac{2nl-3l}{p_s-3l} \right\rfloor)\Delta t$.
- Network reconfiguration works better than the static scheme in reducing the network-induced error and improving the control system performance.
- Both network delay and reliability plan an essential role in the control system performance. Consecutive message losses can degrade the control system performance, but network reconfiguration can compensate for them dynamically.
- Dynamic packet priority determination considering both network delay and reliability is necessary and effective to reduce overall network-induced error for the control system with multiple physical systems.

8.3 Future Work

There are two research directions that can be explored in the future:

In wireless control systems, interference sources can be equipment noise, electromagnetic interference, radio frequency interference (RFI) and fading [Low et al., 2005; Fadel et al., 2015; Chiwewe et al., 2015]. Many empirical studies of low power wireless links have been conducted with various interference sources [Lin et al., 2009; Tang et al., 2007; Hackmann et al., 2008; Guo et al., 2012; Srinivasan et al., 2010], showing temporal and spatial characteristics of wireless link qualities. In this dissertation, we studied the network reconfiguration for the time-varying link failure. It is also necessary to explore the spatial link failures. Many research works [Lapinsky and Easty, 2006; Xu, 2007; Wei et al., 2016; Xu et al., 2006] show that the degree of interference is related to the distance to interference sources: the longer distance, the less interference. Spatial link failures in WCS affect the packet delivery and can even make the network disconnected, thus can severely degrade the control system performance, which is necessary to be explored.

The worst-case end-to-end delay analysis is an important research area in WCS with the propose of meeting the control system deadline. In this dissertation, we studied the analysis for periodically transmitting one measurement message in the network, when the message deadline is possible to be greater than its period. This work can be further extended by considering multiple messages with different periods, under the same condition that each message deadline is possible to be greater than its period.

9.0 Bibliography

Wirelesshart specification. http://www.hartcomm2.org., 2007.

Testbed: http://wan.cse.wustl.edu/index.php/Testbed., 2017.

Control system introduction. Control system introduction: https://www.tutorialspoint.com/control_systems/control_systems_introduction.htm, 2018. Accessed: 2019-01-11.

Tuncer Can Aysal, Mehmet Ercan Yildiz, Anand D Sarwate, and Anna Scaglione. Broadcast gossip algorithms for consensus. *IEEE Transactions on Signal processing*, 57(7):2748–2761, 2009.

Nouha Baccour, Anis Koubâa, Luca Mottola, Marco Antonio Zúñiga, Habib Youssef, Carlo Alberto Boano, and Mário Alves. Radio link quality estimation in wireless sensor networks: A survey. *ACM Transactions on Sensor Networks (TOSN)*, 8(4):34, 2012.

Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530, 2006.

Jonathan L Bredin, Erik D Demaine, MohammadTaghi Hajiaghayi, and Daniela Rus. Deploying sensor networks with guaranteed capacity and fault tolerance. In *MobiHoc 2005*.

Alberto Cerpa, Jennifer L Wong, Miodrag Potkonjak, and Deborah Estrin. Temporal properties of low power wireless links: modeling and implications on multi-hop routing. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 414–425. ACM, 2005.

Dah-Ming Chiu and Raj Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN systems*, 17(1):1–14, 1989.

Tapiwa M Chiwewe, Colman F Mbuya, and Gerhard P Hancke. Using cognitive radio for interference-resistant industrial wireless sensor networks: An overview. *IEEE Transactions on Industrial Informatics*, 11(6):1466–1481, 2015.

Etimad Fadel, Vehbi C Gungor, Laila Nassef, Nadine Akkari, MG Abbas Malik, Suleiman Almasri, and Ian F Akyildiz. A survey on wireless sensor networks for smart grid. *Computer Communications*, 71:22–33, 2015.

András Frank and Éva Tardos. An application of submodular flows. *Linear algebra and its applications*, 114:329–348, 1989.

Deepak Ganesan, Ramesh Govindan, Scott Shenker, and Deborah Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(4):11–25, 2001.

Konstantinos Gatsis, Miroslav Pajic, Alejandro Ribeiro, and George J Pappas. Opportunistic scheduling of control tasks over shared wireless channels. In *Cyber-Physical Systems (ICCPS), 2014 ACM/IEEE International Conference on*, pages 48–59. IEEE, 2014.

Konstantinos Gatsis, Alejandro Ribeiro, and George J Pappas. Control-aware random access communication. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–9. IEEE, 2016.

Sameh Gobriel, Sherif Khattab, Daniel Mossé, José Brustoloni, and Rami Melhem. Ridesharing: Fault tolerant aggregation in sensor networks using corrective actions. In *2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, volume 2, pages 595–604. IEEE, 2006.

Sameh Gobriel, Sherif Khattab, Daniel Mossé, and Rami Melhem. Considering link qualities in fault-tolerant aggregation in wireless sensor networks. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–6. IEEE, 2009a.

Sameh Gobriel, Daniel Mosse, and Robert Cleric. Tdma-asap: Sensor network tdma scheduling with adaptive slot-stealing and parallelism. In *Int'l Conf on Distributed Computing Systems (ICDCS) 2009.*, pages 458–465, 2009b.

Sherrell R Greene, Jess C Gehin, David Eugene Holcomb, Juan J Carbajo, Dan Ilas, Anselmo T Cisneros, Venugopal Koikal Varma, William R Corwin, Dane F Wilson, Graydon L Yoder Jr, et al. Pre-conceptual design of a fluoride-salt-cooled small modular ad-

vanced high-temperature reactor (smahtr). *Oak Ridge National Laboratory, Oak Ridge, TN, Report No. ORNL/TM-2010/199, Fig*, pages 8–1, 2010.

Yu Gu, Tian He, Mingen Lin, and Jinhui Xu. Spatiotemporal delay control for low-duty-cycle sensor networks. In *Real-Time Systems Symposium, 2009, RTSS 2009. 30th IEEE*, pages 127–137. IEEE, 2009.

Arpan Gujarati, Mitra Nasri, and Björn B Brandenburg. Quantifying the resiliency of fail-operational real-time networked control systems. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 106. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

Vehbi C Gungor and Gerhard P Hancke. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *IEEE Transactions on industrial electronics*, 56(10):4258–4265, 2009.

Vehbi C Gungor, Bin Lu, and Gerhard P Hancke. Opportunities and challenges of wireless sensor networks in smart grid. *IEEE transactions on industrial electronics*, 57(10):3557–3564, 2010.

Wenqi Guo, William M Healy, and MengChu Zhou. Impacts of 2.4-ghz ism band interference on ieee 802.15. 4 wireless sensor network reliability in buildings. *IEEE Transactions on Instrumentation and Measurement*, 61(9):2533–2544, 2012.

Rachana Ashok Gupta and Mo-Yuen Chow. Networked control system: overview and research trends. *Industrial Electronics, IEEE Transactions on*, 57(7):2527–2535, 2010.

Gregory Hackmann, Octav Chipara, and Chenyang Lu. Robust topology control for indoor wireless sensor networks. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 57–70. ACM, 2008.

S. Han, X. Zhu, A. Mok, D. Chen, and M. Nixon. Reliable and real-time communication in industrial wireless mesh networks. In *17th IEEE Real-Time and Embedded Technology and Applications Symp*, 2011.

Xiaofeng Han, Xiang Cao, Errol L Lloyd, and Chien-Chung Shen. Fault-tolerant relay node

placement in heterogeneous wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 9(5):643–656, 2010.

Junyoung Heo, Jiman Hong, and Yookun Cho. Earq: Energy aware routing for real-time and reliable communication in wireless industrial sensor networks. *IEEE Transactions on Industrial Informatics*, 5(1):3–11, 2009.

Shengyan Hong, Xiaobo Sharon Hu, Tao Gong, and Song Han. On-line data link layer scheduling in wireless networked control systems. In *2015 27th Euromicro Conference on Real-Time Systems*, pages 57–66. IEEE, 2015.

Naira Hovakimyan and Chengyu Cao. *Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation*. SIAM, 2010.

Romain Jacob, Marco Zimmerling, Pengcheng Huang, Jan Beutel, and Lothar Thiele. End-to-end real-time guarantees in wireless cyber-physical systems. In *Proceedings 2016 IEEE Real-Time Systems Symposium. RTSS 2016*, pages 167–178. IEEE, 2016.

Farid Jusuf and Endra Joelianto. Stabilization of networked control system with time delay induced by network imperfections. In *ICCSII 2012*.

Gholamreza Kakamanshadi, Savita Gupta, and Sukhwinder Singh. A survey on fault tolerance techniques in wireless sensor networks. In *Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on*, pages 168–173. IEEE, 2015.

Soummya Kar and José MF Moura. Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise. *IEEE Transactions on Signal Processing*, 57(1):355–369, 2009.

Junsung Kim, Karthik Lakshmanan, and Ragunathan Raj Rajkumar. Rhythmic tasks: A new task model with continually varying periods for cyber-physical systems. In *Int't Conference on Cyber-Physical Systems*, pages 55–64, 2012.

Kyoung-Dae Kim and PR Kumar. The importance, design and implementation of a middleware for networked control systems. In *Networked Control Systems*, pages 1–29. Springer, 2010.

Stephen E Lapinsky and Anthony C Easty. Electromagnetic interference in critical care. *Journal of critical care*, 21(3):267–270, 2006.

HyungJune Lee, Alberto Cerpa, and Philip Levis. Improving wireless simulation through noise modeling. In *2007 6th International Symposium on Information Processing in Sensor Networks*, pages 21–30. IEEE, 2007.

Philip Levis, Nelson Lee, Matt Welsh, and David Culler. Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 126–137. ACM, 2003.

Bo Li. *Cyber-Physical Co-Design of Wireless Control Systems*. PhD thesis, Washington University in St. Louis, 2015.

Bo Li, Lanshun Nie, Chengjie Wu, Humberto Gonzalez, and Chenyang Lu. Incorporating emergency alarms in reliable wireless process control. In *Int'l Conference on Cyber-Physical Systems*, pages 218–227. ACM, 2015.

Bo Li, Yehan Ma, Tyler Westenbroek, Chengjie Wu, Humberto Gonzalez, and Chenyang Lu. Wireless routing and control: a cyber-physical case study. In *ACM/IEEE International Conference on Cyber-Physical Systems*, 2016.

Ning Li and Jennifer C Hou. Flss: a fault-tolerant topology control algorithm for wireless networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 275–286. ACM, 2004.

Xiang-Yang Li, Peng-Jun Wan, Yu Wang, and Chih-Wei Yi. Fault tolerant deployment and topology control in wireless networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 117–128. ACM, 2003.

Shan Lin, Gang Zhou, Kamin Whitehouse, Yafeng Wu, John A Stankovic, and Tian He. Towards stable network performance in wireless sensor networks. In *Real-Time Systems Symposium, 2009, RTSS 2009. 30th IEEE*, pages 227–237. IEEE, 2009.

Ke Liu, Nael Abu-Ghazaleh, and K-D Kang. Jits: Just-in-time scheduling for real-time

sensor data dissemination. In *Pervasive Computing and Communications, 2006. PerCom 2006. Fourth Annual IEEE International Conference on*, pages 5–pp. IEEE, 2006.

Kay Soon Low, Win Nu Nu Win, and Meng Joo Er. Wireless sensor networks for industrial environments. In *Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, volume 2, pages 271–276. IEEE, 2005.

Ajay Mahimkar and Theodore S Rappaport. Securedav: A secure data aggregation and verification protocol for sensor networks. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, volume 4, pages 2175–2179. Citeseer, 2004.

Amit Manjhi, Suman Nath, and Phillip B Gibbons. Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 287–298. ACM, 2005.

Alexandre Mouradian and Isabelle Augé-Blum. Formal verification of real-time wireless sensor networks protocols with realistic radio links. In *Proceedings of the 21st International conference on Real-Time Networks and Systems*, pages 213–222. ACM, 2013.

Suman Nath, Phillip B Gibbons, Srinivasan Seshan, and Zachary Anderson. Synopsis diffusion for robust aggregation in sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 4(2):7, 2008.

Miroslav Pajic, Shreyas Sundaram, George J Pappas, and Rahul Mangharam. Topological conditions for wireless control networks. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pages 2353–2360. IEEE, 2011a.

Miroslav Pajic, Shreyas Sundaram, George J Pappas, and Rahul Mangharam. The wireless control network: A new approach for control over networks. *IEEE Transactions on Automatic Control*, 56(10):2305–2318, 2011b.

Yash Vardhan Pant, Houssam Abbas, Kartik Mohta, Truong X Nghiem, Joseph Devietti, and Rahul Mangharam. Co-design of anytime computation and robust control. In *Real-Time Systems Symposium, 2015 IEEE*, pages 43–52. IEEE, 2015.

Pangun Park, Sinem Coleri Ergen, Carlo Fischione, Chenyang Lu, and Karl Henrik Johansson. Wireless network design for control systems: A survey. *IEEE Communications Surveys & Tutorials*, 20(2):978–1013, 2018.

Ram Ramanathan and Regina Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 404–413. IEEE, 2000.

Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen. Real-time scheduling for wirelesshart networks. In *Real-Time Systems Symposium (RTSS)*, pages 150–159. IEEE, 2010.

Abusayeed Saifullah, You Xu, Chenyang Lu, and Yixin Chen. End-to-end delay analysis for fixed priority scheduling in wirelesshart networks. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2011 17th IEEE*, pages 13–22. IEEE, 2011.

Abusayeed Saifullah, Dolvara Gunatilaka, Paras Tiwari, Mo Sha, Chenyang Lu, Bo Li, Chengjie Wu, and Yixin Chen. Schedulability analysis under graph routing in wirelesshart networks. In *Real-Time Systems Symposium, 2015 IEEE*, pages 165–174. IEEE, 2015.

Yogesh Sankarasubramaniam, Özgür B Akan, and Ian F Akyildiz. Esrt: event-to-sink reliable transport in wireless sensor networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 177–188. ACM, 2003.

Paolo Santi. Topology control in wireless ad hoc and sensor networks. *ACM computing surveys (CSUR)*, 37(2):164–194, 2005.

Kannan Srinivasan, Prabal Dutta, Arsalan Tavakoli, and Philip Levis. An empirical study of low-power wireless. *ACM Transactions on Sensor Networks (TOSN)*, 6(2):16, 2010.

John A Stankovic, TE Abdelzaher, Chenyang Lu, Lui Sha, and Jennifer C Hou. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91(7):1002–1022, 2003.

Lei Tang, Kuang-Ching Wang, Yong Huang, and Fangming Gu. Channel characterization

and link quality assessment of ieee 802.15. 4-compliant radio for factory environments. *IEEE Transactions on industrial informatics*, 3(2):99–110, 2007.

Wenchen Wang, Christopher D'Angelo, Daniel Mosse, and Daniel Cole. Integrating control and fault-tolerant wireless network design for small modular nuclear reactors. In *Information Reuse and Integration (IRI)*, pages 332–342, 2016.

Wenchen Wang, Daniel Mosse, Daniel Cole, and Jason G Pickel. Work-in-progress: Wireless network reconfiguration for control systems. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE*. IEEE, 2017a.

Wenchen Wang, Daniel Mosse, Jason G Pickel, and Daniel Cole. Work-in-progress: Cross-layer real-time scheduling for wireless control system. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE*. IEEE, 2017b.

Wenchen Wang, Daniel Mosse, and Daniel Cole. Dynamic packet scheduling for wireless control system with multiple physical systems. In *39th IEEE Real-Time Systems Symposium, Under Revision*. IEEE, 2018a.

Wenchen Wang, Daniel Mosse, Daniel Cole, and Jason G Pickel. Dynamic wireless network reconfiguration for control system: A nuclear reactor case study. In *26th International Conference on Real-Time Networks and Systems (RTNS18), Under Revision*. ACM, 2018b.

Xianglin Wei, Tongxiang Wang, Chaogang Tang, and Jianhua Fan. Collaborative mobile jammer tracking in multi-hop wireless network. *Future Generation Computer Systems*, 2016.

Yi-Hung Wei, Quan Leng, Song Han, Aloysius K Mok, Wenlong Zhang, and Masayoshi Tomizuka. Rt-wifi: Real-time high-speed communication protocol for wireless cyber-physical control applications. In *Real-Time Systems Symposium (RTSS), 2013 IEEE 34th*, pages 140–149. IEEE, 2013.

Alec Woo and David E Culler. *Evaluation of efficient link reliability estimators for low-power wireless networks*. Computer Science Division, University of California Oakland, Calif, USA, 2003.

Wenyuan Xu. Channel surfing: defending wireless sensor networks from interference. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pages 499–508. IEEE, 2007.

Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang. Jamming sensor networks: attack and defense strategies. *IEEE network*, 20(3):41–47, 2006.

John Yackovich, Daniel Mosse, Anthony Rowe, and Raj Rajkumar. Making wsn tdma practical: Stealing slots up and down the tree. In *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2011 IEEE 17th International Conference on*, volume 1, pages 41–50. IEEE, 2011.

Li Yang. Building k edge-disjoint spanning trees of minimum total length for isometric data embedding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10): 1680–1683, 2005.

Kan Yu, Zhibo Pang, Mikael Gidlund, Johan Åkerberg, and Mats Björkman. Realflow: Reliable real-time flooding-based routing protocol for industrial wireless sensor networks. *International Journal of Distributed Sensor Networks*, 10(7):936379, 2014.

Jiao Zhang, Fengyuan Ren, Shan Gao, Hongkun Yang, and Chuang Lin. Dynamic routing for data integrity and delay differentiated services in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 14(2):328–343, 2015.

Lixian Zhang, Huijun Gao, and Okyay Kaynak. Network-induced constraints in networked control systemsa survey. *IEEE Transactions on Industrial Informatics*, 9(1):403–416, 2013.

Tianyu Zhang, Tao Gong, Chuancai Gu, Huayi Ji, Song Han, Qingxu Deng, and Xiaobo Sharon Hu. Distributed dynamic packet scheduling for handling disturbances in real-time wireless networks. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE*, pages 261–272. IEEE, 2017.

Wei Zhang, Michael S Branicky, and Stephen M Phillips. Stability of networked control systems. *Control Systems, IEEE*, 21(1):84–99, 2001.

Weiyi Zhang, Guoliang Xue, and Satyajayant Misra. Fault-tolerant relay node placement in wireless sensor networks: Problems and algorithms. In *INFOCOM 2007*.

Wen-An Zhang and Li Yu. Modelling and control of networked control systems with both network-induced delay and packet-dropout. *Automatica*, 44(12):3206–3210, 2008.

Bin Zhou, Lek Heng Ngoh, Bu Sung Lee, and Cheng Peng Fu. A hierarchical scheme for data aggregation in sensor network. In *Networks, 2004.(ICON 2004). Proceedings. 12th IEEE International Conference on*, volume 2, pages 525–529. IEEE, 2004.