MASTER THESIS

MODELING AND PERFORMANCE EVALUATION OF WIRELESSHART

Xian Wu

EWI
DESIGN AND ANALYSIS OF COMMUNICATION SYSTEMS (DACS)

EXAMINATION COMMITTEE
Dr. Anne Remke
Prof. dr. ir. Boudewijn Haverkort

DOCUMENT NUMBER

UNIVERSITY OF TWENTE. 31-AUG-2012
Abstract

In traditional process industries, wired systems are deployed for supervisory and control applications. In recent years, a new tendency to replace the wired system by wireless networks emerged. This migration towards wireless technology can provide the industrial control system with notable advantages on flexibility, installation cost and maintenance. WirelessHART, as the first international standard aiming for process supervisory and control, becomes the main stream of this migration and received notable academic attention. However, wireless communication inevitably introduces time delays and message losses, which may degrade the system reliability and efficiency. Since there is little insight into the performance of WirelessHART, this thesis makes an attempt to model and evaluate this standard.

In this thesis, we model the WirelessHART network using Discrete-Time Markov Chains. The hierarchical model consists of two tiers, i.e. the link layer and the path layer. We derive several measures of interest from the DTMC model to evaluate the network performance in a typical environment from different perspectives. An analysis tool with graphical interface was developed to facilitate such modeling and analysis in an automated manner.

The evaluation shows that although the performance of WirelessHART network is influenced by several factors, it is capable to deliver reliable and satisfactory service under typical industrial environments, even when co-existing with other wireless networks such as IEEE 802.11. As a control system, the stability of control loops is a critical issue. The evaluation provides relevant measures like reachability and delay so that it becomes feasible to judge the stability of a control loop. Moreover, WirelessHART is justified to be robust against transient link failures by the evaluation results. In addition, the proposed model can be used to predict performance and to provide routing suggestion. The modeling approach and evaluation results can be used as reference and suggestion in industrial settings.
Contents

Abstract i

Symbols ix

1 Introduction 1
   1.1 Related Works ............................................. 2
   1.2 Contributions ............................................ 3
   1.3 Outlook .................................................. 4

2 WirelessHART 5
   2.1 WirelessHART Overview ................................... 5
   2.2 WirelssHART Architecture ................................ 7
   2.3 Standard Specifications ................................. 8
      2.3.1 Physical layer ...................................... 8
      2.3.2 Data link layer ..................................... 9
      2.3.3 Network layer ...................................... 11
   2.4 Formal Description ...................................... 11

3 Discrete-Time Markov Chains 15
   3.1 Discrete-Time Markov Chains ........................... 15
   3.2 Binary Symmetric Channel Model ....................... 18
   3.3 Link Model .............................................. 18
   3.4 Failure Transition Probability ......................... 20

4 Hierarchical Path Model 23
   4.1 Control Loop ............................................. 23
      4.1.1 Stability ........................................... 24
      4.1.2 Superframe .......................................... 25
      4.1.3 Reporting interval .................................. 25
      4.1.4 Message life cycle .................................. 26
   4.2 Path Model ................................................ 27
      4.2.1 Modeling target ..................................... 27
      4.2.2 Communication schedule ............................. 28
4.2.3 Path model DTMC ............................................. 28
4.2.4 Hierarchical model ........................................ 30
4.2.5 Model complexity .......................................... 31

5 Path Analysis and Discussion .......................... 33
  5.1 QoS Measures .................................................. 33
  5.2 Methodology ................................................... 35
  5.3 Example Analysis ............................................ 36
  5.4 Analysis Tool ................................................. 39
  5.5 Influential Factors .......................................... 41
    5.5.1 Link availability ...................................... 41
    5.5.2 Path hop number ...................................... 44
    5.5.3 Communication schedule .............................. 45
  5.6 Path Conjunction .......................................... 45

6 Network Performance Evaluation ....................... 47
  6.1 The Typical Network ...................................... 47
  6.2 Analyzer Graphical Interface ............................ 48
  6.3 Overall Performance ...................................... 49
    6.3.1 Reachability ........................................... 49
    6.3.2 Overall delay distribution ........................... 50
    6.3.3 Utilization rate ........................................ 51
  6.4 Scheduling .................................................. 53
  6.5 Stability and Robustness ................................. 55
    6.5.1 Transient error ....................................... 55
    6.5.2 Random failure ........................................ 56
    6.5.3 Permanent link failures ............................... 57
  6.6 Reporting Interval and Fast control ................. 57
  6.7 Performance Prediction ................................... 59

7 Conclusion and Future Work ............................. 63
  7.1 Conclusion .................................................. 63
  7.2 Future Work ................................................ 64

A Program Code of the analyzer .......................... 65
## List of Figures

1.1 WirelessHART products in the plant from Emerson Process Management . . . 2  
2.1 WirelessHART Network Architecture . . . . . . . . . . . . . . . . . . . . . . . . . 7  
2.2 HART protocol stack . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8  
2.3 Frequency Channel and Transmission Power of 802.15.4 and 802.11 . . . . . 9  
2.4 WirelessHART TDMA frame structure . . . . . . . . . . . . . . . . . . . . . . . . . 10  
2.5 WirelessHART data link layer protocol stack . . . . . . . . . . . . . . . . . . . . 10  
2.6 An Example WirelessHART network . . . . . . . . . . . . . . . . . . . . . . . . . . 12  
3.1 Binary Symmetric Channel model . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18  
3.2 Threshold of receiver signal strength and two link states . . . . . . . . . . . . . 19  
3.3 Two-state DTMC link model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19  
4.1 WirelessHART control basic scenario . . . . . . . . . . . . . . . . . . . . . . . . . 24  
4.2 A typical superframe/macrocycle . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25  
4.3 Control signals, sample and reporting interval . . . . . . . . . . . . . . . . . . . . 26  
4.4 An example of path, schedule and transmission . . . . . . . . . . . . . . . . . . . 28  
4.5 DTMC diagram of the path model of a three-hop path when \( I_s = 1 \) . . . . . 30  
4.6 DTMC diagram of the path model of a three-hop path when \( I_s = 2 \) . . . . . 32  
5.1 Delay and jitter of message transmission . . . . . . . . . . . . . . . . . . . . . . . 34  
5.2 Transient probabilities of goal states when \( I_s = 4 \) . . . . . . . . . . . . . . . . 37  
5.3 Delay distribution of the example path . . . . . . . . . . . . . . . . . . . . . . . . 38  
5.4 Analyzer tool block diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 41  
5.5 Influence of link availability on path performance . . . . . . . . . . . . . . . . . 43  
5.6 The influence of path hop number on reachability . . . . . . . . . . . . . . . . . 44  
5.7 Conjunction of an exiting path and a peer path . . . . . . . . . . . . . . . . . . . 45  
6.1 Connectivity graph of the typical WirelessHART network . . . . . . . . . . . . 48  
6.2 Screenshot of the analyzer’s graphical interface . . . . . . . . . . . . . . . . . . . . 49  
6.3 The reachability of all paths in the typical WirelessHART network . . . . . . 50  
6.4 The overall delay distribution of the example WirelessHART network . . . . . 52  
6.5 The expected delays of all paths with the schedule \( a \) . . . . . . . . . . . . . . . . 52  
6.6 The overall delay distribution with schedule \( \eta_b \) . . . . . . . . . . . . . . . . . . 54
6.7 The expected delays with schedule $\eta_a$ and $\eta_b$ ........................................ 54
6.8 Link recovery from a transient failure ......................................................... 56
6.9 Messages reach situation with different reporting intervals ..................... 58
6.10 Reachability probabilities of paths with $I_s = 2$ and $I_s = 4$ ................. 59
6.11 The scenario of a new node joins the network ........................................ 60
List of Tables

2.1 Characteristics of wireless communication technologies .................. 6
5.1 Influence of $\pi(up)$ on the reachability and expected delay .............. 44
6.1 Influence of $\pi(up)$ on the utilization rate of the example network ...... 53
6.2 The reachability probabilities with a random link failure lasting one cycle 57
6.3 Utilization rates of the example network in fast control .................... 58
6.4 Example of performance prediction by path conjunction .................... 60
Symbols

\( D \)  Control loop
\( G \)  Connectivity graph
\( n_i \)  Node i
\( e_i \)  Edge/link i
\( P \)  Transition probability matrix
\( p(0) \)  Initial distribution
\( p(t) \)  Transient distribution
\( \pi \)  Steady-state distribution
\( p \)  Transition probability
\( BER \)  Bit error rate
\( SNR \)  Signal-to-noise ratio
\( I_s \)  Reporting interval
\( F_s \)  Superframe size
\( F_{up} \)  Uplink frame size
\( \eta \)  Communication schedule
\( \mu \)  Individual schedule
\( Re \)  Reachability
\( \tau \)  Delay distribution
\( d \)  Delay
\( E[\tau] \)  Expected delay
\( U \)  Utilization rate
\( f(a) \)  Age probabilities
\( g(x) \)  Cycle probability function
\( N \)  Path hop number
\( \pi(up) \)  Stationary link availability
\( \Gamma \)  Overall delay distribution
\( E[\Gamma] \)  Overall mean delay
Chapter 1

Introduction

The HART (Highway Addressable Remote Transducer) Protocol is a global standard for digital information transmission across wires between smart devices and a control or monitoring system [1]. The HART Protocol was developed in the mid-1980s by Rosemount Inc. originally. In 1993, the registered trademark and all rights in the protocol were transferred to the HART Communication Foundation (HCF), including 37 industrial leaders. Nowadays, approximately 30 million HART devices are installed worldwide [2]. The typical use cases include Inventory Management, Safety Integrity, Cost-Saving Applications. Success practices involve industrial organizations such as Shell Petroleum, MOL Danube Refinery, Mitsubishi Chemical and Appleton Papierfabrik.

However, as the need for process measurements increases, customers seek a more reliable and secure method to deliver measurement to control systems without the need of wires. To meet such demands of wireless control networks, The HART Communication Foundation developed the cutting-edge WirelessHART technology with the legacy of HART technology, so as to continue delivering an operational and satisfactory solution to users.

WirelessHART is the first wireless international standard for process monitoring and control. It is claimed to be Simple, Reliable and Secure [2]. The standard was initiated in early 2004 and developed by HART Communications Foundation (HCF) companies. In April 2010, WirelessHART was approved by the International Electrotechnical Commission (IEC) as a full international standard [3], which makes it the first open wireless communication standard specifically designed for process measurement and control applications. Corresponding WirelessHART products are certified and launched in the market. Figure 1.1 shows WirelessHART products, provided by Emerson Process Management, deployed in control applications.

For the remainder of the introduction, Section 1.1 summaries some important WirelessHART research ideas and results that are related with this thesis. Section 1.2 lists the contribution of this work and compares it with others’. Lastly, Section 1.3 gives an overview of the organisation of this thesis.
1.1 Related Works

Besides the official standard, WirelessHART has been a popular topic in academia. First of all, joint work from American and European scholars [4] proposes a formal syntax and semantics of multi-hop control networks and explicitly translates it into switched systems. Their syntax is widely used in other work, including this thesis. The same group also proposes formal models for analyzing the robustness of control when wireless links suffer disruptions. They further prove the stability under difference failure conditions in [5].

Song and Chen made the first effort to build a WirelessHART prototype according to the standard definition and then implemented a simple WirelessHART network for the purpose of demonstration in [6] and [7]. Due to the possibility to use Time Division Multiple Access, link scheduling for WirelessHART became a critical and interesting issue to pursue global optimization. According to the different understanding and application of WirelessHART features, some scheduling schemes are based on single-channel communication, and focus on making clever use of spatial-reuse to decrease convergecast latency, such as [8] and [9]. Others such as Zhang and Soldati focus on scheduling on multi-channels in depth. Although the channel assignment makes the problem more complex, they establish lower bounds on the number of channels for time-optimal convergecast under different packet buffering capabilities [10], and present a heuristic algorithm for time and channel-optimal convergecast scheduling in reports [11] and [12].

To justify and evaluate the proposed ideas or algorithms, many researchers resort to the simulation of WirelessHART networks. As the bottom layer, the physical layer is fundamental to the network simulations. Tanghe and et al. performed experiments in an industrial environment to study the large-scale fading of industrial radios [13]. Hamida [14] discusses physical layer issues, including link and interference modeling, and then analyzes the influence of the physical layer modeling on the performance and the accuracy of simulations. In the same year, Rousselot [15] and Dominicis [16] made their own efforts to simulate the
1.2 CONTRIBUTIONS

WirelessHART network according to the standard specifications by the common simulator OMNet++ and Mobility Framework. Rousselot simulates small scale network of 2 and 3 nodes with broadcast and unicast traffic in [15]. Dominicis explores the transmission performances when WirelessHART co-exists with IEEE 802.11b (WiFi) networks. While Snickars from the KTH in Sweden, attempts to use an alternative simulation platform. In his thesis [17], a MatLab-based simulator True-Time is implemented to study the sources of delay in WirelessHART control loops, and delay compensation by different types of controllers. All these simulations focus on standard implementation or control theories, but not the network performance.

Some notable efforts towards performance evaluation are summarised as follows. Petersen and Carlson from industry go further into reality. They performed laboratory experiments to study the performance of WirelessHART network co-existing with 802.11 networks [18]. The package loss rates under different parameters are compared. The results demonstrate that WirelessHART is capable of reliable operation in an industrial environment, even when coexisting with IEEE 802.11 networks. Pesonen, also from KTH, models the network message convergecast in Markov chain and analyses it to derive end-to-end probabilities, as a performance measure. This part of work was firstly published on the Emerging Technologies & Factory Automation conference in 2009 [19] and then refined in his master thesis in 2010 [20].

1.2 Contributions

Similarly, we are interested in the performance of WirelessHART networks. Because in WirelessHART, time is slotted and synchronized across all devices, all events such as message transmission happen in discrete time slots. Therefore, it is possible to model the system as a Discrete-Time Markov Chains (DTMC). Markov chain modeling is quick and convenient to implement. In many cases, it appears that simple Markov models are sufficient to capture the key characteristics of observed package delivery in WirelessHART [18]. So, we choose this approach of Markov chain modeling to evaluate performances.

Coincidently, we start along the same path as Pesonen’s work mentioned above, however, manage more complicated situations with a generalized model, and ultimately achieve broad and thorough evaluations. Compared with his preliminary model, this thesis goes much further and deeper in the following aspects:

- We discuss link model parameters and justify the choice according to the WirelessHART characteristics. This is not present in Pesonen’s work.

- Based on the link model, Pesonen only shows the calculation of end-to-end probabilities (similar to ‘Reachability’ in this thesis). While this thesis explicitly proposes a path model including the transition diagram, construction algorithms and etc.

- Pesonen’s model considers only one control cycle, hence, is capable to derive only the end-to-end probabilities. In contrast, our model considers a period that may vary from
1.3 Outlook

a single to multiple cycles, so that it can produce delay distributions across more than one cycle.

- All the link models are homogeneous in Pesonen’s model. In contrast, they can be distinct from each other in our case, which grants the model greater analysis flexibility, as the abstraction of real system.

In one word, Pesonen’s model can be regarded as a special case of a part of our model when all the links are homogeneous and the reporting interval is one. The DTMC model developed in this thesis is more powerful and can deal with multiple variables to produce more measures of interest.

The main contributions of this thesis are as follows:

- The WirelessHART network is modeled using Discrete-Time Markov Chains. According to the standard, a hierarchical path model is proposed, including link failure models. The correlation between the channel model and the link model transitions is developed.

- Three Quality-of-Service (QoS) measures are defined. The methodology to derive such measure of interest from the proposed DTMC model is developed. Both the DTMC model and the analysis algorithms are implemented in an automatic tool with a graphic interface.

- Using the QoS measures that are derived from the DTMC model, this work evaluates the performance of the typical WirelessHART network, including the system stability and robustness, with different influential factors.

- The conjunction of existing paths is studied and applied in performance prediction.

1.3 Outlook

The thesis is organized as follows. Chapter 2 introduces WirelessHART, as the new tendency of Industrial Control System. Its advantages over other technologies are justified by the illustration of the architecture and communication protocols. A formal model to describe the WirelessHART system is introduced and used in this work. Chapter 3 provides the background knowledge of discrete-time Markov chains. After that, a DTMC model is proposed to describe the dynamic states of wireless links. In Chapter 4, firstly the control loops in WirelessHART and related concepts are explained. Secondly, the hierarchical path model is proposed and clarified in detail. Chapter 5 derives the QoS measures including reachability, delay and utilization from the DTMC model, and uses them to analyze example paths. In Chapter 6, we evaluate the typical WirelessHART network, discuss the scheduling principle, assess the system robustness under three types of link failures and show a performance prediction use case. Lastly, Chapter 7 concludes the whole thesis and proposes recommendations for future work.
Chapter 2

WirelessHART

Industrial Control Systems (ICS) are typically used in public and industrial sectors and critical infrastructures such as electricity, water, oil and gas distribution. In these systems, information from remote stations is received by a centralized controller, and then automated or operator-driven supervisory commands can be pushed to remote station control devices, which are often referred to as field devices. These control network operations are generally ‘communicating the necessary sensory and actuation information for closed-loop control’ [21].

As a typical example of ICS, Supervisory Control and Data Acquisition (SCADA) systems [22] are used to monitor and control a plant or equipment in industries such as telecommunications, water control, energy, oil and gas refining and transportation. These networked SCADA systems, as Moyne [21] states, ‘often provide a supervisory-level factory-wide solution for coordination of machine and process diagnostics, along with other factory floor and operations information’. Although traditional ICS widely use Ethernet for system diagnostics and control, and security features under strict scrutiny, Moyne foresaw the trends towards wireless communication in all ICS categories already five year ago [21]. Nowadays such migration is coming true in the context of a mature standard, called WirelessHART [3].

In the following, Section 2.1 introduces the history of WirelessHART and its advantages over other wireless technologies. Section 2.2 illustrates its network architecture and main components. Section 2.3 examines the significant designs in communication protocol that offer it advantages. Section 2.4 introduces a widely used formal description for WirelessHART.

2.1 WirelessHART Overview

WirelessHART is a ‘Wireless Mesh Network Communications Protocol designed to meet the needs for process automation applications’ [23]. It is based on the wired HART standard and is backward compatible with existing HART devices and applications. The standard was initiated in early 2004 and developed by 37 HART Communications Foundation (HCF) companies. In April 2010, WirelessHART was approved by the International Electrotechnical Commission (IEC) as a full international standard [3], which makes it the first wireless communication standard specifically designed for process measurement and control applica-
2.1. WIRELESSHART OVERVIEW

Before WirelessHART was released, there have been a few publicly available wireless communication technologies that are summarized in Table 2.1. Among them, ZigBee and Bluetooth are both promising to be applied in Wireless Control Networks. However, they are not the most ideal in manufacturing automation. This is, because industrial control has stringent requirements on time-synchronization and reliability. Specifically, as Song [6] pointed out, ‘the sensory information collected from a sensor is not necessarily replaceable by that from other nearby sensors’. Hence, to ensure global control over all the source nodes, it calls for strict time-synchronization and reliable communication. Bluetooth, although simple and low-cost, assumes a quasi-static star network, which is not scalable enough to be used in large process control systems. ZigBee, which completely complies to the IEEE 802.15.4 standard, is tailored for low data rates and large scale ad-hoc networks. Nevertheless, as Zheng’s experiment report [24] reveals, it has difficulties to meet the real-time and reliability requirements in the industrial environment.

In contrast, WirelessHART is a secure and robust networking technology operating in the 2.4GHz ISM (Industrial, Scientific, and Medical) radio band. WirelessHART utilizes IEEE 802.15.4 compatible radios with Frequency-hopping Spread Spectrum (FHSS) by packet basis. Taking advantage of the FHSS [25], it can better co-exist with other wireless networks, avoiding degrading each others’ performance. Moreover, it uses Time Division Multiple Ac-

| Table 2.1: Characteristics of wireless communication technologies |
|---------------------|---------------------|---------------------|---------------------|---------------------|
|                     | IEEE 802.15.1 (WPAN) | Bluetooth v2.1      | IEEE 802.15.4 (low rate WPAN) | ZigBee             |
| Throughput          | 1 Mbps (raw data rate) | 3 Mbps (EDR, raw data rate) | 20-250kbps (raw data rate) | 20-250kbps (raw data rate) | 2-54 Mbps (raw data rate) | 20-250kbps (raw data rate) |
| Packet length       | 366, 1622, and 2870 bits | 366, 1622, and 2870 bits | 1024 bits / MAC layer packet | 34-234s bytes | 127 bytes |
| MAC protocol type   | Dynamic TDMA         | Dynamic TDMA         | Slotted and unslotted CSMA/CA | Slotted and unslotted CSMA/CA | CSMA/CA |
| Frequency hopping   | YES                  | YES                  | Not specified               | Not specified     | YES |
| Encryption          | E0                   | E0                   | 128-bit AES for encryption and/or integrity | Key exchange for AES encryption | WEP (802.11i: WPA) |
| Frequency bands     | 2.402-2.405 GHz      | 2.402-2.405 GHz      | 868.902-928, 2400-2443.5 MHz | 868.902-928, 2400-2443.5 MHz | 2.4-2.5 GHz 5, 15-5, 855 GHz |
| Effective range     | 1-100 m              | 1-100 m              | 10m nominal (1-100m based on settings) | 10m nominal (1-100m based on settings) | -75 m outdoor, -25 m indoor |
| Supported number of nodes | Protocol dependent | 1 master and up to 7 active slave nodes per panid | 255 devices per network | 255 devices per network | Practical limitation due to collisions |
|                     | Protocol dependent   | Protocol dependent   | Protocol dependent          | Protocol dependent | 250 devices per network |


tions.

Before WirelessHART was released, there have been a few publicly available wireless communication technologies that are summarized in Table 2.1. Among them, ZigBee and Bluetooth are both promising to be applied in Wireless Control Networks. However, they are not the most ideal in manufacturing automation. This is, because industrial control has stringent requirements on time-synchronization and reliability. Specifically, as Song [6] pointed out, ‘the sensory information collected from a sensor is not necessarily replaceable by that from other nearby sensors’. Hence, to ensure global control over all the source nodes, it calls for strict time-synchronization and reliable communication. Bluetooth, although is simple and low-cost, assumes a quasi-static star network, which is not scalable enough to be used in large process control systems. ZigBee, which completely complies to the IEEE 802.15.4 standard, is tailored for low data rates and large scale ad-hoc networks. Nevertheless, as Zheng’s experiment report [24] reveals, it has difficulties to meet the real-time and reliability requirements in the industrial environment.

In contrast, WirelessHART is a secure and robust networking technology operating in the 2.4GHz ISM (Industrial, Scientific, and Medical) radio band. WirelessHART utilizes IEEE 802.15.4 compatible radios with Frequency-hopping Spread Spectrum (FHSS) by packet basis. Taking advantage of the FHSS [25], it can better co-exist with other wireless networks, avoiding degrading each others’ performance. Moreover, it uses Time Division Multiple Ac-
cess (TDMA) [25] technology to arbitrate and coordinate communications between network devices, as listed in Table 2.1. These two schemes make WirelessHART more competitive than ZigBee. The details of WirelessHART TDMA will be explained in Section 2.3.2.

## 2.2 WirelessHART Architecture

The WirelessHART architecture is designed to be user friendly, reliable and inter-operable. Figure 2.1 demonstrates all possible components within a hierarchy. Among them, three principle components are:

- **Field Devices** are attached to the process or plant equipment. They can be either wire-powered or battery-powered. These devices are the mesh network nodes that encompass sensors, actuators and wireless components. The sensors are responsible for collecting monitoring data such as flow speeds, fluid levels, or temperatures. Actuators, e.g. valves and pumps, perform the control command they receive. In practice, there may be some pure relay field devices that only forward messages for other devices.

- **Gateways**, like the network hub, enable communication between Host Applications and Field Devices in the WirelessHART Network. Each gateway can support one or more Access Points.

- **Network Manager** is responsible for the configuration of the network, i.e., scheduling communication between field devices, management of the routing tables and monitoring and reporting the health of the network.
2.3. **STANDARD SPECIFICATIONS**

The above three main components form the backbone of a typical WirelessHART network. Fields devices and the gateway are essential to an operative network. Host application presents the monitoring data to end users and send the control signal to the access point of the plant through a gateway. The links between these parts are wired. In addition, to be compatible with existing HART devices deployed in plants before, Wireless Adapters bridge the gap between Wire and Wireless, as shown in the right bottom of Fig 2.1.

### 2.3 Standard Specifications

The advantage of WirelessHART, as discussed in Section 2.1, comes not only from its architecture, but also from its communication protocols. Figure 2.2 [6] illustrates the HART protocol stack contrasted to the ISO OSI seven-layer model. According to this diagram, the WirelessHART protocol stack includes five layers: the physical layer, the data link layer, the network layer, the transport layer and the application layer. The top two layers are only roughly defined, to increase application flexibility. Thus, only the bottom three layers will be examined hereafter.

#### 2.3.1 Physical layer

The WirelessHART physical layer is fundamentally based on the IEEE Standard 802.15.4-2006 DSSS version, thereby shares the following features of the IEEE 802.15.4:

- **Data Rate**: 250Kbps

![Figure 2.2: HART protocol stack](image-url)
### 2.3. STANDARD SPECIFICATIONS

<table>
<thead>
<tr>
<th>Frequency [MHz]</th>
<th>Tx power [dBm]</th>
<th>Channel 1</th>
<th>Channel 11</th>
<th>Channel 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>2400-2483.5</td>
<td></td>
<td>2412</td>
<td>2437</td>
<td>2462</td>
</tr>
<tr>
<td>2412-2437</td>
<td></td>
<td>2400</td>
<td>2462</td>
<td>2483.5</td>
</tr>
<tr>
<td>2437-2462</td>
<td></td>
<td>2400</td>
<td>2483.5</td>
<td>2412</td>
</tr>
<tr>
<td>2462-2483.5</td>
<td></td>
<td>2400</td>
<td>2412</td>
<td>2437</td>
</tr>
</tbody>
</table>

**Figure 2.3: Frequency Channel and Transmission Power of 802.15.4 and 802.11**

- Operating Frequency: 2400-2483.5 MHz,
- Channel: 16 non-overlapping channels, each occupies 5MHz band.
- Modulation: O-QPSK; Direct Sequence Spread Spectrum (DSSS)
- IEEE compliant Physical Layer PDU; Maximum payload 127 bytes.

The IEEE 802.15.4 radios were chosen because they are relatively low power instruments suited to wireless process control applications. They use 10dB amplifiers to allow communication up to 100 meters away from the next instrument, which is a considerably long distance among field devices in all types of plant environments. Furthermore, the sensory data conveying either temperature, pressure or speed information, is much shorter than multi-media data as, e.g. audio packages. Therefore, the 250Kbps data rate is sufficiently fast.

Figure 2.3 [26] shows the potential overlap between IEEE 802.11 and IEEE 802.15.4 radios. A given IEEE 802.11b/g (WiFi) access point will only use one of the three non-overlapping channels and will only broadcast periodically, so the channel is not in continuous use. Pseudo-random frequency channel hopping inherent to WirelessHART instruments ensures that they do not use the same channel being used by an IEEE802.11 network for any lengthy period of time. In conjunction with channel hopping, WirelessHART supports another feature: Channel Blacklisting [6], to further reduce the interference. Channels that are highly utilized by other networks and suffer constant interferences will be put into the blacklist and excluded from the active channel list.

#### 2.3.2 Data link layer

As mentioned before, one merit of WirelessHART is the time-synchronized data link layer. WirelessHART defines a strict 10 millisecond time slot and utilizes Time Division Multiple Access (TDMA) to provide collision-free and deterministic communications. Specifically, only one transaction is permitted in each frequency channel at a given time slot across the

---

- Operating Frequency: 2400-2483.5 MHz,
- Channel: 16 non-overlapping channels, each occupies 5MHz band.
- Modulation: O-QPSK; Direct Sequence Spread Spectrum (DSSS)
- IEEE compliant Physical Layer PDU; Maximum payload 127 bytes.

The IEEE 802.15.4 radios were chosen because they are relatively low power instruments suited to wireless process control applications. They use 10dB amplifiers to allow communication up to 100 meters away from the next instrument, which is a considerably long distance among field devices in all types of plant environments. Furthermore, the sensory data conveying either temperature, pressure or speed information, is much shorter than multi-media data as, e.g. audio packages. Therefore, the 250Kbps data rate is sufficiently fast.

Figure 2.3 [26] shows the potential overlap between IEEE 802.11 and IEEE 802.15.4 radios. A given IEEE 802.11b/g (WiFi) access point will only use one of the three non-overlapping channels and will only broadcast periodically, so the channel is not in continuous use. Pseudo-random frequency channel hopping inherent to WirelessHART instruments ensures that they do not use the same channel being used by an IEEE802.11 network for any lengthy period of time. In conjunction with channel hopping, WirelessHART supports another feature: Channel Blacklisting [6], to further reduce the interference. Channels that are highly utilized by other networks and suffer constant interferences will be put into the blacklist and excluded from the active channel list.
2.3. STANDARD SPECIFICATIONS

According to the data in Table 2.1, to transmit a payload in the WirelessHART takes $127 \times 8/250k \approx 4ms$. So the 10ms slot is sufficient to transmit one message and to receive the corresponding acknowledgement. As presented in Figure 2.4 [17], a series of consecutive slots that repeat every cycle forms a ‘Superframe’.

Figure 2.5 describes the overall design of the data link layer which consists of the following six modules. The two interfaces describe the service primitives from the physical layer and to the network layer. State Machine consists of three primary components including the TDMA state machine. Timer is responsible for time synchronization in all the network devices. Communication Tables, as shown in sub blocks in Fig. 2.5, maintain a collection of tables to store scheduling and routing information. Link scheduler is to determine the next transmission slot based on the communication schedule in the superframe table and link table. The details of their separate responsibilities can be explored in reference [6].
2.3.3 Network layer

The network layer determines how the messages are routed from a source node to the gateway and vice versa. With WirelessHART’s mesh networking, field devices do not need to have a direct forwarding path to the network gateway. Each field device is capable of routing the message of other instruments along a route that will ensure the message reaches its ultimate destination. According to the WirelessHART data sheet [23], it supports a variety of routing algorithms:

- Upstream and downstream graph routing. Provides redundant path routing for maximum reliability and managed latency.
- Source routing for ad-hoc communications and confirmation of path viability.
- Supports Broadcast, multi-cast and unicast transmissions.

The transportation layer and application layer are not thoroughly defined in the WirelessHART protocol, leaving flexibility for user customization, and are thus not examined here. In conclusion, through the assessment of WirelessHART’s specifications on different layers, WirelessHART is justified to be the most appropriate communication protocol for industrial control systems. Since it does not only inherit IEEE 802.15.4’s merits, but goes even beyond that by integrating other technologies such as FHSS and TDMA in an innovative manner.

2.4 Formal Description

As introduced in the last section, WirelessHART reflects not only regular networking but control aspects as well. In 2009, Alur and D’Innocenzo et al. from the University of Pennsylvania proposed a formal syntax of multi-hop control networks [4]. They take a control system in Figure 2.6 [4] as an example. In this example network two wireless nodes are used to measure information from two plant equipment, four others send the information to a controller (connected with the gateway) and then pass it back to actuate the plants. This usage scenario is typical among WirelessHART networks. Furthermore, they propose the following syntax involving the aspects of (1) a mathematical model for the control loops, (2) the topology of the network, (3) the location of sensors/actuators, and (4) a routing strategy.

Definition 1 [4] A multi-hop control network is a tuple $\mathcal{N} = (\mathcal{D}, \mathcal{G}, \Omega, \mathcal{R})$, where

- $\mathcal{D} = \{(A_i, B_i, C_i), (\tilde{A}_i, \tilde{B}_i, \tilde{C}_i)\}_{p=1}^P$ models the control loops. Each control loop is described by a pair of triplets of matrices. The first triplet in each pair defines the dynamics of the plant and the second triplet defines the dynamic of the control algorithm, both in terms of matrices of Linear Time Invariant (LTI) systems. Let $\mathbb{I} = \cup_{i=1}^P \{y_{i,1}, \ldots, y_{i,m_i}\}$ be the set of input signals for the plants and $\mathbb{O} = \cup_{i=1}^P \{u_{i,1}, \ldots, u_{i,m_i}\}$ be the set of output signals from the plants.
2.4. FORMAL DESCRIPTION

\[ G = \{N, E\} \] is a directed graph models the radio connectivity of the network, where vertices are nodes of the network, and edges \((n_1, n_2)\) represents the wireless link between node pairs that can sustain reliable communication. An edge \((n_1, n_2)\) exists if and only if node \(n_1\) and \(n_2\) could communicate reliably with each other in dual directions. Let \(P\) be the set of simple paths in \(G\) that start or end with the controller.

- \(\Omega : I \cup O \to N\) assigns every pair of input and output signal the node that implements sensing and actuation.
- \(\mathcal{R} : I \cup O \to 2^P\) is a map, which associates to each input/output signal a set of allowed simple paths from/to the controller.

In addition, for a given network \(\mathcal{N}\), a communication schedule is given by a function \(\eta : \mathcal{N} \to 2^E\) that associates with each time \(t\) a set of edges in the graph \(G\). An edge \((n_1, n_2) \in \eta(t)\) if and only if at time \(t\) the content of node \(n_1\)'s memory is copied to the node \(n_2\).

We take an example for the control loop \(\mathcal{D}\) in Fig. 2.6. In this formal model, each triplet \(A_i, B_i, C_i\) models an LTI plant and each triplet \(\tilde{A}_i, \tilde{B}_i, \tilde{C}_i\) models an LTI feedback block. The relationship between control signals is illustrated in the following set of equations. Among them, the first two equations represents signals in the plant, the third and fourth equations are about the controller signal. The last two equations stand for the input signal from the plant to the controller and the opposite. Note that, the wireless network introduces both measurement and actuation delays, instead of direct interconnections.

\[
\begin{align*}
x_i(t+1) &= A_i x(t) + B_i u(t) \\
y_i(t) &= C_i x(t) \\
\tilde{x}_i(t+1) &= \tilde{A}_i \tilde{x}(t) + \tilde{B}_i \tilde{u}(t) \\
\tilde{y}_i(t) &= \tilde{C}_i \tilde{x}(t) \\
u_i &= \tilde{y}_i \\
\tilde{u}_i &= y_i
\end{align*}
\]
The above definition integrates fundamental topology and control loop theories, thus is considered appropriate and widely accepted in related research. In this work, this syntax is used as it stands and supplemented with more WirelessHART specific notations such as ‘path’. More details will come in chapter 4.
Chapter 3

Discrete-Time Markov Chains

In Chapter 2, an Industrial Control System, namely WirelessHART was introduced. It has a standardized architecture, communication protocols and formal description. As a fresh control-oriented standard, the performance of WirelessHART networks has not received much attention yet. So we take a scrutiny of the network’s functionaries and present a modeling and analysis approach in the following. In this chapter, Section 3.1 introduces fundamental Markov chain theories; Section 3.2 introduces the Binary Symmetric Channel as a discrete channel model; Section 3.3 proposes our DTMC link model as the first modeling step; and Section 3.4 reveals the correlation between the channel model and the link model and find a method to derive the failure transition probability from practical measurement.

3.1 Discrete-Time Markov Chains

Markov Chain model is selected as the mathematical tool to describe a system because it is quick, powerful and analytically tractable. Moreover, it is widely applied in performance evaluation and other technical fields. A Discrete-Time Markov chain is a stochastic process that consist of a series of random variables \( X_t \). The values of random variables \( X_t \) are discrete and are called states. Without loss of generality, the states can be denoted as \( x_i, i \in \mathbb{N} \). The set of all possible states forms the finite state space \( S = \{ x_i, x_2, \ldots, x_n \} \). Such a discrete-state space process is referred as a ‘chain’. Discrete-Time Markov Chain is a special type of the chain.

**Definition 2 (Discrete-Time Markov Chain) [27]**

A discrete-time, discrete-state space chain \( \{ X_t | t \in \mathbb{N} \} \) is called Discrete-time Markov Chain (DTMC), if it satisfies the following property:

\[
Pr\{X_{n+1} = x_{n+1} | X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n\} = Pr\{X_{n+1} = x_{n+1} | X_n = x_n\} \tag{3.1}
\]

Equation 3.1 is generally called Markov Property, which states that the future behaviour of Discrete-Time Markov Chains only depends on its current state and not on the states assumed in the past [28]. Most often, Markov processes used for performance evaluation are
3.1. DISCRETE-TIME MARKOV CHAINS

invariant to time shifts, resulting in the so called time-homogeneous Markov processes. Under these circumstance, the conditional probability of moving from state $i$ at time $m$ to state $j$ at time $n$ only depends on the time difference $l = n - m$. Therefore $p_{i,j}(l) = Pr\{X_{m+l} = j|X_m = i\}$ are called the $l$-step transition probabilities. Especially, the 1-step transition probabilities are denoted as

$$p_{i,j} = Pr\{X_{n+1} = j|X_n = i\}.$$ 

Let $p_j(t) = Pr(X_t = j)$ denote the probability of "being" in state $j$ at time $t$. The initial distribution of the Markov Chain is defined as

$$p(0) = [p_0(0), p_1(0), \ldots, p_n(0)].$$

It is a vector where the entries correspond to the state space $S$ of the Markov Chain. In case of a time-homogeneous Markov processes, the DTMC is totally described by the initial distribution and the 1-step transition probabilities. All the 1-step transition probabilities can be combined in a transition probability matrix $P = [p_{i,j}]$.

$$P = \begin{bmatrix}
p_{0,0} & p_{0,1} & p_{0,2} & \cdots \\
p_{1,0} & p_{1,1} & p_{1,2} & \cdots \\
\vdots & \vdots & \vdots & \ddots \\
\vdots & \vdots & \vdots & \ddots 
\end{bmatrix}, \quad (3.2)$$

where all the entries $p_{i,j}$ satisfy

$$0 \leq p_{i,j} \leq 1 \text{ and } \sum_j p_{i,j} = 1, \text{ for all } i.$$

Consider the $(m+n)$-step transition from state $i$ to state $j$ via an intermediate state $k$, then the Markov property implies that these two sub-transitions are independent, which yields

$$p_{i,j}(m+n) = \sum_k p_{i,k}(m)p_{k,j}(n). \quad (3.3)$$

The above equation is one form of the well-known Chapman-Kolmogorov equation [29], which provides an efficient method of calculating the $n$-step transition probabilities. Let $P(n)$ denote the $n$-step transition probability matrix, then by using the matrix form of Equation 3.3,

$$P(n) = P \cdot P(n-1) = P^n.$$ 

The transient distribution of the Markov chain is a row vector

$$p(n) = [p_0(n), p_1(n), \ldots, p_n(n)],$$

where $p_j(n) = Pr(X_n = j) = \sum_i p_i(0)p_{i,j}(n)$. Then the transient distribution is given by

$$p(n) = p(0)P(n) = p(0)P^n, \quad (3.4)$$
where \( p(0) \) is the initial distribution. This implies that the transient distributions of a time-homogeneous DTMC is completely determined by the initial distribution \( p(0) \) and the one-step transition probability matrix \( P \).

In order to introduce the long-run or steady-state probability, we first need to classify the states of a Markov chain into those that the system visits infinitely and those that it visits only a finite number of times.

**Definition 3 (Transient State)** A state \( i \) is said to be transient (or non-recurrent) if and only if there is a positive probability that the process will not return to this state.

**Definition 4 (Recurrent State)** A state \( i \) is said to be recurrent if and only if, starting from state \( i \), the process eventually returns to state \( i \) with probability one. A recurrent state \( i \) is said to be positive recurrent if its mean recurrence time is finite.

**Definition 5 (Absorbing State)** A state \( i \) is said to be an absorbing state if and only if \( p_{i,i} = 1 \).

**Definition 6** The following definitions [28] classify Markov chains and their states as periodic or aperiodic:
1. For a recurrent state \( i \), define the period of the state \( i \), denoted by \( d_i \), as the greatest common divisor of the set of positive integers \( n \) such that \( p_{i,i}(n) > 0 \).
2. State \( i \) is said to be aperiodic if \( d_i = 1 \).
3. A Markov chain is said to be aperiodic if all states are aperiodic.

**Definition 7 (Irreducible Markov Chain)** A Markov chain is said to be irreducible if every state can be reached from every other state in a finite number of steps.

Recall the transient distribution as defined in Equation 3.4, as \( n \to \infty \), the \( n \)-step transient probabilities may converge towards common limits on the following condition.

**Theorem 1 (Limiting distribution)** For an aperiodic Discrete-Time Markov chain, the limiting distribution

\[
\mathbf{v} = \lim_{n \to \infty} p(n) = \lim_{n \to \infty} p(0)P^n
\]

exists. Moreover, it can be computed as

\[
\mathbf{v}(P - I) = \mathbf{0} \quad \text{and} \quad \sum_j v_j = 1.
\]

This Theorem helps to calculate the limiting distribution, although it is not unique, because of the dependency on the initial distribution. The next theorem expresses when a DTMC has a unique stationary distribution.

**Theorem 2 (Stationary distribution)** In an irreducible and aperiodic DTMC with positive recurrent states, the limiting distribution \( \mathbf{v} \) is unique and independent of the initial distribution \( p(0) \), thus becomes the stationary (steady-state) probability distribution \( \pi \).
3.2 Binary Symmetric Channel Model

To model a WirelessHART network, the first step is to model the communication channels that connect two or more field devices. Traditionally, there are two approaches to simulate a channel: a) Represent the transmitted signal, noise, interference, and other channel disturbances by samples of waveforms, and then a waveform-level simulation on a sample-by-sample basis; b) Abstract for the physical (waveform) channel to the discrete (digital) channel model, where the channel is completely characterized by a small set of parameters. Because the Discrete Channel Model (DCM) method calls for less computation and running time, it is selected in this work rather than the waveform-level channel model.

In Discrete channel models, the Binary Symmetric Channel (BSC) is the most simple and fundamental one. And it is widely used in communication system analysis because many problems in communication theory can be reduced to a BSC. It is a ‘memoryless’ model of a transition from the transmitter coder to the receiver decoder, under the assumption that there is no temporal correlation in the transition mechanism.

Figure 3.1 demonstrates an example of a binary symmetric channel model. The \( x_k \in \{0, 1\} \) denotes transmitted symbol in one bit, \( y_k \) denotes received symbol. \( p_k \) denotes the transmission error probability for both cases that 0 is decoded as 1 and 1 is decoded as 0. This error probability is independent of the past and future bits. Formally, this probability is referred to as the bit error rate (BER). The BER is an important channel parameter that varies according to different noise level and modulation technologies.

![Binary Symmetric Channel model](image)

Figure 3.1: Binary Symmetric Channel model

3.3 Link Model

In a WirelessHART network, a link is the direct transmission path between two nodes. The same link can work on different channels. As introduced in Section 2.3, WirelessHART
3.3. LINK MODEL

Figure 3.2: Threshold of receiver signal strength and two link states

uses Time Division Multiple Access (TDMA) for medium access to achieve reliable real-time communication. The main feature of the TDMA is that time is synchronized and slotted, which facilities the modeling of wireless links as a Discrete-Time Markov Chain.

Consider a dynamic link where the received signal strength is above an acceptable threshold part of the time, and below the threshold with strong noises, as shown in Figure 3.2 [30]. If only the threshold conditions are concerned of importance, this link can be described by the state space $S = \{UP, DOWN\}$. In the $UP$ state, the transmission error probability is negligible (e.g. $P_E < 10^{-3}$); while in the $DOWN$ state, the received signal strength is so low that the error probability is unacceptably high ($P_E > 10^{-3}$).

Assuming time is measured in increments of a slot, then these $UP$ and $DOWN$ states change per slot (10ms). Taking one slot as one discrete time step in the DTMC, we are capable of modeling the link by a two-state discrete-time Markov chain. The two-state DTMC transition diagram is shown in Figure 3.3. In case the link is $UP$, the entire message would be transmitted successfully without any bit error; in case the link is $DOWN$, the message transmission fails due of one or more bit error and it needs to be re-send later. The link state remains the same during one slot and may change in the next slot. These changes comply to the following transition probabilities: failure probability $p_{fl}$ and recovery probability $p_{rc}$ as Fig. 3.3 shows.

Figure 3.3: Two-state DTMC link model

According to Equation 3.2, the transition probability matrix of this DTMC is given by

$$
P = \begin{bmatrix}
1 - p_{fl} & p_{fl} \\
p_{rc} & 1 - p_{rc}
\end{bmatrix}.
$$

(3.5)
Note that both the UP and DOWN state can definitely return to themselves after finite time steps, therefore they are positive recurrent according to Definition 4. In addition, the greatest common divisor of the time to return is 1, i.e. \( d_i = 1 \). According to Definition 6, the states are aperiodic and the DTMC is aperiodic too. Since both states can be reached from the other one directly, the DTMC is irreducible, according to 7.

Hence, the link model DTMC is irreducible and aperiodic with positive recurrent states. According to Theorem 2, it has a unique steady-state distribution \( \pi \), which is described by:

\[
\pi(P - I) = 0.
\]

By this equation, the steady-state probability distribution can be derived as

\[
\pi = [\pi(up), \pi(down)] = \left[ \frac{p_{rc}}{p_{rc} + p_{fl}}, \frac{p_{fl}}{p_{rc} + p_{fl}} \right].
\] (3.6)

The steady-state probability of the link state ‘UP’ i.e. \( \pi(up) \) is the stationary availability of the link, indicating the percentage of time when the link is operational. The link availability will serve as a critical parameter in the WirelessHART path model, which is to be proposed in the next chapter.

### 3.4 Failure Transition Probability

In the above sections, the Binary Symmetric Channel Model and the Two-state Link Model have been proposed. Although the two models take different perspectives, they are actually consistent with each other. The binary symmetric channel model describes the transmission of a single bit, while the two-state link model describes the transmission of one message. Assume the typical WirelessHART message is \( L \) bits long. Recall that the failure probability \( p_{fl} \) is the probability that at least one bit is decoded wrongly within one received message. The success transmission of each bit (with probability \( 1 - BER \)) follows a Bernoulli distribution. So the failure transition probability is given by:

\[
p_{fl} = 1 - (1 - BER)^L.
\] (3.7)

Equation 3.7 reveals the correlation between the link model status and the bit error rate of the active channel. Recall that in Section 2.3, the modulation technology of WirelessHART radio is OQPSK (Offset quadrature phase-shift keying). According to Rappaport’s book [25], the Bit Error Rate of OQPSK modulation in a AWGN (Additive white Gaussian noise) channel is given by:

\[
BER_{OQPSK} = \frac{1}{2} erfc \left( \sqrt{\frac{E_b}{E_0}} \right),
\] (3.8)

where \( erfc() \) represents the complementary error function, and \( E_b/E_0 \) represents the energy per bit to noise power spectral density ratio, which is a normalized signal-to-noise ratio (SNR) measure and can be regarded as the ”SNR per bit” Combined Equation 3.7 with
Equation 3.8, it can be seen that the failure probability $p_{fl}$ is determined by the SNR. The received SNR can be measured by pilot packages that are transmitted from one node to the other via the wireless link. In this way, probability $p_{fl}$ can be derived from the measured SNR and then be used in our path models, which will be proposed in Chapter 4.
Chapter 4

Hierarchical Path Model

In this chapter, we propose a DTMC Path Model to describe the message forwarding in a uplink path in the WirelessHART networks. The path model, co-operated with the link models explained in Section 3.3, is capable of producing message age probabilities when they reach the gateway. This information is fundamental to obtain performance measures. Section 4.1 presents the control loops in WirelessHART to explain how the network performs its duty as a control system. The important concepts such as superframe, reporting interval and communication schedule will be introduced, because they are directly related to the Path Model. Section 4.2 demonstrates the Hierarchical Path Model in all aspects. For example, the modeling target is selected to be a uplink path. DTMC states are tuples of message ages. And the constructing algorithm will be explicitly clarified as well.

4.1 Control Loop

WirelessHART, as a closed-loop control system, uses feedback to control the outputs of industrial instruments. This control loop $D$ is realized through the components of the WirelessHART network. Recall the WirelessHART architecture as presented in Section 2.2: Field devices, including sensors and actuators, can be regarded as the source nodes and relay nodes in a wirelessHART network. The gateway, as the network routing destination, has wired connection to the controller and then the application host. The HART Foundation [31] presents a basic usage scenario in Figure 4.1, where the controller adjusts a valve (the actuator) to manipulate the flow rate through the cooling jacket (the sensor).

First take a look at the control loop diagram as presented in Fig. 4.1(b). A raw temperature measurement from a cooling jacket is converted into engineering units and checked for violation of alarm limits using an Analog Input (AI) function block. Next, a PID (Proportional Integral Derivative) function block is used to maintain the temperature of the operating target, here, the reactor temperature setpoint. An Analog Output (AO) function block then drives the actuator valve. Correspondingly, in the physical setup 4.1(a), the con-

\[1\] Despite their name, Analog Input (AI) and Analog Output (AO) function blocks work with the digital information in WirelessHART communications.
4.1. CONTROL LOOP

(a) Basic scenario physical setup

(b) Basic scenario control loop

Figure 4.1: WirelessHART control basic scenario

trol functions, i.e., the ID block is run in centralized, wire-powered controllers. The AI block located in the sensor node (cooling jacket). The input signal $I$, as mentioned in Definition 1, is transmitted wirelessly via the mesh network to the gateway. This is the so called Uplink. On the other end, the output signal $O$ is transmitted to the AO block on the actuator (valve) via the Downlink. Besides this deployment, AI and AO blocks can be alternatively deployed in the central controller to further simplify the field device implementation and minimize their power consumption to extend the battery lifetime.

4.1.1 Stability

For multi-hop control networks such as WirelessHART, the stability of control loops is a critical issue. Gera Weiss and Innocenzo [5] model and analyze the effects of link failures on the stability of the control loops. They classify link failures into the following three categorizes:

- Permanent link failures, when the duration of communication error is long compared to the speed of the control system.
- Transient error model, where links fail for one time slot independently of the past and of other links.
- Errors with random time span, where links can recover from failures after some time.
Moreover, they prove that analyzing the stability with permanent link failures is a NP-hard problem. Because channel hopping is used in WirelessHART, in the networks mostly transient errors occur. The control loop with transient errors is proved to have ‘almost sure stability’ [5] under certain conditions. No matter which link failure model is considered, the stability of the control loop depends on the reachability and delay of these messages that travel between source node and the gateway. The more the messages delay, the less stable the control loops become.

4.1.2 Superframe

As previously introduced in Section 2.3.2, the WirelessHART MAC layer is slotted and synchronized, taking advantage of TDMA to provide collision-free medium access. A series of consecutive slots forms a group unit called superframe. One superframe is one macro cycle. Hereafter, the superframe size, i.e., the number of slots in the superframe, is denoted as $F_s$. On the one hand, it serves as the cycle of network communication schedule; on the other hand, the superframe encompasses one cycle of all possible control loops, as discussed above, across the whole WirelessHART network. All field nodes share the same superframe (or macro cycle) and are allocated part of the superframe slots to transmit messages Uplink/Downlink. Figure 4.2 elaborates the structure of a superframe. One superframe starts with the Analog Input (AI) blocks, which sample and digitalise the sensory data, send them in different uplink slots to the gateway. The gateway runs the PID control function, generates the output message and sends it back to the field devices in different downlink slots. The received output messages go through Analog Output (AO) blocks to close the control loop.

In practice, the execution time of AI, AP and PID control blocks are very short compared to the transmission slot [32].

![Figure 4.2: A typical superframe/macrocyle](image)

4.1.3 Reporting interval

Originally, in one superframe, the sensory data is sampled once and the control loop is executed once. However, WirelessHART allows longer ‘reporting intervals’ that meet the requirements of most control loops while at the same time minimizing the impact on field devices that may be powered by a battery. For instance, a pair of control input signal and output signal are presented in Figure 4.3. The typical rule of thumb is that feedback control should be executed 4 to 8 times faster than the process response time [32], which equals the process Time Constant plus Deadtime as shown in this figure. After the control command is
executed, the output remains the same during the deadtime and then starts to increase until it reach the new steady value again. The first line of arrows below indicates every potential point for control execution during this process. And the interval among them stands for the sampling interval, which is one superframe in our case.

Figure 4.3: Control signals, sample and reporting interval

Without compromising control stability, it is desirable to reduce the frequency that measurements are taken and communicated in order to save wireless communication overhead and extend field devices’ battery life. This motivation yields an important variable, i.e., ‘Reporting Interval $I_s$’, which indicates how often the nodes report the measurement back to the gateway for monitoring and control. The value is assumed to be $2^n$ times the superframe. For example, $I_s = 2$ means that the data is still sampled by each sampling interval/superframe, but is only sent back to the gateway every two superframes. The bottom two lines of arrows in Figure 4.3 demonstrate the situation of $I_s = 2$ and $I_s = 4$, respectively. In this way, the reporting interval length is $I_s \times F_s$. Hence, reporting intervals play a significant role in WirelessHART control. The influence of different reporting intervals on the network performance and corresponding trade-off will be evaluated using the path model later on.

### 4.1.4 Message life cycle

Through the mesh network, the sensory messages may suffer an extremely long delay that exceeds its reporting interval. These out-dated message is not useful for the real-time monitor and control applications, thus the system should limit the message life span in the following way. When a message is generated in a sensor node, it is stamped with a born time $T_{born}$ and attached with a Time-to-Live (TTL) field. With each time slot, the TTL field is decreased by one. However, uplink messages ‘sleep’ during downlink slots and do not decrease their
TTL and vice versa. As soon as the TTL reaches zero, the message ‘dies’ and should be discarded from the system to keep the registers clean.

4.2 Path Model

As explained in Section 4.1, the reporting interval, including one or more superframes, is a complete and independent control unit. So the temporal scope of our modeling is one reporting interval. To simplify the problem, we make the following assumption.

Assumption 1 Location of sensors/actuators $\Omega$: Without loss of generality, it is assumed that every field device includes both a sensor and an actuator, such that the sets of uplink and downlink end nodes are identical.

Under the above assumption, the uplink and downlink of the WirelessHART network can be considered symmetric, and their performance should be similar. So the uplink framesize $F_{up} = \frac{1}{2} F_{s}$. In this work, we focus on the performance evaluation of uplinks. The derived conclusions fits the symmetric downlink counterparts, as well.

4.2.1 Modeling target

As explained in Section 4.1.4, the sensory messages are time stamped with $T_{born}$. The initial value of the TTL field should be set to $I_s * F_{up} = \frac{1}{2} I_s * F_{s}$. The message may arrived at the destination gateway at a specified time $T_{rec}$. Right then, the received message’s age can be derived as

$$\text{Age} = T_{rec} - T_{born},$$

where the time unit is one slot (10ms).

At the end of every reporting interval, the gateway receives messages at different ages with different probabilities. And the set of age probabilities is a fundamental indicator, that most other performance measures, such as delay distribution, can be derived from. To generate such age probabilities for further processing is the main modeling goal.

The modeling target is the finite message forwarding Path, denoted as $i : n_j^i \rightarrow n_k^i \rightarrow \cdots \rightarrow G$, $j, k \in \mathbb{N}$, from a source node $n_j^i$ to the destination gateway $G$ via intermediate node $n_k^i$ and others. One path corresponds to one control loop in the system. The path can be either single-hop or multi-hop. The former can be seen in the control loop scenario in Figure 4.1, where the field devices are directly connected to the gateway by a wireless link. The multi-hop path instance can be found in the previous Fig 2.6 in Section 2.4. The uplink path from node 2 to the gateway via node 5 is a two-hop path.

It is worth mentioning that the modeling formalism presented in this thesis focuses on paths and is therefore independent of a specific routing algorithm. It can hence be used for both, source routing and graph routing.
4.2.2 Communication schedule

The communication schedule is one of the main issues that must be considered in the path modeling. To guarantee timely and reliable data delivery, the communication schedule is ‘centrally computed at the network manager (refer to Fig. 2.1), which has global knowledge of the network state, and then disseminated to all devices in the network’ [12]. Recall the formal description of WirelessHART in Section 2.4, the communication schedule \( \eta : \mathcal{N} \rightarrow 2^E \) associates the edges by order in the graph \( G \). It is denoted as \( \eta = (e_1, e_2, \ldots, e_n) \), where \( e_i = (n_j, n_k) \). The total length of the schedule is the superframe’s uplink size, \( F_{up} = \frac{1}{2} F_s \).

From the communication schedule \( \eta \), every node can generate its own individual schedule \( \mu \), which includes three node statuses: ‘Idle’, ‘Transmit’, and ‘Receive’.

Take the following example to illustrate the concept of a path and a schedule. Figure 4.4(a) gives a two-hop path \( n_4 \rightarrow n_1 \rightarrow G \) with two links \( a \) and \( b \). Assume the uplink framesize equals \( F_{up} = 6 \) and the network communication schedule is \( \eta = (\ast, (n_4, n_1), \ast, (n_1, G), \ast, \ast) \), where \( \ast \) means that slot is not relevant for this control loop. From it, the node 1’s individual schedule should be \( \mu_1 = (\text{Idle, Receive, Idle, Transmit, Idle, Idle}) \). Figure 4.4(b) shows the schedules in one superframe. Green slots represent successful transmission on link \( a \) and \( b \). However, there is always some risk that the transmission may fail due to various reasons. In that case, the node will keep the message and attempt to re-send it again in the same slot of the next superframe, as long as \( I_s > 1 \). In Figure 4.4(c), the transmission from node 4 to node 1 on link \( a \) failed (the red slot), and the second attempt in the next superframe succeed. Such re-send mechanism is closely related to the following path model.

![Example path, schedule and transmission](image)

Figure 4.4: An example of path, schedule and transmission

### 4.2.3 Path model DTMC

The above example shows that TDMA facilities the Discrete-Time Markov Chain modeling. This is because all the events (e.g. message transmission) happen in one slot unit, rather than in continuous time. Otherwise, it has to be modeled in Continuous-Time Markov Chain.

A state \( s \) in the path model DTMC represents the age of the messages at each node on the path. Each state represents a certain age of the message at each node on the path.
Hence, for a path with \( n \) hops, the state descriptor is a tuple of size \( n \): \((\text{age}_1, \text{age}_2, \ldots, \text{age}_n)\). For example, a state denoted as \((1, 8, -)\) stems from a three-hop path. The message at the first node is at the age of 1, the message at the second node is at the age of 8, and the third node has not received any message yet. In this manner, the state space \( S \) is a complete set of all possible age tuples on a path during a reporting interval. Note that not all combinations of \( n \) entries of age are possible in the state space, due to the constraint of transmissions according to the communication schedule. The path DTMC consists of mainly transient age states and the following two categories of absorbing (refer to Definition 5) states.

- Goal states, indicating that the message reaches the gateway at a certain age, e.g. state \((R7)\) means the message reach the gateway at age 7. For a reporting interval \( I_s \), the path DTMC has \( I_s \) goal states. This is because the link transmissions towards the gateway are scheduled in the same slot of different superframe cycle. For all received messages, the possible ages are given by:

\[
a_i = a_0 + (i - 1) \times F_{up},
\]

where \( F_{up} \) represents the uplink framesize and \( a_0 \) represents the transmission slot of the path link connecting to the gateway in the communication schedule \( \eta \). \( a_i \) can be regarded as the age of messages that reach the gateway in the \( i \)-th cycle. The goal states can be denoted as \( S_\varphi = \{s_i, \ldots, s_j\} \), where the indexes of the goal states belong to a set \( \varphi \), i.e. \( i, j \in \varphi \).

- ‘Discard’ state, indicating the drop of this message due to zero TTL values. This is similar to the concept of ‘package loss’ that appears in some literatures.

In the following we present an algorithm to derive the underlying DTMC for each path with a communication schedule. Under the scope of one reporting interval, the DTMC moves from one of the transient states to the next and ultimately reaches one of the absorbing states. It is constructed according to the following rules:

(a) The system starts with empty node registers and then a fresh message is generated at the source node, i.e. the initial state \( s_0 = (1, -, -) \).

(b) With each new slot, the age of all messages on the path is increased by one. So in the DTMC, the horizontal axis can be seen as a time line. If there is no transmission scheduled in \( \eta \) for those nodes that have a message, the state \((\text{age}_1, -, \ldots)\) evolves to a new state \((\text{age}_1 + 1, -, \ldots)\) with transition probability one.

(c) If a node is scheduled to transmit in a certain slot and has a message to forward right then, it attempts to send the message to the next hop via the link between them. Success or not, the node always keeps a copy of the message. As a result, two new states are generated. In case of successful transmission, the DTMC moves from the present state \((\text{age}_1, -, \ldots)\) to state \((\text{age}_1 + 1, \text{age}_1 + 1, \ldots)\) with a transition probability \( p_s \). In case the transmission fails, the DTMC moves from the present state \((\text{age}_1, -, \ldots)\) to the
4.2. PATH MODEL

Consider a three-hop path \( n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow G \) as an example. Assume the reporting interval to be \( I_s = 1 \), so that the model scope is only one superframe. Take the uplink framesize \( F_{up} = 7 \) and the communication schedule \( \eta = (*,(n1,n2),*,*,(n2,n3),*,(n3,g)) \). The path DTMC is constructed following the above rules and shown in Figure 4.5. The initial state is \((1,-,-,-)\). In the schedule \( \eta \), the first slot is idle, so the DTMC moves to the second state \((2,-,-,-)\) with probability 1, according to rule (b). In the second slot, the scheduled edge \((n_1,n_2)\) indicates a transmission on the link between \( n_1 \) and \( n_2 \). According to rule (c), the DTMC moves from state \((2,-,-,-)\) to state \((3,3,-,-)\) with probability \( p_{s1} \), and to state \((3,-,-,-)\) with probability \( p_{f1} \). After \( F_{up} = 7 \) steps, it may either reach the goal state \( R7 \) at the seventh slot or reach the ‘Discard’ state when \( TTL = 0 \) at the end of this cycle.

4.2.4 Hierarchical model

The hierarchical model, as presented in this thesis, actually consists of two tiers. One tier is the path model as introduced above. The other is the two-state link models, as previously proposed in Section 3.3. Recall that the link model has two states ‘UP’ and ‘Down’, failure transition \( p_{fl} \) and recovery transition \( p_{cr} \). The states of the link DTMC then determine the success transition probability \( p_s \) and the failure transition probability \( p_f \) in the path model.

In a WirelessHART network, a message is transmitted successfully if and only if the wireless
link remains operational in that slot. Therefore, the probability $p_s$ equals the transient link availability at the very transmission slot $t$, and vice versa. The dependency is expressed in the following equation:

$$[p_s(t), p_f(t)] = p_{\text{link}}(t). \tag{4.2}$$

Especially, if the link is in steady-state during the transmission, then

$$[p_s, p_f] = [\pi(\text{up}), \pi(\text{down})] = \left[ \frac{p_{\text{rc}}}{p_{\text{rc}} + p_{\text{fl}}}, \frac{p_{\text{fl}}}{p_{\text{rc}} + p_{\text{fl}}} \right]. \tag{4.3}$$

Equations 4.2 and 4.3 form the connection between the two tiers of the model. That is, the path model relies on the link models to model whether a message can be transmitted successful or not. For an $n$-hop path model, another $n$ link models exist and evolve together with the path DTMC simultaneously. Step by step, the link model keep providing its transient state probabilities as the transition probabilities $p_s$ and $p_f$. The hierarchical idea makes the DTMC model be capable to describe dynamic behaviours when links are initially down, or burst error independent of past states.

The alternative modeling is deploying the link model and the path model in one tier by multiply their states. In this way, the new state space expands to a double size. To avoid such a huge state space is our motivation to choose the hierarchical model.

### 4.2.5 Model complexity

The first example assume that a fresh message is produced in each superframe ($I_s = 1$), which is the simplest case. In this section, we develop the DTMC model on the same path but for a different reporting interval. Generally, for a reporting interval of size $I_s = m$, the number of goal states is $m$ and every node has $m$ slots to send its message. Thus, the DTMC size is proportional to $I_s$. As mentioned in Section 4.2.1, the initial value of TTL is set as $TTL_o = I_s * F_s$. Observing the model diagram, one can find that the horizontal length equals $TTL_o$ and the vertical height equals the number of path hops $n$. Therefore, the following theorem can be concluded.

**Theorem 3 (Model complexity)** For an $n$-hop path with superframe size $F_s$ and reporting interval $I_s$, the computational complexity of the path model is $O(I_s F_s n)$.

This theorem indicates that the model complexity expands linearly with the increase of $I_s$, $F_s$ or $n$. As an example, Figure 4.6 shows the DTMC diagram for $I_s = 2$ of the same path $n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow G$. Compared to Figure 4.5, the second goal state ($R_{14}$) is added, and the complexity is almost doubled, which complies with Theorem 3. After the first superframe (slot 7), the message forwarding continues along the path, e.g. the transition from the state $(7, 7, -)$ to the state $(8, 8, -)$. Every link has an extra opportunity to deliver the message to the gateway in the second superframe, which is represented by the transition from the state $(12, 12, -)$ to the state $(13, 13, -)$. This indicates that the long reporting interval, the high probability to reach goal states.
4.2. PATH MODEL

Figure 4.6: DTMC diagram of the path model of a three-hop path when $I_s = 2$
Chapter 5
Path Analysis and Discussion

After the introduction of the hierarchical path model that represents the age of messages at each hop of a path, in this chapter we focus on how to derive measures of interest from this model and how to use them to evaluate path performance. Firstly, in Section 5.1 the Quality of Service (QoS) measures used in this thesis are defined. Section 5.2 explicitly presents the methodology to derive such measures from the path model step by step. The analysis methodology is applied in an example case in Section 5.3. Such path analysis is facilitated by an automatic tool developed in Section 5.4. After that, Section 5.5 evaluates the influence of link availability and path hop number on the path performance. At last, Section 5.6 proves a theorem to predict the performance of a new conjunction path.

5.1 QoS Measures

The Quality of Service of a network is a multi-dimensional parametrized measure of how well the network performs this function [21]. These measures include:

- Speed and throughput of a network, i.e. the amount of data that can be transmitted in a time interval.
- Delay and jitter associated with data transmission. Delay is the duration it takes for a packet to make its way from the source to the destination, while jitter is the variation in delay between different communication cycles.
- Reliability and security of the network infrastructure

Among them, the latter two types of QoS are crucial to WirelessHART networks, because they can directly influence the control validity, stability and precision. Specifically, the following QoS measures of interest are used for performance evaluation in this thesis:

(a) Reachability $R_e$.

While in DTMCs ‘reachability’ is normally defined as the probability to reach a certain set of states before a given time $t$, in this thesis we use a slightly different notation based
on reporting intervals. In the following Reachability \( R_e \) denotes the probability that a message from the source node reaches the gateway before the end of a reporting interval. If a message fails to reach the gateway, then the input signal \( I \) (refer to Definition 1) is lost, causing severe instability to the control loop.

The messages in different reporting intervals are independent of each other. Then given the reachability for a single reporting interval, the time until the first message loss is geometrically distributed and the expectation is given by \( E[N] = 1/p = 1/(1 - R_e) \). For example, when \( R_e = 0.9 \), the input signal is expected to be lost every ten times. Hence, to make sure that the WirelessHART system is stable, \( R_e \) is required to be as close to 1 as possible.

(b) Delay distribution \( \tau \) and Expected delay \( E[\tau] \)
In WirelessHART, excessive delay extends the deadtime of control responses and impairs the stability of the control loop, thus it can lead to significant degradation in system performance. As shown in Figure 5.1, the delay in a WirelessHART network is defined as the time difference between the born time \( T_{born} \) and reception time \( T_{rec} \). In other words, it equals the message age in the Path model. So the delay distribution \( \tau \) can be easily derived from age probabilities. The expected delay \( E[\tau] \) is the mean value of \( \tau \).

(c) Utilization rate \( U \)
\( U \) indicates how many slots actually performed message transmission in a reporting interval, i.e. the busy time rate. The network communication overhead and power consumption are directly related to this utilization rate. It will be used to evaluate the network performance in the next chapter.

Figure 5.1: Delay and jitter of message transmission

All the above three QoS measures can be derived from the path model DTMC as explained in the following section.
5.2 Methodology

First of all, the age probabilities can be derived from the transient distribution of the DTMC model. Recall Equation 4.1, \( a_i \) represents the age of messages that reach the gateway in the \( i \)-th cycle. The cycle index \( i \) belong to the goal state set, i.e., \( i \in \psi \). The age probabilities are denoted as \( Pr(a_i) \) in this thesis. As a matter of fact, the age probabilities are the transient probabilities of the goal states, i.e.,

\[
Pr(a_i) = p_i(t), \; i \in \psi.
\]

Since we are interested in the messages at the end of every cycle, the time \( t \) equals the cycle length in the DTMC model. According to Section 4.2, \( t = I_s * F_{up} \). Recall the equation that was introduced in Section 3.1, the transient distribution is given by:

\[
p(t) = p(0)P^t = p(t-1)P, \tag{5.1}
\]

where \( p(0) \) is the initial distribution.

The path model starts with the initial state \((1, -, \cdots, -)\), thus the initial distribution is given by \( p(0) = [1, 0, \ldots, 0] \). Note that the transition probability matrix \( P \) contains dynamic transition probabilities \( p_s \) and \( p_f \), which probably change every time slot according to the link model status. Therefore, the entries of \( P \) have to be calculated in every step, and the matrix multiplication is performed according to Equation 5.1 to obtain \( p(t) \), e.g., \( p(1) = p(0)P, p(2) = p(1)P \) and so on. This method to obtain age probabilities is summarized in Algorithm 1.

---

**Algorithm 1** Generate transient distribution from a path model DTMC

---

**Require:** initial distribution \( p(0) \), transition matrix \( P \) with dynamic entries

**Ensure:** transient distribution \( p(t) \)

1. **for** time \( t = 1 : EndofInterval \) by step 1 **do**
   2. retrieve transient probability \( p_{up}(t) \) and \( p_{down}(t) \) from all link DTMCs
   3. fill \( p_{up}(t) \) and \( p_{down}(t) \) into matrix \( P \), get \( P_t \)
   4. \( p(t) = p(t-1)P_t \)
   5. **end for**
6. retrieve goal states probability \( p_i(t) \ (i \in \psi) \) from \( p(t) \)

---

**Definition 8 (Cycle probability function)** With age probabilities \( Pr(a_i) \) and cycle index \( i \in \psi \), the cycle probability function is defined as \( g(i) = Pr(a_i) \).

The major measure Reachability \( Re \) is then given by the sum of all cycle probabilities, i.e. the sum of all the transient probabilities of the goal states at the end of the interval.

\[
Re = \sum_i g(i) = \sum_i p_i(t), \; \text{where} \; i \in \psi, \; t = I_s * F_{up}. \tag{5.2}
\]
5.3 Example Analysis

As mentioned in Section 5.1, the delays \( d_i \) can be transformed from received message ages. The age measured in slots has to be converted to the absolute time in millisecond. Furthermore, the downlink duration \( T_{\text{downlink}} \) should be taken into account.

\[
d_i = (\text{age} + T_{\text{downlink}}) \times 10. \tag{5.3}
\]

For each delay \( d_i \), the delay probability is the percentage of messages with delay \( d_i \) among all the received messages, i.e. the averaged transient probability. This is given by:

\[
f_\tau(d_i) = \frac{p_i(t)}{\sum_i p_i(t)} = \frac{p_i(t)}{I_s \times F_{\text{up}}}, \quad t = I_s \times F_{\text{up}}. \tag{5.4}
\]

Therefore, the expected delay \( E[\tau] \) is defined as

\[
E[\tau] = \sum_{i \in \varphi} d_i \times f_\tau(d_i). \tag{5.5}
\]

Lastly, the utilization rate \( U \) can be derived from the age probabilities \( Pr(a_i) \) and reachability \( Re \) by the following approach. Consider an \( n \)-hop path, every message that reaches the gateway in the first cycle must have passed \( n \) links (i.e. \( n \) times transmission); every message that reaches the gateway in the second cycle must have passed \( n + 1 \) links (\( n \) times succeed, and 1 time failed); and so on. Note that the discarded message (with probability \( 1 - Re \)) should also be taken into account. Then counting all these link numbers together, the utilization rate of one path is given by:

\[
U_p = \frac{\sum_{i \in \varphi}[Pr(a_i) \times (n + i - 1)] + (1 - Re) \times (n + I_s - 1)}{I_s \times F_{\text{up}}}. \tag{5.6}
\]

And the overall utilization rate of the entire network is given by:

\[
U = \sum_p U_p. \tag{5.7}
\]

5.3 Example Analysis

In this part, we derive QoS measures for an example path using the above mentioned methodology. The instance is a three-hop path \( n_1^1 \rightarrow n_2^2 \rightarrow n_3^3 \rightarrow G \) with uplink framesize \( F_{\text{up}} = 7 \) and communication schedule \((*,*,(n1,n2),*,*,(n2,n3),(n3,G))\). The reporting interval is set as \( I_s = 4 \). For simplicity, all the links on the path are considered homogeneous, i.e., share the same transition probabilities. In this example, the link model transition probabilities are set as \( p_{fl} = 0.3, p_{cr} = 0.9 \). Moreover, assume that all links are already in steady states at time 0. By Equation 5.1 and Algorithm 1 that was developed in the last section, the transient probabilities for all goal states are derived and plotted in Figure 5.2. As can be seen, time (in slots) is indicated on the x-axis. Since \( I_s = 4 \), there are four goal states in the path model DTMC, i.e. \( S_\varphi = \{(R7),(R14),(R21),(R28)\} \). Correspondingly, there are four
possible ages at which messages can reach the gateway: \( a_1 = 7, a_2 = 14, a_3 = 21 \) and \( a_4 = 28 \). This is consistent with Equation 4.1 in Section 4.2.3. For example, since \( a_0 = 7 \) and \( F_{up} = 7 \), \( a_2 = 7 + 7 = 14 \). As the data labels tells, the message reaches the gateway at time 7 with probability 0.4219 (in blue), at time 14 with probability 0.3164 (in green), at time 21 with probability 0.1582 (in red).

As explained in Section 5.2, \( t = I_s * F_{up} = 4 * 7 = 28 \). The age probabilities, i.e. transient probabilities are:

\[
Pr(7) = p_{\varphi(1)}(28) = 0.4219, \quad Pr(14) = p_{\varphi(2)}(28) = 0.3164, \\
Pr(21) = p_{\varphi(3)}(28) = 0.1582, \quad Pr(28) = p_{\varphi(4)}(28) = 0.0659.
\]

For the measure of interests, according to Equation 5.2, the reachability is given by

\[
Re = \sum_{i \in \varphi} p_i(28) = 0.4219 + 0.3164 + 0.1582 + 0.0659 = 0.9624.
\]

And the discard probability equals \( 1 - Re = 0.0376 \). According to Equation 5.3, the delays are computed as \( d_1 = 70, d_2 = 210, d_3 = 350, d_4 = 490 \) (millisecond). Following Equation 5.4 yields the following delay probabilities:

\[
\begin{align*}
f_r(70) &= \frac{0.4219}{0.9624} = 0.4384, \\
f_r(210) &= \frac{0.3164}{0.9624} = 0.3288, \\
f_r(350) &= \frac{0.1582}{0.9624} = 0.1644, \\
f_r(490) &= \frac{0.0659}{0.9624} = 0.0684.
\end{align*}
\]
5.3. EXAMPLE ANALYSIS

Note that $\sum_i f_\tau(i) = 1$, proves that $\tau$ is a probability distribution. Figure 5.3 shows the delay distribution $\tau$ of this example path. By Equation 5.5, the expected delay is given by

$$E[\tau] = 70 \times 0.4384 + 210 \times 0.3288 + 350 \times 0.1644 + 490 \times 0.0684 = 190.8 \text{ (ms)}.$$ 

The utilization rate of this single path is computed according to Equation 5.6:

$$U_p = \frac{0.4219 \times 3 + 0.3164 \times 4 + 0.1582 \times 5 + 0.0659 \times 6 + 0.0376 \times 6}{4 \times 7} = 0.14.$$ 

So far we have derived the age probabilities, reachability, delay distribution and utilization rate from the DTMC model of the example path. All the results including Figure 5.3 reveal the following findings:

- The reachability $Re = 96.24\%$ is close to 1. But there is a small possibility that the sensory message cannot reach the destination within the reporting interval. As mentioned in Section 5.1, on average one out of $1/(1 - 0.9624) = 26.6$ messages will fail to reach the gateway, causing the loss of the control-loop input signal.

- The message delays take discrete values and the jitter is always a multiple of the uplink framesize $F_{up}$. This is because the last link connecting the gateway can only transmit at the same slot in the superframe schedule.

- Figure 5.3 shows that $f_\tau(70) > f_\tau(210) > f_\tau(350) > f_\tau(490)$. In general, the probability for a short delay is higher than the probability for a long delay, which shows that the most of the messages reach the gateway in the first cycle.
5.4 Analysis Tool

To facilitate the analysis of the hierarchical path model, we have developed a tool to automatically derive the underlying DTMC of a WirelessHART path, which is referred to as ‘WirelessHART Analyzer’. It was developed in Java SE Runtime Environment version 1.6 using the Eclipse Indigo platform.

The analyzer is not only capable to build the path model DTMCs, but also to derive the QoS measures by the methodology illustrated in Section 5.2. This can be seen from the function block diagram in Figure 5.4. As is shown, the analyzer takes input parameters including the path hops $n$, link transition probabilities and communication schedule $\eta$; outputs the values of measures such as delay distribution, reachability and path utilization rate. It consists of three internal modules to process the model step by step. At the first place, a hierarchical model, including path DTMC and link DTMC, is constructed using all the input parameters. Secondly, the transient distribution of the model in a reporting interval ($I_s = 4$ by default) is derived using Algorithm 1. Finally, the QoS measures are calculated and outputted, either in raw data or through a graphic interface (refer to Section 6.2).

In the first function block, an algorithm is developed to construct the DTMC corresponding to the rules in Section 4.2.3. For instance, a recursive function ‘ConstructForward()’ is the core method that builds the complete path DTMC starting from the initial state. Algorithm 2 gives the pseudo code of this function. Lines 5-6 follow rule (d) in Section 4.2.3 to reach the ‘Discard’ state; Lines 8-10 follow rule (c) to create failure states with transition probability $p_f$, which is derived from the sending link model. Lines 15-17 construct success states with transition $p_s$, which is derived from the sending link model too. Lines 22-24 follow rule (b), dealing with slots where no transmission take place. This method implements state transitions and the two-tier dependency in the path model.

The analyzer is used to obtain QoS measures in the following discussion in Section 5.5 and evaluation in Chapter 6, where also a graphic interface is presented to directly present the analysis outputs.
Algorithm 2 A recursive function to construct age model DTMC

Require: initial state $s_i$
Ensure: a complete age model DTMC

1: function ConstructForward state $s_c$
2:  if $s_c$ is to transmit message then
3:      identity transmission link $n$
4:         \triangleright$\text{firstly construct failure state}$
5:  if $TTL \leq 0$ then
6:      add transition to the ‘discard’ state return
7:  end if
8:  new a state $s_{fail}$ with age $+ 1$
9:  add transition from $s_c$ to $s_{fail}$ with probability $p_f = p(down)$ of link $n$
10:  ConstructForward($s_{fail}$)
11:     \triangleright$\text{then construct success state}$
12:  if $n$ is the last link leads to gateway then
13:      add transition to the goal state with age return
14:  end if
15:  new a state $s_{suc}$ with age $+1$ and copied to the next link $n + 1$
16:  add transition from $s_c$ to $s_{suc}$ with probability $p_s = p(up)$ of link $n$
17:  ConstructForward($s_{suc}$)
18:  else \hspace{1cm} \triangleright$\text{No transmission attempt}$
19:  if $TTL \leq 0$ then
20:      add transition to the ‘discard’ state return
21:  end if
22:  new a state $s_{next}$ with age $+1$
23:  add transition from $s_c$ to $s_{next}$ with probability $p = 1$
24:  ConstructForward($s_{next}$)
25:  end if
26: end function
5.5 Influential Factors

In this section, we investigate two factors that possibly influence the QoS of a path: the link availability and the path hop number.

5.5.1 Link availability

A $n$-path link consists of $n$ individual links, some of them are shared with other paths in the network. For convenience of analysis, we firstly discuss the case that all path links are homogeneous, i.e., share the same transition probabilities and the case with heterogeneous links will be investigated later on.

As illustrated in Section 4.2.4, in the hierarchical DTMC model, the dynamic transition probabilities $p_s$ and $p_f$ of the path model rely on the transient link availability $p_{\text{link}}(t)$ at that transmission slot. Recall Equation 4.2 and combine it with the link model transition probability matrix in Equation 3.5, we obtain

$$ [p_s(t), p_f(t)] = p(t) = p(0) \begin{bmatrix} 1 - p_{fi} & p_{fi} \\ p_{rc} & 1 - p_{rc} \end{bmatrix}^t. $$

Especially, if the links are in steady-state during transmission, then

$$ [p_s, p_f] = [\pi(up), \pi(down)] = \begin{bmatrix} \frac{p_{rc}}{p_{rc} + p_{fi}} & \frac{p_{fi}}{p_{rc} + p_{fi}} \end{bmatrix}. \quad (5.8) $$

![Figure 5.4: Analyzer tool block diagram](image-url)
5.5. INFLUENTIAL FACTORS

Under such circumstances, the transition probabilities $p_s$ and $p_f$ are reduced to the stationary link availability $\pi(\text{up})$, which is derived from the link model transitions $p_{rc}$ and $p_{fl}$. As specified by standard [26], the 2.4 GHz frequency band is divided into 16 non-overlapping frequency channels. WirelessHART instruments use a pseudo-random channel hopping to reduce the interference with other networks, such as IEEE802.11b/g (Wi-Fi) which operates in the same ISM frequency band. In other words, whenever the link suffers a bad frequency channel, it will hop to a new channel in the next slot. And this new channel has a high probability to be up, because the network manager maintains a list of active channels. All the down channels are banned to the blacklist after a certain period of time. However, there is still a little probability that the new channel is not working either. Therefore, in the corresponding link DTMC, the recovery transition probability $p_{rc}$ should be very close to 1, but not equal to 1. In this work, we choose $p_{rc} = 0.9$, which could be easily adjusted according to real practice. As a result, the link availability is determined by only one variable: the link failure probability $p_{fl}$. As discussed in Section 3.4, the correlation between $p_{fl}$ and the bit error rate is expressed in the following equation:

$$p_{fl} = 1 - (1 - BER)^L.$$  

According to WirelessHART standard specification [33], a typical WirelessHART MAC layer payload length is 127 bytes, i.e. $L = 127 \times 8 = 1016$. In this manner, the link availability can be determined by the bit error rate in WirelessHART channels. For instance, if $BER = 1 \times 10^{-4}$, using the above Equation 5.8, we obtain $p_{fl} = 0.0966$ and the stationary link availability $\pi(\text{up}) = 0.9031$. Hence, the lower the bit error rate, the lower the failure probability, which leads to a higher link availability.

Now take a three-hop path with steady-state links as example. We investigate the reachability and delay distribution of the same path with different link availabilities. This can happen when the links work in different channels, with different bit error rates (BER). Figure 5.5 shows the two measures of interest in separate plots. Firstly, Fig. 5.5(a) shows the reachability $Re$ of this path under different stationary link availabilities $\pi(\text{up})$, ranging from 0.69 to 0.95. The reachability rises and converges to 1 with the link availability. When $\pi(\text{up}) < 0.75$, the $Re$ is below 95% and probably degrades the networks performance and control loop stability. When $\pi(\text{up}) > 0.9$, the $Re$ is above 99.8%, indicating excellent performance and timely delivery. Secondly, Fig. 5.5(b) compares the delay distributions $\tau$ with different link availabilities. The delays are the same, but their probabilities vary. A higher link availability leads to a steeper and more concentrated delay distribution; while a lower link availability results in a flatter distribution with a longer tail. Specifically, when $\pi(\text{up}) = 0.948$, 98.5% of the messages have a delay that is shorter than 200ms and those with longer delays can be neglected. In contrast, when $\pi(\text{up}) = 0.774$, only 77.8% of the messages have a delay shorter than 200ms and more than 5.3% of the messages have a delay as long as 470ms, which may be unacceptable in some control systems. From this point of view, high link availability brings smaller jitters, as well. The expected (mean) delays $E[\tau]$, calculated using Equation 5.5, are listed in Table 5.1. From this, the effects of link availability can be seen clearly.
To conclude, by the comparison of DTMC model analysis results, the link availability shows its significant influence on the path performance. A high link availability is desirable in order to achieve a high message reachability and short delays.
5.5. **INFLUENTIAL FACTORS**

Table 5.1: Influence of $\pi(up)$ on the reachability and expected delay

<table>
<thead>
<tr>
<th>Link availability</th>
<th>0.774</th>
<th>0.83</th>
<th>0.903</th>
<th>0.948</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reachability (%)</td>
<td>97.37</td>
<td>99.07</td>
<td>99.89</td>
<td>99.99</td>
</tr>
<tr>
<td>Expected Delay (ms)</td>
<td>179</td>
<td>151</td>
<td>113</td>
<td>93</td>
</tr>
</tbody>
</table>

5.5.2 **Path hop number**

Apart from the link availability, the other influential factor is the number of hops $N$ on a path. During the initialization of a WirelessHART network, field devices are self-organized. If a node is located far away from the gateway’s access point, it needs more intermediate hops to relay. According to the official guideline [32], the maximum distance from a node to the gateway in WirelessHART should not exceed 4 hops. This is meant to guarantee that networking delays do not harm control performance.

Therefore, under the constraint of $N \leq 4$, we analyze a one-hop path, a two-hop path, a three-hop path and a four-hop path. Assuming similar path links with stationary link availability $\pi(up) = 0.83$, we investigate the reachability probabilities of the four paths, as shown in Figure 5.6. It can be seen that the one-hop path has the highest reachability of 0.9992. With more hops, the reachability decreases, and for the four-hop path, it finally drops to 0.9812. This is because a larger hop number results in a higher probability of a transmission failure along the way. Hence the reachability decreases with the hop counts. These finding suggests that it is beneficial to minimize the path hop number in a WirelessHART network so as to ensure control loop stability. The influence of the hop number on the delay distribution will be discussed in Section 6.3.2.

![Figure 5.6: The influence of path hop number on reachability](image-url)
5.5.3 Communication schedule

As implied in the example above, the communication schedule has little impact on the reachability of a single path and on the delay distribution. However, it plays a role in shaping the overall delay of a network. This effect will also be studied in Chapter 6.

5.6 Path Conjunction

The hierarchical path model describes end-to-end message delivery. Normally, one end is the gateway. If both ends are field devices, a peer-to-peer communication is performed on the second type of path, which is referred to as a peer path in this thesis. All the modeling and analysis methodologies work in a similar manner on peer paths, e.g. age probabilities $Pr(a_i)$ can be derived as well. A new path can be formed by the conjunction of an existing path with a peer path if they share one end. An example is shown in Figure 5.7. The peer path from node 5 to node 3 is connected to the existing path from node 3 to the gateway, thereby forming a new path from node 5 to the gateway.

Before the conjunction, there are two old path DTMC models; after it, a new path model can be established. However, this is unnecessary, because the cycle probabilities (recall Definition 8) of the new path can be derived using the old models according to the following theorem.

**Theorem 4 (Conjunction path cycle probabilities)** The conjunction path's cycle probability function $g_c(x)$ is the convolution of cycle probability functions $g_e(x)$ and $g_p(x)$ of the old paths, time-shifted by one.

$$g_c(x + 1) = g_e(x) \ast g_p(x) = \sum_{i=0}^{\infty} g_e(i) g_p(x - i), \text{i.e.}$$

$$g_c(x) = \sum_{i=0}^{\infty} g_e(i) g_p(x - 1 - i).$$

**Proof:** Assume that the message reach the end of the peer path in the $m$-th cycle. In the same cycle, the forwarding is continued along the existing path towards the gateway. If it takes $n$ cycles to reach the gateway, then the message reach the destination in $m + n - 1$ cycles in total after generation. The cycles of the existing path and the peer path are independent.
of each other. According to probability theory, the probability of the sum of two independent random variables is the convolution of their individual probabilities.

With the cycle probability function, the new path reachability can be derived using Equation 5.2. Applying Theorem 4 to predict path reachability is useful in at least two scenarios: a) In network routing, for the node with multiple existing paths to connect, the reachability prediction helps to decide which route should be preferred. b) In dynamic topology, when a new node joins the network, it has to select an existing node as the first option to link to. The reachability prediction can provide a reasonable preference on one of those options.
Chapter 6

Network Performance Evaluation

In Chapter 5, we mainly analysed the performance of a single path. In this chapter, we look at the big picture instead of only the parts. First of all, we propose a typical WirelessHART network in Section 6.1 and then evaluate its QoS measures such as reachability, delay distribution and utilization rate in Section 6.3. After that, a scheduling principle is presented in Section 6.4. In addition, the system robustness against three types of link failures will be assessed in Section 6.5. Section 6.6 elaborates the influence of reporting interval on the performance by investigating a fast control case. Finally, Section 6.7 shows the application of the analyzer on performance prediction.

6.1 The Typical Network

Similarly to the paths, we take a typical network instance as the evaluation target. As the HART Communication Foundations claims, in real plant settings, on average 30% of the nodes communicate directly with the gateway access points and about 50% are two hops away. The remaining 20% may be 3 or 4 hops away. Using this ratio, a typical WirelessHART network is established as Figure 6.1, which consists of ten nodes and a gateway.

As shown in the connectivity graph $G$, there are ten nodes in $N = (n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9, n_{10})$. Every node connects to another node or the gateway with a bi-directional wireless link $e_i$ in $E$. The network suits Assumption 1, so that the uplink and downlink are symmetric. Each node serves as a source node, inserts messages to the network. At the same time, it can be an intermediate node that forwards other messages to the destination, if necessary. In this manner, a chain of nodes and links form a path to the gateway. According to the mentioned ratio, in the typical network, there are 3 one-hop paths, 5 two-hop paths and 2 three-hop paths, which are represented in different colors in Fig. 6.1. These paths require that the uplink framesize $F_{up}$ should be at least 19 slots ($3 \times 1 + 5 \times 2 + 3 \times 2$). From the other point of view, the network topology is a tree with the gateway as the root. The child nodes are situated at different depth. A node’s connecting link is shared with its child nodes, e.g. the link $e_1$ is shared with $n_4$ and $n_5$. Thus, the communication traffic on the first hops should be much heavier than the second and third hops.
6.2 Analyzer Graphical Interface

Recall that the analysis tool, as presented in Section 5.4 has been developed and used for path evaluation. For WirelessHART network, the same tool can be extended to derive path and overall QoS measures. Moreover, the user-tool interaction is improved by a graphical interface. Figure 6.2 shows an example screenshot.

As can be seen, the interface window is separated into two parts. The right-side part is a control console, while the left-side part contains a dynamic network diagram, where circles represent network nodes and arrows represent the wireless links similar to Fig. 6.1. The links are labelled with the current link availabilities, and the nodes are filled in different colors to indicate the reachability probabilities on paths sourced from those nodes. (E.g. green node when $Re > 96\%$, yellow node when $90\% < Re < 96\%$, red node when $Re < 90\%$). Link availability can be either adjusted by the scale at the top of the console, or specified in the link labels. After the parameters are set, users press the button to start the automatic analysis process, which executes the functions shown in Fig. 5.4. Among the output measures, path reachability is implied by the node color, utilization rate is shown at the bottom of the console and a progress bar, as well. The delay measures are not presented in this interface, but outputted into raw data files.

Take a look at the example network that is analyzed in Figure 6.2, the availabilities of link 3 and link 10 are different from other links. The low availability (0.6) leads to low
reachability probabilities on the related four paths at the right side. This example shows that our model is capable to analysis heterogeneous networks.

6.3 Overall Performance

In Chapter 5, we examined two QoS measures of a path, especially with different link availabilities. In this section, we evaluate the performance of the example network as shown in Fig. 6.1 by three overall measures: Reachability, Overall Delay Distribution and Utilization Rate.

It is worth mentioning that these overall measures play different roles in the assessment. Among them, the reachability is the prime measure of interest and is the most valued in this thesis. As explained before, without a sufficiently high (depends on the real applications) message reachability, the control-loop stability cannot be guaranteed, and the WirelessHART system may be out of order. So the criterion is that the network with high a reachability outperforms those with a low reachability. In case the reachability probabilities are close, overall delay can be used to tell the difference.

6.3.1 Reachability

In every reporting interval, ten distinct messages containing sensory data on the devices are forwarded to the gateway. As explained in Section 6.1, assume the superframe size $F_s$ is 20
and the communication schedule is \( \eta_a = (\langle n_1, G \rangle, \langle n_2, G \rangle, \langle n_3, G \rangle, \langle n_4, n_1 \rangle, \langle n_1, G \rangle, \langle n_5, n_1 \rangle, \langle n_1, G \rangle, \langle n_6, n_2 \rangle, \langle n_2, G \rangle, \langle n_7, n_3 \rangle, \langle n_3, G \rangle, \langle n_8, n_3 \rangle, \langle n_3, G \rangle, \langle n_9, n_6 \rangle, \langle n_6, n_2 \rangle, \langle n_2, G \rangle, \langle n_{10}, n_7 \rangle, \langle n_7, n_3 \rangle, \langle n_3, G \rangle) \). (For scheduling, refer to Section 6.4.) Each message has a positive probability to reach the gateway at the end of reporting interval, so the main measure reachability \( Re \) remains the same as defined for a path. Figure 6.3 demonstrates the reachability probabilities on all the ten paths, separately. The sequences of the paths are labelled by their source nodes. The reachability probabilities with different link availabilities are represented in four colors. One can see that the more hops a path has, the lower reachability it bears. However, when the link availability equals \( \pi(up) = 0.9 \), messages reach the gateway with the probability \( Re > 0.999 \) even on the three-hop paths. On the opposite, if the links suffer from a large bit error rate during transmission, causing the \( \pi(up) \) to drops to 0.69, the reachability consequently falls to 92.5%, which means one message loss out of 13 on average. This loss probably threats the stability of the corresponding control-loop and furthermore compromise the whole WirelessHART system. In conclusion, the longest path with the lowest link availability becomes the bottleneck of the network. The improvement of bottleneck paths can efficiently optimize the network performance.

### 6.3.2 Overall delay distribution

In Section 5.1, the delay distribution \( \tau \) for a path is defined. Similarly, for the entire network, the Overall Delay Distribution \( \Gamma \) can be derived by additive averaging of all path delay distributions \( \tau \) as follows. The delays equal \( D_i = d_i \). The delay probabilities are given by

\[
    f_\Gamma(d_i) = f_\tau(d_i)/j, \tag{6.1}
\]

Figure 6.3: The reachability of all paths in the typical WirelessHART network
where \( j \) represents the total number of paths. E.g., in the example network, \( j = 10 \). The overall mean delay \( E[\Gamma] \) is defined as the average of all expected delays. Here, all the message delays are weighted the same when averaged.

\[
E[\Gamma] = \frac{\sum_i E[\tau_i]}{j}. \tag{6.2}
\]

Again, with the same communication schedule and stationary link availability \( \pi(up) = 0.83 \), the example WirelessHART’s overall delay distribution is derived according to Equation 6.1 and shown in Figure 6.4. The graph reveals that:

- It shows the global picture of how messages reach the gateway in the entire network. Because strict TDMA is used, different messages arrive at different time slots. The blanks between arrivals represent the slots that are used for downlink traffic.

- As observed in Section 5.3, generally the probability of a short delay is higher than the probability of a long delay. In addition, the figure shows that less hop numbers of a path leads to a steeper and concentrated delay distribution. Hence, the subtotal delay probabilities within one cycle decline linearly with time. E.g. in the distribution graph, 70.8% of the messages reach the gateway in the first cycle while only 21.7% of them do so in the second cycle. In other words, 92.6% of the messages reach the gateway at the end of the second cycle (600ms) and approximately 98.3% reach it by the end of the third cycle (1000ms). This provides the possibility of fast control with shorter reporting intervals, which will be discussed in Section 6.6.

- This figure could serve as the traffic graph of the gateway, because the time a message reaches the gateway is the same time that the gateway receives it. So it implies the gateway’s traffic distribution. It is apparent that the workload is imbalanced. At the beginning of every interval, the gateway receiver is almost fully occupied, and then gets less busy with more idle slots as time passes.

Moreover, the expected delay of the ten paths \( E[\tau] \), are listed in Figure 6.5. Derived from \( E[\tau] \), according to Equation 6.2, the overall mean delay \( E[\Gamma] \) is 235 milliseconds. Fig. 6.5 clearly shows that the mean delays on the paths vary a lot. Take a look at path 10, it is the worst case with a delay expected to be 421 milliseconds, which is almost a twice of the overall mean delay 235. This bottleneck can be eliminated by appropriate scheduling as discussed in Section 6.4.

### 6.3.3 Utilization rate

As mentioned in Section 5.1, the utilization rate \( U \), which is the percentage of time slots that actually performed wireless transmission, indicates transmission cost of the network nodes. As pointed out by Heinzelman [34], the energy consumption of wireless radio transmission dominates all the node power consumption because those energies used for sensing and CPU
6.3. OVERALL PERFORMANCE

![Graph showing overall delay distribution](image1)

**Figure 6.4:** The overall delay distribution of the example WirelessHART network

![Graph showing expected delays](image2)

**Figure 6.5:** The expected delays of all paths with the schedule $a$

Computing is relatively low. So the utilization rate is regarded as an indicator of the network power consumption.

Taking advantage of Equation 5.6 and Equation 5.7, the utilization rate $U$ for the example
network is derived. Table 6.1 lists the utilization rate with different link availabilities. From it, e.g., when \( \pi(up) = 0.693 \), \( U = 0.313 \), and when \( \pi(up) = 0.989 \), \( U = 0.24 \), i.e. the worse link channel yields \((0.313 - 0.24) / 0.24 = 30\% \) more transmission overhead than the good link. It means that bad link channels not only degrade the control stability but also bring more communication overhead and power consumption to the network.

<table>
<thead>
<tr>
<th>Link availability</th>
<th>0.693</th>
<th>0.774</th>
<th>0.83</th>
<th>0.903</th>
<th>0.948</th>
<th>0.989</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utilization rate</td>
<td>0.313</td>
<td>0.297</td>
<td>0.283</td>
<td>0.263</td>
<td>0.25</td>
<td>0.24</td>
</tr>
</tbody>
</table>

6.4 Scheduling

The aim of scheduling is to generate the best schedule for a network in order to achieve optimal performance. In our WirelessHART system, one channel among the active channel list is dynamically allocated in every transmission. Different links may use the same or different frequency. This scheme is called ‘per-transaction channel hopping without channel reuse’ in [12]. On the opposite, some scholars study the other scheme that assigns two or more channels to part of the network so that some links could transmit at the same slot in different frequency channel. This is referred to as ‘multi-channel scheduling’. Its advantage is to shorten the communication schedule, and hence shorten the control loop. In that case, scheduling is not only about time, but also includes channel scheduling. However, in this thesis we only consider the former scheme, since it strictly follows the TDMA specification and allows designing simple scheduling policies without constructing interference graphs.

Recall that when introducing the path model DTMC, we mentioned that the smallest possible message age for a path depends on when the last transmission slot is scheduled (Equation 4.1). Since the communication schedule \( \eta \) coordinates the arrival time of messages, it can directly influence the expected delays of paths. However, scheduling does not change the message reachability \( Re \), because the probability and times of transmission remain the same. In this thesis, the schedule coordinates transmissions by prioritizing different paths. The path with high priority forwards its message prior to that with low priority.

In Section 6.3, the example network works with the communication schedule \( \eta_a \). Now we can see the idea of \( \eta_a \) is to offer high priority to those paths with less hops, e.g. let the one-hop paths transmit ahead of two-hop paths. The overall delay distribution and expected delays with \( \eta_a \) are shown in Section 6.3.2. The opposite order of scheduling produces the communication schedule \( \eta_b = (\langle n9, n6 \rangle, \langle n6, n2 \rangle, \langle n2, G \rangle, \langle n10, n7 \rangle, \langle n7, n3 \rangle, \langle n3, G \rangle, \langle n4, n1 \rangle, \langle n1, G \rangle, \langle n5, n1 \rangle, \langle n1, G \rangle, \langle n6, n2 \rangle, \langle n2, G \rangle, \langle n7, n3 \rangle, \langle n3, G \rangle, \langle n8, n3 \rangle, \langle n3, G \rangle, \langle n1, G \rangle, \langle n2, G \rangle, \langle n3, G \rangle), \) where the long hop paths are offered a high priority. The new overall delay distribution of the example network is shown in Figure 6.6. Comparing Fig. 6.6 with Fig 6.4, one can find that the priority change is directly reflected in delay distributions. With schedule \( \eta_b \), the gateway traffic is slightly more balanced than with schedule \( \eta_a \).
To better identify the differences caused by scheduling alternatives, the expected delays $E[\tau_i]$ are compared in Figure 6.7. In addition, the new overall mean delay is computed as
$E[\Gamma]_b = 272$ milliseconds. In Fig. 6.7, the expected delays of schedule $\eta_b$ are obviously more balanced than that with $\eta_a$, since most paths delay are at the same level. The old bottleneck path 10 is eliminated ($E[\tau]_{10}$ drops from 421 to 291). Instead, the longest delay in schedule $\eta_b$ is on the path 7 with $E[\tau]_7 = 317 < 421$. Although $E[\Gamma]_b = 272$ is slightly larger than $E[\Gamma]_a = 235$, the second schedule $\eta_b$ is considered better than $\eta_a$ in most cases, because it achieves a global balance of delay performance. Therefore, a scheduling principle can be concluded as ‘to offer high priority to the paths with larger expected delay is beneficial for the network performance optimization’.

So far, in the example network evaluation, all links are assumed to have the same stationary link availability. Imagine the case that two paths with the same hop length have different link conditions. For example, path 4 and path 5 in Fig. 6.1 use their own links $e_4$ and $e_5$ connecting to node 1. For those two, if $\pi(up)_4 > \pi(up)_5$, the scheduling has to determine whom to offer a higher priority. According to the discussion in Section 5.5.1, path 5 originally involves a larger expected delay i.e. $E[\tau]_5 > E[\tau]_4$. Under such circumstances, applying the scheduling principle as concluded above, path 5 should be scheduled prior to path 4.

6.5 Stability and Robustness

As introduced in Section 4.1.1, there are three categorizes of link failures in multi-hop control networks: transient error, random time span failure and permanent failure. In WirelessHART, these three types of failures would be caused by different reasons. In this section, we examine these cases and their influence on system stability, one by one.

6.5.1 Transient error

When a channel suffers from strong noises or co-exists with other wireless network such as WiFi, the strong signal interference produces a large bit error rate, and makes it impossible to transmit any package correctly. As a consequence, the link breaks down in this time slot. However, due to frequency hopping, the link probably recovers in the next slot. Hence, this type of failure can be seen as transient. Figure 6.8 shows how a link recovers from such transient errors. The black curve represents the case with transition probability $p_{fl} = 0.05$ and the red curve represents that when $p_{fl} = 0.184$. As explained in Section 5.5, the recovery probability is always considered to be 0.9. One can see that, in both cases, the link returns to its steady-state at the next second slot after the transient error.

Because different links transmit in different TDMA slots, the transient error that happens in one of them will not influence others at all. And the quick recovery implies that transient errors usually have little effect on the network performance. However, the exception is that the error coincidently occurs in the transmission slot of a link. In that case, the transmission is born to fail and must wait for the next cycle, so the reachability of that message is delayed by one cycle. The loss of one cycle yields a slightly lower reachability than the normal case.
6.5. Stability and Robustness

Figure 6.8: Link recovery from a transient failure

(approximately 0.2% to 2% according to Fig 6.4). This shows that the WirelessHART is stable and robust against transient errors.

6.5.2 Random failure

Unlike frequency channel interferences, temporary physical obstruction (losing Line of Sight) causes link failures for a random period of time. Frequency hopping does not help in this case. Consider the example network, as mentioned in Section 6.1, the the links take different workload of communication traffics. E.g. the link \( e_3 \) (connecting \( n_3 \) and the Gateway) is shared by four paths (3, 7, 8, and 10). If it suffers a random failure, all the four paths will be affected. In this worst case, assume the failure lasts about one cycle (400 milliseconds), Table 6.2 lists the path reachability probabilities with random failure, and compares it with those with normal links. From the table, it can be seen that the reachability of path 3 \( R_{e_3} \) falls slightly from 99.92% to 99.51%; \( R_{e_7} \) and \( R_{e_8} \) drop from 99.64% to 98.3% and \( R_{e_{10}} \) drops from 99.07% to 96.28%. Other paths that do not use the failed link \( e_3 \) are not affected. From the perspective of delay, all the delay distributions of the affected paths are time-shifted by one cycle. Hence, those messages that reach the gateway in the last cycle can not make it under the random link failure. This explains why the reachability \( R_{e_{10}} \) on the three-hop path decreases more than \( R_{e_3} \) on the one-hop path. If the random failure lasts even longer (i.e. 2 or 3 cycles), it will definitely degrade the performance more severely. Therefore, random link failures probably impair the system’s robustness and control loop stability.
Table 6.2: The reachability probabilities with a random link failure lasting one cycle

<table>
<thead>
<tr>
<th>Path</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hop number N</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Reachability (%)</td>
<td>99.92</td>
<td>99.92</td>
<td>99.92</td>
<td>99.64</td>
<td>99.64</td>
<td>99.64</td>
<td>99.64</td>
<td>99.64</td>
<td>99.07</td>
<td>99.07</td>
</tr>
<tr>
<td>Reachability with link failure</td>
<td>99.92</td>
<td>99.92</td>
<td>99.51</td>
<td>99.64</td>
<td>99.64</td>
<td>99.64</td>
<td>98.30</td>
<td>98.30</td>
<td>99.07</td>
<td>96.28</td>
</tr>
</tbody>
</table>

6.5.3 Permanent link failures

When the link failure duration is long compared to the control loop or reporting interval, it is regarded as permanent. Under such circumstances, it cannot be solved by the current routing graph. However, the failed link need to be removed from the routing graph, and the messages should be routed to other intermediate nodes to establish new paths heading for the gateway. Another alternative countermeasure may be to identify the cause of the failure and to repair it (e.g. remove the obstacle physically).

6.6 Reporting Interval and Fast control

In the previous evaluation, the reporting interval $I_s = 4$ was used for all paths. From the delay distribution in Fig. 6.4, one can identify the longest delay to be 1400ms, which may be not acceptable in some control scenarios. In this section, alternative $I_s$ values will be discussed to speed up the control response.

To make it more concrete, take a one hop path as example. Considering the unique link with $\pi(\text{up}) = 0.903$, Figure 6.9 shows the reachability probabilities of all received messages, which are labelled at its initial cycle (represented by blocks). Observing consecutive four cycles, when the reporting interval is one, every cycle produces a message that reaches the gateway with probability 0.903. When the reporting interval becomes 2, two messages that are generated at the first and third cycle separately, can be collected with probability 0.99 during the same period. Lastly, when the reporting interval is four, only one message that is generated at the first cycle may be received with probability 0.999. It is evident that the longer the reporting interval, the less messages received by the controller, but the higher reachability on each of them. Thereby, a shorter reporting interval is shown to be efficient to speed up the control loop, and provides fresher data for real-time monitoring. However, it involves more communication and power overhead to the system as well. Therefore, it is important to achieve a good balance by selecting an appropriate $I_s$ according to real application requirements.

Note that usually one $I_s$ value is used across the network to obtain a unified management. However, to improve flexibility, it is also feasible for paths to work with different values of $I_s$. Consider a fast control scenario when $I_s = 2$, which means that one control loop lasts only two cycles and the reporting frequency is doubled, compared to the regular control discussed.
6.6. REPORTING INTERVAL AND FAST CONTROL

Again for the example network, the reachability probabilities with fast control are analyzed and represented as a mesh in Figure 6.10. The reachability probabilities with regular control are included as contrast data (in blue). It can be seen that the reachability probabilities with fast control (in red) are lower than those with regular control, and the difference between the reachability probabilities with different reporting intervals increases when the link availability drops. E.g. for the path 1, when $\pi(up) = 0.948$ the reachability difference is $0.27\%$; while $\pi(up) = 0.774$ the reachability difference becomes $4.85\%$. Similarly, the reachability differences increase when the path hop number extends. For instance, when $\pi(up) = 0.948$, the reachability difference for one-hop path is $0.27\%$; and the difference for three-hop path (e.g. path 10) grows to $1.51\%$. This is because either low link availability or large path hops yields a flatter delay distribution and a long tail of delay. In fast control, those messages that are delayed longer than two cycles are discarded. Such message loss causes the reachability difference and these two trends.

In fact, not only the performance degrades, but also communication cost rise in fast control. Comparing utilization rates in Table 6.3 with previous Table 6.1, the utilization rate in fast control increases sharply, up to 100%. This indicates a double communication expense on field devices. In principle, a path can work in fast control only when the performance degradation and cost is within acceptance. From the data in Figure 6.10, this may be possible only for one-hop paths when link channels are in good conditions.

![Figure 6.9: Messages reach situation with different reporting intervals](image)

Table 6.3: Utilization rates of the example network in fast control

<table>
<thead>
<tr>
<th>Link availability</th>
<th>0.693</th>
<th>0.774</th>
<th>0.83</th>
<th>0.903</th>
<th>0.948</th>
<th>0.989</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utilization rate</td>
<td>0.426</td>
<td>0.468</td>
<td>0.488</td>
<td>0.497</td>
<td>0.493</td>
<td>0.48</td>
</tr>
</tbody>
</table>
### 6.7 Performance Prediction

In this section, we consider the scenario where a new node joins the network and show how to make routing decisions through performance prediction, using the theorems developed in previous sections.

The scenario is presented in Figure 6.11. In this WirelessHART network, node 3 connects to the gateway along path 1, which has $m$ hops; node 4 connects to the gateway along path 2, which has $n$ hops. When a new node 5 joins the network topology, it has to establish a route through the mesh network to the gateway. This can be done by connecting to an existing path within its communication range, e.g. path 1 or path 2. Recall the peer path definition in Section 5.6, the link between node 5 and node 3 yields a 1-hop peer path, and the new path from node 5 to the gateway is a conjunction path of the peer path and the existing path. (For the link between node 5 and node 4, it is the same.) Since the performance of the existing paths can be either measured in the real system or analyzed by the proposed analyzer, we are capable to predict the performance of the conjunction path by Theorem 4. However, before the peer path (e.g. path 3) has been established, the cycle probabilities of the peer path $g_p(x)$ is unknown. Nevertheless, it is possible to estimate the peer path cycle probabilities by pilot packages in the following way.

According to Chapter 5, the performance of a 1-hop path is determined by the transition probabilities of its unique link DTMC. Since the recovery transition probability $p_{rc}$ is justified as a fix value, the performance solely depends on the failure transition probability $p_{fl}$. And then according to Equation 3.7 and Equation 3.8, the probability $p_{fl}$ can be derived from the received Signal-to-noise ratio. It is convenient to measure the received SNR by transmitting pilot packages via the link. For example, in the above case, pilot packages can be sent from

![Figure 6.10: Reachability probabilities of paths with $I_s = 2$ and $I_s = 4$](image)
6.7. PERFORMANCE PREDICTION

Figure 6.11: The scenario of a new node joins the network

node 3 to node 5. Once both link transition probabilities $p_{rc}$ and $p_{fl}$ are known, the cycle probabilities of the peer path $g_p(x)$ can be derived according to the methodology in Section 5.2. To sum up, the predicted work flow can be represented as follows.

$$\text{SNR} \implies \text{BER} \implies p_{fl} \implies g_p(x) \implies \text{conjunction path } g_c(x)$$

$$\text{Eq. 3.8} \quad \text{Eq. 3.7} \quad \text{Eq. 3.8}$$

Table 6.4: Example of performance prediction by path conjunction

<table>
<thead>
<tr>
<th>Peer path</th>
<th>Existing path</th>
<th>Conjunction path</th>
<th>Reachability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_3(x)$</td>
<td>$g_1(x)$</td>
<td>$g_\alpha(x) = [0.6274, 0.2694, 0.0784, 0.0193]$</td>
<td>$R_{e\alpha} = 99.46%$</td>
</tr>
<tr>
<td>$g_4(x)$</td>
<td>$g_2(x)$</td>
<td>$g_\beta(x) = [0.6573, 0.2485, 0.0707, 0.0180]$</td>
<td>$R_{e\beta} = 99.45%$</td>
</tr>
</tbody>
</table>

Coming back to the example, we assume that path 1 involves 2 hops and path 2 involves 1 hop, and their links have the same stationary availability $\pi(\text{up}) = 0.83$. Assume the SNR of the channel between node 3 and node 5 is measured and normalized to $E_b/E_{03} = 7$, while the SNR of the channel between node 4 and node 5 is measured and normalized to $E_b/E_{04} = 6$. Following the above work flow, we obtain $BER_3 = \frac{1}{2}erfc(\sqrt{7}) = 9.14 \times 10^{-5}$ and $p_{fl3} = 1 - (1 - BER_3)^{1016} = 0.089$. Similarly, $BER_3 = \frac{1}{2}erfc(\sqrt{6}) = 2.66 \times 10^{-4}$ and $p_{fl4} = 1 - (1 - BER_3)^{1016} = 0.237$. With these path parameters, we use the automatic tool to analyze the performance of the two existing paths and the two peer paths. Corresponding cycle probability functions and reachability probabilities can be derived. As a result, $R_{e1} = 99.64\%$, $R_{e2} = 99.92\%$, $R_{e3} = 99.99\%$ and $R_{e4} = 99.82\%$. The conjunction path via node 3 is denoted as path $\alpha$ and the other conjunction path via node 4 is denoted as path $\beta$. According to Theorem 4, we obtain $g_\alpha(x + 1) = g_3(x) * g_1(x)$ and $g_\beta(x + 1) = g_4(x) * g_2(x)$. When reporting interval is $I_s = 4$, the results are calculated and listed in Table 6.4. As can be seen, the reachability probabilities of the path $\alpha$ and path $\beta$ are about the same, i.e.
Under such circumstances, we further compare their delay measures. Since path α consists of 3 hops in total while path β only has 2 hops, this requires one more slot for path α in the communication schedule. As a consequence, $F^\alpha_s = F^\beta_s + 1$, and the expected delay of path α will be longer than that of path β, i.e. $E[\tau]_\alpha = E[\tau]_\beta + 10$ (ms). Hence, to achieve a better performance, path β is preferred to be the first option from the new node to the gateway. A routing preference has been made by performance prediction using the proposed model.
6.7. PERFORMANCE PREDICTION
Chapter 7

Conclusion and Future Work

7.1 Conclusion

Despite the rapid development of wireless technology in consumer and public space applications, the deployment of wireless solutions in industry and process automation is still at the initial phase. In 2007, WirelessHART was approved by the IEC as the first international standard specifically aimed at wireless control for factory automation industry. Based on the IEEE 802.15.4 physical specifications, WirelessHART integrates other technologies such as frequency hopping and Time-division multiple access, so that it exhibits advantages on reliability over ZigBee, which fully complies with the IEEE 802.15.4 standard. Most of the recent research focuses on routing and scheduling algorithm or network simulation, the performance of WirelessHART networks has not received much attention yet. Meanwhile, TDMA enables system modeling by Discrete-Time Markov Chains (DTMC). So we choose DTMCs to model and analyze the WirelessHART system.

A formal description including the connectivity graph and control loops [4] is used in this thesis. The complete WirelessHART network has been modeled step by step. At the first place, a binary symmetric channel is used to describe binary data transmissions. A DTMC link failure model has been proposed to analyze message transmissions within a TDMA slot. Secondly, a hierarchical path model has been proposed, which comprises a path DTMC and several link DTMCs in two tiers. The transition probabilities in the path DTMC depend on the transient state of the link models. Thirdly, a typical network that consists of several paths was developed according to the official guidance. After that, three Quality-of-Service measures have been defined to evaluate the network performance. They can be derived from the DTMC models according to the presented methodology. Moreover, an analyzer with graphical interface has been developed to perform modeling and analysis automatically.

Through evaluation, the influence of link availability and path hop number has been clarified. A scheduling principle was concluded as the guidance to achieve optimal communication schedules. The proposed model is capable of modeling both transient link failures and random link failures. As a result, the WirelessHART network was proven to be robust with transient link failures. Alternative reporting interval to realize a fast control has been
analyzed and compared with the normal case, it can be concluded that users should strike
achieve a good balance between control speed and performance/cost in practice. Lastly, a
theorem about path conjunction was proved and applied in a performance prediction usage
scenario.

Compared to Pesonen’s work in [20], our model generates similar results with the same
parameter setting. This validates the correctness of our model. Moreover, as mentioned
in Chapter 1, we can manage more complex problems with a more general model, so that
we have evaluated WirelessHART performance from different perspectives and discussed
thoroughly according to multiple criteria. To conclude, a good start has been made in this
thesis towards the modeling and evaluation of WirelessHART performance. It is promising
to use the evaluation data in control theories or as a reference in real system deployment.

7.2 Future Work

To continue, there are at least the following future efforts worth investigation.

- Section 1.1 points to some related works on WirelessHART simulation. However, they
do not focus on generating performance measurements. With their experience, simu-
lation on OMNet++ platform with the specific goal of performance evaluation can be
done, in order to further validate the proposed Markov chain model. By comparing the
modeling and simulation results, it is possible to improve the modeling and analysis
methodology.

- Since we follow the formal description of WirelessHART that was presented by the
group from University of Pennsylvania, the evaluation measures such as reachability
and delay can be easily adapted and used to refine their study on control loop stability.
Combined together, unambiguous judgements on whether a WirelessHART system is
stable or not can be made, which is significantly valuable in practice.

- Furthermore, various network routing algorithms and multi-channel scheduling can be
taken into account. It is recommended to include these factors into our path model so
as to evaluate different algorithms proposed in other literatures.
Appendix A

Program Code of the analyzer

```java
public class DTMC {
    protected String name;
    protected state[] states;
    protected ArrayList<transition> transitions = new ArrayList<
        transition>();
    protected Matrix P;  //transition matrix
    protected Matrix P1;

    protected int statesNum = 0;
    //initial distribution
    protected Matrix I;

    public DTMC(){
        states = new state[500];
    }

    public DTMC(int num_states) {
        statesNum = num_states;
        states = new state[statesNum];
    }

    //setter
    public DTMC setName(String aname){
        name = aname;
        return this;
    }

    public boolean considequal(double a, double b){
        //if the difference is smaller than
        if (Math.abs(a-b) < 0.001)
            return true;
        else
            return false;
    }

    //steady-state distribution
    public Matrix steadyState(){
        EigenvalueDecomposition eig = P.transpose().eig();
        Matrix eigenvalue = eig.getD();
        Matrix eigenvector = eig.getV();
        int i;
```
```java
for (i = 0; i < eigenvalue.getRowDimension(); i++) {
    if (considerEqual(eigenvalue.get(i, i), 1))
        break;
}
return ithcollum(eigenvector, i);

// add a new state by age, return new state's name
public String addState(int nlinks, int []aage) {
    states[statesNum] = new state(statesNum, nlinks, aage);
    statesNum++;
    return states[statesNum - 1].getName();
}

// for states without age, return new state's name
public String addState(String aname) {
    states[statesNum] = new state(statesNum, aname);
    statesNum++;
    return states[statesNum - 1].getName();
}

// add a new transition
public transition addTransition(int Fromstate, int Tostate) {
    transition newtran = new transition(Fromstate, Tostate);
    transitions.add(newtran);
    states[Fromstate].NewOutTran(newtran);
    states[Tostate].NewInTran(newtran);
    return newtran;
}

public void addTransition(int FromstateSEQ, int TostateSEQ, double possibility) {
    addTransition(FromstateSEQ, TostateSEQ).setPosi(possibility);
}

// overload
public void addTransition(String FromstateName, String TostateName, double possibility) {
    // converting
    int FromstateSEQ;
    int TostateSEQ;
    try {
        FromstateSEQ = getStatebyName(FromstateName).getSEQ();
        TostateSEQ = getStatebyName(TostateName).getSEQ();
        addTransition(FromstateSEQ, TostateSEQ).setPosi(possibility);
        System.out.println("add_transition_from_"+FromstateName+"_to_"+TostateName+"_with_prb_"+possibility);
    }
    catch (IOException e) {
        e.printStackTrace();
    }
}

// remove a transition
public double removeTransition(int FromSEQ, int ToSEQ) {
    double OrgPro;
    // search
transition deltran = null;
    try {
        deltran = getTransition(FromSEQ, ToSEQ);
    }
    catch (IOException e) {
```
System.out.println(e.getMessage());
e.printStackTrace();

//get original probability
OrgPro = deltran.getPosi();

//remove from 3 lists
transitions.remove(deltran);
states[FromSEQ].OutTrans.remove(deltran);
states[ToSEQ].InTrans.remove(deltran);
return OrgPro;

//generate matrix P from transitions
public void generateP(){
    //verify first
    System.out.println(statesNum);
    P= new Matrix(statesNum, statesNum);
    for (transition tran : transitions){
        P.set(tran.getFromSEQ(), tran.getToSEQ(), tran.getPosi());
    }
    P1 = P.copy();
}

//initial states
public void generateI(){
    I= new Matrix(1, statesNum);
}

public transition getTransition(int FromSEQ, int ToSEQ) throws IOException{
    for (transition curTran : transitions){
        if (curTran.getFromSEQ()==FromSEQ && curTran.getToSEQ()==ToSEQ){
            return curTran;
        }
    }
    throw new IOException("failed to find transition");
}

public state getStatebyName(String name) throws IOException{
    // ATEN!! statesNum i= states.length
    for (int i =0; i<statesNum; i++){  
        if (states[i].getName().equals(name)){
            return states[i]; //return the first one
        }
    }
    //failed to find any state
    throw new IOException("no such state name");
}

//whether specified state already existed
public boolean stateExisted(String name){
    for (int i =0; i<statesNum; i++){  
        if (states[i].getName().equals(name)){
            return true; //found
        }
    }
    return false;
}
```java
public static void main(String[] args) {
    DTMC dtmc = new DTMC();
    dtmc.addState(2, new int[]{0, 0});
    dtmc.addState(2, new int[]{1, 0});
    dtmc.addState(2, new int[]{1, 1});
    dtmc.addTransition(0, 1, 0.7);
    dtmc.generateP();
    dtmc.generateI();
    // dtmc.saveMAT(dtmc.TransientDistribution(10));
    // dtmc.steadyState().print(4, 4);
}
```

Listing A.1: Discrete-Time Markov Chain class

```java
public class SWCN extends DTMC {
    protected int[] links;
    private int[] nodes;
    protected int numOfLinks;
    protected int framesize;
    protected int firstRT;
    private int[] cycles;
    private String whichlinksent;
    private String InitialStateName;
    protected double steadyUP;
    protected int samplerate; //reporting interval

    public SWCN(int numOfLinks, int[] slots, int aframesize, int cycles, int sample) {
        super();
        links = new int[numOfLinks];
        //simplest case, nodes= links, excluding controller.
        nodes = new int[numOfLinks];
        for (int i = 0; i < numOfLinks; i++)
            nodes[i] = slots[i];
        numOfLinks = numOfLinks;
        framesize = aframesize;
        //when the last link sends
        firstRT = nodes[numOfLinks - 1];
        Cycles = cycles;
        samplerate = sample;
        //possible reach cycles+1 states
        //0 discard
        this.addState("Disc");
        addTransition(statesNum - 1, statesNum - 1, 1);
        for (int i = 0; i <= cycles; i++)
            this.addState("R" + (firstRT + framesize * i));
        //self-loop
        addTransition(statesNum - 1, statesNum - 1, 1);
    }

    //construct the starting state
    int[] fage = new int[numOfLinks];
    fage[0] = 1;
}
```

```java
public SWCN(int numOfLinks, int[] slots, int aFramesize, int cycles, int sample, Matrix aI)
    this(numOfLinks, slots, aFramesize, cycles, sample);

//initialize the link one by one
public void initializeLink(double Pud, double Pdu, link initial)
    for(int i = 0; i < links.length; i++)
        if(links[i] == null)
            links[i] = new link(Pud, Pdu, i, initial);

SteadyUP = links[0].steadyUP();

//allocate existed initialized link
public void allocateLink(link aLink)
    for(int i = 0; i < links.length; i++)
        if(links[i] == null)
            links[i] = aLink;

//is this the time to send on this link?
public boolean timeToSend(int age, int link)
    int mod = age%framesize;
    if(mod==0)mod=framesize;
    if(mod == nodes[link])
        return true;
    else
        return false;

public String NextAgeFail(int[] age)
    int n = age.length;
    int curLink = numOfLinks-n;

    int lage = age[0];
    int nextAge;
    //first link, where two cases differ
    if(curLink==0)nextAge = nextmod(lage);
    else {
        //no message wont increase age
        if(lage==0){nextAge=0;}
        //increase with time
        else nextAge = (lage+1)%((nodes[curLink]+Cycles*framesize)+2);
    }
```
if (n==1){
    return nextAge+"];"
} else {
    //recursion
    return nextAge+".0"+NextAgeFail(Arrays.copyOfRange(age, 1, age.length));
}

// next age when succeed to send
public String NextAgeSucceed(int[] age){
    //String newAge="["
    int n = age.length;
    int curLink = numOfLinks-n;
    //get the working digit
    int lage = age[0];
    int nextAge;
    //first link, where two cases differ
    if (curLink==0)nextAge = nextmod(lage);
    else {
        //no message wont increase age
        if (lage==0){nextAge =0;)
        //increase with time
        else nextAge = (lage+1)%((nodes[curLink]+Cycles*framesize+2); // max
    }
    //has message and try to send
    if (lage>=0 && timeToSend(lage, curLink)){
        //first link keeps the sent message
        if (curLink==0){
            //record
            whichlinksent = "0";
            //only one link, directly to the destination
            if (n==1){
                return "R"+lage+"]"+whichlinksent;
            }
        }
        //copy to the next digit
        try{
            return nextAge+".0"+nextAge+".0"+NextAgeSucceed(Arrays.copyOfRange(age, 2, age.length));
        } catch (ArrayIndexOutOfBoundsException e){
            return nextAge+".0"+nextAge+"]"+whichlinksent;
        }
    }
    else {
        whichlinksent= Integer.toString(curLink); //last sent to destination
        if (n==1){
            return "R"+lage+"]"+whichlinksent;
        }
        else {
            try{
                return nextAge+".0"+nextAge+".0"+NextAgeSucceed(Arrays.copyOfRange(age, 2, age.length)); // CHANGE O TO NEXTAGE
            } catch (ArrayIndexOutOfBoundsException e){
        } //other link ALSO!! keep the sent message, including the case of sent to controller
    } //other link ALSO!! keep the sent message, including the case of sent to controller
}
```java
    return nextAge+"","+nextAge+"+whichlinksent;
  }
}

}  //not send
else{
  //the last digit, end of recursion
  if(n==1){
    return nextAge+""+whichlinksent;
  }
  else{
    //recursion
    return nextAge+"","+NextAgeSucceed(Arrays.copyOfRange(age, 1, age.length));
  }
}

//main method in Algorithm 2
public void constructForward(String curSName) throws IOException{
  state curS = getStatebyName(curSName);
  whichlinksent="";
  String NextAgeFail = NextAgeFail(curS.age);
  String NextAgeSucceed = NextAgeSucceed(curS.age);

  //one way, not send point, only one subsequent state
  if(DTMConeway(NextAgeFail, NextAgeSucceed)){
    NextAgeFail = StringUtils.chop(NextAgeFail);
    String FSubName = "","+NextAgeFail+"";
    //System.out.println(FSubName);
    if(ToBeDiscard(curSName, NextAgeFail)){
      System.out.println("discard from "+curSName+" to "+NextAgeFail);
      addTransition(curSName, "Disc", 1);
      //no more recursion
      return;
    }
    //old state
    if(this.stateExisted(FSubName)){
      addTransition(curSName, FSubName, 1);
      return;
    }
    //new state
    else{
      //create new state
      String nextState = this.addState(numOfLinks, StringAgeToArray(NextAgeFail));
      addTransition(curSName, nextState, 1);
      //recursion
      constructForward(nextState);
    }
    }  //f-subsequent and s-subsequent
  else{
    int whlinksent = Integer.parseInt(StringUtils.right(NextAgeSucceed, 1));
  }
```

NextAgeFail = StringUtils.chop(NextAgeFail);
//chop last 2 chars
NextAgeSucceed = StringUtils.substring(NextAgeSucceed, 0, NextAgeSucceed.length() - 2);
String FSubName =ificaciones[""+NextAgeFail+""];
String SSubName = [""+NextAgeSucceed+""];

//reach controller sent by the last link
if (whlinsent==numOfLinks-1){
   //update SSubName to “R7” i.e.
   SSubName = StringUtils.split(NextAgeSucceed, "", "")[numOfLinks -1];
   //System.out.println(SSubName);
}

//f-sub first
if (this.stateExisted(FSubName)){
   addTransition(curSName, FSubName, 30+whlinsent); //30+i for Pf
   return;
}
else{
   String nextState = this.addState(numOfLinks, StringAgeToArray(
      NextAgeFail));
   addTransition(curSName, nextState, 30+whlinsent);
   constructForward(nextState);
}

//whether to discard
//System.out.println(NextAgeSucceed);
if (ToBeDiscard(curSName, NextAgeSucceed)){
   System.out.println("discard_from_"+curSName+"_to_"+
      NextAgeSucceed);
   addTransition(curSName, "Disc", 20+whlinsent);
   //no more recursion
   return;
}
else{
   String nextState = this.addState(numOfLinks, StringAgeToArray(
      NextAgeSucceed));
   addTransition(curSName, SSubName, 20+whlinsent);
   constructForward(nextState);
}

//judge whether the specific message is to be discard by the MAXAGE decrease
public boolean ToBeDiscard(String FromStateN, String ToStateN){
   //to reach states
   if (ToStateN.contains("R")) return false;

   int MaxageF=0;
   try {
      MaxageF = getStatebyName(FromStateN).getMaxAge();
   } catch (IOException e) {
      e.printStackTrace();
   }
   int ageT[] = null;
}}
try {
    ageT = StringAgeToArray(ToStateN);
} catch (IOException e) {
    e.printStackTrace();
}

int MaxageT=0;
for(int i=0; i<ageT.length; i++){
    if(ageT[i]>MaxageT)MaxageT=ageT[i];
}

//MAXAGE decrease means replaced by younger message
return MaxageT < MaxageF;

//wrapped construct function
public void constructAWCN() throws IOException {
    try {
        constructForward(InitialStateName);
    } catch (IOException e) {
        System.out.println(e.getMessage());
    }
    generateP();

    //about I
    if(I==null){
        generateI();
        I.set(0, Cycles+2, 1); /*+1
    }
    //predefined I=T
    else {
        if(I.getColumnDimension() != statesNum){
            throw new IOException("Matrix I's dimension doesn't fit!");
        } else {
            //use I, nothing to do
        }
    }

    public double [][] TransientDistribution(double dtime){
        //time can only be integer
        int time = (int)Math.round(dtime);
        return TransientDistribution(new double [time]);
    }

    //Overload
    public double [][] TransientDistribution(double [] times){
        double [][] arrays = new double [3+Cycles][times.length];
        Matrix T=I;
        //fill array
        for(int i=0; i<times.length; i++){
            times[i]=i;
            if(i>0){
                //update P
                fillP(i);
                //multiplex P matrix
                T=T.times(P);
            }
            for(int j=1;j<Cycles+3;j++){
arrays[j][i] = T.get(0, j-1);
}
arrays[0] = times;
return arrays;

public void fillP(int time){
    for (int i=0; i<P1.getRowDimension(); i++){
        for (int j=0; j<P1.getColumnDimension(); j++){
            int t = (int) P1.get(i, j);
            //pf
            if (t>=30){
                double Ps = links[t].TransientUP(time-1); //TransientUP(time-1)*links[t].getPuu();
                //time-1
                P.set(i, j, 1-Ps); //
            }
            //ps
            else if (t>=20){
                t=t%10;
                double Ps = links[t].TransientUP(time-1); //*links[t].getPuu();
                P.set(i, j, Ps);
            }
        }
    }
}

public void saveMAT(double[][] arrays, String namefix){
    MLD double mlDouble = new MLDDouble( name+"xtime", arrays[0], 1);
    MLD double mlDouble2 = new MLDDouble( name+"discard", arrays[1], 1);
    String filename = "SWCN+namefix+.mat"; //
    //+Math. round(100*Math. random());
    ArrayList<MLArray> list = new ArrayList<MLArray>();
    list.add( mlDouble );
    list.add( mlDouble2 );
    for (int i=2; i< arrays.length; i++){
        list.add( new MLDDouble(name+"R"+(firstRT+framesize*(i-2)), arrays[i], 1));
    }
    try {
        MatFileWriter wt = new MatFileWriter( filename, list );
    } catch (IOException e) {
        e.printStackTrace();
    }

    //how often to send new message?
    public int nextmod(int j){
        int mod = (j+1)%framesize*samplerate;
        if (mod==0)mod=framesize*samplerate;
        return mod;
    }

Listing A.2: WirelessHART Path Model
Bibliography


