

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
MASTER IN EMBEDDED SYSTEMS

MASTER THESIS FINAL REPORT

Analysis of functional redundancy in automotive ethernet networks

Author:
Sai Krishna Kalluri
Student ID: 1035631

Supervisors:
TU/e
dr. D. (Dip) Goswami
NXP
dr.ir. Abhijit Deb
ir. Lulu Chan

Public Version
Friday 17th August, 2018

Abstract

Redundancy is being widely used in many fields over the years and has been known for providing increased reliability and robustness to the corresponding application, system, or network of interest. However, these improvements often come with a certain cost and/or constraints attached to them.

The objective of this project is to quantify and analyze reliability and robustness characteristics of the network upon employing the redundancy mechanisms specified by the standards of the IEEE TSN (Time Sensitive Networking) task group for Automotive Ethernet networks, particularly the redundancy features specified in the IEEE 802.1 CB standard.

To reach this objective metrics related to these characteristics are formulated and analytical models are developed to analyze these characteristics for different types of losses possible in the network (losses due to the failure of links and nodes, and losses due to buffer overflow). The analytical models are implemented in MATLAB/Simulink, and these implementations are used for analyzing reliability and robustness characteristics of the different example network configurations. These analytical models are also validated using appropriate validation mechanisms (except for node losses).

The implementations of these analytical models have exponential worst-case time. This creates a scalability problem. For this reason, two types of speedups are proposed, parallel processing using Graphical Processing Units(GPU's), and utilization of approximation algorithms where computation time is traded off for accuracy. These methodologies are implemented and significant results are achieved in reducing the computation time.

Finally, multi-objective optimization techniques are used to obtain optimal designs with the objectives being maximizing the reliability and robustness of the network and minimizing the cost of the network. In addition to the above, contributions from this thesis also include interactive Graphical User Interface (GUI's), reusable MATLAB live scripts, and automated routines to automate various visualization and analysis operations required to ease the process for new users and developers.

Key words: Time Sensitive Networking (TSN), Automotive Ethernet, IEEE802.1 CB, reliability, robustness, cost, redundancy, analytical models.

Acknowledgements

Not many projects would provide the experience of a lifetime. I am grateful to be blessed with one such experience for my Master thesis project. The lessons learned and the wisdom I gained during this period are truly transformational and I have little doubt that these would positively impact all my future endeavors. All of this wouldn't be possible without the help, encouragement, and guidance of the following people.

First of all, I would like to sincerely thank my mentors at NXP **Dr. ir. Abhijit Deb** and **ir. Lulu Chan** for trusting me with this project and guiding me all throughout to its completion. Your feedback and valuable suggestions during all the progress meetings, feedback with the organization and corrections of my report have helped me a lot. Especially I would like to thank Lulu, for patiently reading my entire report and providing me with useful corrections. It was indeed a pleasure working with both of you.

I would like to thank my Supervisor at TU/e, **Dr. D. (Dip) Goswami** for his guidance and support at crucial moments during the project, and also for his support with various academic-related activities.

I would especially like to thank **Dr. Pieter Cuijpers** for his valuable inputs during the progress meetings. Your critical thinking attitude and constructive criticism constantly pushed me to look deeper into the problem. After a certain point in time, every work I developed started with the following questions in my mind, *'How does Pieter look at this problem? What questions will he ask when I present him this solution?'*. It wasn't a pleasure answering your questions, but it is what helped me see new perspectives and dimensions, and I am grateful for your interest in this project.

I would like to extend my thanks to **Prof. Joe Blitzstein** and Harvard University for their Stat 110 project [39]. Your video lectures and course material provided new insights into probability theory which eventually lead to strong confidence in my results. I would also like to extend my thanks to **Dr. Georgios Exarchakos** for his inputs during my preparation presentation.

Finally, my huge huge thanks to all my closest friends, my parents and my sweet little sister for their love, support and encouragement throughout my Masters...

Sai Krishna Kalluri

Contents

| | |
|--|------------|
| Contents | vii |
| List of Figures | ix |
| List of Tables | xi |
| 1 Introduction | 1 |
| 1.1 Ethernet in automotive domain | 2 |
| 1.1.1 Ethernet AVB | 2 |
| 1.1.2 Time Sensitive Networking | 3 |
| 1.2 IEEE 802.1CB | 5 |
| 1.3 Types of faults under analysis | 7 |
| 1.4 Problem Statement | 9 |
| 1.5 Thesis Structure | 9 |
| 2 Mathematical Modeling | 11 |
| 2.1 In-Vehicle Ethernet Network Topology | 11 |
| 2.2 Link and Node Losses | 12 |
| 2.2.1 Link Losses | 12 |
| 2.2.2 Node Losses | 12 |
| 2.2.3 Metrics | 13 |
| 2.2.4 Net Flows Out of links | 16 |
| 2.3 Buffer Losses | 17 |
| 2.3.1 Metrics | 18 |
| 2.4 Summary | 20 |
| 3 Analytical Modeling | 21 |
| 3.1 Link and Node Losses | 21 |
| 3.1.1 Common Analytical Models used in Reliability Analysis | 21 |
| 3.1.2 Technical Papers related to the analysis of IEEE 802.1CB | 23 |
| 3.2 Buffer Losses | 24 |
| 3.3 Thesis Contributions | 25 |
| 3.4 Background | 26 |
| 3.4.1 Markov Chains | 26 |
| 3.5 Summary | 26 |
| 4 Link and Node Loss Analysis | 27 |
| 4.1 Analysis of a single link network | 27 |
| 4.1.1 Deriving expressions for reliability metrics | 30 |
| 4.1.2 Deriving expressions for robustness metrics | 31 |
| 4.2 Analysis of Multi Link Network | 31 |
| 4.2.1 Deriving expressions for reliability metrics | 35 |
| 4.2.2 Deriving expressions for robustness metrics | 36 |

| | | |
|----------|--|-----------|
| 4.2.3 | Computing Net-flows out of the link | 38 |
| 4.3 | Analysis of Node and Link Losses | 41 |
| 4.3.1 | Metrics | 43 |
| 4.4 | Use Case Analysis Using Symbolic Math | 43 |
| 4.4.1 | Overall Reliability | 44 |
| 4.4.2 | N Link failure reliability contribution | 45 |
| 4.4.3 | Cumulative N link failure reliability contribution | 50 |
| 4.4.4 | NetFlows | 53 |
| 4.4.5 | Individual link robustness | 55 |
| 4.5 | Summary | 62 |
| 5 | Case studies and Validation of Link Losses | 63 |
| 5.1 | Case Studies | 63 |
| 5.1.1 | Series Parallel graphs | 63 |
| 5.1.2 | Ladder Redundancy Networks | 66 |
| 5.1.3 | Cross Link Analysis: | 67 |
| 5.2 | Validation Methodology | 67 |
| 5.2.1 | Configuration and Preparation | 67 |
| 5.2.2 | Simulation Methodology | 70 |
| 5.2.3 | Comparing Analytical and Experimental Results | 72 |
| 5.3 | Validation Results | 74 |
| 5.4 | Summary | 81 |
| 6 | Buffer Loss Analysis | 82 |
| 7 | Computational Speed up | 83 |
| 7.1 | Computation Time complexity of the algorithm | 83 |
| 7.2 | Parallel components and Speed up using GPU's | 83 |
| 7.2.1 | Results | 84 |
| 7.3 | Approximation algorithms | 85 |
| 7.3.1 | Approximate Reliability Calculation | 85 |
| 7.3.2 | Lower and Upper Bounds | 87 |
| 7.3.3 | Results | 87 |
| 7.4 | Summary | 90 |
| 8 | Design Space Exploration | 91 |
| 9 | Conclusions and Future Work | 92 |
| 9.1 | Conclusions | 92 |
| 9.2 | Future Work | 93 |
| 9.2.1 | Improvements of existing work | 93 |
| 9.2.2 | Extensions of the work to other projects/domains | 94 |
| | Bibliography | 94 |
| | Appendices | 98 |
| A | Neural Networks for estimating buffer loss probability | 99 |
| A.1 | Dataset | 99 |
| A.2 | Neural network training | 100 |

List of Figures

| | | |
|------|--|----|
| 1.1 | Evolution of automotive networking technologies (adapted from [5]) | 1 |
| 1.2 | Domain Master gateways (beyond 2020) [27] | 3 |
| 1.3 | Automotive perspective on transition from AVB to TSN [27] | 4 |
| 1.4 | Timeline of TSN task group with key milestones [5] | 4 |
| 1.5 | Network implementing FRER. | 6 |
| 1.6 | A simple network incorporating Frame Replication and Elimination (FRER) . . . | 7 |
| 1.7 | Different types of faults responsible for packet loss in the network | 8 |
| 1.8 | Highlevel overview of the workflow used in this thesis | 10 |
| 2.1 | Undirected graph representation of network shown in Figure 1.5 | 12 |
| 2.2 | Net Flows Example | 17 |
| 2.3 | Modeling of Buffer Overflow | 18 |
| 3.1 | Figure illustrating network with links having dependent and independent failures. . | 23 |
| 3.2 | Figure showing birth death process. | 25 |
| 4.1 | Figure showing the basic work flow for computing metrics | 28 |
| 4.2 | Single Link Network | 28 |
| 4.3 | Markov Chain of a single link network | 28 |
| 4.4 | Figure showing a sample single cross link network along with its pass probabilities | 32 |
| 4.5 | State of the network shown in figure 4.4 when cross link fails | 34 |
| 4.6 | Figure illustrating the conversion of undirected graph to directed graph | 39 |
| 4.7 | Figure showing a sample single cross link network along with its pass probabilities [22] | 44 |
| 5.1 | Figure showing the reliability and robustness plots for series (only) connected con- figurations | 65 |
| 5.2 | Figure showing the reliability and robustness plots for parallel (only) connected configurations | 66 |
| 5.3 | Figure showing the reliability and robustness plots for combined (series and parallel) configurations | 66 |
| 5.4 | A sample directed graph with the directed edge Id's highlighted. | 68 |
| 5.5 | A sample validation use case where all links have a pass probability of 0.9 | 75 |
| 5.6 | Comparison of analytical and experimental net flows for $n = 1000$ and $N_{iter} = 5000$ | 75 |
| 5.7 | Pmf and cdf plots at $n = 1000$ and $N_{iter} = 5000$ | 77 |
| 5.8 | Bootstrap Testing for $n = 1000$, $N_{iter} = 5000$, $k = 10000$ | 77 |
| 5.9 | Pmf and cdf plots at $n = 10$ and $N_{iter} = 5000$ | 78 |
| 5.10 | Bootstrap Testing for $n = 10$, $N_{iter} = 5000$, $k = 10000$ | 78 |
| 5.11 | Pmf and cdf plots at $n = 1$ and $N_{iter} = 5000$ | 79 |
| 5.12 | Bootstrap Testing for $n = 1$, $N_{iter} = 5000$, $k = 10000$ | 79 |
| 7.1 | Configuration of GPU used | 84 |

LIST OF FIGURES

| | | |
|-----|--|-----|
| 7.2 | Cumulative Density Function (cdf) plot of binomial distribution for $N = 10$ | 86 |
| 7.3 | Four Cross Link Network with 16 edges. | 88 |
| 7.4 | Figure showing the plot for computation time for a given level K (computation time in seconds) | 88 |
| A.1 | 3D scatter plots of $E(A)$ vs $E(D)$ vs <i>BufferLossProb</i> for varying buffer sizes . . . | 100 |
| A.2 | Neural Network architecture used for training | 101 |
| A.3 | Error histogram of the training on different data sets (training, validation and test) | 101 |

List of Tables

| | | |
|-----|---|----|
| 3.1 | Table showing comparison of RBD,FT and Markov Chains [1] | 22 |
| 4.1 | Different Algorithms for computing shortest path between nodes in a graph, their descriptions and worst case time complexity (Reference: [38]). | 34 |
| 5.1 | Table showing the summary results from the bootstrap hypothesis for different configurations | 80 |
| 7.1 | Comparison of CPU and GPU time for computing the probabilities of every state in the state space | 85 |
| 7.2 | Table illustrating correlation between cumulative sum of $\binom{16}{K}$ and computation time took by the approximation algorithm | 89 |

Chapter 1

Introduction

With Big Bang came Darwins theory -Survival of the fittest if not vice versa. Since then, in a planet governed by intelligence, humans dominated the rest of the species for survival as machines evolved alongside humans to optimize comprehensive control. In this quest for control, we have produced optimal and efficient systems on one hand while simultaneously breaking constraints posed by technology with the other. Evolution in transportation systems, particularly in the automotive domain has a significant role in this process of acting as a backbone of many major innovations. Advancements in electronics revolutionized the automotive industry resulting in a rapid growth in the number of electronic systems being used since the mid 20th century. Modern automobiles contain as much as 80 Electronic Control Units (ECU) and this count is expected to increase with time [12]. According to [3], the projected cost of automotive electronics cost as a percentage of total car cost worldwide in 2030 reaches 50 percent. This rapid increase in the number of ECU's meant point to point connections for communication between ECU's used in early In-Vehicle-Networks (IVN) is no longer efficient (from both cost and management perspective).

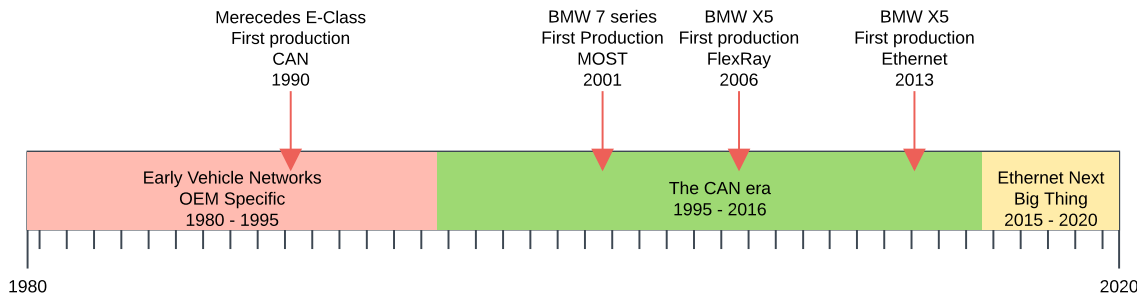


Figure 1.1: Evolution of automotive networking technologies (adapted from [5])

Figure 1.1 shows a brief overview of the evolution of IVN technologies over time. It can be observed from the figure that the future of IVN is moving towards Ethernet. This is primarily due to the limitations in bandwidth capability of CAN, which started becoming a bottleneck with the increasing amounts of data traffic. For example, consider the case of the autonomous driving system. The requirements of this system to work reliably include high data communication between ECU's of various domains along with increased resolution and redundancy of traffic from various sensors. This application generates an enormous load on the network and requires high bandwidth capabilities to guarantee the required Quality of Services (QOS's) for safe operation. Ethernet in this context has been proven to be a technology satisfying the high bandwidth requirements.

The organization of the rest of the chapter is as follows. Section 1.1 provides a brief introduction to automotive ethernet and evolution of Ethernet standards developed for automotive applications. In Section 1.2, the key functions of IEEE 802.1CB is described. This is followed by the problem statement of this thesis in Section 1.4. Section 1.5 provides an overview of the rest of the chapters in this thesis.

1.1 Ethernet in automotive domain

The data traffic both inside application domains and in between application domains keeps on increasing. For example for autonomous driving, the number of sensors as well as their resolution is critical for safe driving. Also, packet loss or high latency in processing these hard real-time tasks can prove fatal. In this context, Ethernet can act as a solution due to its high speed. Moreover, Ethernet is low in cost and a widely proven network technology. The issue with Ethernet however is, it is a best effort network. This means there are no guarantees in the quality of services offered by the network. For example, the latency highly depends on network traffic, and there is no time synchronization between different Ethernet LAN(Local Area Network) nodes. Both of the above features are critical for automotive applications. Many studies are being conducted to explore the capabilities of Ethernet to handle mixed critical traffic(hard/soft real-time and best effort traffic). In 2011 Audio Video Bridging group (AVB) group [4] published a set of technical standards aimed mostly to support the infotainment functions of the car. The success of AVB standards revolutionized the use of Ethernet in the automotive domain. Upon successful results in the infotainment domain, the task group changed its name to Time-Sensitive Networking (TSN) group extending the AVB standards to support the transfer of time-sensitive control data. The evolution of Ethernet standards over time is shown in Figure 1.4. With TSN, Ethernet is expected to act as a backbone to support data transfer between multiple domains. This architecture with Ethernet acting as a backbone network taken from [27] is shown in Figure 1.2. Notice, Ethernet TSN (AVB) is used for communication in the driver assist domain (high bandwidth requirements) and also communication between domains (backbone). Following sections provide a brief overview of standards Ethernet AVB and Ethernet TSN.

1.1.1 Ethernet AVB

Ethernet AVB is a set of standards developed by IEEE 802.1 working group with the main focus of transporting audio/video traffic over IEEE 802 bridged networks. The key standards of Ethernet AVB are as follows:

1. Time Synchronization Specifications- generalized Precision Time Protocol (gPTP) (IEEE 802.1 AS)[20]
This standard provides mechanisms for all the clocks in the network remain synchronized with respect to a common grandmaster clock.
2. Stream Reservation Protocol (SRP) (IEEE 802.1Qat) [23] This standard provides mechanisms for end-to-end management of the stream's resources in order to guarantee the quality of services.
3. Forwarding and Queuing for time-sensitive streaming applications (IEEE 802.1Qav) [21]
This standard provides mechanisms that allow bridges to provide guarantees for time-sensitive real-time audio video (AV) data transmission. Mainly, it provides mechanisms for per priority ingress metering, priority regeneration, and timing-aware queue draining algorithms.

While the above standards are sufficient for transferring audio-video data, they are still inadequate when it comes to satisfying strict QoS requirements needed by safety critical data. TSN aims to

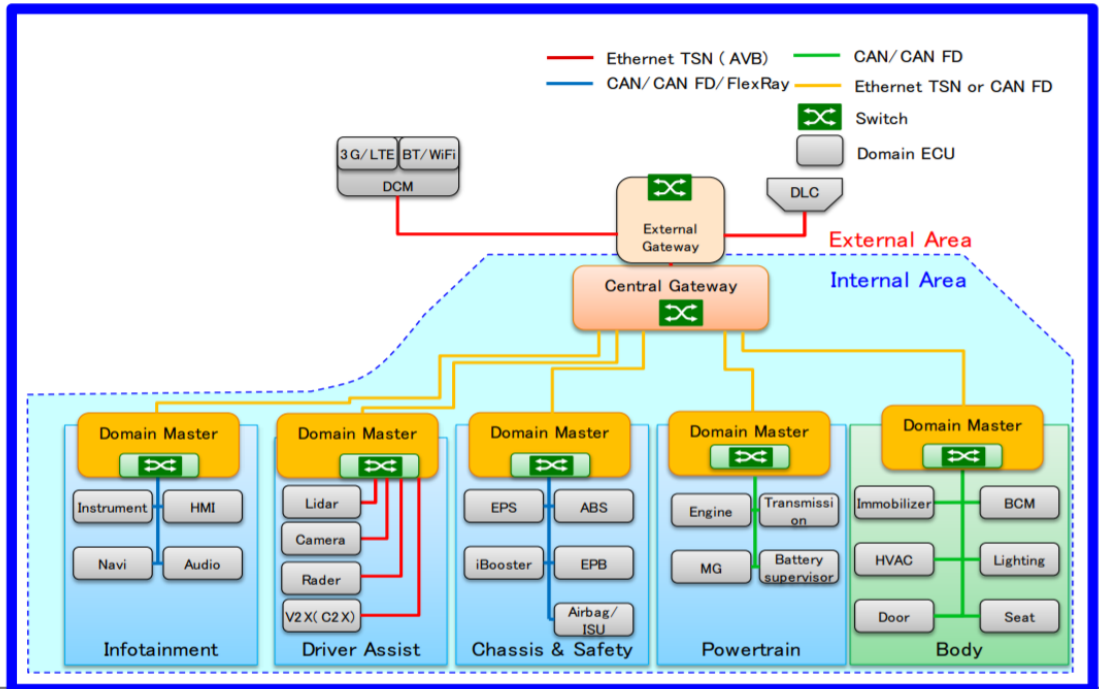


Figure 1.2: Domain Master gateways (beyond 2020) [27]

solve this problem by extending the AVB standards. In the next subsection, a brief overview of TSN standards is discussed.

1.1.2 Time Sensitive Networking

Time Sensitive Networking (TSN) are a set of standards that are under development by the Time Sensitive Networking task group. TSN group formed in 2012 after renaming its name from Audio Video Bridging (AVB) group. Figure 1.3 shows the automotive perspective on the transition from AVB to TSN (Currently, not all specifications are in draft. Some of them are standardized). We can observe the focus has been moved from infotainment to control applications. These extra requirements resulted in enhancements in existing AVB standards as well as adding new standards. For example from the Figure 1.3 we can observe additional measures are taken to control latency of frames and provide reliability to the frames.

The key objectives of TSN as presented in [19] are as follows:

1. Extending the use cases from audio/video applications to control systems.
2. Reducing worst-case delays
3. Improving the robustness and reliability of the time and safety-critical traffic.
4. Scaling to larger networks.

| AVB (Automotive profile) | TSN |
|--|---|
| <ul style="list-style-type: none"> <u>Intention</u> <ul style="list-style-type: none"> Audio video synchronisation <u>Attractive Feature</u> <ul style="list-style-type: none"> Clock synchronisation Static Stream reservation Traffic shaping <u>Automotive area</u> <ul style="list-style-type: none"> Infotainment <u>Specification Stability</u> <ul style="list-style-type: none"> Yes, automotive profile | <ul style="list-style-type: none"> Control application Clock synchronisation Stream reservation Traffic shaping Ingress policing Low latency for control frames Reliability ADAS Control application All Specifications in Draft No automotive profile |

Figure 1.3: Automotive perspective on transition from AVB to TSN [27]

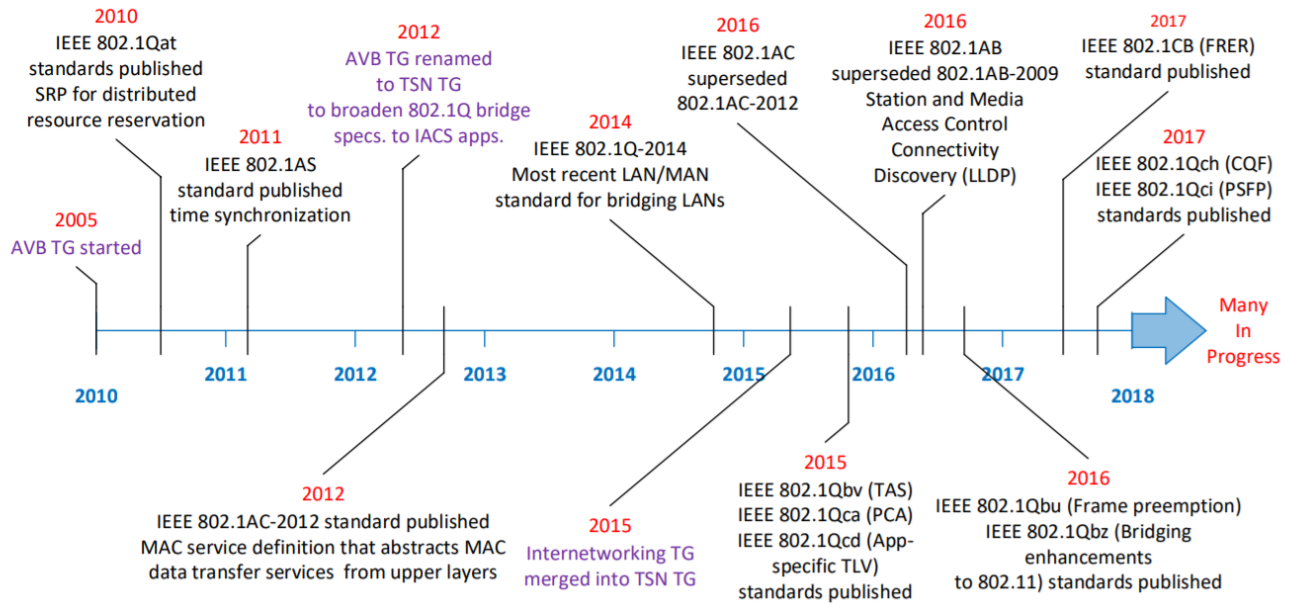


Figure 1.4: Timeline of TSN task group with key milestones [5]

Various TSN standards related to current automotive landscape and their functions are as follows:

1. **IEEE 802.1Qbv:** Time-aware shaping

This standard provides mechanisms for blocking all ports except one based on a time schedule in order to prevent delays during a scheduled transmission. This ensures that the data is not delayed for time-sensitive data.

2. **IEEE 802.1As-Rev:** Timing and Synchronisation

This standard ensures that all the clocks in the network remain synchronized with respect to a grandmaster clock. It also supports the use of multiple grandmasters and multiple

connections (redundancy), to ensure that the clocks in the network stay synchronized by taking time from the redundant grandmasters.

3. **IEEE 802.1Qbu**: Frame Preemption

This standard provides mechanisms for preempting long frames (still transmitting even after the slot time expires) in order to facilitate the transmission of higher priority frames.

4. **IEEE 802.1CB**: Frame Replication and Elimination for Reliability (FRER)

Messages are copied and sent over parallel disjoint paths. The duplicate packets are eliminated at the merge points. Improves the probability of packet getting delivered to the destination.

5. **IEEE 802.1Qcc**: Enhancements and improvements for stream reservation

This standard is an enhancement over existing standard Stream Reservation Protocol (SRP) of AVB in order to meet the requirements of professional, industrial and automotive markets.

6. **IEEE 802.1Qca**: Path Control and reservation

This standard specifies mechanisms for discovering the redundant paths in the network by collecting topology information from the nodes. Useful particularly when 802.1 CB standard is employed in the network.

7. **IEEE 802.1Qch**: Cycling Queuing and forwarding

This standard provides mechanisms for collecting packets according to their traffic class and forwarding them in one cycle. This standard provides an upper boundary for latency. Useful in applications where controlled timing is important but low latency is not that important.

8. **IEEE 802.1Qci**: Per-Stream Filtering and Policing

This standard provides mechanisms for filtering frames at the ingress ports depending on the factors like arrival times, rates and bandwidth. Protects against excess usage of bandwidth by nodes and also against faulty endpoints.

We can observe from the above descriptions, there are redundancy features implemented in the IEEE 802.1 CB (Frame Replication and Elimination for Reliability) and 802.1 AS-Rev (Enhanced generalized Precision Time Protocol). These standards can be implemented to increase the reliability and robustness of the network. The scope of this project, in particular, is to analyze the redundancy features presented in IEEE 802.1 CB, understand the pros and cons of implementing these standards and finally provide useful recommendations for optimal network designs. In the subsequent section, the description of the functionality of redundancy features specified in IEEE 802.1 CB is described followed by the problem statement of this thesis in section 1.4.

1.2 IEEE 802.1CB

IEEE 802.1 CB standard deals with offering seamless redundancy to safety critical packets. The key objective is to improve the probability of the packet not getting lost during transmission. Fundamentally, the standard proposes to achieve this by replicating the packets, transmitting them across separate paths and eliminating duplicate packets at various merge points. The elimination is important to prevent flooding of data. This specific feature of replication and elimination of frames is called Frame Replication and Elimination for Reliability (FRER). Incorporating this feature is expected to increase the reliability and robustness of the network, which we will quantify as part of this project.

Before diving into the functions of FRER, it is important to understand some preliminary concepts of the standard. Data transfer in this standard is represented as streams. The stream is also the entity for which the Quality of Services are offered (QOS's). A stream is made up of two key characteristics, the maximum packet size and the number of packets transmitted per time interval. These characteristics help determine the streams maximum throughput, which can be used to determine the resources needed such as link bandwidth and buffer space needed at every hop to guarantee finite latency and zero congestion loss. This standard, however, doesn't deal with the methods needed to achieve zero congestion loss but recommends that the standards that guarantee zero congestion loss to be employed for better performance. Also, the path the packets take is also not determined by this standard. A stream is identified by following sub-parameters specific to the IEEE 802.1 CB standard.

1. **Stream Handle:** This is an integer for identifying the stream to which the packet belongs.
2. **Sequence Number:** This is an unsigned integer used to represent the order in which the packets were transmitted relative to other packets in the same compound stream.

Figure 1.5 shows the key functional components of FRER. The functionality of each function is as follows.

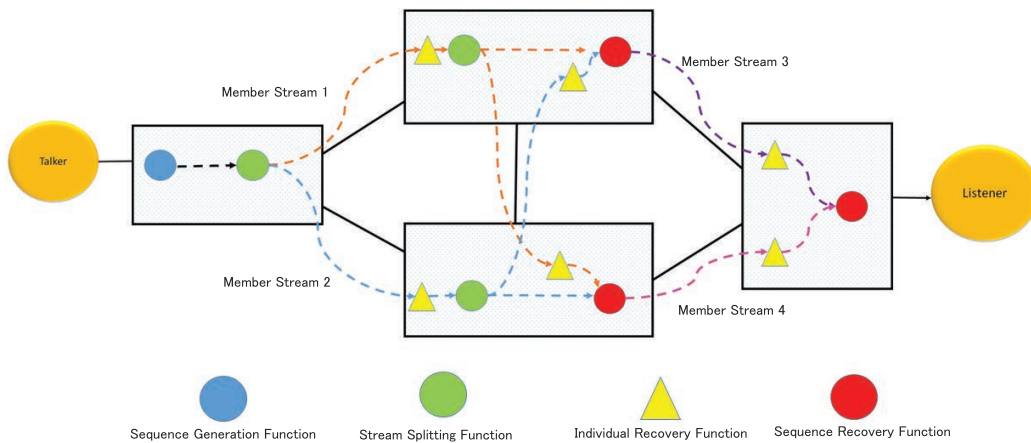


Figure 1.5: Network implementing FRER.

1. **Sequence Generation Function:** The *Sequence Generation Function* supplies a sequential value to the sequence number sub-parameter for packets passed down from the upper layers for transmission along the network. The application can be seen in the talker end system shown in Figure 1.5.
2. **Stream Splitting function:** This function replicates each packet passed down to it and assigns them a different stream handle. In other words, this is the function that is responsible for replication of the packets. When packets are passed up to it, they are passed unchanged.
3. **Sequence Recovery Function:** The *Sequence Recovery Function* examines the sequence numbers of the packets received from multiple streams and discards the packets that are duplicate (already passed). There are two types of methods described in the standard for performing this recovery operation, namely vector recovery and match recovery. At a high level, these recovery algorithms require memory of the switch to store the sequence number of the packets already passed. The choice of recovery algorithm and size of the memory is crucial for the correct operation of FRER. Throughout this project, the effects of these design choices are ignored and it is assumed that the recovery function

performs correctly.

4. **Individual Recovery function:** The function of *Individual Recovery Function* is the same as the *Stream Recovery Function*, except the latter operates on multiple streams, while *Individual Recovery Function* as the name suggests operates on individual member streams. The idea is to make the network robust to the cases where the transmitter is stuck and keeps on transmitting the same sequence number again and again. The difference between the individual and sequence recovery can also be observed in the Figure 1.5.

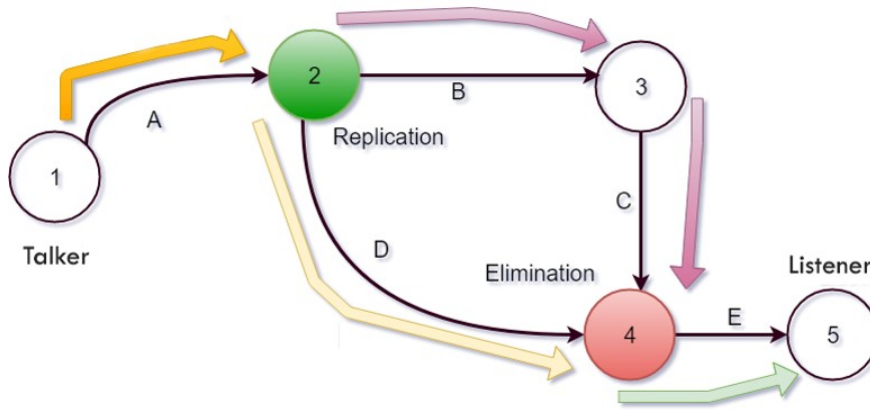


Figure 1.6: A simple network incorporating Frame Replication and Elimination (FRER)

From the above functionalities, the key functions of interest for analyzing the impact they have on reliability and robustness characteristics of the network are the *Stream Splitting (a.k.a replication)* and *Sequence Recovery (a.k.a elimination)* functions. Figure 1.6 shows a simplified network incorporating only the above functions. The frames are replicated at node 2 and eliminated at node 4. Throughout this project the terms frames and packets are used synonymously. In the next section we describe the types of faults under analysis in this thesis.

1.3 Types of faults under analysis

As described in the section 1.2, the objective of FRER is to improve the reliability and robustness of the network against failures. Before quantifying and analyzing these characteristics, it is important to study different kinds of faults possible in the network. Classifying the faults into different types helps in understanding and differentiating the impact of implementing FRER on the network under different conditions, and thereby aid in reliable, robust and cost-effective designs.

The frame loss in the network is classified into three categories based on its location and nature. Figure 1.7 shows the impact of each type of losses on the data flow for the network described in figure 1.5. The description of the losses is as follows.

1. **Link Losses:** This type of losses are due to loss of frames due to faults on the links. Mostly, the link losses are due to electromagnetic interference from external radiation. Link losses impact the member streams of the network. Observe in figure 1.7 that this type of fault impacts the member stream. However, the packet reaches the destination due to redundancy.
2. **Node Losses:** FRER is primarily designed to protect the network against link losses by providing redundant paths. However, it is interesting to study the impact on the network

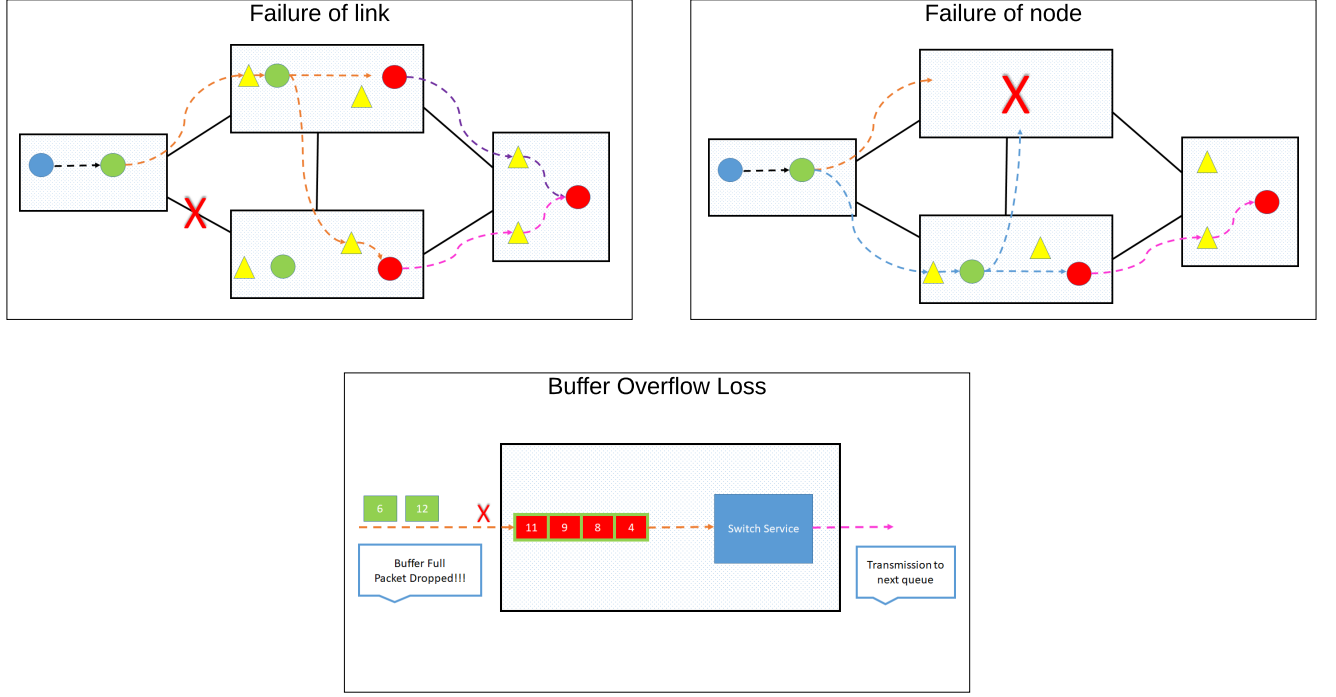


Figure 1.7: Different types of faults responsible for packet loss in the network

due to losses on the nodes. Node losses compared to link losses are longer in duration. For this reason, failure of node unlike link loss results in network disconnecting from all the streams connected to it. Notice from figure 1.7 the number of streams reaching the talker reduced compared to the failure of link loss. Node losses can be due to reasons like power failure, faulty hardware/software, e.t.c.

3. **Buffer Losses:** Even though buffer losses happen inside a node, these losses are analyzed separately in this thesis. The difference between buffer and node losses is that in node losses, the node completely fails and all the streams connected to it are isolated, whereas, in case of buffer loss, the node doesn't accept the frame from an individual stream due to the overflow of buffer connected to the stream. Buffer losses are a result of congestion in the network and depend on the inter-arrival distribution of frames corresponding to the stream and the service distribution characteristics offered by the node.

Out of these losses, link losses are studied in detail since its the focus of FRER (Refer 1.1.2). Buffer losses are also studied in detail since the replication operation causes more flow into the nodes. Section 1.5 provides information on the chapters containing analyses of these faults.

1.4 Problem Statement

The objective of this thesis is to analyze the effects of functional redundancy in automotive Ethernet networks. Particularly the effects on network characteristics upon employing redundancy features specified in the standards IEEE 802.1 CB are considered. Following research questions will be studied as part of the analysis:

Question 1: *How does employing Frame Replication and Elimination for Reliability (FRER) features affect reliability characteristics of the network?*

- What are the various metrics that quantify the reliability of a network?
- How do these metrics relate to the network configuration parameters (component loss probabilities, Buffer size, switch operating frequency, Inter-arrival distribution of frames)?
- What is the effect on these metrics upon employing FRER?

Question 2: *How does employing Frame Replication and Elimination for Reliability (FRER) feature affect robustness characteristics of the network?*

- What are the various metrics that quantify the robustness of a network?
- How do these metrics relate to the network configuration parameters (component loss probabilities, Buffer size, switch operating frequency, Inter-arrival distribution of frames)?
- What is the effect on these metrics upon employing FRER?

Question 3: *How does employing Frame Replication and Elimination for Reliability (FRER) affect cost metrics of the network?*

- What are the various metrics that quantify cost properties of a network?
- What is the effect of employing Frame Replication and Elimination (FRER) mechanisms on these cost metrics?

1.5 Thesis Structure

As discussed in the section 1.2 and section 1.4, the key focus of this thesis is to quantify and analyze the reliability, robustness, and cost of the network. Figure 1.8 shows an overview of the various steps needed to achieve this goal. The organization of the rest of the chapters follows the steps described in Figure 1.8.

Chapter 2: The first step in performing the quantitative analysis is to transform the textual descriptions into mathematical representations. In this chapter, abstraction and mathematical modeling of FRER is developed. This involves transforming the Ethernet network into an undirected graph, modeling the different types of losses as discussed in section 1.3 (both to be used as network configuration parameters as shown in Figure 1.8) and representing reliability, robustness, and cost of the network into quantifiable metrics.

Chapter 3: In this chapter, the focus is on choosing the suitable analytical model that maps the network configuration to metrics. For this purpose, different analytical models commonly used for quantitative analysis are studied and compared. This chapter also provides users with the necessary background that will prove useful in the subsequent chapters.

Chapter 4: This chapter contains the analysis related to link and node losses on the network. This chapter also contains a quantitative study of an example network analyzed using Symbolic Math [41].

Chapter 5: This chapter focuses on detailed comparison and analysis of different case studies

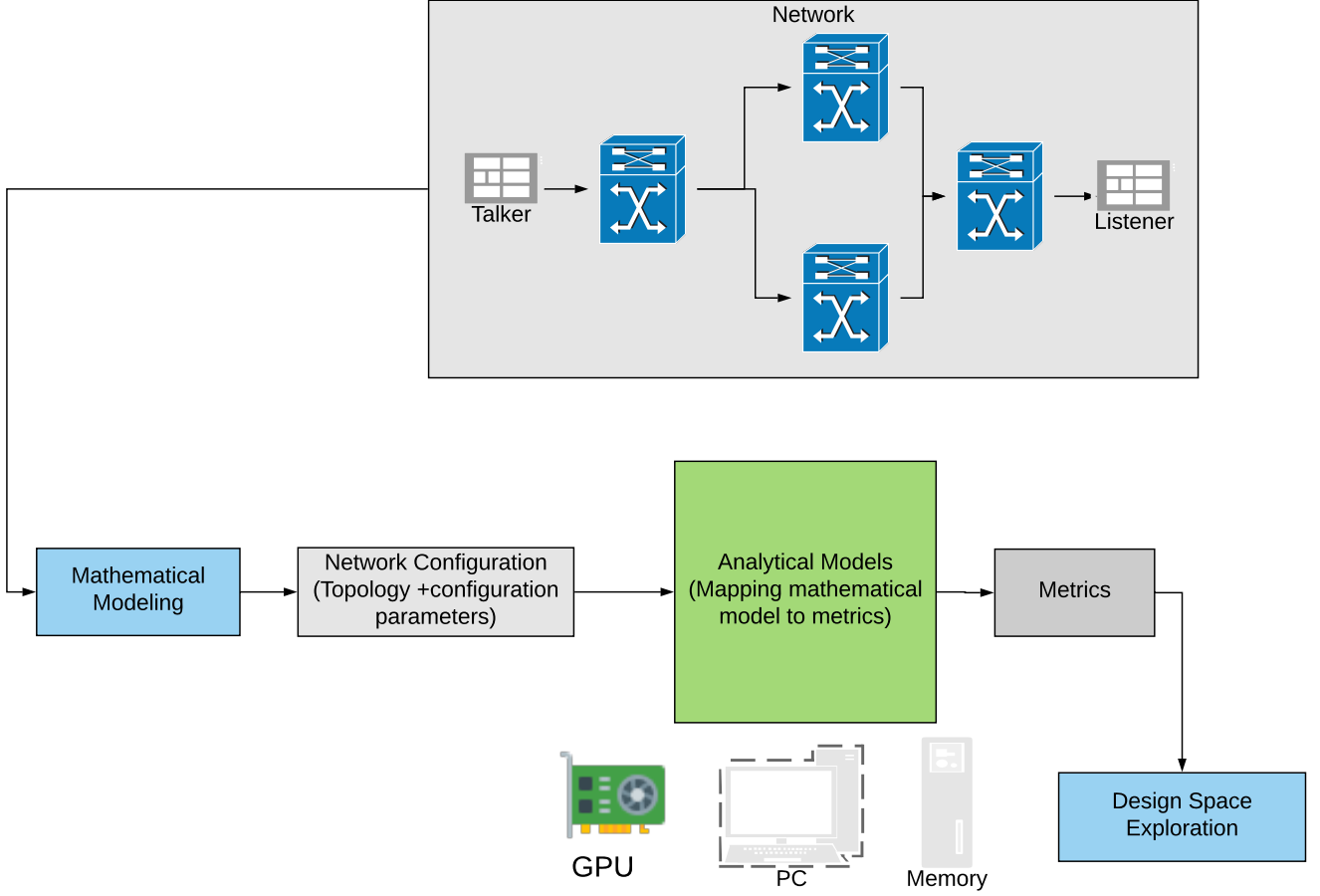


Figure 1.8: Highlevel overview of the workflow used in this thesis

related to link losses based on the analysis described in chapter 4. This chapter also contains a validation methodology for validating the analysis for link losses.

Chapter 6: In this chapter, analysis related to buffer loss probability, particularly its dependency on the buffer size, variations in the inter-arrival time between packets and in the service time offered to them are studied.

Chapter 7: This chapter focuses on improving the computational speed of algorithms described in the previous chapters. This methodology is useful for analyzing large networks.

Chapter 8: This chapter focuses on techniques to design the networks utilizing the metrics developed in the previous chapters. The aim is to optimize the designs based on the objectives of the network (reliability, robustness, and cost) along with satisfying the constraints (Cost properties of the network, if present) imposed. In particular, the analysis related to the position of cross-links and finding optimal designs using multi-objective optimization are discussed.

Chapter 9: This chapter provides a brief summary and conclusion of the work done during this thesis. This chapter also provides a list of future works that can be developed/improved on top of this thesis.

Chapter 2

Mathematical Modeling

In Section 1.3, various possible failures were discussed. In this chapter, we start our quantitative analysis of the network reliability and robustness by describing the mathematical modeling of the network and its components. In Section 2.1, we model the In-Vehicle Ethernet network topology. In Section 2.2, we model the failures of links and nodes (switches/End Systems), and quantify metrics related to reliability, robustness, and cost of the network under these failures. In Section 2.3, mathematical modeling necessary for analyzing buffer losses is discussed. This constitutes modeling the dependency of buffer losses on inter-arrival time between packets and service time offered to them, quantifying metrics in terms of reliability, robustness, and cost of the network associated with these losses.

2.1 In-Vehicle Ethernet Network Topology

The In-Vehicle Ethernet network topology can be modeled as an undirected graph with switches as vertices and edges as the links connecting them. The reason for representing the network as an undirected graph is mainly because ethernet links are full duplex. In other words, these links allow the bidirectional transfer of data. However, the flow of data between two nodes is directional and acyclic. The weights of the edges represent the pass probabilities of packets passing through the links. Therefore a network can be represented as follows. $G = (V, E)$ where G is the graph of underlying network topology. V represents the set of vertices (Switches/EndSystems) and E represent the set of links connecting the vertices.

The number of links (N) in the network from the graph description is equal to the number of edges ($N = |E|$) and number of nodes is represented using M ($M = |V|$). For example, consider the network represented by the undirected graph shown in the Figure 2.1. For this graph

$$G = (V, E)$$

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(1, 2), (2, 3), (2, 4), (3, 4), (3, 5), (4, 5), (5, 6)\}$$

$$N = |E| = 7$$

$$M = |V| = 6$$

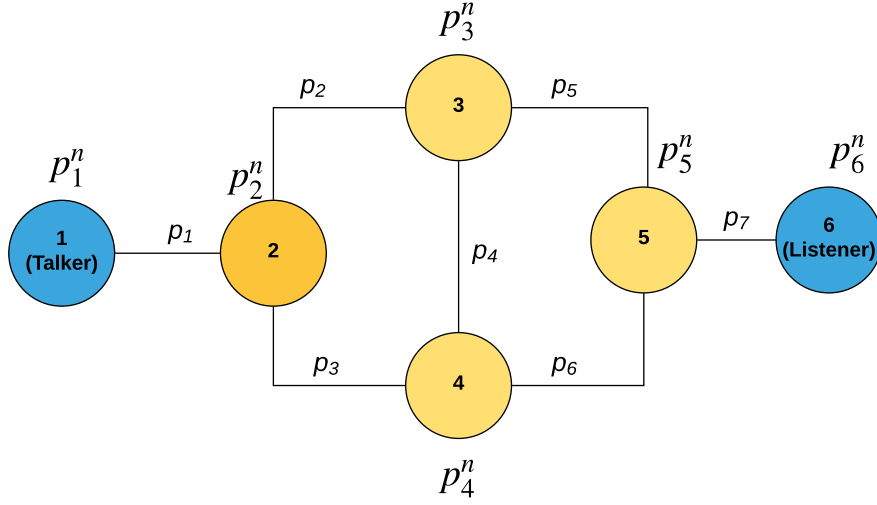


Figure 2.1: Undirected graph representation of network shown in Figure 1.5

2.2 Link and Node Losses

In Section 1.3, we classified the network failures into three categories. In this Section, we mainly focus on modeling the link and node failures. Figure 2.1 shows the network model incorporating the link and node failures in accordance with the notations defined below.

2.2.1 Link Losses

Previously in Section 1.3, we determined link losses are caused mainly due to electromagnetic interferences. The amount of losses depend on the environment the link is in. Thus, link losses can be modeled as a probability distribution over all the environments. The loss probability of each environment can be measured experimentally by transmitting a large number of packets and computing the ratio between the number of packets successfully delivered to the number of packets transmitted. Since our objective is to quantify reliability and robustness characteristics, for simplistic reasons, we model the loss in a link using a single parameter called pass probability. Pass probability of a link is defined as the probability with which a link allows packets to pass through it successfully. We denote p_i as the pass probability of link i in the network.

2.2.2 Node Losses

Losses in nodes are mainly due to power failures. For this reason, the losses in the nodes are longer in duration compared to the link losses. According to analysis in [25], the MTTF (Mean Time To Failure) and MTTR (Mean Time to Repair) are in hours. Since we are interested in steady-state analysis, we can directly model the pass probability of node i as p_i^n , where p_i^n denotes the pass probability of node i . The pass probability can also be represented as the availability of the node, where availability can be computed from MTTF and MTTR information as

$$p_i^n = 1 - \frac{MTTR}{MTTF + MTTR}$$

2.2.3 Metrics

Metrics help in quantifying the characteristics of interest of the network. In our case, these are reliability, robustness and cost of the network. In this Section a few metrics that are of relevance are described. However, in practice, the relevance varies on application nature and the reader/responsible engineer is encouraged to create his/her own metrics depending on the need.

Lets start with modeling the non determinism in successful delivery of the packet to the listener. Denote by X an indicator random variable, indicating whether a transmitted packet has successfully reached its intended destination or not. In other words,

$$X = \begin{cases} 0, & \text{if packet is lost} \\ 1, & \text{if packet is successfully delivered} \end{cases}$$

Note, the variable X is random, since we do not know a random packet transmitted will be delivered or not. The probability with which a packet gets successfully delivered depends on the network configuration (Network Topology, Link Losses, Node Losses) modeled in the previous sections.

The metrics are classified into the following types

Reliability Metrics

Reliability is defined as the ability of the network to successfully deliver the transmitted packets. Experimentally it can be calculated as follows.

$$Reliability = \frac{N_{received}}{N_{sent}} \quad (2.1)$$

where N_{sent} and $N_{received}$ correspond to the number of packets transmitted by the talker and number of packets successfully received successfully. With this context the metrics for computing the reliability of the network for any given configuration are provided below. Metric 2.2 directly corresponds to the definition above while metric 2.3 and metric 2.5 represent the reliability of the network under special conditions.

Note:

1. The notation $Pr(e)$ denotes the probability of event e happening.
2. $Pr(e_1, e_2)$ denotes the probability of events e_1 and e_2 jointly happening.
3. $\frac{\partial F}{\partial x}$ denotes the partial derivative of function F with respect to variable x .
4. All the metrics described below are functions of pass probabilities of all the components (nodes and links).

1. **Packet Pass Probability:** This metric indicates the probability with which a packet sent by talker successfully reaches its end listener without getting lost in the way.

Mathematical Representation:

$$Pr(X = 1) \quad (2.2)$$

2. **Reliability contribution by (n links and m nodes) failure cases:** This metric quantifies the contribution made to the reliability of the network by n links and m nodes failure cases.

Mathematical Representation:

$$Pr(X = 1, failures^{link} = n, failures^{node} = m) \quad (2.3)$$

where $failures^{link}$ and $failures^{node}$ indicate the number of link and node failures respectively. This metric is related to overall reliability of the network as follows:

$$Pr(X = 1) = \sum_{i=0}^N \sum_{j=0}^M Pr(X = 1, failures^{link} = i, failures^{node} = j) \quad (2.4)$$

This metric is an improvement over the one used for illustration in IEEE 802.1CB[22] standard. In Section 7.1 of the standard it is illustrated how the network under analysis (Figure 7.1 in [22]) is protected against all the 7 possible one link failures and against 16 of 21 possible 2 link failures. In other words this corresponds to the metric $Pr(X = 1 | failures^{link} = k)$. The draw back of this metric is that it solely depends on the topology and doesn't consider the pass probabilities of the components. For this reason the above metric (metric 2.3) is studied in this thesis. The advantage is that the number of successful cases can still be extracted by summing all the coefficients in the resultant expression provided by this metric. An example of this is shown in symbolic math analysis section 4.4.

3. **Reliability contribution by (n or more links and m or more nodes) failure cases:** The metric helps in quantifying the contributions made to the reliability of the network by the failure cases corresponding to failures of n or more links and m or more nodes.

Mathematical Representation:

$$Pr(X = 1, failures^{link} \geq n, failures^{node} \geq m) \quad (2.5)$$

This metric is related to overall reliability of the network as follows:

$$Pr(X = 1) = Pr(X = 1, failures^{link} \geq 0, failures^{node} \geq 0) \quad (2.6)$$

This metric is a cumulative version of metric 2.5 and provides insights into amount of redundancy in the network. For example in networks with no redundancy support, this metric results in 0 for $n >= 1$.

The classification of metrics 2.3 and 2.3 into reliability of robustness is debatable. However, since they directly contribute to the reliability of the network, they are classified as reliability metrics. For better readability from now on we denote reliability of the network as R , defined by the equation 2.7. In other words, we use R as a substitute for $Pr(X = 1)$ which is a function of all the component pass probabilities. We will use this notation for quantifying robustness metrics.

$$Reliability = R(p_1, p_2, \dots, p_N, p_1^n, p_2^n, \dots, p_M^n) = Pr(X = 1) \quad (2.7)$$

Robustness Metrics

Robustness according to wikipedia is the property of being strong and healthy in constitution. In our case, this can be transposed into the ability of the network to deliver packets successfully despite perturbations in the pass probability of the components. In another words it is the change in reliability of the network despite these perturbations. With this idea, the metrics for robustness are defined as follows.

1. **Robustness of the network with respect to perturbations in pass probability of a single Link/Node** This metric quantifies the change in reliability of the network with respect to change in the pass probability of the link/node. This metric gives an idea of how sensitive the network is to variations in the pass probability of $link_i/node_i$. The lower this value the robust (better) is the network with respect to perturbations in the pass probability of this link/node.

Mathematical Representation:

$$\frac{\partial R}{\partial p_i} \text{ (link)} \quad \text{or} \quad \frac{\partial R}{\partial p_i^n} \text{ (node)} \quad (2.8)$$

2. **Mean Robustness w.r.t variations of all the links/nodes (computed individually):** This metric quantifies the average robustness of the network to the changes in pass probability of the links/nodes.

Mathematical Representation:

$$\frac{\sum_{i=1}^N \frac{\partial R}{\partial p_i}}{N} \text{ (link)} \quad \text{or} \quad \frac{\sum_{i=1}^M \frac{\partial R}{\partial p_i^n}}{M} \text{ (node)} \quad (2.9)$$

3. **Collective Robustness:** Robustness of the network when all the links/nodes pass probabilities change by a factor of Δ . Signifies the shift in environment which affects all the links/nodes equally.

Mathematical Representation:

$$\frac{1}{N} \cdot \frac{\Delta R}{\Delta p} = \frac{R(p_1, p_2, \dots, p_N, p_1^n, p_2^n, \dots, p_M^n) - R(p_1 - \Delta, p_2 - \Delta, \dots, p_N - \Delta, p_1^n, p_2^n, \dots, p_M^n)}{\Delta \cdot N} \text{ (links)}$$

(or)

$$\frac{1}{M} \cdot \frac{\Delta R}{\Delta p} = \frac{R(p_1, p_2, \dots, p_N, p_1^n, p_2^n, \dots, p_M^n) - R(p_1, p_2, \dots, p_N, p_1^n, p_2^n - \Delta, \dots, p_M^n - \Delta)}{\Delta \cdot M} \text{ (nodes)} \quad (2.10)$$

The relation between metrics 2.9 and 2.10 is that metric 2.9 corresponds to individual component probability variations while the latter correspond to all the components varying simultaneously. Interested readers can also form intermediate metrics where probability change happens in k components simultaneously.

Cost metrics

Cost can be measured in multiple ways. Broadly, the cost of a network is the sum of the cost of the individual components it contains, and the cost of an individual component is a function of elements that make the component. For example a link with more shielding is better protected against external interference. Generalizing the statement to a component, cost of a can be computed as

$$cost_component = f_l(p_{component})$$

where f_l is some arbitrary function of $p_{component}$ (pass probability of the component), and the cost of the network can be computed as

Mathematical Representation:

$$cost_{network} = \sum_{i=1}^k cost_component_i \quad (2.11)$$

For simplistic reasons, we assume the cost due to variations in the design are negligible and cost of the network is computed as the number of links and switches that make the network. The cost metric plays a role in the design choice of the networks. More on designing the network is presented in the chapter 8.

2.2.4 Net Flows Out of links

In Section 2.2.3, we formulated metrics for reliability of the network, which represents the probability a packet successfully reaches its listener. When transposed to a stream of data flowing through the network, reliability of the network can be thought as the net amount of flow coming out of the link connected to the listener after all the losses are considered. Computing the net flows helps in visualizing the journey of streams through the network and losses associated with various locations. Throughout this thesis, the net flow out of link i is represented as γ_i . An example network consisting of 5 nodes is shown in figure 2.2. For simplicity node losses are assumed to be 0 in this case. The top part of the figure shows the network model (topology + configuration parameters), while the bottom part shows the net flows coming out of every link. Convince yourself that the reliability of the network is equal to γ_5 .

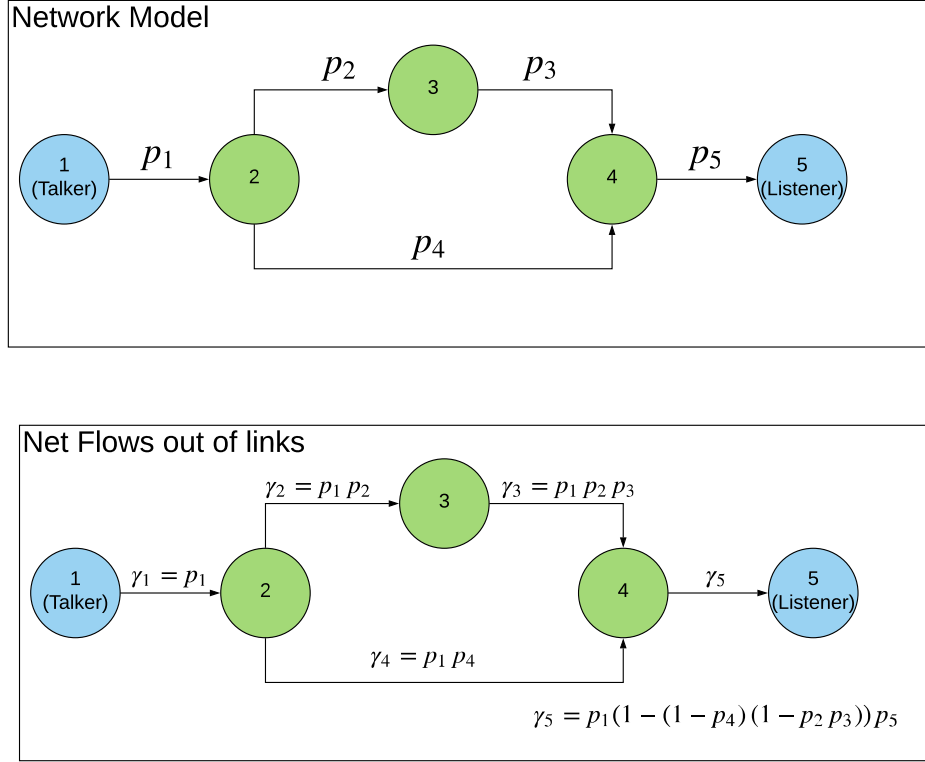


Figure 2.2: Net Flows Example

2.3 Buffer Losses

In this section mathematical modeling necessary for analyzing buffer losses is described. As described in 1.3, overflow losses in buffer happen due to insufficient buffer space for the incoming packet. Buffer overflow loss can be computed using the following parameters as shown in figure 2.3.

1. Inter-arrival time distribution (γ)

The flow into the network is not periodic. Under bursty traffic the inter arrival time between successive packets reduces. This non determinism in packet arrivals can be modeled using this probability distribution. The representation γ is intentionally used for this (same as net flows out of link shown in Section 2.2.4), since the input into the buffer is the net-flow out of a link. There is a subtle difference though. The net flow signifies the number of arrivals distribution, while inter-arrival distribution signifies the non determinism in the time between successive packets. For all practical purposes these distributions can be computed from one another and using same notations helps in understanding the concepts better (for example refer the integration of buffer loss into link losses described in Section 2.3.4)

2. Service time distribution (μ)

Packets can have unequal sizes. This causes variations in the service time of the switch in forwarding these packets. Variations in service time can also arise due to design differences in software and hardware components of the switch.

3. Buffer Size (B)

This parameter represents the size of the buffer used for storing the incoming packets. An incoming packet is lost when the buffer is full.

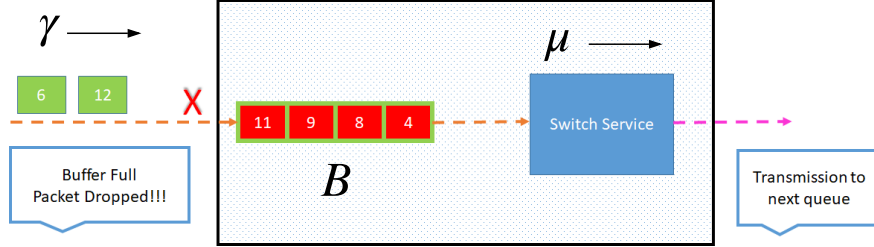


Figure 2.3: Modeling of Buffer Overflow

Buffer loss probability is the probability that an arriving packet gets lost. It happens when a new packet arrives and buffer is full. Denote B_{Loss} as the probability that an incoming packet is lost and f be a function that computes buffer loss from the parameters γ , μ and B . Therefore,

$$B_{Loss} = f(\gamma, B, \mu) \quad (2.12)$$

The analysis for deriving the expressions for buffer loss probability from the above parameters is studied in chapter 6.

2.3.1 Metrics

Reliability Metric

Reliability in this context is the ability of the buffer to store the incoming packets without getting overflowed. Thus, reliability can be quantified using the equation 2.12 as

Mathematical Representation:

$$R(\gamma, B, \mu) = 1 - B_{Loss} \quad (2.13)$$

Robustness Metrics

Robustness in this context can be divided into multiple categories.

1. Robustness with respect to variations in the input traffic characteristics:

Typically the links in the middle part of the network can have variations in the input traffic characteristics due to multiplexing (particularly we are interested in the safety critical data). In this context, our network should still guarantee the safe delivery. Thus the property of change in reliability with respect to variations in input traffic can serve as useful robustness metric.

Mathematical Representation:

$$Robustness(\Delta\gamma, B, \mu) = \frac{R(\gamma, B, \mu) - R(\gamma - \Delta, B, \mu)}{\Delta} \quad (2.14)$$

Note, Δ , in this case, is discrete. Hence, since we are interested in small variations in the characteristics, we can choose Δ to be one time unit. In other words, the whole inter-arrival input probability distribution is shifted to the left by one time unit.

2. Robustness with respect to variations in the service traffic characteristics:

Similar to the above, it is also important to study the robustness of the network with respect to variations in the service characteristics of the network. Non-safety critical data, when multiplexed with the safety critical data, consume buffer space and service time. The variations just due to change in the service characteristics can be captured as follows.

Mathematical Representation:

$$Robustness(\gamma, B, \Delta\mu) = \frac{R(\gamma, B, \mu) - R(\gamma, B, \mu + \Delta)}{\Delta} \quad (2.15)$$

Notice, the shift in the service time distribution is towards right.

3. Robustness with respect to variations in buffer size:

Similar to the above cases, the variations just due to change in the effective buffer capacity can be captured as follows.

Mathematical Representation:

$$Robustness(\gamma, \Delta B, \mu) = \frac{R(\gamma, B, \mu) - R(\gamma, B - \Delta, \mu)}{\Delta} \quad (2.16)$$

4. Unified Robustness:

Finally, combining all the above, the robustness due to variations of all the elements at the same time can be computed as follows.

Mathematical Representation:

$$Robustness(\Delta\gamma, \Delta B, \Delta\mu) = \frac{R(\gamma, B, \mu) - R(\gamma - \Delta, B - \Delta, \mu + \Delta)}{\Delta} \quad (2.17)$$

The relation between these metrics based on the problem nature would be a interesting future work to study.

Cost Metrics

The cost in this context of buffer losses can be divided into two categories.

1. Buffer Size (B)
2. Energy Consumption of the switch: $(k_{Operating} \cdot f^2) * (1 - P_{Idle}) + (k_{Idle} \cdot f^2) * (P_{Idle})$

(In this thesis, we don't concern much with the energy consumption of the switch. We deal with computing the idle time of the switch (Section ■■, equation ■■ and it is left to interested people to compute the energy consumption according to their configuration $(k_{Operating}, f)$).

2.4 Summary

In this chapter mathematical models of In-Vehicle Ethernet Network topology and its components were developed. Together these form the Network configuration shown in the figure 1.8. Metrics for quantifying reliability, robustness and cost of the network are also developed. However, we still need an analytical model to map these two components to perform quantitative analysis. In the next chapter we aim to find the suitable analytical model to solve this problem.

Chapter 3

Analytical Modeling

In Chapter 2, mathematical representation in the form of network configuration and metrics are developed. The focus of this chapter is to find a suitable analytical model for mapping network configuration to metrics.

Performance can be measured either using analytical methods or through simulation. Analytical methods use mathematical models to represent the system and performance measures are evaluated by this model using mathematical solutions. Simulation methods, on the other hand, represent the system by simulating the actual process. In these processes, non-deterministic events are simulated using random processes. The disadvantages with simulation are that it requires large computing time and hence not used extensively if alternate analytical methods are available. On the other hand, if the analytical methods are too complex to model or when too much simplification is done during the modeling process, the probabilistic simulations produce better approximations. Either way, the results in both cases are only as good as the model derived for the system, appropriateness of evaluation technique (analytical/simulation) and the quality of assumptions used for modeling various components of the system [8].

The organization of this chapter is as follows. In Section 1.3, we classified the faults into three types. Building analytical models for the combined case is complex (huge state-space) and computationally intensive. Moreover, it is relatively harder to get insights from the combined model. For this reason, the analysis is split into individual parts. Section 3.1 focuses on analytical models at network level. In other words the focus in this section is to find appropriate analytical model to analyze link and node losses. Section 3.2 focuses on analytical modeling at component level, specifically on buffer losses. This is followed by thesis contributions made during this thesis period in Section 3.3. In Section 3.4, necessary background that will prove useful in the subsequent chapters is discussed.

3.1 Link and Node Losses

The section is divided into two parts. Subsection 3.1.1 deals with a survey of common analytical models used in reliability analysis, a comparison between these models, and finally their appropriateness to our problem. Subsection 3.1.2 focuses on existing analysis in literature related to the analysis of FRER.

3.1.1 Common Analytical Models used in Reliability Analysis

Some of the commonly used analytical models for reliability analysis of systems are Reliability Block Diagrams (RBD), Fault Trees(FT), and Markov chains (MC)[8]. The criteria for choosing these analytical models depend on the problem nature. Table 3.1 shows the comparison of

| Features | Reliability Block Diagram | Fault Tree | Markov Chain |
|---|---------------------------|------------|--------------|
| Success Domain | ✓ | | ✓ |
| Failure Domain | | ✓ | ✓ |
| Top-Down Approach | ✓ | ✓ | ✓ |
| Identification and prevention of faults | ✓ | ✓ | ✓ |
| Combinatorial Problems | ✓ | ✓ | ✓ |
| Non- Combinatorial Problems | | | ✓ |
| Large and Complex systems | ✓ | ✓ | |

Table 3.1: Table showing comparison of RBD,FT and Markov Chains [1]

Reliability Block Diagrams(RBD), Fault Trees(FT) and Markov chains(MC).

From the table, we can observe that Reliability Block Diagrams(RBD) and fault trees (FT) are mostly similar. The difference is that the RBD's are used if we are interested in the successful working of the system while fault trees are useful for modeling the failure relationship between individual components and the failure of the system. The difference between these models and Markov Chains is that Markov Chains can model non-combinatorial problems where RBD and FT fail. To illustrate this example better, consider the network is shown in figure 3.1. Links A and B have pass probability of p_A and p_B respectively and the remaining links are ideal. Denote $F(A)$ and $F(B)$ as the failure probabilities of links A and B . Therefore

$$F(A) = 1 - p_A$$

and

$$F(B) = 1 - p_B$$

The listener doesn't receive a packet if both links A and B fails. Denote $R(p_A, p_B)$ as the reliability of the network. Therefore,

$$R(p_A, p_B) = 1 - F(A, B)$$

Now if links A and B are independent, then

$$R(p_A, p_B) = 1 - F(A, B) = 1 - F(A)F(B) = 1 - (1 - p_A)(1 - p_B) \quad (3.1)$$

However consider the case where interference from the external source effects both the links at the same time (for example, due to the close proximity of both links to each other and to the interference source). In this case, the reliability of the network transposes to

$$R(p_A, p_B) = 1 - F(A, B) = 1 - F(A)F(B|A) = 1 - F(B)F(A|B) \quad (3.2)$$

When link failures are completely dependent $F(A|B) = 1$ we get $R = 1 - p_A = 1 - p_B$. In this case, redundancy is not useful. In other words, the reliability of the network depends on how external interference(s) affect the joint failure probability of the individual components. Extending this behavior to interference sources with dynamic nature (some time effects jointly and at some other time the failures are independent) computing reliability requires, modeling these failure dependencies between component failures in-accordance to the state of the external source. This modeling is not feasible using RBD's and fault trees where total component independence is assumed [43]. Markov chains, on the other hand, allow modeling these dependencies through the state transition probability matrix. This flexibility means that steady-state probabilities can be more accurately calculated (from the state transition probability matrix). When independent failure assumption is used, the steady-state probabilities transposes to product form representation (similar to equation 3.1). In [44] more information is provided using an example analysis as to why the results of fault trees are conservative compared to Markov Chains. The downside with Markov chains is that the computation time and storage required grows exponentially and hence not suitable for larger or complex systems (as shown in table 3.1). This can, however, be mitigated by using decomposition

techniques [43],[18] to partition the network into multiple parts and analyzing each component separately. Another way is to use approximation algorithms exploiting the properties of state-space (one such is used in chapter 7). Both of the above approaches result in approximate outputs and the error depends on the appropriateness of the assumptions and modeling technique used. While the analysis used in this thesis assumes independent failures of components, the analysis is built using Markov chains mainly due to the flexibility they offer. Markov chain analysis operates on the concept of states (tuple of inputs). While this approach is good for modeling finer details, in order to perform quantitative analysis, states must be mapped to outputs of interest. This can be solved using techniques from graph theory, the inspiration for which is found in article [25]. In the next subsection, existing analysis directly related to FRER is discussed

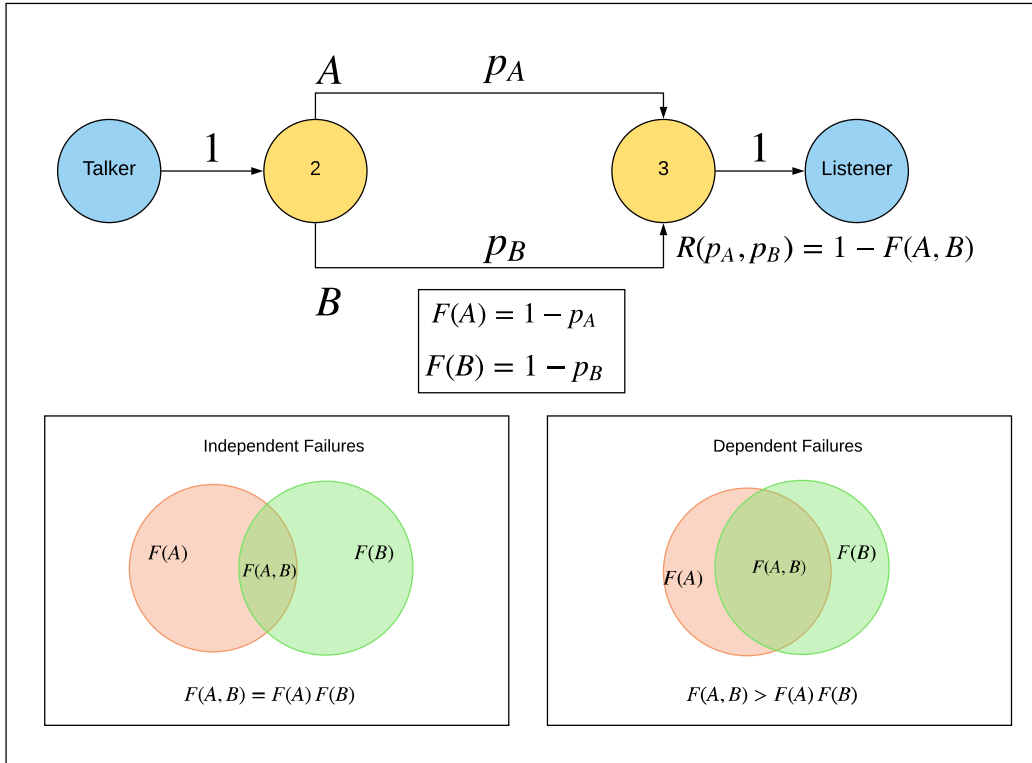


Figure 3.1: Figure illustrating network with links having dependent and independent failures.

3.1.2 Technical Papers related to the analysis of IEEE 802.1CB

In this Section, technical articles containing analysis related to FRER and their limitations are described. From the timeline of TSN standards shown in Figure 1.4, it can be observed that development of the FRER (IEEE 802.1CB) standard is relatively new. For this reason, there is not much available in literature related to the analysis of FRER. Some basic analysis is presented in the articles [14] and [25]. Article [14] focuses on only the node losses and is also only applicable to a partial ladder (Appendix C [22]) or core attached ring networks. Also, the analysis is performed using approximations. Moreover, the analysis is performed for only single node failure cases.

Article [25] extends the analysis made in article [14] by considering both link and node failures and also the analysis is not limited to particular kind of topologies. The approach enumerates all the possible states and computes the output of each state using reachability algorithms. However, it has the following limitations.

1. It assumes failures of components are independent of each other. While this is a reasonable assumption, the article does not consider the idea of dependent failures and appropriate methods to model such cases.
2. The reachability concept is confusing (the matrix A is described as the adjacency matrix of the graph G , while its contents show reachability between nodes in a particular state) and is computationally intensive ($O(|V|^3 \cdot 2^{|V|+|E|})$). However, this serves as an inspiration for using reachability algorithms to map outputs to states.
3. The methodology used to improve the computational speedup is to use the Warshall algorithm for network condensation, which doesn't apply to networks like completely ladder redundant topologies (Appendix C [22]).

Moreover, analysis related to the robustness of the network due to these failures is not studied in either of the papers. In the next section, analytical models related to buffer loss analysis is studied.

3.2 Buffer Losses

As described in section 2.3, buffer loss probability depends on input inter-arrival distribution (γ), Buffer Size (B) and service time distribution (μ). As pointed out in [44] fault trees and RBD's are not suitable for time-evolving networks. "It is not possible to use methods such as fault tree analysis, to assess the reliability or the availability of time evolutive systems. Stochastic processes have to be used and among them the Markov processes are the most interesting ones[44]." For this reasons analysis related to buffer loss is mostly done using queuing theory [42], where the analytical model used is Markov Chains. Markov Chains are usually classified into two categories: Discrete Time Markov Chains(DTMC) and Continous Time Markov Chains (CTMC). The choice of which depends on the system behavior. CTMC should be used in cases γ and μ are continuous time distributions (or to get an approximation when there are many discrete states) and DTMC if both are discrete-time.

Birth-death process is one of the commonly used stochastic processes to model queuing networks such as in telecommunications [17]. A birth-death process is a CTMC process with a state-space of the form $S = 0, 1, \dots, I$ (I can be infinite). All the transitions from state i to state $i + 1$ is considered as birth and transitions from state i to state $i - 1$ is considered as death. Figure 3.2 shows a birth-death process for a buffer of size B with $B + 1$ states($0 - B$). In case of Poisson arrivals and exponential service time distributions, all the probabilities corresponding to the transitions from state i to state $i + 1$ becomes γ and all the probabilities corresponding to the transitions from state i to state $i - 1$ is equal to μ . This only applies to Poisson arrivals, exponential service time distributions. The loss analysis for these systems is represented according to Kendall's notation as M/M/1/K [42]. However, the downside is that most practical situations do not follow this property.

The loss analysis corresponding to general arrivals and general service time distributions are represented according to Kendall's notation as G/G/1/K, where G corresponds to general distributions and K corresponds to the finite queue capacity. Computing exact values in the case of G/G/1/K for CTMC's is not possible and only approximations are available. In these kinds of cases, additional parameters have to be included to represent the state of the system such as the remaining time until next arrival and remaining time for the service completion of the existing packet. One such methodology is used in [26], where an approximate value for computing the loss probabilities of a for G/G/1/K queue. This approach uses the first two moments of the arrival and service distributions for computing approximate values for the probabilities in the birth-death process. While the paper claims the approximation is exact for M/M/1/K systems, a closer look shows that it is not true. However, the part until before the approximations (sections 1 until

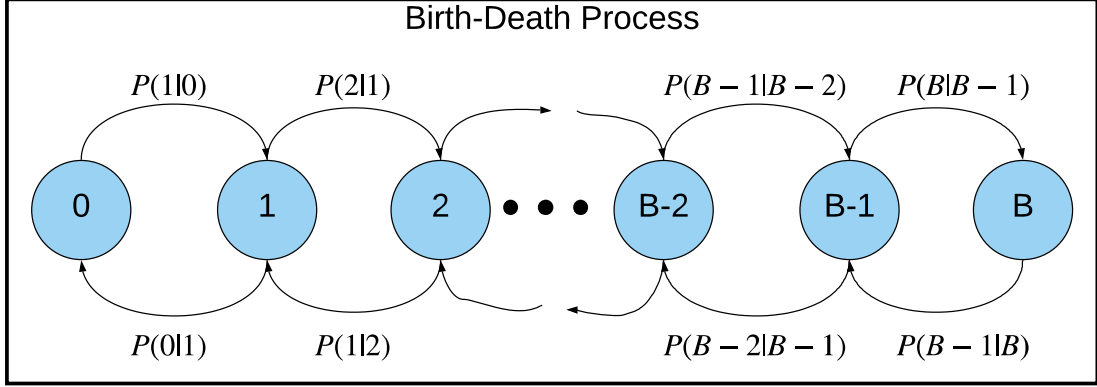


Figure 3.2: Figure showing birth death process.

3) are accurate and served as an inspiration for modeling the discrete time variant for general distributions described in chapter 6.

In the next section contributions from this thesis are discussed.

3.3 Thesis Contributions

This section contains the set of contributions made during this thesis. They are as follows

1. Development of metrics for quantifying reliability, robustness and cost of the network (Chapter 2).
2. Development of analytical models to study the impact of FRER using Markov chains and Graph Theory techniques (Chapters 3, 4, 6).
3. Development of code implementing the analytical models and code for validating these analytical models in MATLAB/Simulink. (Chapters 4, 5, 6).
4. Methods for reducing the time complexity of the algorithms developed using parallel processing and approximation algorithms(Chapter 7).
5. Quantitative study of commonly used topologies and Design Space Exploration for optimizing reliability, robustness and cost of the network (Chapters 4, 5, 8).
6. Future Work Suggestions based on my work (Chapter 9).

Currently, the work is binded by a confidentiality agreement signed by me with NXP. For this reason, no publications of this work are planned. This might however change in future based on NXP's decision.

3.4 Background

In this section, necessary background concepts that will prove useful in the subsequent chapters is discussed.

3.4.1 Markov Chains

"A Markov Chain is a stochastic model describing a sequence of events in which the probability of each event depends only on the state attained in the previous event" [29]. Markov Chains are broadly divided into two categories, Discrete Time Markov Chains (DTMC) and Continuous Time Markov Chains (CTMC) based on the discrete set of times or continuous set of times used for analysis. In this project we mainly use DTMC. A brief back ground of DTMC is as follows.

A discrete time Markov Chain is a sequence of random variables X_1, X_2, \dots satisfying the Markov property, which is the probability of moving to next state solely depends on the current state and not on the previous state. Therefore,

$$Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) = Pr(X_{n+1} = x | X_n = x_n)$$

provided both the conditional probabilities are well defined,i.e

$$Pr(X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) > 0$$

A time homogeneous Markov Chain is a Markov Chain where the probability of transition is independent of n (time).

$$Pr(X_{n+1} = x | X_n = y) = Pr(X_n = x | X_{n-1} = y)$$

Denote S the set of countable states that X can take and denote the probability of transitioning from state i to state j as p_{ij} . Hence the transition probability matrix is of size $|S| \times |S|$.

Therefore, the probability of going from state i to state j in 1 time step is

$$p_{ij} = Pr(X_{n+1} = j | X_n = i)$$

In this project we are mainly interested in the steady state distribution. The vector π is a stationary distribution provided

$$\begin{aligned} 0 &\leq \pi_j \leq 1 \\ \sum_{i \in S} \pi_i &= 1 \\ \pi_j &= \sum_{i \in S} \pi_i p_{ij} \end{aligned}$$

Throughout this project we assume necessary conditions required for existence of stationary distribution are satisfied. For more information on Markov Chains please refer [29].

The background needed for other concepts are provided in appropriate chapters.

3.5 Summary

In this Chapter we studied various analytical models commonly used for quantitative analysis. A comparative study is done on those models and Markov Chains are proved suitable for all the analysis because of the flexibility they offer for modeling component dependent failures. Later in Section 3.3, thesis contributions during this project followed by a brief background of Markov chains needed for further analysis in this project is discussed in Section 3.4. In the next chapter we analyze link and node losses using Markov Chains.

Chapter 4

Link and Node Loss Analysis

In Section 1.3, packet losses in the network are classified into multiple types based on their location and nature, and in Chapter 3, we chose to perform analytical modeling using Markov chains. In this Chapter, we focus on the analysis related to packet loss on links and nodes. In Chapter 6, we analyze the losses due to the buffer overflow. Broadly, the chapter is divided into the two components namely *Analysis* and *Quantitative Study using Symbolic Math*. The organization of these components is as follows

1. **Analysis:** The analysis component focuses on deriving the expressions for the metrics formulated in Section 2.2.3 using *Markov Chains* and *Graph Theory*. In Section 4.1, expressions for the metrics are derived for a single link network. In Section 4.2, this analysis is extended to a multi link network. In both these Sections it is assumed that node losses are ignored. Finally, in Section 4.3, we analyze both link and node losses together. In all these analysis, it is assumed that the input packet arrival and the failure of the components are independent. Also, it is assumed that the state-space of the Markov Chains are finite and limiting distributions exist.
2. **Quantitative Study using Symbolic Math:** The analysis described in section 4.2 is automated in MATLAB. The focus of this component is to use this software and perform a quantitative study on example networks, with the objective of analyzing and comparing networks quantitatively. Section 4.4 presents the results using symbolic math with a focus on understanding the significance of metrics and their components using an example network with only link losses. Section 5.1 focuses on comparative study of important example cases.

4.1 Analysis of a single link network

In this Section, the analysis for a network consisting of a single link is studied. The objective is to derive the metrics described in Section 2.2.3 using the workflow shown in the Figure 4.1. We start by formulating the state space of the network from the network description. State space is defined as the set of all the possible states the network can be in. In the next step, we compute the steady-state probabilities of every state in the state space. These probabilities are proportional to the average amount of time the network spends in a particular state. This is followed by computing the reachability for every state. Reachability of a state provides the information whether the network can deliver packets successfully or not when it is in a particular state. Finally, all this information is used to compute the metrics of interest. The analysis for a single link network is as follows.

Network Description:

The network consisting of only a single link is shown in figure 4.2. Packets are transmitted from talker to the listener across this link.

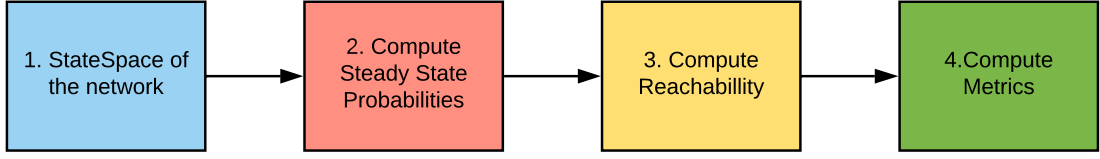


Figure 4.1: Figure showing the basic work flow for computing metrics

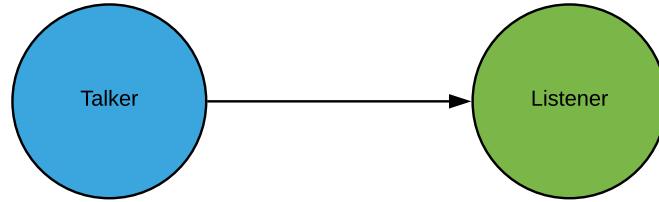


Figure 4.2: Single Link Network

State Space of the Network:

A link can be in either *Good* or *Fail* states. *Good* state is the state where the link successfully transmits the frame through it, while *Fail* state indicates the state where the link discards the frame it received and does not transmit the frame across itself. Thus the state-space of a single link network is

$$S = \{0, 1\} \quad (4.1)$$

where 0 represents *Fail* state and 1 represents *Good* state.

Steady state probabilities of the states:

Denote by p_1 the transition probabilities of changing from *Fail* state to *Good* state, and denote by p_2 the transition probabilities of changing from *Good* state to *Fail* state. The Markov chain of the link can thus be represented as shown in figure 4.3.

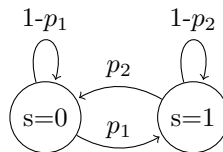


Figure 4.3: Markov Chain of a single link network

The state transition probability matrix of a network with two states 0 and 1 is of the form

$$T = \begin{bmatrix} P(0|0) & P(1|0) \\ P(0|1) & P(1|1) \end{bmatrix}$$

where $P(s_j|s_i)$ denote the probability of network in state s_i transitioning to state s_j in the next time step.

Thus, the state transition probability matrix for the Markov chain shown in figure 4.3 is as follows:

$$T = \begin{bmatrix} P(0|0) & P(1|0) \\ P(0|1) & P(1|1) \end{bmatrix} = \begin{bmatrix} 1 - p_1 & p_1 \\ p_2 & 1 - p_2 \end{bmatrix}$$

Denote by P^i the state probability matrix at time step i . One step transition probabilities can be calculated by multiplying the transition matrix with the current state probability matrix. For example for an initial state of $P^0 = [1 \ 0]$ the one-step transition probabilities P^1 can be computed as $P^1 = P^0 \cdot T$. We are however interested in the steady-state probabilities. We know at steady state the probabilities get saturated and doesn't change with further time steps. Therefore for a time step n where $n \rightarrow \infty$, the following holds

$$P^n \cdot T = P^{n+1} = P^n$$

$$P^n \cdot T = P^n \tag{4.2}$$

Let $P^n = [P0 \ P1]$ be the steady state probabilities of network being in *Fail* and *Good* states respectively. Substituting in equation 4.2, we get,

$$P0 \cdot (1 - p_1) + P1 \cdot p_2 = P0$$

and

$$P0 \cdot p_1 + P1 \cdot (1 - p_2) = P1$$

.

$$\frac{P0}{P1} = \frac{p_2}{p_1}$$

we also know that the sum of probabilities of all states at any given point of time is 1 ($P0 + P1 = 1$). Therefore we get,

$$P0 = \frac{p_2}{p_1 + p_2} = \frac{P(0|1)}{P(1|0) + P(0|1)}$$

and

$$P1 = \frac{p_1}{p_1 + p_2} = \frac{P(1|0)}{P(1|0) + P(0|1)}$$

Typically, the next state of the link doesn't depend on the current state (memoryless). Therefore, we get

$$P(0|1) = P(0|0) = P0$$

$$P(1|0) = P(1|1) = P1$$

Applying in above equations we get, $p_1 + p_2 = 1$, $P0 = 1 - p_1$ and $P1 = p_1$. In other words, the steady state probabilities of the network to be in *Fail* and *Good* States are $1 - p_1$ and p_1 respectively.

Reachability of States:

From the network architecture, since we assume there are no other losses for the packet, we can determine that the packet is successfully transmitted whenever the link is in good state.

Denote by $f : S \mapsto X$. Therefore we have,

$$f(0) = 0, \text{ and } f(1) = 1$$

That is,

$$X = \begin{cases} 0, & \text{if } s = 0, \\ 1, & \text{if } s = 1 \end{cases}$$

4.1.1 Deriving expressions for reliability metrics

The reliability of the network according to the metric 2.2 is

$$\begin{aligned} Pr(X = 1) &= \sum_{s \in S} Pr(X = 1|s) \cdot Pr(s) = \sum_{s \in S} f(s) \cdot Pr(s) \\ Pr(X = 1) &= \sum_{s \in S} f(s) \cdot Pr(s) = (1) \cdot p_1 + 0 \cdot (1 - p_1) = p_1 \end{aligned}$$

$$Pr(X = 1) = p_1 \quad (4.3)$$

The reliability of the network for successfully delivering the packet is p_1 which in this case is equal to the reliability of the link.

Since the network consists of only a single link, the reliability of the network under no failure and 1 failure (corresponding to metric 2.3) is

$$Pr(X = 1, failures^{link} = 0) = p_1 \quad (4.4)$$

$$Pr(X = 1, failures^{link} = 1) = 0 \quad (4.5)$$

Similarly, the cumulative reliability of the network under no failure and single failure case (corresponding to metric 2.5) is

$$Pr(X = 1, failures^{link} \geq 0) = p_1 \quad (4.6)$$

$$Pr(X = 1, failures^{link} \geq 1) = 0 \quad (4.7)$$

(Note: Since we assumed no node losses, $failures^{node} = 0$ and not shown in the above equations to keep them simple)

4.1.2 Deriving expressions for robustness metrics

In the previous subsection, we derived the metrics for reliability. Denote function $R(p_1)$ as the reliability of the network. Therefore from equation 4.3,

$$R(p_1) = p_1$$

The expressions for robustness metrics are as follows (corresponding to metrics 2.8, 2.9 and 2.10 respectively)

$$\frac{\partial R}{\partial P_i} = \frac{\partial p_1}{\partial p_1} = 1 \quad (4.8)$$

$$\frac{\sum_{i=1}^N \frac{\partial R}{\partial P_i}}{N} = \frac{\partial p_1}{\partial p_1} = 1 \quad (4.9)$$

$$\frac{1}{N} \cdot \frac{\Delta R}{\Delta p} = \frac{R(p_1, p_2, \dots, p_N) - R(p_1 - \Delta, p_2 - \Delta, \dots, p_N - \Delta)}{\Delta \cdot N} = \frac{p_1 - (p_1 - \Delta)}{\Delta \cdot 1} = \frac{\Delta}{\Delta} = 1 \quad (4.10)$$

Notice, the value of robustness metrics in all the cases is equal to 1. This indicates the network is highly sensitive to variations in the pass probability of the link. This can be expected since the network has no redundancy. In section 4.2, we extend the analysis to a multi link network.

4.2 Analysis of Multi Link Network

In the previous Section, we computed the metrics for a network consisting of only one link. In a multi-hop network, the packet has to pass through multiple links before reaching its destination. In this section, following similar workflow is shown in figure 4.1, we explore the state space of the network and determine the steady-state probabilities of each state. Later we compute reachability for each state using graph theory techniques. Based on this data, we compute the metrics of interest in the subsequent subsections.

Network Description:

The network can be represented as an undirected graph with switches as vertices and edges as the links connecting them. The weights of the edges represent the pass probabilities of packets passing through the links. Therefore a network can be represented as follows. $G = (V, E)$ where G is the underlying network topology. V represents the set of vertices (Switches/EndSystems) and E represent the set of links connecting the vertices.

The number of links (N) in the network from the graph description is equal to the number of edges ($N = |E|$). For example, consider the network represented by the undirected graph shown in the figure 4.4. For this graph

$$G = (V, E)$$

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{(1, 2), (2, 3), (2, 4), (3, 4), (3, 5), (4, 5), (5, 6)\}$$

$$N = |E| = 7$$

$$M = |V| = 6$$

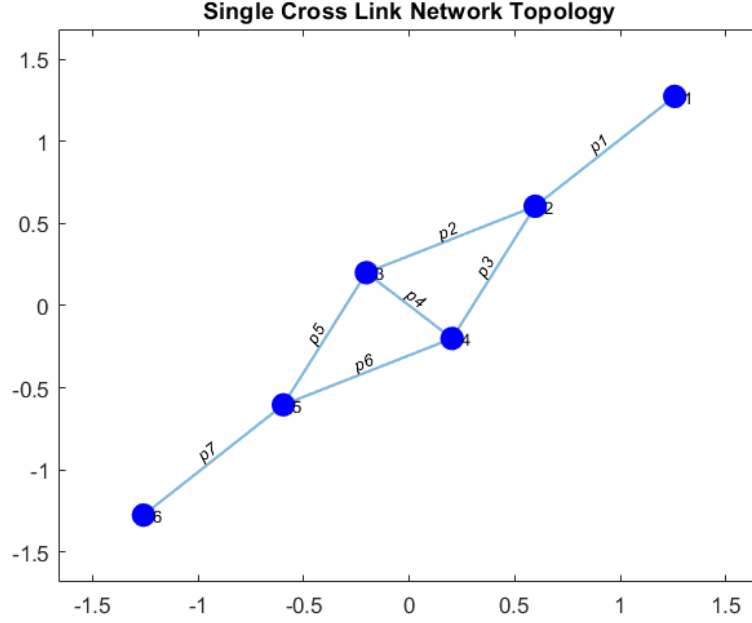


Figure 4.4: Figure showing a sample single cross link network along with its pass probabilities

State Space of the Network:

As mentioned in the previous section, each link can be in two states. Therefore for a network with a total of N links we have a total of 2^N states. The state of the system can thus be defined as a tuple of states of the all the links.

$$S_j = (s_{1j}, s_{2j}, \dots, s_{Nj}), \quad j \in \{0, 1, 2, 3, \dots, 2^N - 1\}$$

where $s_{ij} \in \{0, 1\}$ represents the state of link i , where $i \in \{1, 2, 3, \dots, N\}$ corresponding to the j^{th} state of the system.

Now the probability of state S_j is (from Baye's chain rule [9]),

$$Pr(S_j) = Pr(s_{1j}, s_{2j}, \dots, s_{Nj}) = Pr(s_{1j}) \cdot Pr(s_{2j}|s_{1j}) \cdot Pr(s_{3j}|(s_{1j}, s_{2j})) \cdots Pr(s_{Nj}|(s_{1j}, s_{2j}, \dots, s_{(N-1)j}))$$

Thus, the state-space of the network is

$$S = \{S_j | j \in \{0, 1, 2, \dots, 2^N - 1\}\}$$

Steady state probabilities of the states(Independent component failures):

Considering all the links are independent of each other in any given state of the network.

$$Pr(s_{ij}|C) = Pr(s_{ij}) \quad \forall \quad j \in \{0, 1, 2, 3, \dots, 2^N - 1\}, \quad i \in \{1, 2, 3, \dots, N\}, \quad C \in S$$

Therefore,

$$Pr(S_j) = Pr(s_{1j}, s_{2j}, \dots, s_{Nj}) = \prod_{i=1}^N Pr(s_{ij}) \quad (4.11)$$

Since p_i denotes the probability that link i is in 'Good' state.

$$Pr(s_{ij}) = \begin{cases} p_i, & \text{if } (s_{ij} = 1) \\ 1 - p_i, & \text{if } (s_{ij} = 0) \end{cases} \quad (4.12)$$

Since this is mostly the case and focus of FRER, we use these equations (4.11 and 4.12) for computing the steady state probabilities of states which are applied in all the quantitative studies discussed in the rest of the chapters.

Steady state probabilities of the states(Dependent component failures):

In the the case of state-dependent transitions, we need to use a similar approach as used for single network analysis. Create Markov Chain with state transition matrix representing the dependency of the states. The network consists of 2^N states, therefore the state transition matrix T is a $(2^N \times 2^N)$ matrix. The matrix is as follows.

$$T = \begin{bmatrix} Pr(S_0|S_0) & Pr(S_1|S_0) \cdots & Pr(S_{2^N-1}|S_0) \\ \vdots & \vdots & \vdots \\ Pr(S_0|S_{2^N-1}) & Pr(S_1|S_{2^N-1}) \cdots & Pr(S_{2^N-1}|S_{2^N-1}) \end{bmatrix}$$

Using similar analysis at steady state

$$P^n \cdot T = P^{n+1} = P^n$$

$$P^n \cdot T = P^n$$

$$P^n = [Pr(S_0) \quad Pr(S_1) \cdots \quad Pr(S_{2^N-1})]$$

Thus equations for the steady-state probabilities, (observe we get the equation for the law of total probability [28])

$$Pr(S_j) = \sum_{i=0}^{2^N-1} Pr(S_j|S_i) \cdot Pr(S_i)$$

For computing steady state probabilities, one needs to solve the above linear equations. Solving these linear equations is out of scope of this project but appropriate methods efficient(time) can be found at [34].

Notice, under memoryless conditions, we have $Pr(S_j|C) = Pr(S_j)$ and we reach the same conclusion derived before.

Reachability of states:

Define *Reachable* : $S \mapsto X$, where

$$Reachable(s) = \begin{cases} 1, & \text{if path exists between talker and listener in state } s, \\ 0, & \text{otherwise} \end{cases}$$

Therefore,

$$I = \{Reachable(s)|s \in S\}$$

for which I denotes a set of indicator variables $I = \{I_j|j \in \{0, 1, 2, \dots, 2^N - 1\}\}$ where I_j indicates whether the packet is successfully passed or not when the network is in the State j . I_j is 1 if there exists a path between the talker node and listener node when the network is in State S_j else it is 0. This can be determined by running a reachability algorithm on the modified graph (G_j) corresponding to the State S_j .

There are many ways to determine the reachability between two nodes. In this project, *shortestpath* algorithm is used to verify if a path exists between the talker and listener node. If the returned path is not empty then $I_j = 1$.

| Method Name | Description | Worst Case Time Complexity |
|--------------|---|----------------------------|
| BFS | Assumes weights of the edges are equal | $O(V + E)$ |
| Dijkstra | Assumes weights of the edges are positive | $O(\log(V) \cdot E)$ |
| Bellman-Ford | Assumes that weights of the edges are all non zero | $O(V \cdot E)$ |
| Acyclic | Input Graph must be acyclic. Assumes that weights of the edges are all nonzero | $O(V + E)$ |

Table 4.1: Different Algorithms for computing shortest path between nodes in a graph, their descriptions and worst case time complexity (Reference: [38]).

This can also be done by other methods, for example, by observing the steady state probability of Listener Node corresponding to the Markov chain with state transition probability matrix as normalized adjacency matrix of the modified graph (> 0 implies reachable), performing breadth-first search or depth-first search on the modified graph starting from talker node and check if the listener node is part of the visited node list.

Currently, *graphshortestpath* algorithm from MATLAB library [38] is being used to compute the shortest path between talker and listener. Computing reachability using *shortestpath* helps to compute net flows out of the links, the analysis of which is shown in Section 4.2.3.

Table 4.1 shows options of various methods used in the library. From the time complexity we can observe as the network size increases, 'BFS' performs better than the rest. However, it has to be noticed that this method operates under the assumption that all the weights of the edges are equal. This is not a problem in our case since we are only interested in the paths packets take to reach the listener. The pseudo code of the algorithm for computing the reachability of all the states is shown in algorithm 1.

For example, consider the state of the network shown in figure 4.4 where the cross link (link with bidirectional flow (3,4)) fails. In this case, the network transposes to the one shown in figure 4.5

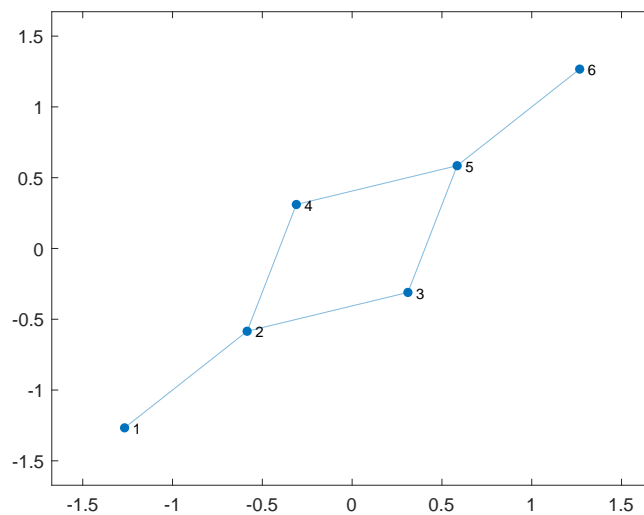


Figure 4.5: State of the network shown in figure 4.4 when cross link fails

The probability of the network to be in this state (assuming independence of links) is $p_1 p_2 p_3 (1 - p_4) p_5 p_6 p_7$. The network is able to deliver packets successfully, through path $1 - 2 - 4 - 5 - 6$ or through $1 - 2 - 3 - 5 - 6$. Hence I_j corresponding to this state is 1. Now consider the states where links connected to the source or listener fails. I_j 's corresponding to these states are 0 since these links are not protected by redundancy.

Note: The purpose of all the pseudo code provided in this thesis is to provide a high level overview of the implementation. For exact working of the algorithms, the reader is advised to refer the implemented code.

4.2.1 Deriving expressions for reliability metrics

In this subsection, we derive expressions for metrics 2.2, 2.3 and 2.5 described in section 2.2.3.

$$Pr(X = 1) = \sum_{s \in S} Pr(X = 1|s) \cdot Pr(s) = \sum_{s \in S} Reachable(s) \cdot Pr(s)$$

$$Pr(X = 1) = \sum_{s \in S} Reachable(s) \cdot Pr(s) = \sum_{j=0}^{2^N-1} I_j \cdot Pr(S_j) \quad (4.13)$$

The pseudo code of the algorithm for computing reliability of the network is shown in the algorithm 2.

(P_{Link} in the pseudo code denotes the array of link probabilities. $P_{Link}(i) = p_i$, for all $i \in \{1, 2, \dots, N\}$)

In order to compute the reliability of the network under exactly n link failure cases, we need to first isolate the states containing exactly n failures. Let $K_n \subseteq S$ represents a collection of states where the system is subjected to exactly n link failures. for which,

$$K_n = \{S_j | \text{if } \sum_{i=1}^N s_{ij} = N - n, S_j \in S\}$$

$$Pr(X = 1, failures^{link} = n) = \sum_{s \in K_n} Reachable(s) \cdot Pr(s) \quad (4.14)$$

The cumulative probability distribution for reliability when subjected to n link failures can be computed from the above expression as

$$Pr(X = 1, failures^{link} \geq n) = \sum_{i=n}^N Pr(X = 1, failures^{link} = i) \quad (4.15)$$

Notice, in the above equation, the value of i decreases from N to n and at $n = 0$, $Pr(X = 1, failures^{link} \geq n)$ gives the overall reliability of the network.

(Note: Since we assumed no node losses, $failures^{node} = 0$ and not shown in the above equations to keep them simple)

Denote by R , the overall reliability of the network,

$$R(p_1, p_2, \dots, p_N) = \sum_{j=0}^{2^N-1} I_j \cdot Pr(S_j) \quad (4.16)$$

In the next subsection we derive metrics for robustness of the network. R is used as a substitute in accordance with the equation 4.16 whenever simplification is required. Notice, R is a function of all the pass probabilities of the components. For more intuition on this, refer the symbolic math analysis done in section 4.4.

4.2.2 Deriving expressions for robustness metrics

From the above reliability function, the robustness of the network with respect to perturbations in the pass probability of link i (corresponding to 2.8) can be computed as

$$\frac{\partial R}{\partial p_i} = \frac{\partial(\sum_{j=0}^{2^N-1} I_j \cdot Pr(S_j))}{\partial p_i} \quad (4.17)$$

Similarly, the mean and collective robustness of the network (corresponding to the metrics 2.9 and 2.10 respectively) can be computed as

$$\frac{\sum_{i=1}^N \frac{\partial R}{\partial p_i}}{N} = \frac{\sum_{i=1}^N \frac{(\partial(\sum_{j=0}^{2^N-1} I_j \cdot Pr(S_j)))}{\partial p_i}}{N} \quad (4.18)$$

$$\frac{1}{N} \cdot \frac{\Delta R}{\Delta p} = \frac{R(p_1, p_2, \dots, p_N) - R(p_1 - \Delta, p_2 - \Delta, \dots, p_N - \Delta)}{\Delta \cdot N} \quad (4.19)$$

The analysis of these metrics for various use cases is shown in case studies section (Section 5.1)

Algorithm 1 $(I, allPaths) = ComputeReachability(S)$

1: **for** $j := 0$ to $2^N - 1$ **do**

2: Find the links that are failed in state S_j

$$Fail_j \subseteq Links$$

3: Construct the modified graph G'_j by removing all the links present in the set $Fail_j$.

$$G' = G.removeEdge(\{Link_{Id} \in Fail_j | Link_{Id} \in Links\})$$

4: Compute the shortest path between the talker and listener.

$$Path_j = ShortestPath(G'_j, Talker_{Id}, Listener_{Id})$$

5: **if** $Path_j == \emptyset$ **then**

6: $I_j = 0$

7: **else**

8: $I_j = 1$

9: **end if**

10: **end for**

11: Compute the set of unique paths (minus the null set) from the path information computed over all the states.

$$allPaths = \{unique(\{Path_j | j \in \{0, 1, \dots, 2^N - 1\}\}) \setminus \emptyset$$

12: **return** $(I, allPaths)$

Algorithm 2 $R = ComputeReliability(P_{Link})$

1: Enumerate all the states in the network for a given link count N .

$$S = \{S_j | j \in \{0, 1, \dots, 2^N - 1\}\}$$

2: **for** $j := 0$ to $2^N - 1$ **do**

3: Compute the probability of network to be in state j

$$Pr(S_j) = \prod_{i=1}^N Pr(s_{ij}) \text{ (independent failure assumption)}$$

4: **end for**

5: Compute the reachability of the network in every state. That is if a packet can be received by the listener or not for any given network state s

$$(I, allPaths) = ComputeReachability(S_j)$$

6: Compute reliability of the network (pass probability) as the sum of conditional probabilities over all the states.

$$R = Pr(X = 1) = \sum_{s \in S} Pr(X = 1 | s) \cdot Pr(s) = \sum_{j=0}^{2^N-1} I_j \cdot Pr(S_j)$$

7: **return** R

4.2.3 Computing Net-flows out of the link

In section 4.2.1 we derived expressions for computing the reliability of the network. In this section methodology for computing, the net flows out of every link in the network is discussed. Net flows out of every link can be computed using the same approach used for computing reliability of the network. This is done by appropriately masking the link probabilities and computing the reliability of the network. After all, reliability is the net-flow out of the link connected to the listener (Refer Figure 2.2).

For computing net flows out of links, we need to first create a directed graph representing the flow from talker to listener using the undirected graph. Algorithm 3 shows the pseudo code for creating directed graph using the *allPaths* information computed in the algorithm 1.

$$G_d = CreateDirectedGraph(G, allPaths)$$

where $G_d = (V, E_d)$ with E_d denoting number of edges in the directed graph.

Figure 4.6 shows results of conversion from undirected to directed for a single crosslink network. Denote N_d , the number of links in the directed graph ($N_d = |E_d|$). Since Ethernet is full duplex, in presence of cross-links, the number of directed edges will be more than the number of edges ($|E_d| \geq |E|$) in the undirected graph. This can be observed in the Figure 4.6 where the undirected edge (3,4) transformed to directed edges (3,4) and (4,3) both having the same pass probability p_4 (Thus the number of edges in the directed graph increased to 8).

Followed by this the pseudo-code for computing the net flow out of every link is shown in the algorithm 4, where $NetFlows(i)$ denotes the net flow out of the link i . Notice, how the algorithm is using the existing routine *computeReliability* with masked probabilities. An example analysis of net flows using symbolic math is presented in section 4.4.4

So far, we assumed the node losses are ignored. In the next Section, we add in the affect of node losses and show the updated algorithms needed to accommodate this change.

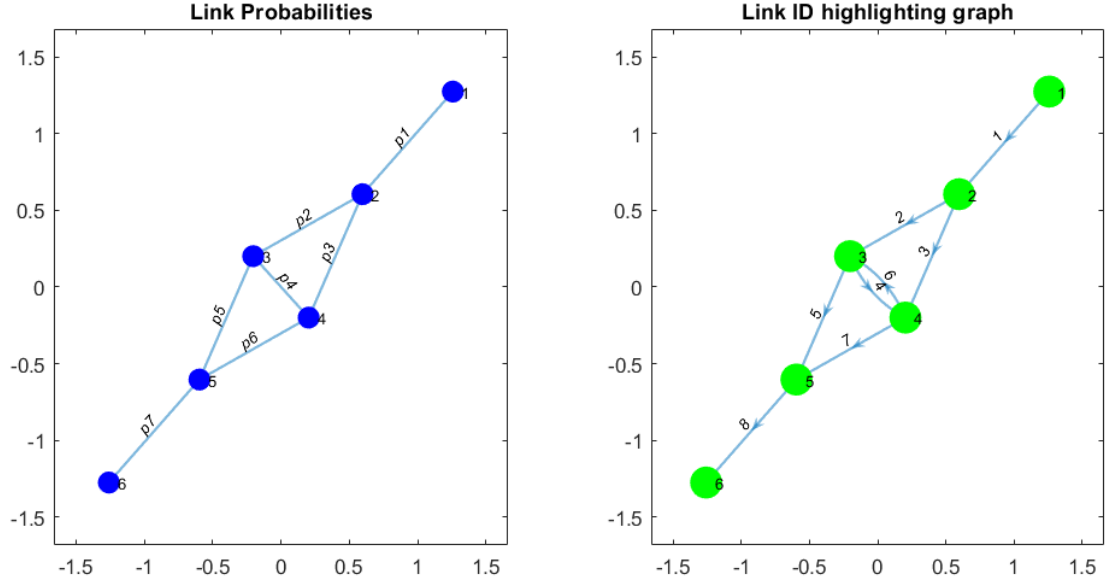


Figure 4.6: Figure illustrating the conversion of undirected graph to directed graph

Algorithm 3 $G_d = \text{CreateDirectedGraph}(G, \text{allPaths})$

- 1: **for** every vertex V_i in G **do**
- 2: Compute out neighbors set from all the paths

$$\text{OutNeighbours}_i = \text{computeOutNeighbors}(V_i, \text{allPaths})$$

- 3: **end for**
 - 4: **for** every vertex V_i in G **do**
 - 5: **for** every $t \in \text{OutNeighbours}_i$ **do**
 - 6: Add a directed edge (V_i, t) in Graph G_d .
 - 7: **end for**
 - 8: **end for**
 - 9: **return** G_d
-

Algorithm 4 $NetFlows = ComputeNetFlows(P_{Link})$

```

1: for  $i := 1$  to  $|E_d|$  do
2:   Generate mask for each link (0-Remove Link, 1-Make Link loss less, 2- Use same link
     probability as present in the configuration)

$$mask = generateMask(i, P_{Link});$$

3:   Get the masked probabilities from the generated mask.

$$P\_mask = getProbabiltyMask(mask, P_{Link});$$

4:    $Netflows(i) = computeReliability(P\_mask);$ 
5: end for
6: return  $NetFlows$ 

```

Algorithm 5 $mask = generateMask(Link_{Id}, P_{Link})$

```

1: for  $i := 1$  to  $N$  do
2:    $mask(i) = 0;$ 
3: end for
4: for  $i := 1$  to  $NumberOfPaths$  do
   (Iterate over all the paths between talker and listener)
5:    $path = allPaths(i)$ 
6:   if  $Link_{Id} \in path$  then
7:     Assign a mask value of 2 to all the links after  $Link_{Id}$  in the path.
8:     Assign a mask value of 1 to all the links after  $Link_{Id}$  in the path.
9:   end if
10: end for
11: return  $mask$ 

```

Algorithm 6 $P_{mask} = getProbabiltyMask(mask, P_{Link})$

```

1: for  $i := 1$  to  $N$  do
2:   if  $mask(i) == 0$  then
3:      $P_{mask}(i) = 0;$ 
4:   else
5:     if  $mask(i) == 1$  then
6:        $P_{mask}(i) = 1;$ 
7:     else
8:        $P_{mask}(i) = P_{Link}(i);$ 
9:     end if
10:  end if
11: end for
12: return  $P_{mask}$ 

```

4.3 Analysis of Node and Link Losses

In this Section, we perform analysis where we consider both link and node losses. The approach used in this Section to derive metrics is an extension of the analysis discussed in Section 4.2.

To quickly recap, $G = (V, E)$, where G is the underlying network topology, V represents the set of Vertices (Switches/EndSystems) and E represents the set of links connecting the Vertices.

In this case both links and switches can be in two states.

$$s^{link} \in \{0, 1\}$$

,

$$s^{node} \in \{0, 1\}$$

Thus, the state-space of the network is

$$S = \{S_j | j \in \{0, 1, 2, \dots, 2^{M+N} - 1\}\}$$

where,

$$S_j = (s_{1j}^{link}, s_{2j}^{link}, \dots, s_{Nj}^{link}, s_{1j}^{node}, s_{2j}^{node}, \dots, s_{Mj}^{node}), \quad j \in \{0, 1, 2, 3, \dots, 2^{M+N} - 1\}$$

where $s_{lj}^{link} \in \{0, 1\}$ represents the state of link l , $l \in \{1, 2, 3, \dots, N\}$ corresponding to the j^{th} state of the system, and $s_{ij}^{node} \in \{0, 1\}$ represents the state of node i , where $i \in \{1, 2, 3, \dots, M\}$ corresponding to the j^{th} state of the system. Using the similar analysis as shown in Section 4.2, and assuming independent failures between all the components we reach,

$$Pr(S_j) = Pr(s_{1j}^{link}, s_{2j}^{link}, \dots, s_{Nj}^{link}, s_{1j}^{node}, s_{2j}^{node}, \dots, s_{Mj}^{node})$$

$$= Pr(s_{1j}^{link}) \cdot Pr(s_{2j}^{link}) \cdot Pr(s_{3j}^{link}) \dots Pr(s_{Nj}^{link}) \cdot Pr(s_{1j}^{node}) \cdot Pr(s_{2j}^{node}) \cdot Pr(s_{3j}^{node}) \dots Pr(s_{Mj}^{node})$$

where,

$$Pr(s_{ij}^{link}) = \begin{cases} p_i, & \text{if } (s_{ij}^{link} = 1) \\ 1 - p_i, & \text{if } (s_{ij}^{link} = 0) \end{cases}$$

and

$$Pr(s_{ij}^{node}) = \begin{cases} p_i^n, & \text{if } (s_{ij}^{node} = 1) \\ 1 - p_i^n, & \text{if } (s_{ij}^{node} = 0) \end{cases}$$

The reachability algorithm corresponding to combining node and link losses is provided in algorithm 7. Notice, the number of states grows to 2^{M+N} . The function *MapNodetoLinks* returns the set of links connected to the node. This is done in accordance with the node loss concept discussed in 1.3, where when a node is failed, it is disconnected from all the links connected to it, and thus it can be treated analogous to where all the links connected to the failed node are failed. In other words, for a given state j , in-order to create the modified graph, we not only remove the links failed in that state, but also remove the links connected to the failed node (if they are already not removed). The methodology for computing the net-flows out of links is same as the one shown in section 4.2.3, except the state space is 2^{M+N} .

Algorithm 7 $(I, allPaths) = ComputeReachability(S_j)$

1: Identify the node to link set mapping for every node.

$$MapNodeToLinks : Node_{Id} \mapsto Links$$

2: **for** $j := 0$ to $2^{M+N} - 1$ **do**

3: Find the nodes that are failed in state S_j

$$Fail_j^{node} \subseteq Nodes$$

4: Find the links that are failed in state S_j

$$Fail_j^{links} \subseteq Links$$

5: Compute the set of links to be removed in state S_j ,

$$F = \left(\bigcup_{n \in Fail_j^{node}} MapNodeToLinks(n) \right) \cup (Fail_j^{links})$$

6: Construct the modified graph G'_j by removing all the links computed in the above step.

$$G' = G.removeEdge(\{Link_{Id} \in F | Link_{Id} \in Links\})$$

7: Compute the shortest path between the talker and listener.

$$Path_j = ShortestPath(G'_j, Talker_{Id}, Listener_{Id})$$

8: **if** $Path_j == \emptyset$ **then**

9: $I_j = 0$

10: **else**

11: $I_j = 1$

12: **end if**

13: **end for**

14: Compute the set of unique paths (minus the null set) from the path information computed over all the states.

$$allPaths = \{unique(\{Path_j | j \in \{0, 1, \dots, 2^{M+N} - 1\}\})\} \setminus \emptyset$$

15: **return** $I, allPaths$

The metrics for combined link and node losses are as follows.

4.3.1 Metrics

The process of deriving metrics is same as one shown in Section 4.2.1. For this reason, directly the resultant equations are provided.

$$Pr(X = 1) = \sum_{s \in S} Pr(X = 1|s) \cdot Pr(s) = \sum_{j=0}^{2^{(M+N)}-1} Pr(X = 1|s_j) \cdot Pr(s_j) = \sum_{j=0}^{2^{(M+N)}-1} I_j \cdot Pr(s_j)$$

$$Pr(X = 1) = \sum_{j=0}^{2^{(M+N)}-1} I_j \cdot Pr(s_j) \quad (4.20)$$

Denote by $K_{mn} \subset S$ representing the collection of states where the system is subjected to exactly n link failures and m node failures.

Therefore,

$$K_{mn} = \{S_j | \text{if } \sum_{i=1}^N s_{ij}^{link} = N - n, \sum_{i=1}^M s_{ij}^{node} = M - m, S_j \in S\}$$

$$Pr(X = 1, failures^{link} = n, failures^{node} = m) = \sum_{s \in K_{mn}} Reachable(s) \cdot Pr(s) \quad (4.21)$$

where $Reachable(s)$ corresponds to I_j of state s . The cumulative probability for reliability of the network when subjected to n link failures and m node failures is defined as

$$Pr(X = 1, failures^{link} \geq n, failures^{node} \geq m) = \sum_{i=N}^n \sum_{j=M}^m Pr(X = 1, failures^{link} = i, failures^{node} = j) \quad (4.22)$$

The metrics for Robustness described in Section 2.2.3 can be computed in the same fashion as discussed in Section 4.2 using the expressions of reliability.

Since, the key focus of FRER is to protect against link losses, in the rest of this thesis we analyze the reliability and robustness characteristics of link losses more deeply. In the next Section, we analyze link losses on an example case study using Symbolic math.

4.4 Use Case Analysis Using Symbolic Math

In this section, the results of implementation of analysis described in section 4.2 for use case topology shown in figure 4.7 is discussed. Symbolic expressions make it easier to identify various components that make the metrics by maintaining the abstraction (symbolic variables instead of numerics). However, this abstraction comes at a higher computation cost. Computing symbolic expressions are more time consuming since they require specialized data structures to store and operate on the symbolic variables. Once the expressions are computed, the post-analysis is a lot faster. Symbolic Math toolbox from Mathworks is used for this implementation [41].

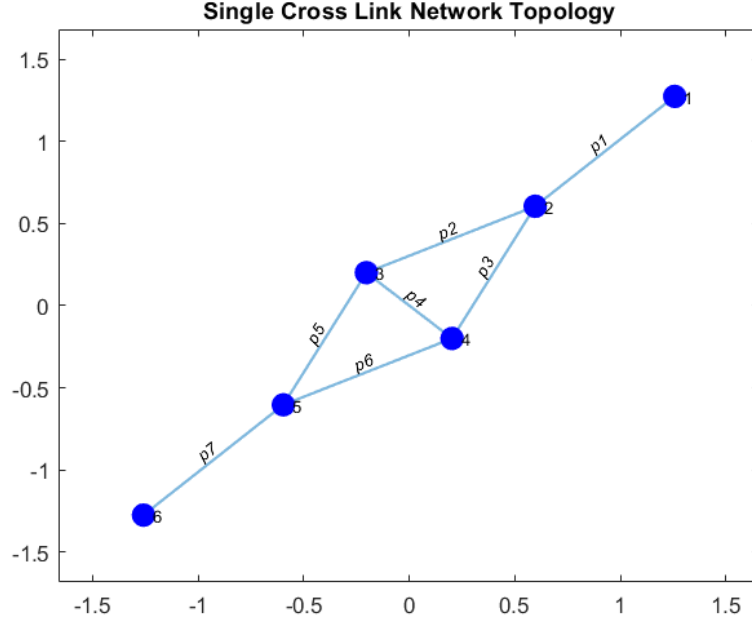


Figure 4.7: Figure showing a sample single cross link network along with its pass probabilities [22]

4.4.1 Overall Reliability

$$\begin{aligned}
 R(p_1, p_2, \dots, p_7) = & p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + \\
 & p_1 p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + \\
 & p_1 p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1) + \\
 & p_1 p_2 p_3 p_4 p_5 p_6 p_7 - p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) - \\
 & p_1 p_2 p_3 p_4 p_5 p_7 (p_6 - 1) - p_1 p_2 p_3 p_4 p_6 p_7 (p_5 - 1) - p_1 p_2 p_3 p_5 p_6 p_7 (p_4 - 1) - \\
 & p_1 p_2 p_4 p_5 p_6 p_7 (p_3 - 1) - p_1 p_3 p_4 p_5 p_6 p_7 (p_2 - 1)
 \end{aligned}$$

Notice, each product component in the above summation expression for reliability is in the form $\prod_{i=1}^N L_i$, where L_i is either equal to p_i or $1 - p_i$. In order to understand what this product form means, let's take the first product component in the above expression, which is

$$p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1)$$

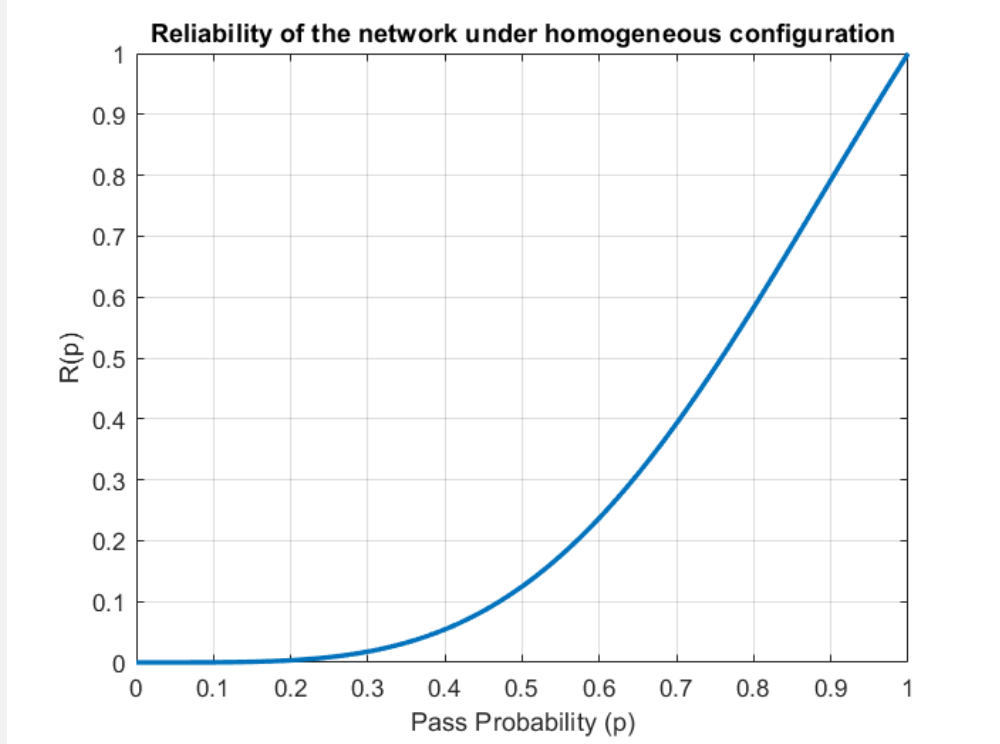
So, for the extreme case where one of the links 4 and 6 fail or both fail (i.e. $p_4 = 0$, $p_6 = 0$), the network is still capable of delivering the packets successfully to the listener. In other words, the expression indicates that the network is protected by a simultaneous fault on links 4 and 6. Notice, there is no product form component containing the terms $(1 - p_1)$ or $(1 - p_7)$. This is due to the fact that these links are connected directly to the talker and listener and no redundant paths are available as can be observed from the network topology. Hence, the absence of the term of the form $(1 - p_i)$ in any of the components indicates that link i is critical to the performance of the network to function reliably.

Homogeneous Links

In case of homogenous links all the links have same pass probability. In other words,

$$p_i = p \quad \forall i \in \{1, 2, \dots, N\}$$

$$R(p) = 8p^5 (p - 1)^2 - 2p^4 (p - 1)^3 + p^7 - 5p^6 (p - 1)$$



4.4.2 N Link failure reliability contribution

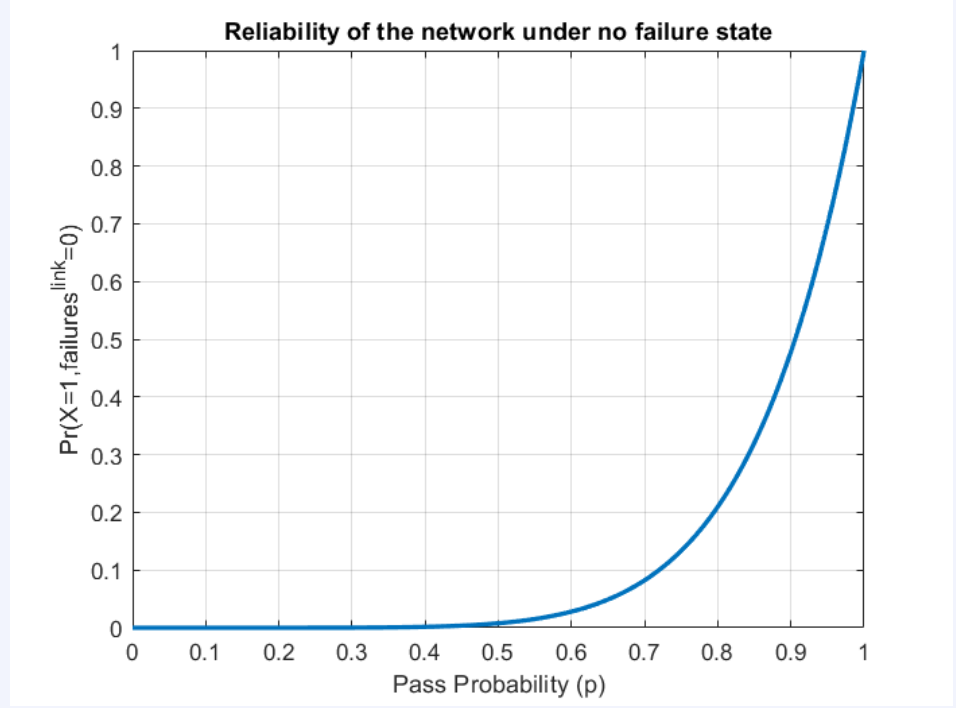
N Link failure reliability provides a decomposed version of the overall reliability, where the expressions provide the specific set of n link failures the network is resistant to. Also notice, the coefficient of the expression under homogeneous configuration indicates the number of cases, the network is resistant to, out of possible $\binom{N}{n}$ possible n link failure cases.

1. Reliability contribution by 0 link failure cases

$$Pr(X = 1, failures^{Link} = 0) = p_1 p_2 p_3 p_4 p_5 p_6 p_7$$

Homogeneous Links

$$p^7$$



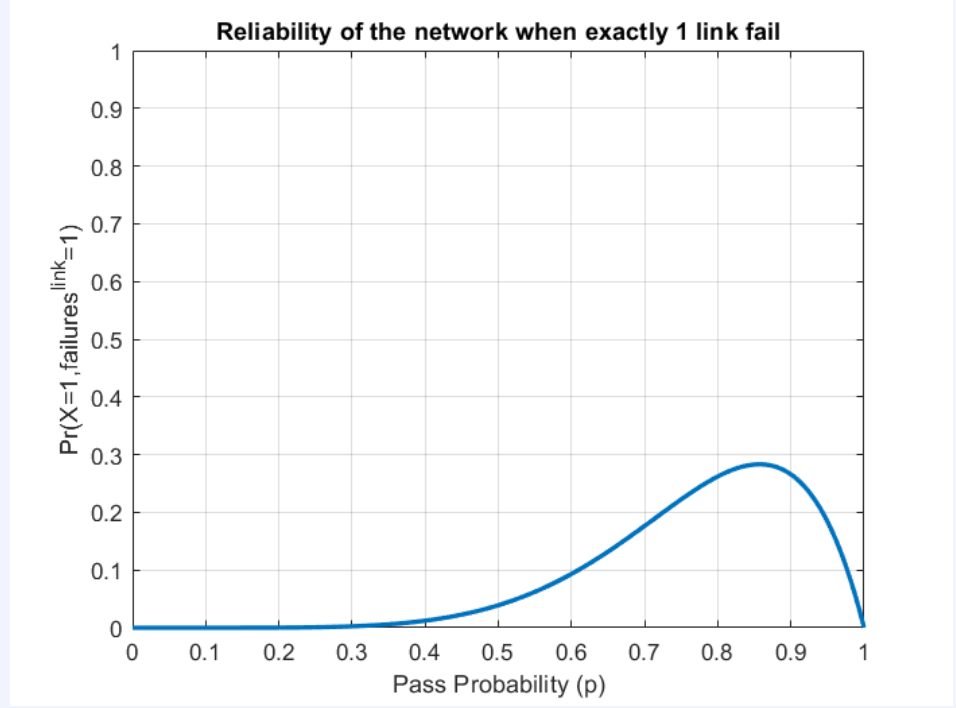
2. Reliability contribution by 1 link failure cases

$$Pr(X = 1, failures^{Link} = 1) = -p_1 p_2 p_3 p_4 p_5 p_7 (p_6 - 1) - p_1 p_2 p_3 p_4 p_6 p_7 (p_5 - 1) - p_1 p_2 p_3 p_5 p_6 p_7 (p_4 - 1) - p_1 p_2 p_4 p_5 p_6 p_7 (p_3 - 1) - p_1 p_3 p_4 p_5 p_6 p_7 (p_2 - 1)$$

Notice, the network is resistant to faults on all the links except on links 1 and 7. This expression, in particular directly provides the critical links, where the redundant paths are not available. For an entirely redundant network, the number of product-form components in the summation will be equal to the number of links.

Homogeneous Links

$$-5p^6(p-1)$$



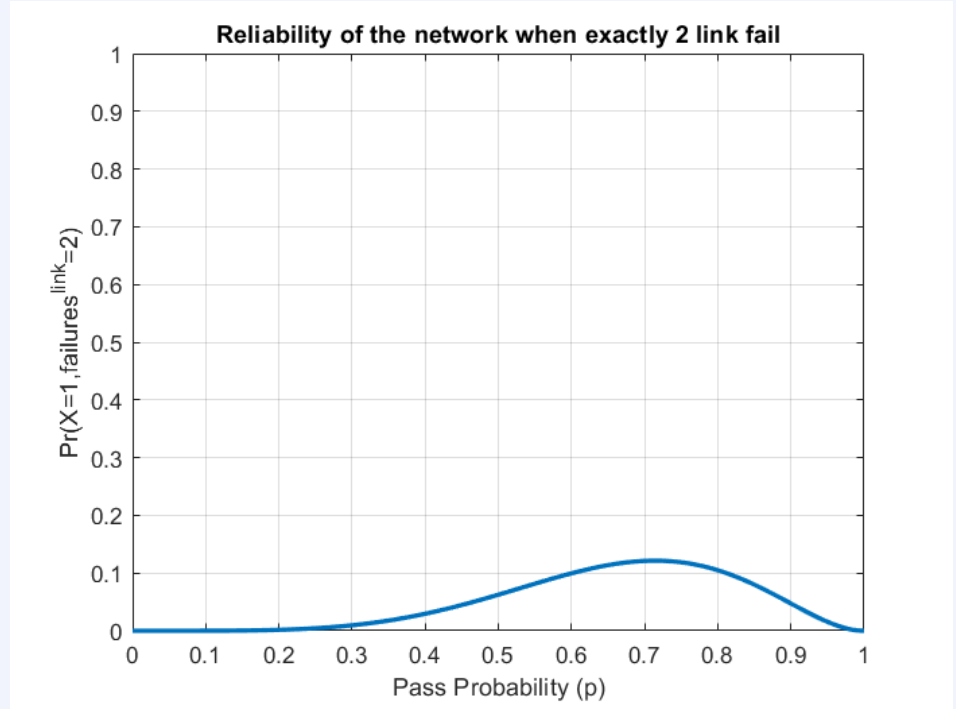
3. Reliability contribution by 2 link failure cases

$$Pr(X = 1, failures^{Link} = 2) = p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + p_1 p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + p_1 p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1)$$

From the expression we can observe that out of 21 possible 2 link failures, the network is resistant to 8 of them. The other interesting note from the above expression is that the components involving the term $(1 - p_4)$ are more (4 out of possible 6). In fact, 4/4 if we consider the combination with links resistant to 1 link failure). Thus, the variations in pass probability of link 4 doesn't affect the reliability of the network by much. This is called robustness or sensitivity of the network. The behavior of this variation is studied in detail in the robustness analysis.

Homogeneous Links

$$8p^5(p-1)^2$$



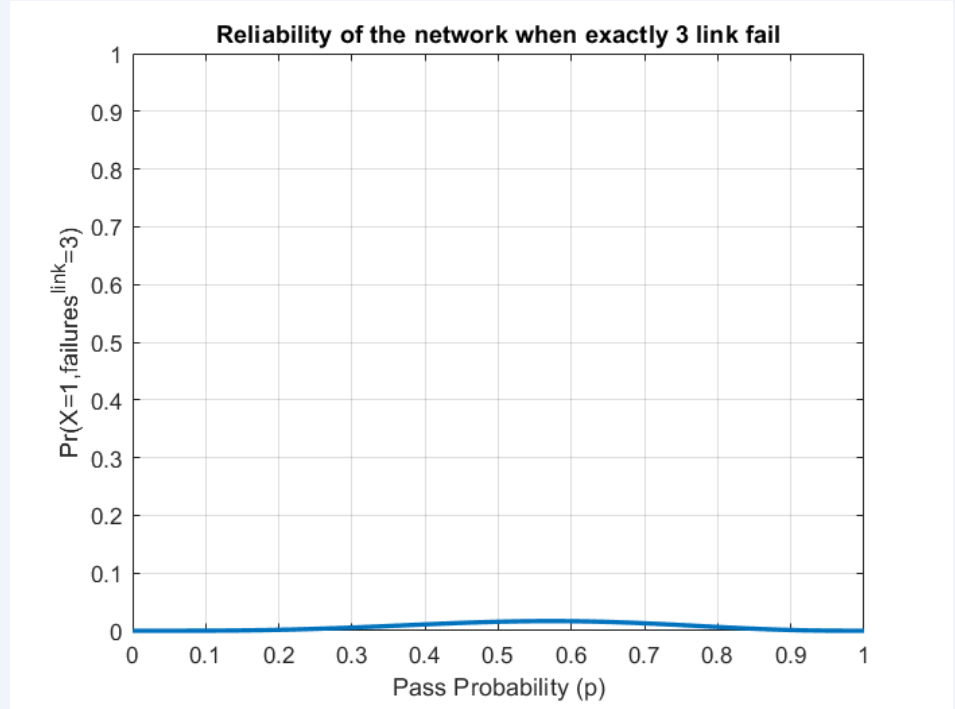
4. Reliability contribution by 3 link failure cases

$$Pr(X = 1, failures^{Link} = 3) = -p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1)$$

The network is only resistant to 2 out of possible $\binom{7}{3}$ 3 link failures.

Homogeneous Links

$$-2 p^4 (p - 1)^3$$



5. Reliability contribution by 4 link failure cases

$$Pr(X = 1, failures^{Link} = 4) = 0$$

6. Reliability contribution by 5 link failure cases

$$Pr(X = 1, failures^{Link} = 5) = 0$$

7. Reliability contribution by 6 link failure cases

$$Pr(X = 1, failures^{Link} = 6) = 0$$

8. Reliability contribution by 7 link failure cases

$$Pr(X = 1, failures^{Link} = 7) = 0$$

Common Observations: It can be noticed, the peak value shifts towards the left. This can be explained from the characteristics of a binomial distribution. As the pass probability of the links reduces, the probability of number of links failing simultaneously increases. Thus there is a correlation between the characteristics of binomial distribution and the peaks. However, the rate at which the peak value drops depends on the topology information. More redundancy, slower is the drop.

Note, the coefficients of expressions related to homogeneous case provide $Pr(X = 1 | failures^{link} = 1)$

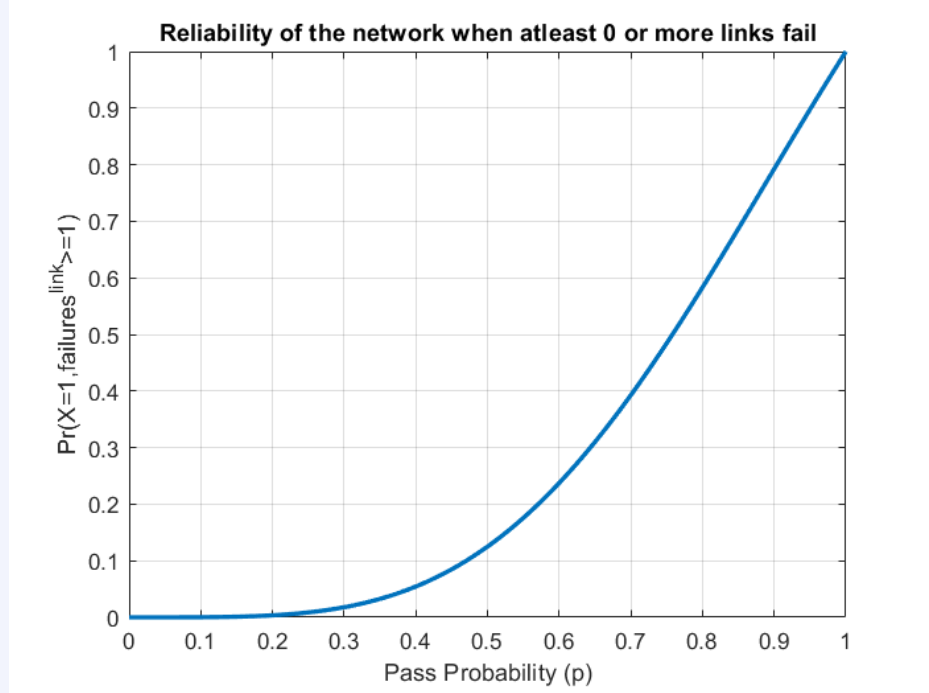
4.4.3 Cumulative N link failure reliability contribution

1. Reliability contribution by 0 or more link failure cases

$$\begin{aligned} Pr(X = 1, failures^{Link} \geq 0) = & p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + \\ & p_1 p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + \\ & p_1 p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1) + \\ & p_1 p_2 p_3 p_4 p_5 p_6 p_7 - p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) - \\ & p_1 p_2 p_3 p_4 p_5 p_7 (p_6 - 1) - p_1 p_2 p_3 p_4 p_6 p_7 (p_5 - 1) - p_1 p_2 p_3 p_5 p_6 p_7 (p_4 - 1) - \\ & p_1 p_2 p_4 p_5 p_6 p_7 (p_3 - 1) - p_1 p_3 p_4 p_5 p_6 p_7 (p_2 - 1) \end{aligned}$$

Homogeneous Links

$$8p^5(p-1)^2 - 2p^4(p-1)^3 + p^7 - 5p^6(p-1)$$

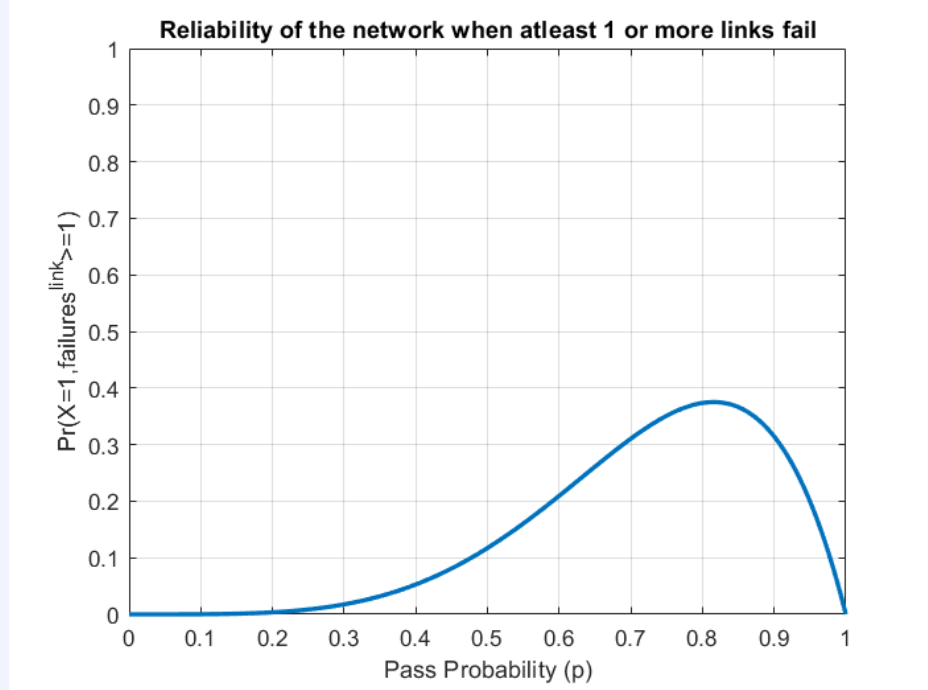


2. Reliability contribution by 1 or more link failure cases

$$\begin{aligned} Pr(X = 1, failures^{Link} \geq 1) = & p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + \\ & p_1 p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + \\ & p_1 p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1) - \\ & p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) - \\ & p_1 p_2 p_3 p_4 p_5 p_7 (p_6 - 1) - p_1 p_2 p_3 p_4 p_6 p_7 (p_5 - 1) - p_1 p_2 p_3 p_5 p_6 p_7 (p_4 - 1) - \\ & p_1 p_2 p_4 p_5 p_6 p_7 (p_3 - 1) - p_1 p_3 p_4 p_5 p_6 p_7 (p_2 - 1) \end{aligned}$$

Homogeneous Links

$$8p^5(p-1)^2 - 2p^4(p-1)^3 - 5p^6(p-1)$$

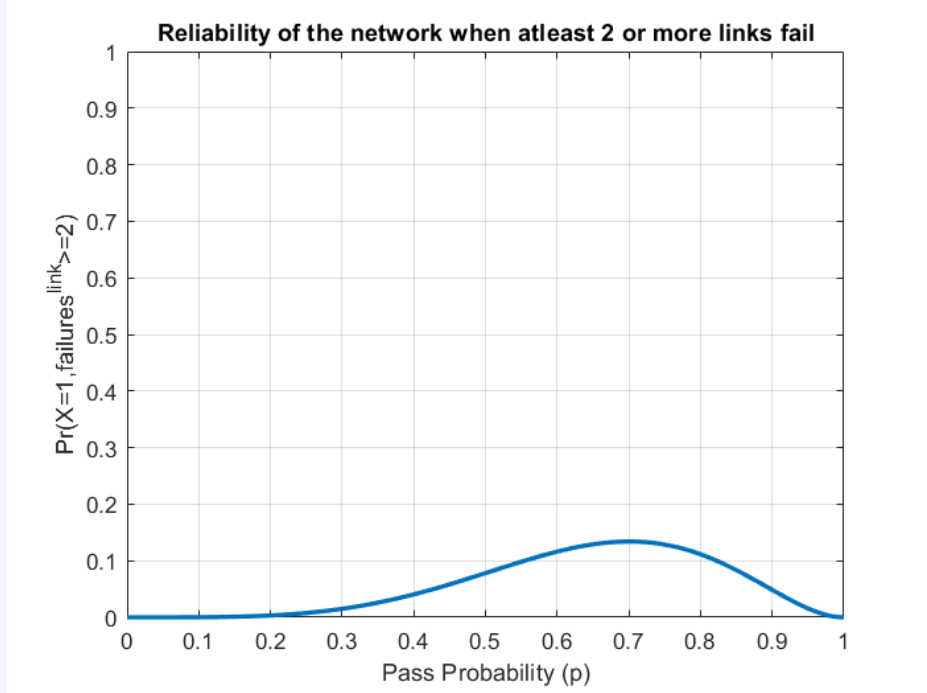


3. Reliability contribution by 2 or more link failure cases

$$\begin{aligned} Pr(X = 1, failures^{Link} \geq 2) = & p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + \\ & p_1 p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + \\ & p_1 p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1) - \\ & p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) \end{aligned}$$

Homogeneous Links

$$8p^5(p-1)^2 - 2p^4(p-1)^3$$

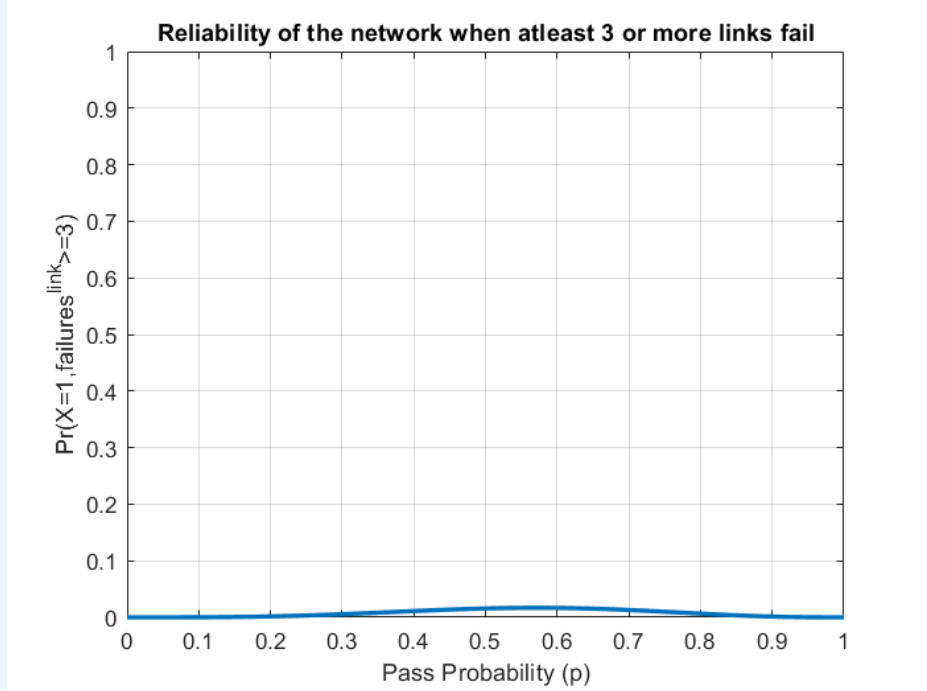


4. Reliability contribution by 3 or more link failure cases

$$Pr(X = 1, failures^{Link} \geq 3) = -p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1)$$

Homogeneous Links

$$-2 p^4 (p - 1)^3$$



5. Reliability contribution by 4 or more link failure cases

$$Pr(X = 1, failures^{Link} \geq 4) = 0$$

6. Reliability contribution by 5 or more link failure cases

$$Pr(X = 1, failures^{Link} \geq 5) = 0$$

7. Reliability contribution by 6 or more link failure cases

$$Pr(X = 1, failures^{Link} \geq 6) = 0$$

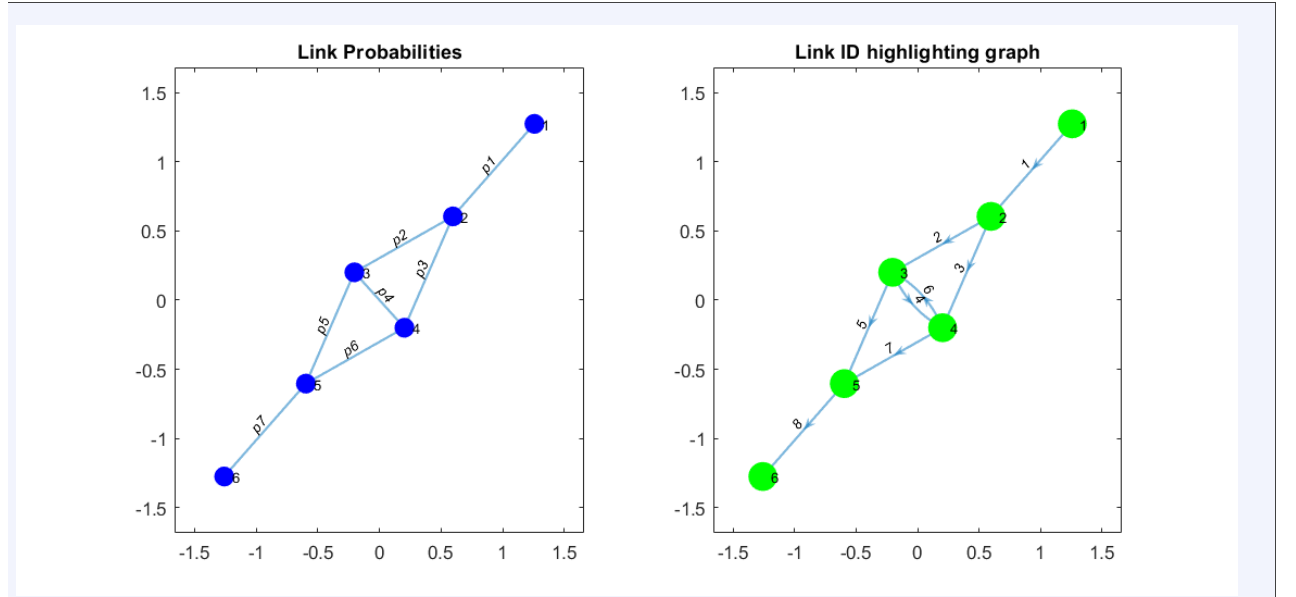
8. Reliability contribution by 7 or more link failure cases

$$Pr(X = 1, failures^{Link} \geq 7) = 0$$

It can be observed when 4 or more links fail, the network cannot deliver the packets successfully. Also notice, the shift of the peak to the left is lesser compared to when exactly N link failure cases contribution is concerned.

4.4.4 NetFlows

As described in the section 4.2.3, net flows gives us the net amount of data flowing out of the link considering all the losses in the network. The analysis for the use case shown in figure 4.7 is as follows.



The right part of the above figure shows the directed graph representation of the use case. It also lists the Link Id's of all the directed links for which the net flows out of them are as follows.

1. **Net Flow out of edge(1,2)**

$$\gamma_1 = p_1$$

In the case of *Link*₁, the losses are only due to the losses of *Link*₁. The data is then split into two streams and passed through links 2 and 3 respectively.

2. **Net Flow out of edge(2,3)**

$$\gamma_2 = p_1 p_2$$

The input to the *Link*₂ is the net flow from *Link*₁. Hence, the net flow coming out of *Link*₂ is both due to the losses in *Link*₁ and *Link*₂. That is $\gamma_2 = \gamma_1 \cdot p_1$. Same is the case with net flow out of *Link*₃.

3. **Net Flow out of edge(2,4)**

$$\gamma_3 = p_1 p_3$$

4. **Net Flow out of edge(3,4)**

$$\gamma_4 = p_1 p_2 p_4$$

Notice, from the above-directed graph representation, the edge(3,4) is a cross-link receiving all the data received by edge(2,3). Hence the net flow out of the link is ($\gamma_4 = \gamma_2 * p_4$). Similar computations apply for net flow out of edge (4,3).

5. **Net Flow out of edge(3,5)**

$$\gamma_5 = p_1 p_2 p_5 (p_3 - 1) (p_4 - 1) + p_1 p_2 p_3 p_4 p_5 - p_1 p_2 p_3 p_5 (p_4 - 1) - p_1 p_2 p_4 p_5 (p_3 - 1) - p_1 p_3 p_4 p_5 (p_2 - 1)$$

In-order to verify the above expression, consider the input into the edge(3,5). It is the merge of the streams $\{(2-3), (2-4-3)\}$ which are split at Node 2. Hence

$$\gamma_5 = p_5 \cdot \text{Merge}(\{(2-3), (2-4-3)\}) \cdot p_1$$

The packets received at node 3 are passed to node 5 if they don't get lost in either of the paths $(2-3)$, $(2-4-3)$. Therefore,

$$\text{Merge}(\{(2-3), (2-4-3)\}) = \text{Passes through either of } (2-3) \text{ or } (2-4-3)$$

$$\text{Merge}(\{(2-3), (2-4-3)\}) = 1 - (\text{Lost in both paths})$$

$$\text{Merge}(\{(2-3), (2-4-3)\}) = 1 - (\text{Lost in } 2-3 \& (\text{Lost in either } (2-4) \text{ or } (4-3) \text{ or both}))$$

$$\text{Merge}(\{(2-3), (2-4-3)\}) = 1 - (\text{Lost in } 2-3) \& (1 - \text{pass through both})$$

$$\text{Merge}(\{(2-3), (2-4-3)\}) = (1 - (1 - p_2) \cdot (1 - p_3 p_4))$$

Solving both of the above expressions give $\gamma_5 = (p_2 + p_3 p_4 - p_2 p_3 p_4) \cdot p_1 p_5$. Similarly all other net flows can be verified.

6. Net Flow out of edge(4,3)

$$\gamma_6 = p_1 p_3 p_4$$

7. Net Flow out of edge(4,5)

$$\gamma_7 = p_1 p_3 p_6 (p_2 - 1) (p_4 - 1) + p_1 p_2 p_3 p_4 p_6 - p_1 p_2 p_3 p_6 (p_4 - 1) - p_1 p_2 p_4 p_6 (p_3 - 1) - p_1 p_3 p_4 p_6 (p_2 - 1)$$

8. Net Flow out of edge(5,6)

$$\begin{aligned} \gamma_8 = & p_1 p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + \\ & p_1 p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + \\ & p_1 p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1) + \\ & p_1 p_2 p_3 p_4 p_5 p_6 p_7 - p_1 p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) - \\ & p_1 p_2 p_3 p_4 p_5 p_7 (p_6 - 1) - p_1 p_2 p_3 p_4 p_6 p_7 (p_5 - 1) - p_1 p_2 p_3 p_5 p_6 p_7 (p_4 - 1) - \\ & p_1 p_2 p_4 p_5 p_6 p_7 (p_3 - 1) - p_1 p_3 p_4 p_5 p_6 p_7 (p_2 - 1) \end{aligned}$$

It can be observed that the net flow coming out of the last link (net data reaching the talker) is equal to the reliability of the network.

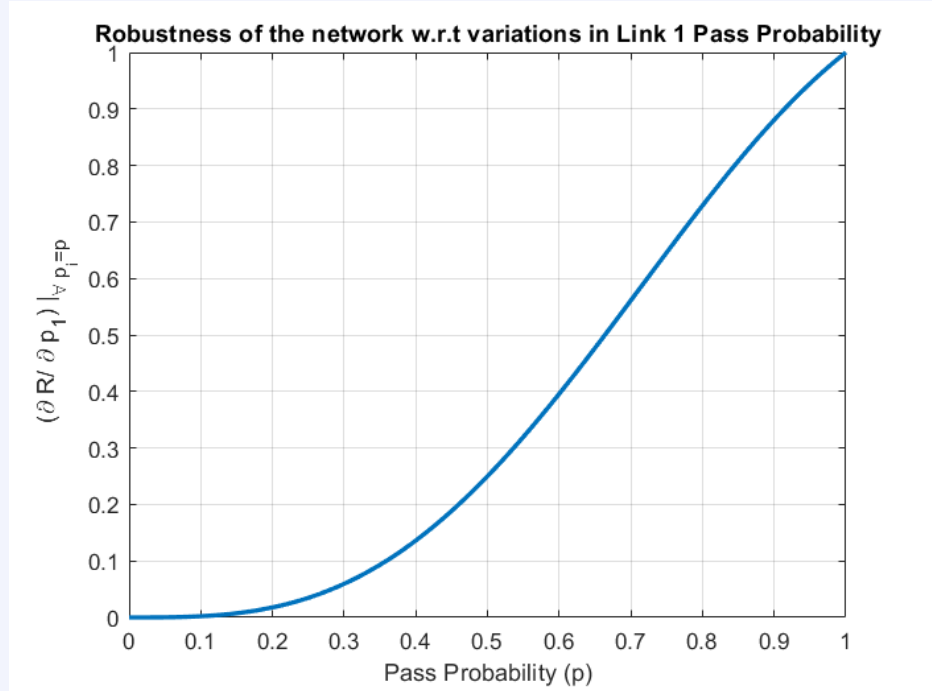
4.4.5 Individual link robustness

1. Robustness of the network w.r.t variations in pass probability of Link 1

$$\begin{aligned} \frac{\partial R}{\partial p_1} = & p_2 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) - p_3 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) - p_2 p_3 p_4 p_5 p_7 (p_6 - 1) - \\ & p_2 p_3 p_4 p_6 p_7 (p_5 - 1) - p_2 p_3 p_5 p_6 p_7 (p_4 - 1) - p_2 p_4 p_5 p_6 p_7 (p_3 - 1) - p_3 p_4 p_5 p_6 p_7 (p_2 - 1) - \\ & p_2 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) + p_2 p_3 p_6 p_7 (p_4 - 1) (p_5 - 1) + p_2 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + \\ & p_2 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_2 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) + p_3 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + \\ & p_3 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_3 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1) + p_2 p_3 p_4 p_5 p_6 p_7 \end{aligned}$$

Homogeneous Links

$$8p^4(p-1)^2 - 2p^3(p-1)^3 + p^6 - 5p^5(p-1)$$



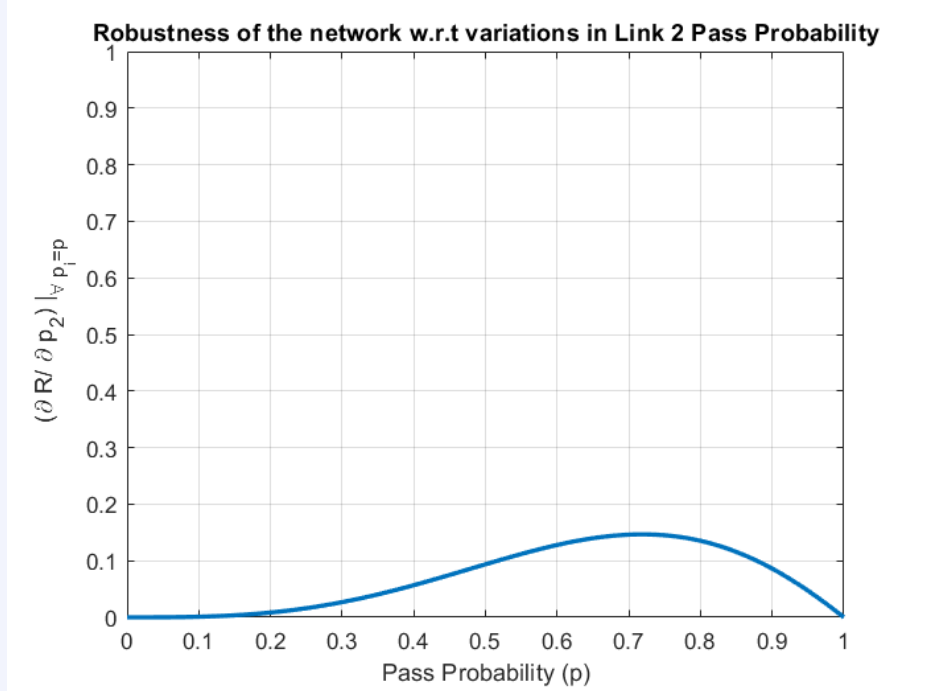
Notice, the robustness is not good at higher values of the pass probability. This can be explained from the position of the link in the network. The link is not protected by redundancy and hence any change in the reliability of this link affects the network reliability more. Convince yourself that Link 7 should also have same characteristics

2. Robustness of the network w.r.t variations in pass probability of Link 2

$$\begin{aligned} \frac{\partial R}{\partial p_2} = & p_1 p_3 p_5 p_7 (p_4 - 1) (p_6 - 1) - p_1 p_4 p_5 p_6 p_7 (p_3 - 1) - p_1 p_5 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) + \\ & p_1 p_4 p_5 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_4 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_5 p_6 p_7 (p_3 - 1) (p_4 - 1) \end{aligned}$$

Homogeneous Links

$$4p^4(p-1)^2 - p^3(p-1)^3 - p^5(p-1)$$



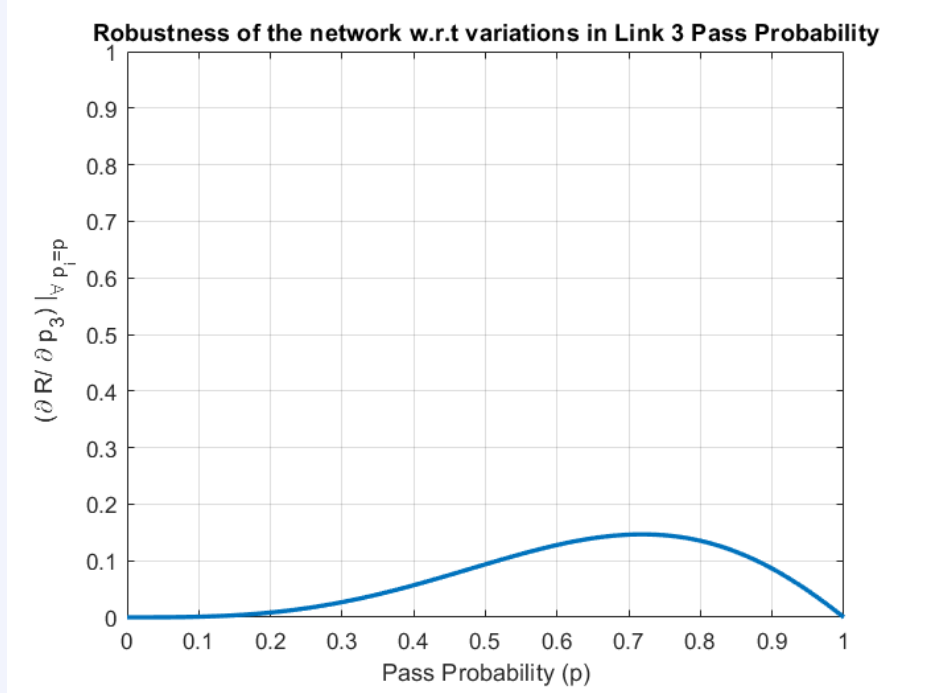
Unlike Link 1, this link has redundancy available and hence at higher values of pass probability, the robustness of the network is high since despite the variations in this link, the network is protected redundancy, hence robustness metric is near zero (highly robust). Convince yourself that links 3,5 and 6 should have same characteristics.

3. Robustness of the network w.r.t variations in pass probability of Link 3

$$\frac{\partial R}{\partial p_3} = p_1 p_2 p_6 p_7 (p_4 - 1) (p_5 - 1) - p_1 p_4 p_5 p_6 p_7 (p_2 - 1) - p_1 p_6 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) + p_1 p_4 p_5 p_7 (p_2 - 1) (p_6 - 1) + p_1 p_4 p_6 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_5 p_6 p_7 (p_2 - 1) (p_4 - 1)$$

Homogeneous Links

$$4p^4(p-1)^2 - p^3(p-1)^3 - p^5(p-1)$$

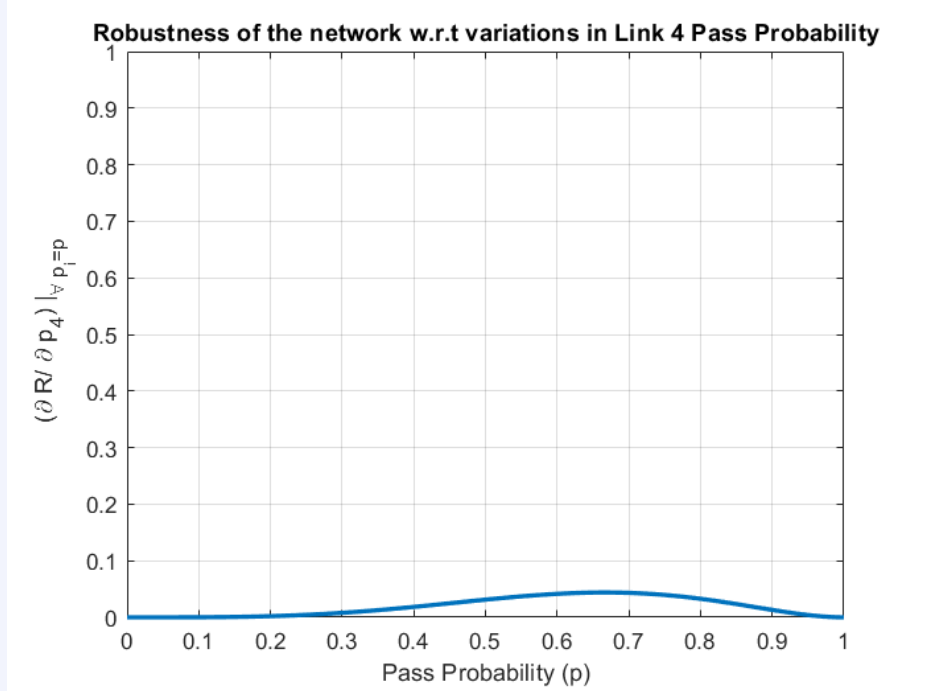


4. Robustness of the network w.r.t variations in pass probability of Link 4

$$\frac{\partial R}{\partial p_4} = p_1 p_2 p_6 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_3 p_5 p_7 (p_2 - 1) (p_6 - 1)$$

Homogeneous Links

$$2 p^4 (p - 1)^2$$



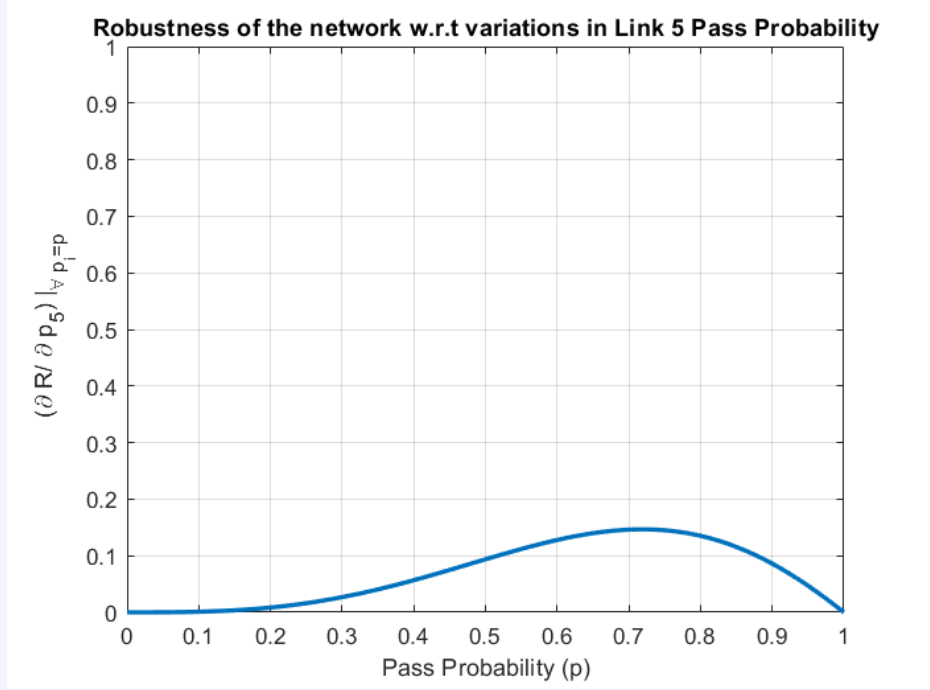
This link is more robust than all the other networks since, there are multiple redundant paths available to the network.

5. Robustness of the network w.r.t variations in pass probability of Link 5

$$\frac{\partial R}{\partial p_5} = p_1 p_2 p_3 p_7 (p_4 - 1) (p_6 - 1) - p_1 p_2 p_3 p_4 p_7 (p_6 - 1) - p_1 p_2 p_7 (p_3 - 1) (p_4 - 1) (p_6 - 1) + p_1 p_2 p_4 p_7 (p_3 - 1) (p_6 - 1) + p_1 p_2 p_6 p_7 (p_3 - 1) (p_4 - 1) + p_1 p_3 p_4 p_7 (p_2 - 1) (p_6 - 1)$$

Homogeneous Links

$$4p^4(p-1)^2 - p^3(p-1)^3 - p^5(p-1)$$

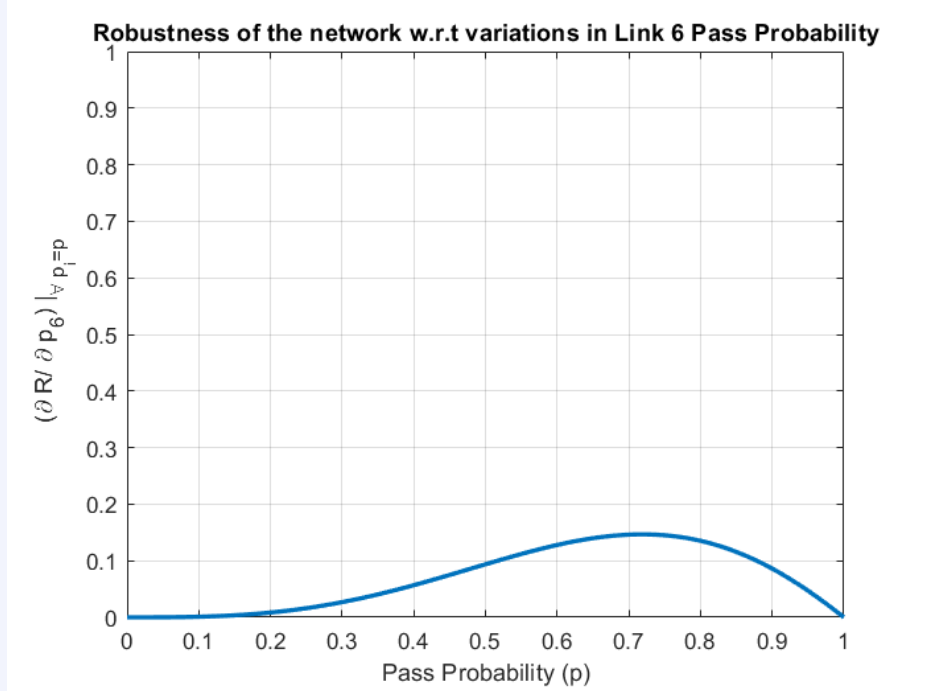


6. Robustness of the network w.r.t variations in pass probability of Link 6

$$\frac{\partial R}{\partial p_6} = p_1 p_2 p_3 p_7 (p_4 - 1) (p_5 - 1) - p_1 p_2 p_3 p_4 p_7 (p_5 - 1) - p_1 p_3 p_7 (p_2 - 1) (p_4 - 1) (p_5 - 1) + p_1 p_2 p_4 p_7 (p_3 - 1) (p_5 - 1) + p_1 p_3 p_4 p_7 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_7 (p_2 - 1) (p_4 - 1)$$

Homogeneous Links

$$4 p^4 (p - 1)^2 - p^3 (p - 1)^3 - p^5 (p - 1)$$

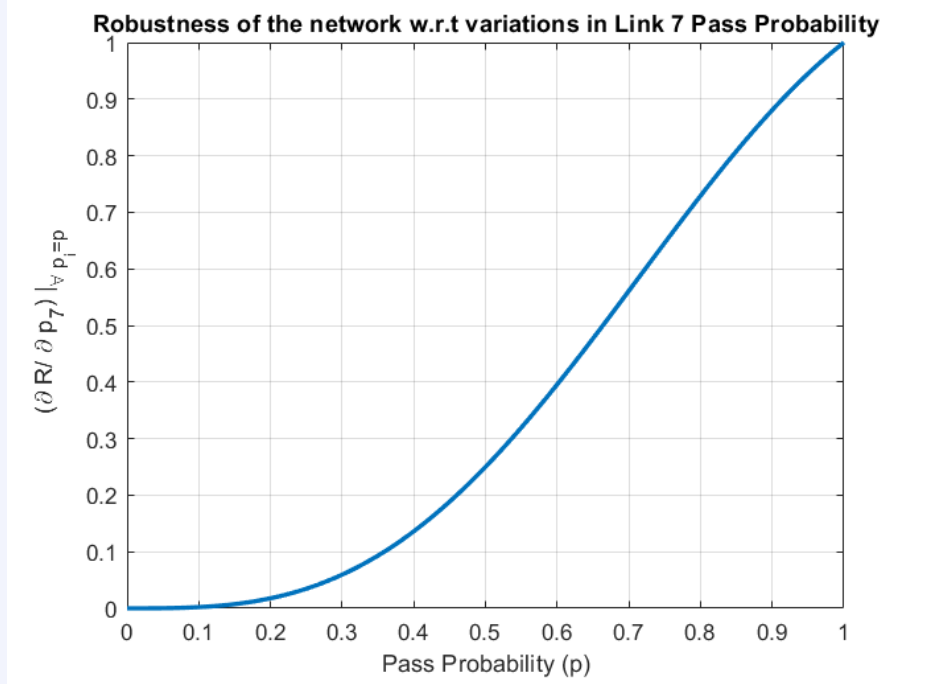


7. Robustness of the network w.r.t variations in pass probability of Link 7

$$\begin{aligned} \frac{\partial R}{\partial p_7} = & p_1 p_2 p_3 p_5 (p_4 - 1) (p_6 - 1) - p_1 p_3 p_6 (p_2 - 1) (p_4 - 1) (p_5 - 1) - p_1 p_2 p_3 p_4 p_5 (p_6 - 1) - \\ & p_1 p_2 p_3 p_4 p_6 (p_5 - 1) - p_1 p_2 p_3 p_5 p_6 (p_4 - 1) - p_1 p_2 p_4 p_5 p_6 (p_3 - 1) - p_1 p_3 p_4 p_5 p_6 (p_2 - 1) - \\ & p_1 p_2 p_5 (p_3 - 1) (p_4 - 1) (p_6 - 1) + p_1 p_2 p_3 p_6 (p_4 - 1) (p_5 - 1) + p_1 p_2 p_4 p_5 (p_3 - 1) (p_6 - 1) + \\ & p_1 p_2 p_4 p_6 (p_3 - 1) (p_5 - 1) + p_1 p_2 p_5 p_6 (p_3 - 1) (p_4 - 1) + p_1 p_3 p_4 p_5 (p_2 - 1) (p_6 - 1) + \\ & p_1 p_3 p_4 p_6 (p_2 - 1) (p_5 - 1) + p_1 p_3 p_5 p_6 (p_2 - 1) (p_4 - 1) + p_1 p_2 p_3 p_4 p_5 p_6 \end{aligned}$$

Homogeneous Links

$$8p^4(p-1)^2 - 2p^3(p-1)^3 + p^6 - 5p^5(p-1)$$



4.5 Summary

In this chapter, analytical models related to packet losses on links and nodes are developed. Pseudo code corresponding to the implementations of analytical models in MATLAB is provided. Example case study analysis using symbolic math is performed for link losses and the significance of the metrics is described.

One can observe from the analysis described in Section, the size of the state-space grows exponentially with increase in the number of nodes and links. This becomes computationally intensive as the size of the network increases. Chapter 7 discusses methods for improving the computation speedup.

Chapter 5

Case studies and Validation of Link Losses

In Chapter 4, we discussed the analysis related to link and node losses. We also saw a quantitative study of the link losses using symbolic math on a sample example network. In this chapter, we dive deeper into different case studies related to link losses for more insights. This chapter also contains validation methodology for validating the analysis developed for link losses in Section 4.2.

The organization of the chapter is as follows: Section 5.1 focuses on comparative study of important example cases. Section 5.2 provides the methodology for validating the analysis described in Section 4.2, and Section 5.3 provides the validation results for an example case based on the methodology described in Section 5.2.

5.1 Case Studies

The objective of case studies is to perform comparison studies between topologies and observe how the metrics of interest vary. With this context, three varieties of case studies are formulated. The description of each type and their purpose is as follows.

1. **Series Parallel Networks:** These networks are the simplest form of networks employing redundancy. These networks demonstrate the impact of redundancy on a fundamental scale.
2. **Ladder Redundancy networks:** The purpose of these networks is to demonstrate how links with the same probabilities affect the network differently based on their position in the topology.
3. **Cross Link Analysis:** This set of use cases helps in understanding the importance of cross-link in ladder redundancy networks.

5.1.1 Series Parallel graphs

The simplest set of use cases that can demonstrate the significance of the metrics are graphs containing components connected in series-parallel configurations. All graphs can be condensed into these graphs.

The series parallel graph can be represented in the following form:

$[n_1, n_2, \dots, n_K]$, where $n_i \in \{0, 1, \dots, M\}$. The network consists of K components in series where each component can contain at-most M parallel links.

Denote by p_{ij} , the pass probability of link j in the i^{th} series component. The overall reliability of the graph can be written as

$$Pr(X = 1) = \prod_{i=1}^K (1 - \prod_{j=1}^{n_i} (1 - p_{ij})) \quad (5.1)$$

Under homogeneous configuration,

$$Pr(X = 1) = R(p) = \prod_{i=1}^K (1 - (1 - p)^{n_i}) \quad (5.2)$$

The mean collective robustness of the network can be computed as

$$\frac{1}{N} \cdot \frac{\Delta R}{\Delta p} = \frac{R(p_1, p_2, \dots, p_N) - R(p_1 - \Delta, p_2 - \Delta, \dots, p_N - \Delta)}{\Delta \cdot N} = \frac{1}{N} \cdot \frac{\partial R}{\partial p} \quad (5.3)$$

for small values of Δ

Series Configurations.

In series only connected networks, the number of parallel links in every component is 1. Hence, substituting $n_i = 1$ in the equation 5.2, we get reliability of the network

$$Pr(X = 1) = R(p) = \prod_{i=1}^K (1 - (1 - p)^1) = p^K \quad (5.4)$$

and mean collective robustness as

$$\frac{1}{N} \cdot \frac{\Delta R}{\Delta p} = \frac{1}{N} \cdot \frac{\partial R}{\partial p} = p^{(K-1)} \quad (5.5)$$

Figure 5.1 shows the effect on reliability and robustness of the network with an increase in the number of series components. We can observe from the reliability graph, as the number of links in the series increases, in order to maintain the same reliability of the network, higher pass probability is needed.

The lower the value of robustness metric, the better it is. From the graph we can observe as the number of links in the series increases, the robustness value at lower values of p is better. This is because at lower values of p , change in p doesn't influence the reliability value of the network too much (Notice, the change in the reliability graph at lower values of p). However, at pass probability near 1, the robustness of all the networks reach 1. This implies that at higher values of p slight variations in the pass probability of the links effect the network largely.

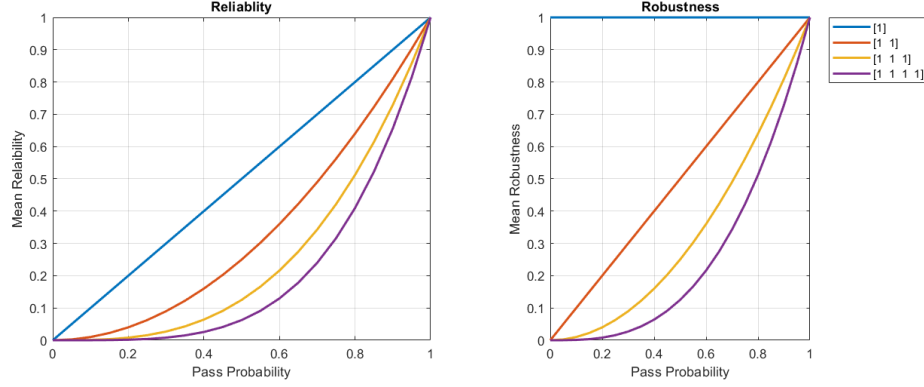


Figure 5.1: Figure showing the reliability and robustness plots for series (only) connected configurations

Parallel Configurations

In parallel only connected networks, the number of series components is equal to 1. Hence, substituting $K = 1$ in the equation 5.2, we get the reliability of the network

$$Pr(X = 1) = R(p) = \prod_{i=1}^1 (1 - (1 - p)^n) = (1 - (1 - p)^n) \quad (5.6)$$

and mean collective robustness as

$$\frac{1}{N} \cdot \frac{\Delta R}{\Delta p} = \frac{1}{N} \cdot \frac{\partial R}{\partial p} = (1 - p)^{(n-1)} \quad (5.7)$$

Figure 5.2 shows the effect on reliability and robustness of the network with an increase in the number of parallel components. We can observe from the reliability graph, as the number of links in parallel increases, in order to maintain the same reliability of the network, lower pass probability is needed.

Contrary to the series configuration, the robustness value at higher values of p is better. This is because, at higher values of p , variations in p doesn't influence the reliability value of the network too much since we have multiple redundant paths. (notice the change in the reliability graph at higher values of p). As p approaches 1, robustness value reaches 0 (compared to 1 in series configuration).

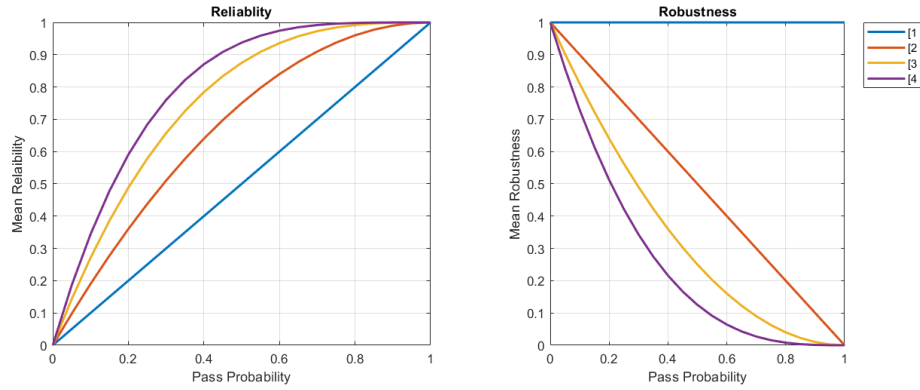


Figure 5.2: Figure showing the reliability and robustness plots for parallel (only) connected configurations

Series Parallel Configurations

Under combined configuration, as we can observe from figure 5.3, the networks have mixed features of both the above configurations. Particularly, the robustness graph starts with zero (like series configuration) and settles at 0. This makes the peak value an interesting observation point. Notice as the number of parallel components increases, the peak shifts towards left. In other words, the parallel components push the peak to become lower and towards left (towards $p = 0$) while the series components influence the peak to become higher and more towards the right (towards $p = 1$).

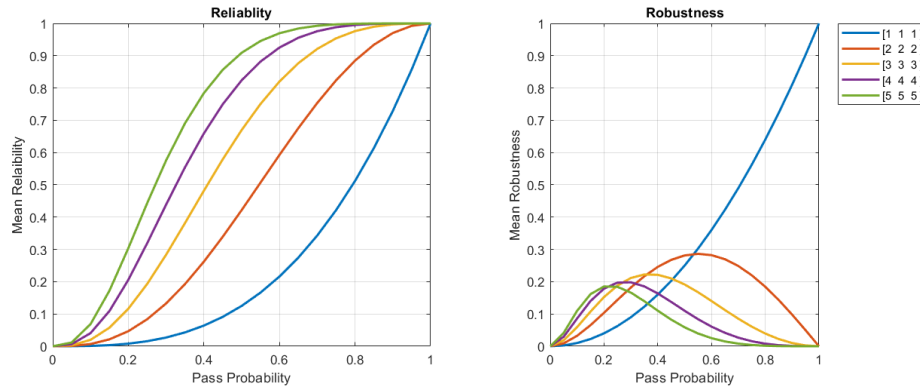


Figure 5.3: Figure showing the reliability and robustness plots for combined (series and parallel) configurations

One other interesting point to take away is that, the robustness is high (close to zero) at low pass probabilities of the links. This seems counterintuitive without the analysis. But from analysis, we can notice this is due to the fact that the reliability at low pass probabilities is low and hence the variations in pass probability doesn't effect the network too much.

5.1.2 Ladder Redundancy Networks

Contents removed for confidentiality reasons

5.1.3 Cross Link Analysis:

Contents removed for confidentiality reasons

5.2 Validation Methodology

In this Section, validation methodology for validating link losses is described. The best validation mechanism involves modeling all the specifications described in the standard and simulating using a discrete event simulator for large amount of time (Some of the commonly used discrete event simulators recommended for future work are OMNeT++[33], SimEvents[36] and Simpy[37]). However, this accurate building of simulation model is complex and the amount of time required to build it is not feasible in the time frame of this project. For this reason an abstract version of simulation is created. In this process, the notion of time is ignored and it is assumed all the packets that are not lost on a link will eventually reach the next link. The decision if a packet will be lost on a link or not is done using a Bernoulli trial on the packet, where the probability of success is equal to the pass probability of the link. This way of validating limits the complexity of modeling but it also meant various preprocessing steps needed for a simulation has to be developed from scratch. The description of the validation process is as follows.

The validation process contains three steps. The first step deals with processing the configuration data and network topology to facilitate the necessary inputs needed for simulation. The simulation process is the part where the actual simulation takes place. In this process, a certain n packets are passed through the network in accordance with the probability configuration. The process is repeated N_{iter} times. Every time, the packets successfully received at the listener is noted and the reliability of the network is computed as the ratio between the number of packets successfully received to the number of packets transmitted. The final step involves the post-processing of the data computed during the simulation process. This step involves performing a bootstrap hypothesis testing to check if the mean reliability obtained from the experiments is equal to the analytical mean. All the steps are implemented in MATLAB and the results corresponding to an example configuration is shown in Section 5.3. The detailed description of each process is as follows.

5.2.1 Configuration and Preparation

At high level validation is divided into N_{iter} experiments. In each experiment n packets are sent. Thus the inputs needed are as follows:

1. Undirected Graph Topology (G)
2. Pass Probabilities of all the links ($P_{Link} = \{p_1, p_2, \dots, p_N\}$)
3. Number of packets per experiment (n)
4. Number of experiments (N_{iter})

Following are the preparation steps needed for simulation.

1. Construct Directed Graph from the undirected topology.

$$G_d = createDirectedGraph(G, P_{Link})$$

The motivation and the pseudo code for this algorithm (Algorithm 3) is already presented in Section 4.2.3.

2. Compute the link probabilities of the directed graph from the link probabilities of the undirected graph.

$$P_{Link}^d = convertToDirected(G_d, P_{Link})$$

This function is about mapping the probabilities of edges of the undirected graph to the probabilities of edges of the directed graph. An illustration of this is presented in Section 4.2.3, where the directed edges $(3,4)$, $(4,3)$ gets a probability of p_4 corresponding to the undirected edge $(3,4)$. Since this operation is trivial no pseudo code is presented.

3. Compute dependency set for each link.

$$D = \text{computeDependency}(e, \text{allPaths})$$

This function lists out the dependencies of edge e on other links. In order to compute the flow out of e , all the flows out of the edges in the dependency set has to be already computed.

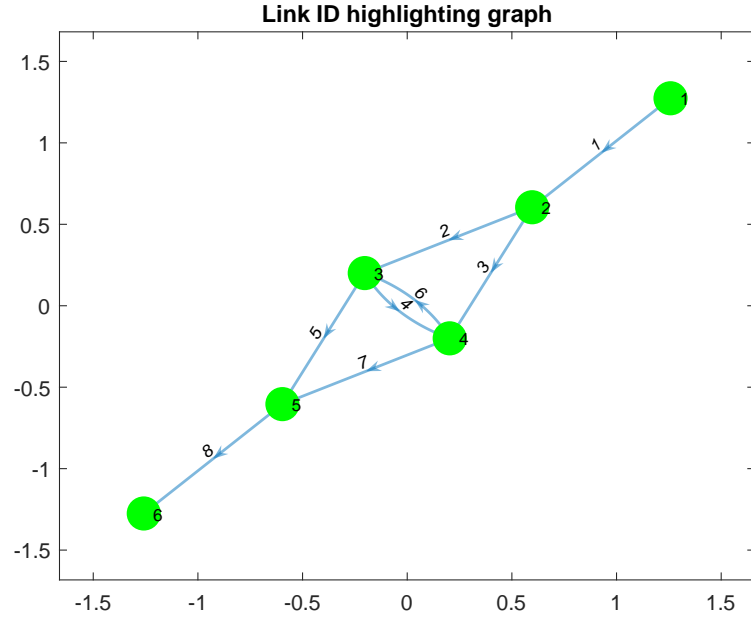


Figure 5.4: A sample directed graph with the directed edge Id's highlighted.

For example, consider the directed graph shown in Figure 5.4 with its directed edge Id's highlighted. For this graph, in order to compute the flow out of the edge $(3,5)$ we need the flow information from $(4,3)$ and $(2,3)$ since both these flows merge at node 3 and produce the flow $(3,5)$. Hence the dependency set for the edge with Id 5 is 2, 6. The pseudo code for this algorithm is shown in Algorithm 9.

4. Compute Link Execution Order

$$\text{Order} = \text{ComputeLinkExecutionOrder}(G_d, \text{allPaths})$$

This function aims at computing the execution order of the edges, such that when the flows are computed in this order, all the dependencies corresponding to edges are already

computed. For example, the execution order for the network shown in Figure 5.4 is 1 – 2 – 3 – 4 – 6 – 7 – 5 – 8. Notice, for computing the flow out of edge Id 5, the flows out of edges 4 and 6 should already be computed. Hence 6 must appear in the order before 5. The pseudo code for this algorithm is shown in Algorithm 8

Algorithm 8 *Order = ComputeLinkExecutionOrder($G_d, allPaths$)*

1: Initialize empty Queue.

$$Q = \emptyset$$

2: **for** every edge $e_i \in G_d.E_d$ **do**

3: $Visited_i = 0$;

4: $ENQUEUE(Q, i)$

5: Compute the dependency of this link on other links.

$$D_i = computeDependency(e_i, allPaths)$$

6: **end for**

7: $p = 0$;

8: **while** $Q \neq \emptyset$ **do**

9: $n = DEQUEUE(Q)$

10: $satisfied = 1$;

11: **for** every $j \in D_n$ **do**

12: **if** $Visited_j \neq 1$ **then**

13: $satisfied = 0$;

14: **end if**

15: **end for**

16: **if** $satisfied == 1$ **then**

17: $Visited_n = 1$;

18: $p = p + 1$;

19: $Order(p) = n$;

20: **else**

21: $ENQUEUE(Q, n)$

22: **end if**

23: **end while**

24: **return** $Order$

Algorithm 9 $D = \text{computeDependency}(e, \text{allPaths})$

- 1: Extract Source and target nodes of this directed edge e as (s, t)
- 2: Extract the subset of paths containing this link.

$$P_e = \{\text{path} \in \text{allPaths} | e \in \text{path}\}$$

- 3: Compute the set of In-neighbor nodes of s as I
 - 4: **for** every $\text{path} \in P_e$ **do**
 - 5: **for** every node n in I **do**
 - 6: **if** $\text{edge}(n, s) \in \text{path}$ **then**
 - 7: Add the edge (n, s) to the set D
 - 8: **end if**
 - 9: **end for**
 - 10: **end for**
 - 11: **return** D
-

5.2.2 Simulation Methodology

The simulation primarily contains the computing net flows out of links through the simulation process. The flow out a link i with pass probability p_i can be simulated by performing a Bernoulli trail on each packet. If the random number generated is less than p_i , then we pass the packet through the link, else we discard it. The net flows out all the links are computed in the execution order shown in the above Subsection. The pseudo code of simulation for a given n packets and N_{iter} iterations is shown in Algorithm 10.

Algorithm 10 *Simulate*($n, N_{iter}, P_{Link}^d, Order$)

```

1: for  $j := 1$  to  $N_{iter}$  do
2:   Create an array packets numbered 1 to  $n$ .

                                      $Source = \{1, 2, \dots, n\}$ 

3:   for  $i := 1$  to  $|Order|$  do
4:     Get set of dependent links for the current link in the order

                                      $e_i = Order(i)$ 

                                      $D_i = computeDependency(e_i)$ 

5:     if  $D_i = \emptyset$  then
6:       Input to the link is directly from talker

                                      $Link_{Input}\{i\} = Source$ 

7:       Pass packets with a probability of  $P_{Link}^d(i)$ .

                                      $Link_{Output}\{i\} = \{packet \in Link_{Input}\{i\} | rand \leq P_{Link}^d(i)\}$ 

8:     else
9:       Compute input by merging the outputs of sources corresponding to this link.

                                      $Link_{Input}\{i\} = \bigcup_{j=1}^{|D_i|} Link_{Output}\{j\}$ 

10:      Pass packets with a probability of  $P_{Link}^d(i)$ .

                                      $Link_{Output}\{i\} = \{packet \in Link_{Input}\{i\} | rand \leq P_{Link}^d(i)\}$ 

11:    end if
12:  end for
13:  Compute interested metrics.
14: end for

```

5.2.3 Comparing Analytical and Experimental Results

In this Subsection, we discuss the methodology for comparing the analytical and experimental (from the simulation performed in the above Subsection) results. This Subsection is divided into three parts. In the first part, we recap on the results obtained for reliability discussed in Section 4.2 and use this results to extrapolate on the characteristics of analytical distribution if n packets are passed. In the second part, we compute the reliability of the network and its distribution from experimental data for the same n packet configuration. In the third part, we discuss a bootstrap hypothesis testing method to validate if the reliabilities computed in both the above parts are equal.

Analytical Results:

Given we have a Bernoulli distribution for all the links. Thus the flow out of a link is Bernoulli with probability p_i . The reliability of the network is also Bernoulli with probability R in accordance with the equation 4.16 described in Section 4.2 ($R = Pr(X = 1)$). Therefore,

$$Link_Output_i \sim Bernoulli(p_i)$$

$$X \sim Bernoulli(R)$$

Now if we repeat the experiment with n packets and normalize it (to represent the reliability between 0 and 1), we get

$$Z \sim Binomial(n, R)/n \quad (5.8)$$

where Z is the expected output distribution if n packets are passed. For large values of n , the binomial distribution approximates normal distribution.

$$Z \sim Normal(\mu, \sigma^2), \mu = R, \sigma = \sqrt{\frac{R \cdot (1 - R)}{n}} \quad (5.9)$$

where μ and σ represent the mean and standard deviation of the normal distribution.

Therefore the resultant distribution is characterized by the value of n , the number of packets that are being sent.

Experimental Results:

As shown in the simulation algorithm (Algorithm 10), we perform the experiment by sending n packets and computing how many packets successfully received, and we repeat the experiment N_{iter} times. This lets us identify the characteristics of the distribution better.

The reliability of each experiment is calculated as

$$R_{exp}^{(i)} = \frac{N_{received}}{N_{sent}} \quad (5.10)$$

Note: The superscript denotes index not exponent.

Denote Z_{exp} the experimental distribution for reliability computed from the $R_{exp}^{(i)}$'s for all $\{i \in 1, 2, \dots, N_{iter}\}$

Denote, the overall mean over all the experiments as R_{exp} . This can be computed by computing the average over all the iterations.

$$R_{exp} = E(Z_{exp}) = \frac{\sum_{i=1}^{N_{iter}} R_{exp}^{(i)}}{N_{iter}} \quad (5.11)$$

For the analytical method to be validated, we should have the analytical mean an experimental mean equal and analytical distribution should match with experimental distribution. However,

since the experimental method is limited by n and N_{iter} , both the results won't be equal, and therefore we need a testing methodology for verification. In the next part, we discuss a methodology called Bootstrap hypothesis testing[6] to test the equality of means of both the distributions (analytical and experimental).

Bootstrap Hypothesis Testing for equal means:

In part 1 and part 2 of this Subsection, we showed expressions for analytical reliability(R) and experimental reliability (R_{exp}). Since the experimental approach is limited by n and N_{iter} , we have to ensure that the mean of the experimental distribution (Z_{exp}) for large values of n and N_{iter} (tending to infinity) is indeed equal to the analytical mean. For this reason, we perform bootstrap hypothesis testing for equal means [6]. The test takes in the experimental values ($R_{exp}^{(i)}$), expected mean (analytical mean R) as inputs and outputs a p-value (p) corresponding to the probability that the actual value is as extreme as the one that is observed. We start by formulating the null and alternate hypothesis as follows.

Null Hypothesis (H_0): The mean of the experimental distribution ($E(Z_{exp})$) is equal to the analytical reliability R .

Alternate Hypothesis ($H_{alternate}$): The mean of the experimental distribution ($E(Z_{exp})$) is not equal to the analytical reliability R .

The methodology for computing the p-value as described in [6] is as follows.

1. Shift the experimentally observed values such that their mean is equal to R .

$$R_{shifted}^{(i)} = R_{exp}^{(i)} - R_{exp} + R \quad (5.12)$$

2. Create bootstrap samples from the shifted values of size N_{iter} and calculate the mean of those samples.

- (a) Choose N_{iter} values randomly from $\{R_{shifted}^{(i)} | i \in \{1, 2, \dots, N_{iter}\}\}$ with repetition.

$$B_{sample} = \{random(\{R_{shifted}^{(i)} | i \in \{1, 2, \dots, N_{iter}\}\}) | j \in \{1, 2, \dots, N_{iter}\}\}$$

- (b) Compute the mean of the above values.

$$R_{boot} = E(B_{sample})$$

3. Repeat the experiment k times. Now we have a new distribution (denote by Z_{boot}) with k samples.

$$data_{boot} = \{R_{boot}^{(i)} | i \in \{1, 2, \dots, k\}\}$$

$$Z_{boot} \sim distribution(data_{boot}) \quad (5.13)$$

4. For this distribution we calculate the significance value p , the probability of achieving the value as extreme as the one observed.

Therefore,

$$p = \begin{cases} \frac{Pr(Z_{boot} < R_{exp})}{Pr(Z_{boot} < R)}, & \text{if } R_{exp} < R, \\ \frac{Pr(Z_{boot} > R_{exp})}{Pr(Z_{boot} > R)}, & \text{otherwise} \end{cases} \quad (5.14)$$

Now that we have computed the p-value, we need to set the significance level.

Significance Level (α): It is the probability of making a wrong decision.

Finally, we make the decision between null and alternate hypothesis as follows. If ($p < \alpha$) reject null hypothesis else accept the null hypothesis. Typically, the values of α are around 5%. Higher the value of p , the higher is the significance that our null hypothesis is true. That is the observed value from our experiments has a mean value of that computed analytically at steady state.

Section 5.3 shows the validation results for an example configuration using this analysis.

5.3 Validation Results

In this Section, validation results of a sample use case shown in figure 5.5 is discussed. The validation is done under the case where all the links have a pass probability of 0.9. The experiment is performed by passing 1000 packets (n) and repeating the experiment 5000 times (N_{iter}).

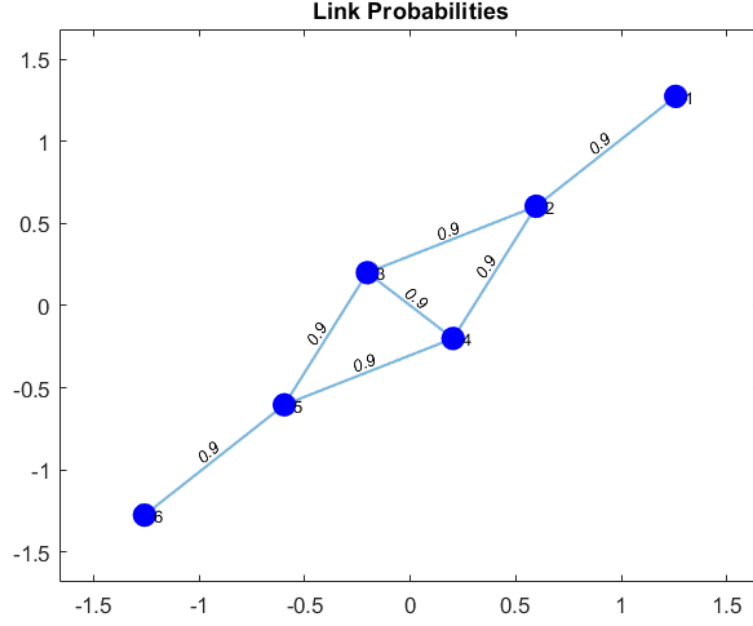


Figure 5.5: A sample validation use case where all links have a pass probability of 0.9

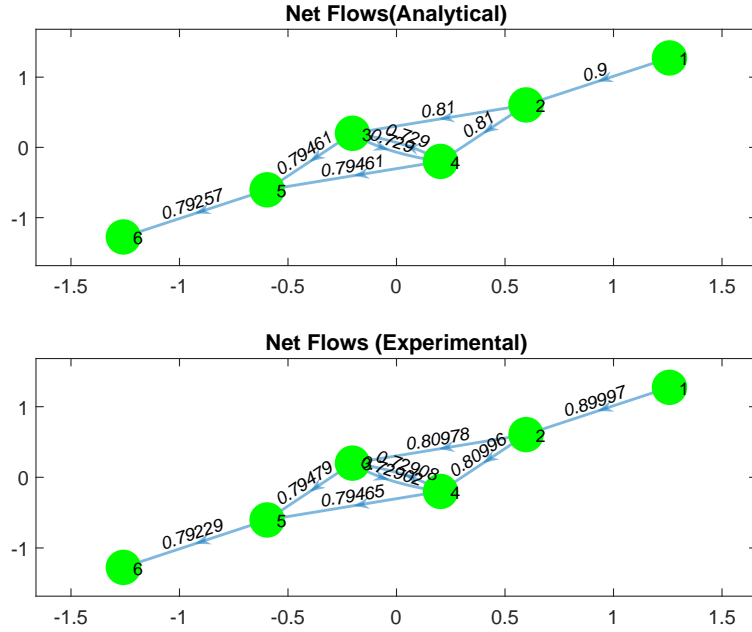


Figure 5.6: Comparison of analytical and experimental net flows for $n = 1000$ and $N_{iter} = 5000$

Figure 5.6 shows the comparison of analytical and experimental net flows out of links. To start with, the reliability of the network 0.79257 (net flow out of the last link) closely matches with the experimental mean. This is good but not enough to validate our analysis. In fact, we are interested in the characteristics of the whole distribution and not just mean. In section 5.2 we proposed, the number of packets received successfully should follow a binomial distribution in

n and R (Analytical mean). Figure 5.7 shows the comparison of expected and analytical distributions ($n = 1000$, $N_{iter} = 5000$). Notice, the distributions totally overlap. The behavior can be more observed in detail for the case $n = 10$ and $n = 1$ (*Bernoulli*) shown in the figures 5.9 and 5.11 respectively. To further strengthen the argument that the experimental mean is indeed equal to the analytical mean bootstrap hypothesis testing described in 5.2 is performed. The test static of interest is the mean of the distribution and the number of bootstrap iterations is chosen 10000. Figures 5.8, 5.10 and 5.12 show the histograms of the shifted bootstrap samples and the region of observing more extreme mean value, for the cases $n = 1000$, $n = 10$ and $n = 1$ respectively. In case where $n = 1$, we observe the analytical mean($Pr(X = 1)$).

The results from the tests are shown in the table 5.1. Notice, the p value for the bootstrap tests in all cases is very high ($p > 5\%$ and observed mean is within the 95% confidence interval bounds in all cases), which confirms our null hypothesis that the mean of the experimental distribution is equal to the analytically computed mean. Notice, the standard deviation and confidence interval range is decreasing as the value of n increases. This can be expected since we are doing more measurements (high simulation time). A similar trend can be expected for increasing N_{iter} .

During this hypothesis testing, only the comparison of the means of the distributions is done. However, the data and methodology can be used to validate any other behavior of interest by formulating the appropriate test statistic. To conclude, in this Section, we successfully validated that our analytical method is correct.

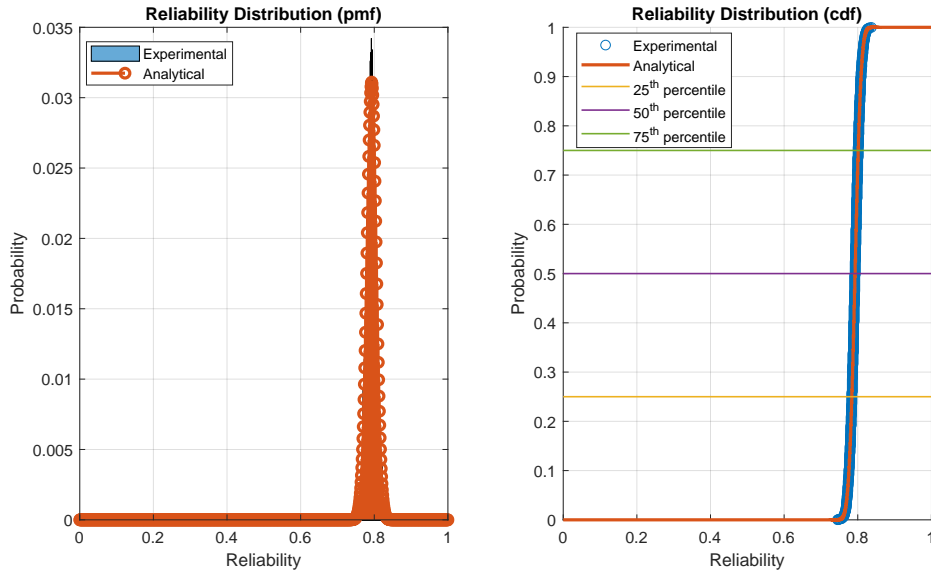


Figure 5.7: Pmf and cdf plots at $n = 1000$ and $N_{iter} = 5000$

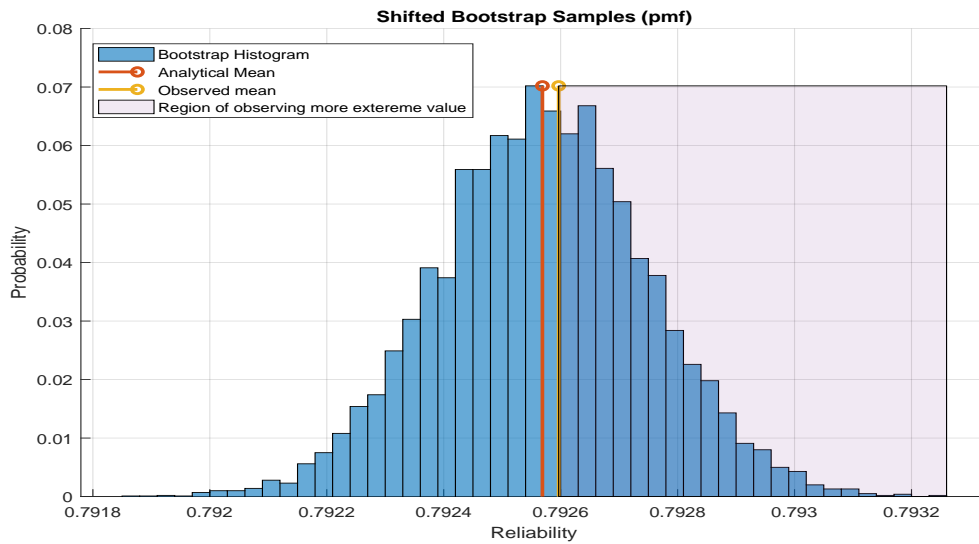


Figure 5.8: Bootstrap Testing for $n = 1000$, $N_{iter} = 5000$, $k = 10000$

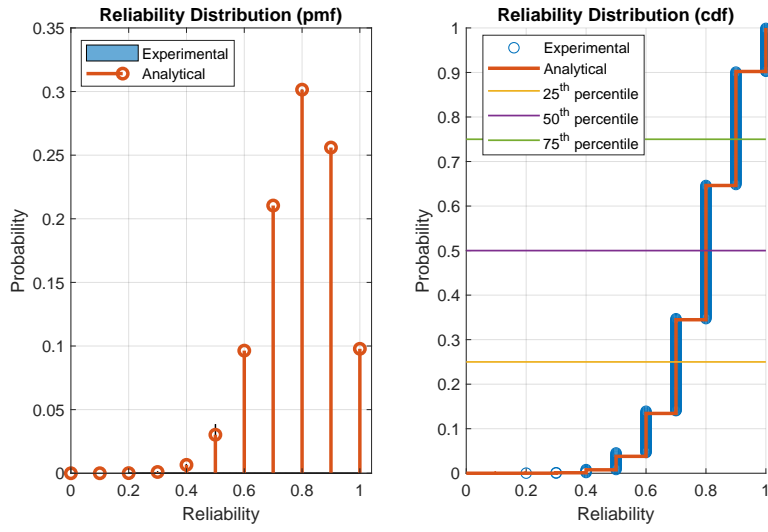


Figure 5.9: Pmf and cdf plots at $n = 10$ and $N_{iter} = 5000$

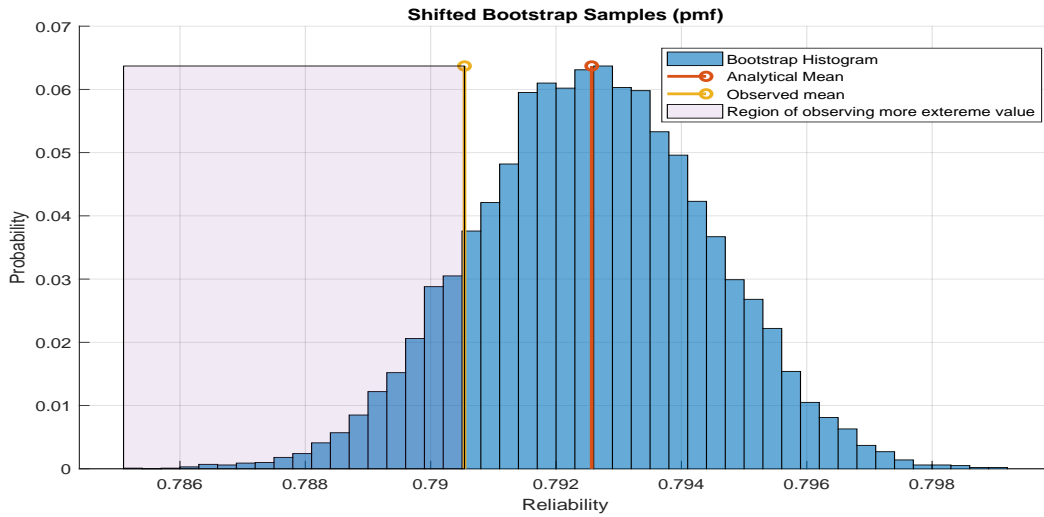


Figure 5.10: Bootstrap Testing for $n = 10$, $N_{iter} = 5000$, $k = 10000$

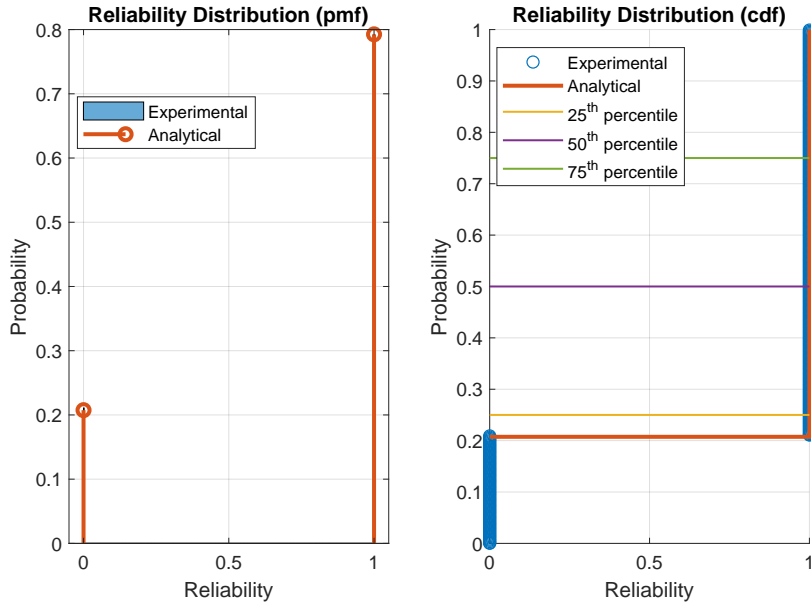


Figure 5.11: Pmf and cdf plots at $n = 1$ and $N_{iter} = 5000$

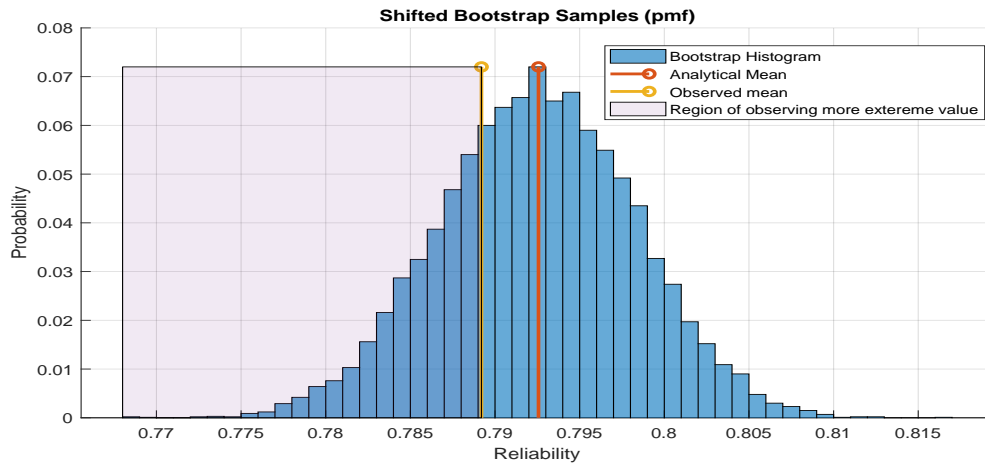


Figure 5.12: Bootstrap Testing for $n = 1$, $N_{iter} = 5000$, $k = 10000$

| | n=1000 | n=10 | n=1 |
|--|---------------|-------------|------------|
| Analytical Mean | 0.79257 | 0.79257 | 0.79257 |
| Observed Mean | 0.7926 | 0.79054 | 0.7892 |
| p | 0.87713 | 0.27652 | 0.57889 |
| 2.5th percentile (shifted bootstrap distribution) | 0.79222 | 0.78899 | 0.78119 |
| 97.5 percentile (shifted bootstrap distribution) | 0.79292 | 0.79617 | 0.80379 |
| Standard deviation (shifted bootstrap distribution) | 0.00017963 | 0.0018566 | 0.0057661 |
| Min Value (shifted bootstrap distribution) | 0.79185 | 0.7851 | 0.768 |
| Max Value (shifted bootstrap distribution) | 0.79326 | 0.7992 | 0.817 |

Table 5.1: Table showing the summary results from the bootstrap hypothesis for different configurations

5.4 Summary

In this Chapter, we discussed different case studies with each serving its purpose. With Series Parallel graph case studies, we observed the effects of redundancy at a fundamental level. With Ladder redundancy networks we noticed how different links based on their position of the network differently. In the cross link analysis case studies we observed how the reliability and robustness of the network get effected with the number of cross links in the network, and discussed the importance of design space exploration. Later in Sections and 5.3, we discussed the validation mechanism for link losses and validated the analysis corresponding to the same successfully. In the next Chapter, we proceed to analyzing the buffer overflow losses caused by insufficient buffer size.

Chapter 6

Buffer Loss Analysis

Contents removed for confidentiality reasons

Chapter 7

Computational Speed up

In Chapter 4, we noticed the network with both link and node losses has 2^{M+N} states (M nodes, N links). This exponential increase in the number of states creates a bottleneck (computation time and memory required) in computing the interested metrics. In this Chapter, an approximation algorithm for reducing computation time by trading off accuracy is discussed.

The organization of the chapter is as follows.

In Section 7.1, we compute the worst case computation time of the algorithm and discuss methods for improving the speed. Section 7.2 deals with improving the speed by doing parallel processing. Section 7.3 deals with techniques by computing the metrics for a reduced state-space, thereby trading accuracy for computation time.

7.1 Computation Time complexity of the algorithm

Since we are interested in the worst case time complexity, we have to first identify the component responsible for it. The reachability algorithm is shown in algorithm 7 loops through all the 2^{M+N} states and for each state the reachability of the graph takes $O(|V| + |E|)$. Hence, the effective time complexity of the algorithm is

$$T_{worst} = O((|V| + |E|) \cdot 2^{|V|+|E|}) \quad (7.1)$$

where T_{worst} is the worst case computation time in $|V|$ and $|E|$. From the above expression we can clearly identify, the bottleneck is the number of states.

The reduction of computation time can be done in two ways.

1. Identifying parallel components and using Graphical Processing Units(GPU).
2. Using approximation algorithm which operates only on a subspace for computing metrics and thereby compromising on accuracy.

The methodology for speedup using GPU's is discussed in Section 7.2, and the methodology for speedup using an approximation algorithm is discussed in Section 7.3.

7.2 Parallel components and Speed up using GPU's


In this Section, a methodology for reducing computation time using parallel processing is discussed. The network consists of $2^{|V|+|E|}$ states and all the computations needed from each state are independent of another. For this reason, parallel processing can be used on all the states. GPU's

are most commonly used for parallel processing. Currently, simultaneously maximum of 65535 threads can be run. Exceeding this limit would mean, the computations have to be done in batches. In other words,

$$T_{GPU} = \frac{T_{worst}}{65535} \quad (7.2)$$

While this approach is not entirely useful for very large networks, it is sufficient enough for the processing of TSN related analysis, where the maximum hops are generally around 7.

7.2.1 Results

The table 7.1 shows the comparison of Central Processing Unit(CPU) and Graphical Processing Unit(GPU) times for computing the probabilities of each state for the entire state space. The configurations correspond to networks shown in figures  analyzed in Chapter 4. Observe, the improvement factor saturated for state space more than 65535 states. More cross-link analysis could not be done due to memory overflow caused by the size of state-space. In Section 7.3, an approximation algorithm is discussed, which will prove useful for very large networks. Figure 7.1 shows the info of GPU device used.

```
CUDADevice with properties:
    Name: 'Quadro M1000M'
    Index: 1
    ComputeCapability: '5.0'
    SupportsDouble: 1
    DriverVersion: 9
    ToolkitVersion: 9
    MaxThreadsPerBlock: 1024
    MaxShmemPerBlock: 49152
    MaxThreadBlockSize: [1024 1024 64]
    MaxGridSize: [2.1475e+09 65535 65535]
    SIMDWidth: 32
    TotalMemory: 4.2950e+09
    AvailableMemory: 3.5051e+09
    MultiprocessorCount: 4
    ClockRateKHz: 1071500
    ComputeMode: 'Default'
    GPUOverlapsTransfers: 1
    KernelExecutionTimeout: 1
    CanMapHostMemory: 1
    DeviceSupported: 1
    DeviceSelected: 1
```

Figure 7.1: Configuration of GPU used

| <i>N_edges</i> | <i>N_nodes</i> | <i>Statespace Size</i> | <i>CPU Time (seconds)</i> | <i>GPU Time (seconds)</i> | <i>Improvement Factor</i> |
|----------------|----------------|------------------------|---------------------------|---------------------------|---------------------------|
| 7 | 6 | 8192 | 0.0221 | 0.002554 | 8.6531 |
| 10 | 8 | 262144 | 0.794032 | 0.047802 | 16.6109 |
| 13 | 10 | 8388608 | 29.622051 | 1.872038 | 15.8234 |

Table 7.1: Comparison of CPU and GPU time for computing the probabilities of every state in the state space

7.3 Approximation algorithms

In the previous chapters, an approach for exact computations is presented. Performing, exact computations are critical in-order to compare different configurations accurately. The problem however with computing the exact metrics is the amount of computation time required to compute those metrics. Computing exact metrics is feasible for smaller networks and also recommended since the number of end to end hops in a TSN network is not high. However, for faster comparisons, an approximate approach for computing the metrics is provided in the following subsequent sections. The metrics computed by this approximation algorithms returns bounds instead of exact values, where the following relation holds.

$$LB \leq OPT \leq \min\{UB, 1\} \quad (7.3)$$

where LB represents the lower bound returned by the algorithm, UB represent the upper bound returned by the algorithm and OPT represent the exact value of the metric.

Throughout the rest of the chapter, for simplicity reasons, we perform the analysis for only link failures, and it is assumed all the links have equal pass probability of p (this need not be the case, it just makes the explanation of analysis easier). Also, the time complexity of the shortest path algorithm is ignored since our primary interest is to reduce the state space. For this reasons for the rest of the section, we treat (focus) the computation time as a function of the size of the state space ($O(2^{|E|}) = O(2^N)$, N representing the number of links in the network).

7.3.1 Approximate Reliability Calculation

In Section 4.2.1, the reliability of the network corresponding to link losses is calculated as:

$$Reliability = Pr(X = 1) = \sum_{j=1}^{2^N} I_j \cdot Pr(S_j) \quad (7.4)$$

Notice, in the above equation we are adding the summations of all the possible 2^N states resulting in the time complexity of $O(2^N)$. Therefore for reducing the computation time, we should reduce the number of summations in the above equation. However, now, since we are only considering a sub-space of the total state-space, the net computation is an approximation and the objective of our algorithm should be that the net value of the subspace is as close to the exact value. Before deciding on the approach on how to maximize the output, let us understand some key properties of the above equation. The equation is a product of two terms. In order to have a good approximation, we have to choose a subspace which maximizes the product of the two terms. The first term I_j depends on the topology of the network and cannot be determined without knowing the topology of the network. However, the second term doesn't depend on the topology information and solely depends on the total link count.

Since, we assumed all the links of equal pass probabilities, the probability of k link failures follows a binomial distribution.

$$Pr(failures^{link} = k) = \frac{N!}{(k!) \cdot (N - k)!} \cdot (p^{N-k}) \cdot (1 - p)^k \quad (7.5)$$

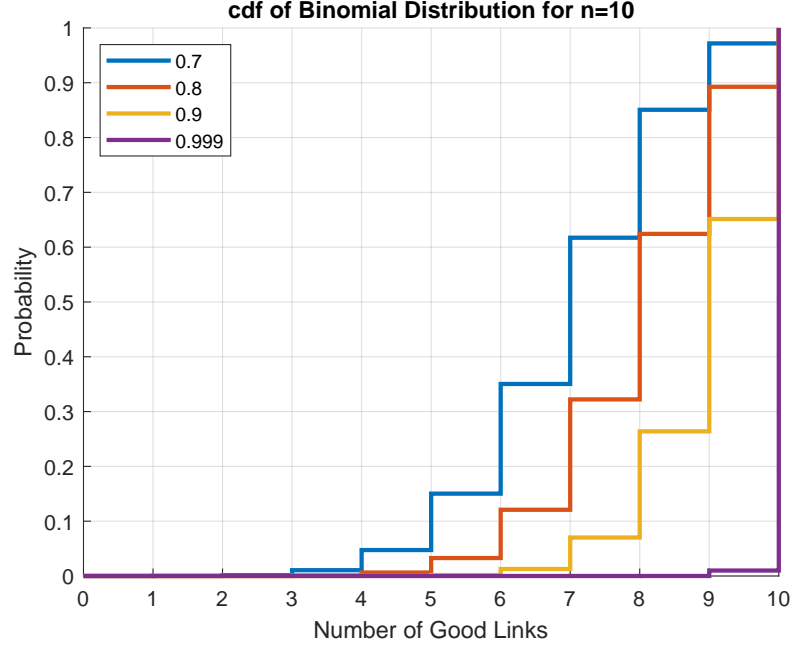


Figure 7.2: Cumulative Density Function (cdf) plot of binomial distribution for $N = 10$

Figure 7.2 shows the Cumulative Density Function (cdf) of binomial distribution of a 10 link network for different pass probabilities of links.

Notice, as the value of p increases, the probability of the number of bad links decreases. Since, the pass probability of the links is generally high, from the characteristic of above distribution we can minimize the error by choosing the subspace in the order of no link failures to k link failures, where when k becomes N would result in an exact solution.

Denote by Z_k , the probability of a packet getting successfully delivered given k link failures.

$$Z_k = Pr(X = 1 | failures^{link} = k) \quad (7.6)$$

Therefore, the equation for reliability can be written as follows (assuming network definitely fails when all links fail):

$$Reliability = \sum_{k=0}^{N-1} Z_k \cdot Pr(failures^{link} = k) \quad (7.7)$$

Now, let us define our approximation algorithm as $AIG(G, k)$ which computes the reliability of subspace containing all the states from 0 link failures up-to k link failures. Therefore, the worst case time complexity of the algorithm is

$$t = O\left(\sum_{j=0}^k \frac{N!}{(k!) \cdot (N-k)!}\right) \quad (7.8)$$

When, $k \rightarrow N$, $t = O(2^N)$, and when $k \ll N$ we get $t = O(N^k)$.

Thus by the trading of accuracy, we can compute an approximate value in polynomial time.

7.3.2 Lower and Upper Bounds

Since we do not know the network topology beforehand, it is difficult to compute the output of a state S_j beforehand. However, we know from the general network characteristics, that the mean reliability of a k link failure network will be greater than or equal to $k + 1$ link failure network. In other words we have (assuming $Z_0 = 1$ and $Z_N = 0$),

$$1 \geq Z_1 \geq Z_2 \geq \dots Z_{N-1} \geq 0$$

Assuming, all the subsequent conditional probabilities are same as Z_k , we get the worst case error in Reliability calculation is

$$Error(k, N) = \sum_{j=k+1}^{N-1} Z_k \cdot Pr(failures^{link} = k) \quad (7.9)$$

The worst case error from the above equation happens for $Z_k = 1$ which is equal to the area covered by the non computed portion of the binomial distribution.

Thus the LB and UB returned by the algorithm $Alg(G, k)$ are

$$LB(k) = \sum_{j=0}^k Z_k \cdot Pr(failures^{link} = k) \quad (7.10)$$

$$UB(k) = \min\{LB(k) + \sum_{j=k+1}^{N-1} Z_k \cdot Pr(failures^{link} = k), 1\} \quad (7.11)$$

The upper bound in the above equation is loose. This is because we are assuming all the un-computed mean reliabilities is equal to the mean reliability of the last computed case. This can be improved a lot for $Z_k < 1$. For example consider the case of $N = 6$ and $k = 1$. Let $Z_1 = 5/6$. That is there is one case of 1 link failure which results in the network failure. Now by the above calculations, we assumed that the maximum value of Z_2 is also $5/6$. However, from the analysis of 1 link failures, we know that there is a link (say L_n), failure of which result in the network failure. Therefore out of the 15 $\left(\frac{N!}{(2!) \cdot (N-2)!}\right)$ two link failure cases possible, we know for sure that there are 5 $(N - 1)$ cases where the link L_k is failed. Thus the maximum value of Z_2 possible is $10/15 < 5/6$. Proceeding in a similar fashion, the bounds can be improved by analyzing all the computations of S_j computed until Z_k . Tighter bounds will be derived in course of time (not high priority, since this is only useful for large networks which is not the case with TSN.)

7.3.3 Results

In this Section, we show the impact on computation time for a given level K described in the above section. Figure 7.3, shows a network with 4 cross-links. As can be observed it has 16 edges and hence contains a state space of 2^{16} . Figure 7.4 shows the plot for computation time taken for a chosen K level.

As we can observe, the approximate algorithm takes very less time for low values of K and approaches exact computation time (35.4386 seconds) as K becomes close to 16. The correlation between the computation time taken by the approximation algorithm and the cumulative sum of binomial coefficients $\binom{16}{K}$ can be observed in the table 7.2.

The analysis can be easily extended to including node losses (state-space of 2^{M+N}) as well. Also, the analysis is valid for cases where the probabilities of the components are not equal. However, care should be taken that the approximation level(K) is sufficient enough (particularly for components with low pass probability) to avoid too much error in the approximation.

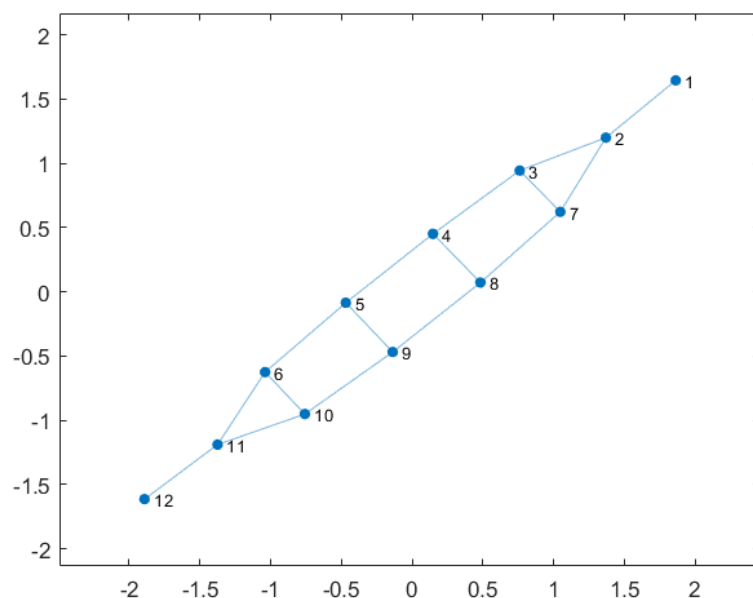


Figure 7.3: Four Cross Link Network with 16 edges.

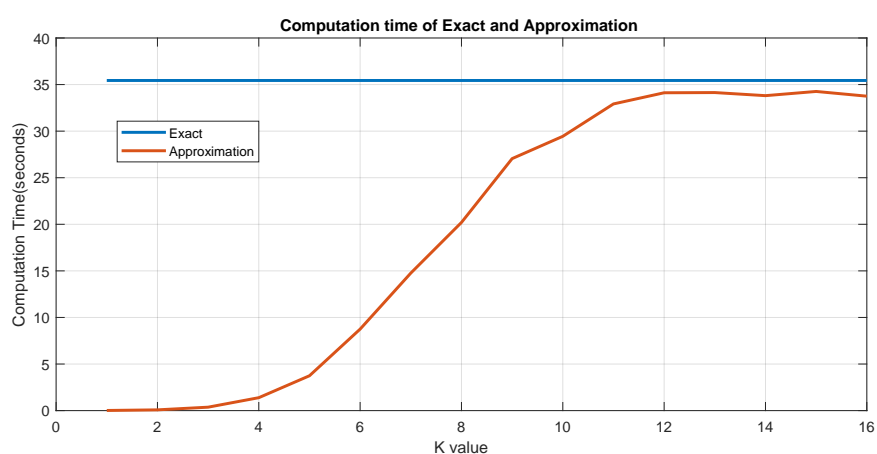


Figure 7.4: Figure showing the plot for computation time for a given level K (computation time in seconds)

| <i>K</i> | <i>16 Choose K</i> | <i>Cumlattive sum</i> | <i>Approximation Alg (sec)</i> |
|----------|--------------------|-----------------------|--------------------------------|
| 0 | 1 | 1 | 0.018 |
| 1 | 16 | 17 | 0.023520594 |
| 2 | 120 | 137 | 0.081003916 |
| 3 | 560 | 697 | 0.368423402 |
| 4 | 1820 | 2517 | 1.387301837 |
| 5 | 4368 | 6885 | 3.740172042 |
| 6 | 8008 | 14893 | 8.748328888 |
| 7 | 11440 | 26333 | 14.76205611 |
| 8 | 12870 | 39203 | 20.19826505 |
| 9 | 11440 | 50643 | 27.05100053 |
| 10 | 8008 | 58651 | 29.44429704 |
| 11 | 4368 | 63019 | 32.92065692 |
| 12 | 1820 | 64839 | 34.11945412 |
| 13 | 560 | 65399 | 34.14065433 |
| 14 | 120 | 65519 | 33.80882553 |
| 15 | 16 | 65535 | 34.26035734 |
| 16 | 1 | 65536 | 33.75328297 |

Table 7.2: Table illustrating correlation between cumulative sum of $\binom{16}{K}$ and computation time took by the approximation algorithm

7.4 Summary

In this chapter to overcome the exponential running time problem of analytical models (for computing metrics), two types of speed up's are proposed. The speedup from GPU's saturate at $\sim 16X$ due to limitations in the maximum number of threads, and the speedup using approximation algorithms is configurable (with parameter K) and the user can trade-off computation time for accuracy. It is shown in Table 7.2 that the approximate computation time is strongly correlated with cumulative sum of binomial coefficients $\binom{N}{K}$

Chapter 8

Design Space Exploration

Contents removed for confidentiality reasons.

Chapter 9

Conclusions and Future Work

This chapter presents the conclusions and future work related to this project. Section 9.1 provides the summary and conclusions of work done in this project, and Section 9.2 provides the future work that can be improved and extended from this project.

9.1 Conclusions

As mentioned in the problem statement (Section 1.4), the objective of this project is to quantify and analyze the reliability, robustness, and cost of the network implementing FRER. Modeling the network topology and various loss components associated with it is a crucial first step in achieving this objective. This along with quantifiable metrics for the reliability, robustness, and cost was formulated with appropriate abstractions (Chapter 2).

The rest of the conclusions are broadly divided into two parts. (1) Conclusions related to building analytical models and their application in case studies. (2) Conclusions related to the implementation of these analytical models and their usage for building optimal designs.

The conclusions related to building analytical models and their application in case studies are as follows:

- **Link and node loss analysis:** Since the primary focus of FRER is to increase reliability against packet losses on the links, these losses are studied in detail (Chapter 4). This includes deriving detailed expressions for a sample use case using symbolic math (Section 4.4) and case studies on different network topologies. It has also been discussed on how these expressions can aid in understanding the network characteristics better. Case studies on series-parallel graphs demonstrated how redundancy affects the network characteristics at a fundamental level, and case studies on ladder redundant topologies provided insights into quantitative comparison and analysis of more realistic networks (Chapter 5). For node losses, analytical models and a pseudocode corresponding to the implementation in MATLAB are provided (Chapter 4). However, case studies are not provided due to insufficient time and less priority compared to the study of link losses. Interested readers are advised to run the code on various use cases for more insights into these losses.
- **Buffer Loss Analysis:** *Contents removed for confidentiality reasons.*

The conclusions related to implementation of these analytical models, and their usage for building optimal designs are as follows:

- **Implementation of the analysis in MATLAB/Simulink:** To start with, all the analysis and validation mechanisms described in this thesis are implemented in MATLAB/Simulink and the results from the implementations are discussed in this report. In addition, various

demo analysis is scripted in the format of MATLAB live scripts. These scripts aid in understanding various concepts discussed in this thesis, and are reusable, thus making it easier to perform analyses on different configuration settings. Graphical User Interfaces (GUI) are also built to analyze link and node losses (along with optional speedup settings, and automatic design space exploration capabilities) to improve the ease of use for new users and developers.

- **Computation Speedup:** The analytical models for link and node losses have exponential running time and therefore are not suitable for large networks. Two types of speedup are implemented to overcome this problem. The improvements using GPU have saturated at $\sim 16X$ improvement due to limitations in the maximum number of parallel threads. The speedup using approximation algorithms is configurable (with parameter K) and the user can trade-off computation time for accuracy.
- **Design Space Exploration:** *Contents removed for confidentiality reasons.*

In the next Section, a list of future work that can be improved/extended on top of this work is presented.

9.2 Future Work

This section lists out the future works that can be done on top of this project. This section is divided into two parts. Section 9.2.1 lists out improvements to concepts discussed in this project and Section 9.2.2 provides extensions of the work to other projects/domains where the concepts discussed in this project will prove useful.

9.2.1 Improvements of existing work

The improvements of existing work are as follows

- **Analysis of Vector and Match Recovery Algorithms:** As pointed out in 1.2, the impact of choice of recovery algorithms and memory size used by them are ignored as part of the analysis in this project. This analysis is crucial since the memory used transposes to the cost of the network. Lesser memory might lead to elimination function not working properly and thereby causing congestion in the network. Similarly, if the memory is very large, this might lead to more service time of the packet (notice the for loop described in Section 7.4.3.6 of [22] which runs proportional to memory size used), thereby could result in buffer losses.
- **Analysis of MultiCast messages:** Currently, the analysis is focused on only unicast messages. However, in practice, the talker can transmit multicast messages. In this context, it would be a nice improvement to extend the analysis to accommodate the multicast transfer.
- **Latency analysis:** In this project all the analysis is focused on the successful delivery of the packet. In applications having hard deadlines, the packets should also reach on time. For this reason, analysis of the latency distribution of packets can be a good future work. This analysis can be done either using Discrete Time Markov Chains (DTMC) modeling (as done in Chapter 6) or by computing approximations from Continuous Time Markov Chains (CTMC) modeling (as done in [26], and using Jackson networks). Other ways of analysis is to use the concepts from network calculus [11] and stochastic network calculus [24].
- **Buffer loss analysis at network level:**
Contents removed for confidentiality reasons

- **Improvements in parallel processing capabilities:** As discussed in Chapter 7 and in Section 7.2, the implementation of computations over Graphical Processing Units (GPU) is currently limited to computing the probabilities of states. However, this can be improved to also compute the reachability of every state. It is also noted that at high state-space, memory overflow of CPU happens. To prevent this, one has to fetch data from the external memory, process it in the internal memory and then store it back to the external memory. This process introduces additional delay in fetching time to internal memory. In literature to perform these computations efficiently (time), IO efficient algorithms are generally used [30].
- **Improvements in approximation algorithm:** In Chapter 7 and in Section 7.3.2, an explanation is provided related to the scope for improving the upper bound of approximate reliability calculation. In addition approaches similar to the ones used in Streaming Algorithms [40] can prove useful. The key idea in these cases is to compute the reachability of a random state from the state-space and use this information along with the state properties (number of failed components) to update the probabilities of the state-space, and for updating the lower and upper bounds.
- **Auto Construction of Networks:**
Contents removed for confidentiality reasons.
- **Neural Networks for approximating buffer loss probability:** As discussed in Chapter 6, the computation time required for computing buffer losses can become high (if state transition matrix is large) and not always quantization approach (Section ■) can prove useful. In such cases, especially when the whole calculations have to be repeated for a little change in inputs, the process is not computationally efficient. Neural networks in this context can prove as a useful solution. Neural networks can be trained to predict the outputs for a given input configuration (More the features and better their quality, better is the prediction). Some initial results using only fully connected layers are developed. The results are promising but not sufficient(Appendix A).

9.2.2 Extensions of the work to other projects/domains

A lot of concepts discussed in this thesis are not specific to FRER and can be applied in a wider context. Particularly, the analysis can be extended to analyze other TSN standards described in Section 1.1.2.

- **Analysis of IEEE 802.1 AS-Rev:** As pointed in the Section 1.1.2, IEEE 802.1 AS-Rev, similar to IEEE 802.1CB employs the concept of redundancy using multiple grandmaster clocks and multiple connections. For this reason, the analysis described in this thesis will prove useful for analyzing this standard.
- **Analysis of IEEE 802.1 Qbv:** IEEE 802.1 Qbv describes mechanisms for time-aware shaping where ports are opened according to a time schedule in order to prevent delays during a scheduled transmission. The concepts described in buffer loss analysis will prove useful for analyzing this standard.
- **General applications:** The concepts described in this thesis, particularly analytical modeling using Markov Chains, formulating metrics from state-space, validation using bootstrap hypothesis testing, reduction of computation time using approximation algorithms, parallel processing using GPU's (also interfacing CUDA code with MATLAB) and building optimal designs using design space exploration are applicable to a wider context of problems. The code related to analysis, validation and visualization, and documentation developed in this process will prove useful in such cases.

Bibliography

- [1] Waqar Ahmed, Osman Hasan and Sofiene Tahar. “Formal Dependability Modeling and Analysis: A Survey”. In: *arXiv:1606.06877 [cs, math]* (22nd June 2016). arXiv: 1606.06877. URL: <http://arxiv.org/abs/1606.06877>.
- [2] Sanjeev Arora. “Polynomial Time Approximation Schemes for Euclidean Traveling Salesman and Other Geometric Problems”. In: *J. ACM* 45.5 (Sept. 1998), pp. 753–782. ISSN: 0004-5411. DOI: 10.1145/290179.290180. URL: <http://doi.acm.org/10.1145/290179.290180>.
- [3] *Automotive electronics cost as a percentage of total car cost worldwide from 1950 to 2030*. URL: <https://www.statista.com/statistics/277931/automotive-electronics-cost-as-a-share-of-total-car-cost-worldwide/>.
- [4] *AVB task group*. URL: <http://www.ieee802.org/1/pages/avbridges.html>.
- [5] C. M. Kozierok C. Correa R. B. Boatright and J. Quesnelle. *Automotive Ethernet: The definitive guide*. 2005.
- [6] *Bootstrap Hypothesis Test*. URL: <http://www.stat.ucla.edu/~rgould/110as02/bshypothesis.pdf>.
- [7] *Calculation of Pareto points - File Exchange - MATLAB Central*. URL: <https://nl.mathworks.com/matlabcentral/fileexchange/22507>.
- [8] Marko Cepin. *Assessment of Power System Reliability*. 2011. URL: <http://link.springer.com/10.1007/978-0-85729-688-7>.
- [9] *Chain rule (probability)*. URL: [https://en.wikipedia.org/wiki/Chain_rule_\(probability\)](https://en.wikipedia.org/wiki/Chain_rule_(probability)).
- [10] *Create discrete-time Markov chain - MATLAB - MathWorks Benelux*. URL: <https://nl.mathworks.com/help/econ/dtmc.html>.
- [11] Joan Adrià Ruiz De Azua and Marc Boyer. “Complete Modelling of AVB in Network Calculus Framework”. In: *Proceedings of the 22Nd International Conference on Real-Time Networks and Systems*. RTNS ’14. Versaille, France: ACM, 2014, 55:55–55:64. ISBN: 978-1-4503-2727-5. DOI: 10.1145/2659787.2659810. URL: <http://doi.acm.org/10.1145/2659787.2659810>.
- [12] *Electronic control unit*. URL: https://en.wikipedia.org/wiki/Electronic_control_unit.
- [13] *Erlang Fixed Point Method*. URL: <https://github.com/saikrishh123/erlang>.
- [14] Norman Finn. “Failure rates and P802.1CB”. In: (2013), p. 20. URL: <http://www.ieee802.org/1/files/public/docs2013/cb-nfinn-packet-loss-ratio-10-13-v03.pdf>.
- [15] *Fixed Point Methods for Loss Networks*. URL: <https://people.smp.uq.edu.au/PhilipPollett/talks/KOREA/slides/korea.html>.
- [16] Marc Geilen et al. “An Algebra of Pareto Points”. In: *Fundam. Inf.* 78.1 (Jan. 2007), pp. 35–74. ISSN: 0169-2968. URL: <http://dl.acm.org/citation.cfm?id=1366007.1366010>.
- [17] Giovanni Giambene. *Queuing Theory and Telecommunications: Networks and Applications*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. ISBN: 0387240659.

- [18] Armin Heindl. “Decomposition of general queueing networks with MMPP inputs and customer losses”. In: *Performance Evaluation* 51.2 (2003), pp. 117–136. ISSN: 0166-5316. DOI: [https://doi.org/10.1016/S0166-5316\(02\)00091-3](https://doi.org/10.1016/S0166-5316(02)00091-3). URL: <http://www.sciencedirect.com/science/article/pii/S0166531602000913>.
- [19] *IEEE 802 Time Sensitive Networking: Extending Beyond AVB*. URL: https://standards.ieee.org/events/automotive/08_Teener_TSN.pdf.
- [20] “IEEE Standard for Local and Metropolitan Area Networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks”. In: *IEEE Std 802.1AS-2011* (Mar. 2011), pp. 1–292. DOI: 10.1109/IEEESTD.2011.5741898.
- [21] “IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams”. In: *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)* (Jan. 2009), pp. C1–72. DOI: 10.1109/IEEESTD.2009.5375704.
- [22] “IEEE Standard for Local and metropolitan area networks—Frame Replication and Elimination for Reliability”. In: *IEEE Std 802.1CB-2017* (Oct. 2017), pp. 1–102. DOI: 10.1109/IEEESTD.2017.8091139.
- [23] “IEEE Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)”. In: *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)* (Sept. 2010), pp. 1–119. DOI: 10.1109/IEEESTD.2010.5594972.
- [24] Yuming Jiang and Yong Liu. *Stochastic Network Calculus*. London: Springer-Verlag, 2008. ISBN: 978-1-84800-126-8. URL: <http://www.springer.com/gp/book/9781848001268>.
- [25] Feng Luo Jinpeng Xu. “Failure rate analysis for time-sensitive networking”. In: (2016), p. 20. URL: download.atlantispress.com/php/download_paper.php?id=25864354.
- [26] Nam K. Kim and Kyung C. Chae. “Transform-free analysis of the GI/G/1/K queue through the decomposed Little’s formula”. In: *Computers & Operations Research* 30.3 (2003), pp. 353–365. ISSN: 0305-0548. DOI: [https://doi.org/10.1016/S0305-0548\(01\)00101-0](https://doi.org/10.1016/S0305-0548(01)00101-0). URL: <http://www.sciencedirect.com/science/article/pii/S0305054801001010>.
- [27] Nobukatsu KITAJIMA and Dnyaneshwar Kulkarni. “Time Sensitive Network enabling next generation of automotive E/E architecture”. In: Oct. 2015.
- [28] *Law of total probability*. URL: https://en.wikipedia.org/wiki/Law_of_total_probability.
- [29] *Markov Chain*. URL: https://en.wikipedia.org/wiki/Markov_chain.
- [30] Sraban Kumar Mohanty. “I/O Efficient Algorithms for Matrix Computations”. In: *CoRR* abs/1006.1307 (2010). arXiv: 1006.1307. URL: <http://arxiv.org/abs/1006.1307>.
- [31] *Multi-objective optimization*. URL: https://en.wikipedia.org/wiki/Multi-objective_optimization.
- [32] P. Ngatchou, A. Zarei and A. El-Sharkawi. “Pareto Multi Objective Optimization”. In: *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*. Nov. 2005, pp. 84–91. DOI: 10.1109/ISAP.2005.1599245.
- [33] *OMNeT++ Discrete Event Simulator - Home*. URL: <https://www.omnetpp.org/>.
- [34] Bernard Philippe, Youcef Saad and William J. Stewart. “Numerical Methods in Markov Chain Modelling”. In: *Operations Research* 40 (1996), pp. 1156–1179.
- [35] *PRISM - Probabilistic Symbolic Model Checker*. URL: <https://www.prismmodelchecker.org/>.
- [36] *SimEvents*. URL: <https://nl.mathworks.com/products/simevents.html>.
- [37] *SimPy*. URL: <http://simpy.readthedocs.io/en/latest/>.

- [38] *Solve shortest path problem in graph - MATLAB graphshortestpath*. URL: <https://nl.mathworks.com/help/bioinfo/ref/graphshortestpath.html>.
- [39] *Statistics 110: Probability*. URL: <https://projects.iq.harvard.edu/stat110/home>.
- [40] *Streaming Algorithms*. URL: https://en.wikipedia.org/wiki/Streaming_algorithm.
- [41] *Symbolic Math Toolbox*. URL: <https://nl.mathworks.com/products/symbolic.html>.
- [42] János Sztrik and Zalán Heszberger. “Basic Queueing Theory”. In: 2012.
- [43] *The applicability of Markov Analysis methods to reliability, Maintainability, and Safety*. URL: <https://www.dsiac.org/sites/default/files/reference-documents/markov.pdf>.
- [44] Leiming Xing, Karl N. Fleming and Wee Tee Loh. “Comparison of Markov model and fault tree approach in determining initiating event frequency for systems with two train configurations”. In: *Reliability Engineering & System Safety* 53.1 (1st July 1996), pp. 17–29. ISSN: 0951-8320. DOI: 10.1016/0951-8320(96)00033-6. URL: <http://www.sciencedirect.com/science/article/pii/0951832096000336>.

Appendices

Appendix A

Neural Networks for estimating buffer loss probability

In this chapter, the neural network training used for estimating buffer losses is presented. The organization of the chapter is as follows. Section A.1 presents features of the dataset and how it is created. Section A.2 discusses the architecture of the model and training results.

A.1 Dataset

The dataset used is of size 11000 X13 and is created by using the analytical models described in Chapter 6. Random probability distributions are generated for discrete inter-arrival and service time distributions. The buffer loss is analyzed on this random distributions for a random buffer size. All these randomly generated configurations are constrained by suitable limits. 13 different features are then extracted from the inter-arrival and service time distributions. These features are mean, variance, skewness, kurtosis, min state and max state of inter-arrival and service time distribution. These 12 along with the buffer size form the 13 features of the dataset. Figure A.1 shows the 3D scatter plot of all the data points for the features mean inter arrival time , mean service time ad buffer size. In the next Section, we describe the training method used to train the neural network.

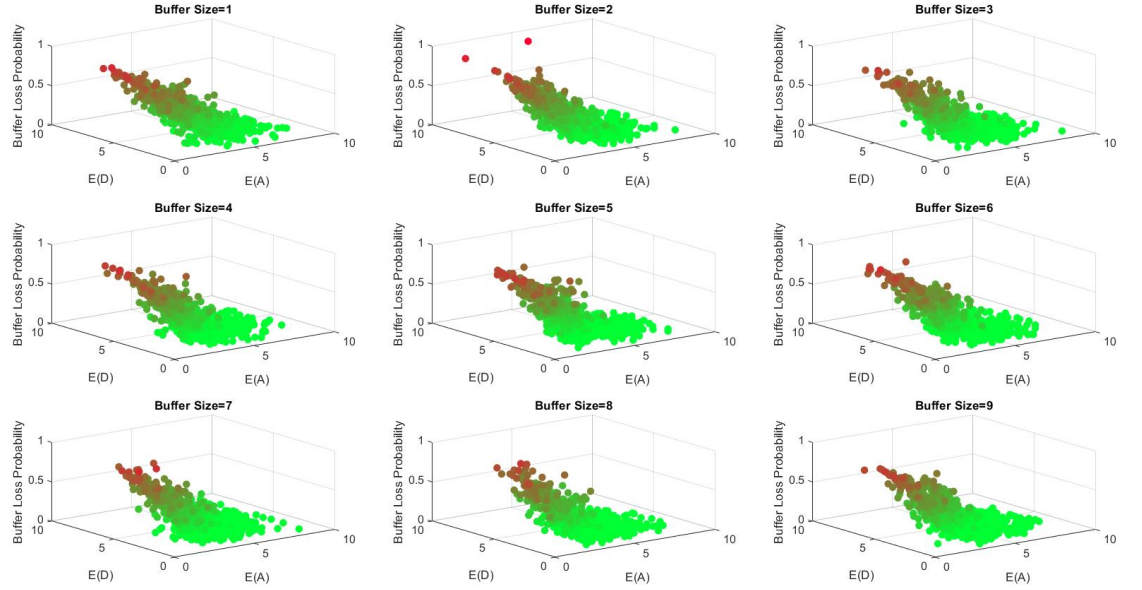


Figure A.1: 3D scatter plots of $E(A)$ vs $E(D)$ vs $BufferLossProb$ for varying buffer sizes

A.2 Neural network training

Figure A.2 shows the architecture of the network used for training. 3 hidden layers each of size 20 nodes are used. The activation functions in all the layers used are tangential sigmoid (This is mostly because, the buffer losses are in general 0 to 1 and tangential sigmoid layers model non linearity in this range well). For the final layer pure linear activation is used.

The results of the training process are shown in Figure A.3. It can be observed from the Figure that most of the error is concentrated around 0, since the scale of the buffer losses is low, this error is significant. Other ways to improve the model is to improve the input features quality (currently very basic features are presented). Also the use of Convolution and LSTM layers can add more value and will be explored in the future.

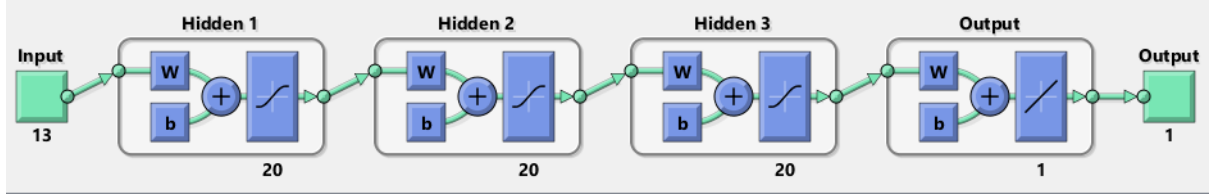


Figure A.2: Neural Network architecture used for training

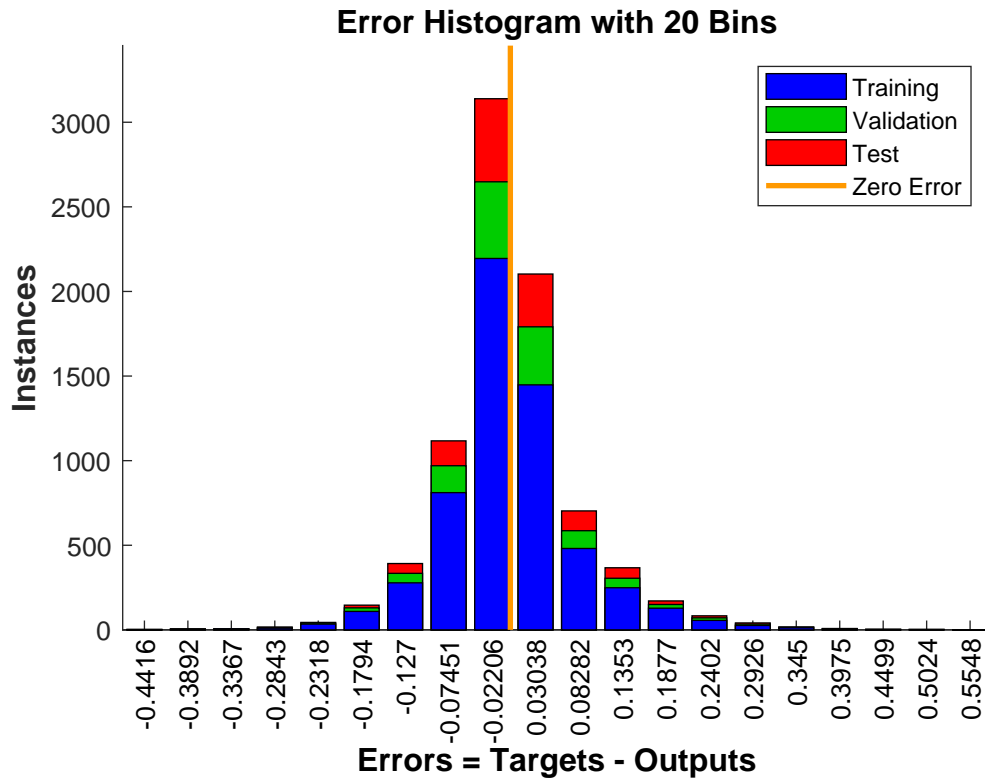


Figure A.3: Error histogram of the training on different data sets (training, validation and test)