A topological reliability model for TCP/IP over Ethernet networks

E Coetzee

12080578

Dissertation submitted in fulfilment of the requirements for the degree *Magister in Computer and Electronic Engineering* at the Potchefstroom Campus of the North-West University

Supervisor: Prof ASJ Helberg

May 2014

A topological reliability model for TCP/IP over Ethernet networks

Eugene Coetzee

A topological reliability model for TCP/IP over Ethernet networks

Eugene Coetzee

Revision 2 - printed on 28 April 2014 as final examined and corrected submission.

Declaration

I, Eugene Coetzee, hereby declare that the dissertation entitled: “A topological reliability model for TCP/IP over Ethernet networks”, submitted in fulfilment of the requirements for the degree M.Eng is my own work, except where acknowledged in the text, and has not been submitted to any other tertiary institution in whole or in part.

Signed at Potchefstroom.

_____ _____

Eugene Coetzee Date

Abstract

Network failures can originate from or be located in any one of several network layers as described by the OSI model. This investigation focuses on the role of physical topological design parameters in determining network reliability and performance as can be expected from the point of view of a typical client-server based connection in an Ethernet local area network. This type of host-to-host IP connection is found in many commercial, military and industrial network based systems. Using Markov modelling techniques reliability and performability models are developed for common network topologies based on the redundancy mechanism provided by IEEE spanning tree protocols. The models are tested and validated using the OPNET network simulation environment. The reliability and performability metrics calculated from the derived models for different topologies are compared leading to the following conclusions. The reliability of the entry-nodes into a redundant network is a determining factor in connection availability. Redundancy mechanisms must be extended from the entry-node to the connecting hosts to gain a significant benefit from redundant network topologies as network availability remains limited to three-nines. The hierarchical mesh network offers the highest availability (seven-nines) and performability. Both these metrics can be accurately predicted irrespective of the position of the entry-node in the mesh. Ring networks offer high availability (five to seven-nines) and performability if the ring remains small to medium sized, however for larger rings ($N \geq 32$) the availability is highly dependant on the relative position of the entry-node in the ring. Performability also degrades significantly as the ring size increases. Although star networks offer predictable and high performability the availability is low (four-nines) because of the lack of redundancy. The star should therefore not be used in IP networked systems requiring more than four-nines availability. In all the topologies investigated the reliability and performability can be increased significantly by introducing redundant links instead of single links interconnecting the various nodes, with the star topology availability increasing from four-nines to seven-nines and performance doubling.

Keywords. network topology, reliability, availability, performability, Ethernet LAN, switch, spanning tree, redundancy

Opsomming

Netwerkfalings kan ontstaan of geleë wees in enige van verskeie netwerkvlakke soos beskryf deur die OSI model. Hierdie ondersoek fokus op die rol van fisiese topologiese ontwerpsparameters in die bepaling van netwerk betroubaarheid en verrigting in die konteks van 'n tipiese kliënt-bedienergebaseerde verbinding in 'n Ethernet lokale-area netwerk. Hierdie tipe van gasheer-tot-gasheer IP verbinding kom voor in verskeie kommersiële, militêre en industriële netwerkgebaseerde stelsels. Deur gebruik te maak van Markov stelselmodellering word betroubaarheid en verrigbaarheid modelle ontwikkel vir algemene netwerk topologieë gebaseer op die oortolligheidsmeganisme wat in die IEEE lusbeheer protokolle ingebou is. Die modelle word dan getoets en gevalideer met behulp van die OPNET netwerksimulasie-omgewing. Die betroubaarheid en verrigbaarheid statistieke bereken uit die modelle vir verskillende topologieë word met mekaar vergelyk en lei tot die volgende gevolgtrekkings. Die betroubaarheid van die intreenodes in 'n oortollige netwerk is 'n bepalende faktor wat beskikbaarheid betref. Oortolligheidsmeganismes moet uitgebrei word vanaf die gasheer tot by die intreenode om 'n beduidende voordeel uit oortolligheid te verseker, anders bly beskikbaarheid beperk tot drie-neges. Die hiërargiese lusnetwerk bied die hoogste beskikbaarheid (sewe-neges) asook verrigbaarheid. Beide hierdie statistieke kan akkuraat voorspel word, ongeag van die posisie van die intreenode in die netwerk. Ringnetwerke bied hoë beskikbaarheid (vyf tot sewe-neges) en verrigbaarheid indien die ring klein tot medium grootte bly, maar vir groter ringe ($N \geq 32$) is die beskikbaarheid afhanklik van die relatiewe posisie van die intreenode in die ring. Verrigbaarheid neem ook aansienlik af as die ring grootte toeneem. Hoewel sternetwerke voorspelbare beskikbaarheid bied saam met hoë verrigbaarheid is die beskikbaarheid laag (vier-neges) as gevolg van die gebrek aan 'n oortolligheidsmeganisme. Die ster topologie moet dus nie gebruik word in IP netwerkstelsels wat meer as vier-neges beskikbaarheid vereis nie. In al die topologieë wat ondersoek is kan die betroubaarheid en verrigbaarheid aansienlik verhoog word deur gebruik te maak van oortollige skakels in plaas van enkel skakels wat die verskillende nodes verbind, met die ster topologie beskikbaarheid wat verhoog van vier-neges tot sewe-neges en met 'n gepaardgaande verdubbeling in verrigting.

Acknowledgements

I hereby express my gratitude and appreciation to the following people whose contributions, support and guidance contributed hugely to the completion of this work.

Professor Albert Helberg who was my mentor and provided expert advice and guidance.

The researchers at the LAND laboratory of Federal University of Rio de Janeiro/Brazil (UFRJ) for making available under a royalty-free use and modification license the excellent Tangram II computer system modelling environment for conducting academical research.

My loving wife and sons who always understood and supported me during the long weekends of struggle.

Table of Contents

List of Figures

List of Tables

Abbreviations

ADSL	Asymmetric Digital Subscriber Line
AP	Access Point
ARP	Address Resolution Protocol
AS	Autonomous System
ASIC	Application-Specific Integrated Circuit
BER	Bit Error Rate
BGP	Border Gateway Protocol
BMS	Building Management System
CDF	Cumulative Distribution Function
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
CSV	Comma Separated Values
CTMC	Continuous Time Markov Chain
DES	Discrete Event Simulation
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DSCP	Differentiated Services Code Point
DTMC	Discrete Time Markov Chain
EGP	Exterior Gateway Protocol
EIGRP	Enhanced Interior Gateway Routing Protocol
ERP	Ethernet Ring Protection
GLBP	Gateway Load Balancing Protocol
GSPN	Generalized Stochastic Petri Net
GTH	Grassmann-Taksar-Heyman
HMM	Hybrid Markov Model
HSRP	Hot Standby Router Protocol
ICMP	Internet Control Message Protocol
ID	Identifier

IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGRP	Interior Gateway Routing Protocol
IP	Internet Protocol
ISP	Internet Service Provider
IT	Information Technology
ITU	International Telecommunication Union
JMT	Java Modelling Tool
LACP	Link Aggregation Control Protocol
LAN	Local Area Network
MAC	Media Access Control
MACMT	Mean Active Corrective Maintenance Time
Mbps	Megabits Per Second
MMPP	Markov Modulated Poisson Process
MRM	Markov Reward Model
MRP	Media Redundancy Protocol
MSTP	Multiple Spanning Tree Protocol
MTBF	Mean Time Between Failure
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
MVB	Multifunction Vehicle Bus
NAT	Network Address Translation
NIC	Network Interface Card
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
PCM	Packet Count Method
PDF	Probability Density Function
PEPA	Performance Evaluation Process Algebra
PFM	Path Fault Method

PLR	Packet Loss Ratio
PVST	Per-VLAN Spanning Tree
QoS	Quality of Service
QPN	Queueing Petri Net
REP	Resilient Ethernet Protocol
RFC	Request for Comments
RIP	Router Information Protocol
RSTP	Rapid Spanning Tree Protocol
SNMP	Simple Network Management Protocol
SPN	Stochastic Petri Net
SRN	Stochastic Reward Net
SSID	Service Set Identification
STP	Spanning Tree Protocol
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TGIF	TANGRAM Graphic Interface Facility
TTL	Time To Live
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
VOIP	Voice Over Internet Protocol
VRRP	Virtual Router Redundancy Protocol
WAN	Wide Area Network
WAP	Wireless Access Point
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access

Chapter 1. General Introduction

1.1. Overview

This chapter explores the general application and reliability of IP networks, a problem statement is formalised and clear scope and objectives are defined for this investigation.

1.2. Background and motivation

IP networks are replacing conventional analogue data communication systems.

The combination of the suite of Internet protocols over the Ethernet physical layer is also popularly referred to as the "TCP/IP over Ethernet network" or simply the "IP network", and is increasingly used to replace traditionally "hard-wired" links in general purpose communication systems including voice [1] and video [3], [4]. This convergence of disparate and isolated communication systems and the hosting of multiple applications on a common and shared network infrastructure is depicted in Figure 1.1.

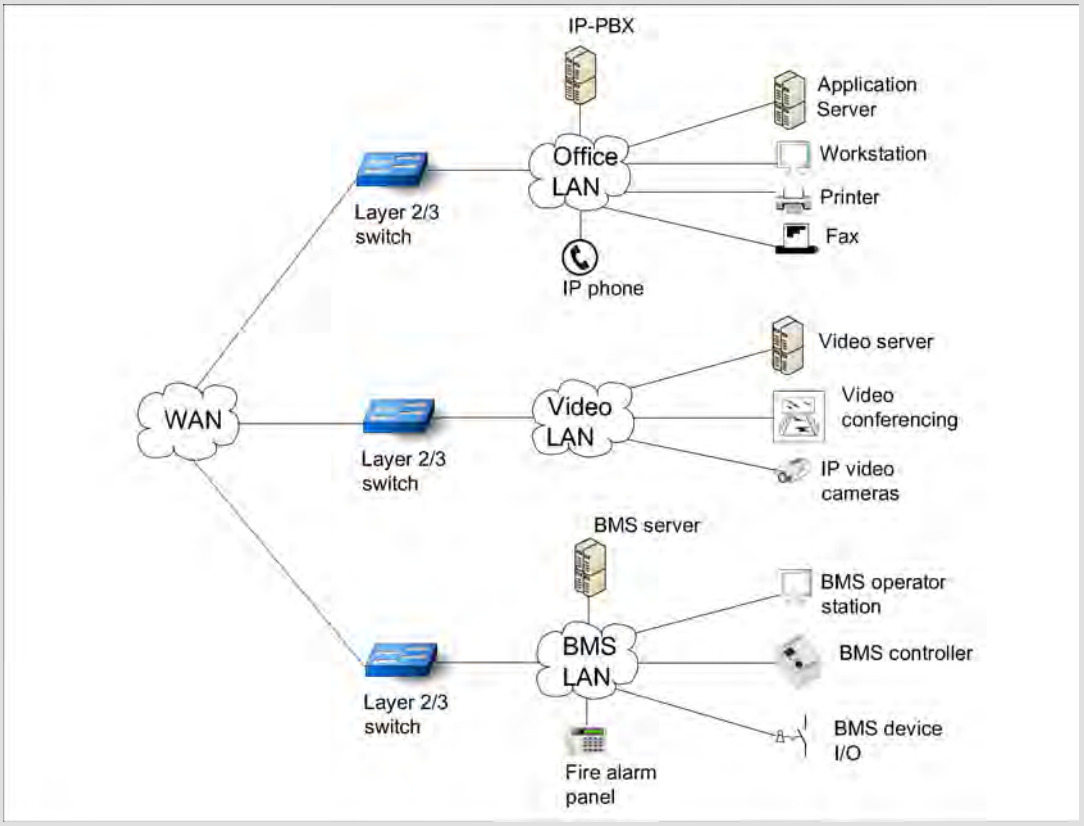

Figure 1.1. Network convergence

The corporate data network is hosted on the same network as the telephone system (VOIP) and the video conferencing systems. Also depicted in Figure 1.1 the building management system (BMS) and security subsystems are also deployed on the same shared network infrastructure.

This tendency towards IP networks replacing traditional "hard-wired systems" is also occurring in industrial control systems [5]. Traditional industrial serial protocols that would be deployed

using serial buses based on the RS422/485 physical layer, for example Modbus, DNP3, Profibus and various other fieldbus systems are now also deployed over the IP network [6], [7], [8], [9] as indicated in Figure 1.2.

PROTOCOL	BRAND NAME	STANDARDS
Modbus-TCP	Modbus	IEC 61158/ IEC 61784
DNP3 over TCP/IP	DNP3	IEC 60870
Foundation Fieldbus HSE	Foundation Fieldbus	IEC 61158/ IEC 61784
PROFINET CBA	Profibus	IEC 61158/ IEC 61784
Ethernet/IP	Devicenet	IEC 61158/ IEC 61784
IEC 61850	n/a	IEC 61850

Figure 1.2. Industrial networks

The IP network is an example of the modern data packet network that relies on the mechanism of packet switching for routing between transceivers. Data communication systems evolved over the past 40 years from simple analogue systems to more complex digital systems. Historically analogue signals would be "hard-wired" between transceivers through individual conductors. The next step in the evolution of data communication involved the digitization of analogue signals. The digital signal consists out of a continuous stream of data or **data packets**. Data streams can either be circuit switched or packet switched [10], [3] as indicated in Figure 1.3.

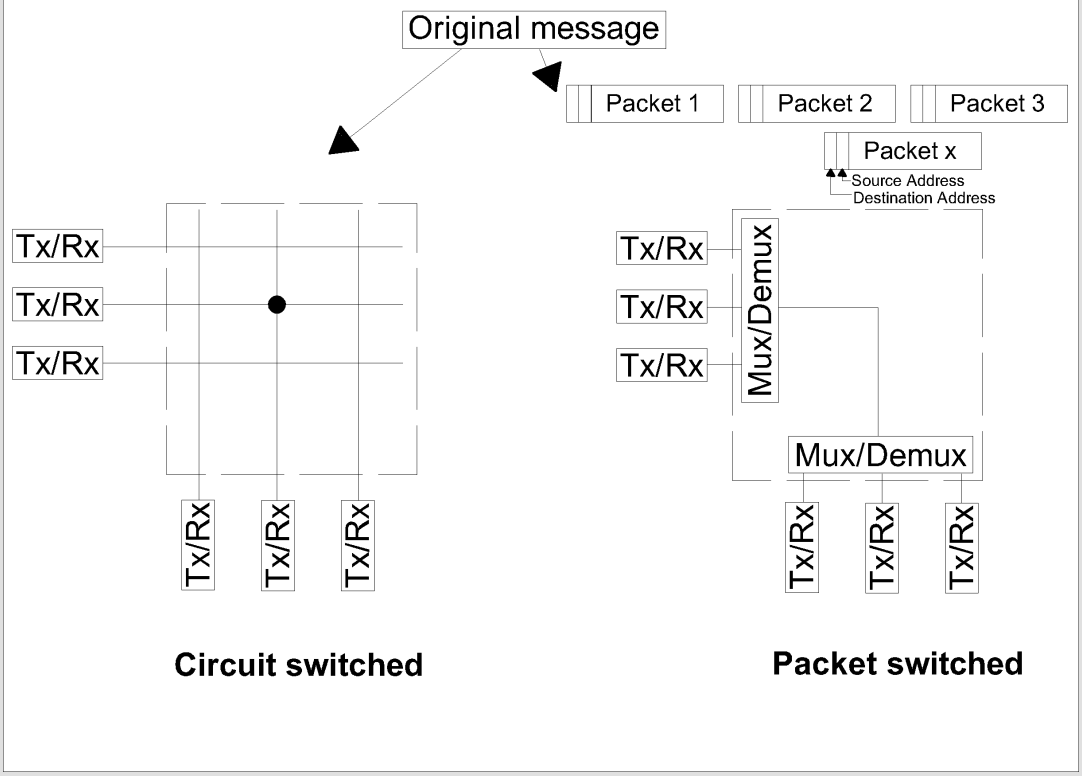

Figure 1.3. Circuit switched versus packet switched networks

The advantage of the circuit switched network as illustrated above is that every channel is independent with dedicated bandwidth, data packets can also be transported between transceivers in one continuous, sequential order. In the packet switched network however, channels are shared between transceivers. The implication is that bandwidth is also shared and the continuous data flow between individual transceivers can become interrupted. Depending on the routes available through the network data packets will not necessarily arrive sequentially. This has implications for applications that rely on real-time data exchange and low data channel latency. There are however obvious advantages to the packet switch network including flexibility and scalability which has led to its large scale adoption [3].

Digital networks have also dramatically improved in performance, measured in bandwidth, from byte-orientated, asynchronous serial digital networks running on low-bandwidth (9600 bps) modem links to high speed, synchronous, Ethernet networks with bandwidth of 100 Mbps to 1000 Mbps. The concurrent growth of the Internet and the development of the TCP/IP ARPA network model has contributed significantly to the establishment of a set of universal and commonly used communication standards and protocols [3]. The ARPA TCP/IP model is a layered model that can be subdivided and categorised in accordance to the ISO Open Systems Interoperability (OSI) model as indicated in Figure 1.4. The various **open** communication standards and protocols [11] depicted in Figure 1.4 collectively constitute what has become known as the Internet protocol suit including the following important protocols and standards:

- Transmission Control Protocol (TCP) [12] and User Datagram Protocol (UDP) [13] both on the transmission layer (layer-4);
- Internet Protocol (IP) [14] on the network layer (layer-3);
- Ethernet [15] on the data/physical layers (layer-2) of the OSI model.

The above open communication standards are universally implemented in "Commercial Off-The-Shelf" (COTS) network equipment.

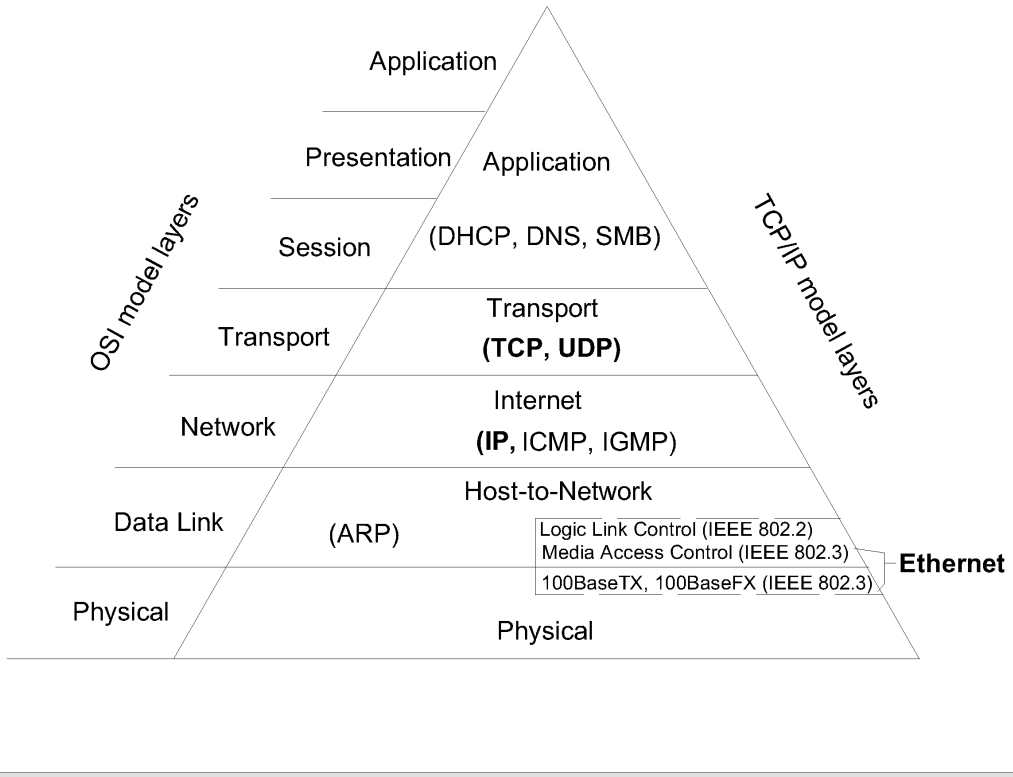

Figure 1.4. Internet Protocol OSI model

The IP network in Figure 1.5 has become the dominant Local Area Network (LAN) technology adopted in Information Technology (IT). The Ethernet based LAN, consisting of a set of interconnected Ethernet (or layer-2 switches), is ubiquitous and used universally [3] with other 300 million switched Ethernet ports installed worldwide by the year 2002 [16]. Accompanying the growth of the Ethernet LAN has been the TCP/IP based client-server software model. Virtually all network based applications rely entirely on services provided by the TCP and UDP networking protocols [3], [35].

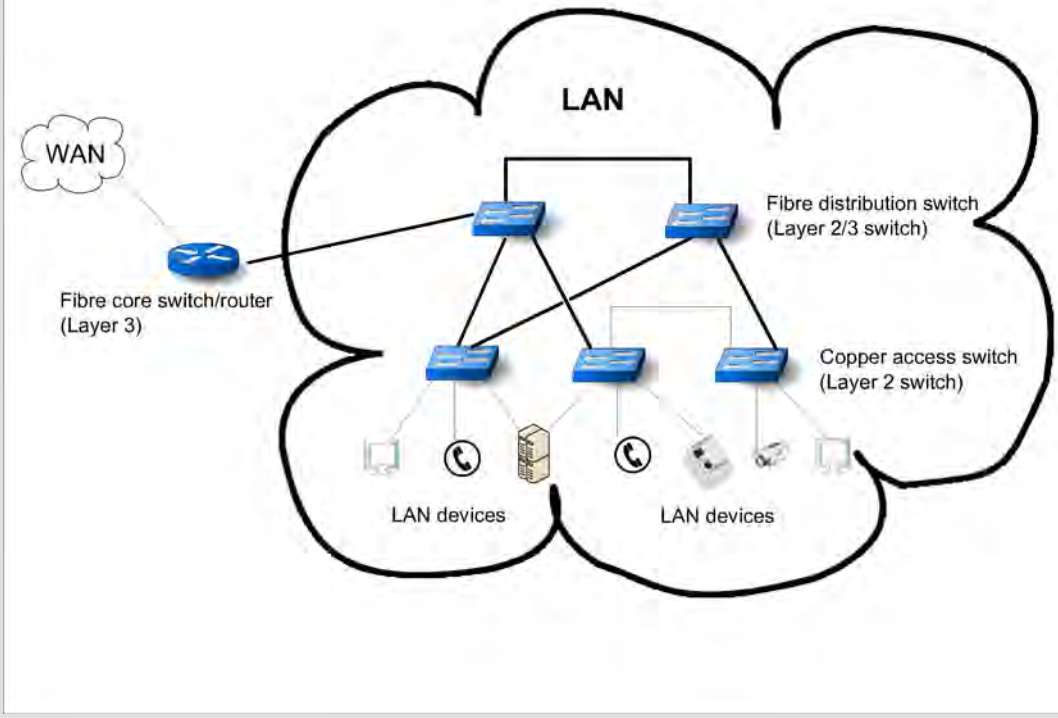

Figure 1.5. TCP/IP over Ethernet local area network

Although IP networks can be deployed in a variety of physical topologies - depicted in Figure 1.6 [29], [5]. [36], [37] they are often cabled in a physical star topology with the Ethernet (layer-2) switch at the centre of the star. Spanning tree protocols are used to build redundancy into a LAN [29], [38] with a range of possible physical topologies ranging from ring to mesh topologies including combinations of the two depicted in Figure 1.6.

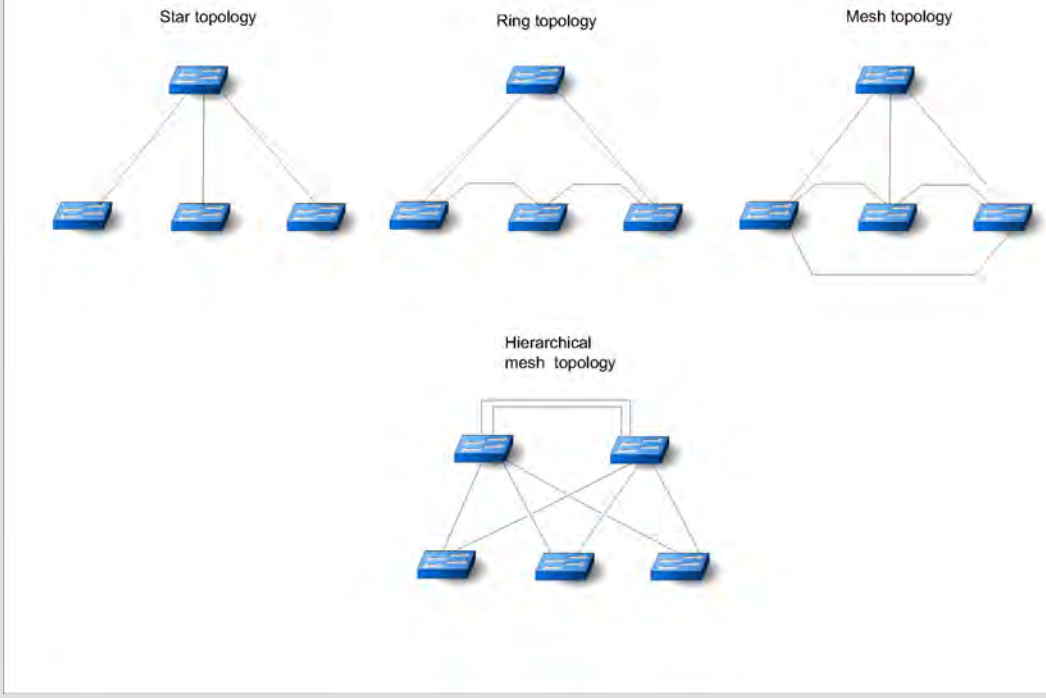

Figure 1.6. Network topology

Over and above the physical topology discussed, the IP network can also be deployed in various logical topologies. Virtual Local Area Networks (VLAN) [39] can be configured on VLAN-capable Ethernet switches [3], [29]. The VLAN is a logical partitioning technology that is deployed to isolate IP sub-networks that are physically connected to the same set of interconnected Ethernet switches (LAN).

It is difficult to build reliable IP networks. The communication stack depicted in Figure 1.4 consists of several layers of hardware and software that must interoperate correctly [27], [34]. Networks are part of larger, complex computing systems that consists of both hardware and software sub-systems and units. The complexity by itself becomes a cause of unreliability [53], not only because of the amount of software and hardware components involved, but also because of accompanying configuration and maintenance issues because of system complexity [54], [55]. The word "reliability" is used here in the broader sense than its formal technical definition, reliability refers to the broader *"measure of how a system matches its user's expectations"* [53]. Reliability also includes the concept of robustness [74] that is improved availability achieved by using redundancy techniques to improve the network fault-tolerance [29]. Modern manufacturers and suppliers of network based systems require five-nines network availability, **they expect less than five minutes of downtime per year** [26]. However, in the absence of the application of any redundancy techniques the typical availability of the Ethernet (layer-2) LAN is approximated to be in the order of three-nines to four-nines i.e. the network is **unavailable between two to eight hours a year** [1], [128], [28]. WAN availability figures are much more difficult to obtain or to calculate but are generally approximated to be two-nines to three-nines i.e. **unavailable between eight hours to four days a year** [1].

IP networks are packet switched networks, reliability therefore also includes performance issues around reliable and predictable packet delivery that are often the result of network congestion [3], also referred to as the Quality of Service (QoS) [56], [87]. Sub-optimal network architecture [29], sub-optimal network configuration [55], inadequate bandwidth and broadcast traffic are important factors that have a direct influence on QoS. For some time-critical applications including telephony and control systems, poor or unpredictable QoS leads to high network latency and a degradation in real-time performance [8], [5] - and therefore unreliability.

There are many contributing factors [3], [36], [27], [34], [37], [1], [55], [5], including both hardware and software systems and configurations, that must be taken in consideration when constructing a complete model that can be used to predict the reliability of the IP network - arranged according to the layers of the OSI model [2] and briefly summarised in Table 1.1. According to [2] up to 30% of network failures are related to the physical and data link layers.

Table 1.1. Reliability factors arranged to OSI layers

OSI layers	Contribution to system reliability
Medium	Physical topology of the LAN including cable ways and redundant links or paths for cables
	Cables and connectors
Physical and data link layers	Ethernet layer hardware including layer-2 switches and network interface cards (NICs)
	Redundant layer-2 switch configurations and STP convergence
	Logical topology of the LAN including QoS at layer-2, broadcast domain, network size and VLAN partitioning
Network and transmission layer	IP layer hardware including layer-3 switches and routers
	Routing protocols
	Redundant layer-3 switch/router configurations and redundant routing protocol convergence [89]
	QoS profile at layer-3
	Multicasting and IGMP snooping [88]
Application layer	Network management and configuration applications
	DHCP services
	DNS services
	SMB network file sharing
User layer	Client and server hardware
	Operating systems and network discovery services
	Application profiles
	Malware

1.3. Problem statement

There is no applied performability model that can be used for the evaluation of TCP/IP over Ethernet network topology.

"Performability" refers to a set of combined reliability metrics that is inclusive of performance or bandwidth utilisation [60], [61], [62].

There are various mathematical and statistical models that deal with network reliability and performance in generalised terms [57], [58]. On the other end of the spectrum there are also applied practical rule-of-thumb design rules and guidelines for Ethernet network topology using VLAN partitioning and redundant units i.e. cable links, switches and routers to improve network reliability, robustness [36], [37], [1], [5] and performance. There are however no applied, comparative models for TCP/IP over Ethernet network topology [26], [59], [32], [34] that combine network reliability and performance metrics.

1.4. Objectives of the investigation

The following formal objectives are set for this investigation:

- Develop a reliability model for commonly used Ethernet link topologies focusing on physical layer connectivity based on representative reliability parameters.
- Use the above link reliability model as a departure point to develop a comparative performability model for commonly used TCP/IP over Ethernet network topologies.
- Validate and test the performability models [90] of the above network topologies by subjecting them to validation simulation tests [72], [73] to confirm the influence of the identified topological factors on network reliability and performance.
- Evaluate the comparative model solutions and validation test results to make recommendations on best practice for designing reliable TCP/IP over Ethernet networks with performance as an important criteria.

1.5. Scope of the study

1.5.1. Scope definition

The scope of investigation is summarised as follows:

Develop a reliability and performability model for often used TCP/IP over Ethernet network topologies inside the IEEE and RFC standard compliant local area network (LAN) as indicated in Figure 1.5 with reference to the following factors that influence network reliability (see Table 1.1):

- Physical topology of the LAN including cable ways and redundant links or paths for cables.
- Cables and connectors.
- Ethernet layer-2 hardware including switches and network interface cards (NICs).
- Redundant layer-2 switch configurations and spanning tree convergence.

Proprietary protocols and product/vendor specific technologies that are not IEEE [15] or RFC [11] standards based are excluded from this investigation.

1.5.2. Investigation execution plan

The theoretical flow of the initial investigation execution plan is indicated in Figure 1.7 although there are likely to be deviations from and modifications to the initial plan that would result in a recursive approach, for example the literature study may have to be updated as unexplained or unexpected results in the experimental testing work are observed or the scope and objectives may become more focused to adapt to time and resource constraints.

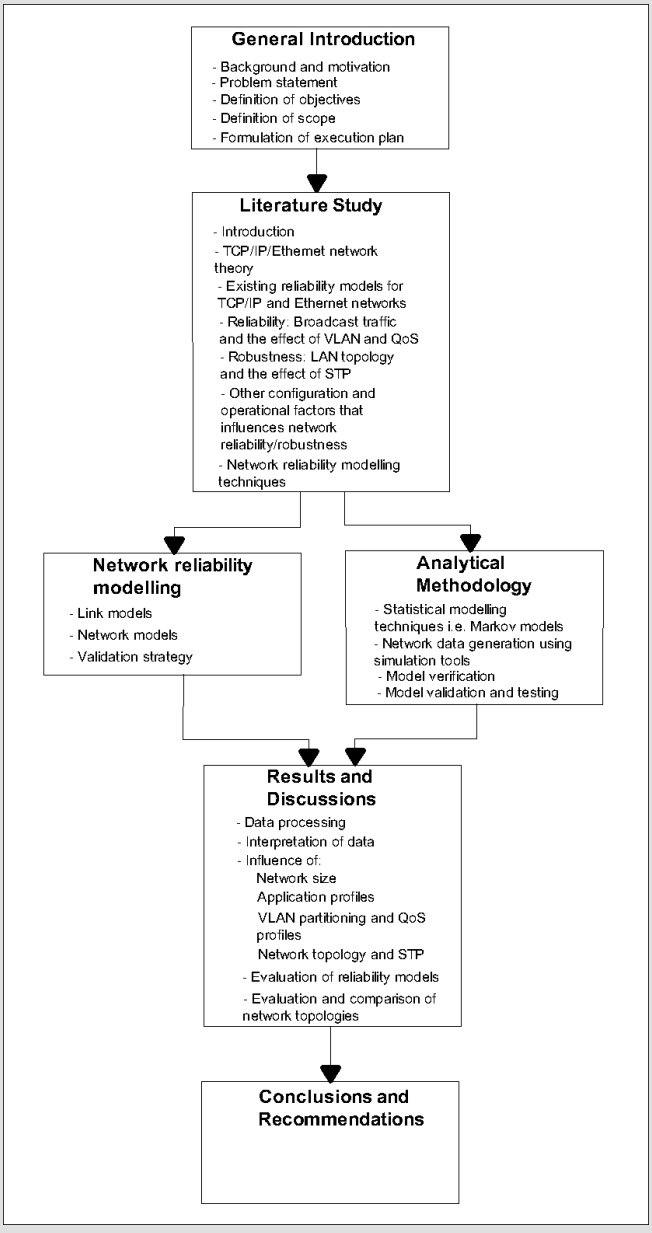

Figure 1.7. Investigation execution plan

Chapter 2. Literature Study

2.1. Introduction

In this chapter follows an in depth look at IP networks and in particular topological aspects that has an influence on IP network reliability. A thorough literature survey done on these aspects provides the necessary theoretical background to the IP network reliability models, validation techniques and observations presented.

The five major contributors to network based system failures or downtime are listed in [28] as including the following factors:

- hardware;
- software;
- environmental conditions;
- network operations and human error;
- network design.

The reliability of computing systems, focusing mostly on hardware and software, has been studied extensively since the early 1970s. [53] offers a comprehensive introduction of reliability issues that affect computing systems that extend beyond traditional hardware unit failures to include very complex systems with software units, network links, error detection, error recovery and protective redundancy measures. Improvements in reliability of hardware components have to some extent been negated by ever increasing complexity to such a degree that the complexity itself has become a cause of unreliability. This investigation adopts the Bellcore Reliability Prediction Procedure (RPP) terminology that views electronic systems as hierarchical **assemblies** consisting of a [147]:

- **Component**: A basic electronic device or part. The component can also refer to a single software routine or algorithm.
- **Unit**: Any assembly of constituent components (or devices) that performs a specific function or purpose. Reliability prediction figures will normally apply to a unit.
- **System**: An assembly of units. For the purpose of modelling "a service" will refer to the functionality provided by a software system, for example the DNS service will be provided by a server unit called the DNS server and a client unit called the DNS client [91].

The literature study is structured as follows: in Section 2.2 the basic building blocks of IP networks with emphasis on the network services and equipment in Figure 2.1 that are relevant to reliability modelling are discussed. In Section 2.3 existing reliability models are explored with an emphasis on models that represents or incorporate topological factors that determine network reliability. Section 2.6 takes an in depth look at how the physical topology influences IP network reliability while Section 2.7 looks into the influence of the logical topology. Section 2.7 looks at other reliability factors that are related to network topology.

Section 2.4 presents the most important network reliability analyses techniques and Section 2.9 discusses various popular network model validation techniques.

2.2. IP network theory

Various network units, systems and services have to work together in order to guarantee a working network [3]. The basic network services as described by the network communication layers in the Figure 1.4 are presented as a background theory to the reliability modelling of IP networks. It should be noted that although the OSI layers are often depicted as separated autonomous entities they work together through various software algorithms. These algorithms are embedded in the operating system network stack and, from a reliability point of view, the distinction between the different layers is irrelevant, depending on what reliability modelling approach is used [92]. Packet filters, for example, work over various OSI layers by filtering on MAC addresses (layer-2), IP addresses (layer-3) and TCP/UDP port numbers (layer-4) [54].

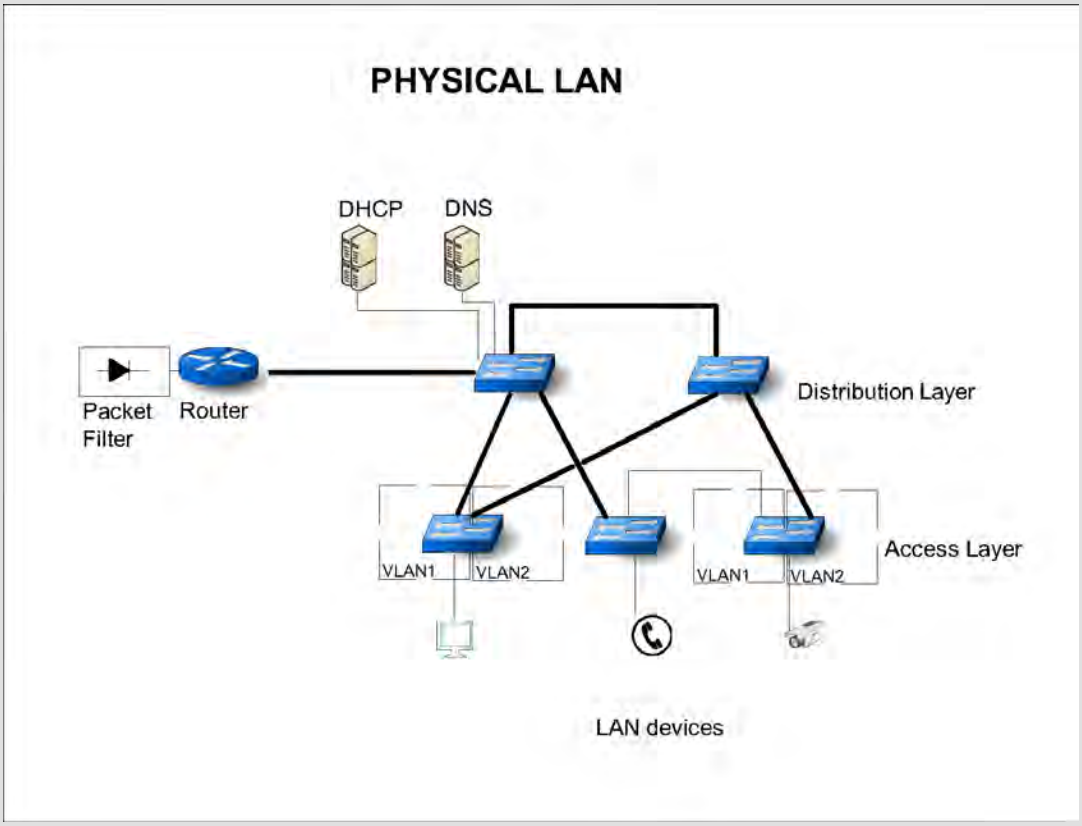

Figure 2.1. IP network system overview

The Internet Protocol communication stack has been standardised and documented as RFCs in a long-standing, evolutionary process of international cooperation and consensus coordinated by the Internet Engineering Task Force [11]. It should also be noted that various customised and vendor specific solutions exists to build IP networks. These solutions make use of proprietary protocols such as PVST, IGRP, EIGRP, GLBP, HSRP, REP or proprietary industrial ring protocols such as Real-Time Ring™, HIPER-Ring™ and Turbo Ring™ [29], [38], [151], [150]. Proprietary protocols are excluded from the scope and will not be included in the literature study.

2.2.1. Network access layer

The network access layer consists of the physical media and the data link layers, also referred to as layer-2 services including cables, connectors, network interface cards, network infrastructure equipment such as wireless access points (WAPs), switches and hubs. The Internet protocols

can be deployed on various physical topologies [54]. The literature study will focus on the most commonly used physical layers: the Ethernet LAN.

2.2.1.1. Ethernet and Wireless LAN

The IEEE LAN standards [15] for cabled systems (also known as Ethernet) and for wireless LAN systems (also known as Wifi) [93] define a set of standards and protocols that defines the physical layer communication on a local area network. The common addressing function that serves as an interface for the upper communication layers into the network access layer is known as the Media Access Control (MAC) layer. The MAC layer allocates a unique physical address, known as the MAC address, to every host that is attached to the network. The MAC address is a local embedded hardware address that uniquely identifies every network attached host and is only visible within the LAN or the physical domain. The Ethernet controller chip embedded in the network interface card negotiates and controls the physical access to the cabled LAN automatically [94], [54].

The Wifi standards differs from Ethernet in the provision that is made in the protocol for additional handshaking, data receipt acknowledgement, network identification and encryption security. These features are required since the medium is the open air, less reliable in comparison to cable and accessible to other untrusted Wifi enabled hosts. Wireless access points (WAPs) act as a bridge or entry point into the cabled LAN. A WAP broadcasts a Service Set Identifier (SSID) that assist the other Wifi-enabled hosts to identify a specific network associated with a WAP or group of WAPs. Security in the form of data encryption is implemented in a variety of encryption standards including WEP and WPA [95], [54].

2.2.1.2. Switches and hubs

Switches and hubs are the common interconnecting nodes at the core of the modern Ethernet LAN. They are used to interconnect network hosts through twisted-pair copper or fibre cable. Hubs act as simple signal regenerating node connecting all attached hosts to a single bus sharing the same physical collision domain. Switches, on the other hand, behave like the packet switch indicated in Figure 1.3 - the MAC addresses or layer-2 information in the Ethernet frames indicating the source and destination hosts [100]. Hubs are considered to be legacy equipment and should not be used in a well designed IP network because of inefficient utilisation of bandwidth. This investigation will focus on the influence of Ethernet switches deployed in different physical and logical (VLAN) topologies on IP network reliability.

The switch performs a MAC address learning routine whenever it senses an Ethernet frame entering a port. The MAC address-switch port mapping is stored in a MAC address table. Ethernet frames are then directed or switched between the ports attached to the source and destination ports creating a dedicated path. The basic working of the Ethernet switch is described as follows [100]:

- the switch receives an Ethernet frame from a host;
- the switch reads the MAC source and destination addresses in the Ethernet frame;
- the switch looks up the destination MAC address in the MAC switching table;
- the switch then forwards the frame through a dedicated virtual path to the destination host;
- the switching table is updated with the source MAC address and associated switch port.

Broadcast traffic is transmitted on all switch ports. It is important to note that an Ethernet switch "fails open", when the destination MAC address does not appear in the MAC table, the Ethernet frame is then broadcast on all the switch ports [100], [38].

2.2.1.3. Link aggregation and network card bonding

Link aggregation or "port trunking" refers to a technology that allows for more than one network link to be combined at (layer-2) into a single "trunk" and treated as a single physical network path [46], [47], [48]. Link aggregation is based on the link aggregation control protocol (LACP) for Ethernet as defined in IEEE 802.1ax (previously IEEE 802.1ad). The advantage is that the aggregated links allow for much higher throughput and automatic redundancy in the event where one of the links fail. As an example - 4 times 100 Mbps Ethernet links may be combined into one 400 Mbps aggregated link. There are limitations on how a trunk can be configured, notable a trunk must consist of links that are located on the same physical switch which implies that the end switches are still common points of failure. This limitation can be overcome with proprietary protocols including split multi-link trunking protocol in [49].

Seen from a host perspective, NIC bonding or NIC teaming (Linux operating system [50], [51] Windows operating system [52]) is a technique used to combine two network cards together to work as one link. Two basic modes of NIC bonding operation are used [51]. In the adaptive load balancing mode the links are wired into two different switches to provide "cold-standby" redundancy (see discussion in Section 2.4) with only one NIC operational and the other link in standby mode until a failure occurs in the primary link. In the link aggregation mode both links must be terminated on the same switch to derive the same benefits achieved with link aggregation between network switches discussed above, i.e. hot-standby redundancy and increased bandwidth.

2.2.1.4. Spanning Tree Protocol

Ethernet was initially not designed to be deployed in a ring topology, more than one physical path or link between two Ethernet nodes is problematic since it can cause a switching loop resulting in a "broadcast storm", severely compromising the correct operation of the network [100], [110]. A spanning tree protocol [39] is deployed on spanning tree enabled switches and hubs to prevent this from happening. The spanning tree enabled switches indicated in Figure 2.2 communicate with each other, and having detected the loop one or more of the switches block the ports that participates in the switch loop. In the event where a redundant link or switch fails the blocking port automatically unblocks and consequently redundant switch configurations are possible. A problem that arises from the older Spanning Tree Protocol (STP) is that the sensing of a failed network path and the resulting unblocking of the alternative path, depending on the network spanning diameter, can take up to thirty seconds to propagate or converge. It is therefore recommend that the improved Rapid Spanning Tree Protocol (RSTP) protocol is used instead of STP [110], [5].

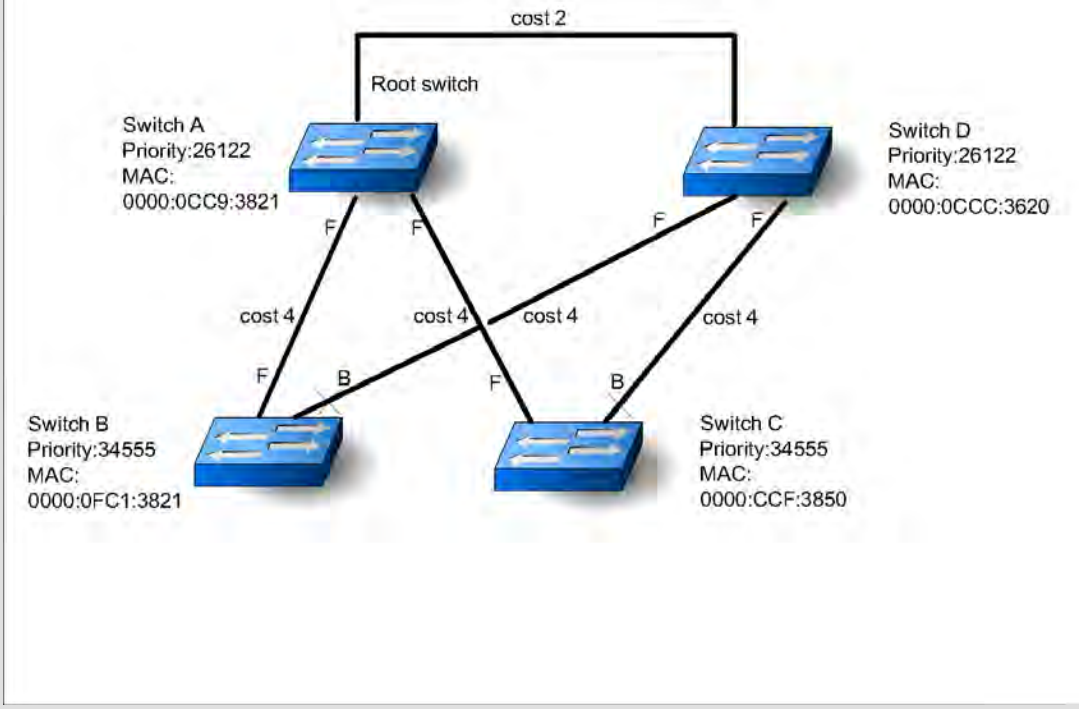

Figure 2.2. Rapid Spanning Tree Protocol

Referring to Figure 2.2, a brief explanation of the working of the spanning tree protocols follows [41], [43]. Each switch (or bridge) has a unique bridge ID that is a combination of a device ID (MAC address) and a configurable bridge priority number. Within a bridge each bridge port also has a unique port ID. A cost is assigned to each link interconnecting the bridges in the network. The spanning tree protocol elects the root bridge, that is the bridge with the lowest bridge ID by exchanging spanning tree information known as bridge protocol data units (BPDU) between the various bridges thereby communicating network topology changes. Every bridge then elects the port with the least-cost path to the root bridge as the root port and the port on the destination bridge on the opposite side of the link that provides the path to the root bridge is elected as a designated port. The path cost is the sum of the costs of every network segment that forms part of a path: where each individual segment's path cost is determined by its bandwidth, for example a 100 Mbps link has an associated path cost of 19 and a 1 Gbps link has an associated path cost of 4. All ports on the root bridge is set to the forwarding mode, designated and root ports on non-root bridges are also set to the forwarding mode while all other ports are set to a blocking mode thus eliminating all loops.

The first implementation of Spanning Tree Protocol (STP) was standardised as IEEE 802.1D [39], [41]. Rapid Spanning Tree Protocol (RSTP) standardised as IEEE 802.1W [40] was developed to offer improvements in convergence time, improving redundancy fail-over time from around 60 seconds to three seconds depending on the network diameter. The three second convergence time is based on maximum network diameter of seven bridges and can be considerably longer in larger diameter networks with the convergence time also being sensitive to the position of the root bridge and the relative position of the failed link or bridge relative to the root bridge [44], [45]. Multiple Spanning Tree Protocol (MSTP), standardised as IEEE 802.s [42], is based on RSTP

but allows "per-VLAN" or multiple spanning trees for each VLAN group in a physical LAN and blocks all but one of the possible alternate paths within each spanning tree instance.

2.2.1.5. Virtual Local Area Network

Although the layer-2 switch limits the physical collision domain, Ethernet broadcast traffic that is a necessity for the correct operation of many higher layer protocols including IP can still saturate the available bandwidth as the amount of hosts sharing the LAN and application diversity increases [96]. The advantage of deploying virtual LAN (VLAN) is that it **isolates broadcasting traffic** between different sub-networks at the physical layer [3]. Although Ethernet switches isolate the physical collision domain, various protocols including IP rely on broadcast (or multicast) Ethernet frames [3] that leads to a reduction in available bandwidth and network congestion as the amount of network hosts and applications hosted on a LAN increases. VLANs isolate sub-networks at the physical layer. In order for hosts to communicate between different sub-networks IP packets have to be forwarded between the different sub-networks using either routers or layer-3 switches. In general layer-3 switches are used inside the LAN to route IP packets between the isolated VLANs whereas routers are used to route IP packets between LANs across the wide area network (WAN) [3].

Virtual Local Area Network (VLAN) [101] is used to divide or partition a physical Ethernet switch into smaller logical switches, as well as creating logical switches that extends over more than one physical switch [100], [110].

Inter-VLAN routing, a configurable feature supported on layer-3 switches, is required whenever hosts in one LAN needs to communicate to hosts in another LAN [100], [146].

2.2.2. Addressing and transmission layers

The addressing and transmission OSI layers are fundamental to the working of the IP network. In this section we discuss how hosts are addressed at the networking layer, the logical partitioning of the network in terms of network addresses and the controlling mechanism behind the flow of data between hosts. We also look at the mechanisms employed to route data between different networks and how data can be filtered and selectively forwarded.

2.2.2.1. IP

The Internet Protocol (IP) addressing scheme has been standardised and adopted through the RFC processes [14]. Two major versions of the IP protocol have been developed - IPv4 and IPv6 [102]. IPv4 is the most common IP system with IPv6 the future migration path. The main difference between IPv4 and IPv6 is that IPv4 network addresses are 32 bits and IPv6 are 128 bits allowing for a much larger address space. IPv6 also offers some improvements in built-in support for security, QoS and multicasting [96]. Most modern operating system network stacks will support both IPv4 and IPv6 addresses. IPv4 addresses are written as 4 consecutive bytes expressed as decimal numbers, for example 192.168.0.15.

The host destination and source IP addresses are included in the IP protocol header. The IP addressing scheme is hierarchical, it consists of a network ID and a host ID. The subnet mask consists of a series of consecutive 1s followed by 0s, for example 255.255.255.0 and is used to differentiate between network and host ID parts by performing a binary operation. The bit-wise logical AND performed with the IP address and the subnet mask yields the network ID, the remainder is the host ID [96].

The TTL (Time To Live) field in the IP protocol packet is an important field that is decremented every time an IP packet crosses a router (hop) as the IP packet is routed across a WAN. The TTL

field prevents an IP packet from being routed indefinitely in the event of router misconfiguration resulting in a routing loop [99].

2.2.2.2. TCP/UDP

The Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) are both part of the data flow control layer of the IP network stack. TCP provides a connection-orientated flow control while UDP is connectionless. The TCP/UDP service port number, together with the host's IP address are referred to as a socket or the end point of a connection. The TCP/UDP socket is the application layer interface into the network. TCP is designed on the principles of end-to-end delivery and robustness. TCP has dynamic flow control and error checking mechanisms built into the protocol in order to ensure reliable packet delivery. TCP uses a three-way handshake to initiate a session. A TCP server first binds to a port where it idles, waiting for client connections in the passive open or LISTEN state. A client can connect to the listening socket by opening a client-side socket. A three-way handshake now occurs during the TCP session that progresses through the following states [100], [3]:

- SYN state: An active open is performed when the client sends a SYN packet to the server.
- SYN-ACK state: The server replies with a SYN-ACK packet. The acknowledgement number is set to the received sequence number plus one and the sequence number that the server chooses for the packet is a random number.
- ACK state: The client now sends an ACK packet back to the server. The sequence number is set to the received acknowledgement number and the acknowledgement number is set to one more than the received sequence number.

TCP is used in most applications where reliable data transfer is important, it is a reliable protocol but due to the robust design can also behave very sluggish because of the data flow and verification overhead. TCP relies on a number of data flow controlling functions including data packet ordering, error detection with retransmissions of lost or corrupted packets, flow control to guarantee reliable data delivery and congestion control. UDP, on the other hand, have no built-in hand-shaking, flow control, error detection or congestion control and is therefore often used for time critical applications where data error correction is not as important as real time delivery. Typical UDP based applications are VOIP, video and some real time control applications [96], [3].

2.2.3. Packet routing, filtering and QoS

This section will focus on the mechanisms employed to manipulate and redirect data packets on and between IP networks. IP routing is the mechanism employed to route packets between different IP networks across the WAN. Packet filters are employed on and between IP networks to filter and control the flow of packets. Both these mechanisms can have a major influence on network reliability. We also discuss the Quality of Service (QoS) features that can control the flow of data packets and are embedded on the IP and physical layers.

2.2.3.1. IP Routing

Switches direct packets based on layer-2 addresses (MAC addresses) in the physical domain inside the LAN, but layer-2 addresses are not visible beyond the LAN. Layer-3 routing between different LANs is based on the destination IP addresses as contained in the IP protocol header. Although every host's network stack contains a routing table, special nodes called routers found on the interconnecting borders of networks are dedicated to the task of routing IP packets between

the various networks. Routers work with the logical or layer-3 addresses, this makes it possible for routers to connect different types of layer-2 networks. A good example is a router connecting the Ethernet LAN with an Internet service provider (ISP) via an ADSL link. Routers therefore typically have more than one network interface [96], [100].

A router's routing table may be configured manually (static routing) or dynamically through the deployment of dynamic routing protocols. Routers may be used inside Autonomous Systems (AS) [103], that is in IP networks that implement the same policy and/or are managed by the same authority. Routing protocols deployed inside the AS are referred to as Interior Gateway Protocols (IGP) and routing protocols deployed between ASes are collectively referred to as Exterior Gateway Protocols (EGP). A distinction is also made between "stub networks", ASes that are located on the edge of the WAN and ASes that are used as "transit networks" between ASes [3], [100], [98], [54].

Static routing is suitable for small networks with no alternative routes and are typically deployed on stub networks connected to the WAN through a single router. Static routing is simple to configure and to maintain and eliminates the need for a dynamic routing protocol. The routing table entries consists of all the possible routes, the destination IP network, associated gateway or next hop router, the appropriate network interface to send the packet out and a metric which is an indication of the "cost" associated with the particular route [3], [100], [54].

Dynamic routing protocols depicted in Figure 2.3 such as RIP [104] and OSPF [105] are deployed inside larger ASes consisting of multiple interconnected routers. RIP is a simple routing protocol that makes use of the distance vector, that is the number of interconnection routers to the destination or "hop count" to decide the most optimal route. Routing tables are periodically advertised or broadcast on the network for the updating of the other routers' routing tables. RIP has become obsolete due to some limitations, such as slow convergence after network changes and the inability to include the link quality and cost into calculating the routing metric [3], [100], [98], [54].

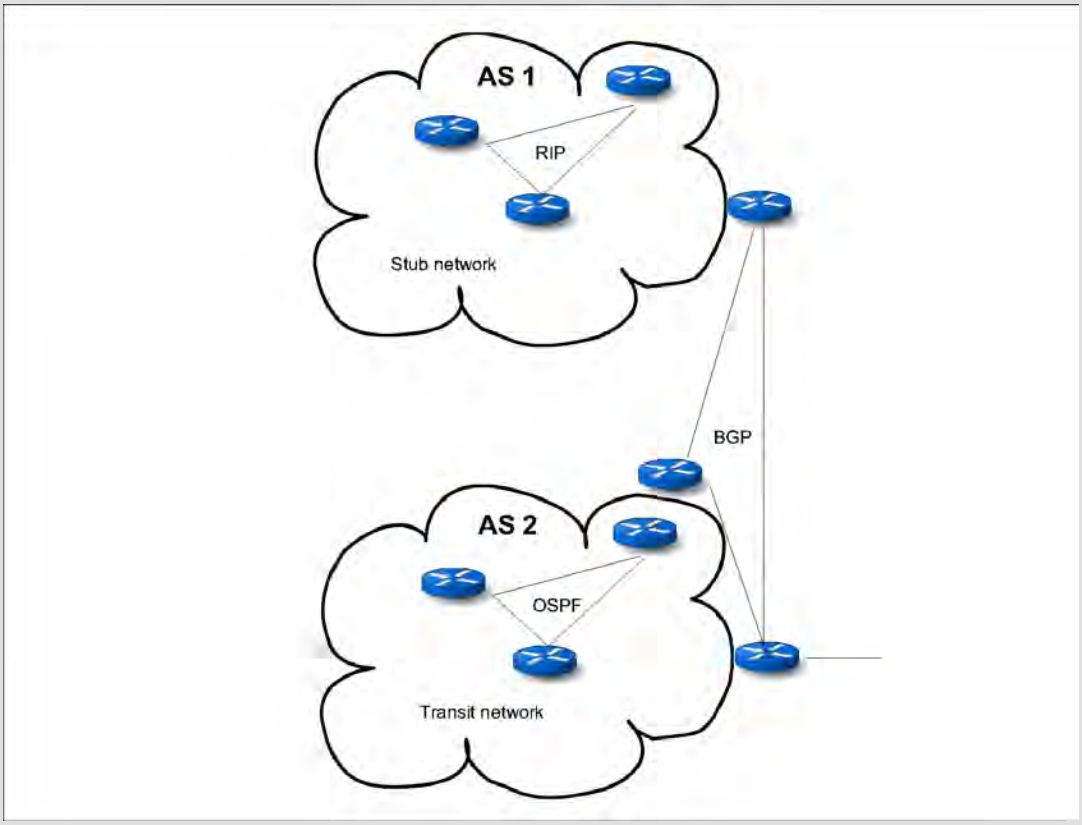

Figure 2.3. Dynamic routing protocols

Open Shortest Path First (OSPF) overcomes some of the shortcomings of RIP and is known as a link state routing protocol. OSPF builds up a connectivity graph which is transformed using Dijkstra's algorithm for making routing decisions. The cost factors used in the routing metric includes the distance to the next hop (round trip time), link bandwidth and link availability. Border Gateway Protocol (BGP) [106] is a backbone protocol responsible for core routing decisions on the Internet and other large WANs. BGP is based on path vectors and is a widely used EGP.

Every host on an IP network that needs to connect to a host on a different IP network must be assigned a "default gateway" or router address that is on the same IP network. Virtual Router Redundancy Protocol (VRRP) [89] makes it possible to assign a virtual router or gateway address to a host thus eliminating the single point of failure. The virtual router is an abstraction of multiple routers, master and backup routers, working together as a group thus providing router or default gateway redundancy on the IP network.

2.2.3.2. Packet filters

Routers switch IP packets between different network interfaces based on a set of rules contained in routing tables. Similarly a packet filter (or firewall) filters IP packets based on a set of rules contained in a filter rule table. Packet filters are usually deployed on a network border to filter and control data flow between two different network zones, but they can also be deployed on individual computer workstations and are then referred to as a "personal firewalls" [96] , [100].

One of the most common packet filters is the Netfilter framework that is found in the modern Linux kernel. Netfilter offers a good example of how a firewall works. The interface program to the Netfilter framework is known as iptables and serves as a configuration front-end to a set

of tables that includes a "filter", a "nat" and a "mangle" table. Every table consists of chains i.e. named groups of rules with the default chains predefined [97], [54].

The "nat" and "mangle" tables specify how Netfilter changes the content of IP packets to do, for example, network address translation (NAT) manipulation where source and destination addresses of IP packets may be modified. The "filter" table allows filtering on MAC address, source and destination IP addresses, and TCP/UDP service port numbers. Transport layer information including the TCP state of connections can also be specified in the tables. The actions to be performed are specified using policy chain rules where, chains can be configured to perform the following typical actions [97], [54]:

- ACCEPT - packets are accepted and allowed to pass through the filter.
- DROP - packets are dropped silently, no feedback is given to the sender.
- REJECT - packets are dropped with notification to the sender.
- LOG - packets are logged without any other intervention for recording and auditing purposes.

As can be seen from the above packet filters can influence network reliability by causing the loss of data packets or other forms of unexpected behaviour.

2.2.3.3. Quality of Service

IP network performance issues related to reliable and predictable packet delivery, or the Quality of Service (QoS) [56], [87] is a major contributor to the overall reliability of an IP network. In the network layer-2, QoS is supported inside the MAC frame header by priority marking within a VLAN tag as prescribed by VLAN IEEE 802.1Q and IEEE 802.1p [101]. QoS is implemented in the network layer-3 with the commonly used “DiffServ” or differentiated services [87] model. IP packets are marked according to the level of service to be allocated to those packets, Differentiated Services Code Point (DSCP) markings are used in IP packet headers. Switches that support IEEE 802.Q/p and routers that support DiffServ use queuing techniques to prioritise packets according to the QoS tag. QoS support is critical for real time applications such as VOIP and video streaming that are sensitive to latency, jitter and packet loss [3], [96], [107].

Some bandwidth is usually reserved by default for network control packets such as ICMP and broadcast traffic that is critical for the correct functioning of protocols like IP, DHCP and OSPF. Under conditions of high congestion non-prioritised traffic can subsequently be discarded. QoS is often used with other bandwidth managing mechanisms such as traffic shaping (rate limiting) and optimised TCP flow window control [96], [107].

2.2.4. Network address support services

This section will discuss the various mechanisms employed to resolve and configure IP addresses. Since the IP address is the unique identifier of every host on the IP network and is a configured entity there are specialised systems in an IP network that support the automatic mapping and configuration of IP addresses. Two important services that will be discussed are the DHCP and DNS services indicated in Figure 2.1.

2.2.4.1. ARP

It is clear from the above discussion in Section 2.2.1 and Section 2.2.2.1 that MAC addresses are used at the physical layer and that IP addresses are used at the networking layer and that there must be a mechanism for mapping IP addresses to MAC addresses. This is achieved through the

Address Resolution Protocol (ARP) [108] that is an integral part of the way IP networks operate. ARP sends out broadcast packets when a host needs to communicate with another host and the destination IP address is known, but the destination MAC address required for the Ethernet/Wifi frame is unknown. ARP broadcasts are limited to the physical domain or the LAN. As discussed under Section 2.2.3.1 layer-2 addresses are not visible beyond the LAN and layer-3 routing is performed by the router based on the destination IP addresses [94], [96].

2.2.4.2. DHCP

Dynamic Host Configuration Protocol (DHCP) [83] is an automatic IP configuration service. IP addresses can either be configured manually or can be automatically allocated by a centralised DHCP server. The DHCP client running on individual hosts that require (usually on start-up) an IP address broadcast DHCP discovery requests probing for a DHCP service on the network. The DHCP server maintains a database of allocated IP addresses and then responds to the DHCP discovery request with a DHCP offer - a unique IP address and other configuration information including the subnet mask, default gateway and DNS server IP addresses. DHCP servers can be configured to allocate a random IP address from a pool of IP addresses or to map the same IP address to the same MAC address every time. DHCP clients can also inform DNS servers of their new automatically issued IP addresses. A redundant configuration of DHCP servers is possible, since DHCP clients can respond to multiple DHCP offers by broadcasting a DHCP request for the selected offer and thus informing other DHCP servers of the offer that was accepted by the client so that the servers can keep track of IP addresses that are used and those that are still available [3], [96], [54].

2.2.4.3. DNS

Domain Name System (DNS) [84] is a system whereby domain names and host names are allocated at a human-friendly level. IP addresses are dynamically looked up in a central directory type services. Host names can also be statically allocated using a local "host file" which must be manually configured and stored on every individual host. Only IP addresses are used in the IP network stack, domain names of hosts must therefore first be resolved to the associated IP address. Reverse DNS lookups maps IP addresses to domain names and are used for logging and debugging purposes [3], [96], [109], [54]. The TCP protocol is used for zone transfers between DNS servers. A DNS query consist of a single UDP request from the client followed by a UDP reply from the server. Typically a primary and backup DNS server IP can be specified for every host making a redundant DNS server configuration possible [109].

2.3. Existing reliability models for IP networks

Reliability is formally defined as *"the ability of a system or component to perform its required functions* ***under stated conditions*** *for a specified period of time"* [75].

Mathematically, reliability is expressed as the probability that a system, unit or component continues to perform its intended function during the interval *(0,t)* under stated conditions. The reliability function *R(t)* [77], [120], is related to the cumulative distribution function (cdf) of the system lifetime *F(t)* (or the probability of a unit failing by time *t* and depicted in Figure 2.4 by:

$$R(t) = P(x > t) = 1 - F(t) \tag{2.1}$$

where $F(t) = \int_0^t f(x)\,\mathrm{d}x$ and where the random variable x is the time to failure and $f(x)$ is the probability density function (pdf).

From the above it follows that:

$$R(t) = \int_t^{\infty} f(x)\,\mathrm{d}x \tag{2.2}$$

and:

$$f(t) = -\frac{\mathrm{d}}{\mathrm{d}t}(R(t)) \tag{2.3}$$

The hazard or failure rate function, that is the number of failures occurring per unit time, is given by:

$$\lambda(t) = \frac{\mathrm{f(t)}}{1-\int_0^t f(x)\,\mathrm{d}x} \tag{2.4}$$

$$= \frac{\mathrm{f(t)}}{\mathrm{R(t)}} \tag{2.5}$$

and the mean time to failure (*MTTF*) is the defined as:

$$\mathrm{MTTF} = E[x] = \int_0^{\infty} t f(x)\,\mathrm{d}x \tag{2.6}$$

The instantaneous or point availability *A(t)* is similar to the reliability function *R(t)* in that it represents the probability that a system will be up and running at time *t*, however the point availability incorporates maintenance or repair information [77], [78] and is given by:

$$\mathrm{A}(t) = \mathrm{R}(t) + \int_0^t \mathrm{R}(t-\mathrm{u})\,\mathrm{m}(\mathrm{u})\,d\mathrm{u} \tag{2.7}$$

where *u* represents the last repair time and $0 < u < t$ and *m(u)* represent the renewal density function.

The average availability at time *t* is defined as *Av(t)*:

$$\mathrm{Av}(t) = \frac{1}{t}\int_0^t \mathrm{A}(\mathrm{u})\,d\mathrm{u} \tag{2.8}$$

and the steady-state availability as *A*:

$$\mathrm{A}(\infty) = \lim_{t\to\infty} \mathrm{A(t)} \tag{2.9}$$

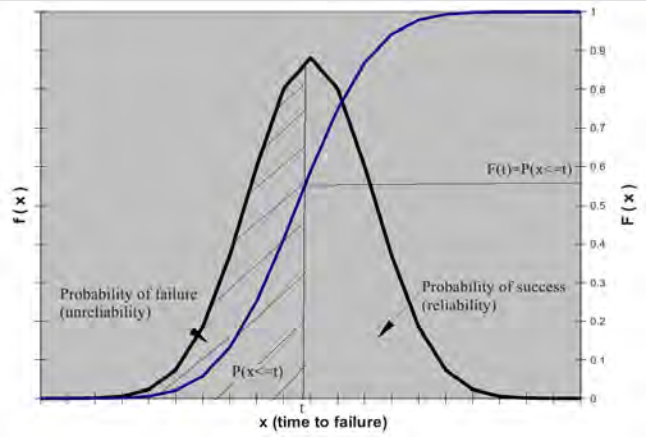

Figure 2.4. Failure distribution function

Steady-state availability can also expressed as:

$$A = \frac{MTTF}{MTTF + MTTR} \tag{2.10}$$

Where *MTTF* is the meantime to failure and *MTTR* is the mean time to repair. In practice point availability converges to the steady-state availability after a period of approximately four times the *MTTF*. Steady state availability can also be viewed as the operational availability or the ratio of the total time the system was functioning to the operating cycle i.e. the overall time period of operation being investigated and can be expressed as:

$$A = \frac{Uptime}{Operating\ cycle} \tag{2.11}$$

From the above equation, availability is expressed as a fraction that is commonly approximated and expressed in terms of the amount of most significant repeated "nines" in the fraction. For example, a five-nines availability indicates an availability fraction of $A = 0.99999$ and can be directly related to an uptime of 8765.91234 hours in a year or alternatively interpreted as 0.08766 hours (5 minutes) of downtime in a year.

The general shape of the failure rate function in Equation 2.5 is described as the "bathtub curve" [77], [80] and is depicted in Figure 2.5. Failure rates decrease during the infant mortality period and increase when equipment reach the end of lifetime period, however during the normal operational lifetime time of a component or unit the failure rate is approximated as being constant.

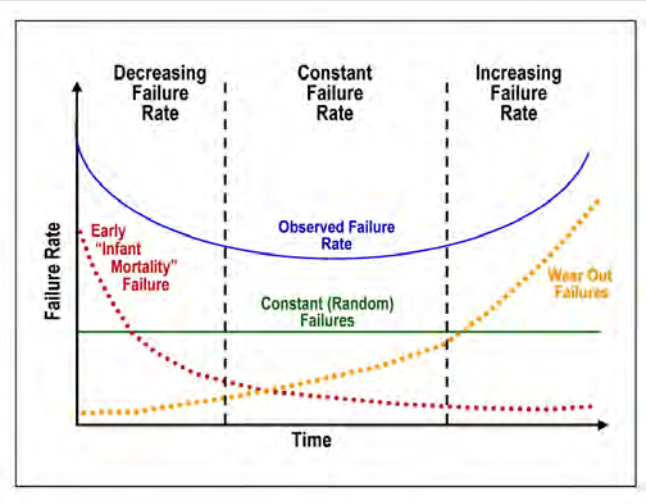

Figure 2.5. Bathtub curve

Given a constant failure rate λ the reliability function *R(t)* Equation 2.2 becomes:

$$R(t) = e^{-\lambda t} \tag{2.12}$$

where the mean time to failure (*MTTF*) is:

$$MTTF = \frac{1}{\lambda} \tag{2.13}$$

As the amount of mission critical applications that are deployed on IP networks increase network reliability becomes increasingly important. The accurate reliability modelling of telecommunication networks has become paramount because of increased capacity and accompanying application of network techniques to improve the quality of service and network reliability. These reliability models can be very complex because network elements can fail randomly. A multilayer reliability modelling approach is presented in [57]. The results indicate that a generic modelling structure can accommodate both the multilayer structure as well as the multi-layered reliability protection techniques deployed. Efficient lower and upper bounds for performance indices are also derived. In [58] a two state reliability model is used to represent failure in each layer of network elements. Real values from network planning outputs tools are assigned to the model that combines state space reduction techniques and stratified sampling. Where the reliability function in Equation 2.12 refers to the hardware failure probability of a single device or unit during its normal operational lifetime, the aggregated multi-layered network reliability *E* can be calculated according to the approach in [141], [143] from every performance index function *Perf(s)* for every possible defined failure state *s* where:

$$E = \sum_{\forall S} Perf(s) \cdot P(s) \tag{2.14}$$

and the multi-layered network availability function *E(g)* is defined as:

$$E(g) = \sum_{\forall S:Perf(s) \leq g} P(s) \tag{2.15}$$

A stratified sampling technique used to reduce the large state-space problem that arises as the modelled network gets bigger and/or more complex is also presented in [142] and is further developed in [141].

The modern IP network is a complex system and requires various sub-systems and units to work together in ensuring an overall reliable system. A typical IP network consists of units providing network services at the following OSI layers [54]:

Table 2.1. Network services required in a reliable IP network

OSI layer	**Service description**
Application	DHCP and DNS
Transport (TCP/UDP)	Packet filtering
Network (IP)	Routing
Network Access (Data Link and Physical)	Ethernet and WLAN

The reliable functioning of the above services are configuration sensitive, configuration errors made during standard maintenance activities can play a major role in reducing the network reliability. Some of these services are hosted on shared units and are therefore sensitive to network topology. Such shared services are also known as "common points of failure" and redundancy is often employed to improve reliability [54].

A lot of research is done on traffic engineering focusing on performance metrics like throughput and maximum utilisation, but the reliability as perceived by users remains to be properly defined. Typical IP network reliability figures reported range from 99% to 99.99% but it is not clear how this is calculated. [128] proposes a model and methodology to define IP network reliability using service level agreements and then evaluate the various factors that have an influence on reliability. These factors include:

- IP network topology: the physical arrangement of routers and links between routers.
- Traffic demands: the network utilisation or loading.
- QoS metrics: connectivity, end-to-end delay or latency and end-to-end packet loss.
- Service level agreement: user expectations and criteria.

Using failure characteristics as proposed in [130] for an IP backbone, a failure model is constructed and used to generate synthetic failure events in order to stress test the network. These failure characteristics include maintenance activities, simultaneous (common point) failures, overlapping or dependent failures and individual link or isolated failures. Different algorithms used for IP restoration or protection techniques, i.e. the built-in routing protocols interconnecting routers that detect broken links and then reroute traffic through a different link, are evaluated. It is found that IP layer reliability techniques improve network reliability but that not all the reliability factors can be improved at the same time because different IP restoration techniques have an impact on different factors of reliability [128].

[59] introduces two categories of failure in IP networks and puts them in the context of the purpose and requirements of the application, for example the minimum acceptable network delay or latency for a video stream may be 400ms and for a file transmission may be 60s. As is the case with hardware failures two basic categories of failure are defined:

- Function failure: referring to the loss of connectivity, for example the loss of a link.
- Parameter failure: deviation from the acceptable range of performance parameters such as delay, packet loss, and transmission errors respectively measured as delay threshold, Packet Loss Ratio (PLR) threshold, and Bit Error Rate (BER) threshold.

2.4. Network reliability modelling techniques

Electronic hardware component reliability is a well understood technical field of study and has been the topic of much academic and practical research. Reliability specification and testing has been standardised in the form of specifications and standards by many international bodies such as International Organisation for Standardisation (ISO), International Electrotechnical Commission (IEC), European Committee for Electrotechnical Standardization (CENELEC), Institute of Electrical and Electronic Engineers (IEEE), Telcordia Technologies (Bellcore) and US Department of Defense Military standards, handbooks and specifications (MIL STDs, MIL HDBKs, MIL SPECs) [124].

Network performance can be analysed using two basic techniques: analytical modelling and computer simulation [113]. A good example of using OPNET to model and simulate an IEEE802.1x network is presented in [73]. System reliability and availability is usually analysed using analytical models including combinatorial models, state-space models or hierarchical models. Combinatorial models include reliability block diagrams, reliability graphs and fault trees. These combinatorial mathematical modelling techniques are adequate for modelling reliability in systems where the failure events in systems and units are independent from each other, but cannot accurately deal with inter-system and inter-unit dependencies.

[36] uses fault trees to model the reliability of different Ethernet topologies in electrical substation control and monitoring networks. Approximated mean time to failure (*MTTF*) figures are used for representing the reliability of Ethernet hubs, switches, routers and NICs. The fault tree model is then evaluated for different network configurations featuring a shared hub versus a shared switch and redundant switch and hub configurations.

The approach in [36] may be satisfactory for simplified reliability comparisons but cannot be used to accurately model IP network reliability because of network unit and system dependencies that cannot be modelled using fault tree analyses. Failure dependencies in real system include correlated failures, repair dependencies and fault-tolerance. State-space models must be used to capture such dependencies. Markov models, also known as Markov chains, are one of the most commonly used state-space modelling techniques to model gracefully degradable systems [117], [119].

The Markov chain is a random process that undergoes state transitions in a chainlike manner. The Markov chain is stochastic and "memoryless", meaning it exhibits the Markov property that implies that the next state only depends on the current state and not on any previous states or system history. The semi-Markov model is often used for the construction of reliability models. The semi-Markov process is similar to a pure Markov process, however the transition times between states are random variables with probability distribution. The transition rate in a particular state depends on the time spent in that state but is independent of the path traversed to get to that state. The Markov chain model in Figure 2.6 is represented as a directed graph or state transition bubble diagram with the failure rates λ and the repair rates μ interconnecting the bubbles or the different states. A semi-Markov process where all the holding times are exponentially distributed (a Poisson process) is called a continuous time Markov chain (CTMC) [120], [121], [119].

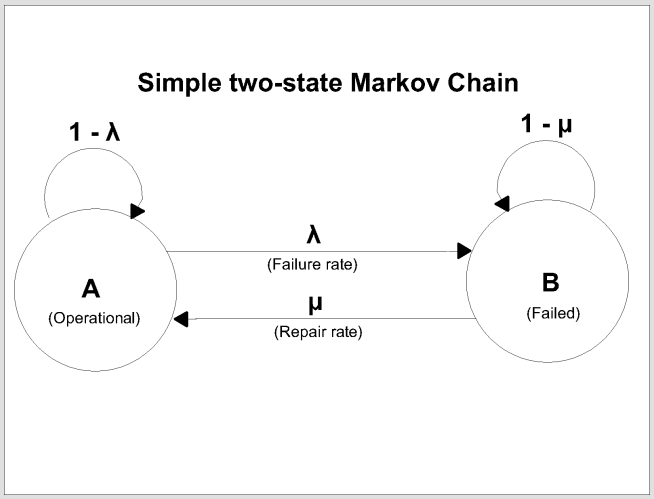

Figure 2.6. Two state Markov reliability chain

Figure 2.7 depicts an absorbing Markov chain model, with S3 being an absorbing state, the system cannot move out of state S3 representing a system failure. The state transition diagram is used to derive the generator or one-step transition matrix *G* [122], [123] that can be transformed into its canonical form with:

- *I* the identity matrix representing all the absorbing states;
- *O* the zero matrix;
- *R* is the non-absorbing to absorbing transition matrix;
- *Q* is the non-absorbing to non-absorbing transition matrix.

The fundamental matrix *F* with elements f_{ij} is calculated from *Q* as indicated in the figure and represents the average time spent in state *j* given that the system started in state *i*. The *MTTF* can be calculated from fundamental matrix *F* where:

$$MTTF = f_{i1} + f_{i2} + ... + f_{ij} \tag{2.16}$$

In the example provided the system starts in state S1 therefore *i = 1*.

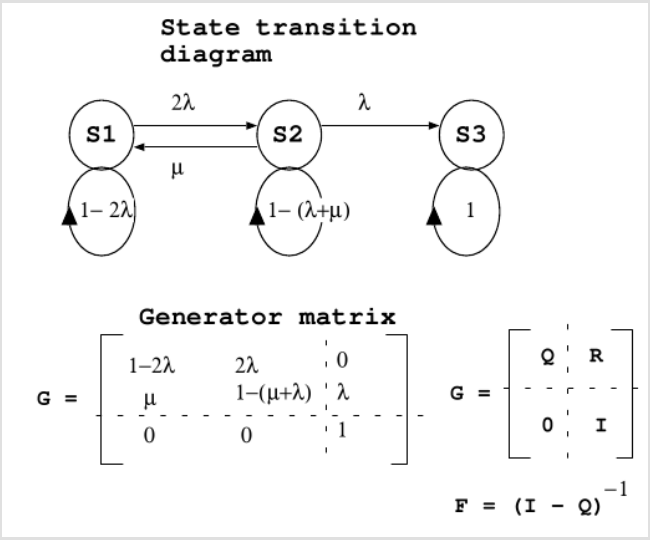

Figure 2.7. Absorbing Markov chain with generator matrix

Hierarchical models combine combinatorial models with state-space modelling techniques. The parts of the system that are independent are modelled using combinatorial techniques , for example reliability block diagrams. The parts of the system with dependencies are modelled using state-space techniques resulting in a simplified model with a reduced state-space requirement [119], [126], [140]. Hierarchical block diagram models can be used when certain assumptions apply [127] summarised here:

1. The system is series-parallel (S-P) with a single input node and single output node.
2. The system is composed only of repairable, perfectly monitored assemblies. The monitored state is always accurate and the repair process does not damage the system.
3. The failure and repair processes of all components, units and systems are mutually independent.
4. Every component, unit and system can assume one of two states - working or failed.
5. The transition rates of failure (λ) and repair (μ) processes are constant.
6. The product of the failure rate (λ) and the down time is much less than unity, typically less than 0.1.
7. No simultaneous failures can occur.
8. The system is working and all components and units are working in the initial state.
9. The reliability (R) of the system is close enough to unity for some approximations to be valid.

Simplified equations for determining equivalent *MTTFs* for series and parallel connected reliability block diagrams are derived in [127], [31], [80]. The equivalent *MTTF* for serial connected components or units can be simplified by adding up individual failure rates:

$$MTTF_{ser} = \frac{1}{\lambda_1 + \lambda_2 .. \lambda_n} \tag{2.17}$$

and the equivalent *MTTF* for parallel components or units can be calculated from:

$$MTTF_{par} = \frac{1}{\lambda_1} + \frac{1}{\lambda_2} - \frac{1}{\lambda_1 + \lambda_2} \quad (2.18)$$

Figure 2.8 depicts different parallel redundant configurations and compares *MTTFs* calculated using Markov chains for "hot" and "cold" standby systems with and without repairs [123]. In a hot-standby configuration the parallel units are both operational and therefore shares the work load at all times. In a cold-standby configuration one unit is not operational and can therefore not fail since it is not carrying any work load, when the operational unit fails the standby unit is selected through an automatic switch mechanism. A hot-standby configuration generally exhibits lower *MTTF* than the equivalent cold-standby configuration. Block diagrams c) and d) illustrates the effect of repairs on the *MTTF* of a redundant system and how the *MTTF* increases dramatically when repairs are performed where the failure rate λ is much smaller than the repair rate μ. The following *MTTF* formulas (assuming $\lambda << \mu$) for parallel repaired units are summarised for reference:

$$MTTF\ (\text{hot standby}) = \frac{\mu}{2\lambda^2} \quad (2.19)$$

$$MTTF\ (\text{cold standby}) = \frac{\mu}{\lambda^2} \quad (2.20)$$

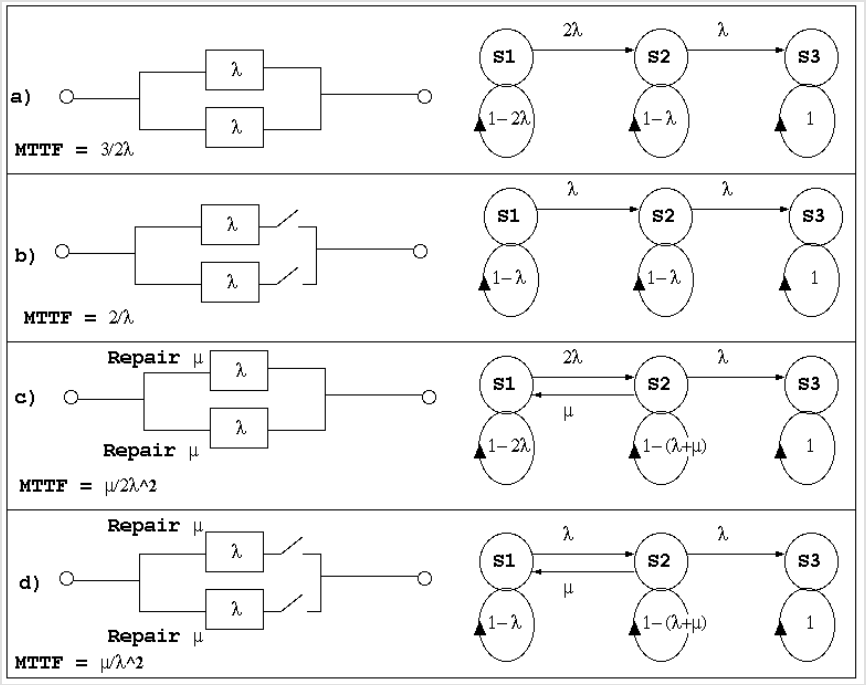

Figure 2.8. Hot/cold standby MTTFs

A hierarchical model is used in [125] to compare the physical layer reliability of Multifunction Vehicle Bus (MVB) with Ethernet in train-born communication networks. The purpose of the

study is to compare service reliability with "asset reliability", or mean time to failure (*MTTF*) and mean active corrective maintenance time (MACMT). The reliability parameters of the end units are calculated using reliability block diagrams. The network reliability parameters are calculated using a Markov model and the overall reliability parameters are calculated using the hierarchical block model.

A two-state Markov Modulated Poisson Process (MMPP) is used to model the effect of bursty IP traffic on hub based Ethernet LAN in comparison to Token Ring LAN in [144]. The MMPP is a Markovian arrival process used in queuing theory where Poisson processes are switched between by an underlying Markov process.

Markov chains have been used to model the operation of the TCP protocol by various authors. [116] presents a Markov model to analyse TCP flow in multihop wireless networks using the absorbing Markov chain where the Discrete Time Markov Chain (DTMC) finally stops at one of the absorbing states. [148] explores the impact of TCP flow control mechanisms on VOIP and video streaming using a Markow chain TCP delay model. Real time delivery over TCP is found to be enhanced by parallel connections.

A major problem with using Markov chains is the large state-space required for modelling complex systems [114], [117], [121]. Stochastic Petri nets can be used to automatically generate the underlying Markov chains providing a high level abstraction for the specification of the underling Markov model. A Petri net is a mathematical modelling language, consisting out of directed, bipartite graphs, that can be used to describe distributed systems. Stochastic Reward Nets (SRNs) can be used to evaluate the reliability of complex systems. SRNs are special cases of stochastic Petri nets (SPNs) that have been augmented with the ability to specify outputs as reward-based functions. The SRN is solved by generating and analysing the underlying Markov reward model [117]. A performance evaluation process algebra language (PEPA) based on compositional models and used to generate the continuous time Markov process is introduced in [118].

TCP flow has been modelled using SRN. In [114], for example, the focus is on the TCP Reno variant that is used in most operating systems. The performance of TCP Reno is investigated by analysing the following TCP flow control mechanisms: congestion window evolution, slow start and congestion avoidance, packet transmissions, packet loss management due to time-outs and triple duplicate acknowledgements. Performance metrics such as throughput and delay obtained through the SRN model are then used to demonstrate the unfairness between downstream and upstream data flow at the wireless Access Point (AP). The proposed model is then evaluated through simulation [114]. TCP flow has also been modelled using Markov processes that evolve deterministically and are therefore suitable to be modelled using Poison-driven stochastic differential equations. The discreet time Markovian process is modelled using a state-dependent time-rescaling technique [115].

2.5. Performability as an enhanced reliability metric

Performability is a measure of interest that combines both reliability and performance into a single metric. Performability analysis captures the interaction of the failure-repair or availability of system with the effect of the resulting availability on the system performance [60], [61], [62], [63], [64], [65], [66]. Various Petri net techniques can be used for performability modelling, but the most popular technique is making use of the Markov reward model (MRM) [60] based on the Markov chain discussed in Section 2.4. The Markov reward model consists of a behaviour model and a rewards structure [60], [61], [62]. The Markov state or generator matrix that is used to

analyse the failure-repair state transitions in the system can be used as the behaviour model. This behaviour model is the same state matrix that is used to calculate the system availability for the purpose of reliability modelling, however - in a reliability model each state is assigned a "1" or "0" binary reward level that indicates whether a system is operational or failed. The reward structure for the performability model assigns a relative performance measure that can assume any value corresponding to a specific state of operation. The reward structure is represented as a reward matrix that can be multiplied with the generator matrix in order to calculate the accumulative performance reward in Figure 2.9. The state function *Z(t)* indicates the random amount of time that the system spends in each state S1, S2, or S3. *X(t)* indicates the reward states associated with *Z(t)*, i.e. each state is rewarded with performance measure associated with a state. *Y(t)* is the accumulated award at time *t* calculated from the area under *X(t)*. Performability can then be calculated as the average accumulated award over a period of time [67], [66]:

$$\mathrm{Yv}(t) = \frac{1}{t}\int_0^t \mathrm{Y}(\mathrm{u})\,d\mathrm{u} \tag{2.21}$$

and the steady-state formability as *Y* in Equation 2.22:

$$\mathrm{Y}(\infty) = \lim_{t\to\infty} \mathrm{Yv}(t) \tag{2.22}$$

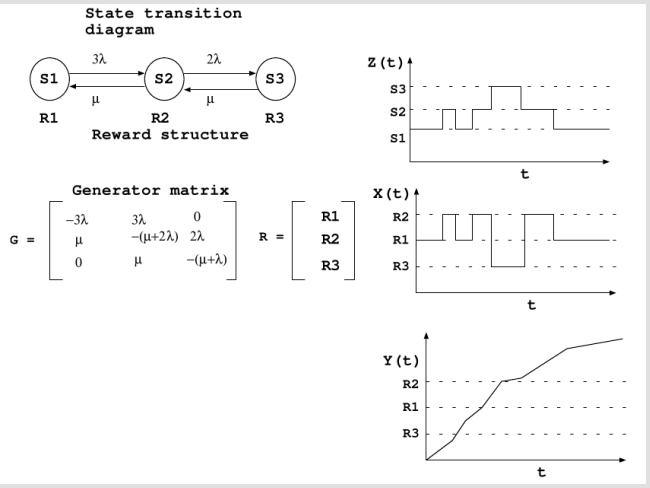

Figure 2.9. Performability

[63] develops a performability model for analysing the effect of access point (AP) failures in Wifi networks by using a Markov chain and assigning a penalty to every network state. The penalty is calculated as a weighted combination of factors such as number of data bursts lost, vehicle speed and the failure position in the network. A Markov chain based performability model for evaluating fault-tolerant microprocessor systems is presented in [66]. The authors point out that the use of steady-state performability values can be misleading since it tends to average out performance peaks over time and should never be seen as fully representative of a system's dynamic performance, a conclusion that is also drawn in [67]. In [68] a performability model is developed to assess the influence of redundancy in the transmission media on the reliability and formability of a ring versus a double ring topology fibre optic LAN. The mean number of frames sent through the network is used as the performability measure.

2.6. Physical IP network topology

2.6.1. General background on topological parameters determining network reliability

Traditionally network reliability studies have utilised graph theory to model networks as graphs consisting of a set of nodes (vertices) interconnected by a set of edges or links. Accordingly the network is modelled as a linear graph $G = (N,L)$, where N represents the non-empty set of n computing nodes and L is a set of full or half-duplex l links that constitute the communication links between the nodes [19], [20], [22], [23], [26], [21], [27], [25]. Network reliability is then analysed by determining the node and link connectivity using the properties of the graph and a range of reliability measures of interest including the cohesion and connectivity between nodes in the graph. A range of reliability evaluation techniques and algorithms for small, moderate and large networks is summarised in [18] and [17]. As with all large and complex systems the amount of states that must be analysed increases exponentially as the number of components in the system increases which results in computational difficulties in terms of computing time and computational resources. Various techniques and algorithms for graph simplification is therefore utilised to reduce the large amount of states by approximation.

Typically graph theory methods use as reference one of the following reliability measures to approximate the reliability of the network [26], [23], [20]:

- k-terminal: the probability that there is a connection path between at least k defined nodes;
- s-t (2-terminal): the probability that there is a connection path between a defined start and termination node;
- all-terminal: the probability that all the nodes are connected or
- all-operating terminal: the probability that all operating nodes are connected

2-terminal availability is used in reliability studies where the data traffic flows primarily between source and a single destination node. Generally all these graph theory methods assume that network connectivity is the only factor that determines network reliability therefore neglecting congestion, latency and channel capacity [23].

It is desirable for the graph of a network to have a reasonably small diameter [19] as processing and queuing delays increase as the path between two nodes must pass through many intermediate computing nodes. A reliability measure based on network diameter is introduced in terms of the minimum amount of nodes or links that must fail in order for the diameter of the network to exceed a specified value. [19] also pointed out that in order to achieve high network reliability, design procedures concentrate on the "duplication of communication facilities" (redundancy) sometimes at considerable additional costs. It noted that most network reliability studies based on graph modelling assumed that all network and node elements fail independently and that it was impossible for nodes and links to fail simultaneously. [20] compares the reliability measures of ring and counter-rotational ring networks with active and passive star networks - distinguishing between branch or link failures and vertex or node failures and deriving mathematical equations for each type of reliability measure as a function of the network size or amount of nodes (N), the amount of redundant rings or star links (m) and the probability of link or node failure (p). Useful equations for counter-rotational ring and active star networks are listed here for reference [20].

Averaged bidirectional 2-terminal reliability of a counter-rotational ring network as a function of link failure probability p and the amount of nodes N is given by Equation 2.23:

$$R(p,m,N) = \frac{2m^2}{N-1}(1-p)^2 \quad (2.23)$$

Averaged bidirectional 2-terminal reliability of an active star network as a function of link failure probability *p* and the amount of nodes *N* is given by Equation 2.24:

$$R(p,m) = m^2(1-p)^2 \quad (2.24)$$

In general, link availability can be expressed and calculated as a function of failure rate expressed in failures per unit time per unit length for example in [26] expressed as Equation 2.25:

$$Ass = \frac{\mu}{\mu + \lambda * L/100} \quad (2.25)$$

Where μ is the repair rate per hour, λ is the failure rate per hour per 100 km and *L* is the link length in kilometres.

[21] provides a synopsis of combinatorial methods to simplify graph presentations of networks and applies fault trees with minimal spanning tree analysis to compare the topology dependant reliability of double-ring versus braided networks. It is noted that combinatorial methods cannot capture the dynamic features of a networked system i.e. fault latency and path-regrowth time. The minimal spanning tree analyses is based on the assumption that the probability of network failure is equivalent to the probability that no minimal spanning tree exists - that as long as at least one subset of links that allows all nodes in the graph to be connected (minimal spanning tree) remains operational. The number of minimal spanning trees can be very large, even for networks with as few as ten nodes introducing the familiar state or event "space explosion" associated with graph methods and resulting in computational difficulties. The study found that the two-braid network is more reliable than the double ring and the following generalised observations:

- as the probability of link failure decreased by one order of magnitude the probability of system failure decreased by four orders of magnitude;
- increasing the number of nodes decreased the reliability of the network.

[22] notes that a network design should be cost-effective given a certain reliability requirement and proposes an optimisation algorithm based on a minimum amount of links to achieve the exact optimal design. [23] explores a heuristic network design algorithm designed to explore the placement of additional or redundant links in order to improve the reliability of a specific network. Using the source-to-terminal (s-t) reliability as measure, that is the probability that a given source can communicate with a given terminal a design strategy is presented which enhances the network reliability by adding links and therefore modifying the network topology. A root node, source node and "simple path" defined as the path that joins two nodes and traverses no node more than once are used to calculate s-t reliability using an algorithm based on the graph theoretical methods. The design algorithm assumes that nodes cannot fail and that link failures occur independent of source terminal failures. It is also capable of calculating the traffic distribution in the improved network by including traffic generated by each node - each node assumed to generate equal amount of traffic. New links are added into the network to reduce traffic in high traffic paths with a large hop-distance.

Similar to [23], in [26] an algorithm programmed as a Java/C++ software module is used to transform the graph of the network and to calculate 2- and all-terminal availabilities. The

algorithm is a generic one that can transform any network topology, which is a distinct advantage over the Markov state-space modelling technique discussed earlier in Section 2.4 that requires a different model for every topology. A method for calculating all-terminal availability from 2-terminal availability is proposed to calculate network availabilities for three different topologies with the same amount of nodes as summarised in Table 2.2 below. The authors infer that ring networks offer the best availability versus the cost measured in the amount of links used.

Table 2.2. Availabilities for three network topologies with seven nodes

Network topology type	**2-terminal Av**	**all-terminal Av**
mesh with 11 links	0.9999980	0.9999951
mesh with 9 links	0.99999975	0.9999983
ring with 8 links	0.9999977	0.9999964

Results obtained in Table 2.2 will be discussed and compared with the results of this investigation in Section 7.4.

2.6.2. IP network topology and redundant configurations

IP networks can be deployed in a variety of physical topologies as depicted in Figure 1.6, [29] ,[5]. [36], [37]. The first cabled IP based or Ethernet networks were bus-based systems interconnected with a common multi-dropped coaxial cable, forming a single physical collision domain, and therefore sharing the available bandwidth amongst all the hosts connected to the bus [3]. Modern IP networks in the local area network (LAN) are deployed on either copper/fibre cabled or wireless physical medium. Copper and fibre cable based Ethernet networks (100 Mbps/1000 Mbps) [15] are ubiquitous [3], [16] and constitute the focus of this investigation. They are usually wired in a physical star topology with the Ethernet (layer-2) switch at the centre of the star acting as a packet switch, consequently **isolating the physical collision domain** (Section 2.2.1). The Ethernet switch can also be wired in a ring as is quite often found in industrial control applications [2], [37], [152]. It should however be noted that Ethernet was not designed to be wired in a ring or mesh because redundant paths are created between transceivers. In order for this topology to function a spanning tree protocol such as Rapid Spanning Tree Protocol (RSTP) or Multiple Spanning Tree Protocol (MSTP) [39] must be activated on the switches that enables the switches to block redundant paths (Section 2.2.1.4), [3]. Spanning tree protocols are therefore also used to build redundancy into a LAN [29], [38] with a range of possible physical topologies - from rings to mesh topologies including combinations of rings and mesh in Figure 1.6.

[27], [140], [34] and [69] approach the calculation of IP network reliability from the perspective of the application layer. Before the 1980s network topology reliability studies neglected network congestion and delay, but due to ever increasing network loads these factors became important in determining network reliability. IP routing dynamics for example, are also now included in some IP network reliability studies [27]. The status quo in network reliability research can be classified into two types [34]:

- inherent reliability depending on connectivity and topology focusing on the network structure using probability and graph theory;
- applicable reliability and the actual performance of the network depending on the applications deployed and analysed using linear programming.

According to [34] there is no model that incorporates both inherent reliability and applicable reliability, therefore the purpose of the study in [34] is to "map the application into the network topology" by using a Markov modelling technique. It is assumed that node and link failures occur independently and the transfer of data between nodes or the data flow is a Markov process (see discussion in Section 2.4). [34] finds that the application-centric analysis of the reliability of a certain topology is related to the reliability of the components as well as the application profile and the relationship between applications in the profile. [25] uses graph theory method to show that a graph with a connectivity of greater than or equal to two is a sufficient condition for designing an Ethernet ring mesh topology as a carrier-class network used in conjunction with Ethernet ring protection (ERP) as introduced in ITU-T G.8032. It is found that the network availability increases with more interconnecting rings and decreases with more links.

The hierarchical network topology has been the dominant and recommended design topology used for IP networks. Accordingly the network is structured into a 3-layered access topology or three **topological layers** (not to be confused with OSI networking layers) [29], [149]:

- Access layer: The network layer where network hosts are physically attached to the network via a cable or a wireless AP. The access layer includes the physical or OSI layer-1 and layer-2 domains consisting out of hubs and layer-2 switches. The access layer may partition the physical collision domain through MAC-based packet switching (layer-2 switching) and full-duplex access links. The broadcast domain may be partitioned making use of IP subnetting and VLAN.
- Distribution layer: The network layer that takes care of IP routing, firewalling (packet filtering), QoS and broadcast/multicast traffic control using VLAN. The distribution layer includes the logical or IP connection domain using layer-3 switches and routers.
- Core layer: The network layer that is responsible for providing the network backbone and is responsible for fast and reliable packet switching using low latency circuits, redundancy and load-sharing with multiple redundant network paths.

For the purpose of this investigation the core layer is considered combined with the distribution layer in Figure 1.5.

A range of IP networks based on hierarchical mesh topologies are evaluated for availability in [28]. Calculation shows that end-to-end network availability is generally limited to three-nines - a typical value calculated being 0.9998. Although there are redundant parts of the network with high availability (typically six-nines) the limiting factor is the serial network connections that connect the end-nodes (hosts) into the network. A comparison with traditional telephone system with a VOIP system shows that the traditional system availability is typically four-nines versus the VOIP system availability of three-nines. However, a fully redundant hierarchical mesh topology with redundant links extending right down to the end-nodes improves the network availability to six-nines.

Ring network topologies are popular in industrial networks. The main technical motivation for the ring topology is that it fits in well with the existing industrial plant cabling layout. STP and RSTP are considered to converge too slowly for some industrial control systems. [32] takes an in-depth look at the fast spanning tree protocols and the role of spanning tree recovery on delay and performance in real-time Ethernet networks. The following mathematical formula is used to calculate the all-terminal reliability of a ring network consisting of *M* switches within one year:

$$R(MTBF,M) = 1-\left(1-\frac{1}{MTBF}\right)^{M} \tag{2.26}$$

where *MTBF* for a switch is assumed to be 64 years therefore the probability that at least one switch failure occurs in a ring topology of *M = 50* switches (chosen ring size in study) is 45%. The study evaluates the relationship between the ring size and the impact of spanning tree convergence time in networks with real-time performance requirements indicating that there is a maximum ring size that should not be exceeded given a certain real-time response requirement.

The ring based industrial network has led to the development of several proprietary ring redundancy protocols [37], [152], [151], [150], [30], [33]. The Ethernet ring redundancy protocol Media Redundancy protocol (MRP) proposed for PROFINET IO is a derivative of HIPER-Ring™ and has been standardised in [153] although is generally available in COTS switches and is therefore not part of this investigation. [155] suggest that the ring topology should to be used when long distances are involved in the links between switches and that the major contributor to selecting this topology is the prohibitive costs involved in providing a separate cable for every uplink to a centralised switch. The compromise is the sharing of bandwidth on the ring and introducing a common point of failure that can affect the entire network. The star topology does not suffer from these compromises. The reliability of both ring and star topologies is improved using redundant configurations.

A comparative study comparing the availability of different meshed Ethernet network topologies is conducted in [35]. Availability analyses are based on the probability of failure of all the equipment in the "minimal path" between two defined end-nodes i.e. a path that does not contain other paths as a proper subset. The end-nodes are chosen to be an application server and an arbitrary remote client exchanging data with this server. A path function *P* is defined as:

$$P_j(x) = \prod_{i \in P_j} x_i = \begin{cases} 1 \text{ if all path elemenst available} \\ 0 \text{ if any path element unavailable} \end{cases} \tag{2.27}$$

and a system function *S* defined as 1 when the system is operable; and 0 if the system is in a failed state derived as:

$$S(x) = 1 - \prod_{j=1}^{p} (1 - P_j(x)) = 1 - \prod_{j=1}^{p} \left(1 - \prod_{i \in P_i} x_i\right) \tag{2.28}$$

The following reliability parameters are then used to evaluate different combinations of ring topologies and mesh topologies:

Table 2.3. MTTF and MTTR values assumed for comparative study availability study with 16 nodes

Unit	MTTF	MTTR	Av
Cable	43 800 hours	24 hours	0.99945
Switch	43 800 hours	24 hours	0.99945

Figure 2.10 depicts the different topologies initially consisting of 16 nodes that are compared in [35] in terms of availability.

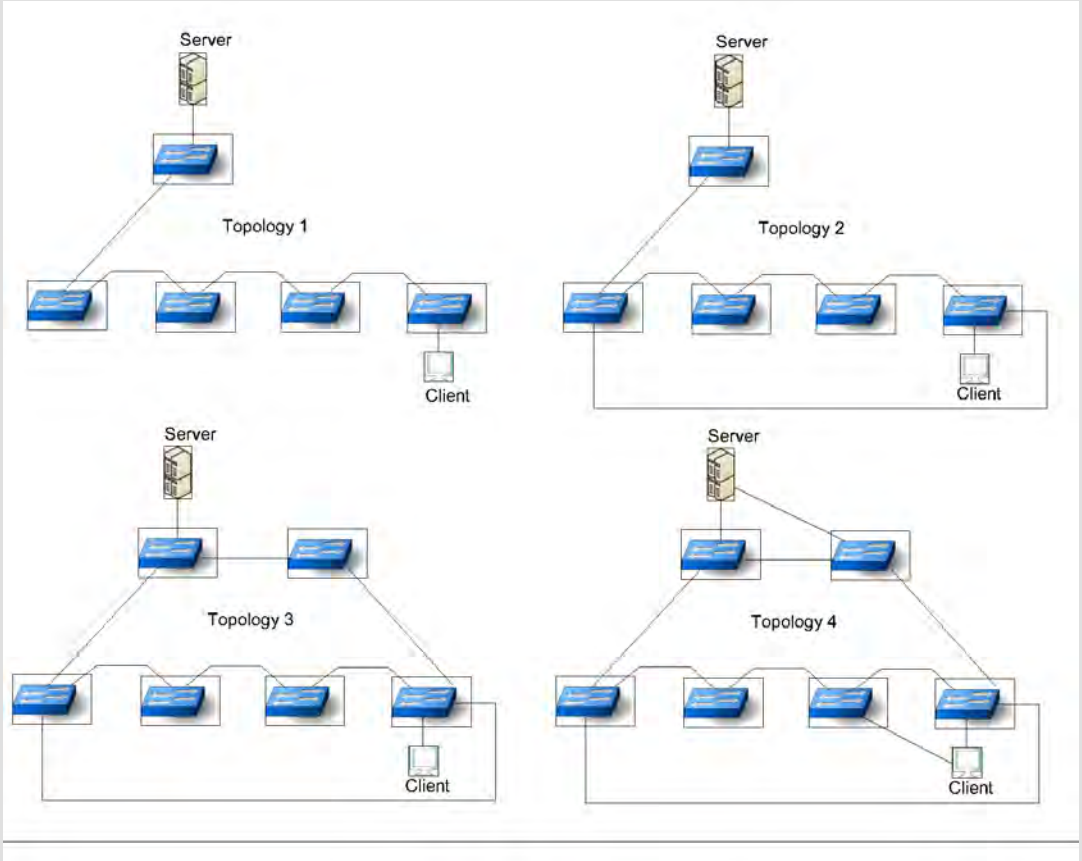

Figure 2.10. Comparative availability ring/mesh Ethernet topologies

Table 2.4 compares the availabilities for the above topologies as calculated in [35] using the system function in Equation 2.28.

Table 2.4. Comparative availability ring/mesh topologies results with 16 nodes

Topology no.	**Description**	**Av**
1	Serial connected - no ring	0.99399128
2	Ring	0.99890391
3	Ring with hierarchical mesh	0.99926765
4	Redundant links to end-nodes/ devices	0.99999684

From the above analyses the following important design guidelines are recommended to achieve high availability:

- Reliable components, cables and connectors must be used.
- Various network design and managerial strategies should be adopted to decrease the *MTTR*.
- The network design must eliminate single points of failure including electrical supply to network equipment.

- Redundant connections to the end-nodes or devices increases the network availability dramatically.

[31] uses reliability block diagrams to evaluate practical Ethernet architectures deployed in electrical substation industrial control systems as described in the IEC 61850 standard [154], [153]. Ring and mesh network topologies similar to those presented in [35] and summarised in Figure 2.10 are compared with the addition of a double ring topology (topology no. 5) with full redundancy right up the end-nodes or devices. Only switch failures are included in the study and the links or cables are ignored. The assumed *MTTF* for a switch is 50 years and the assumed *MTTR* is eight hours. The approximate availability in [31] for a network with 6 nodes is listed in Table 2.5:

Table 2.5. Comparative availability ring/mesh topologies results with 6 nodes

Topology no.	Description	Av	MTTF
1	Serial connected - no ring	0.999945175	16.6
2	Ring	0.999963476	25.0
3	Ring with hierarchical mesh	0.999963476	25.0
5	Redundant ring with redundant links to end-nodes/ devices	0.999999976	37966.6

It is concluded from the above calculations that the hierarchical mesh (or star-ring) and redundant rings provided higher reliability than ring network but at higher related costs. The *MTTF* calculation for the ring architecture is comparable to similar findings in previous research but it must be noted that the related literature does not provide reliability and/or availability analyses for the hierarchical mesh or redundant ring topologies.

2.7. Logical and other topological factors that influence network reliability

Data load, congestion or over-subscription has a major effect on the reliability of the IP network. In a real LAN environment with distributed applications various studies have shown that IP network traffic tends to be "bursty", meaning that data packets arrive in clusters with the instantaneous inter-packet arrival time within the clusters much smaller than the average inter-packet arrival time measured over a longer period of observation. The degree of "burstiness" of the LAN traffic is application dependent. The following effects have been noted [144]:

- Applications that generate higher degrees of bursty traffic suffer higher data delays.
- As the data traffic load increases, the detrimental effect of bursty traffic on network performance also increases.
- The larger the percentage of long data packets versus short data packets, the longer the delays experienced

This effect of bursty traffic and long data packets is more noticeable and pronounced on applications that requires real time response. [145] found that bursty background traffic on a

Ethernet switch based LAN can have a large detrimental effect on real time network dependent application performance even when the network utilization is low. This finding challenges the assumption that network over-sizing alone can lead to high levels of QoS.

In order to limit the problem of over-subscription and the resulting decline in reliability because of data congestion, a standard technique employed is to divide a large IP network into smaller logical networks by physical partitioning or by deploying VLAN as described in Section 2.2.1.5. There are many possible causes for poor performance or poor reliability in the modern VLAN partitioned network. These causes can be grouped into the following three categories [55]:

- Physical collision domain issues: over-subscription to a network segment, excessive cable lengths, damaged cables and connectors, switch ports/NIC Ethernet incompatibility issues, NIC hardware and driver problems.
- Broadcast domain issues: over-subscription on the VLAN, traffic loops, congestion on switch inband path, ingress errors on cut-through switches, switch CPU overload, switch software/ hardware bugs and VLAN configuration errors.
- Inter VLAN connection issues: inter VLAN routing misconfiguration, software instead of optimised hardware (ASIC) switching, host IP layer misconfiguration including incorrect default gateway settings.

Topological factors including redundant configurations and the elimination of common points of failure are critical in designing reliable systems [38], [5], [54]. The role and topology of centralised systems like routers, DNS and DHCP services are therefore an important part of network reliability. It is however hard to find any published literature on the role of DNS and DHCP services related to common point of failure mode and physical location in the network. It has been pointed out for example in Section 2.2.4.2 and Section 2.2.4.3 that both DHCP and DNS can be configured manually to reside on the individual hosts instead of being located on the network. Intuitively this should, for small fixed networks at least, result in higher reliability.

2.8. Network reliability modelling tools

[132] evaluates various tools for computer performance modelling and reliability analysis and is briefly summarised below:

- GreatSPN is a tool suited to the solution of stochastic Petri nets (GSPNs) [117], [118] and with a variety of evaluation techniques including structural analyses, state-space analysis, simulation, and Markov chain generation.
- Java modelling tool (JMT) consists of a suite of applications for queueing network analysis and workload characterisation.
- Mobius is a network modelling environment that supports the specification of combined performance and reliability models based on stochastic activity networks (SANs), fault trees, and stochastic process algebra.
- OPEDO - is a tool suited to the optimisation of performance and dependability models with support for the numerical optimisation of performance or dependability metrics in discrete event systems.
- PEPA Eclipse plug-in supports the definition of stochastic process algebra models based on the performance evaluation process algebra (PEPA) language [118]. It is dependant on the Pepato

library and supports the static Markovian differential equation analysis [133] of these models by means of numerical and simulation methods.

- PIPE2 is a tool suited to the solution of GSPNs supporting structural and performance analysis, integrating the graphical specification of performance queries utilising a client-server software architecture for parallel and distributed model analysis.

- PRISM is a probabilistic model checking tool that supports Markov chains and reward models. It includes a language for textual modelling that allows for the validation of model properties.

- QPME is a tool suited to the specification of queueing Petri nets (QPNs) a modelling technique that integrates queues within stochastic Petri nets. It allows evaluation of QPNs by Monte Carlo simulations [134] and supports various performance data collection modes.

- SHARPE is a tool suited to the analysis of fault trees and reliability block diagrams, acyclic series/parallel graphs, Markov models, GSPNs and queueing networks.

- SMART is a tool suited to the model checking, as well as numerical evaluation and simulation of stochastic models extended from GSPNs.

- Tangram-II is a suite of tools with support for system modelling and traffic measurement based on Markovian, non-Markovian models, hybrid Markov models (HMMs) and fluid-models. It also includes network and performance simulation modules.

The Tangram-II modelling environment is introduced in [131], [135] and evaluated in [136]. It is based on an object orientated model description syntax and implemented in the C++ programming language. It makes use of the TGIF (TANGRAM Graphic Interface Facility) [156] user interface to construct the models. The tool integrates different modelling environments and combines a range of mathematical techniques for developing and analysing computer and communication models. The user estimates a range of values for the system parameters collected from real systems, experimentation or assumed from past experience. The model is then constructed using these estimated system parameters and solved using an analytical or simulation technique. Tangram was designed to support research, application development and education in system reliability and performance evaluation and includes modules to conduct measurements in computer networks and to collect relevant reliability and performance statistics. The environment includes techniques for deriving and solving underlying Markov models, transient analysis, as well as a specific class of non-Markovian models. The Tangram numerical solver supports mathematical techniques and supporting optimised computing algorithms based on the randomisation technique to calculate cumulative operational time distributions for repairable systems introduced in [139]. Formulas are derived for calculating the cumulative operational time distribution for a system, modelled as a homogeneous Markov process, i.e. the distribution of the total time that a system was operational during a finite observation period. The following quantities of interest are derived:

- Average (or expected) availability *E[A(t)]* (see Equation 2.8).

- System reliability *R(t)* (see Equation 2.12).

- Expected lifetime *E[L(t)]* where:

$$\mathrm{E[L(t)]}(t) = \int_0^t \mathrm{R}(\mathrm{x})\,d\mathrm{x} \tag{2.29}$$

- *MTTF* where:

$$MTTF = \lim_{t \to \infty} E[L(t)] \tag{2.30}$$

The use of the Tangram modelling environment is further discussed in Section 3.4.

2.9. Network model simulation and validation tools

The basic principles of model validation, verification and testing (VV&T) are described in [90].

Several network simulation software tools exist. OPNET [70] is a popular commercial tool used for creating network simulation models [113], [73]. It is a multi-purpose network simulation framework that includes (from [70]):

- real-time visualisation module;
- diagramming and documentation module;
- audit and policy compliance module;
- planning and engineering module;
- simulation module.

The usage of the OPNET simulation module is further discussed in Chapter 6.

The OMNet++ simulation framework [71] is an Open Source licensed set of network simulation tools developed as an object orientated C/C++ library. In [72] the OMNet++ libraries are used to develop the R-simulator sensor node module with a full protocol stack including application, network, data link and physical layer. Hardware components including battery, CPU, memory and mobility are also included in the model.

A functional description of "network performance" in terms of speed, capacity and transaction distortion, includes such parameters as latency, available bandwidth, packet loss, jitter rate and packet reordering and can be included in a single performance framework. It must be noted that there is a difference between performance as measured across a specific network path and the network as an entity, the average network performance can differ substantially from the performance of an individual path [112].

IP networks are used in very complex computing environments and network reliability and "network performance" is application specific. A strict testing methodology is proposed in [59] based on the following principles:

- Building test profiles: the operation conditions specifying packet inter-arrival time and packet size, both factors determined by the application.
- Sample selection: sample selection method, either unit or packet sampling. An IETF standard has been set up [129] for packet sampling.
- Identification of failure criteria: the boundary criteria that are defined to determine whether a failure has occurred. Two main categories are defined - function failure and parameter failures.

A combined indexed performance analysis method introduced in [140] is used to evaluate reliability for the network as a function of the reliability of the individual units. [140] proposes

a combined reliability parameter for a unit that includes and combines the following parameters that would describe the network reliability as viewed from the application layer:

- Timeliness: data rate and delay where the data rate reflects network congestion.
- Correctness: connectivity rate, lower application layer quality and BER.
- Completeness: data loss rate and flow rate.

Different indexed reliability parameters may be selected for different applications. [59] uses the following indexed failure criterion for an FTP application indicated in Table 2.6 below:

Table 2.6. Failure criterion for FTP application

Failure Type	Failure criterion	Description
Function failure	Connection loss	A TCP client is connected from the TCP server
Parameter failure	Transmit delay < 15s	Minimum delay threshold in seconds
	Transmission loss < 2%	Packet Loss Rate (PLR) threshold as percentage
	Transmission error < 2%	Bit Error Rate (BER) threshold as percentage

Network performance is measured using a variety of techniques and software tools. Lmbench, Netpipe and Netperf are popular Open Source licensed tools that can be used to measure network bandwidth and latency [111]. SNMP management tools are also used to record network performance related SNMP counters [112].

2.10. Chapter closure

In the following chapter the knowledge and insight gained from the literature study, with special reference to Section 2.3 and Section 2.5, is applied in the formulation of a modelling methodology. Appropriate modelling tools are selected based on the literature explored in Section 2.8 and model validation tools explored in Section 2.9 are then used in Chapter 6 to validate the various models developed. Finally, in Chapter 7, results obtained from previous work on network topology and its contribution to reliability and performability discussed in Section 2.6 are compared with the results obtained in this investigation.

Chapter 3. Modelling Methodology

3.1. Introduction and definitions

Reliability and performability in IP networks can be analysed at the link (physical) layer as well as at the connection (packet delivery) layer [27], [140], [34]. The link topology models constructed and analysed in this chapter focus the reliability and performability investigation on the link or physical layer, combining the contribution of the various hardware components and units in different network topologies to determine the overall link reliability. The link is defined as the physical path between an arbitrary end-node or host units on the network - one unit typically hosting a client application and a centrally located end-node unit that typically hosts a server application. This 2-terminal approach is discussed in [26] and [23]. The link unit is therefore the physical path that data would flow through [34], [35] in a node-to-node IP connection in an Ethernet network as indicated in Figure 3.1, it includes interconnecting hardware i.e. network components and equipment units such as network interface cards, cables, connectors, hubs and Ethernet switches.

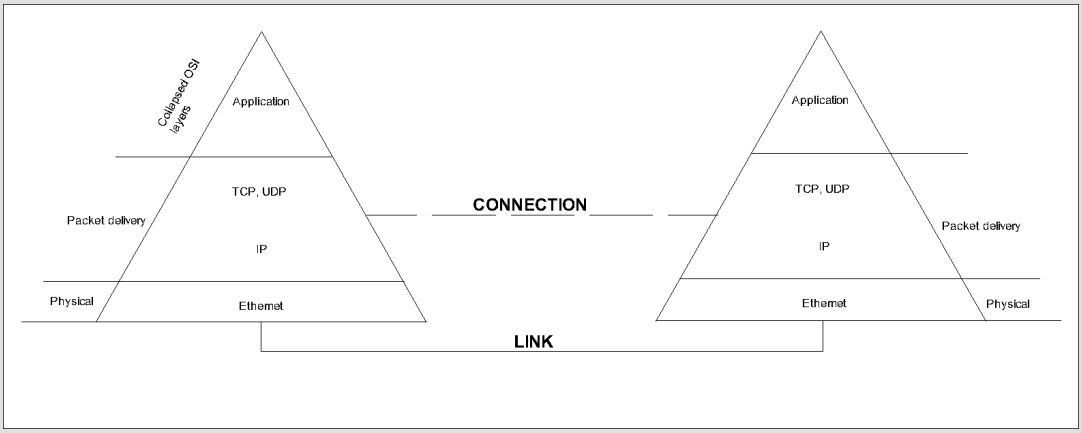

Figure 3.1. Definition of link and connection

The above figure also illustrates how a connection can also be seen as a "virtual link" or a data stream that flows through a physical path or physical link across the OSI defined layers as depicted in Figure 1.4, however if the physical link is broken there can be no connection [27], [34], [69].

The investigation starts with the development of a physical link model that can be used as a basic unit in the subsequent analyses of different Ethernet LAN topologies. The modelling methodology as well as the modelling tools are introduced and the various reliability and performability related measures of interest that can be calculated are explained. The link models generated are then used in more advanced LAN topology models. The very common Ethernet network topologies used in industry and modelled in this chapter are depicted in Figure 1.6 and summarised as:

- star topology network system (stn)
- ring topology network system (rtn)
- mesh topology network system (mtn)
- hierarchical mesh topology network system (hmtn)

In this chapter the modelling methodology is demonstrated at the hand of an example component reliability model generated with the Tangram modelling environment introduced in Section 2.8. The model presented as an example for the discussion of the modelling methodology is a trivial one for a basic network component, that of a network interface card that is used in subsequent models as a basic unit building block.

3.2. Hierarchical approach

A reliability model consists of a hierarchical assembly of components, units and systems (see discussion in Section 2.1). It has also been noted in Section 2.4 that the larger and more complex a model becomes, the larger the state-space required to model the system and the more time consuming and computing-resource intensive solving the model.

The hierarchical approach followed is therefore similar to the one elaborated in [119] and employed in [125], [127], [126] to reduce the large state-space dilemma through a systematic, modular approach, feeding derived results into a bottom-up hierarchical block analysis. The hierarchical approach is valid given that certain assumptions are valid as listed in [127] and summarised in Section 2.4. The weakness in this approach is that the model becomes "hard-coded" at the unit layers with constants where a model with more underlying variables gives more accurate results. However, the goal is not to create a numerically precise solution to the prediction of expected reliability and performance, but rather to create realistic, comparative models based on assemblies with reliability parameters that are representative of generally used LAN equipment and that can be used to evaluate design trade-offs in different network topologies. Such a model can be recalculated bottom-up to adjust to different underlying hardware components for more accurate predictive results.

3.3. Symbols, base data, assumptions and conventions

- The following symbols are used to represent components, units and systems indicated in Figure 3.2:

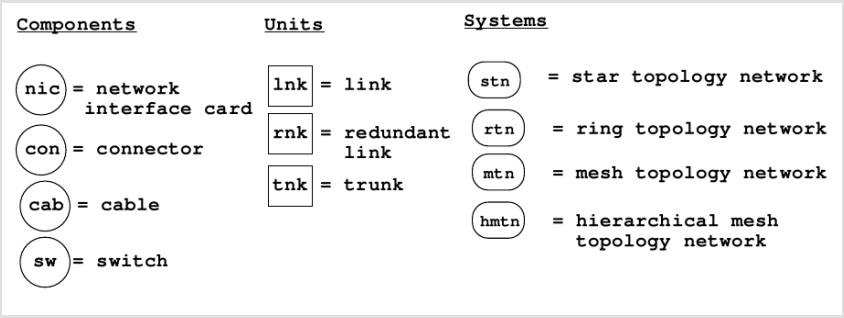

Figure 3.2. Symbols used in hierarchical models

- Base *MTTF* data used in models: Base data is extracted from various industry reports and databases where available. References are supplied for data that can be backed up by academic or reputable technical sources or are otherwise assumed [61]. The following *MTTF* base data is used:
  - Network Interface Card (nic) *MTTF = 30 yr* (figures assumed from popular literature)
  - Cable (cab) *MTTF = 1000 yr* ([79])

- Connector (con) *MTTF = 20 yr* (figures assumed, no data is available for Ethernet type connectors)
- Switch (*sw*) *MTTF = 40 yr* ([32], [31], [28])

(see Section 2.4 for a discussion of *MTTF* and *MTTR*)

- Base *MTTR* data used in models: *MTTR* is generally assumed to be 8 hours (hr) where *MTTR = 1 (8hr) unit* which is the typical duration of a working shift [31], [28], [35] with corresponding repair rate $\mu = 1$. The *MTTF* is then calculated in time units of 8hr with the corresponding failure rate λ as inverse of the *MTTF* in 8hr time units.
- In general, except where otherwise stated, non-shared repair facility components, units and systems are assumed, therefore simultaneous repairs on failed components are possible and every component, unit and system has an independent or non-shared repair rate μ.
- *MTTF* steady-state convergence: Reliability measures reliability *R(t)* and expected lifetime *L(t)* are plotted for the first 1 million hours or 125 000 8hr units.
- *MTTF* upper limits: For models with very low failure rates *MTTF* is calculated for up to 1000 million hours or the first 125 000 million 8hr units. Components, units and systems with *MTTF* larger than 114 077 years are therefore considered never to fail and are excluded from reliability models.
- Rounding off *MTTF*/*MTTR* numbers: Base data rounded to nearest 10 years where a 1 year = 8766 hours. Derived *MTTF* data for components and units expressed in years rounded up to the nearest year. Model calculated network *MTTF* rounded to the nearest hour.
- Failure rate (λ) numbers, or the inverse of the *MTTF*, are rounded to three significant digits.
- Repair rate (μ) numbers, or the inverse of the *MTTR*, are rounded to three significant digits.
- Rounding off steady-state availability numbers: Numbers are rounded to the digit after the last significant '9'.
- Rounding off performability metric *M*, where *M = B/T*: Numbers are rounded to the third significant decimal.

3.4. Modelling environment and supporting software

The Tangram computer system/network and reliability modelling environment was selected for conducting the investigation. The Tangram modelling environment has been used in many academical peer-reviewed computer system/network performance and reliability studies, the algorithmic modules used for the analyses of reliability and performability measures of interest are well documented in [139]. The source code of the software is available under a license that allows the software to be verified and modified if required for "not-for-profit purpose". The use of the program and the underlying modules are well documented in [137].

The following Tangram modules are used:

- Specification Module used to specify reliability models using a graphical object design tool with C-language programming blocks. The underlying model is stored as a file with the ".obj"

suffix and can be rendered for editing using the Tgif drawing tool [156] with an example file depicted in Appendix A, Figure A.1. The corresponding C-language user code file is stored with a "user_code.c" suffix with an example included in Appendix B.

- Mathematical Module used to generate the underlying Markov model from the above model specification. The output is the state-space or generator matrix (Section 2.4) and is stored as a file with the ".generator_mtx" suffix containing all the state transition rates. The corresponding model states are stored in a file with the ".states" suffix and the state variable descriptions are stored in a file with the ".state_variable" suffix.

- Analytical Solution Module used to generate both the transient and steady-state solutions using the above generator matrix as input and one of several analytical solution techniques (Section 2.8) to solve the model. The steady-state solution is generated using the direct solution method to calculate the stationary point probability steady-state vector using the Grassman/ Heymann algorithm [138] (GTH method) and then stored as a file with the ".SS.gth" suffix. The transient solution is arrived at by using one of several transient solution options available in the modelling environment, the "Operation Time and Related Measures" option using the randomisation technique introduced in [139] is used and the solution is stored in a file with the ".TS.operational_time" suffix.

- Measure of Interest Module that can be used to visualise the solutions generated above and to perform various average rate reward and average impulse reward calculations that are relevant to the reliability and performability measures of interest. This module is used to verify the steady-state availability and the rate-based performability measures of interest derived from the steady-state solution.

It should be noted that the Tangram module for "Operation Time and Related Measures" was modified and recompiled to generate text files containing the numerical data results in a column format that is easy to use as the input file to the gnuplot graph plot programme [157] that is used to render the reliability and lifetime graphs.

3.5. Analytical model

The Tangram modelling environment generates the underlying Markov model (see discussion in Section 2.4) from a network reliability specification where the Markov model is characterised by its infinitesimal state-space generator matrix (Q). For simple models the generator matrix can be generated by hand [122]. However, for larger and more complex models, the modelling environment automates this manual process by recursively exploring reachable states in the model and calculating the transition rates between these states [137]. The use of the Tangram Specification Module to specify a reliability model for a network interface card (*nic*) component is shown in Figure 3.3.

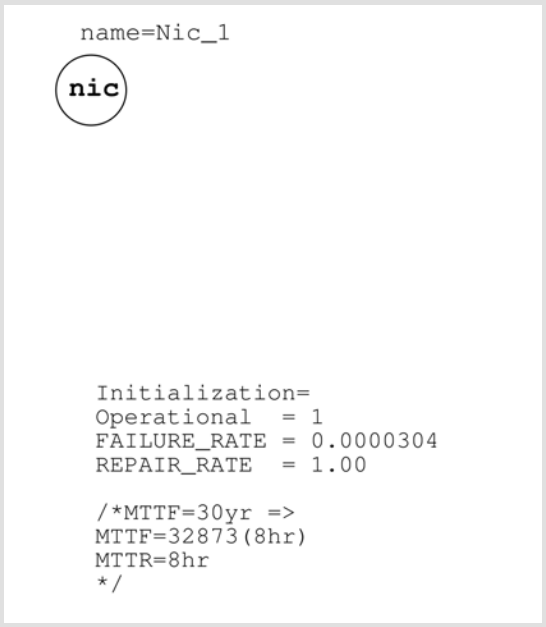

Figure 3.3. Tangram Model: Example

Realistic values extracted from industry data for *MTTF* and *MTTR* are substituted for constants FALURE_RATE ($\lambda = 1/MTTF$) and REPAIR_RATE ($\mu = 1/MTTR$) respectively. See Section 3.3 for a discussion of the base data used in the models. The model is further programmed using a C-language to specify State Variables, Events, Messages and Rewards as described in [137].

The Events used for Markov reliability models are the *FAIL* event with an exponential rate equal to *FAILURE_RATE*, and a *REPAIR* event, that repairs the object with exponential rate equal to *REPAIR_RATE*. Full C-language source code listing for this example model is included in Appendix B.

The following State Variables are defined:

- *Nic_1.Operational*

The following rate Reward variables are defined:

- *Nic_1.nic_availability*

```
Rewards=
rate_reward =nic_availability
condition   = ( Operational == 1 )
value       = 1;
```

The following Global Reward variables are defined:

- *system_availability*

```
global_rewards=
rate_reward=system_availability
condition=((Nic_1.nic_availability ==1))
value = 1;
```

The global Reward variable *system_availability* is used to store the overall or system availability which in the case of a single component model is equal to the component availability.

The corresponding Markov state-space analytical model extracted from the generator characterised by state transition matrix (*Q*) and automatically generated by the Tangram Mathematical Module is shown in Figure 3.4.

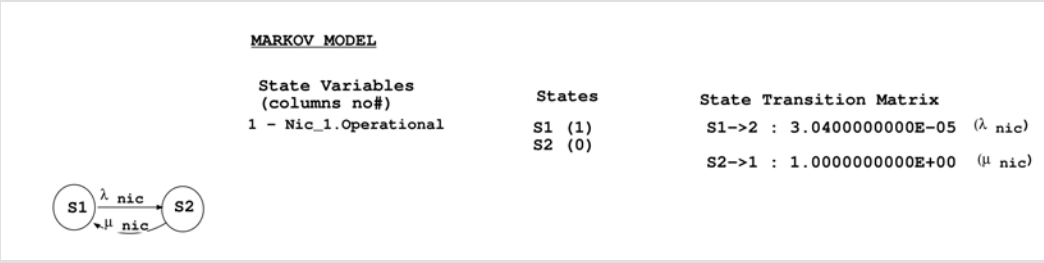

Figure 3.4. Markov Model: Example

In this trivial example a two-state generator matrix representing the fail-repair states of a single component is generated. The only State Variable *Nic_1.Operational* can assume one of two values: where *Nic_1.Operational = 1* (operational) which is labelled state S1 or *Nic_1.Operational = 0* (down) which is labelled state S2. As expected from the theory discussed in [122] the transition from S1 to S2 occurs at constant exponential *FAILURE_RATE*, and the transition from S2 to S1 occurs at constant exponential *REPAIR_RATE*.

3.6. Model solution

The Tangram modelling environment's analytical solver is introduced in Section 3.4. The transient solution is calculated using the Tangram transient analytical solver to calculate the following quantities of interest based on the randomisation technique introduced in [139]:

- Reliability: *R(t)* function calculated using Equation 2.12.
- Average availability: *Av(t)* function calculated using equation Equation 2.8.
- Expected lifetime: *L(t)* function calculated using equation Equation 2.29.
- Mean time to failure: *MTTF* is calculated using equation Equation 2.30 where *MTTF* is the steady-state value of expected lifetime *L(t)* as time goes to infinity.

The *R(t)* graph is shown in Figure 3.5 and the *L(t)* graph is shown in Figure 3.6.

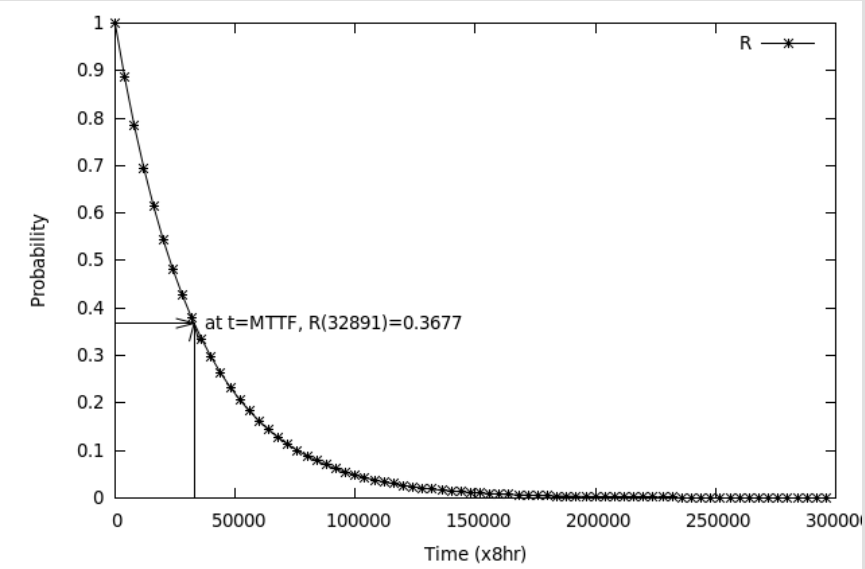

Figure 3.5. Reliability graph: Example

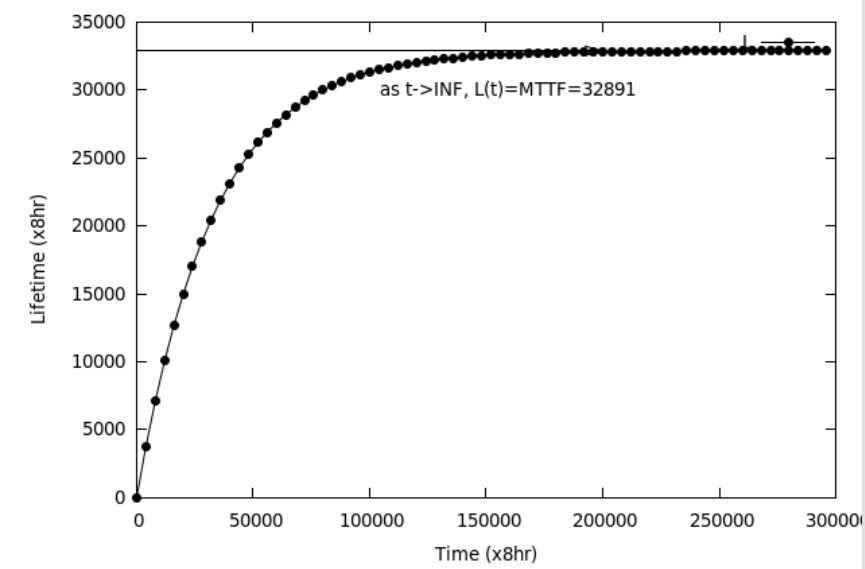

Figure 3.6. Expected lifetime graph: Example

The steady-state solution is calculated using the steady state analytical solver to calculate the following measures of interest:

- Steady-state availability: *A* is the steady-state value of point availability *A(t)* as time goes to infinity (Equation 2.8). The steady-state availability is calculated from rate reward variable *system_availability* using the "Measure of Interest Module" and selecting the stationary point probability steady-state vector file as input.

  A can also be calculated manually by substituting *MTTF* derived from the transient solution above and the mean time to repair (*MTTR*) values into equation Equation 2.13.

It must be noted that for the numerical transient solution for assemblies with very low failure rate are calculated for the first 1 million hours or 125 000 8hr units. The steady state availability is calculated independently using the steady-state solver.

In this example the *MTTF* is found from the steady-state value of *L(t)* as *MTTF = 32891* or 30.0 years. Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated from Equation 2.10 to be *A = 0.99997*, where both *MTTR* and *MTTF* are scaled to 8hr units.

3.7. Model verification and validation

Model verification and validation is straight forward for the above trivial example of a single component model. The underlying software algorithms of the modelling environment and the analytical solvers can be verified by inspecting the solution generated against manually calculated results derived from the standard model, thereby confirming the validity of the algorithms used to calculate the key reliability measures of interest and verifying the correct programming of these algorithms.

Since the single component model is specified with a known failure rate (*1/MTTF = 32873*) and repair rate (*1/MTTR = 1*), the solution to the model is known. The predicted value for the reliability function *R(t)* when *t = MTTF = 32891 time units* is calculated by substitution into Equation 2.2 yielding a value of: $R(32891) = e^{-1} = 0.3677$. Finding the value of *R(t)* for *t = 32891* on the reliability graph produced by the analytical solver indicated in Figure 3.5 the corresponding reliability probability is 0.3677 as expected.

The predicted result for *MTTF* as the steady-state expected lifetime *L(t)* produced by the analytical solver indicated in Figure 3.6 is *MTTF = 32891*. There is a 0.05% difference between the *MTTF* specified in the model and the *MTTF* calculated by the solver as a result of rounding of the failure rate to three significant digits as stated in Section 3.3, thus verifying and confirming the validity of the model.

The predicted value for steady-state availability (*A*) as calculated from *MTTF* and *MTTR* using Equation 2.10 is *A = 0.999969581* rounded to the digit after the last significant 9 as 0.99997. The steady-state availability calculated by the solver from rate reward variable *system_availability* using the Measure of Interest Module is *A = 0.999970* rounded to 0.99997, thus verifying and confirming the validity of the model.

From the above discussion the verification and validation of network reliability models specified will be done using one of the following two approaches depending on the complexity of the model that is being specified:

1. Manual validation by comparing results obtained from the modelling environment against the results obtained from the standard mathematical model. This approach will be used for relatively simple models consisting of serial and/or parallel reliability block diagrams that can be simplified and solved (Section 2.4).

2. Validation by simulation of more complex models that can not be simplified or readily solved by using block reliability equations. Model validation testing is the topic of Chapter 6.

3.8. Chapter closure

In the following chapter the modelling methodology developed in Section 3.2 and Section 3.5, the Tangram modelling tool identified and discussed in Section 3.4 and verified in Section 3.7 are

used to develop the appropriate link topology models that are the building blocks for subsequent network models to be developed in Chapter 5.

Chapter 4. Link Topology Model

4.1. Introduction and definitions

A link model with representative assemblies of common equipment used in the modern LAN is developed by using a hierarchical approach as discussed in Section 3.2. The simple link model is developed to serve as a basic unit to develop more complex unit and system models. Very common network link units used in industry are the following:

- link (*lnk*)
- trunk (*tnk*)
- redundant link (*rnk*)

4.2. Simple link model

The vast majority of Ethernet links consist of only a couple of basic components for which reliability statistics are either available from manufacturer's data sheets or from general industry sources (Section 3.3). The IEEE 802.3 compliant Ethernet network can consist of either copper and/or fibre based links connected from host or end-node to switch, or from switch to switch in a variety of network topologies. The simple link model introduced in this section is based on a typical switch to switch link.

4.2.1. Model specification

The simple link model specification consists of the following three components that are connected in series and is associated with an inter-switch link:

- cable connector (*Con_1*) that connects the cable to a switch port
- cable (*Cab_1*) that interconnects the two switches
- cable connector (*Con_2*) that connects the cable to a switch port

The simple link unit (*lnk*) is described by "Model A" and the model specification is shown in Figure 4.1.

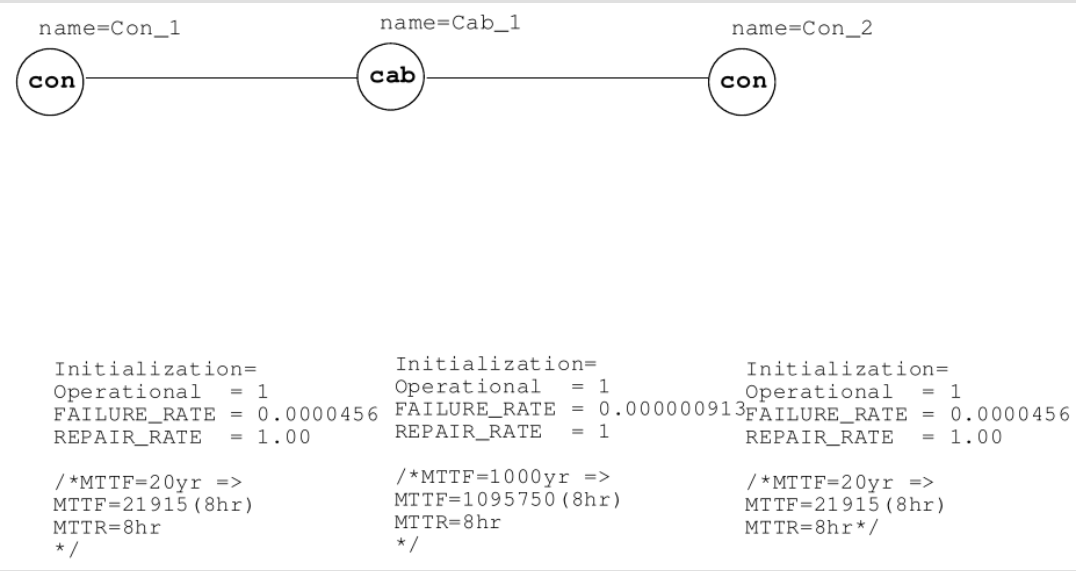

Figure 4.1. Model A: Simple link model

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 4.2.

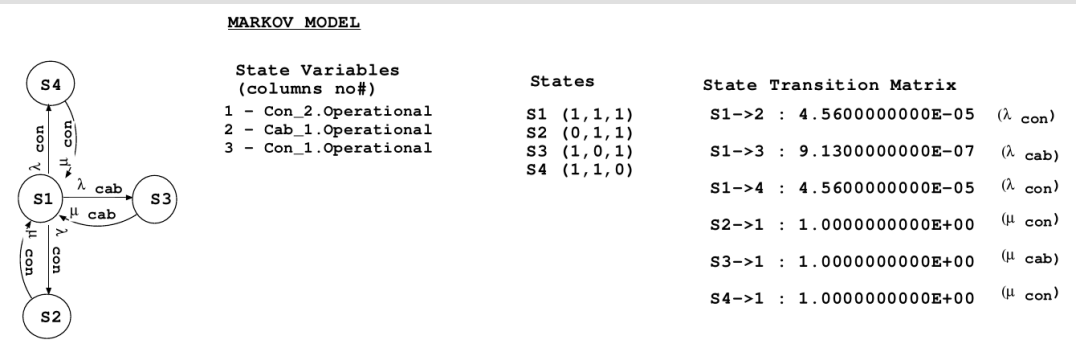

Figure 4.2. Markov Model A: Simple link model

4.2.2. Model solution

The transient and steady-state solutions to the Markov model generator matrix are generated by running the Analytical Solution Module. The transient solution for reliability *R(t)* as a function of 8hr time units for the above simple link model is shown in Appendix A, Figure A.2. The transient solution for expected lifetime *L(t)* is shown in Figure A.3.

The *MTTF* is found from the steady-state value of *L(t)* as *MTTF = 10856* or 10.0 years. (4.1)

Using this calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.99991*. (4.2)

4.3. Host-to-host link model

The host-to-host link model serves as a benchmark for evaluating the reliability of IP networks. The host-to-link is based on a typical host to host network connection. The arbitrary two hosts or end-nodes can be any two networked devices connected to the same switch, for example desktop computers, VOIP phones, IP cameras or a server as indicated in Figure 2.1.

4.3.1. Model specification

The host-to-host link model specification consists of the following five components that are connected in series and is associated with two arbitrary end-nodes or hosts connected via a switch:

- nic (*Nic_1*) attaching the first host to the first link
- link (*Lnk_1*) attaching the first host to the switch
- switch (*Sw_1*)
- link (*Lnk_2*) attaching the second host to the switch
- nic (*Nic_2*) attaching the second host to the second link

The host-to-host benchmark unit is described by "Model B" and the model specification is shown in Figure 4.3.

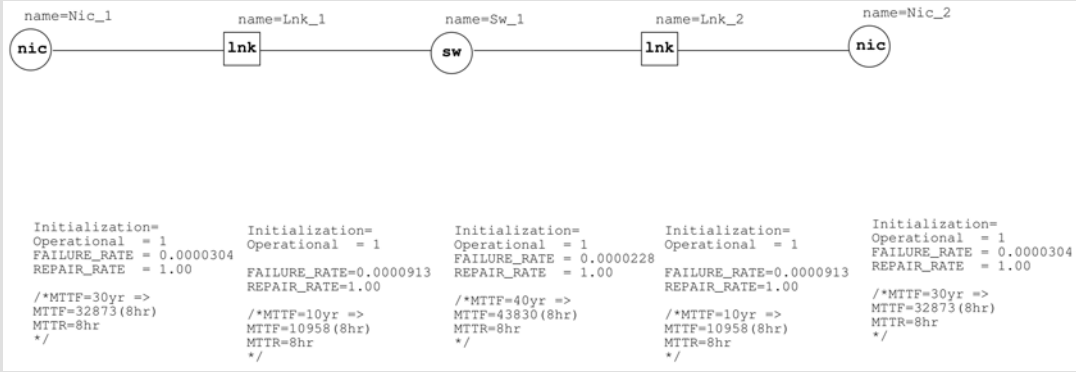

Figure 4.3. Model B: Host-to-host link model

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 4.4.

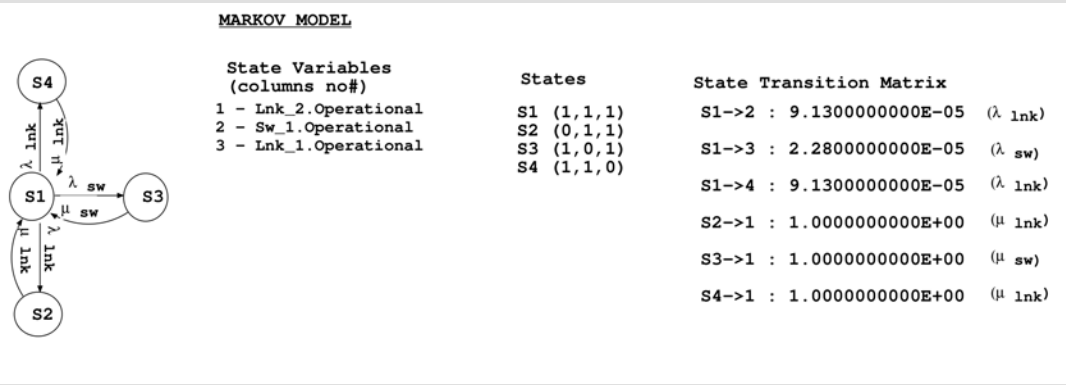

Figure 4.4. Markov Model B: Host-to-host link model

4.3.2. Model solution

The transient solution for reliability *R(t)* as a function of 8hr time units for the above host-to-host link model is shown in Appendix A, Figure A.5. The transient solution for expected lifetime *L(t)* is shown in Figure A.6.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 3757* or 3.4 years. (4.3)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.9997*. (4.4)

4.4. Trunk link model

The dual cable connection acting as one physical link is also referred to as a trunk (Section 2.2.1.3) and is a very common redundancy technique deployed to improve network reliability. In a trunk both cables carry data, and are therefore operational, effectively doubling the data carrying capacity of the link. This configuration is also know in reliability terms as a "hot standby" redundant configuration as explained in Section 2.4.

4.4.1. Model specification

The trunk link model specification consists of the following two components that are connected in parallel in a hot standby redundant configuration:

- link (*Lnk_1*) attaching a switch to a switch derived in Section 4.2
- link (*Lnk_2*) a second link attaching the same two switches

The trunk link unit is described by "Model C" and the model specification is shown in Figure 4.5.

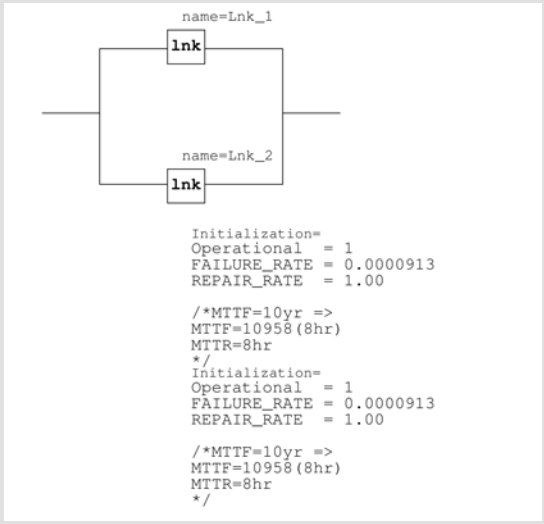

Figure 4.5. Model C: Trunk link model

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 4.6.

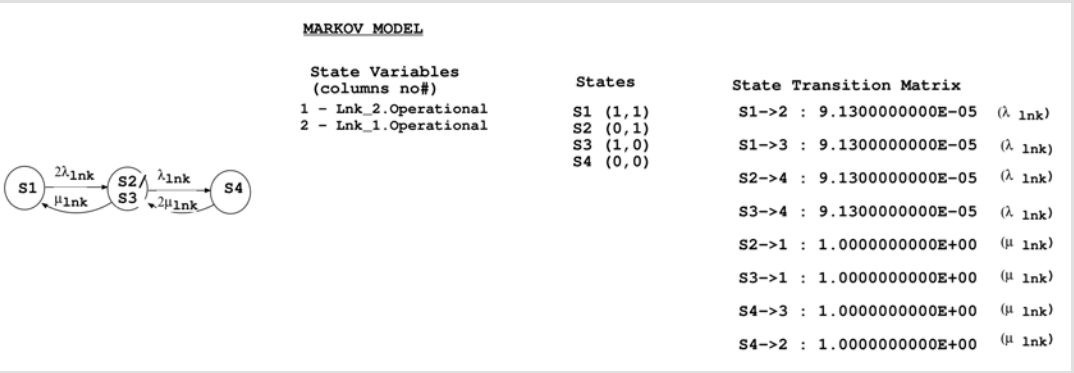

Figure 4.6. Markov Model C: Trunk link model

4.4.2. Model solution

The transient solution for reliability *R(t)* as a function of 8hr time units for the above trunk link model is shown in Appendix A, Figure A.9. The transient solution for expected lifetime *L(t)* is shown in Figure A.9.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 33061306* or 30170 years. (4.5)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.99999998*. (4.6)

4.5. Redundant link model

The dual cable connection with one link being in a blocked mode (see Section 2.2.1.4) is also referred to as a backup or redundant link (Section 2.2.1.3) and is built into the generally used RST protocol. Only one link carries data, the other link is in a non-operational state, also referred to as cold standby redundant configuration as explained in Section 2.4.

4.5.1. Model specification

The redundant link model specification consists of the following two components that are connected in parallel in a cold standby redundant configuration:

- link (*Lnk_1*) attaching a switch to a switch derived in Section 4.2
- link (*Lnk_2*) a second link attaching the same two switches

The redundant link unit is described by "Model D" and the model specification is shown in Figure 4.7.

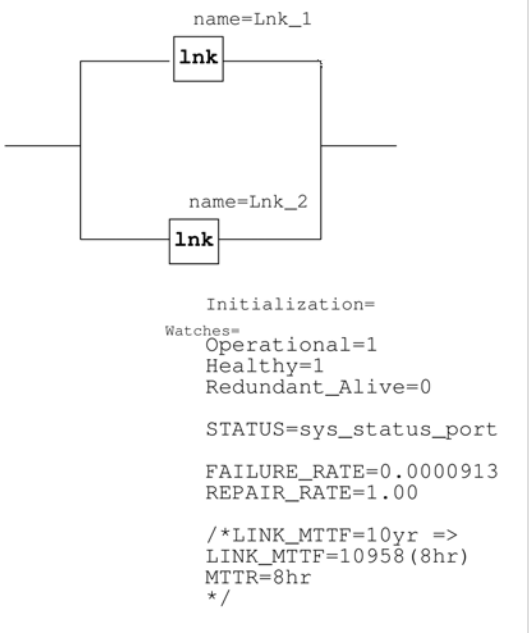

Figure 4.7. Model D: Redundant link model

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 4.8.

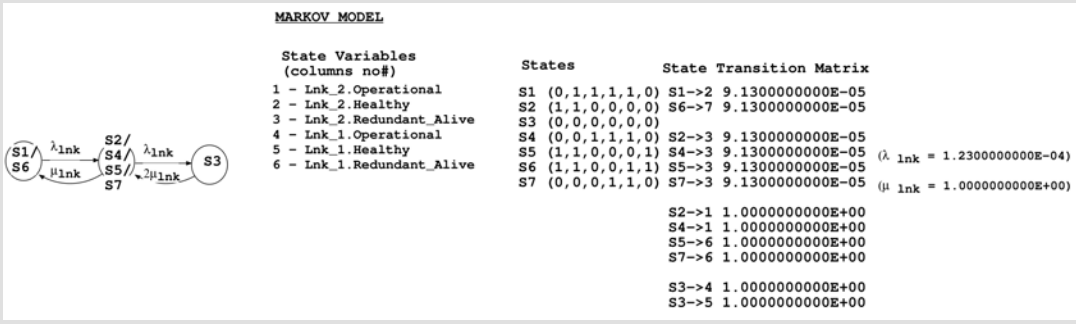

Figure 4.8. Markov Model D: Redundant link model

4.5.2. Model solution

The transient solution for reliability *R(t)* as a function of 8hr time units for the above redundant link model is shown in Appendix A, Figure A.11. The transient solution for expected lifetime *L(t)* is shown in Figure A.12.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 12018513* or 10970 years. (4.7)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.999999992*. (4.8)

4.6. Chapter closure

The link models developed in this chapter are used as basic building blocks in the network models to be developed in the next chapter. In particular the simple link model in Section 4.2 is used as a unit in all subsequent network models and the trunk model reliability parameters are referenced in Chapter 7 when discussing performability enhancements. The host-to-host module in Section 4.3 is also used to verify the failure-repair module developed in Chapter 6 as well as to calculate the *Km* scaling ratio used to compare model predicted performability parameters with the validation test simulation results.

Chapter 5. Network Topology Model

5.1. Introduction and definitions

The most popularly used Ethernet topologies that will be modelled in this chapter are shown in Figure 1.6. A generalised mesh network topology based on the 3-layer access topology [29], [149] (Section 2.6.2) that will be used to derive models for the special ring, star and hierarchical mesh network systems is depicted in Figure 5.1. This type of network system consists of various node junction units represented by either hubs in older generation networks, and switches in modern systems, that are interconnected by single link units although dual redundant link units or trunk units can also be used in high availability configurations. The switching node units employ packets switching based on the layer-2 MAC address as discussed in Section 2.2.1.2. The host units *h0* and *hi* are positioned as two arbitrary end-nodes in such a network system, where *h0* represents the location of a central end-node or host that is typically a server unit and *hi* a client end-node or host unit that is located elsewhere in the mesh. This server-client host functionality is very common in IP network based applications with TCP/UPD over IP connections [35], (Section 2.2.2.2). This simplifies the definition of "network reliability" and the reliability modelling approach since the reliability of the link between two defined end-node units as indicated in Figure 3.1 and discussed in [35], [26], [23] can be used as a reference instead of attempting to define an overall reliability measure for all the possible paths and connections between all the end-nodes in a network as used in various network reliability studies [141], [143], [140]. Such an approach also makes it possible to compare network reliability models to simpler direct link models as derived in Chapter 4.

In Figure 5.1 switch unit *sw_0* connects central end-node unit *h0* to the mesh network system. An arbitrary defined end-node unit *hi* is connected to the network system through a switch unit *sw_i*, where *i = 1 to N* and **N is the total amount of switch units in the network excluding sw_0**. The mesh system is depicted in such a way as to categorise the interconnecting links as star topology link units (*lnk_s*), ring topology link units (*lnk_r*) and mesh topology link units (*lnk_m*). It is clear that the star and ring topologies are special subsets of the general mesh topology, an observation that helps to generalise and simplify the models. It can also be seen that the total amount of star topology links equal the amount of switches (*N*) in the mesh, the total amount of ring links are equal to *N-1* and the total amount of forward connecting mesh links depends on the relative position of *sw_i*, where *i* is an index indicting a reference switch position connecting *hi* into the corresponding ring topology. The amount of forward connecting mesh links at *sw_i* are given by the equation *N-(i+1)*. A redundancy controller service (represented by the spanning tree algorithm running between switches [21], [39]) detects and eliminates rings [32] by blocking active parallel paths, when a link fails it also detects this failure and unblocks a redundant or standby path.

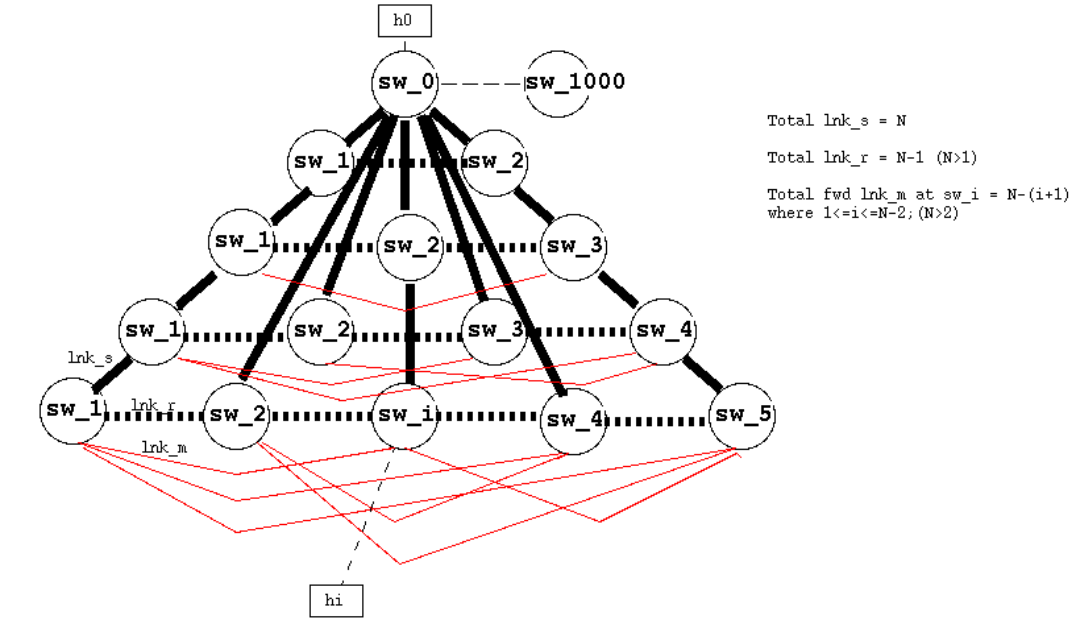

Figure 5.1. Mesh topology overview

Switches *sw_0* and *sw_i*, as well as the links connecting end-nodes *h0* and *hi* are common to all the topologies and these components and units connected in series with the rest of the network offers the same contribution to overall network reliability in all the topologies. During the preliminary modelling phase it was found these components and units are in fact the limiting contributors, by orders of magnitude, to the overall network reliability [35] - an important finding that will be discussed in more detail in Chapter 7. It has therefore been decided to model *sw_0* and *sw_i* using a failure rate of zero in all models so that the network failures contributed by interconnecting switches and links in different topologies of interest between the end-node connecting switches can be compared.

A basic assumption made in modelling is that the spanning tree algorithm that is performing the function of a redundancy controller will always select the minimal spanning tree [21], the least-cost or minimal available path discussed in [35] with system function represented by Equation 2.27, and after detecting connectivity loss due to a failed link or switch will detect and activate the next least-cost path that is available at the time of the failure, where the cost of a redundant path is based on the hop count. This is a valid assumption for Ethernet LANs with similar same-bandwidth links as explained in Section 2.2.1.4. It is also assumed that the redundant path is selected almost instantaneously, in the range of 0 - 3 seconds as is typical of RSTP.

In Section 2.4 the difference between cold and hot standby reliability modelling is discussed. It can be argued that in a spanning tree controlled network all circuits remain active and can therefore fail, the approach followed in this study however is that a link that is "blocked" and thus available as a standby link in event of the failure of the active path is in cold standby mode, i.e. this link cannot fail while in standby. This is assumed to be valid only for the blocked link as seen from reference end-node *hi* as indicated in Figure 5.2. The active switches and links that are not blocked upstream and downstream of *sw_i* that forms part of a standby path as seen from *hi* are considered to be actively carrying data from other end-nodes and therefore to be in hot standby mode therefore they may fail while not acting as an active path for carrying data from *hi*.

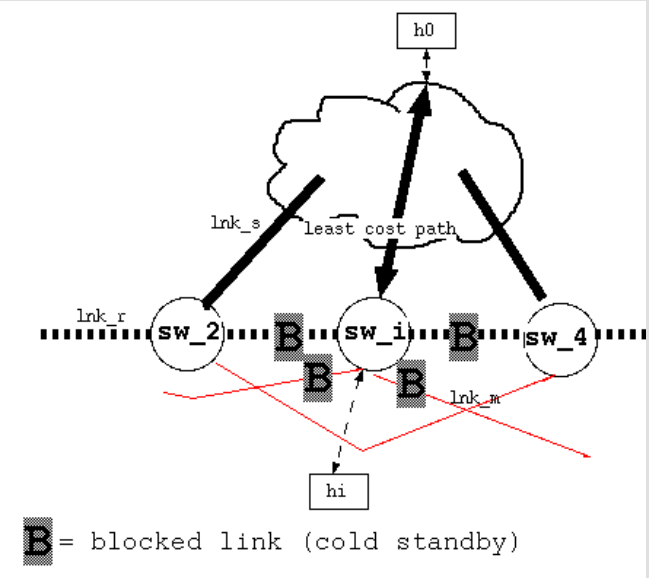

Figure 5.2. Mesh topology with blocked links

The repair strategy in Figure 5.3 and discussed in Section 2.4 used in the networking models are associated with the failure of a specific path, that is the repair facility is common for all the link units and switch components constituting a specific path. Accordingly each path has a separate repair facility. When any component or unit in a path fails the entire path is repaired to a working state. This is a realistic assumption given that a repair crew would probably ensure that a failed path is repaired in its totality - if a link unit failed and a switch component in a path failed more or less at the same time both would be repaired to a working state.

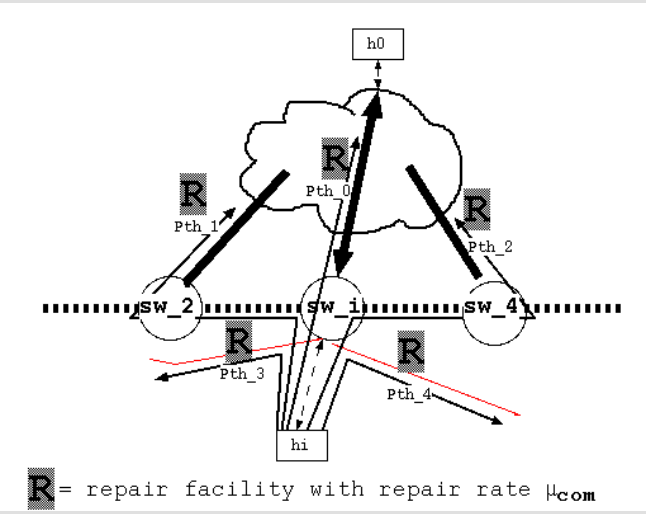

Figure 5.3. Mesh topology with shared repair facility for every path

This chapter introduces performance related metrics Bandwidth (B), Traffic (T) and performability (Section 2.5) metric Merit (M), where $M = B/T$ is a reward variable that represents the utilisation of a path. The traffic T is a measure of the data traffic flowing through a switch where a switch acts as a data traffic source. As indicated in Figure 5.4 the bandwidth B is the data

carrying capability of a specific path. When the traffic flowing through a set of serial connected switches matches the maximum bandwidth of the path connecting the set of switches to an end-node the corresponding merit figure *M* for that path to the end-node is 1. An under-subscribed path has a merit figure greater than 1 and an oversubscribed or congested path has a merit figure less than 1. The performability measure of interest are derived from the Markov model using the merit metric as weighting factor (Section 2.4). The reliability and corresponding performability can therefore be modelled for a given network topology and the derived results both in terms of availability and bandwidth efficiency measures of interest can be compared.

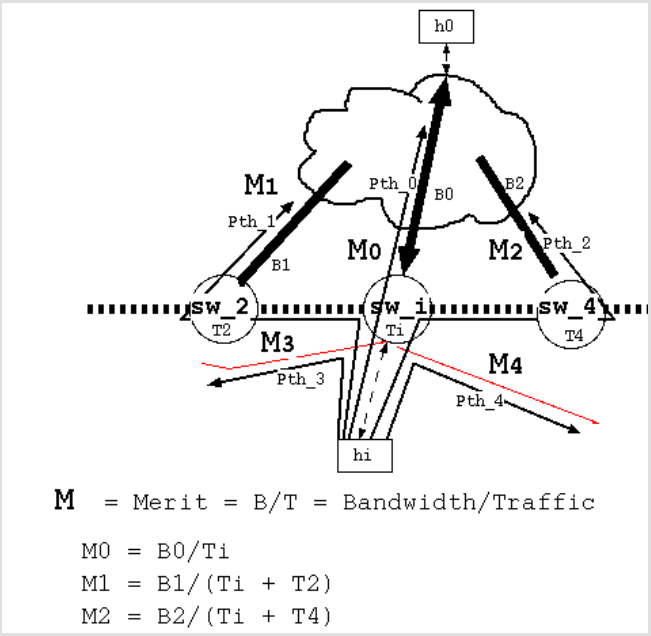

Figure 5.4. Mesh topology with performability metric M calculated for every path

5.2. Reliability and performance base data and assumptions

The following data set for reliability and performance metrics of components and units is used in the network models:

- Switch (*sw*) component *MTTF = 40 yr* based on industry data (Section 3.3).
- Link (*lnk*) unit *MTTF = 10 yr* as derived in Section 4.2.
- It is assumed that switch and link failures occur independently, in particular it is assumed that the redundancy controller does not fail.
- It is assumed that the spanning tree topology convergence time (maximum of three seconds) is negligible in comparison to the downtime caused by unit failures measured in hours.
- A root bridge *sw_0* is configured to be located at the reference end-node position *h0*.
- Every path (*Pth_j*) has its own repair facility that repairs the link at *MTTR = 1*.
- *sw_0* and *sw_i*, and the links from host *h0* and *hi* attached to *sw_0* and *sw_i* are assumed not to fail.

- The Traffic (*T*) through a switch is typically 1000 units representing 40% utilisation of a 24 x 100 unit per horizontal port switch.
- The Traffic (*T*) through a backup switch is typically 10 units representing 0.4 % utilisation of a 24 x 100 unit per horizontal port switch.
- The bandwidth (*B*) or data carrying capacity of a link is 1000 units.

5.3. Mesh topology model

5.3.1. Model specification

Figure 5.5 shows a simplified presentation of a generic mesh topology model. *Pth_j* denotes all the possible least cost paths between end-nodes *hi* and *h0* where *j = 0 to N - 1*, *N* being the amount of switches in the mesh excluding *sw_0*. The total amount of parallel least cost paths (direct link or single hop) consisting of either ring links or mesh links between two arbitrary end-nodes or hosts is therefore given by *N*, equal to the amount of switches in the mesh. In the simplified generic mesh topology model *sw_i* is substituted for *sw_1*, since the relative position *i*, where *i = 1 to N*, of the reference end-node switch *sw_i* connected to host *hi* has no influence on the model.

End-node *hi* is connected into the mesh through switch *sw_1* and end-node *h0* is connected to the mesh through switch *sw_0*, as discussed in Section 5.1 these components are assumed never to fail with failure rate $\lambda = 0$. For the purpose of modelling path *Pth_0* is the primary path - the path representing the star configuration link (*lnk_s*) with the lowest cost path directly linking *sw_i* to *sw_0*. It is also assumed, that when available, *Pth_0* will always be selected by the redundancy controller that is embedded in *Pth_0*, although in real world switch configuration the spanning tree protocol running between all switches would act as the redundancy controller. Failure in redundant (or parallel) paths are communicated to the redundancy controller that is configured to respond by activating either lowest cost *Pth_0* or the next available least cost path *Pth_j*. Each least cost path *Pth_j* consists of a ring path link (*lnk_r*) or a mesh path link (*lnk_m*) connecting to a switch (*sw*) that is then directly connected to *sw_0* through a star link (*lnk_s*). It is assumed that redundant blocked *lnk_r* or *lnk_m* are in cold standby mode (not operational) and cannot fail, however *sw* and *lnk_s* are operational and carrying data traffic at all times and can therefore fail even when not selected as least cost path to carry traffic between *hi* and *h0*.

A specific mesh topology model with a defined amount of switches (*N*) is derived from the generic mesh model and programmed as a Tangram model specification. During the preliminary modelling phase it was found that the mesh network *MTTF* exceeds 125 000 million 8hr units with a mesh model with three redundant paths (*N = 3*) already and it was therefore not necessary to model mesh networks with more than three switches based on the rationale of cost-effective design [22].

5.3.1.1. Mesh topology with *N=2*

The mesh topology model with *N=2* consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_0*) consisting of a star link (*lnk_s*) connecting *sw_1* to *sw_0*. *Pth_0* carries traffic from *sw_1*.
- Redundant least cost path (*Pth_1*) consisting of a ring/mesh link connecting *sw_1* to *sw_0* via ring or mesh link *lnk_r*/*lnk_m*, switch *sw* and star link *lnk_s*. *Pth_1* carries traffic from both *sw_1* and the redundant path switch *sw*.

The following performability data set is used:

- *T for sw_1 = 1000 units; T for sw = 1000 units.*
- *B for Pth_0 = 1000 units; B for Pth_1 = 1000 units.*

The *N=2* mesh topology network is described by "Model E1" and the model specification is shown in Figure 5.5:

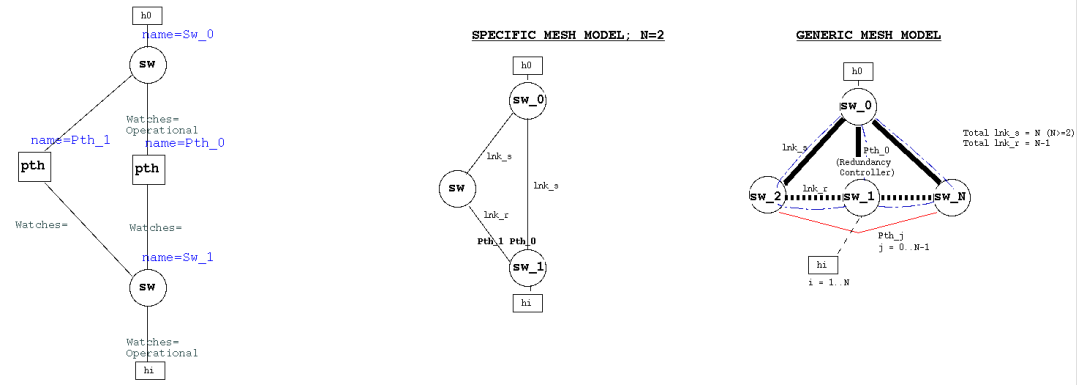

Figure 5.5. Model E1: Mesh network model with switches = 2

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.6.

```
MARKOV MODEL

State Variables
(columns no#)

1 - Pth_0.Slink_Operational
2 - Pth_0.Redundant_Path_Active
3 - Sw_1.Operational
4 - Pth_1.Slink_Operational
5 - Pth_1.Switch_Operational
6 - Pth_1.Rlink_Operational
7 - Pth_1.Path_Healthy
8 - Sw_0.Operational

States

S1 (1,0,.,1,1,0,1,.)
S2 (1,0,.,0,0,0,0,.)
S3 (1,0,.,0,1,0,0,.)
S4 (0,1,.,1,1,1,1,.)
S5 (0,0,.,0,0,0,0,.)
S6 (0,0,.,0,1,0,0,.)
S7 (0,0,.,1,1,0,0,.)
S8 (1,0,.,1,1,0,0,.)

State Transition Matrix
   S1->2 2.2800000000E-05
   S1->3 9.1300000000E-05
   S1->4 9.1300000000E-05
   S2->1 1.0000000000E+00
   S2->5 9.1300000000E-05
   S3->2 2.2800000000E-05
   S3->1 1.1000000000E+00
   S3->6 9.1300000000E-05
   S4->7 9.1300000000E-05
   S4->6 9.1300000000E-05
   S4->5 2.2800000000E-05
   S4->1 1.0000000000E+00
   S5->4 1.0000000000E+00
   S5->2 1.0000000000E+00
   S6->5 2.2800000000E-05
   S6->4 1.0000000000E+00
   S6->3 1.0000000000E+00
   S7->5 2.2800000000E-05
   S7->6 9.1300000000E-05
   S7->4 1.0000000000E+00
   S7->8 1.0000000000E+00
   S8->2 2.2800000000E-05
   S8->3 9.1300000000E-05
   S8->1 1.0000000000E+00
   S8->7 9.1300000000E-05

(λ sw = 2.2800000000E-05
λ lnk = 9.1300000000E-05)
μ com = 1.0000000000E+00)
```

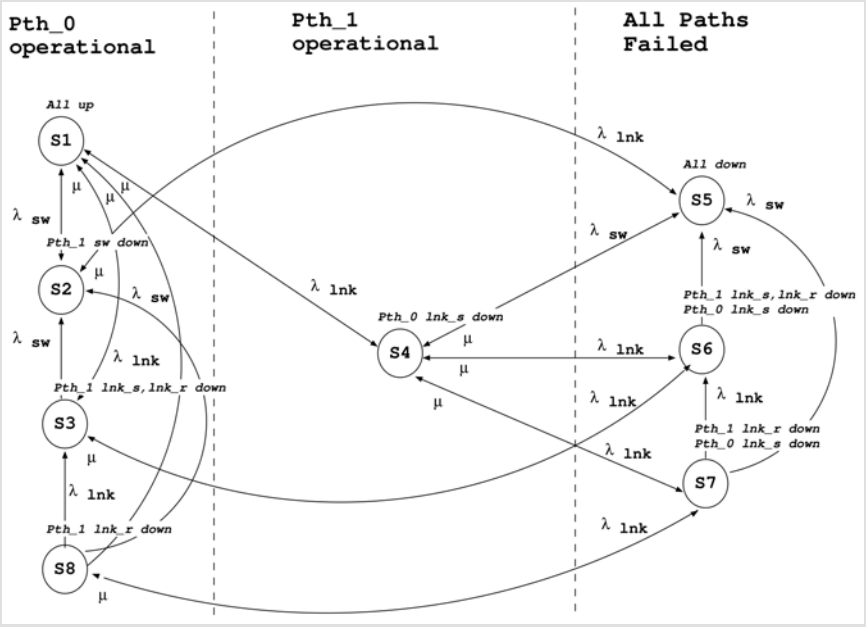

Figure 5.6. Markov Model E1: Mesh network model with switches = 2

The state transition matrix consisting of 8 states and 25 state transitions is used to construct a directed graph to help depict the various system failure states - states that correspond to an actual

network failure. In this model network failures correspond to the set of states that correspond to "All Paths Failed" i.e. both *Pth_0* and *Pth_1* are down.

5.3.1.2. Mesh topology with *N=3*

The mesh topology model with *N=3* consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_0*) consisting of a star link (*lnk_s*) connecting *sw_1* to *sw_0*. *Pth_0* carries traffic from *sw_1*.
- Redundant least cost path (*Pth_1*) consisting of a ring/mesh link connecting *sw_1* to *sw_0* via ring or mesh link *lnk_r*/*lnk_m*, switch *sw* and star link *lnk_s*. *Pth_1* carries traffic from both *sw_1* and the redundant path switch *sw*.
- Redundant least cost path (*Pth_2*) consisting of a ring/mesh link connecting *sw_1* to *sw_0* via ring or mesh link *lnk_r*/*lnk_m*, switch *sw* and star link *lnk_s*. *Pth_1* carries traffic from both *sw_1* and the redundant path switch *sw*.

The following performability data set is used:

- *T for sw_1 = 1000 units; T for sw = 1000 units*.
- *B for Pth_0 = 1000 units;* B *for Pth_1 = 1000 units; B for Pth_2 = 1000 units*.

The *N=3* mesh topology network is described by "Model E2" and the model specification is shown in Figure 5.7:

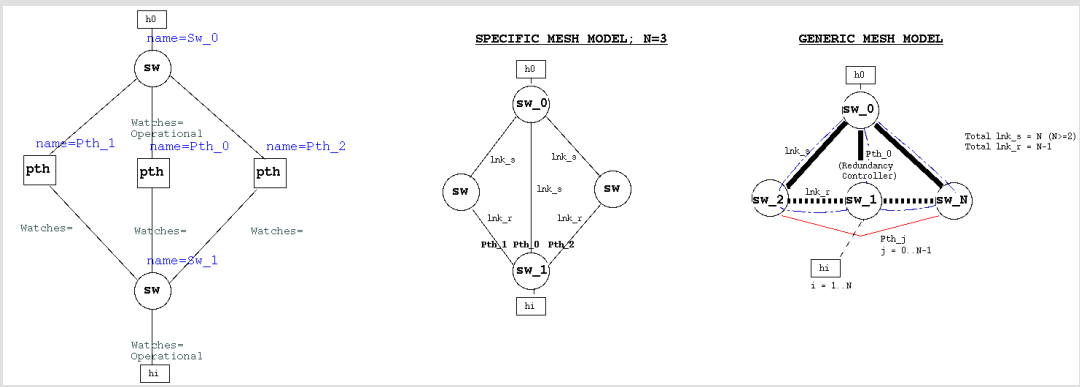

Figure 5.7. Model E2: Mesh network model with switches = 3

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.8 and Figure 5.9.

```
State Variables
(columns no#)

1 Pth_2.Slink_Operational
2 Pth_2.Switch_Operational
3 Pth_2.Rlink_Operational
4 Pth_2.Path_Healthy
5 Pth_0.Slink_Operational
6 Pth_0.Redundant_Path_Active
7 Sw_1.Operational
8 Pth_1.Slink_Operational
9 Pth_1.Switch_Operational
10 Pth_1.Rlink_Operational
11 Pth_1.Path_Healthy
12 Sw_0.Operational

States

S1  (1,1,0,1,1,0,1,1,1,0,1,1)
S2  (0,0,0,0,1,0,1,1,1,0,1,1)
S3  (0,1,0,0,1,0,1,1,1,0,1,1)
S4  (1,1,0,1,1,0,1,0,0,0,0,1)
S5  (1,1,0,1,1,0,1,0,1,0,0,1)
S6  (1,1,0,1,0,1,1,1,1,1,1,1)
S7  (0,0,0,0,1,0,1,0,0,0,0,1)
S8  (0,0,0,0,1,0,1,0,1,0,0,1)
S9  (0,0,0,0,0,1,1,1,1,1,1,1)
S10  (0,1,0,0,1,0,1,0,0,0,0,1)
S11  (0,1,0,0,1,0,1,0,1,0,0,1)
S12  (0,1,0,0,0,1,1,1,1,1,1,1)
S13  (1,1,1,1,0,1,1,0,0,0,0,1)
S14  (1,1,1,1,0,1,1,0,1,0,0,1)
S15  (1,1,1,1,0,1,1,1,1,0,0,1)
S16  (0,0,0,0,0,0,1,0,0,0,0,1)
S17  (0,0,0,0,0,0,1,0,1,0,0,1)
S18  (0,0,0,0,0,0,1,1,1,0,0,1)
S19  (0,1,0,0,0,0,1,0,0,0,0,1)
S20  (0,1,0,0,0,0,1,0,1,0,0,1)
S21  (0,1,0,0,0,0,1,1,1,0,0,1)
S22  (1,1,1,1,0,1,1,1,1,0,1,1)
S23  (1,1,0,0,0,0,1,0,0,0,0,1)
S24  (1,1,0,0,0,0,1,0,1,0,0,1)
S25  (1,1,0,1,1,0,1,1,1,0,0,1)
S26  (1,1,0,0,0,0,1,1,1,0,0,1)
S27  (0,0,0,0,1,0,1,1,1,0,0,1)
S28  (0,1,0,0,1,0,1,1,1,0,0,1)
S29  (1,1,0,0,0,1,1,1,1,1,1,1)
S30  (1,1,0,0,1,0,1,0,0,0,0,1)
S31  (1,1,0,0,1,0,1,0,1,0,0,1)
S32  (1,1,0,0,1,0,1,1,1,0,0,1)
S33  (1,1,0,0,1,0,1,1,1,0,1,1)
```

Figure 5.8. Markov Model E2: Mesh network model with switches = 3, states

State Transition Matrix

Sx	Sy	Transition_rate	Sx	Sy	Transition_rate
1	2	2.2800000000E-05	21	18	2.2800000000E-05
1	3	9.1300000000E-05	21	19	2.2800000000E-05
1	4	2.2800000000E-05	21	20	9.1300000000E-05
1	5	9.1300000000E-05	21	12	1.0000000000E+00
1	6	9.1300000000E-05	21	28	1.0000000000E+00
2	7	2.2800000000E-05	21	15	1.0000000000E+00
2	8	9.1300000000E-05	22	13	2.2800000000E-05
2	9	9.1300000000E-05	22	14	9.1300000000E-05
2	1	1.0000000000E+00	22	1	1.0000000000E+00
3	2	2.2800000000E-05	22	29	9.1300000000E-05
3	10	2.2800000000E-05	22	12	9.1300000000E-05
3	11	9.1300000000E-05	22	9	2.2800000000E-05
3	12	9.1300000000E-05	23	16	2.2800000000E-05
3	1	1.0000000000E+00	23	19	9.1300000000E-05
4	7	2.2800000000E-05	23	29	1.0000000000E+00
4	10	9.1300000000E-05	23	30	1.0000000000E+00
4	1	1.0000000000E+00	23	13	1.0000000000E+00
4	13	9.1300000000E-05	24	17	2.2800000000E-05
5	8	2.2800000000E-05	24	20	9.1300000000E-05
5	11	9.1300000000E-05	24	23	2.2800000000E-05
5	4	2.2800000000E-05	24	29	1.0000000000E+00
5	1	1.0000000000E+00	24	31	1.0000000000E+00
5	14	9.1300000000E-05	24	14	1.0000000000E+00
6	9	2.2800000000E-05	25	27	2.2800000000E-05
6	12	9.1300000000E-05	25	28	9.1300000000E-05
6	15	9.1300000000E-05	25	4	2.2800000000E-05
6	14	9.1300000000E-05	25	5	9.1300000000E-05
6	13	2.2800000000E-05	25	1	1.0000000000E+00
6	1	1.0000000000E+00	25	15	9.1300000000E-05
7	2	1.0000000000E+00	26	18	2.2800000000E-05
7	16	9.1300000000E-05	26	21	9.1300000000E-05
7	4	1.0000000000E+00	26	23	2.2800000000E-05
8	7	2.2800000000E-05	26	24	9.1300000000E-05
8	2	1.0000000000E+00	26	29	1.0000000000E+00
8	17	9.1300000000E-05	26	32	1.0000000000E+00
8	5	1.0000000000E+00	26	15	1.0000000000E+00
9	18	9.1300000000E-05	27	7	2.2800000000E-05
9	17	9.1300000000E-05	27	8	9.1300000000E-05
9	16	2.2800000000E-05	27	2	1.0000000000E+00
9	2	1.0000000000E+00	27	18	9.1300000000E-05
9	6	1.0000000000E+00	27	25	1.0000000000E+00
10	7	2.2800000000E-05	28	27	2.2800000000E-05
10	3	1.0000000000E+00	28	10	2.2800000000E-05
10	19	9.1300000000E-05	28	11	9.1300000000E-05
10	4	1.0000000000E+00	28	3	1.0000000000E+00
11	8	2.2800000000E-05	28	21	9.1300000000E-05
11	10	2.2800000000E-05	28	25	1.0000000000E+00
11	3	1.0000000000E+00	29	9	2.2800000000E-05
11	20	9.1300000000E-05	29	12	9.1300000000E-05
11	5	1.0000000000E+00	29	26	9.1300000000E-05
12	9	2.2800000000E-05	29	24	9.1300000000E-05
12	21	9.1300000000E-05	29	23	2.2800000000E-05
12	20	9.1300000000E-05	29	33	1.0000000000E+00
12	19	2.2800000000E-05	29	6	1.0000000000E+00
12	3	1.0000000000E+00	30	7	2.2800000000E-05
12	6	1.0000000000E+00	30	10	9.1300000000E-05
13	22	1.0000000000E+00	30	33	1.0000000000E+00
13	4	1.0000000000E+00	30	23	9.1300000000E-05
13	23	9.1300000000E-05	30	4	1.0000000000E+00
13	19	9.1300000000E-05	31	8	2.2800000000E-05
13	16	2.2800000000E-05	31	11	9.1300000000E-05
14	13	2.2800000000E-05	31	30	2.2800000000E-05
14	22	1.0000000000E+00	31	33	1.0000000000E+00
14	5	1.0000000000E+00	31	24	9.1300000000E-05
14	24	9.1300000000E-05	31	5	1.0000000000E+00
14	20	9.1300000000E-05	32	27	2.2800000000E-05
14	17	2.2800000000E-05	32	28	9.1300000000E-05
15	13	2.2800000000E-05	32	30	2.2800000000E-05
15	14	9.1300000000E-05	32	31	9.1300000000E-05
15	22	1.0000000000E+00	32	33	1.0000000000E+00
15	25	1.0000000000E+00	32	26	9.1300000000E-05
15	26	9.1300000000E-05	32	25	1.0000000000E+00
15	21	9.1300000000E-05	33	2	2.2800000000E-05
15	18	2.2800000000E-05	33	3	9.1300000000E-05
16	9	1.0000000000E+00	33	30	2.2800000000E-05
16	7	1.0000000000E+00	33	31	9.1300000000E-05
16	13	1.0000000000E+00	33	29	9.1300000000E-05
17	16	2.2800000000E-05	33	1	1.0000000000E+00
17	9	1.0000000000E+00			
17	8	1.0000000000E+00			
17	14	1.0000000000E+00			
18	16	2.2800000000E-05			
18	17	9.1300000000E-05			
18	9	1.0000000000E+00			
18	27	1.0000000000E+00			
18	15	1.0000000000E+00			
19	16	2.2800000000E-05			
19	12	1.0000000000E+00			
19	10	1.0000000000E+00			
19	13	1.0000000000E+00			
20	17	2.2800000000E-05			
20	19	2.2800000000E-05			
20	12	1.0000000000E+00			
20	11	1.0000000000E+00			
20	14	1.0000000000E+00			

(λ_{sw} = 2.2800000000E-05
(λ_{lnk} = 9.1300000000E-05)
(μ_{com} = 1.0000000000E+00)

Figure 5.9. Markov Model E2: Mesh network model with switches = 3, transition matrix

The state transition matrix consists of 33 states and 173 state transitions. It is not useful to construct a directed graph presentation since the state transition matrix is large and the graph does not help to elucidate the underlying Markov model.

5.3.2. Model solution

5.3.2.1. Mesh topology with *N=2*

The transient solution for reliability *R(t)* as a function of 8hr time units for the above *N=2* mesh topology network model is shown in Appendix A, Figure A.14. The transient solution for expected lifetime *L(t)* is shown in Figure A.15.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 34294071* or 31298 years. (5.1)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.99999997* and the performability metric *M* is calculated from the state-space matrix as *M = 1.000*. (5.2)

5.3.2.2. Mesh topology with *N=3*

The transient solution for reliability *R(t)* as a function of 8hr time units for the above *N=3* mesh topology network model is shown in Appendix A, Figure A.16. The transient solution for expected lifetime *L(t)* is shown in Figure A.17.

The *MTTF* is derived from the steady-state value of *L(t)* as $MTTF \geq 250000000$ or 228155 years. (5.3)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be $A \geq 0.999999996$ and the performability metric *M* is calculated from the state-space matrix as *M = 1.000*. (5.4)

5.4. Ring topology model

5.4.1. Model specification

Figure 5.11 shows a simplified presentation of a generic ring topology model. *Pth_j* denotes the two possible least cost paths between end-nodes *hi* and *h0* where *j=1 to 2*.

End-node *hi* is connected into the ring through switch *sw_i* and end-node *h0* is connected into the ring through switch *sw_0*, as discussed in Section 5.1 these components are assumed never to fail with failure rate $\lambda = 0$. The generic ring model is based on the assumption that the redundancy controller, identified as a service entity *Pth_0*, blocks an uplink port (*lnk_r0*) at the node located at a position *N/2 + 1* in the ring, creating two parallel redundant paths to *h0* from the node position

at *Pth_0*. For the purpose of modelling, path *Pth_1* is the upstream section, relative to reference end-node *hi*, of the primary path or the path representing the lowest cost path from *sw_i* to *sw_0*. Path *Pth_1* is the downstream section, relative to reference end-node *hi*, of the secondary path i.e. the path representing the lower cost path from *sw_i* to *sw_0*. It is assumed, that when available, *Pth_2* will be selected by the redundancy controller when *Pth_1* fails. *Pth_1* and *Pth_2* consist of the blocking ring link *lnk_r0* modelled to be connected directly to *sw_i* although in fact situated at the *Pth_0* position, and ring path links (*lnk_r*) interconnecting one or more switches (*sw*) in a serial path between *sw_i* and *sw_0*.

It is assumed that the redundant blocked link *lnk_r0* of *Pth_2* is in cold standby mode (not operational) and cannot fail, however the serial switches (*sw*) and links (*lnk_r*) are operational and carrying data traffic at all times and can therefore fail even when not selected as least cost path to carry traffic between *hi* and *h0*.

It must be noted that the generic ring model is simplified in order to avoid a large state-space matrix introduced by conditional failure of individual serial links. As result of this simplification an already failed serial link (*lnk_r*) can fail even when it has failed already. Given the very large mean time to failure (*MTTF = 10856*) for links in comparison to mean time to repair (*MTTR = 1*) of a link this simplification should result in a negligible deviation from the more accurate model. This simplification is necessary because it is computationally problematic to solve models with large amount of serial links given independently failure state specific recoveries for each individual serial link. Similarly, in order to restrict the size of the state-space matrix, the model is simplified with an assumption that during normal operation *Pth_1* data traffic is equal to the traffic generated by *N/2+1* switches, and that on *Pth_1* failure the traffic in redundant *Pth_2* is equal to traffic generated by *N/2 + 1* switches. The influence of these simplifications will be explored and discussed in Section 7.4.

A specific ring topology model with a defined amount of switches (*N*) is derived from the generic ring model and programmed as a Tangram model specification. A generalised ring model is then inferred by analysing the state-space matrices generated from the above models.

It is assumed that *sw_0* is configured to be the root bridge. Depicted in Figure 5.10, for *N>2*, *Pth_1* and *Pth_2* traverse different amount of serial connected switches and links depending on the position (*i*) of the reference end-node *hi* in the ring, models are therefore constructed for two end-node ring positions:

- at *i = 1* the first node in the ring with a least cost path directly connecting *sw_i* with *sw_0* through *Pth_1*.

- at *i = (N/2 +1)* the central node in the ring with equal cost paths connecting *sw_i* with *sw_0* through *Pth_1* and *Pth_2*.

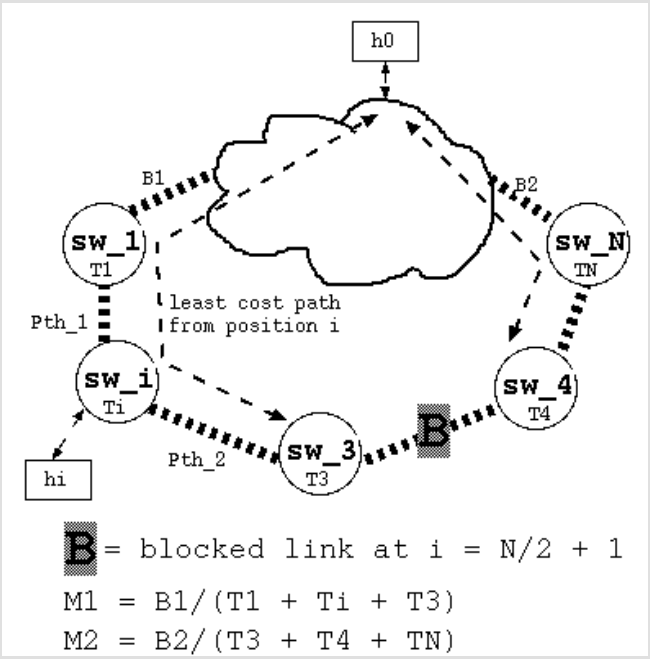

Figure 5.10. Ring topology with performability metric M calculated for both paths

5.4.1.1. Ring topology with *N=2*

The ring topology model with *N=2* consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_1*) consisting of a ring link (*lnk_r0*) connecting *sw_1* to *sw_0*. *Pth_1* carries traffic from *sw_1*.
- Redundant least cost path (*Pth_2*) consisting of a ring link (*lnk_r0*) connecting *sw_1* to *sw* and another ring link (*lnk_r*) in series to *sw_0*. *Pth_2* carries traffic from both *sw_1* and the redundant path switch *sw*.

The following performability data set is used:

- *T for sw_1 = 1000 units; T for sw = 1000 units*.
- *B for Pth_1 = 1000 units; B for Pth_2 = 1000 units*.

The *N=2* ring topology network is described by "Model F1" and the model specification is shown in Figure 5.11:

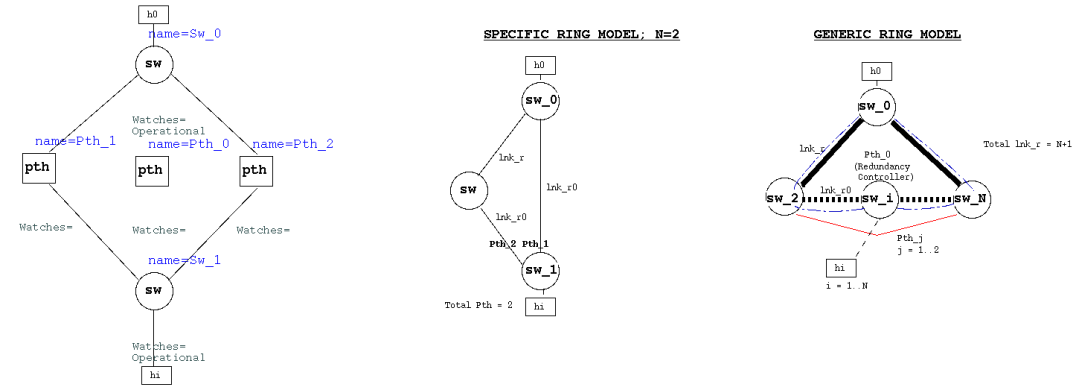

Figure 5.11. Model F1: Ring network model with switches = 2

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.12.

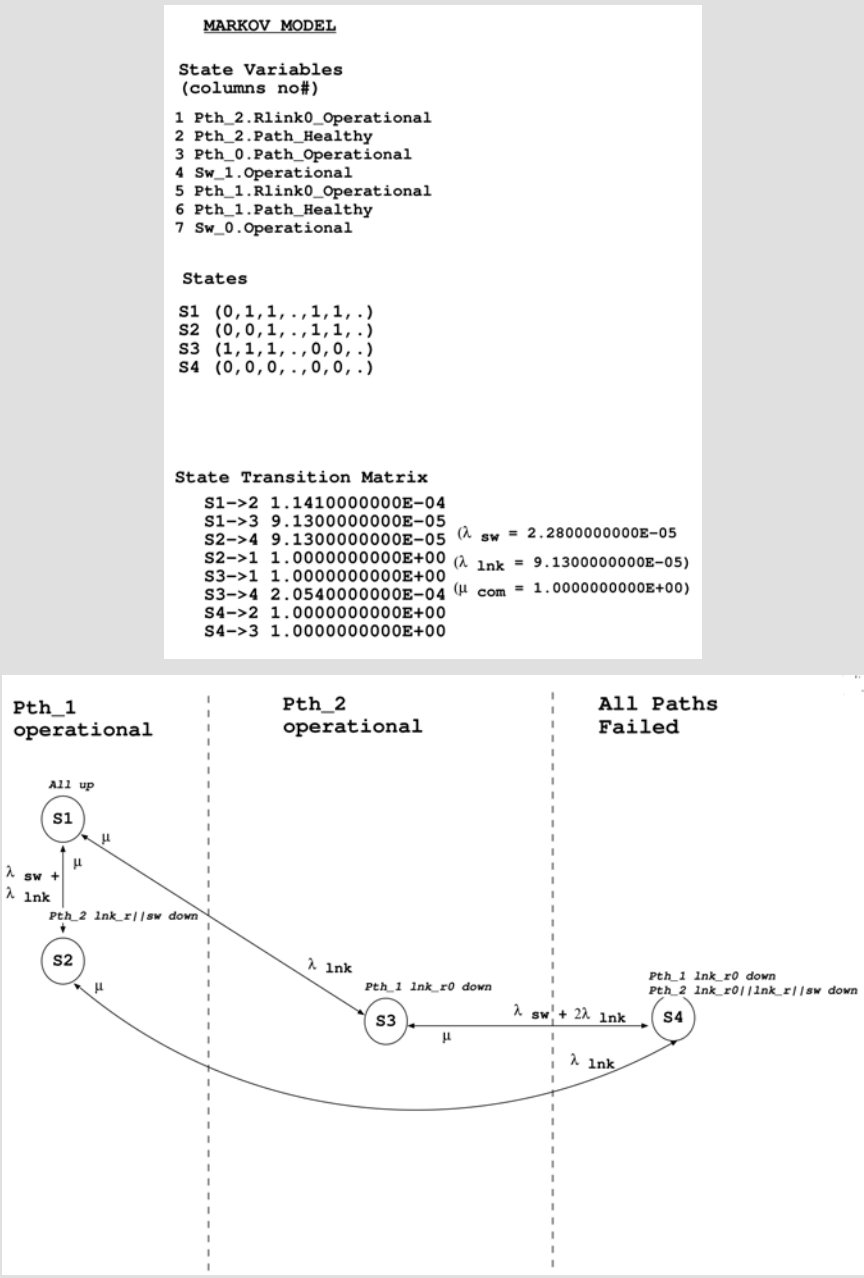

Figure 5.12. Markov Model F1: Ring network model with switches = 2

The state transition matrix consisting of 4 states and 8 state transitions is used to construct a directed graph to help depict the various system failure states - states that correspond to an actual network failure. In this model network failures correspond to the set of states that correspond to "All Paths Failed" i.e. both *Pth_1* and *Pth_2* are down.

5.4.1.2. Ring topology with *N=3, i=1*

The ring topology model with *N=3, i=1*, consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_1*) consisting of a ring link (*lnk_r0*) connecting *sw_1* to *sw_0*. *Pth_1* carries traffic from *sw_1* and switch *sw* located downstream.

- Redundant least cost path (*Pth_2*) consisting of a ring link (*lnk_r0*) connecting *sw_1* to two times a switch (*sw*) and ring link (*lnk_r*) combination in series to *sw_0*. *Pth_2* carries traffic from the two serial redundant path switches *sw*.

The following performability data set is used:

- *T for sw_1 = 1000 units; T for sw = 1000 units*.
- *B for Pth_1 = 1000 units; B for Pth_2 = 1000 units*.

The *N=3, i=1* ring topology network is described by "Model F2" and the model specification is shown in Figure 5.13:

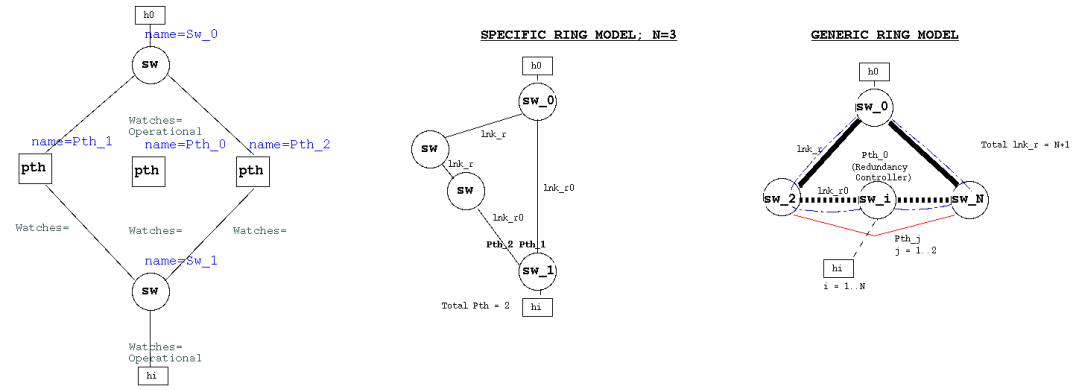

Figure 5.13. Model F2: Ring network model with switches = 3, i = 1

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.14.

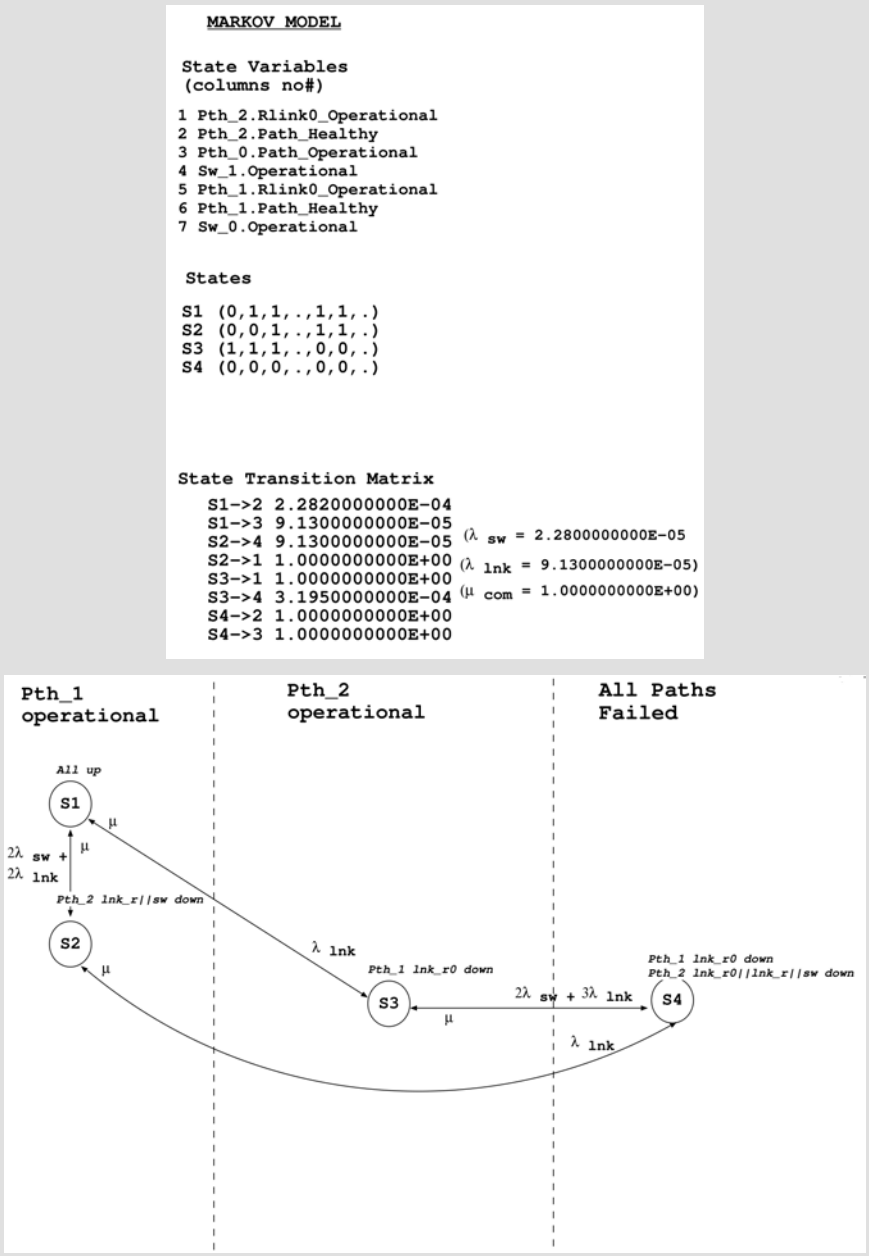

Figure 5.14. Markov Model F2: Ring network model with switches = 3, i = 1

The state transition matrix consisting of 4 states and 8 state transitions is used to construct a directed graph to help depict the various system failure states - states that correspond to an actual network failure. In this model network failures correspond to the set of states that correspond to "All Paths Failed" i.e. both *Pth_1* and *Pth_2* are down.

5.4.1.3. Ring topology with *N=3, i=2*

The ring topology model with *N=3, i=2*, consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_1*) consisting of a ring link (*lnk_r0*) connecting *sw_2* to *sw* and another ring link (*lnk_r*) in series to *sw_0*. *Pth_1* carries traffic from both *sw_2* and switch *sw*.

- Redundant least cost path (*Pth_2*) consisting of a ring link (*lnk_r0*) connecting *sw_2* to *sw* and another ring link (*lnk_r*) in series to *sw_0*. *Pth_2* carries traffic from both *sw_2* and switch *sw*.

The following performability data set is used:

- *T for sw_2 = 1000 units; T for sw = 1000 units*.
- *B for Pth_1 = 1000 units; B for Pth_2 = 1000 units*.

The *N=3, i=2* ring topology network is described by "Model F3" and the model specification is shown in Figure 5.15:

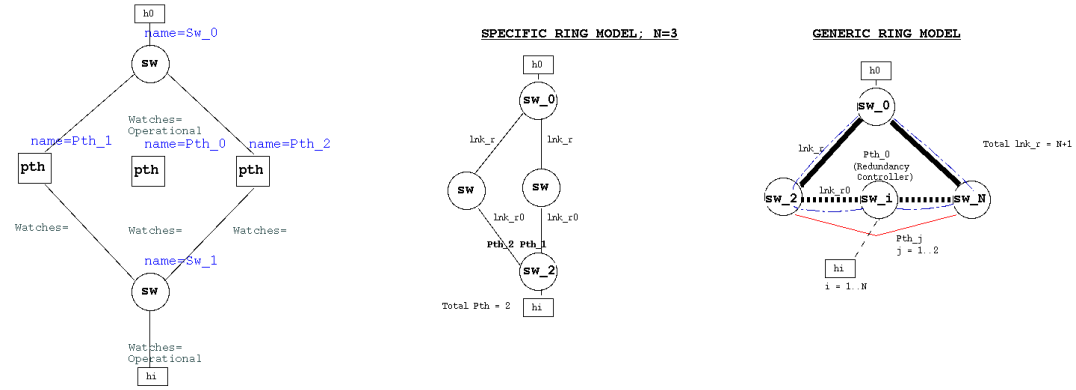

Figure 5.15. Model F3: Ring network model with switches = 3, i = 2

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.16.

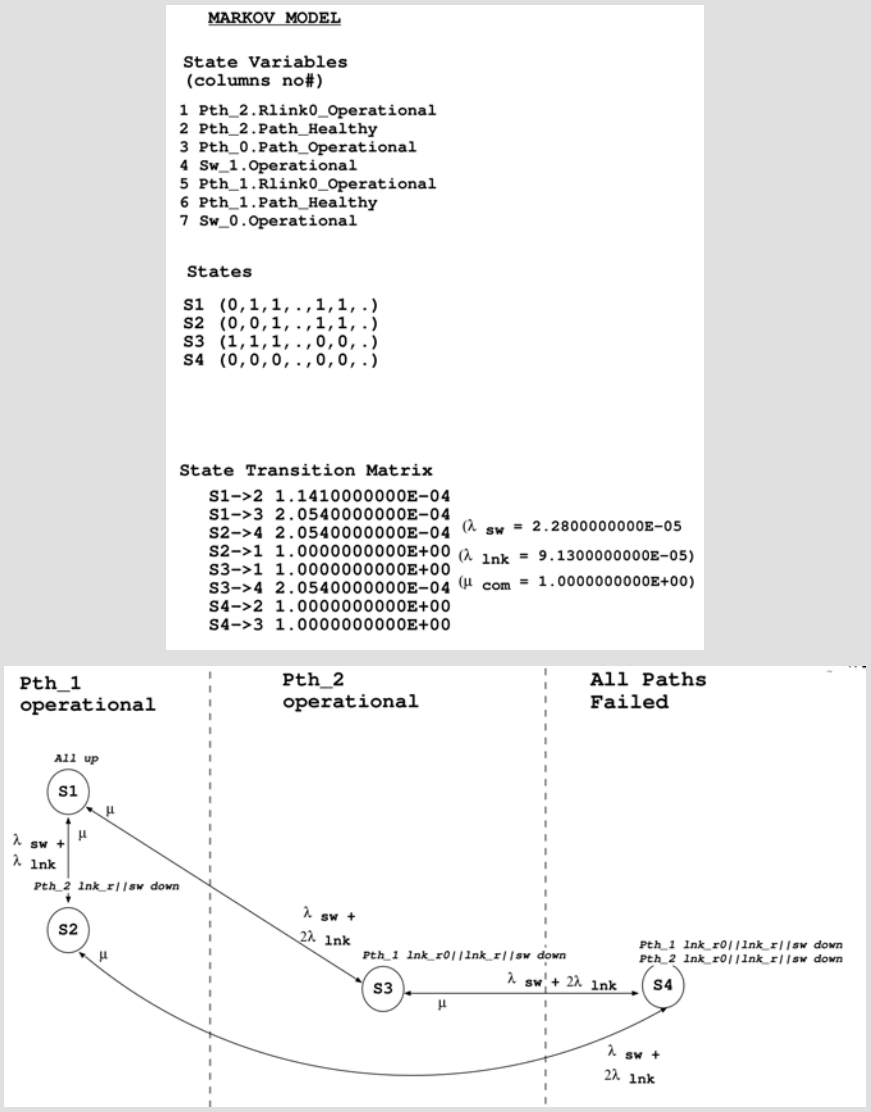

Figure 5.16. Markov Model F3: Ring network model with switches = 3, i = 2

The state transition matrix consisting of 4 states and 8 state transitions is used to construct a directed graph to help depict the various system failure states - states that correspond to an actual network failure. In this model network failures correspond to the set of states that correspond to "All Paths Failed" i.e. both *Pth_1* and *Pth_2* are down.

5.4.1.4. General ring topology model

By constructing ring models for different values of *N* and *i* and inspecting the state-space matrix generated it is evident that the ring model can be generalised in terms of the amount of switches or hops in the ring and the relative position of the entry-node attached to reference end-node *hi* in the ring. The position of the reference node *hi* determines the amount of hops between the entry-node and *sw_0* - where a hop consists of a serial switch *sw* and serial link *lnk_r*. In general terms then the ring topology model with *switches = N*, and reference node position *hi* with *i = 1 to N/2 + 1*, consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_1*) from *sw_i* to *sw_0*, consisting of *i - 1* hops. *Pth_1* carries traffic from *sw_i* and *N/2* downstream switches.

- Redundant least cost path (*Pth_2*) consisting of a ring link (*lnk_r0*) connecting *sw_i* to *sw_0* through *N - i* hops. *Pth_2* carries traffic from *sw_i* and *N/2* downstream switches.

The following performability data set is used:

- *T for sw_i = 1000 units; T for sw = 1000 units*.
- *B for Pth_1 = 1000 units; B for Pth_2 = 1000 units*.

The *swithes = N*, where *i = 1 to N/2 + 1* ring topology network is described by "Model F20" and the model specification is shown in Figure 5.17:

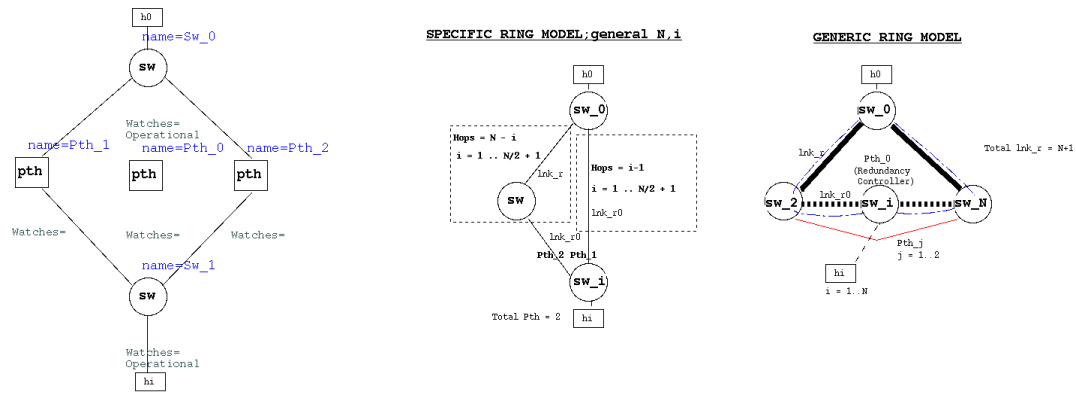

Figure 5.17. Model F20: Ring network model with switches = N, i = 1 to N/2 + 1

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.18.

MARKOV MODEL

State Variables
(columns no#)

1 Pth_2.Rlink0_Operational
2 Pth_2.Path_Healthy
3 Pth_0.Path_Operational
4 Sw_1.Operational
5 Pth_1.Rlink0_Operational
6 Pth_1.Path_Healthy
7 Sw_0.Operational

States

S1 (0,1,1,.,1,1,.)
S2 (0,0,1,.,1,1,.)
S3 (1,1,1,.,0,0,.)
S4 (0,0,0,.,0,0,.)

State Transition Matrix

S1->2 $(N - i) * (\lambda_{sw} + \lambda_{lnk})$
S1->3 $(i-1) * (\lambda_{sw} + \lambda_{lnk}) + \lambda_{lnk}$
S2->4 $(i-1) * (\lambda_{sw} + \lambda_{lnk}) + \lambda_{lnk}$
S2->1 μ
S3->1 μ
S3->4 $(N - i) * (\lambda_{sw} + \lambda_{lnk}) + \lambda_{lnk}$
S4->2 μ
S4->3 μ

$(\lambda_{sw} = 2.2800000000E-05$
$(\lambda_{lnk} = 9.1300000000E-05)$
$(\mu_{com} = 1.0000000000E+00)$

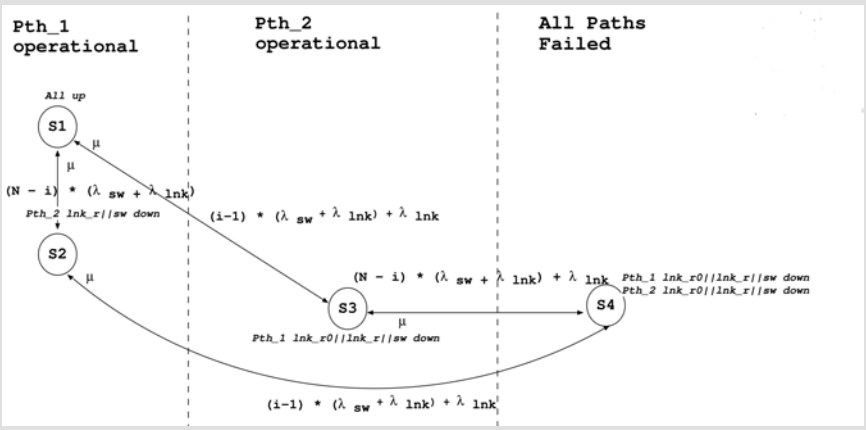

Figure 5.18. Markov Model F20: Ring network model with switches = N, i = 1 to N/2 + 1

The state transition matrix consisting of 4 states and 8 state transitions is used to construct a directed graph to help depict the various system failure states - states that correspond to an actual network failure. In this model network failures correspond to the set of states that correspond to "All Paths Failed" i.e. both *Pth_1* and *Pth_2* are down.

5.4.2. Model solution

5.4.2.1. Ring topology with *N=2*

The transient solution for reliability *R(t)* as a function of 8hr time units for the above *N=2* ring topology network model is shown in Appendix A, Figure A.19. The transient solution for expected lifetime *L(t)* is shown in Figure A.20.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 34294071* or 31298 years. (5.5)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.99999997* and the performability metric *M* is calculated from the state-space matrix as *M = 0.999*. (5.6)

5.4.2.2. Ring topology with *switches = N, i = 1, i = N/2 + 1*

By examining the generic ring model depicted in Figure 5.17 it is clear that the switch at position *i = 1* with only one uplink *lnk_ro* to *sw_0* is expected to have the highest *MTTF* because all other switches downstream are connected through more units and failure rates for units in serial can be added up as indicated by Equation 2.17. Similarly the switch at position *i = N/2 + 1* is expected to have the lowest *MTTF* because it is connected through *N/2* hops consisting of several switches and links in series to *sw_0*. Model solutions for different values of *N = amount of switches* are therefore calculated for switch positions *i = 1*, representing the best case reliability, and *i = N/2 + 1* representing the worse case reliability.

The transient solutions for reliability *R(t)* and expected lifetime *L(t)* as a function of 8hr time units for the solution of the generic topology network model are shown for selected values of *N* and *i* in Section A.8.

The *MTTF*, steady-state availability and performability metric *M* calculated for different amount of switches in the ring (*N*) and switch position in the ring at *i = 1* as well as at *i = N/2 + 1* are shown in Table 5.1.

Table 5.1. Summary of reliability metrics for generic ring model

Model	N	i	MTTF	Av	M
F2	3	1	20008873	0.99999995	0.999
F3	3	2	15246024	0.99999993	0.500
F4	4	1	14126494	0.99999993	0.500
F5	4	2	8895352	0.9999999	0.500
F6	5	1	10917865	0.99999991	0.333
F7	5	3	5719563	0.9999998	0.333
F8	9	1	5722854	0.9999998	0.200
F9	9	5	1821183	0.9999995	0.200
F10	17	1	2935243	0.9999997	0.111
F11	17	9	521063	0.999998	0.111
F12	33	1	1489805	0.9999993	0.059
F13	33	17	140181	0.999993	0.059
F14	65	1	753620	0.999999	0.030
F15	65	33	36541	0.99997	0.030
F16	129	1	382093	0.999997	0.015
F17	129	65	9406	0.9999	0.015
F18	257	1	195475	0.999995	0.008
F19	257	129	2424	0.9996	0.008

5.5. Star topology model

5.5.1. Model specification

Figure 5.19 shows a presentation of a star topology model. *Pth_j*, where *j=0*, is the only possible path between two arbitrary end-nodes or hosts. In the simplified generic star model *sw_i* is substituted for *sw_1*, since the relative position *i*, where *i = 1 to N*, of the reference end-node switch *sw_i* connected to host *hi* has no influence on the model.

End-node *hi* is connected into the star through switch *sw_1* and end-node *h0* is connected to the star through switch *sw_0*, as discussed in Section 5.1 these components are assumed never to fail with failure rate $\lambda = 0$. For the purpose of modelling path *Pth_0* is the primary and only path, that is the path representing the star configuration link (*lnk_s*) with the lowest cost directly linking *sw_1* to *sw_0*. There is no redundancy and no need for a redundancy controller.

A specific star topology model can be constructed for all possible values of *N* since there are no redundant paths that depend on the amount of switches. The specific star topology is programmed as a Tangram model specification.

The star topology model with *N=All* consists of the following path component:

- Only available (*Pth_0*) consisting of a star link (*lnk_s*) connecting *sw_1* to *sw_0*. *Pth_0* carries traffic from *sw_1*.

The following performability data set is used:

- *T for sw_1 = 1000 units*.
- *B for Pth_0 = 1000 units*.

The *N=All* star topology network is described by "Model G" and the model specification is shown in Figure 5.19:

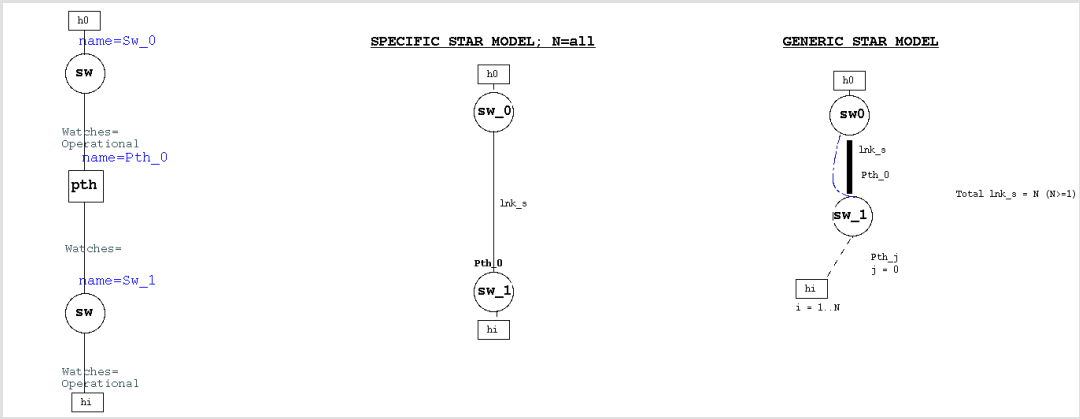

Figure 5.19. Model G: Star network model

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.20.

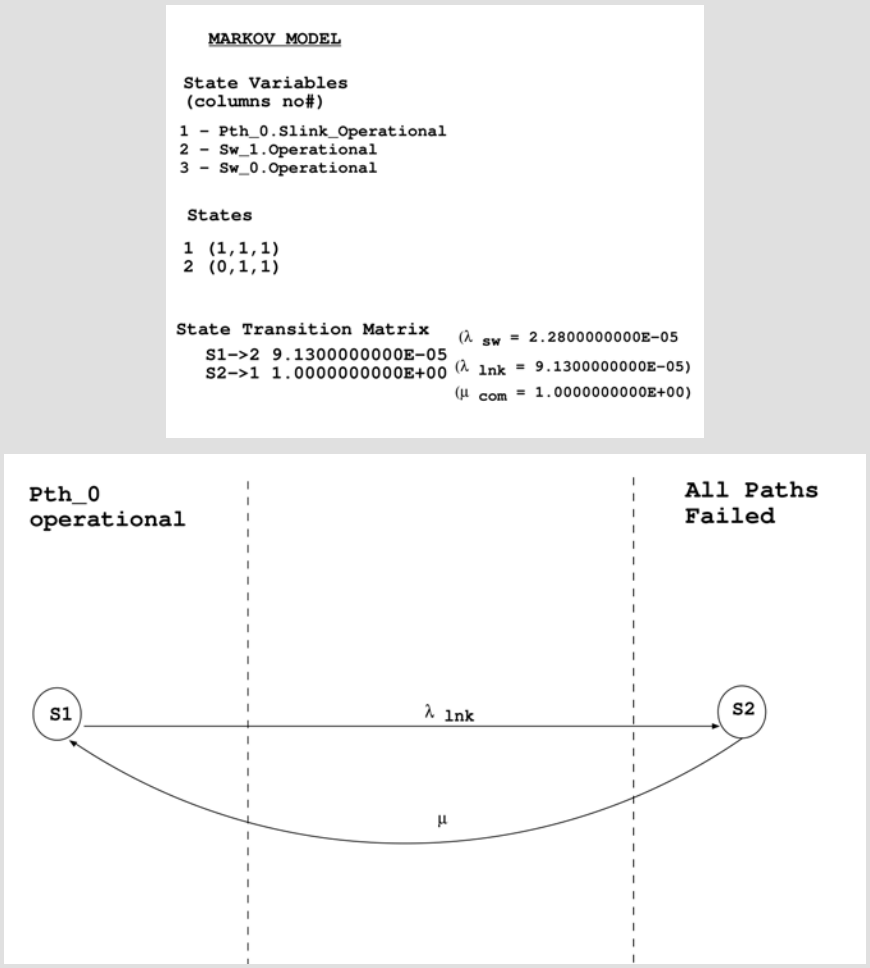

Figure 5.20. Markov Model G: Star network model

The state transition matrix consisting of 2 states and 2 state transitions is used to construct a directed graph to help depict the system failure state i.e *Pth_0* is down.

5.5.2. Model solution

The transient solution for reliability *R(t)* as a function of 8hr time units for the above *N=All* star topology network model is shown in Appendix A, Figure A.31. The transient solution for expected lifetime *L(t)* is shown in Figure A.17.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 34294071* or 10 years. (5.7)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.99991* and the performability metric *M* is calculated from the state-space matrix as *M = 1.000*. (5.8)

5.6. Hierarchical mesh topology model

5.6.1. Model specification

Figure 5.21 shows a presentation of a hierarchical mesh topology model. The hierarchical mesh is modelled as a special case of the mesh topology model with *N=2* in Figure 5.5. *Pth_j* denotes the two possible paths between end-nodes *hi* and *h0* where *j=0 to 1*. In the simplified generic mesh topology model *sw_i* is substituted for *sw_1*, since the relative position *i*, where *i = 1 to N*, of the reference end-node switch *sw_i* connected to host *hi* has no influence on the model.

End-node *hi* is connected into the mesh through switch *sw_1* and end-node *h0* is connected to the mesh through switch *sw_0*, as discussed in Section 5.1 these components are assumed never to fail with failure rate $\lambda = 0$. This hierarchical model introduces *sw_1000* that is a backup switch for *sw_0* and located at the same position in the mesh as *sw_0* shown in Figure 5.1. For the purpose of modelling path *Pth_0* is the primary path or the path representing the star configuration link (*lnk_s*) with the lowest cost path directly linking *sw_1* to *sw_0*. It is also assumed, that when available, *Pth_0* will always be selected by the redundancy controller that is embedded in *Pth_0*, although in real world switch configuration the spanning tree protocol running between all switches would act as the redundancy controller. Failure in the two redundant (or parallel) paths are communicated to the redundancy controller that is configured to respond by activating either lowest cost *Pth_0* or the higher cost path *Pth_1*. The higher cost path *Pth_1* consist of a ring path link (*lnk_r*) connecting to backup switch (*sw_1000*) that is then directly connected to *sw_0* through a star link (*lnk_s*). It is assumed that redundant blocked *lnk_r* is in cold standby mode (not operational) and cannot fail, however *sw_1000* and *lnk_s* are operational and carrying data traffic at all times and can therefore fail even when not selected as least cost path to carry traffic between *hi* and *h0*. It is assumed that since *sw_1000* is a backup switch - it would only be carrying minimal traffic at any point under normal operation because *Pth_0* data traffic from end-nodes would be flowing through *sw_0*.

A specific hierarchical mesh topology model can be constructed for all possible values of *N* since there are no redundant paths that depend on the amount of switches in the network where for every end-node there are only two redundant paths irrespective of the amount of switches *N*. The specific hierarchical mesh topology is programmed as a Tangram model specification.

The mesh topology model with *N=All* consists of the following path components that are connected in parallel:

- Lowest cost path (*Pth_0*) consisting of a star link (*lnk_s*) connecting *sw_1* to *sw_0*. *Pth_0* carries traffic from *sw_1*.
- Redundant path (*Pth_1*) consisting of a ring link connecting *sw_1* to *sw_0* via ring link *lnk_r*, switch *sw* and star link *lnk_s*. *Pth_1* carries traffic from both *sw_1* and the redundant path switch *sw*, however switch *sw* carries minimal traffic since under normal network conditions it serves as a backup switch (Section 5.2).

The following performability data set is used:

- *T for sw_1 = 1000 units; T for sw = 1000 units; T for sw_1000 = 10 units*.
- *B for Pth_0 = 1000 units; B for Pth_1 = 1000 units*.

The *N=All* hierarchical mesh topology network is described by "Model H" and the model specification is shown in Figure 5.21:

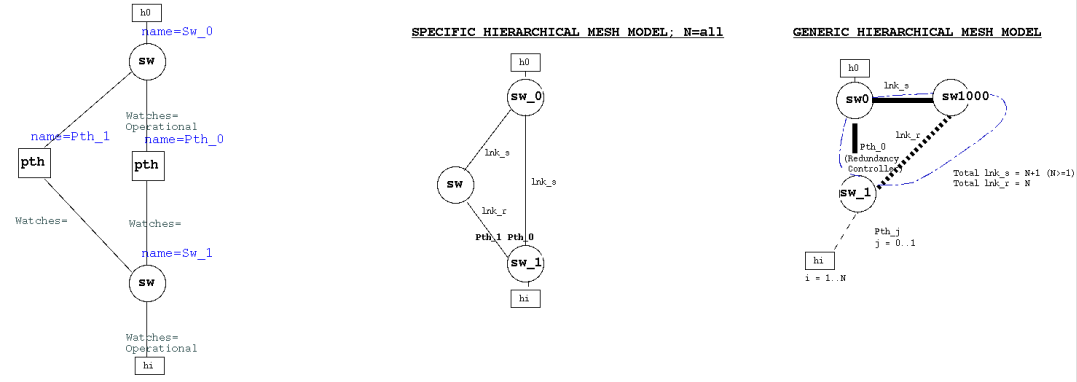

Figure 5.21. Model H: Hierarchical mesh network model

The underlying Markov model is generated from the above model specification by running the Mathematical Module. The Markov model as described by the associated state-space matrix is indicated in Figure 5.6.

```
States

S1 (1,0,.,1,1,0,1,.)
S2 (1,0,.,0,0,0,0,.)
S3 (1,0,.,0,1,0,0,.)
S4 (0,1,.,1,1,1,1,.)
S5 (0,0,.,0,0,0,0,.)
S6 (0,0,.,0,1,0,0,.)
S7 (0,0,.,1,1,0,0,.)
S8 (1,0,.,1,1,0,0,.)

State Transition Matrix
   S1->2 2.2800000000E-05
   S1->3 9.1300000000E-05
   S1->4 9.1300000000E-05
   S2->1 1.0000000000E+00
   S2->5 9.1300000000E-05
   S3->2 2.2800000000E-05
   S3->1 1.1000000000E+00
   S3->6 9.1300000000E-05
   S4->7 9.1300000000E-05 (λ sw = 2.2800000000E-05
   S4->6 9.1300000000E-05 (λ lnk = 9.1300000000E-05)
   S4->5 2.2800000000E-05 (μ com = 1.0000000000E+00)
   S4->1 1.0000000000E+00
   S5->4 1.0000000000E+00
   S5->2 1.0000000000E+00
   S6->5 2.2800000000E-05
   S6->4 1.0000000000E+00
   S6->3 1.0000000000E+00
   S7->5 2.2800000000E-05
   S7->6 9.1300000000E-05
   S7->4 1.0000000000E+00
   S7->8 1.0000000000E+00
   S8->2 2.2800000000E-05
   S8->3 9.1300000000E-05
   S8->1 1.0000000000E+00
   S8->7 9.1300000000E-05
```

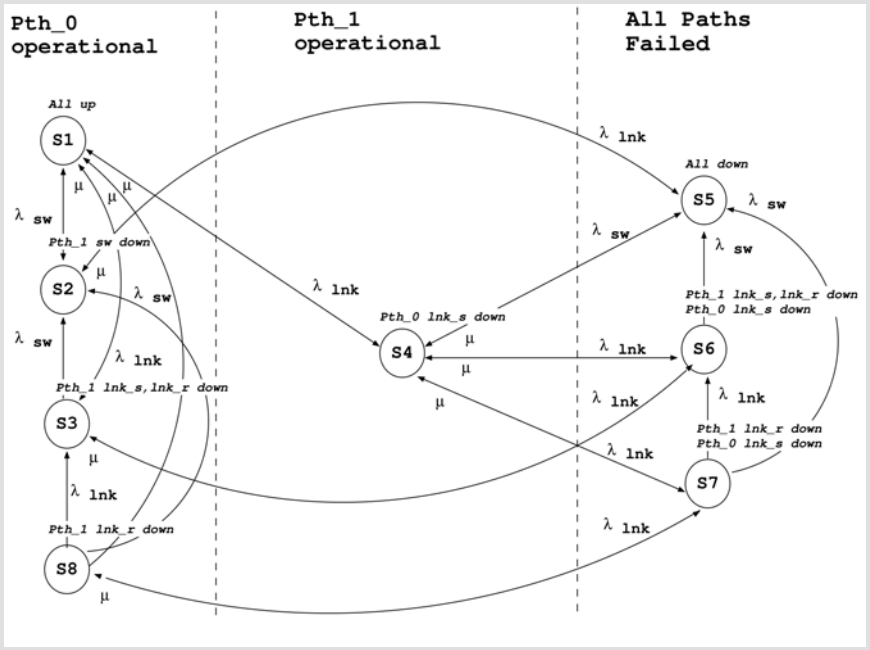

Figure 5.22. Markov Model H: Hierarchical mesh network model

The state transition matrix consisting of 8 states and 25 state transitions is used to construct a directed graph to help depict the various system failure states - states that correspond to an actual network failure. In this model network failures correspond to the set of states that correspond to "All Paths Failed" i.e. both *Pth_0* and *Pth_1* are down.

5.6.2. Model solution

The transient solution for reliability *R(t)* as a function of 8hr time units for the above *N=All* hierarchical mesh topology network model are shown in Appendix A, Figure A.34. The transient solution for expected lifetime *L(t)* is shown in Figure A.35.

The *MTTF* is derived from the steady-state value of *L(t)* as *MTTF = 34294071* or 31298 years. (5.9)

Using the calculated *MTTF* with *MTTR = 1*, the steady-state availability is calculated to be *A = 0.99999997* and the performability metric *M* is calculated from the state-space matrix as *M = 1.000*. (5.10)

5.7. Chapter closure

A network simulation environment will be used in the following chapter to construct and simulate real-world networks in order to validate the analytical network models developed based on Markov state-space modelling techniques and the solutions derived from them using the Tangram Analytical solution Module.

Chapter 6. Model Validation Tests

6.1. Introduction

In this chapter the test methodology for validating the reliability and performability models developed in Chapter 5 is presented. The validation testing environment is discussed in Section 6.2, testing constraints are discussed in Section 6.3.1, data evaluation software developed and experimental data obtained is discussed in the subsequent sections.

The approach is to test and validate the Markov reliability models against the actual behaviour of networks that are built and simulated in the OPNET network simulation environment [113], [73]. The data gathered from these simulations are analysed using two different methods as discussed in Section 6.3.4. Validation test data is presented for the different network topology models in Section 6.5, Section 6.6 and Section 6.8.

6.2. Network testing and simulation environment

Both the OMNet++ and the OPNET network simulation environments discussed in Section 2.9 were evaluated for use as a network simulation environment. The OPNET network simulation environment was selected to construct the different network topologies because the spanning tree modules were found to be more mature and complete in the implementation of the IEEE spanning tree protocol behaviour that was modelled as the redundancy controller. One of the built-in OPNET network equipment models for commercial brand switches (3 Com is used in this investigation) is used as a node unit. The node units are then interconnected using 10 Mbps Ethernet links. A failure-recovery module is then used to induce exponentially distributed failure and recovery events to the switches (nodes) and interconnecting links. For this purpose an existing failure-recovery model was modified to generate random exponentially distributed failures instead of preprogrammed deterministic failures and recoveries. The C++ source code for this modified failure-recovery model is listed in Appendix C.

Figure 6.1 depicts an example of a simple network simulation based on the host-to-host model introduced in Section 4.3. This network is then used to verify the correctness of the failure-recovery model.

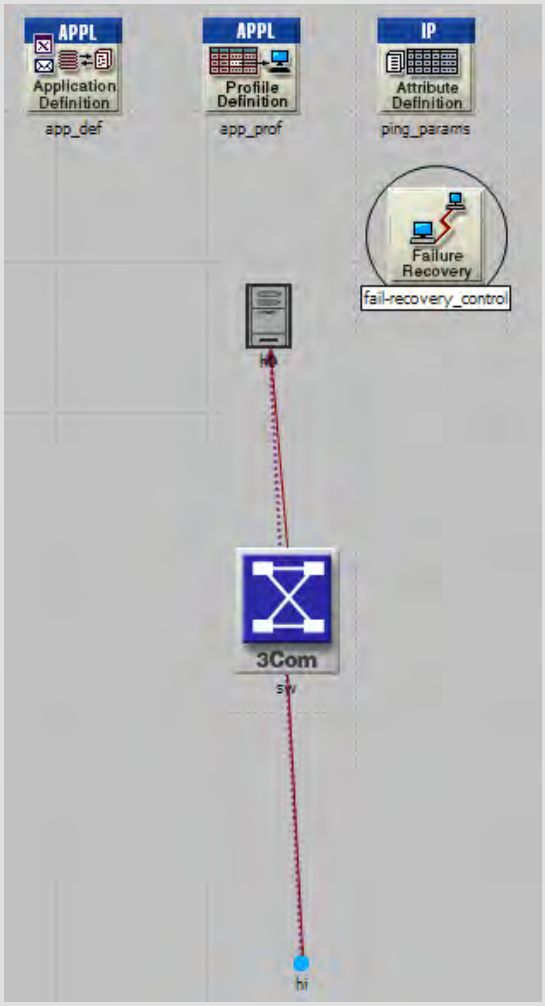

Figure 6.1. OPNET host-to-host simulation - Model B

Two end-node hosts are positioned as indicated in the figure above:

- *hi* representing a client host;
- *h0* representing a server host.

These two nodes are used as reference nodes for conducting all validation tests in accordance with the model reference node placement in Figure 5.1. The IP connection between *hi* and *h0* consists of a ping traffic object that is configured to send ping packets at a configurable rate and with configurable distribution (constant distribution) from the host *hi* to *h0* IP address in Figure 6.2.

Figure 6.2. OPNET ping object: configuration example

The Discrete Event Simulation (DES) module is used to configure all the parameters for the simulation after the network has been constructed and all the modules have been configured. From Figure 6.3 the main configuration parameters consist of:

- Simulation duration in units of either seconds, minutes, hours, days or weeks.
- Seed values to be used for initialising the random function generator.
- Values per statistic that determines the sampling rate as well as the amount of data to be collected.

Since the failure-recovery model uses the random function to generate exponentially distributed failures (see Appendix C) it is important that different seed values are specified for each sample run. See the discussion in Section 6.3.1 on the selection of random seed values.

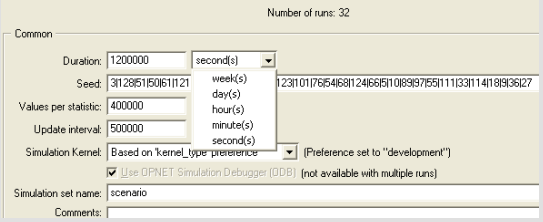

Figure 6.3. Discrete Event Simulation object: configuration example

The ping traffic sent and received statistics generated during a simulation run can be viewed, analysed and exported in CSV format as indicated in the statistics view page indicated in Figure 6.4. The following types of simulation data is gathered for further analyses using appropriate techniques:

- ping replies received from host *h0*;
- link utilisation for identified links on switch *sw_0* attached to end-node *h0*.

The ping protocol is used to generate the network trace or IP data source and the ping replies statistics are used to evaluate network availability (A) and the link utilisation statistics are used to evaluate performability metric M (Section 6.3.4).

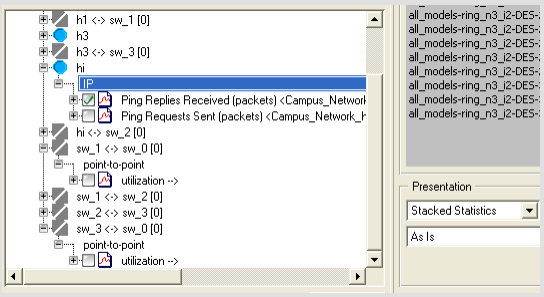

Figure 6.4. OPNET simulation output statistics

The failure-recovery module user interface depicted in Figure 6.5 has been adapted in order to make it possible to specify *MTTF* and *MTTR* values that determines the time parameter used in the exponential distribution function that is used to generate exponentially distributed failure and repair events for each node and link in the network individually. Node failures can be configured to fail links attached to the node as well. The availability calculated from actual packet flows or ping replies received is an indication of the actual availability of the network based on the *MTTF* and *MTTR* duty cycle as determined by Equation 2.10.

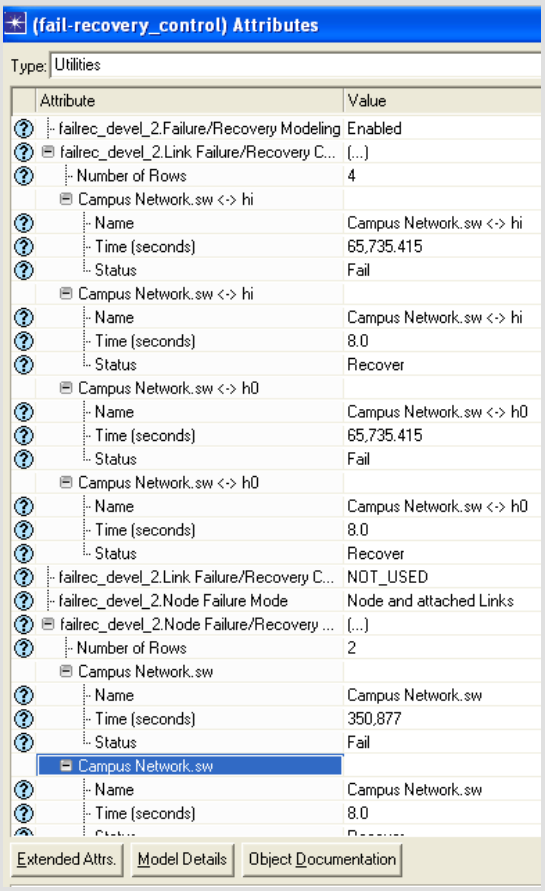

Figure 6.5. OPNET Failure-Recovery object: configuration example

6.3. Verification of simulation software and simulation constraints

Several problem areas in using the network simulation approach have been identified during the initial testing of this methodology. The subsequent sections discusses these problem areas and possible solutions to the challenges they pose.

6.3.1. Simulation run time and sampling constraints

It is important to appreciate the difficulty in simulating network failures for topologies that are estimated to have *MTTFs* in the order of millions, if not tens to hundreds of millions of time units. As can be seen from model prediction results in Chapter 4 and Chapter 5 the *MTTF* for a redundant link is in the order of 120 million units and for an *N=2* mesh/ring/hierarchical mesh networks in the order of 34 million time units. Even when the time unit chosen are seconds the time required to conduct a single network sample run for an *N=2* network with *MTTF = 34 million seconds* is estimated by the OPNET DES module in the order of 7 hours given that the *MTTR = 1 second*. *MTTR* can however not be chosen as 1 second (Section 2.2.1.4, Section 5.1) since spanning tree converges in ideal configuration between 0 - 3 seconds [32]. For small diameter networks (*N=2*) it is possible but not guaranteed that spanning tree will converge within 1 second. The only solution to prevent overlap between spanning tree convergence time and choosing seconds for a time base is to increase the *MTTR* so that it is sufficiently longer than the actual spanning tree convergence time. *MTTR = 8 seconds* was therefore chosen for network simulations (analogous to the eight hours assumed as *MTTR* - Section 3.3). However when *MTTR = 8* then the *MTTF* for the *N=2* networks increase to 8 x 34 million seconds and the simulation execution time for generating a failure in a single simulation run increases to 56 hours. In order to improve the accuracy of simulation results it is also necessary to conduct several simulations using different seed values for each run in order to make use of the law of large numbers [158], [159] that states that the larger the amount of samples the closer the sample mean would approximate the population mean. Assuming ten sample runs are taken the validation of the *N=2* networks can then take up to ten weeks.

Given the issues raised above and the time and resources available to conduct the investigation such long simulation times are not achievable. It is therefore not possible to conduct conclusive validations for network topologies with very large *MTTFs*, instead the approach is to confirm that a given network with a predicted high *MTTF* exceeds a certain value based on a viable duration time used in the simulation test run. For topologies with lower predicted *MTTF*, star networks and ring networks with *N>9*, more conclusive validation results can be obtained.

It is also necessary to obtain a sufficient amount of samples or simulation test runs for every network topology. The default value selected to be used (where practical) is 32 runs, selected to be larger than a "rule-of-thumb" 30 samples where the distribution of sample means would approach a normal distribution [160]. A true random number generator service [161] is used to generate a sequence of 32 random seeds that are used to seed the random function for every individual test run.

In order to reconstruct the point availability *A(t)* function from packet counts where the amount of received packets represents uptime (or *MTTF*) and the amount of lost packets represents downtime (or *MTTR*) (see Equation 2.11) a constant ping packet source or trace is used as discussed in the previous section. Further, to correctly reconstruct the resulting point availability function depicted in Figure 6.6 it is important to sample at a high enough frequency to capture the waveform as required by Nyquist's sampling theorem [162]. Assuming a ping period of one second and an

MTTR of eight seconds implies that the mean value of the downtime is eight seconds the minimum sampling interval should then be four seconds. There is of course a problem with this approach since *MTTR* is an exponential distribution and it is expected to have *MTTRs* that are much smaller than eight seconds thus violating the Nyquist sampling criteria. Alternatively higher sampling data rates can be used but this introduces a new problem since a very large amount of data must be gathered for every sample run. For example - in order to sample at one second interval a simulation test of over 32 runs at 2.4 million seconds for each run implies over 76 million 32 bit float numbers must be collected and stored. This caused memory usage instabilities in the statistics gathering modules and the software to crash. Through trail-and-error it was found that the maximum amount of data points that could be collected per run should not exceed 400000 points per sample run in order for the statistic module to remain stable.

From [78] it is assumed that the point availability *A(t)* converges to steady state availability *A* (Equation 2.9) at approximately *t = 4 x MTTF* i.e. ideally a sample run must be longer than 4 x *MTTF* and using average received packet counts over the operation cycle is therefore not influenced by the sampling interval. The reconstruction of the point availability function *A(t)* is however important to identify non-ideal behaviour of the spanning tree convergence algorithm as will be discussed in Section 6.3.3.

Given the time and resource constraints the following default DES simulation parameters was therefore chosen where feasible for conducting simulations:

- a sample run duration is either 4 x predicted *MTTF* or alternatively 2.4 million seconds and a second is used as the basic simulation time unit.
- *MTTR = eight seconds* and all *MTTFs* obtained in Chapter 4 and Chapter 5 therefore scaled by a factor of eight seconds;
- 32 independently seeded sample runs are used for every validation test;
- a ping object with a constant distribution and one second ping period is used as the sampling probe;
- amount of sample data points to be 1/3 of the maximum sample run time, implying a sample is taken every three seconds with the maximum amount of data points per run not to exceed 400000 points. This implies that when the 2.4 million second simulation run time is used sampling is limited to an interval of six seconds.

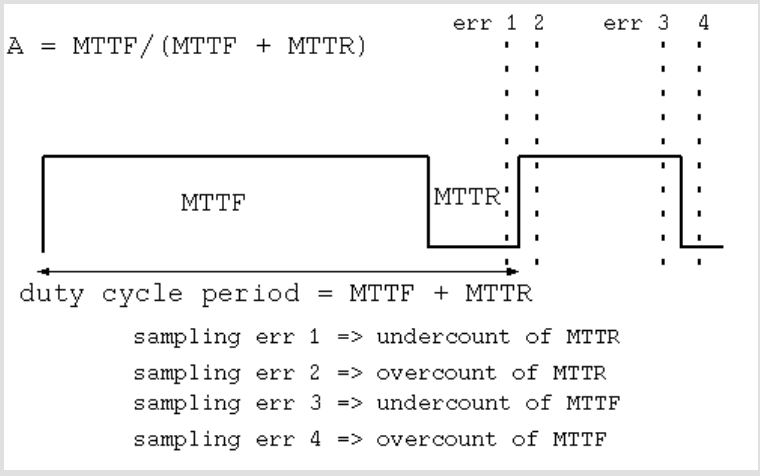

Figure 6.6. Availability duty cycle sampling error

Figure 6.7 depicts the DES simulation progress output. In this example a simulation time of 1.2 million seconds was selected which resulted in a real simulation time of approximately 15 minutes per sample run. The progress information for the first five sample runs is shown. During initial testing it was also found that it is not only the simulation time that determines the real simulation duration but that the amount of network activity is a major contributor to the actual duration of the simulation. It was found for example that increasing the amount of end-node hosts that are configured to generate network traffic also dramatically increases the actual simulation time in the bigger networks ($N>5$) with a factor of 10. This is to be expected since the actual simulation time is primarily determined by the amount of network events that must be simulated. This would not influence network reliability validation tests since only a single host (*hi*) can be configured to generate ping traffic but it did increase simulation time for determining the performability metric M which is a measure of the amount of traffic generated in a network topology through a shared path. The method chosen to determine M is discussed in more detail in Section 6.3.4.

```
Run Dur. Time_Elap. Num_Events Total_Mem. Avg_Ev/s Suffix  Seed
#1 333h 20m 00s 15m 53s  355,112,967 24,615  372,570  DES-1  3
#2 333h 20m 00s 16m 37s  355,085,361 24,541  356,084  DES-2  128
#3 333h 20m 00s 16m 40s  355,253,303 24,541  355,115  DES-3  51
#4 333h 20m 00s 16m 41s  355,193,615 24,671  354,824  DES-4  50
#5 333h 20m 00s 16m 40s  355,133,765 24,542  355,059  DES-5  61
```

Figure 6.7. Discrete Event Simulation - progress status

Debug information logs as depicted in Figure 6.8 are generated for every sample run. The information logged to the debug file includes progress and status information that is useful to analyse and debug model development. The failure-recovery module modified for this project to generate failure and repair events has been programmed to generate and log failure and recovery events together with an accurate time stamp as they occur during a simulation: for example, from the output below it can be seen that the link between *sw_7* and *sw_8* failed at *t = 7185134.1525 seconds* and recovered at *t = 7185139.0556 seconds*. The utilisation of this debug information to validate the correctness of the reliability models is discussed further in Section 6.3.4.

```
------------------------------------------------------------------
Progress:Time (1995 hr. 37 min. 44 sec.); Events (535,002,155)
Speed:Average (414,699 events/sec.); Current (822,370 events/sec.
Time :Elapsed (21 min.); Remaining (3 hr. 45 min.)
DES Log:8 entries
------------------------------------------------------------------

 >>>> Campus Network.sw_7 <-> sw_8 FAILED at time = 7185134.1525

 << Campus Network.sw_7 <-> sw_8 RECOVERED at time = 7185139.0556
------------------------------------------------------------------
Progress:Time (1997 hr. 50 min. 29 sec.); Events (535,502,158)
Speed:Average (414,889 events/sec.); Current (813,013 events/sec.
Time :Elapsed (21 min.); Remaining (3 hr. 48 min.)
DES Log:8 entries
------------------------------------------------------------------

 >>>> Campus Network.sw_3 <-> sw_4 FAILED at time = 7196671.4252

 << Campus Network.sw_3 <-> sw_4 RECOVERED at time = 7196674.3591
------------------------------------------------------------------
Progress:Time (2000 hr. 3 min. 0 sec.); Events (536,002,159)
Speed:Average (415,083 events/sec.); Current (830,567 events/sec.
Time :Elapsed (21 min.); Remaining (3 hr. 44 min.)
DES Log:8 entries
------------------------------------------------------------------
------------------------------------------------------------------
Progress:Time (2002 hr. 17 min. 44 sec.); Events (536,502,160)
Speed:Average (415,276 events/sec.); Current (830,564 events/sec.
Time :Elapsed (21 min.); Remaining (3 hr. 41 min.)
DES Log:8 entries
------------------------------------------------------------------

>>>> Campus Network.sw_5 <-> sw_6 FAILED at time = 7210689.6565

 >>>> Campus Network.sw_9 <-> sw_0 FAILED at time = 7210696.0600

 << Campus Network.sw_9 <-> sw_0 RECOVERED at time = 7210726.9056

 << Campus Network.sw_5 <-> sw_6 RECOVERED at time = 7210758.9244
```

Figure 6.8. Discrete Event Simulation - debug trace

6.3.2. Verification of failure-recovery module

The correct operation of the failure-recovery module (source code included in Appendix C) as modified for this project was verified and confirmed by testing a simple link with reference model A network depicted in Figure 4.1 using simulation times $t = 1500000\ s$, sampling interval 3s and $t = 3000000\ s$, sampling interval 6s respectively with the results listed in Figure 6.9 and Figure 6.10.

The results were analysed using the PCM method described in Section 6.3.4. As expected the difference between the model predicted availability (*A = 0.9999079*) and the calculated availability decreased as the simulation time was increased; with a 0.00009% difference between model predicted and simulation result.

```
*****
SAMPLE SIMULATION NAME = link, Model A, 1.5 mil

TOTAL SIMULATION TIME = 1500000
SAMPLING UNITS = seconds
SAMPLING PERIOD = 3
TOTAL VALUES PER RUN = 500000


RUNS CALCULATED MEAN AVAILABILITY = 0.9999014

MODEL PREDICTED MTTF = 86849.8
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.9999079
ERROR SIMULATION MEAN AVAILABILITY = -0.00065%

*****
```

Figure 6.9. Link topology simulation results - Model A, 1.5 million time units

```
*****
SAMPLE SIMULATION NAME = link, Model A, 3 mil

TOTAL SIMULATION TIME = 3000000
SAMPLING UNITS = seconds
SAMPLING PERIOD = 6
TOTAL VALUES PER RUN = 500000


RUNS CALCULATED MEAN AVAILABILITY = 0.9999070

MODEL PREDICTED MTTF = 86849.8
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.9999079
ERROR SIMULATION MEAN AVAILABILITY = -0.00009%

*****
```

Figure 6.10. Link topology simulation results - Model A, 3.0 million time units

6.3.3. Verification of spanning tree convergence

As discussed in Section 5.1 and Section 6.3.1 the correct operation of the spanning tree mechanism that serves as the network redundancy controller must be verified in order for the network

reliability models that rely on redundant configurations to be validated. The OPNET simulation environment offers three different IEEE802.1 [39] based spanning tree protocols that can be configured for the switch models - 802.1D, 802.1w and 802.1s. These spanning tree protocols and the role of the "bridge priority settings" configuration parameters listed below are discussed in Section 2.2.1.4. After extensive initial testing using the relatively minimal networks in Figure 6.13 with *N=2* switches and Figure 6.11 with *N=3* switches the IEEE 802.1s spanning tree protocol was selected as it proved to be the most predictable and stable. Various anomalies with regards to the correct convergence behaviour using the different versions of the spanning tree protocols were identified by reconstructing the point availability function *A(t)* from received packet counts and the duty cycle depicted in Figure 6.6 - however the discussion of this falls outside of the scope of the investigation. It is reasonable to assume from [45] that for small diameter networks ($N \leq 8$) the spanning tree convergence time should be between one to three seconds. The main problem identified is that when failures were introduced in some of the nodes, spanning tree convergence exceeded documented convergence times for networks even with small diameters and therefore the excessive spanning tree convergence time played a distorting role in analysing the packet loss results due to network configuration changes. The implication of this problem has to be understood when analysing packet losses due to failures when attempting to calculate network availability and is discussed further in Section 6.3.4.

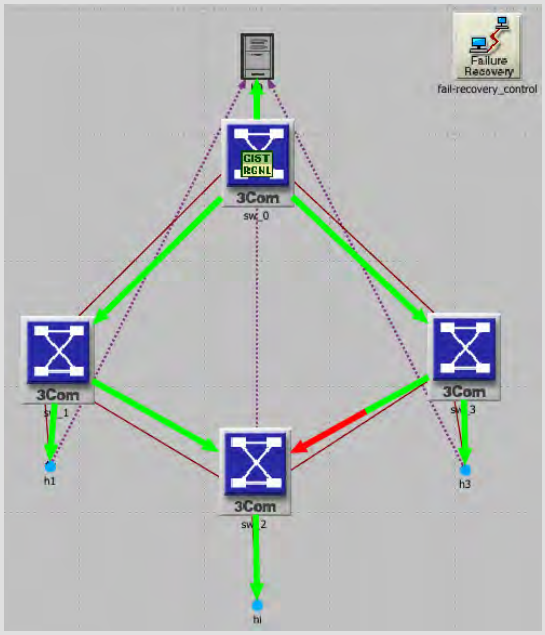

Figure 6.11. OPNET spanning tree visualisation feature

Switch *sw_0* connected to host *h0* is configured as the root bridge and the back-up root bridge is configured to be the switch connected to host *hi*. Figure 6.12 shows the spanning tree configuration options. IEEE802.1s protocol with the following settings used:

- Root bridge *sw_0*: *bridge priority setting = 8192*
- Backup-root bridge *sw_i*: *bridge priority setting = 16384*
- All other switches: *bridge priority setting = 32768*

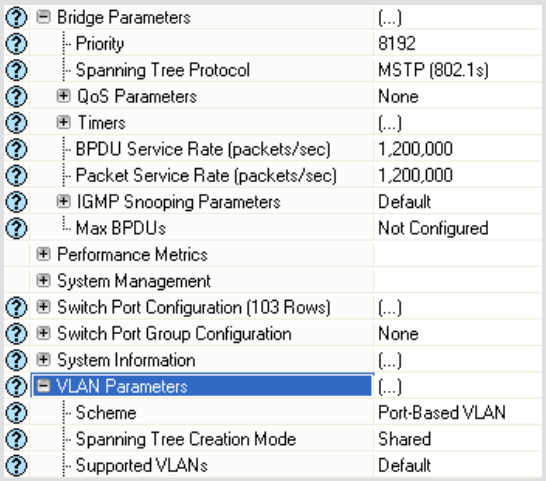

Figure 6.12. OPNET switch object: configuration example

6.3.4. Validation methods

In the previous sections a couple of problematic areas in using network simulation packet loss results to validate the reliability models have been identified and discussed. In this section methods are introduced that will be used to best address the challenges posed and to establish a consistent approach to the validation testing of the different topologies.

The first simulation tool that is useful for general model validation is the spanning tree visualisation feature that helps to validate the basic assumptions behind developing the network models developed in Chapter 5. The network models are based on an interpretation of how spanning tree protocols create redundant paths in the various network topologies: see in particular the modelling assumptions as illustrated in Figure 5.2, Figure 5.3, Figure 5.4 and Figure 5.10. All the models that rely on the spanning tree redundancy mechanism could be evaluated using the OPNET spanning tree visualisation feature and the modelling assumptions were found to be correct, for example, the spanning tree visualisation in Figure 6.14 validates the assumed position of the blocking link (indicated by the red arrow) at $i = N/2$ as assumed in the general ring model presented in Section 5.4.1.4.

Two basic validation testing methods are proposed:

- a direct method using actual measured data flows i.e. dropped data packets due to network downtime that will be referred to as the Packet Count Method (PCM);
- an indirect method detecting equipment failure and repair events leading to path failures and inferring the expected data flow that will be referred to as the Path Failure Method (PFM).

Since very large data sets consisting of up to 12.8 million data points per test have to be analysed a couple of custom data processing scripts were developed in the Python scripting language with the source code for every script listed in Appendix D.

The Packet Count Method (or direct method) takes as input the "pings received" packet count data file as exported from the statistic output module in Figure 6.4 and calculates the mean availability (A) by measuring the ratio between ping packets sent from host *hi* to host *h0* (a known quantity) versus ping packets received by *hi* from *h0*. An example of the output results from this script (listed as script "data_crunch_av.py" in Appendix D) is given in Figure 6.9. This script analyses

the ping data received and then detects the point availability *A(t)* duty cycle period as indicated in Figure 6.6 by searching for the last detectable downtime or *MTTR* and detecting a packet count transition from below the expected ping received count threshold to the expected ping received count when the path between *hi* and *h0* is fully connected. In order for this technique to work the sampling rate must be high enough to detect the availability duty cycle waveform accurately (Section 6.3.1) - auxiliary flags *truncate_on_non_zero* and *truncate_on_last_mttr* are used for availability waveform analyses. Another factor that can distort the waveform detected is spanning tree conversions that result in ping packet losses larger than 1 second which is the ping probe time. In order to compensate for this a *stp_noise_level* flag is included that can be set to ignore perceived downtime due to slow spanning tree convergence. The script is mainly used to calculate average amounts of packets sent and received over the entire operational cycle in order to determine the steady-state availability *A* as discussed in Section 6.3.1.

The Path Failure Method (or indirect method) takes as input the "DES debug" trace files as exported from the DES module in Figure 6.7 and calculates the mean availability (*A*) by detecting failure events that lead to a loss of connectivity or total path failure between node *hi* and *h0*. An example of the output results from this script (listed as script "DES_crunch.py" in Appendix D) is given in Section E.2 under heading "PFM calculation of *A* output with sim. time = 2.4 million seconds". This script analyses the failure and repair events as recorded in Figure 6.8 by searching for marker string identifiers in the debug text files generated during the simulation sequence. The relevant marker strings are highlighted in bold, in this example the last couple of entries in the log indicate simultaneous failures of *sw_5* and *sw_9*. Since this debug output was generated for a ring network and *hi* was attached to a node somewhere in between *sw_5* and *sw_9* this constitutes a total path failure to *h0* and would result in down time until *sw_5* or *sw_9* is repaired. The time stamps associated with the simultaneous failures and subsequent repair events are then used as input to a second processing script (listed as script "calc_from_failures_av.py" Appendix D) that reconstitutes an idealised packet flow only interrupted by total path failure events and then calculates the simulation availability from such the reconstructed packet counts. This technique requires the investigator to interpret the simulated failure and repair events and to select only those that would lead to path failure or network downtime. Since the packet flows are reconstructed the availability results obtained are accurate but are nevertheless idealised since actual network dynamics that may deviate from the anticipated are suppressed.

Many trial simulation runs were conducted with different combinations of simulation run times, ping periods, sampling intervals and scaled *MTTRs*. By analysing the point availability function *A(t)* reconstructed by PCM method and comparing the results with the PFM it was found that due to the various factors and constraints that influenced the behaviour of the simulation, the PCM could only be used to validate reliability of simple networks that are not dependent on or sensitive to idealised spanning tree convergence behaviour: star networks, mesh networks, hierarchical mesh networks and ring networks with diameters $N \leq 3$. The applicable testing methods, simulation results obtained and explanatory comments are given in Section 6.4, Section 6.5, Section 6.6, Section 6.7 and Section 6.8.

The merit figure (*M*) as a measure of the network performability as introduced in Section 2.5 and modelled in Figure 5.4 for mesh networks and Figure 5.10 for ring networks is validated using the Packet Count Method (or direct method) and takes as input the "point-to-point utilisation" data file as exported from the statistic output module in Figure 6.4 and then calculates the mean merit (*M*) by averaging the link utilisation (relative to host *hi*) of the upstream link connected to *sw_0* with data traffic destined to host *h0*, or in other words - the bandwidth usage or congestion in the connected path carrying traffic from *hi*. An example of the output results from this script (listed as script "data_crunch_m.py" in Appendix D) is given in Section E.2 under heading "PCM calculation of *M* output with sim. time = 2.4 million seconds". Merit is thus validated by

examining the average utilisation of the least-cost connection path from host *hi* to *h0*. Since all the factors and constraints mentioned above for availability calculation are also applicable to using the PCM to determine utilisation an approximation of the utilisation before failures occur is used in the relevant affected networks (rings with diameter *N>3*) by limiting the calculation to the time period until the first packet losses are detected due to either actual path failures or to spanning tree convergence time.

It is noted that merit metric *M* defined in Section 5.1 as *M = B/T* is in fact the inverse of link utilisation; where utilisation is defined as the traffic running through a link divided by the link bandwidth. *M* is calculated (Section 5.2) by assuming constants for bandwidth *B = 1000 units* and *T = 1000 units* where *B* is the bandwidth of a link and *T* is the assumed average data traffic carried through a switch. The equivalent constants used in the OPNET simulations is determined from reference model B in Figure 6.1 and validated in Section 6.4 by examining typical data flow generated by 1s ping trace through a single switch into a 10 unit (Mbps) link and an applicable *M* scaling ratio *Km* can be calculated where:

$$\mathrm{Km} = \frac{\mathrm{M_{simulation}}}{\mathrm{M_{model}}} = \frac{10/0.105}{1000/1000} \approx 94.7 \qquad (6.1)$$

The scaling factor *Km* is used to compare model results with simulation results included in the script "data_crunch_m.py" as the variable *M_SCALING_FACTOR*. The appropriateness of the calculation of *Km* is discussed in the context of the validation testing results in Section 7.2.

6.3.5. Selection of models to be validated

Given the amount of time required to test a single model, a subset of models considered to be representative of all the models constructed in Chapter 4 and Chapter 5 have been selected for testing. Table 6.1 lists the models selected along with the testing objective to be realised in validation testing:

Table 6.1. Model validation testing objectives

Mod	N	i	Availability (*A*)	Performability (*M*)
A	-	-	Verify correctness of failure-recovery module and data_crunch_av.py script.	-
B	-	-	Verify correctness of failure-recovery module and data_crunch_av.py script.	Validation results is used to determine *Km*.
F8	9	1	Very large *MTTF* - can only confirm *MTTF* to be higher than test period.	Validate *M*.
F9	9	5	*N* selected to be same as G/H for network size comparison. Validate *A*.	Validate *M*.
F10	17	1	Validate *A*.	Validate *M*.
F11	17	9	Validate *A*.	Validate *M*.
G	9	all	*N* selected to be same as F8/F9 for network size comparison. Validate *A*.	Validate *M*.
H	9	all	*N* selected to be same as F8/F9 for network size comparison. Validate *A*.	Validate *M*.

The subsequent sections in this chapter summarises validation results for all the models tested. The spanning tree visualisation tool discussed in Section 6.3.3 is used to validate the rationale behind the construction of the reliability models as illustrated in Chapter 5. Actual validation results are tabled together with explanatory notes on the testing methods used.

6.4. Link topology simulation

6.4.1. Introduction

The host-to-host link network as represented by model B is depicted in Figure 6.1. The trunk link model in Section 4.4 was validated against Equation 2.19 and the redundant link model was validated against Equation 2.20 through manual calculation and both models were found to be accurate.

6.4.2. Host-to-host link simulation results

Results are summarised in table below with test data outputs in Figure 6.9:

Table 6.2. Host-to-host link topology simulation: Model B

Sim. time	**Sim. units**	**Sample period**	**Model MTTF (hr)**	**Model *A***	**Sim. *A***	**Model *M***	**Sim. *M***
150 thou	sec	3	30053	0.9997	0.9997 (1)	1.000	94.7 (2,3)

Notes:

1. PCM (STP noise level = 0) used.

2. Mean value of *sw* to *h0* link utilisation calculated with OPNET statistics module with single test run and *sw* configured **not to fail**.

3. Simulation *M* is used to determine *Km* as per Equation 6.1.

6.5. Mesh topology simulations

6.5.1. General

The mesh (*N=2*) network as represented by model E1 is depicted in Figure 6.13. The spanning tree visualisation tool indicates the location of the blocked link (in red) at simulation time *t = 2 s*.

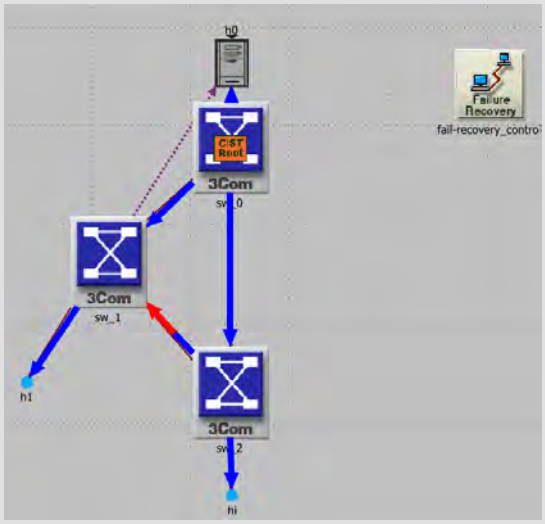

Figure 6.13. OPNET simulation of mesh topology, N=2

Since the mesh with *N=2* network has very large predicted *MTTF* and because of the similarity to the hierarchical mesh network the model is validated in Section 6.8.

6.6. Ring topology simulations

6.6.1. Introduction

The ring (*N=9, i=5*) network as represented by model F1 is depicted in Figure 6.14. The spanning tree visualisation tool indicates the location of the blocked link (in red) at simulation time *t = 2 s*.

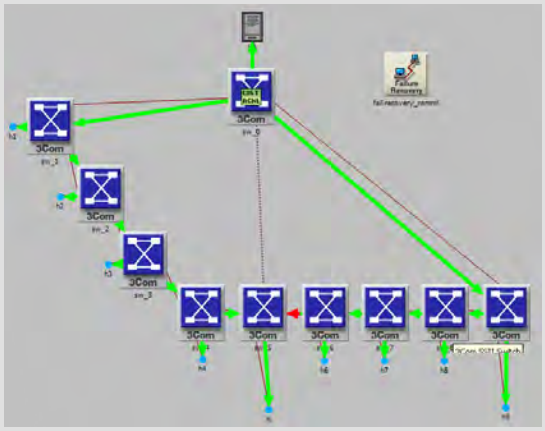

Figure 6.14. OPNET simulation of ring topology

6.6.2. Ring with *N=9, i=5* simulation results

Results are summarised in table below with test data outputs in Section E.3:

Table 6.3. Ring topology with N=9, i=5 simulation: Model F9

Sim. time	Sim. units	Sample period	Model MTTF (hr)	Model *A*	Sim. *A*	Model *M*	Sim. *M*
2.4 mil	sec	6	-	-	-	0.200	0.200 (2)
60 mil	sec	-	14569468	0.99999945	0.99999971 (1)	-	-

Notes:

1. PFM used calculated over 16 test runs. PCM gives incorrect results.

2. PCM used for *M* calculated over *t = 0 to 6240 time units*.

6.6.3. Ring with *N=9, i=1* simulation results

Results are summarised in table below with test data outputs in Section E.2:

Table 6.4. Ring topology with N=9, i=1 simulation: Model F8

Sim. time	Sim. units	Sample period	Model MTTF (hr)	Model *A*	Sim. *A*	Model *M*	Sim. *M*
2.4 mil	sec	6	-	-	-	0.200	0.200 (1)

Notes:

1. PCM used for *M* calculated over *t = 0 to 10308 time units*.

2. No path failures occurred during 2.4 million time units. *MTTF>2400000* and *A>0.999966668*.

6.6.4. Ring with *N=17, i=9* simulation results

Results are summarised in table below with test data outputs in Section E.5:

Table 6.5. Ring topology with N=17, i=9 simulation: Model F11

Sim. time	Sim. units	Sample period	Model MTTF (hr)	Model *A*	Sim. *A*	Model *M*	Sim. *M*
17 mil	sec	-	4168505	0.9999981	0.9999990 (1)	0.111	0.075

Notes:

1. PFM used calculated over 16 test runs. PCM gives incorrect results.

2. PCM used for *M* calculated over *t = 0 to 6240 time units*.

6.6.5. Ring with *N=17, i=1* simulation results

Results are summarised in table below with test data outputs in Section E.4:

Table 6.6. Ring topology with N=17, i=1 simulation: Model F10

Sim. time	Sim. units	Sample period	Model MTTF (hr)	Model *A*	Sim. *A*	Model *M*	Sim. *M*
94 mil	sec	-	23481947	0.99999966	0.99999981 (1)	0.111	0.075

Notes:

1. PFM used calculated over 11 test runs. PCM gives incorrect results.

2. PCM used for *M* calculated over *t = 0 to 6240 time units*.

6.7. Star topology simulations

6.7.1. Introduction

The star (*N=9*) network as represented by model G is depicted in Figure 6.15. The spanning tree visualisation tool indicates the location of the blocked link (in red) at simulation time *t = 2 s*.

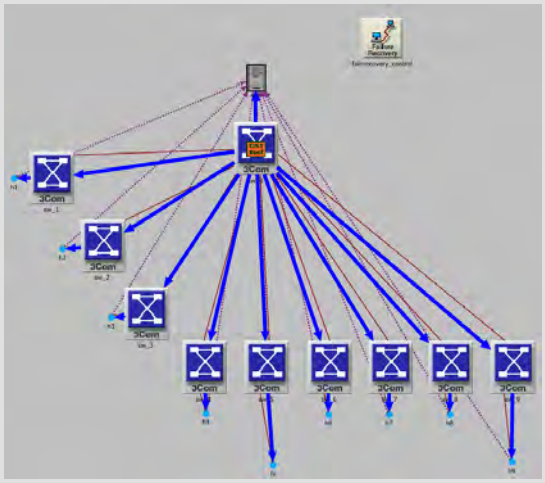

Figure 6.15. OPNET simulation of star topology, N=9

6.7.2. Star with *N=9* simulation results

Results are summarised in table below with test data outputs in Section E.6:

Table 6.7. Star topology with N=9 simulation: Model G

Sim. time	Sim. units	Sample period	Model MTTF (hr)	Model *A*	Sim. *A*	Model *M*	Sim. *M*
360 thou	sec	3	87623	0.999909	0.999907 (1)	-	-
2.4 mil	sec	6	87623	-	-	1.000	1.001 (1)

Notes:

1. PCM (STP noise level = 0) used.

2. PCM used for *M* calculated over *t = 0 to 240000 time units*.

6.8. Hierarchical mesh topology simulations

6.8.1. Introduction

The hierarchical mesh (*N=9*) network as represented by model H is depicted in Figure 6.16. The spanning tree visualisation tool indicates the location of the blocked link (in red) at simulation time *t = 2 s*.

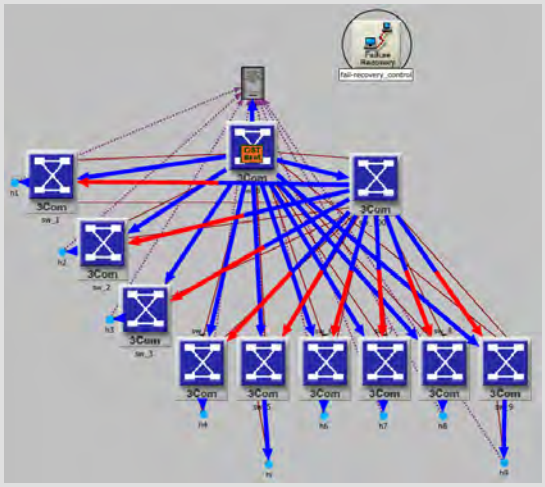

Figure 6.16. OPNET simulation of hierarchical mesh topology, N=9

6.8.2. Hierarchical mesh with *N=9* simulation results

Results are summarised in table below with test data outputs in Section E.7:

Table 6.8. Hierarchical mesh topology with N=9 simulation: Model H

Sim. time	Sim. units	Sample period	Model MTTF (hr)	Model *A*	Sim. *A*	Model *M*	Sim. *M*
120 mil	sec	-	274352565	0.99999997	0.99999996 (1)	-	-
2.4 mil	sec	6	-	-	-	1.000	1.000 (2)

Notes:

1. PFM used calculated over 3 test runs. PCM data capture files too large for simulation environment.

2. PCM used for *M* calculated over *t = 0 to 240000 time units*.

6.9. Chapter closure

This chapter concludes all modelling and validation work required for the successful completion of this investigation as summarised in the scope and objectives (Section 1.4). The following chapter will first analyse and discuss all the data and results obtained so far comparing it to work explored in Chapter 2, which will then lead into the final chapter of conclusions, summary of findings and overall recommendations.

Chapter 7. Results and Discussion

7.1. Introduction

In this chapter results obtained from model solutions and simulations are discussed. The discussion is focussed around the following areas:

- interpretation of the model validation testing data as obtained in Chapter 6;
- comparison of the network topologies under investigation with respect to predications of reliability (*MTTF*), availability (*A*) and performability metric (*M*);
- discussion of the predictions and results when compared with existing literature;
- discussion of the network models and limitations with reference to assumptions made and the modelling simplifications introduced.

7.2. Interpretation of results

A list of factors that influence the interpretation and accuracy of real time network simulation data collected in the validation tests is discussed in Section 6.3.1, Section 6.3.3 and Section 6.3.4. They include the role of limited system resources including simulation time, simulation data gathering sample size and sampling interval constraints. Non-ideal or incorrect behaviour of the spanning tree algorithm has also been discussed. Most of these factors that would influence the quality and accuracy of the validation data were identified during the preliminary validation testing of the network models and methods were designed to analyse the data that would minimise their influence or eliminate them where possible. In general it is noted that network topology models with a very large *MTTF* are much harder to validate accurately using a network simulation approach.

A comparison of reliability model (*MTTF*, *A*) predications versus simulation results are summarised in Table 7.1.

Table 7.1. Model versus simulation results: Availability

Model	**Predicted MTTF (hr)**	**Test method**	**Predicted *A***	**Simulation *A***	**% deviation**
B	30053	PCM	0.9997	0.9997	0.000%
G	87623	PCM	0.99991	0.9991	0.000%
H	274352565	PFM	0.99999997	0.99999996	0.000%
F9	14569468	PFM	0.9999995	0.9999997	0.000%
F10	23481947	PFM	0.9999997	0.9999998	0.000%
F11	4168505	PFM	0.999998	0.9999990	0.000%

It was possible to validate models B and G with relatively small *MTTFs* using PCM based on actual packet losses recorded. A general observation is that given the time and resource constraints and the amount of sample runs required it is not possible to accurately determine the reliability of network models H, E1 and F3 - all models with very large predicted *MTTFs*. The best validation result was obtained for model H which is similar to models E1 and F3, with simulation run time

of 120 million time units which represents approximately 40% of the model predicated *MTTF*. However, for the ring models with a lower *MTTF* increasing the simulation times to 4 times *MTTF* time units and using fewer sample runs to reduce overall simulation time, then using the PFM that is not sensitive to spanning tree noise, it was possible to confirm the availability of models F9, F10 and F11 based on failure-repair event data reconstruction. It can therefore be said that the ring topology models F1 to F19 that can be simplified to a generalised model as defined in Section 5.4.1.4 have all been validated. The only models that can not be accurately validated are models H and E1 constituting the mesh network models with a very high *MTTF = 274352565 hr*.

A comparison of merit figure (*M*) model predications versus simulation results are summarised in Table 7.2.

Table 7.2. Model versus simulation results: Merit

Model	**Predicted *M***	**Simulation *M***	**% deviation**
G	1.000	1.001	0.095%
H	1.000	1.000	-0.027%
F8	0.200	0.200	0.046%
F9	0.200	0.200	0.031%
F10	0.111	0.075	-32.819%
F11	0.111	0.075	-32.819%
Note: PCM is used to calculate all simulation M.			

The merit figure of each model was validated with simulation results and percentage deviation shown above using the PCM. Simulation test results confirmed that the scaling factor *Km* determined in Equation 6.1 and used to compare model results with simulation results in the script "data_crunch_m.py" as the variable *M_SCALING_FACTOR* provided an accurate scaling factor for star, mesh and ring models with $N \leq 9$. However, for $N=17$ the merit figures calculated in simulations are significantly lower (32%) than the merit figures predicated by the models. On closer inspection the reason for this difference became apparent and is presumed to be related to using a 1s ping testing probe to generate network traffic. The link utilisation measured was found to be generally low, in the order of 0.1 of the 10 Mbps links used in the simulations, and generated as result of the combination of ARP (Section 2.2.4.1), ping and spanning tree traffic (Section 2.2.1.4). The ping (or application related traffic) is therefore very low, the spanning tree traffic (proportional to application traffic) increases dramatically in the larger ($N \geq 9$) ring networks. It should then be noted that the model predicted relative *M* figures using a 1s data source could only be validated for ring networks where $N \leq 9$.

The link utilisation was used to calculate *M* from Equation 6.1 where the link utilisation is calculated as an average over the first part of the simulation run until the PCM encounter packet losses attributable to either path failure or spanning tree noise. The merit calculated from the simulation data is therefore only an approximation of how much traffic flows through a shared link but as can be seen from the modelling values there is not a significant difference between model predictions and simulation results with the exception of the larger ring networks where the spanning tree traffic increase significantly. This is because the merit figure (*M*) in all the redundant networks with a large *MTTF* is determined by the amount of data (or nodes) that share a link under normal conditions i.e. when no path failures have occurred, and *MTTR* is very small in comparison with *MTTF*.

7.3. Comparison of link and network topologies

The reliability (*MTTF*, *A*) and performability metrics (*M*) calculated from the link and network topology models under investigation are compared in Table 7.3.

First it can be observed that models E1, F1 and H all yield the same reliability and performability results confirming that although based on different underlying Markov models (Figure 5.5, Figure 5.11, Figure 5.21) consistent results were obtained for the same two node network topology as represented by models with *N=2*.

Model B (Figure 4.3) serves as a baseline model, indicating that for the simplest of network configurations consisting of two hosts connected with two network cards, with connectors and a cable through a single switch the expected host-to-host link unit availability using the component *MTTFs* assumed in Section 3.3 is only three-nines. This simple configuration consists of four connectors, two cables, two network cards and a switch (or single connecting node) all connected in series which explains the relatively low availability since all the failure rates are added together using Equation 2.17.

Comparing link topology model C and D demonstrates the vast improvement in link reliability that is achieved by using redundant network links, improving the single link reliability calculated in model A Figure 4.1 at *MTTF = 86850 hr* to *MTTF = 959903831 hr* (redundant link) or *MTTF = 479995728 hr* (trunk link) which represents an improvement of four orders in magnitude as can be calculated directly from Equation 2.18.

Table 7.3. Comparison of reliability metrics for link and network models

Model	**Description**	***N***	***i***	***MTTF* (hr)**	***A***	***M***
B	host-to-host link	-	-	30053	0.9997	-
C	trunk link	-	-	479995728	0.99999998	-
D	redundant link	-	-	959903831	0.999999992	-
G	star	All	Independent	87623	0.99991	1.000
H	hierarchical mesh	All	Independent	274352565	0.99999997	1.000
E1	mesh	2	Independent	274352565	0.99999997	1.000
E2	mesh	3	Independent	>2000000000	0.999999996	1.000
F1	ring	2	Independent	274352566	0.99999997	0.999
F2	ring	3	1	160070988	0.99999995	0.500
F3	ring	3	2	121968191	0.99999993	0.500
F4	ring	4	1	113011955	0.99999992	0.500
F5	ring	4	2	71162814	0.99999990	0.500
F6	ring	5	1	87342922	0.99999991	0.333
F7	ring	5	3	45756504	0.9999998	0.333
F8	ring	9	1	45782834	0.9999998	0.200
F9	ring	9	5	14569468	0.9999995	0.200
F10	ring	17	1	23481947	0.9999997	0.111
F11	ring	17	9	4168505	0.999998	0.111
F12	ring	33	1	11918441	0.9999993	0.059
F13	ring	33	17	1121449	0.999993	0.059
F14	ring	65	1	6028956	0.9999990	0.030
F15	ring	65	33	292332	0.99997	0.030
F16	ring	129	1	3056746	0.999997	0.015
F17	ring	129	65	75251	0.99990	0.015
F18	ring	257	1	1563800	0.999995	0.008
F19	ring	257	129	19396	0.9996	0.008

The results in the table above are depicted graphically in Figure 7.1, Figure 7.2 and Figure 7.3. It must be noted that the functions plotted are in fact discrete functions of *N - the amount of nodes* in the networks although the points have been interconnected in order to depict the shape of the function and that the mesh network models E1 and E2 have been omitted from these comparisons since the hierarchical mesh model H is sufficiently representative of mesh network topology behaviour. Two topological parameters that can be compared and evaluated in terms of their respective contribution to reliability and performability are the amount of nodes (*N*) and the end-node position (*i*) in the network. The results for models G, H, E1 and E2 suggest that for

star, mesh and hierarchical mesh topologies the topological parameters *N* and *i* do not influence the reliability or the merit figure because in these topologies the failure path between the two end-nodes is not changed with the addition of more nodes. Every end-node is connected via its own dedicated path and the amount of components and units in this dedicated failure path stay the same irrespective of *N*. This is not the case for ring topology networks that indicates a strong dependency of both reliability and merit figure on the amount of nodes in the network since there are only two paths connecting any two end-nodes. As the ring network grows in size the shared failure path includes more links and switches increasing the failure rate. Further every switch acts as a data source sharing the same path leading to more traffic running over the two common paths as the node count increases.

From Figure 7.1 it can be seen that with *N=129* the ring is almost just as reliable as a star network with MTTF approximately 90 thousand hours despite the existence of two redundant paths in the ring topology and the lack of any redundancy in the star topology. As the network size increases (*N>129*) the ring network topology becomes less reliable than the star topology. It is also clear that the hierarchical mesh is the most reliable of the topologies with *MTTF>250 million hr*, however for *N≤17* the ring also offers very high *MTTFs* ranging between 4 million to 120 million hr with reliability displaying asymptotic behaviour decreasing sharply as the ring size doubles until *N≥33* where the reliability deteriorates less significantly as the ring size increases.

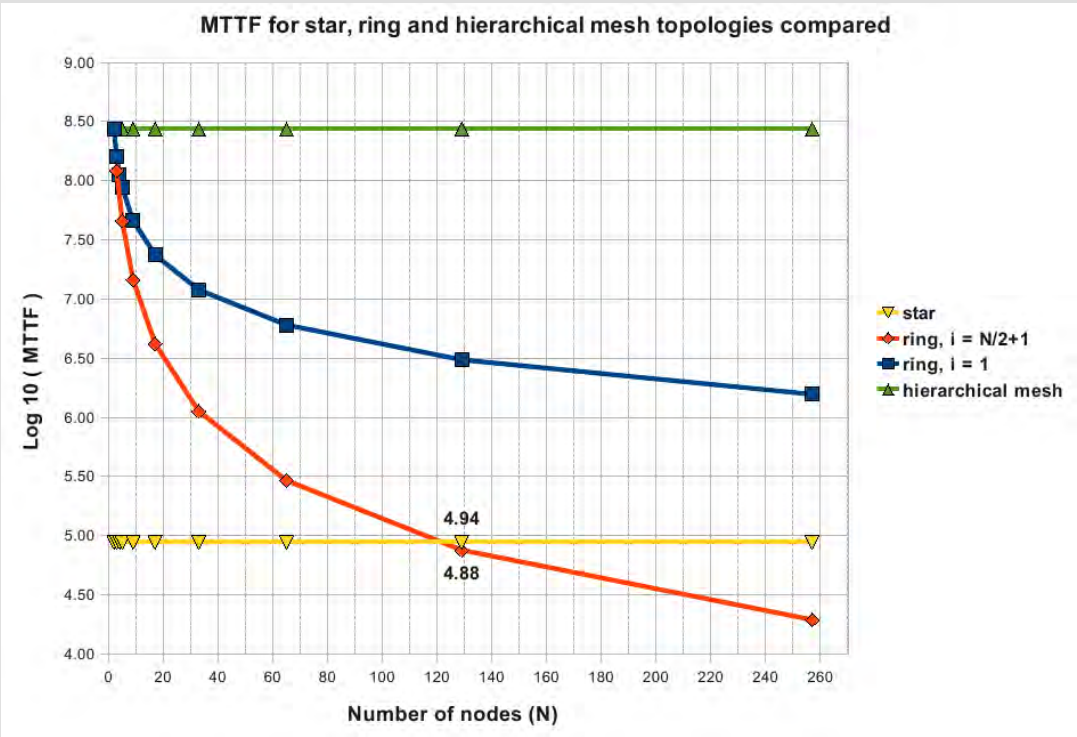

Figure 7.1. Reliability comparison of network topologies

Figure 7.2 compares availability for ring and hierarchical mesh network topologies. Assuming a *MTTR = 8 hr* and that a typical design criteria for high availability networks is six-nines [1] it can be seen from the availability plots that the hierarchical mesh and ring topologies for *N≤17* are all suitable for networks requiring six-nines availability. Not depicted on the graph, the star topology network and ring topologies with *N≥33* (for nodes located near the centre of the ring) have availability of less than five-nines.

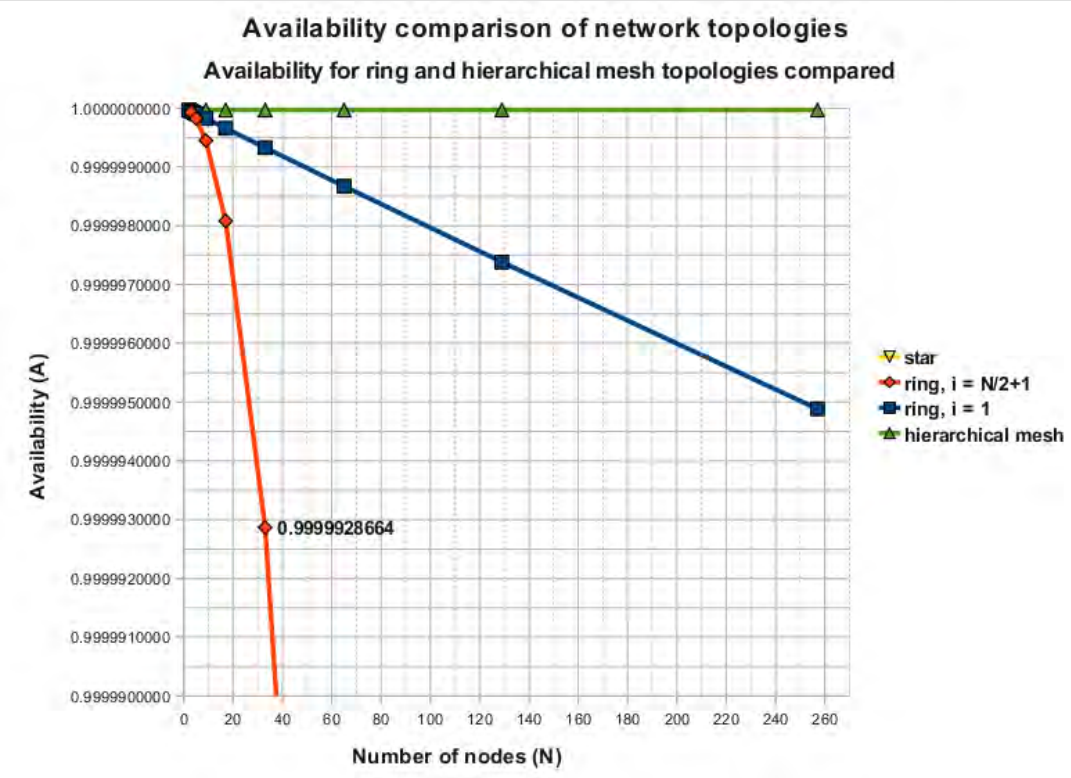

Figure 7.2. Availability comparison of network topologies

Figure 7.3 compares the merit or performability figure for the different network topologies. It is observed that for the star and hierarchical mesh topologies the merit figure *M* is close to unity for all network sizes discussed. For ring topologies the merit figure also exhibits asymptotic behaviour similar to that observed for the *MTTF*. The performability deteriorates sharply even for small rings with $N \geq 3$, this is to be expected because for $N=3$ the path carries the traffic from two switches and therefore the link utilisation is twice as high as topologies where every node is connected through a dedicated path. For rings with size $N \geq 17$, as the data load through the common ring path increases, the merit figure is already ten times smaller than for a similar size star or hierarchical mesh network.

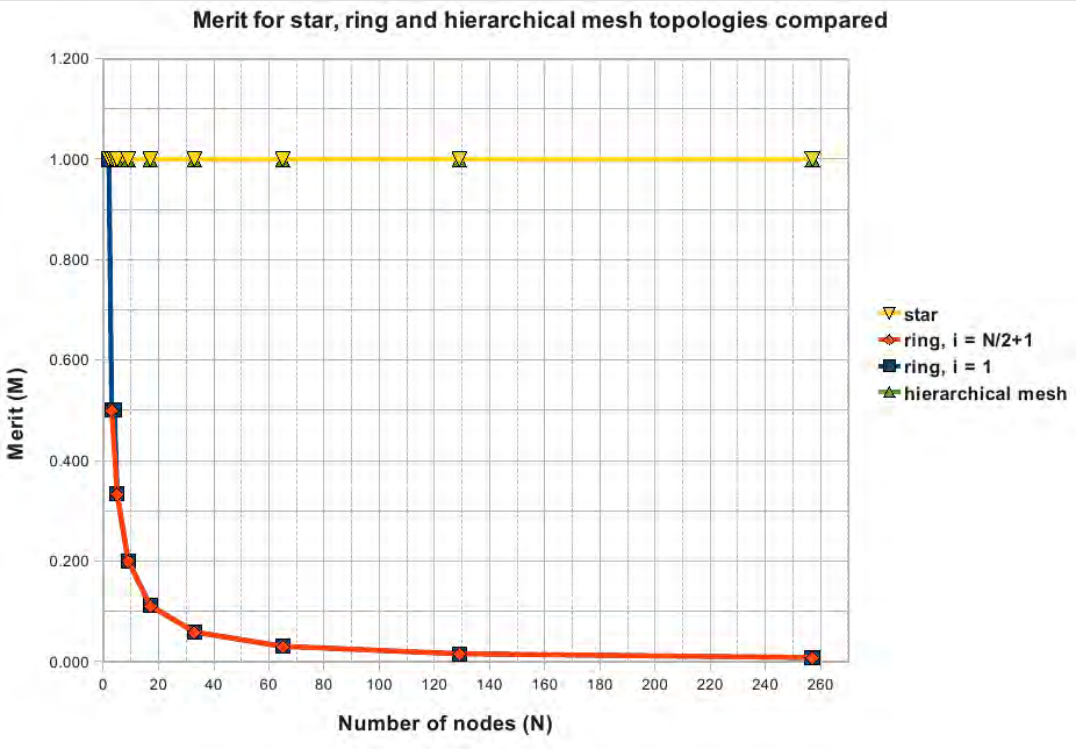

Figure 7.3. Merit comparison of network topologies

It has been noted that for ring topologies the reliability and availability relative to a specific end-node is dependant on its entry position into the ring. A comparison of the role of end-node position (*i*) on the reliability between two end-nodes for the ring network models F2-F19 is shown in Figure 7.4. For ring size $N \geq 17$ the reliability and availability for an end-node position at the centre of the ring ($i = N/2 + 1$) compared to the end-node position located directly downstream of *sw_0* ($i = 1$) begin to differ substantially. The reliability for the end-node position at $i = 1$ remains relatively high for most values of ring size N and remains above nine-fives for all ring sizes including $N=265$. However, the graph for merit figures above indicates that there is no difference in the merit figure for the relative end-node positions given a certain ring size.

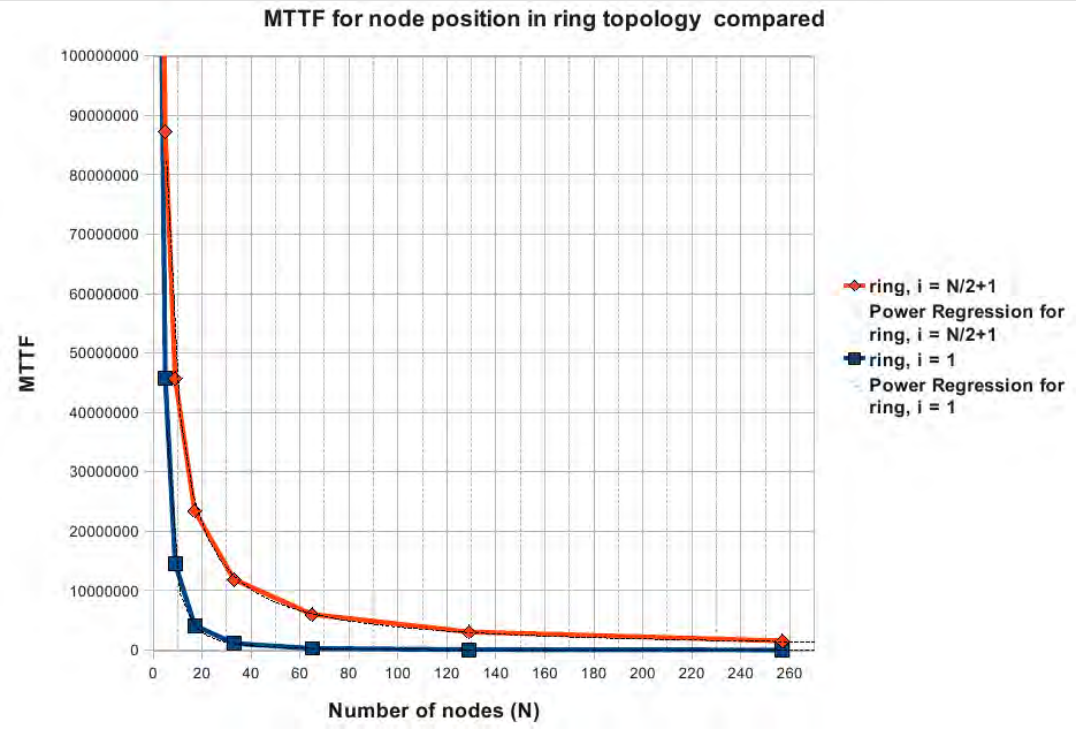

Figure 7.4. MTTF comparison for node positions in ring topology

It has been pointed out that the *MTTF* and *M* values for ring networks when plotted against the rings size (*N*) exhibit asymptotic behaviour. Using a graph fitting program [163] and noting the asymptotic behaviour as that of a polytropic equation while comparing to Equation 2.23 in [20] a polytrope plot fit was performed using Equation 7.1 below:

$$MTTF(N') = \frac{a}{(N'-1)^b} \tag{7.1}$$

In the above equation N' is the actual ring size including *sw_0* where N has been defined to exclude *sw_0* as explained in Section 5.1. The N' in Equation 2.23 is similar to N as defined in this investigation and the results can be compared noting that $N = N' - 1$ where $N' \geq 2$ and N is a positive integer. The coefficients determined by the plot fit program applied to the *MTTF* data generated from the ring topology models are shown in Table 7.4 along with the corresponding fitting error statistics that provide an indication of the goodness of the fit.

Table 7.4. Ring topology plot fit coefficients for MTTF

i	R-squared	Coefficient	std err squared	95% confidence intervals
1	0.9996	a = 5.51055E+08	1.71168E+14	[5.19042E+08, 5.83069E+08]
		b = 1.13036E+00	3.01545E-04	[1.08787E+00, 1.17285E+00]
N/2 + 1	0.999995	a = 1.01259E+09	2.25823E+13	[1.00096E+09, 1.02422E+09]
		b = 1.92627E+00	1.54217E-05	[1.91666E+00, 1.93588E+00]

It can be concluded from the correlation coefficient (R-squared value) in the table above being very close to unity (0.9996 for *i = 1* and 0.999995 for *i = N/2 + 1* respectively), that the calculated curve values for the two plots fit the data well [164]. The standard error squared calculated for the b-coefficients that determines the curve shape is also very small indicating that the b-coefficient is well known. However, the standard error squared for the scaling factor a-coefficient determining the asymptotic y values (very large *MTTFs* for small values of *N*) indicates that this value is not accurate and can vary significantly without affecting the plot fit [164].

Since the *M* values form a similar graph to the *MTTF*, a polytropic Equation 7.2 plot fit was also applied to the *M* values with the coefficients determined by the plot fit program applied to the *M* data shown in Table 7.5 along with the corresponding fitting error statistics that provide an indication of the goodness of the fit.

$$M(N') = \frac{a}{(N'-1)^b} \tag{7.2}$$

Table 7.5. Ring topology plot fit coefficients for M

R-squared	Coefficient	std err squared	95% confidence intervals
0.9992	a = 1.30313E+00	1.12165E-03	[1.22118E+00, 1.38508E+00]
	b = 8.62769E-01	2.89259E-04	[8.21153E-01, 9.04385E-01]

From the above table it is seen that the correlation coefficient (R-squared value) is very close to unity (0.9992) and therefore it is concluded that the calculated curve values fit the data well [164]. The standard error squared calculated for the b-coefficient that determines the curve shape is also very small indicating that the b-coefficient is well known. Similarly the standard error squared calculated for the scaling factor a-coefficient determining the asymptotic y values is also small indicating that the a-coefficient is also well known [164].

7.4. Evaluation of reliability models

When comparing results obtained with existing literature it must be noted that a unique approach was followed in the investigation where the entry-nodes into the network were modelled not to fail as indicated in block diagram a) Figure 7.5. The reason this approach was followed as discussed in Section 5.1 is that initial tests indicated that the reliability of *sw_0* and *sw_i* and links attaching the end-nodes *h0* and *hi* into a network topology is a limiting factor in calculating connection reliability as can be observed from [28], [35], [31]. This is an important observation that helps to compare numerical data from literature. There are also different definitions of network parameters that must be taken in consideration. For example, it was pointed out in the previous section that the definition of the network size *N* excludes *sw_0* and that in similar studies *N′* [20] is used which implies that *N = N′ - 1*. It is also not possible to compare numerical result directly since the underlying assumed component *MTTF* and *MTTR* figures (with the exception of [28], [35], [31]) are seldom given or are denoted as failure probabilities but without reference to the assumed time units or failure rates [26], [23], [20] in order to draw direct comparisons with failure rates assumed.

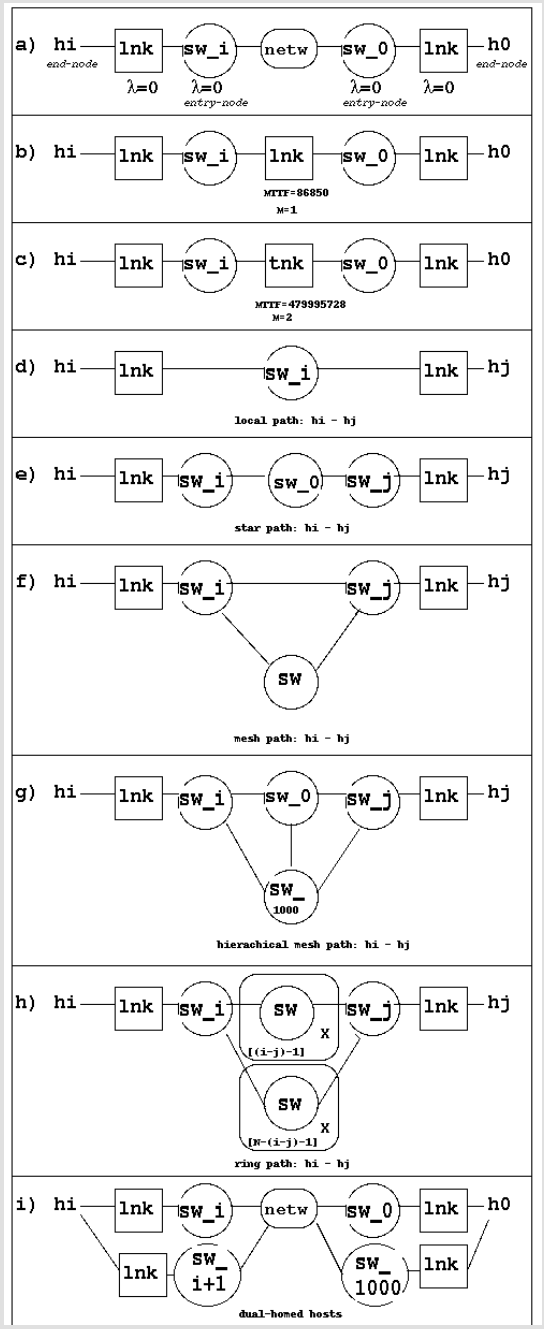

Figure 7.5. Evaluation block diagrams

It is clear that redundancy mechanisms must be extended from the entry-node to the connecting end-nodes to gain a significant benefit from redundant network topologies as network availability remains limited to three-nines or four-nines as demonstrated by models B and G and confirmed by [1], [28], [24]. The lack of redundancy in the star network topology leads to relatively low availability (four-nines as determined with model G) confirming similar findings in [1], [28], [35], [31].

The hierarchical mesh network offers the highest availability (seven-nines) as well as performability similar to findings in [28], [155], [35], [31]. Both these metrics can be accurately predicted irrespective of the position of the entry-node in the mesh as suggested by Equation 2.24 from [20]. The same observation based on Equation 2.24 relating to the independence of

network size or end-node position in determining availability can be made for the star and hierarchical mesh network topologies which is in agreement with the results obtained for these models. Although for reasons mentioned numerical results can not be compared directly, the high availability of mesh networks and small ring networks Table 7.3 compared to [26] (*links = 8 or* $N' = 9$) is supported by the figures provided in Table 2.2.

Ring networks offer high availability (five to seven-nines) in accordance with finding in [35], [31] if the ring remains small to medium sized, however for larger rings ($N \geq 32$) the availability is highly dependant on the relative position of the entry-node in the ring as predicted by Equation 2.23 in [20]. Graph plots of *MTTF* and *M* values (Figure 7.4, Figure 7.3) obtained from ring topology F2 to F19 data exhibit a similar polytropic equation shape and were fitted to Equation 7.1 and Equation 7.2 with good fitting results. [32] discusses optimal ring network sizes based on delay and performance measures and also relates all-terminal reliability to ring size (N') with Equation 2.26. Accordingly, all-terminal performance degrades significantly as the ring size increases substantiating the finding depicted in Figure 7.3 with merit figure *M* fitted to Equation 7.2.

Numerical results from [35] summarised in Table 2.4 can not be directly compared to results in Table 7.3 since the entry-node reliability is the determining factor, however it is clear that the hierarchical mesh (topology 3) is superior to the ring (topology 2) in Figure 2.10, a finding that is also supported by this investigation. Similarly numerical results from [31] summarised in Table 2.5 for ring with $N' = 6$ support the finding that small rings ($N \leq 5$) exhibit similar availability to mesh and hierarchical mesh networks.

In all the topologies investigated the reliability and performability can be increased significantly by introducing trunk (or redundant) links instead of single links interconnecting the various nodes. This is because the *MTTF* of a trunk link as represented by model C, for example, increases the *MTTF* of an interconnecting path between two nodes from *MTTF = 86850 to 479995728 hr* (Section 7.3). The magnitude of improvement is couple of orders for the star topology model where "the network" connecting *sw_i* to *sw_0* consists of a single link in block diagrams b) and c) in Figure 7.5. This finding is supported to some extent by Equation 2.24 from [20] where "m" used in the equation represents the amount of redundant star links although it is not clear how the "active star" differs from a typical star network topology in an Ethernet network. The use of redundant links do not only significantly improve the link reliability, but it also doubles the performability as measured by the merit figure *M* since the bandwidth available is doubled through the link aggregation techniques and protocols discussed in Section 2.2.1.3 and standardised in [46].

When evaluating the reliability and performability models and the results derived from them it is important to recall the basic assumptions and simplifications in the models that were introduced through Chapter 5 and summarised below for brief discussion.

- **Hot versus cold-standby modelling:** All the network reliability models assume that the blocking link in Figure 5.2 is in a standby state and therefore cannot fail because it is not operational. From reliability modelling literature it is not clear if this assumption is correct since the blocking port is operational even when it is in the blocking state. Failures can occur irrespective of the fact that the link is not carrying any data besides for spanning tree messages (Section 2.2.1.4). The effect this has on the model must be evaluated from a functional point of view. In the scenario where the blocking link can fail during the time that the least-cost path is operational a repair action will be initiated. If the blocking link cannot fail during this blocking time, when it does become active the time to failure will only commence when the link is activated. There is also the chance that although the blocking link has failed, and because there is no detectable interruption to data flow, the failure will remain undetected and

a repair action will not be initiated while a latent failure will exist. Whatever the modelling choice made, the assumption is unlikely to have a significant effect on the overall reliability calculation since *MTTF* is so much larger than *MTTR* and because the least cost-path selection algorithm in spanning tree will immediately reactivate the repaired primary path. It must also be noted that the validation testing configurations in Figure 6.5 and verified in Section 6.3.2 does not differentiate between cold and hot stand-by conditions failing links irrespective of their locations.

- **Shared repair facility for every path:** The network reliability models (Figure 5.3) assume there is a separate repair facility for every path - when a path fails (as a unit) the entire path is repaired. The alternative would be that every component will be repaired at an individual rate which would introduce more variability in the distribution of repair events and complexity in the models, either way the path will not recover if it has not been repaired either as a unit or on failed component level. Another option is to model only one global repair facility. Since a constant repair rate of $MTTR = 8$ has been assumed for all network components and units the effect of this assumption is irrelevant since only one *MTTR* is used in all the models.

- **Ring model allows failed units to fail:** In order to accommodate large ring networks with many serial connected components it was necessary to introduce one or two simplifications to reduce the Markov state-space matrix that must be solved (Section 5.4.1). For this purpose it was decided to aggregate the failure rates of all the serial switches and links in a path, the consequence being that the individual failed state of a specific switch or link is ignored. It is therefore possible that a failed switch/link can fail again and that the failure rate will remain constant despite the fact that it must be decremented after failure of a unit. The effect of this simplification is that the *MTTF* for the ring network topology may be slightly underestimated and most significantly noticeable in small rings networks ($N \leq 7$) that has been confirmed to have very large *MTTFs* similar to mesh and hierarchical mesh network topologies (Table 7.3), whereas for larger rings the probability that the same switch or link unit may fail twice before repair action has occurred given the large *MTTF* to *MTTR* ratio is much smaller. This effect can be demonstrated by comparing the MTTF simulation results with predicted results in Table 7.1 and noting that the availability (A) achieved in simulations are generally slightly higher than predicated. The effect of this simplification for the purpose of comparing results between the different topologies is therefore considered to be negligible.

- **Ring model merit figure M is calculated from an assumed fixed point on failure:** The merit figures *M1* and *M2* in Figure 5.10 are calculated from a fixed point where $i = N/2 + 1$ dividing the ring in to two equal size paths, where as to be accurate it should be calculated relative to the location of the failure position in the ring. As with fail states simplification discussed above, for bigger ring networks this requires the Markov state-space matrix to become very large. It should be noted however that this assumption is valid under normal conditions when there are no failures and the spanning tree algorithm blocks the link downstream of $i = N/2 + 1$. Again considering the large *MTTF* to *MTTR* ratio (at least three order of magnitude), the time that the model would spend in the repair state is very small and therefore incorrect results are negligible for the ring sizes being modelled. The performability metric M will be determined by the amount of switches in the serial path under normal conditions.

- **Spanning tree algorithm does not fail:** This assumption that the redundancy controller cannot fail can have a significant influence on the reliability and performability behaviour of all the network topology models (Section 6.3.3) and will be the topic of investigation for future work - see Section 8.4.

- **Spanning tree topology convergence time is negligible:** The assumption that the spanning tree convergence time (Section 2.2.1.4) is predictable (around three seconds) and negligible in

comparison to the actual time scales ranging from *MTTR = 8 hr* and downtime to *MTTF = 250 million hr* proved to be problematic when conducting network simulation tests using seconds as a time base in the validation of the ring networks and led to the development of the PFM to calculate network availability from the validation test data (Section 6.3.3, Section 6.3.4). Problems with the non-ideal behaviour of spanning tree implementation [32], [45] along with the influence of spanning tree algorithm failures will also be investigated in future work.

The network reliability models are based on the 2-terminal reference model in Figure 5.1 for calculating reliability and the merit figure (Section 2.6, Figure 3.1), with a specific reference point selected that would represent a server host as an end-point in networks similar to Figure 1.6 and Figure 2.10. In this general topology the server host, *h0*, is always located at the top of the network hierarchy at a central least-cost path and all the network topology models constructed. The applicability of these models to a more general location of the server host *hj*, attached to entry node *sw_j* can be evaluated. The path for each topology between a client node *hi* and a more generalised server node *hj* are depicted in block diagrams d) to h) of Figure 7.5. By visual inspection and comparison to the topology models developed the following conclusions can be drawn:

- The local path network (hosts attached to the same entry-node *sw_i*, *sw_0* or *sw_j*) in block diagram d) Figure 7.5 can be represented by reliability model B.
- The star network topology in block diagram e) Figure 7.5 as represented by reliability model G cannot be generalised. When *hj* moves to *sw_j* another additional switch *sw_j* is added in serial to the topology and the resulting reliability will become lower.
- The mesh network topology in block diagram f) Figure 7.5 as represented by reliability models E1/E2 can be generalised from path *hi* - *h0* being equivalent to path *hi* - *hj*.
- The hierarchical mesh network topology in block diagram g) Figure 7.5 as represented by reliability model H cannot be generalised. When *hj* moves to *sw_j* another additional switch *sw_j* is added in serial to the topology and the resulting reliability will become lower.
- The ring network topology in block diagram h) Figure 7.5 as represented by reliability models F2-F19 can be generalised from path *hi* - *h0* being equivalent to path *hi* - *hj*.

The above clearly indicates that some network topologies, namely the star and hierarchical mesh topologies are optimised towards maximum reliability and performability in a typical TCP/IP server/client connection.

Chapter 8. Conclusions and Recommendations for Future Work

8.1. Concluding remarks

There are many factors that contribute to a network based system's reliability. The network is also dependant on the infrastructure and supporting environment. The power supply to network equipment is known to contribute significantly [28] to network availability limiting the network availability to around three to four-nines. Another major factor is the reliability of the hosts and applications connecting to the network.

The objectives of this investigation are stated in Section 1.4 and focus on the analyses and modelling of the reliability and performance of a TCP/IP over Ethernet network - but limited to the "static reliability" i.e. physical topology and connectivity between hosts. Accordingly, the TCP/IP server-client relationship was chosen inside a 2-terminal reference frame (Section 2.6). In this type of network the physical layer, generally referred to as Ethernet (Section 2.2.1.1), is packet switched and consists of switches (Section 2.2.1.2) using source and destination MAC tables to switch data frames between the different switch ports. In the proposed reliability modelling approach the nodes consist of switches and the end-nodes consist of the attached hosts as depicted in Section 5.1. The spanning tree protocol (Section 2.2.1.4) running between switching nodes prevents loops in these networks and also behave like redundancy controllers, making it possible to build redundancy into the network to improve the network reliability.

Although the traditional approach to network topology reliability studies is to make use of graph theory methods this investigation used a Markov modelling method and the Tangram modelling environment to construct analytical Markov models and state transition matrices (Section 2.4, Chapter 3). Graph methods tend to be limiting in the complexity of networks that can be constructed (Section 2.6) especially when dealing with failure dependencies and lack flexibility since they rely on recursive algorithms that deal with connectivity irrespective of network topology relying on idealised abstractions of network elements. Graph methods can be applied to all network topologies where Markov models are specifically constructed for each topology of network. The Markov model specification tool in the Tangram environment makes it possible to use Markov state-space techniques applied to both connectivity state (reliability) as well as queue and packet stream analysis that evaluates dynamic effects that can influence reliability. It was assumed that the network redundancy controller never fails, but it is possible to assign a probability of failure to the redundancy controller and recalculate network *MTTFs*. Accordingly it is envisaged that for future work (see discussion in Section 8.4) the reliability models developed in this investigation can be expanded to deal with spanning tree convergence failures as well as other dynamic effects of congestion, packet loss and retransmissions. It is also possible to employ purely experimental methods to calculate network availability based on simulated equipment failures and data packets dropped to derive reliability and performability metrics. The shortcomings in this approach discussed in Section 6.3.1 could be seen from difficulties in conducting simulations to determine the availability of systems with very large *MTTFs*. It was found that graph and Markov modelling methods can predict reliability measures with high accuracy within a much shorter timespan than network simulations given the same computational resources and constraints.

Network topologies are compared in terms of performability, two different measures of interest: reliability and performance are evaluated concurrently (Section 2.5, Section 5.1). The performability is defined as a measure of the bandwidth utilisation given the availability of a

specific network topology. It is important to understand the purpose of these two measures of interest. The reliability measures are an indication of the reliability of network connectivity. Performability is a **relative measure** that includes the bandwidth utilisation as a means of indicating network congestion. The reliability is related to equipment failure rates and redundancy mechanisms, performability is also related to reliability - but also includes bandwidth and bandwidth utilisation resulting from the sharing of links. The performability of the models developed in this investigation are predominantly determined by the amount of shared links per node (Section 7.4). It must be noted that a lower value for the performability metric *M* as observed for the ring topology models is not necessarily detrimental to performance of the network if the bandwidth utilisation does not exceed the channel capacity or node switching capacity. It does however indicate a higher probability that congestion can occur given the same network loads that may lead to data flow reliability issues as indicated in [32].

In Chapter 6 it has been demonstrated that it is possible to validate some of the reliability and performance models developed based on realistic reliability data for the network equipment using a network simulation framework such as OPNET. However, because redundant networks generally display very large mean time to failure (*MTTF*), it is computing intensive and time consuming to use simulations to verify reliability models and customised methods must be developed (Section 6.3.4) to eliminate non-ideal simulator behaviour as exemplified by the inconsistent convergence times of the spanning tree algorithms implemented in the simulator. It was also observed that in the case of very low application layer bandwidth utilisation the spanning tree traffic become a significant factor in determining the merit figure in the larger ring topologies and was significantly lower that the values predicted by the models.

In Section 7.4 comparisons between existing literature and the findings of this investigation were made. It is possible to correlate model predications and generalised indicators with the existing literature. It can be confirmed that the conclusions that follow in sections Section 8.2 and Section 8.3 are consistent with findings in previous studies, with more depth and particulars regarding topological parameters and their influence on network reliability and performance added in the following areas:

- comparison between popular network topologies with regard to the network size;
- the role of the position of a host in the network;
- network performability as a factor to be considered along with reliability.

The last objective of this investigation - recommendations and design guidelines on the best practice for designing TCP/IP over Ethernet networks with high reliability and performance as an important criteria, will be discussed next.

8.2. Main topological factors that influence network reliability and performance

With reference to the results analysed and evaluated in Section 7.3 and Section 7.4 the main topological factors that influence network reliability and performance are summarised below with references to the applicable literature:

8.2.1. Redundant network nodes and links

Irrespective of the topology employed a fully redundant path must be present between two end-nodes for high network availability as indicated in block diagram i) Figure 7.5. It has been

demonstrated that non-redundant entry-nodes and connecting links to end-nodes constitute a limiting factor in determining network availability [28], [1], [35], [31], [24]. It has also been shown that the hierarchical mesh topology with an additional redundant node at the top of the network hierarchy offers the highest availability [28], [155], [35], [31], while star network topology exhibit the lowest availability because of the lack of any redundancy. Performability of a link doubles when redundant links (trunks) are used [46]. When trunk links are used instead of single links, star network topology availability improves [20] by three orders of magnitude and the performability doubles.

8.2.2. Network diameter

Network diameter must be minimised in order to obtain high availability. It has been demonstrated by example of the ring network topology that for high availability the network diameter must be kept as low as possible [20], [19], [32], [35], [31], [24] i.e. for the ring topology N should be less than 17. It is also shown in this investigation that the availability in the ring network for $N \geq 33$ is highly dependant on the relative position of the entry node in the ring, with the availability being the highest at node positions nearest to the server end-node. Performability also degrades significantly as the ring size increases [32].

8.3. Design guidelines

Derived from the principle findings in this investigation (Section 7.3, Section 7.4) as summarised in Section 8.2 the following guidelines are proposed for designing TCP/IP over Ethernet networks that exhibit high reliability and performance:

- **Node and Link redundancy:** Ensure that common points of failure are minimised and a full redundant path exists between any two entry-nodes. This requires selecting a topology that offers a redundant path between entry-nodes that may consist of a combination of nodes and links.

- **Link aggregation**: Redundant links or trunks should be employed where various nodes share a link as is typical of a ring network topology. In general, trunks improve both network availability and performability.

- **Connectors:** Since there are two connectors for every network link (Figure 4.1) and the reliability of the connector is assumed to be orders of magnitude lower than the cable, additional connectors and cross connecting panels should be avoided and eliminated wherever possible. Selection of robust, high quality connectors that do not degrade due to various operational and environmental factors are highly recommended.

- **Dual-homed hosts:** Redundancy mechanisms must be extended from the entry-node to the connecting end-nodes or hosts to achieve network availability higher than three-nines. In particular the "dual-homing" (block diagram i) Figure 7.5) of servers and other critical devices using NIC bonding or teaming is essential. The hierarchical mesh topology is well suited for this purpose on the server end-node side of the connection, since a redundant node *sw_1000* is present at location *sw_0*. To extend dual-homing of hosts to the entire network implies very high additional costs because it essentially implies a fully duplicated network - each host must be connected into the network through two independent entry-nodes.

- **High availability topology:** The hierarchical mesh network offers the highest availability in the order of seven-nines and high performability and should be preferred in high availability designs where performance is an important criteria. This topology requires an additional

redundant node at the server end-node and can be dual-homed to the server-end node to extend the high availability and performability to the server end-node. Besides for the redundant end node, twice as many links to the redundant switch configuration is required when compared to the ring topology with additional associated costs.

- **Star topology considerations:** The star should not be used in IP networked systems requiring more than four-nines availability. However, when the star topology is selected the availability can be increased from four-nines to seven-nines and the performability can be doubled by using redundant (trunk) links. Star topologies can also be used where the performance of the network is a more important design requirement than high availability.

- **Ring topology considerations:** Ring networks offer high availability of five to six-nines if the ring remains small to medium sized when the size does not exceed $N=17$. The performability of a ring network of this size is much lower than the equivalent hierarchical mesh or star networks, therefore where performance is an important configuration or high bandwidth utilisation is expected redundant (trunk) links should be used to interconnect the nodes in the ring. The availability of the ring network relative to end-node position for $N \geq 33$ becomes a function of the node position in the ring and is therefore variable. However, of all the topologies compared small to medium sized rings ($N \leq 33$) and assuming low performance requirements need the fewest amount of links and would be considered a cost effective design option. Cost and economical considerations are separate design considerations and are not a part of this investigation.

8.4. Future work

In conclusion the following areas for future research work have been identified in this investigation:

- The focus in this investigation has been on the role of the network topology, that is "static reliability" involving equipment failures and the network's ability to heal or repair connectivity. However it is clear that there are many factors that play a role (Table 1.1), including but not limited to congestion and delay or "dynamic reliability" as represented by reliable packet flow, logical network partitioning, IP addressing, routing and QoS measures (Section 2.2.3.3). It is envisaged that in future work a more comprehensive reliability model can be constructed including all these factors [2], [27], [140], [34], [69] and then modelling the OSI layers (including the physical topological factors) as serial connected reliability blocks in Figure 3.1.

- It is clear from result and recommendations in this investigation (Section 7.4, Section 8.3) that there are certain popular network topologies that can be improved by addressing certain weaknesses in the topologies and slightly modifying the topology. One topology that can be investigated in future work is the spoked-ring network that may be a good alternative to the ring topology for improving the reliability and performability of large ring networks. Since the full redundant star with dual-homed hosts aught to be the most reliable option another alternative that can be investigated is the collapsed redundant star topology with a single redundant switching node. It should also be noted that when the dual-homing end node is introduced into the model (block diagram i) Figure 7.5), the structure of the network topology is modified and that the Markov models for the networks must to be adjusted accordingly.

- Various assumptions were made and modelling issues were identified related to the IEEE based spanning tree protocols (Section 2.2.1.4, Section 6.3.3). Future work will include building a model where the spanning tree redundancy controller can also fail and spanning tree dependencies can be investigated, spanning tree packet level modelling to fully investigate

the spanning tree convergence time behaviour, clarifying the confusing naming of standards and related terminology found in literature versus actual IEEE implementations in COTS equipment and testing and improving the existing spanning tree implementations in popular network simulation frameworks OPNET and OMNet (Section 2.9).

- It has been demonstrated that redundancy mechanisms must be extended to the hosts (Section 8.2) to significantly improve network reliability. It is envisaged that in future work fully redundant spanning tree compliant network drivers can be evaluated and developed if required. Also see [2] for related discussion of optimised network redundancy protocols that should be included in that investigation.

Bibliography

[1] E. Basart.Shoretel.*"Building Reliable IP Telephony Systems"*,`http://www.shoretel.com/resource_center/white_papers/Building_Reliable_IP_Telephony_Systems_.html`.Accessed on the Internet on 23 April 2011.

[2] M. Huynh.S. Goose.P. Mohapatra.January 2010.*"Resilience technologies in Ethernet"*.Computer Networks.Vol. 54, No. 1.pp. 57–58.

[3] A.S. Tanenbaum.2003.*"Computer Networks"*.Prentice Hall.4th edition.pp. 8, 16-17, 37-44, 68, 147-151, 271-292, 323-325, 328-336, 406-415, 425-427, 456-461, 532-556, 557-573.ISBN 0-13-066102-3.

[4] J. Ellis.C. Pursell.J. Rahman.2003.*"Voice, Video, and Data Network Convergence"*.Elsevier.1st edition.ISBN 978-0-12-236542-3.

[5] Cisco Systems.2010.*"Industrial Ethernet: A Control Engineers Guide"*,`http://www.cisco.com/web/strategy/docs/manufacturing/industrial_ethernet.pdf`.Accessed on the Internet on 23 April 2011.

[6] C. LeBlance.2000.*"Industrial Ethernet Book: The Future of Industrial Networking and Connectivity"*,`http://www.iebmedia.com/index.php?id=4026&parentid=63&themeid=255&hft=2&showdetail=true&bb=1`.Accessed on the Internet on 23 April 2011.Industrial Ethernet Book: Issue 2 (March 2000).

[7] M. Felser.2002.*"The Fieldbus Standards: History and Structures"*,`http://felser.ch/download/FE-TR-0205.pdf`.Accessed on the Internet on 23 April 2011.

[8] S. Lammerman.2008.*"Ethernet as a Real-Time Technology"*,`http://www.lammermann.eu/wb/media/documents/real-time_ethernet.pdf`.Accessed on the Internet on 23 April 2011.

[9] C. Hoga.2007.*"New Ethernet Technologies for Substation Automation"*.Power Tech, 2007 IEEE Lausanne.pp. 707 - 712.

[10] L.G. Roberts.November 1978.*"The Evolution of Packet Switching"* .Proceedings of the IEEE.Vol 66, No 11.pp. 1307-1313.

[11] Internet Engineering Task Force.1996.*"RFC 2026: The Internet Standards Process -- Revision 3"*,`http://tools.ietf.org/html/rfc2026`.

[12] Internet Engineering Task Force.1981.*"RFC 793: Transmission Control Protocol"*,`http://tools.ietf.org/html/rfc793`.

[13] Internet Engineering Task Force.1980.*"RFC 768: User Datagram Protocol"*,`http://tools.ietf.org/html/rfc768`.

[14] Internet Engineering Task Force.1981.*"RFC 791: Internet Protocol"*,`http://tools.ietf.org/html/rfc791`.

[15] IEEE 802.3 Ethernet Working Group.1983.*"CSMA/CD (Ethernet) ACCESS METHOD"*,`http://standards.ieee.org/about/get/802/802.3.html`.

[16] Cisco Systems.2002.*"Strategic Directions: Introduction to 10 Gigabit Ethernet"*,`http://www.cisco.com/warp/public/cc/techno/lnty/etty/ggetty/tech/10gig_wp.pdf`.Accessed on the Internet on 23 April 2011.

[17] F.T. Boesch.1986.*"Synthesis of reliable networks"*.Circuit Theory, IEEE Transactions on Reliability. Vol. R-35, No. 3.pp. 240–246.

[18] C. Hwang.F. Tillman.M. Lee.August 1981.*"System-Reliability Evaluation Techniques for Complex/Large Systems - A Review,"*.Reliability, IEEE Transactions on Reliability. Vol. R-30, No. 5.pp. 416–423.

[19] R. Wilkov.June 1972.*"Analysis and design of reliable computer networks"*.IEEE Transactions on Communications.Vol. COM-20, No. 3.pp. 660-678.

[20] G. Ray.J. Dunsmore.March 1988.*"Reliability of network topologies"*.Proceedings of INFOCOM '88. Networks: Evolution or Revolution. Seventh Annual Joint Conference of the IEEE Computer and Communcations Societies.pp. 842-850.

[21] K. Leister.A. White.K. Hayhurst.1990.*"Using minimal spanning trees to compare the reliability of network topologies"*.NASA Technical Memorandum. TM-4208.

[22] R.H. Jan.F.J. Hwang.S.T. Cheng.1993.*"Topological optimization of a communication Network Subject to a Reliability Constraint"*.IEEE Transactions on Reliability. Vol. 42, No. 1.pp. 63–70.

[23] S. Belovich.September 1995.*"A design technique for reliable networks under a nonuniform traffic distribution"*.IEEE Transactions on Reliability. Vol. 44, No. 3.pp. 377–387.

[24] G. Weichenberg.V. W. S. Chan.M. Médard.November 2004.*"High-Reliability Topological Architectures for Networks Under Stress"*.IEEE Journal on Selected Areas In Communications.Vol. 22, No. 9.pp. 1830–1845.

[25] K. Lee.D. Lee.H. Lee.2012.*"Reliable Network Design for Ethernet Ring Mesh Networks"*.Journal Of Lightwave Technology.pp. 1–9.

[26] M. Cabarkapa.D. Mijatovic.N. Krajnovic.2011.*"Network Topology Availability Analysis"*. Telfor Journal.Vol. 3, No. 1.pp. 3–7.

[27] T. Yu.S. Chen.M. Ai.July 2007.*"“A Framework for Reliability Computation of the IP Network”"*.Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007).pp. 323–327.

[28] C. Oggerino.2001.*"High availability network fundamentals"*.Cisco Press.1st edition.pp. 98, 109, 111, 113, 167.ISBN 1-58713-017-3.

[29] Cisco Systems.May 2008.*"High Availability Campus Recovery Analyses"*,`http://www.cisco.com/en/US/docs/solutions/Enterprise/Campus/HA_campus_DG/hacampusdg.pdf`.Accessed on the Internet on 23 April 2011.

[30] Rockwell Automation.July 2008.*"Ethernet-to-the-Factory 1.2 Design and Implementation Guide, Cisco Validated Design"*,`http://www.cisco.com/en/US/docs/solutions/Verticals/EttF/EttF_bk.pdf`.Accessed on the Internet on 13 August 2013.

[31] M. G. Kanabar.T. S. Sidhu.July 2009.*"Reliability and availability analysis of IEC 61850 based substation communication architectures"*.2009 IEEE Power & Energy Society General Meeting.pp. 1–8.

[32] L. Zhang.R. Lehmann.Z. Wang.September 2009.*"A Risk Metric for Designing a Highly Reliable Real-Time Ethernet Network Control Technology Control Technology"*.Emerging Technologies & Factory Automation, IEEE Conference on.pp. 1–4.

[33] A. Xu.L. Jiang.Y. Chen.June 2009.*"Research of fault-tolerance technique for high availability Industrial Ethernet"*.International Conference on Information and Automation.pp. 301–305.

[34] D. Hou.N. Huang.Y. Chen.2010.*"An evaluation method for communication network topology reliability based on Markov model"*.International Conference on Educational and Information Technology.Vol 2.pp. 345–349.

[35] K.H. Niemann.July 2011.*"Availability calculation of meshed, Ethernet based Automation Networks"*.9th IEEE International Conference on Industrial Informatics.pp. 883–888.

[36] G.W. Scheer.D.J. Dolezilek.2000.*"Comparing the reliability of Ethernet Network Topologies in Substation Control and Monitoring Networks"*,`http://www.selinc.com/WorkArea/DownloadAsset.aspx?id=2584`.Accessed on the Internet on 23 April 2011.

[37] O. Kleineberg.M. Rentschler.September 2010.*"Redundancy Enhancements for Industrial Ethernet Ring Protocols"*.Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on.Print ISBN 978-1-4244-6848-5.

[38] Cisco Systems.May 2008.*"Campus Network for High Availability Design Guide"*,`http://www.cisco.com/en/US/docs/solutions/Enterprise/Campus/HA_recovery_DG/campusRecovery.pdf`.Accessed on the Internet on 23 April 2011.

[39] IEEE 802.1 Working Group.1997-2010.*"IEEE 802.1: Bridging and Management"*,`http://standards.ieee.org/about/get/802/802.1.html`.

[40] Cisco Sytems.2006.*"Understanding Rapid Spanning Tree Protocol (802.1w)"*,`http://www.cisco.com/image/gif/paws/24062/146.pdf`.Accessed on the Internet on 12 October 2013.

[41] Cisco Systems.2006.*"Understanding and Configuring Spanning Tree Protocol (STP) on Catalyst Switches"*,`http://www.cisco.com/image/gif/paws/5234/5.pdf`.Accessed on the Internet on 12 October 2013.

[42] Cisco Sytems.2007.*"Understanding Multiple Spanning Tree Protocol (802.1s)"*,`http://www.cisco.com/image/gif/paws/24248/147.pdf`.Accessed on the Internet on 12 October 2013.

[43] L. Fowler.2007.*"EE 122: Intro to Communication Networks: Spanning Tree Protocol"*,`http://www.eecs.berkeley.edu/~jortiz/courses/ee122/discussion/SpanningTree_discussion_2up.pdf`.Accessed on the Internet on 12 October 2013.

[44] M. Pustylnik.M. Zafirovic-Vukotic.R. Moore..*"White Paper: Performance of the Rapid Spanning Tree Protocol in Ring Network Topology"*,`http://www.ruggedcom.com/pdfs/white_papers/performance_of_rapid_spanning_tree_protocol_in_ring_network_topology.pdf`.Accesse on the Internet on 12 October 2013.

[45] P. Lapukhov.2010.*"Understanding STP and RSTP Convergence"*,`http://blog.ine.com/wp-content/uploads/2011/11/understanding-stp-rstp-convergence.pdf`.Accessed on the Internet on 12 October 2013.

[46] WG802.1 - Higher Layer LAN Protocols Working Group.2008.*"IEEE Standard for Local and metropolitan area networks--Link Aggregation"*,`http://standards.ieee.org/findstds/standard/802.1AX-2008.html`.

[47] J. Menga.2013.*"CCNP Practical Studies: Chapter 3. Trunking and Bandwidth Aggregation"*,`http://www.informit.com/library/content.aspx?b=CCNP_Studies_Switching&seqNum=22`.Accessed on the Internet on 6 October 2013.

[48] Cisco Systems.August 2013.*"Cisco Nexus 1000V Troubleshooting Guide"*,`http://www.cisco.com/en/US/docs/switches/datacenter/nexus1000/sw/4_2_1_s_v_1_4/troubleshooting/configuration/guide/n1000v_trouble.pdf`.Accessed on the Internet on 6 October 2013.pp 8.1.

[49] Nortel Networks.2004.*"White paper: What is Split Multi-Link Trunking?"*,`http://dutta.csc.ncsu.edu/csc791_spring07/wrap/Nortel-SMLT-whitepaper.pdf`.Accessed on the Internet on 12 October 2013.pp 8.1.

[50] C. Schluting.2009.*"Understanding NIC Bonding with Linux"*,`http://www.enterprisenetworkingplanet.com/linux_unix/article.php/3850636/Understanding-NIC-Bonding-with-Linux.htm`.Accessed on the Internet on 12 October 2013.pp 8.1.

[51] T. Rhodes.2010.*"Open Source Advocate: Linux NIC Teaming Recommendations"*,`http://useopensource.blogspot.com/2010/02/linux-nic-teaming-recommendations.html`.Accessed on the Internet on 12 October 2013.

[52] K. Mauldin.2013.*"5 reasons to use NIC Teaming with Windows Server 2012"*,`http://www.trainsignal.com/blog/nic-teaming-windows-server-2012`.Accessed on the Internet on 12 October 2013.pp 8.1.

[53] B Randal.P.A. Lee.P.C. Treleavan.June 1978.*"Reliability Issues in Computing System Design"*.Computing Surveys.Vol 10, No 2.pp. 124-165.

[54] D Buchmann.2008.*"Doctor's Thesis: Verified Network Configuration - Improving Network Reliability"*,`http://ethesis.unifr.ch/theses/BuchmannD.pdf?file=BuchmannD.pdf`.Accessed on the Internet on 23 April 2011.

[55] Cisco Systems.2007.*"Common Causes of Slow intraVLAN and interVLAN Connectivity in Campus Switch Networks"*,`http://www.cisco.com/application/pdf/paws/23637/slow_int_vlan_connect.pdf`.Accessed on the Internet on 23 April 2011.

[56] Internet Engineering Task Force.1994.*"RFC 1663: Integrated Services in the Internet Architecture: an Overview"*,`http://tools.ietf.org/html/rfc1633`.

[57] L. Jereb.1998.*"Network Reliability: Models, Measures and Analyses"*,`http://www.mcl.hu/mmrel/related/network-reliability-models-measures.pdf`.Accessed on the Internet on 23 April 2011.6th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks .pp. T02/1-T02/10.

[58] L. Jereb.P. Bajor.A. Kiss.1999.*"Network Reliability Analysis Based on Multilayer Models"*,`http://cntic03.hit.bme.hu/~jereb/1999/jbk99a.ps`.Accessed on the Internet on 23 April 2011.7th International Conference on Telecommunication Systems/Modelling and Analysis (1999).pp. 490-500.

[59] R. Li.H. Ning.July 2009.*"Reliability Testing Technology for Computer Network Applications"*.Reliability, Maintainability and Safety, 2009. ICRMS 2009. 8th International Conference on .Print ISBN: 978-1-4244-4903-3.pp. 1169-1172.

[60] F. Jawad.E. Johnsen.1995.*"Performability: The vital evaluation method for degradable systems and its most commonly used modelling method, Markov Reward Modelling"*,`http://www.doc.ic.ac.uk/~nd/surprise_95/journal/vol4/eaj2/report.html`.Information Systems Engineering Journal..Accessed on the Internet on 6 October 2013.

[61] E. De Souza e Silva.H.R. Gail.January 1989.*"Calculating availability and performability measures of repairable computer systems using randomization"*.Journal of the ACM.Vol 36, No 1.pp. 171–193.

[62] K. Trivedi.February 1994.*"Markov reward approach to performability and reliability analysis"*.Proceedings of the Second International Workshop on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems.Print ISBN: 0-8186-5292-6.pp. 7 - 11.

[63] H.H. Amer.M.S. Moustafa.R.M. Daoud.September 2009.*"Performability analysis for fault-tolerant WiFi communication system"*.IEEE Conference on Emerging Technologies & Factory Automation.Print ISBN: 978-1-4244-2727-7.pp. 1-4.

[64] B. Cho.H. Youn.E. Lee.2009.*"Performability analysis method from reliability and availability"*.Convergence and Hybrid Information Technology, International Conference on .pp. 1-7.

[65] J. Meyer.2009.*"Defining and evaluating resilience: A performability perspective"*. Proceedings of International Workshop on Performability.pp. NA.

[66] R. Smith.1988.*"Performability analysis: measures, an algorithm, and a case study"*.IEEE Transactions on Computers.Vol 37, No 4.pp. 406-417.

[67] A.A. Guillermo.U. Mustafa.M. Arif.2001.*"Efficient verification of performability guarantees"*.Proceedings of 5th International Workshop on Performability Modelling of Computer and Communication Systems.

[68] V. Catania.L. Milazzo.1999.*"Enhancing reliability in an industrial LAN: design and performability evaluation"*.IEEE transactions on Industrial Electronics.Vol 37, No 6.pp. 433-439.

[69] M. Mohamed.A. Mahmoud.August 2013.*"Computation of Communication Network's Reliability Based on Service Interrupt"*.International Journal of Pure and Applied Research in Engineering and Technology.Vol. 1, No. 11.pp. 1–8.

[70] Riverbed Technology.2013.*"Network Planning & Simulation (OPNET)"*,`http://www.riverbed.com/products-solutions/products/network-performance-management/network-planning-simulation/`.Accessed on the Internet on 6 October 2013.Reliability Hotwire Magazine.Issue 79.

[71] .2013.*"OMNeT++"*,`http://www.omnetpp.org/`.Accessed on the Internet on 6 October 2013.Reliability Hotwire Magazine.Issue 79.

[72] J. Chen.J. Zhang.W. Xu.L. Shu.Y. Sun.2008.*"The Development of a Realistic Simulation Framework with OMNeT++"*.Future Generation Communication and Networking, 2008. FGCN '08. Second International Conference on .Vol 1.pp. 497-500.

[73] K. Yang.J. Ma.2008.*"Implementation of IEEE802.1x in OPNET"*.System Simulation and Scientific Computing, 2008. ICSC 2008. Asia Simulation Conference - 7th International Conference on.Print ISBN: 978-1-4244-1786-5 .pp. 1390-1394.

[74] W. Wang.J. Wu.*"Deterministic Design, Reliability-Based Design, and Robust Design"*,`http://artikelpdf.co.cc/link/deterministic-design-reliability-based-design-and-robust-design/`.Accessed on the Internet on 23 April 2011.

[75] IEEE Std 610.12-1990.1990.*"IEEE Standard Glossary of Software Engineering Terminology"*,`http://standards.ieee.org/findstds/standard/610.12-1990.html`.

[76] D.T. O'Connor.D. Newton.R. Bromley.2002.*"Practical Reliability Engineering"*.John Wiley & Sons Ltd..4th edition.pp. 316-332.ISBN 970-0-470-84462-5.

[77] A Elsayed.1996.*"Reliability Engineering"*.Addison Wesley Longman.Volume 1.pp. 3-19, 175- 185.ISBN 0201634813.

[78] J. Menga.2007.*"Availability and the Different Ways to Calculate It"*,`http://www.weibull.com/hotwire/issue79/relbasics79.htm`.Accessed on the Internet on 6 October 2013.Reliability Hotwire Magazine.Issue 79.

[79] J.H. Sarker.H.T. Mouftah.2008.*"Service reliability with enhanced failure recovery rate for multiple failures in survivable optical networks"*.Communications, 24th Biennial Symposium on.pp. 89–92.

[80] J.S. Pattavina.2004.*"Tutorial on Analyzing High Reliability: Part 1"*,`http://www.eetimes.com/document.asp?doc_id=1277936`.EE Times magazine.Accessed on the Internet on 10 October 2013.

[81] D.W. Coit.June 1997.*"Economic Allocation of Test Times for Subsystem-Level Reliability Growth Testing"*.IIE Transactions .(1998), Vol No 30.pp. 1143-1151.

[82] H Bidgoli.2006.*"Handbook of Information Security"*.John Wiley & Sons Ltd..Volume 3.pp. 77-80.ISBN 0471648337.

[83] Internet Engineering Task Force.1997.*"RFC 2131: Dynamic Host Configuration Protocol "*,`http://tools.ietf.org/html/rfc2131`.

[84] Internet Engineering Task Force.1981.*"RFC 1035: Domain Names - Implementation and Specification"*,`http://tools.ietf.org/html/rfc1035`.

[85] Microsoft MSDN Library.2010.*"Microsoft SMB protocol and CIFS Protocol Overview"*,`http://msdn.microsoft.com/en-us/library/aa365233%28VS.85%29.aspx`.Accessed on the Internet on 26 September 2011.

[86] Microsoft Technet.2006.*"NetBIOS over TCP/IP"*,`http://technet.microsoft.com/en-us/library/bb727013.aspx`.Accessed on the Internet on 26 September 2011.

[87] Internet Engineering Task Force.1998.*"RFC 2475: An Architecture for Differentiated Services"*,`http://tools.ietf.org/html/rfc2475`.

[88] Internet Engineering Task Force.2002.*"RFC 3376: Internet Group Management Protocol, Version 3"*,`http://tools.ietf.org/html/rfc3376`.

[89] Internet Engineering Task Force.2004.*"RFC 3768: Virtual Router Redundancy Protocol (VRRP)"*,`http://tools.ietf.org/html/rfc3768`.

[90] O. Balci.1995.*"Principles and Techniques of Simulation Validation, Verification and Testing"*.Proceeds of the 1995 Winter Simulation Conference.pp. 147-154.

[91] Sun Microsystems Developer Books..*"Distributed Application Architecture"*,`http://java.sun.com/developer/Books/jdbc/ch07.pdf`.Accessed on the Internet on 20 September 2011.

[92] Calomel.org: Open Source Research and Reference..*"Network Tuning and Performance Guide (OpenBSD)"* `https://calomel.org/network_performance.html`.Accessed on the Internet on 20 September 2011.

[93] IEEE 802.11 Wireless Local Area Networks (LANs).2007.*"Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications"*,`http://standards.ieee.org/about/get/802/802.11.html`.

[94] C.E. Spurgeon.2000.*"Ethernet: The Definitive Guide "*.O'Reilly & Associates .1st edition.pp. 39-75, 101-124.ISBN 1565926609.

[95] M. Gast.2005.*"802.11 Wireless Networks: The Definitive Guide"*.O'Reilly Media.2nd edition.pp. 12-66, 101-124.ISBN 0596100523.

[96] D. Corner.2006.*"Internetworking with TCP/IP: Principles, protocols, and architecture"*.Prentice Hall.5th edition.pp. 41-56, 175-234, 235-271, 373-402, 501-519, 543-560, 561-576.ISBN 0131876716.

[97] R.J.. Hontanon.2001.*"Linux Security"*.John Wiley and Sons.1st edition.pp. 245-280.ISBN 078212741X.

[98] Y.D. Black.2000.*"IP routing protocols: RIP, OSPF, BGP, PNNI, and Cisco routing protocols"*.Prentice Hall Professional.1st edition.pp. 41-65, 104-120, 121-156, 157-183, 184.ISBN 0130142484.

[99] F. Shamim.2002.*"Troubleshooting IP Routing Protocols"*.Cisco Press.1st edition.pp. 3-24.ISBN 1587050196.

[100] G. Tomsho.2011.*"Guide to Networking Essentials"*.Cengage Learning.6th edition.pp. 62-69, 124-128, 289-331, 481.ISBN 1111312524.

[101] *"IEEE Standard for Local and metropolitan area networks:Virtual Bridged Local Area Networks"*,`http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf`.

[102] Internet Engineering Task Force.1998.*"RFC 2460: Internet Protocol, Version 6 (IPv6) Specification"*,`http://tools.ietf.org/html/rfc2460`.

[103] Internet Engineering Task Force.1996.*"Guidelines for creation, selection, and registration of an Autonomous System (AS)"*,`http://tools.ietf.org/html/rfc1930`.

[104] Internet Engineering Task Force.1998.*"RIP Version 2"*,`http://tools.ietf.org/html/rfc2453`.

[105] Internet Engineering Task Force.1998.*"OSPF Version 2"*,`http://tools.ietf.org/html/rfc2328`.

[106] Internet Engineering Task Force.2006.*"A Border Gateway Protocol 4 (BGP-4)"*,`http://tools.ietf.org/html/rfc4271`.

[107] Microsoft Technet.2003.*"How QoS Works"*,`http://technet.microsoft.com/en-us/library/cc728211%28WS.10%29.aspx#w2k3tr_qos_how_bdgn`.Accessed on the Internet on 26 September 2011.

[108] Internet Engineering Task Force.1982.*"RFC 826: An Ethernet Address Resolution Protocol"*,`http://tools.ietf.org/html/rfc826`.

[109] C. Liu.P. Albitz.2006.*"DNS and BIND"*.O'Reilly Media, Inc.5th edition.pp.1-88.ISBN 0596100574.

[110] D. Barnes.B. Sakandar.2004.*"Cisco LAN switching fundamentals*.Cisco Press.2nd edition.pp. 83-97, 249-280.ISBN 1587050897.

[111] G.R. Brown.2007.*"ClusterMonkey, Measuring network performance"*,`http://www.clustermonkey.net/content/view/186/32/`.Accessed on the Internet on 1 September 2011.

[112] G. Huston.2003.*"Internet protocol Journal, Measuring IP network performance"*,`http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_6-1/ipj_6-1.pdf`.Accessed on the Internet on 1 September 2011.

[113] L.G. Chang.1999.*"Network Simulations with OPNET"*.Proceedings of the 1999 Winter Simulation Conference.Vol 1.pp. 307-314.

[114] V. Gupta.S. Dharmaraja.M. Gong.2010.*"Analytical modelling of TCP Flow in Wireless LANs"*.Mathematical and Computer Modelling.Volume 53, Issues 5-6.pp. 684-693.

[115] A. Misra.T.J. Ott.J.S. Baras.2003.*"Markov Processes with State-Dependent Failure Rates and Application to RED and TCP Window Dynamics"*.11th Mediterranean Conference on Control and Automation.

[116] T. Issariyakul.E. Hossain.A. Sule Alfa.2006.*"End-to-End Batch Transmission in a Multihop and Multirate Wireless Network: Latency,Reliability and Throughput Analysis"*.IEEE Transactions on Mobile Computing .Vol.5, No.9.pp. 1143- 1155.

[117] K. Muppala.G. Ciardo.1994.*"Stochastic Reward Nets for Reliability Prediction"*.Communications in Reliability, Maintainability and Serviceability Conference .Vol.1.pp. 9-20.

[118] J. Hillston.2005.*"A Compositional Approach to Performance Modelling"*.Cambridge University Press.1st edition.pp. 1-42.ISBN 978-0521673532.

[119] K.S. Trivedi.R. Sahner.A. Puliafito.*"Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the Sharpe Software Package"*.Springer.Illustrated.pp. 261-270.ISBN 9780792396505.

[120] M. Rausand.A. Hoyland.2004.*"System Reliability Theory; Models, Statistical methods, and Applications"*.John Wuiley & Sons.2nd edition.pp. 15-30, 301-110.ISBN 0-471-47133-X.

[121] N.B. Fuqua.2003.*"START Volume 10, Number2, The Applicability of Markov Analysis Methods to Reliability, Maintainability, and Safety"*,`http://src.alionscience.com/pdf/MARKOV.pdf`.Accessed on the Internet on 5 September 2011.

[122] K.S. Trivedi.1982.*"Probability And Statistics With Reliability Queuing And Computer Science Applications"*.PHI Learning Pvt. Ltd..1th edition.Chapter 8 on Continuous-Time Markov Chains.ISBN 8120305086, 9788120305083.

[123] J.S. Pattavina.2004.*"Tutorial on Analyzing High Reliability: Part 2"*,`http://www.eetimes.com/document.asp?doc_id=1277937`.EE Times magazine.Accessed on the Internet on 10 October 2013.

[124] P. O'Connor..*"Reliability, Testing and Related Regulations And Standards"*,`http://www.pat-oconnor.co.uk/reltesteststds.htm`.Accessed on the Internet on 7 September 2011.

[125] A. Peronne.P. Dersin.2011.*"Comparison of High-Availability Automation Networks"*.2011 Proceedings of Reliability and Maintainability Symposium (RAMS).pp. 1-6.

[126] M. Lanus.2003.*"Hierarchical composition and aggregation of state-based availability and performability models"*.IEEE Transactions on Reliability, vol. 52, no. 1.pp. 44-52.

[127] M. Modarres.1984.*"A method of predicting availability characteristics of series-parallel systems"*.IEEE Transactions on Reliability, no. 4 .pp. 309-312.

[128] H. Wang.A. Gerber.A. Greenberg.J. Wang.J.R. Yang.2007.*"Towards Quantification of IP Network Reliability"*.Proceedings of ACM SIGCOMM, Kyoto.pp. 13-24.

[129] Internet Engineering Task Force.2009.*"RFC 5474: A Framework for Packet Selection and Reporting"*,`http://tools.ietf.org/html/rfc5474`.

[130] G. Markopoulou.G. Iannaccone.C. Bhattacharyya.C. Chuah.2004.*"Characterization of Failures in an IP Backbone Network"*.Proceedings of IEEE INFOCOM '04, Hong Kong.

[131] E. De Souza e Silva.1997.*"TANGRAM-II environment for computer and communication system modelling"*,`http://www.land.ufrj.br/tools/tools.html`.Accessed on the Internet on 25 December 2011.

[132] G. Casale.R.R. Muntz.G. Serazzi.March 2009.*"Special issue on tools for computer performance modelling and reliability analysis"*.ACM SIGMETRICS Performance Evaluation Review Archive.Vol 36, No 4.pp. 2-3.

[133] R.W.R Darling.J.R. Norris.April 2008.*"Differential equation approximations for Markov chains"*.Probability Surveys.Vol 5.pp. 37-79.

[134] R.Y. Rubinstein.D.P. Kroese.2008.*"Simulation and the Monte Carlo Method"*.Wiley.2nd edition..ISBN 978-0-470-17794-5.

[135] R.M.L.R. Carmo.L.R.D Carvalho.E.D. Souza.M.C. Diniz.1998.*"Performance/availability modelling with the TANGRAM-II modelling environment"*.Performance Evaluation.Vol 33, No 1.pp. 45-65.

[136] E. D. Souza.R.M.M Leao.R.R. Muntz.A.P.C. DaSilva.2006.*"Modelling, analysis, measurement and experimentation with the Tangram-II integrated environment"*. In Proceeding of International of Conference on Performance Evaluation Methodologies and Tools.

[137] E. D. Souza.R.M.M Leao.R.R. Muntz.W. Cheng.2010.*"Tangram II Manual "*,`http://www.land.ufrj.br/tools/tools.html`.Accessed on the Internet on 25 December 2011.

[138] W.K. Grassmann.2000.*"Computational Probability"*.Springer..1th edition.pp. 83 - 88.ISBN 0792386175, 9780792386179.

[139] E. D. Souza.H. Gail.1986.*"Calculating cumulative operational time distributions of repairable computer systems"*.IEEE Transactions on Computers, vol. c, no. 4.pp. 322-332.

[140] N. Huang.R. Li.W. Chen.R. Kang.2009.*"The Layered Index Method for Network Reliability Analysis"*.Reliability, Maintainability and Safety, 2009. ICRMS 2009. 8th International Conference on.pp. 1155 - 1159.

[141] L. Jereb.A. Kiss.2000.*"Performance Index Based Network Reliability Analysis with Stratified Sampling"*.Proceedings of IEEE International, Computer Performance and Dependability Symposium, 2000. IPDS 2000.pp. 123-130.

[142] J. Carlier.Y. Li.J. Lutton.1996.*"Reliability Evaluation of Large Telecommunication Networks"*.Discrete Applied Mathematics.Volume 76, Issues 1-3.pp. 61-80.

[143] A.M. Rushdi.1988.*"Performance Indexes of a Telecommunication Network"*.IEEE Transactions on Reliability.Volume 37, Issues 1.pp. 57-64.

[144] K.M. Khalil.Y.S. Sun.1992.*"The Effect of Bursty Traffic on the Performance of Local Area Networks"*.Global Telecommunications Conference, 1992. Conference Record., GLOBECOM '92. Communication for Global Users., IEEE .Volume 1.pp. 597-603.

[145] B. Robinson.V. Liberatore.2004.*"On the Impact of Bursty Cross-Traffic on Distributed Real-Time Process Control"*. Proceedings of 2004 IEEE International Workshop on Factory Communication Systems, 2004.pp. 147-152.

[146] Cisco Systems.2006.*"How to Configure InterVLAN Routing on Layer 3 Switches"*,`http://www.cisco.com/en/US/tech/tk389/tk815/technologies_configuration_example09186a008019e74e.shtml`.Accessed on the Internet on 15 September 2011.

[147] A.L. Black.J.M. Martin.1988.*"The Bellcore Reliability Prediction Procedure (RPP); 1988 edition"*.Global Telecommunications Conference, 1988, and Exhibition. 'Communications for the Information Age.' Conference Record, GLOBECOM '88., IEEE.Volume 1.pp. 70 - 76.

[148] E. Brosh.S.A. Baset.V. Misra.D. Rubenstein.H. Schulzrinne.2010.*"The Delay-Friendliness of TCP for Real-Time Traffic"*.IEEE/ACM Transactions on Networking.Volume 18, No. 5.pp. 1478 - 1491.

[149] Microsoft Certification Resources.1998.*"The Cisco Three-Layered Hierarchical Model"*,`http://www.mcmcse.com/cisco/guides/hierarchical_model.shtml`.Accessed on the Internet on 15 September 2011.

[150] Moxa Group.2006.*"Moxa White Paper: Redundant Gigabit Backbone Adds Speed and Reliability to Industrial Networks"*,`http://www.moxa.com/support/request_catalog_detail.aspx?id=65`.Accessed on the Internet on 17 September 2011.

[151] ComConsult Technologie.2003.*"Product Analyses: HiPER RING vs. RSTP"*,`http://www.belden.com/docs/upload/HiPER_Ring_vs_RSTP_WP.pdf`.Accessed on the Internet on 17 September 2011.

[152] M. Felser.2008.*"Media Redundancy for PROFINET IO"*.Factory Communication Systems, 2008. WFCS 2008. IEEE International Workshop on.pp. 325 - 330.

[153] International Electrotechnical Commission.2010.*"Industrial communication networks - High availability automation networks"*,*http://webstore.iec.ch/preview/info_iec62439-1%7Bed1.0%7Den.pdf*.

[154] International Electrotechnical Commission.2013.*"Communication networks and systems for power utility automation"*,*http://webstore.iec.ch/preview/info_iec61850-1%7Bed2.0%7Db.pdf*.

[155] R. E. Crosse.J.E. Bowen.H.T. Combs.D.G. Dunn.M.A. Hildreth.A. Pilcher.2005.*"Smart Industrial Substations "*.Industry Applications Magazine, IEEE .Vol.11, Issue 2.pp. 12 - 20.

[156] William Chia-Wei Cheng.1990.*"Interactive 2-D drawing tool under X11 for Unix, available on Linux and most Unix and Unix-like platforms"*,*http://bourbon.usc.edu/tgif/index.html*.Accessed on the Internet on 6 July 2012.

[157] Thomas Williams, Colin Kelley, Russell Lang, Dave Kotz, John Campbell, Gershon Elber, Alexander Woo and many others.1986.*"Portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms"*,*http://www.gnuplot.info/*.Accessed on the Internet on 6 July 2012.

[158] R. Durrett.2010.*"Probability: Theory and Examples"*.Cambridge University Press.4th edition.pp. 41.ISBN 9780521765398.

[159] R. Runyon.1980.*"Fundamentals of Behavioural Statistics"*.Addison-Wesley Publishing Company.4th edition.pp. 220-221.ISBN 0-2-1-06624-6.

[160] Cross Validate.2010.*"Is there a reference that suggest using 30 as a large enough sample size?"*,*http://stats.stackexchange.com/questions/2541/is-there-a-reference-that-suggest-using-30-as-a-large-enough-sample-size*.Accessed on the Internet on 19 October 2013.

[161] M. Haahr.1998.*"Randomness and Integrity Services Ltd."*,*http://www.random.org/*.Accessed on the Internet on 24 August 2013.

[162] F.G. Stremler.March 2010.*"Introduction to communication systems"*.Addison-Wesley Longman.Illustrated edition.pp. 115.ISBN 9780201072440.

[163] J.R. Phillips.*"Online Curve Fitting and Surface Fitting Web Site"*,*http://zunzun.com/*.Accessed on the Internet on 8 September 2013.

[164] M Ledvij.*"The Industrial Physicist - Curve fitting made easy"*,*http://www.originlab.com/pdfs/curve_fitting_made_easy.pdf*.Accessed on the Internet on 9 November 2013.

Appendix A. Modelling Data

A.1. Model A

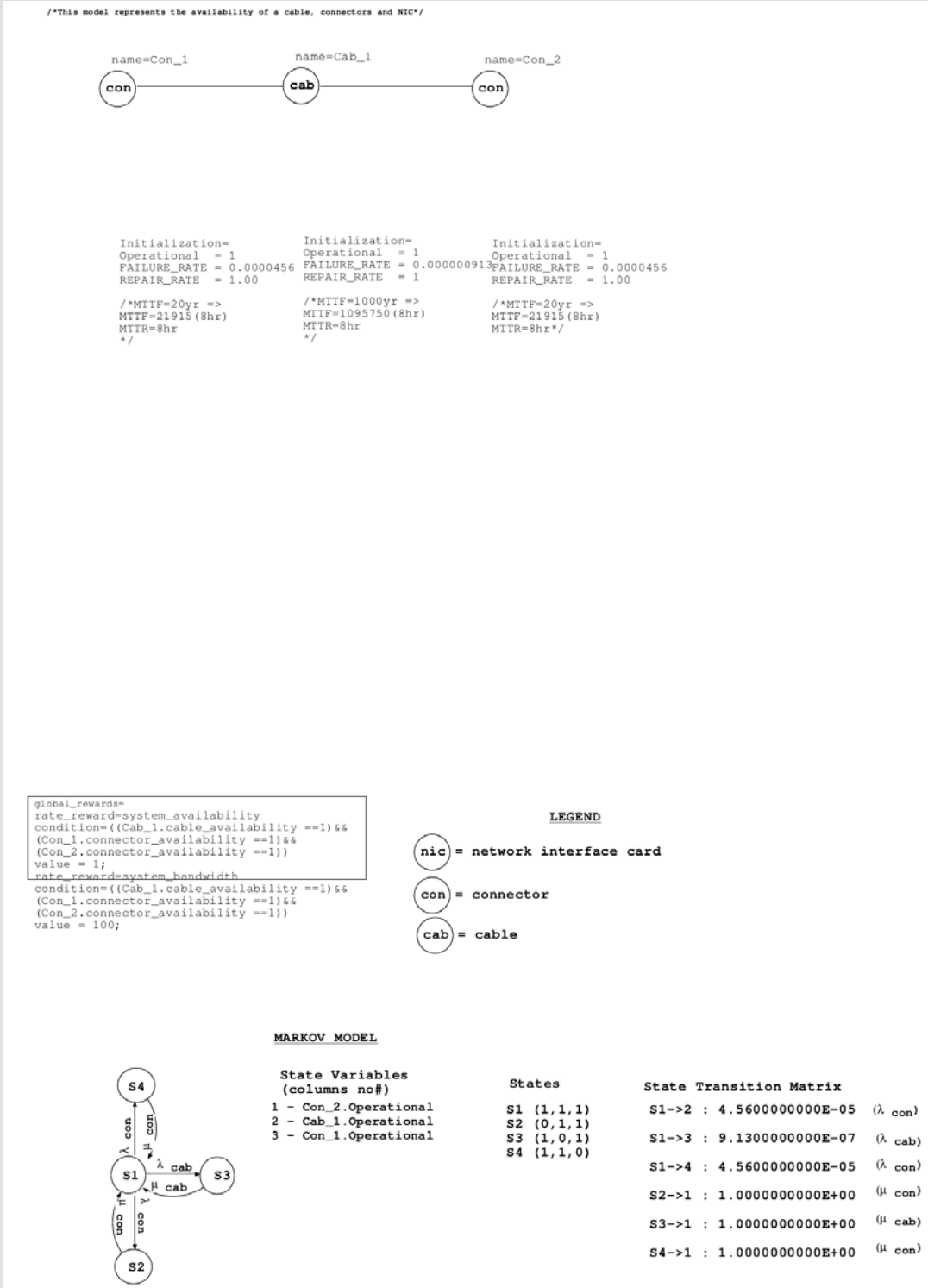

Figure A.1. Analytical Model A: Simple link model

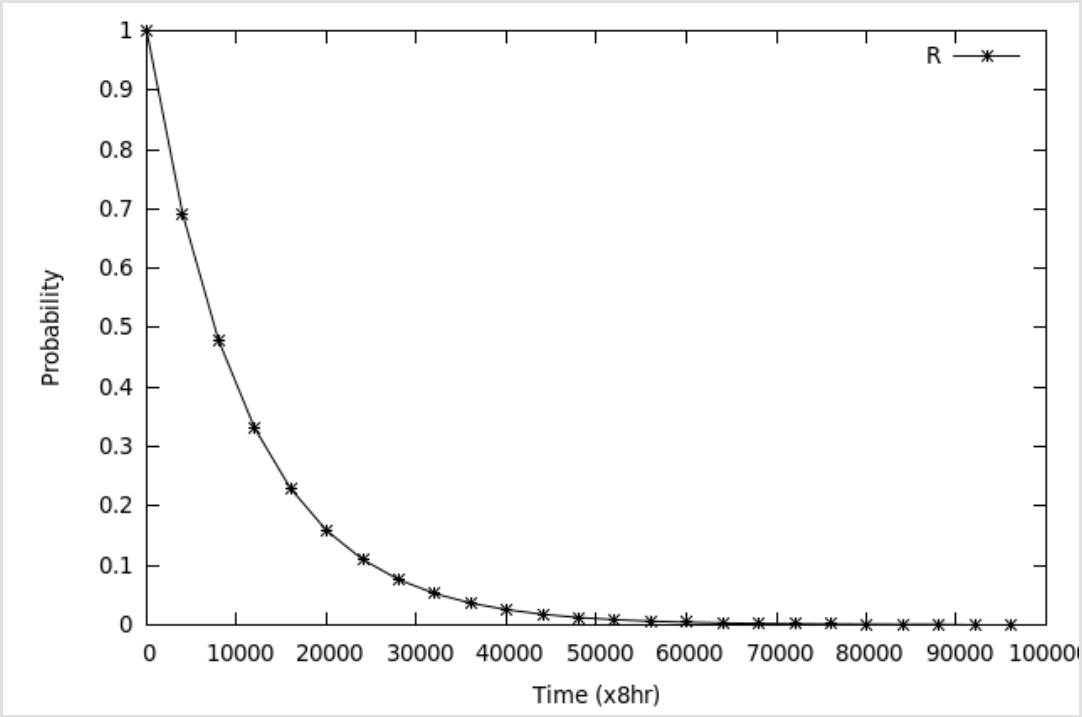

Figure A.2. R(t) Model A: Simple link model

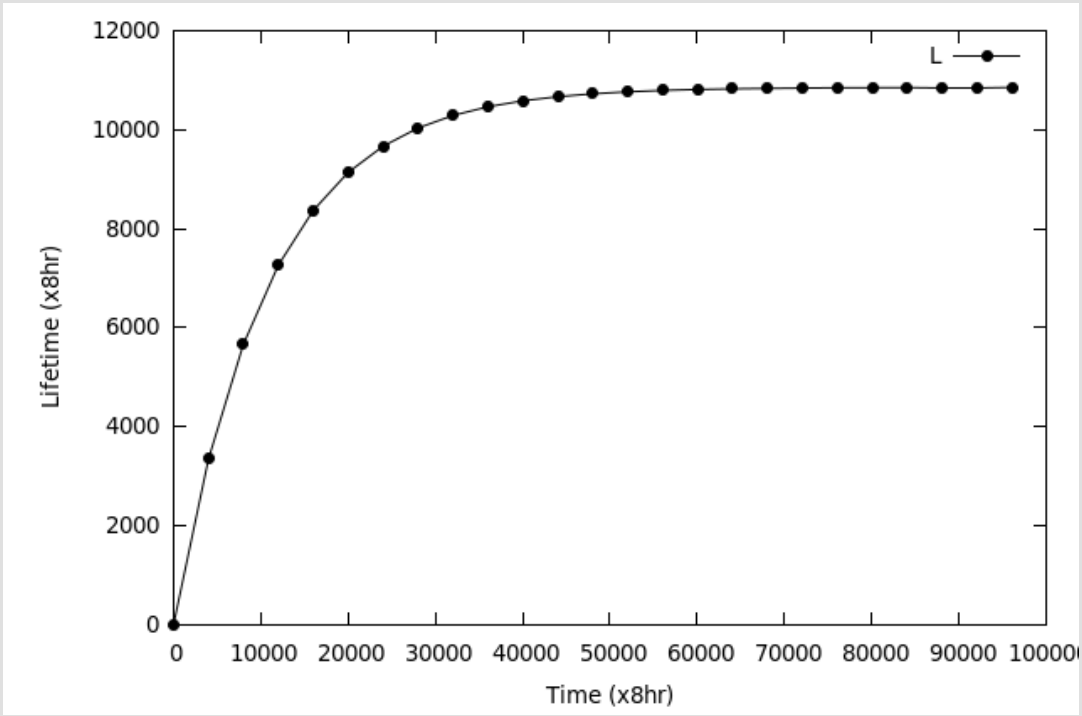

Figure A.3. L(t) Model A: Simple link model

A.2. Model B

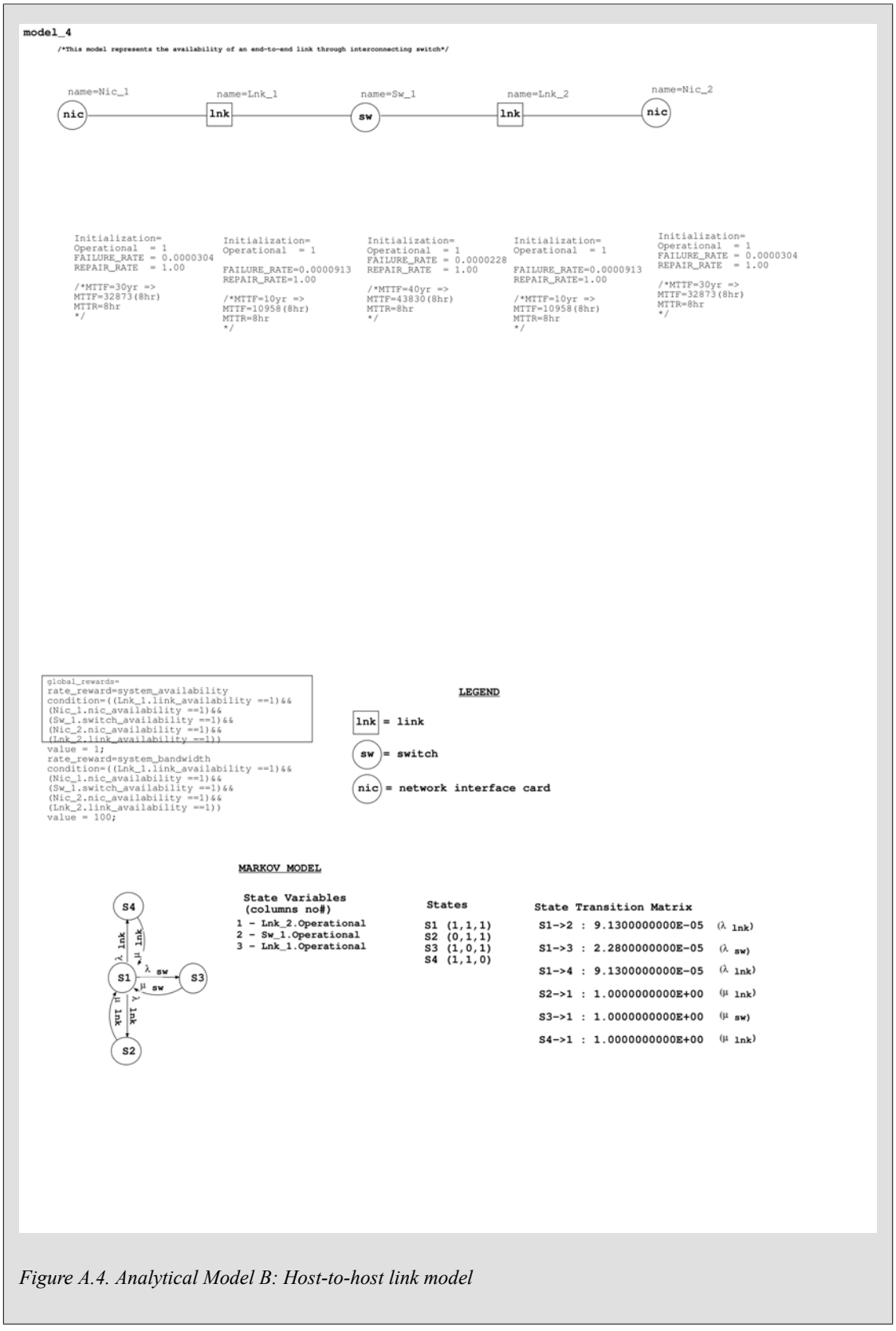

Figure A.4. Analytical Model B: Host-to-host link model

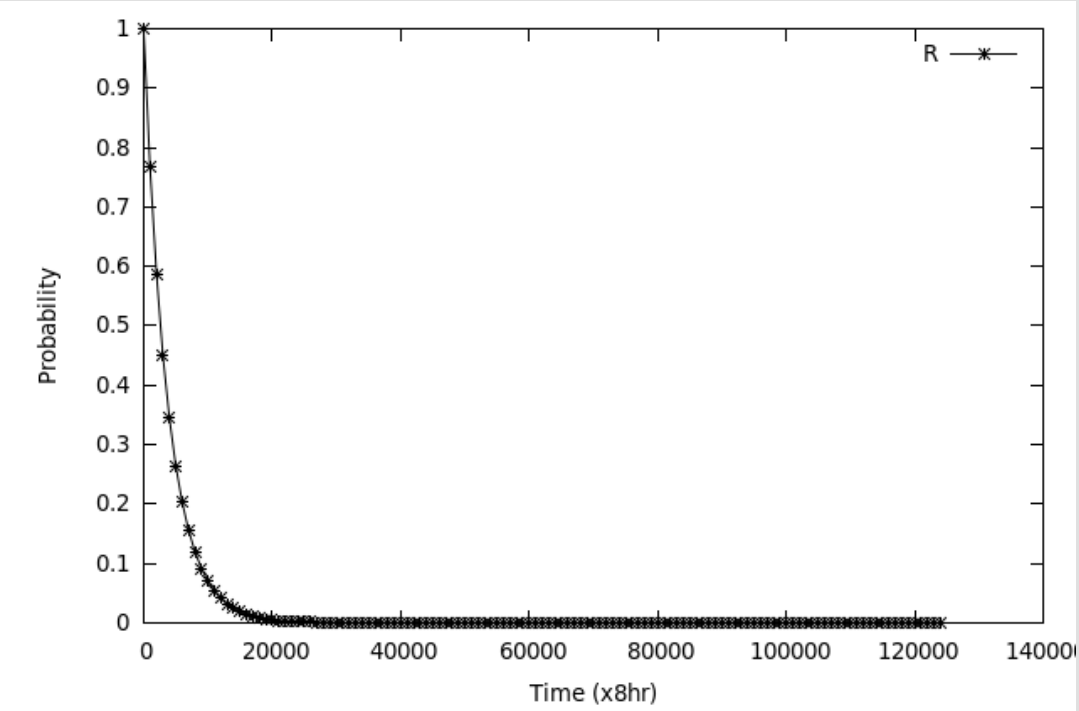

Figure A.5. R(t) Model B: Host-to-host link model

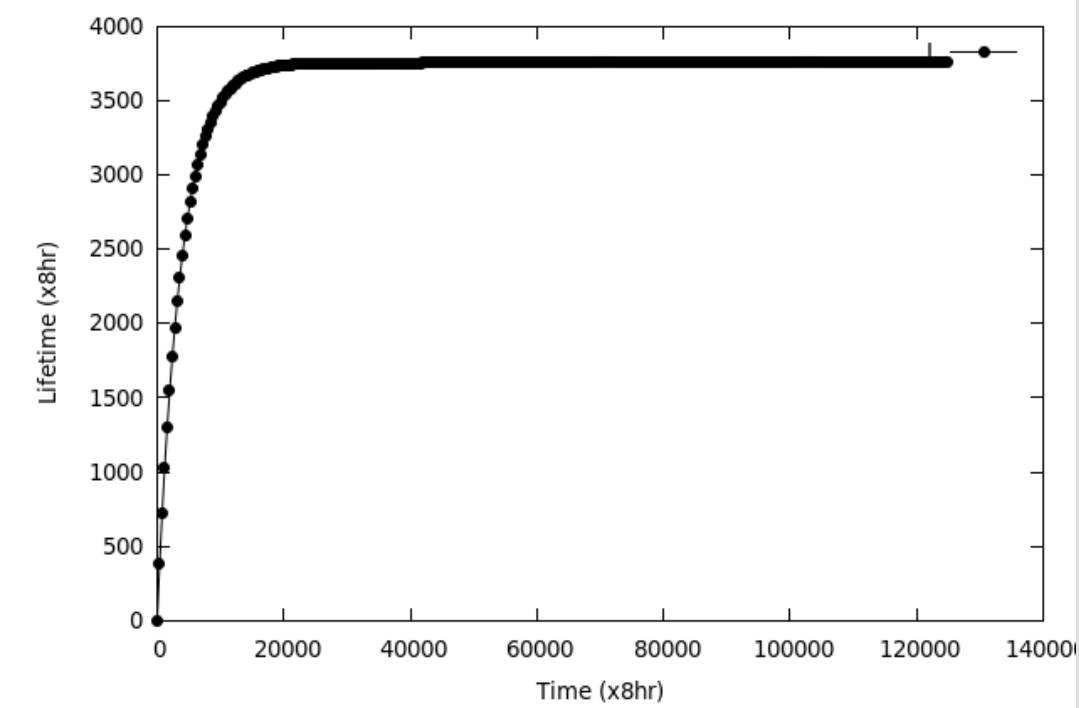

Figure A.6. L(t) Model B: Host-to-host link model

A.3. Model C

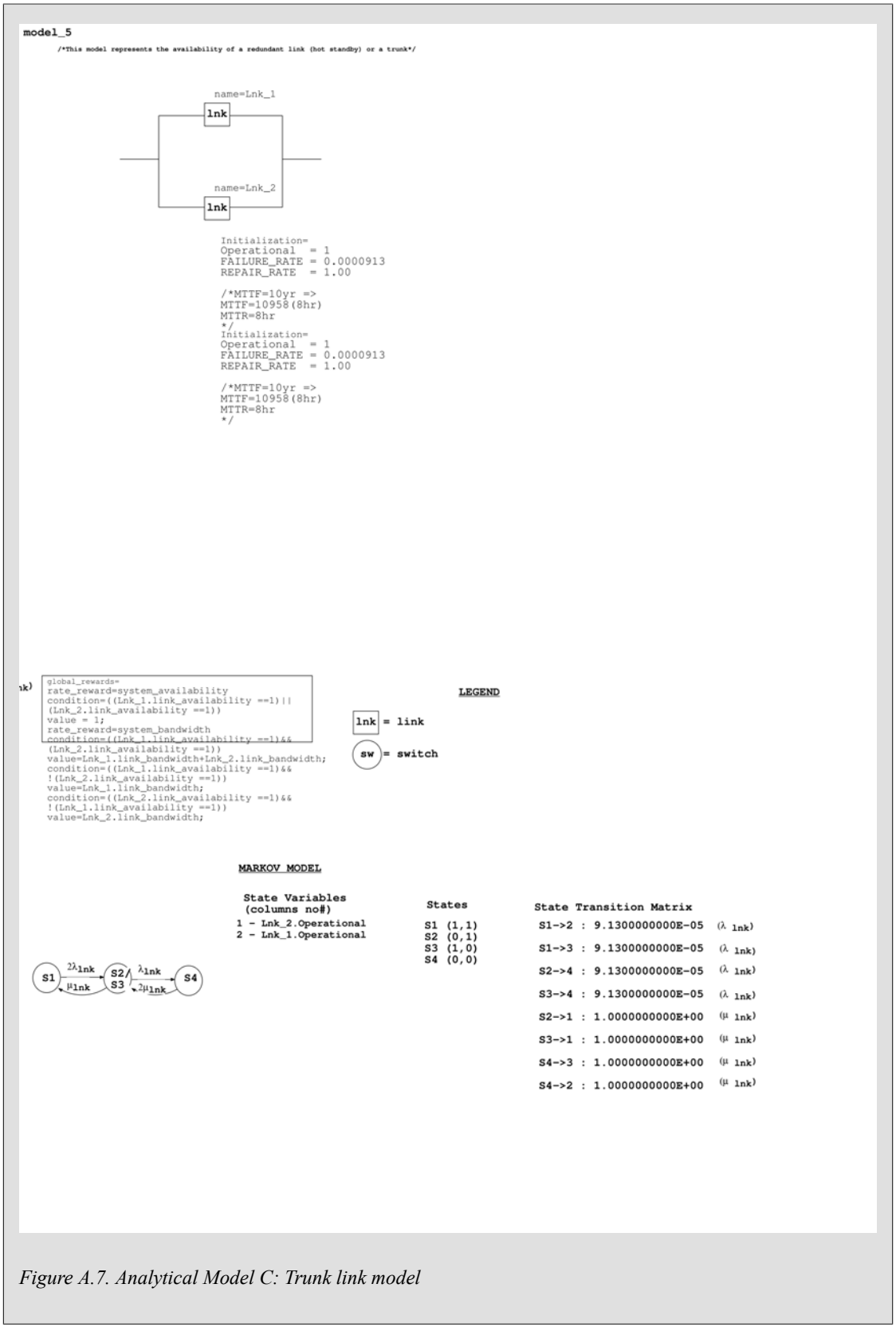

Figure A.7. Analytical Model C: Trunk link model

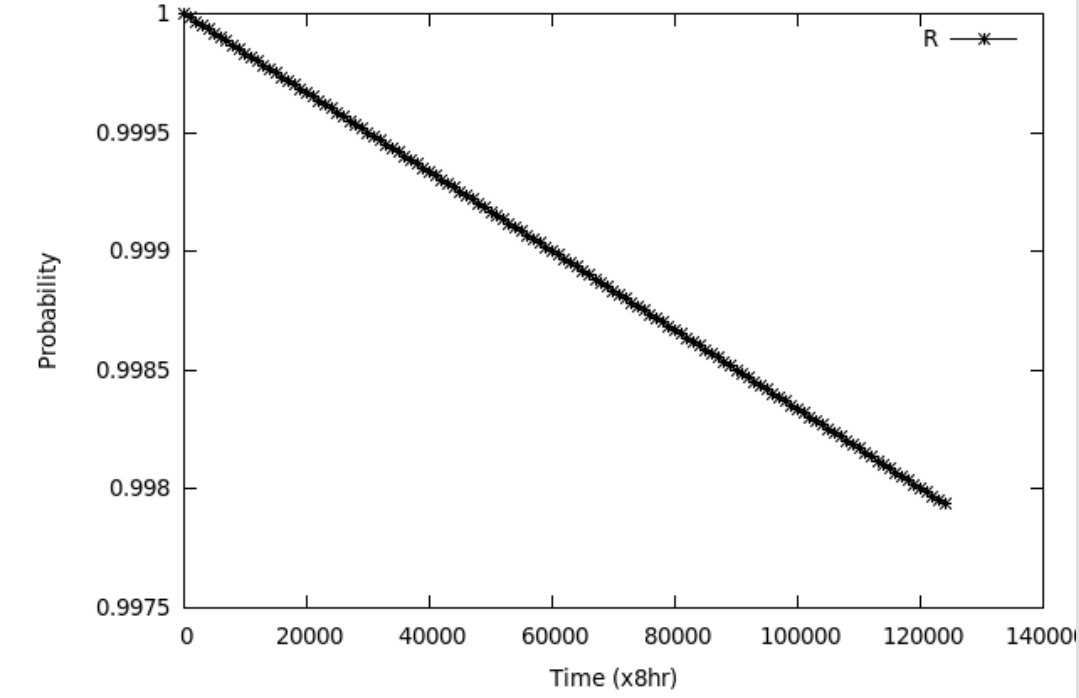

Figure A.8. R(t) Model C: Trunk link model

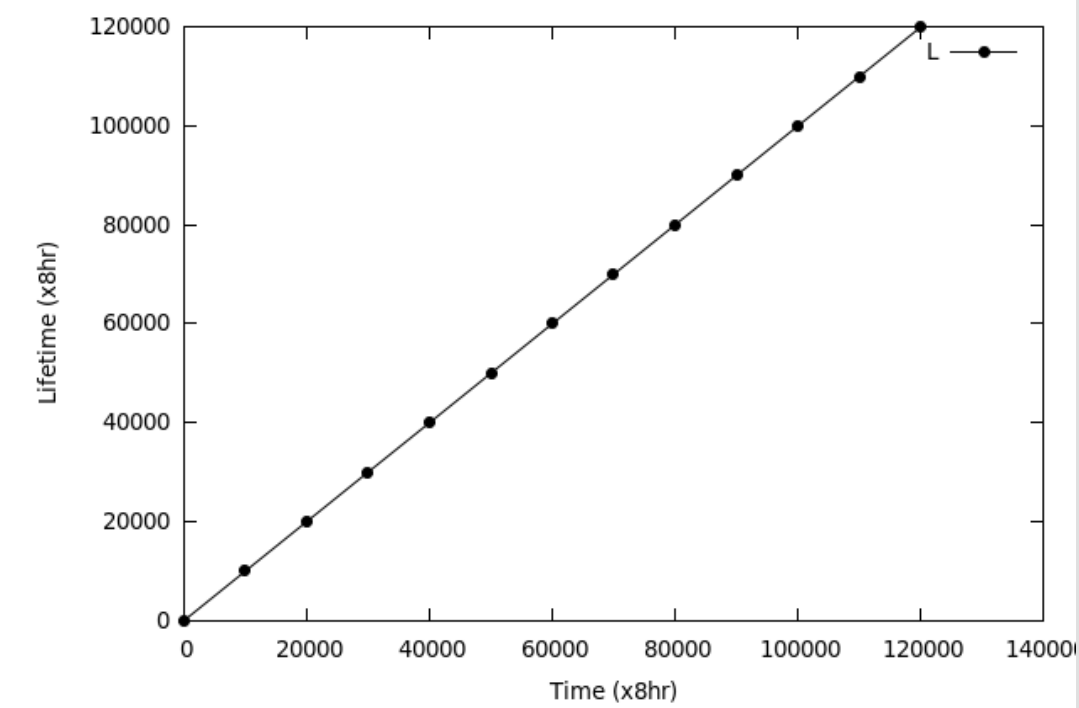

Figure A.9. L(t) Model C: Trunk link model

A.4. Model D

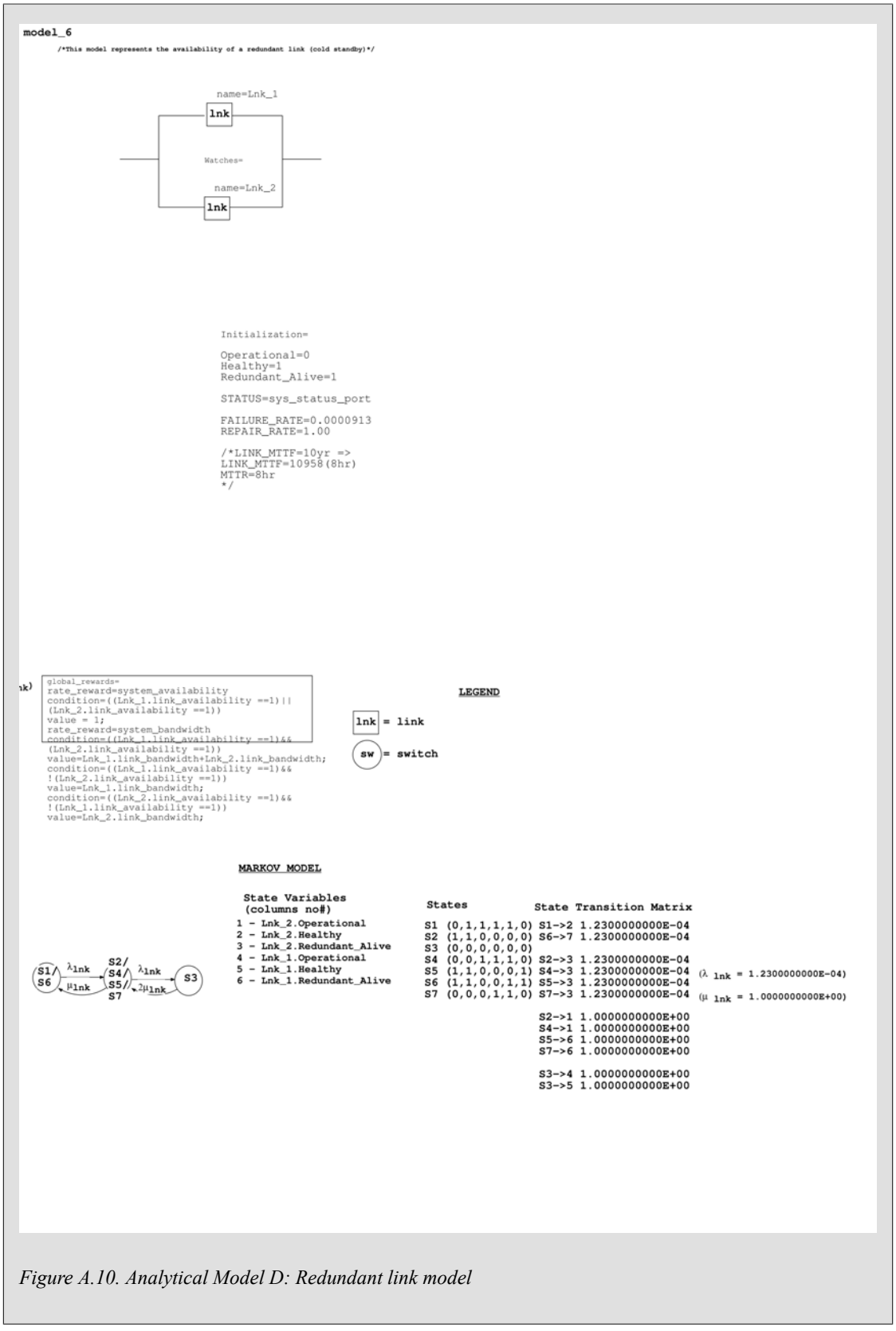

Figure A.10. Analytical Model D: Redundant link model

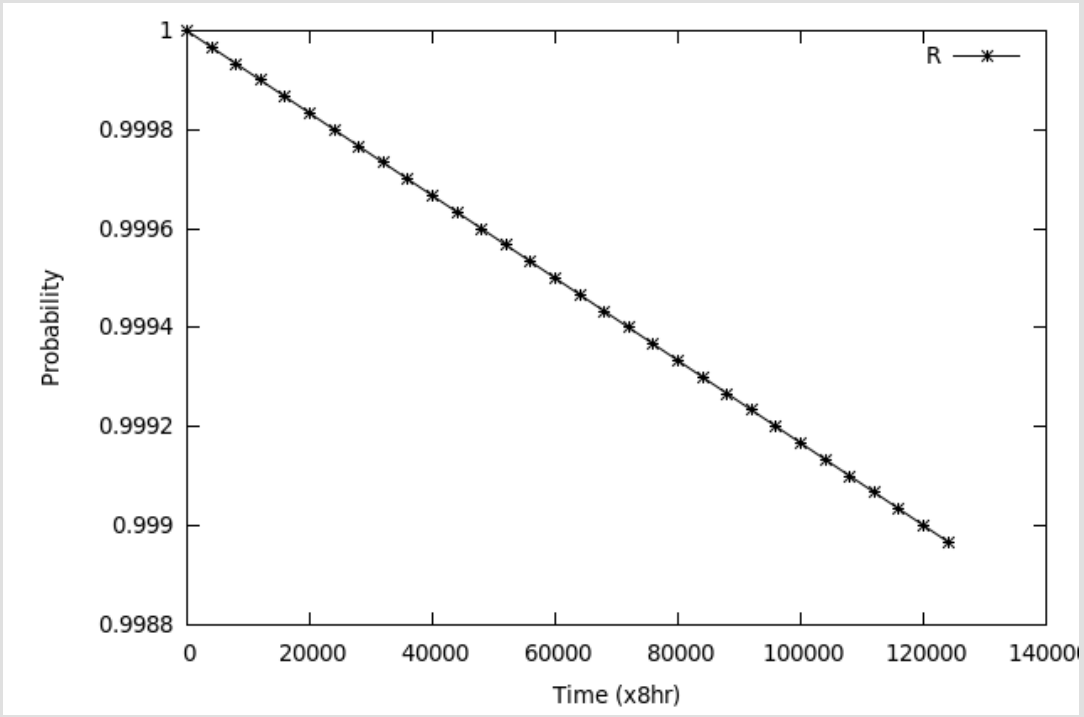

Figure A.11. R(t) Model D: Redundant link model

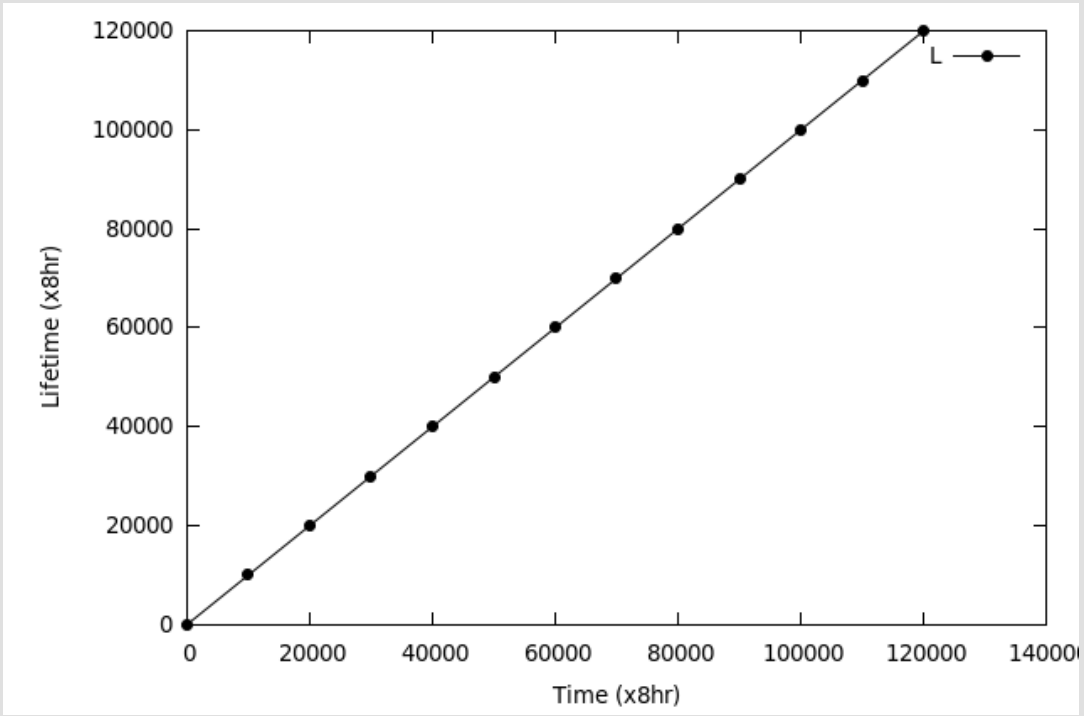

Figure A.12. L(t) Model D: Redundant link model

A.5. Model E1

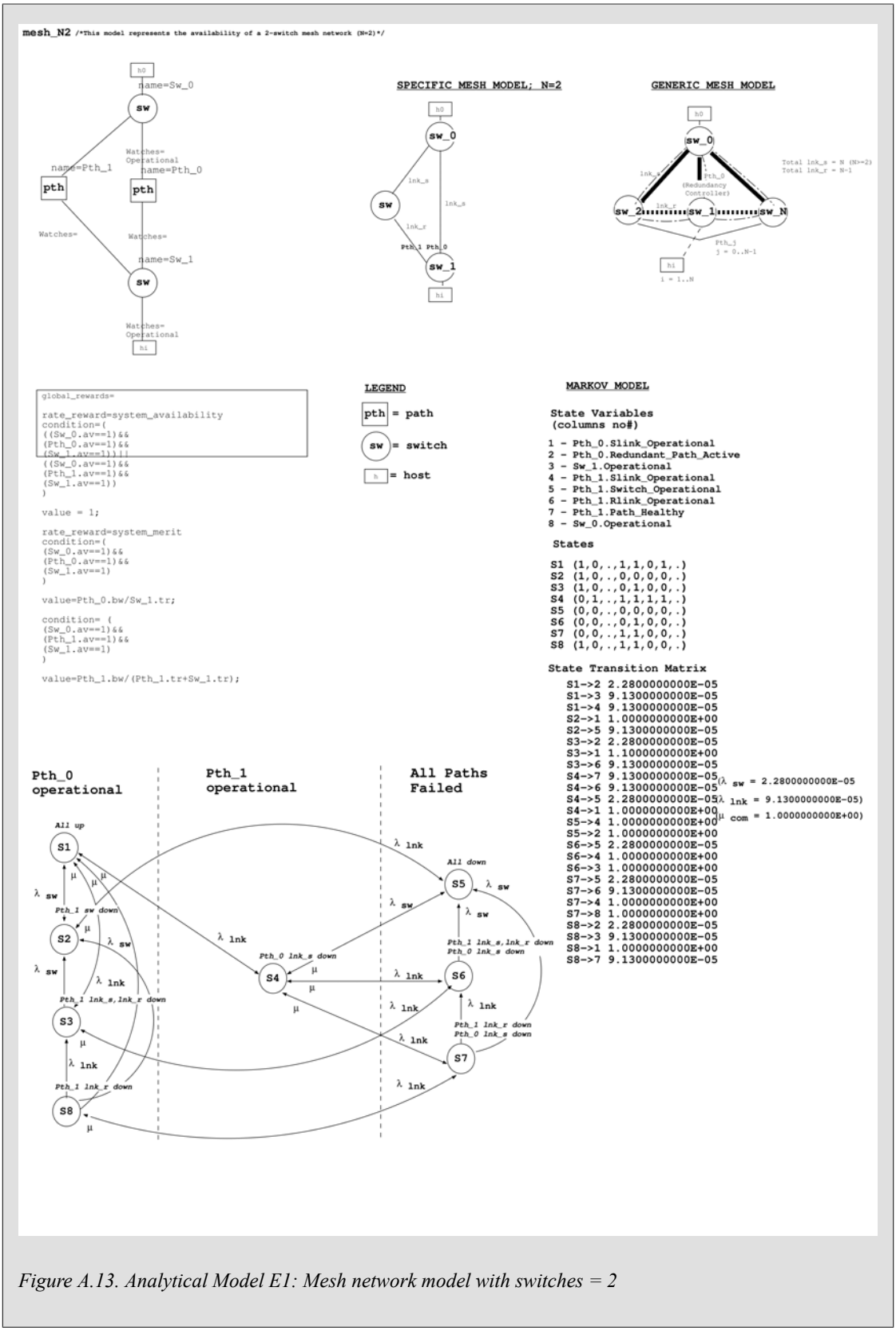

Figure A.13. Analytical Model E1: Mesh network model with switches = 2

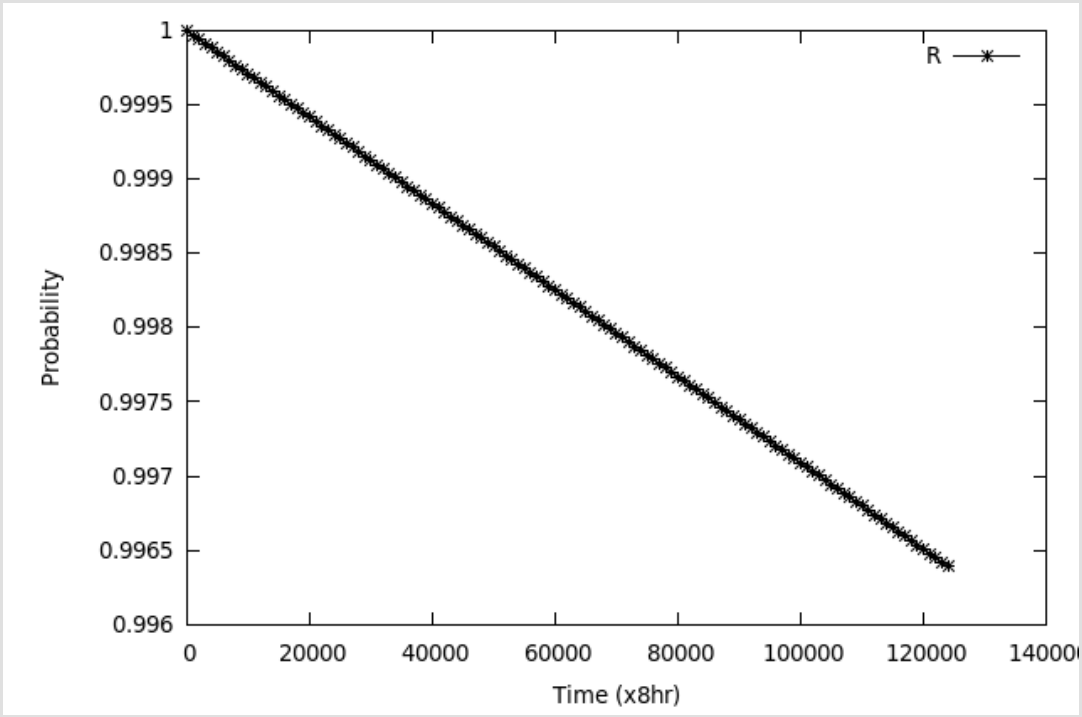

Figure A.14. R(t) Model E1: Mesh network model with switches = 2

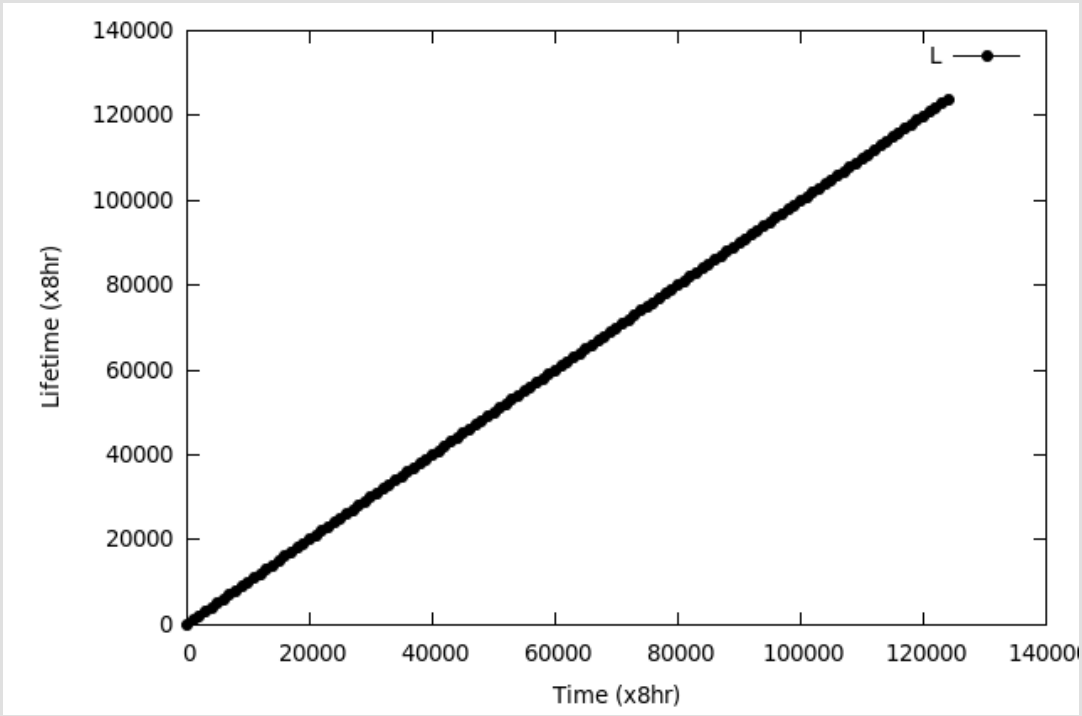

Figure A.15. L(t) Model E1: Mesh network model with switches = 2

A.6. Model E2

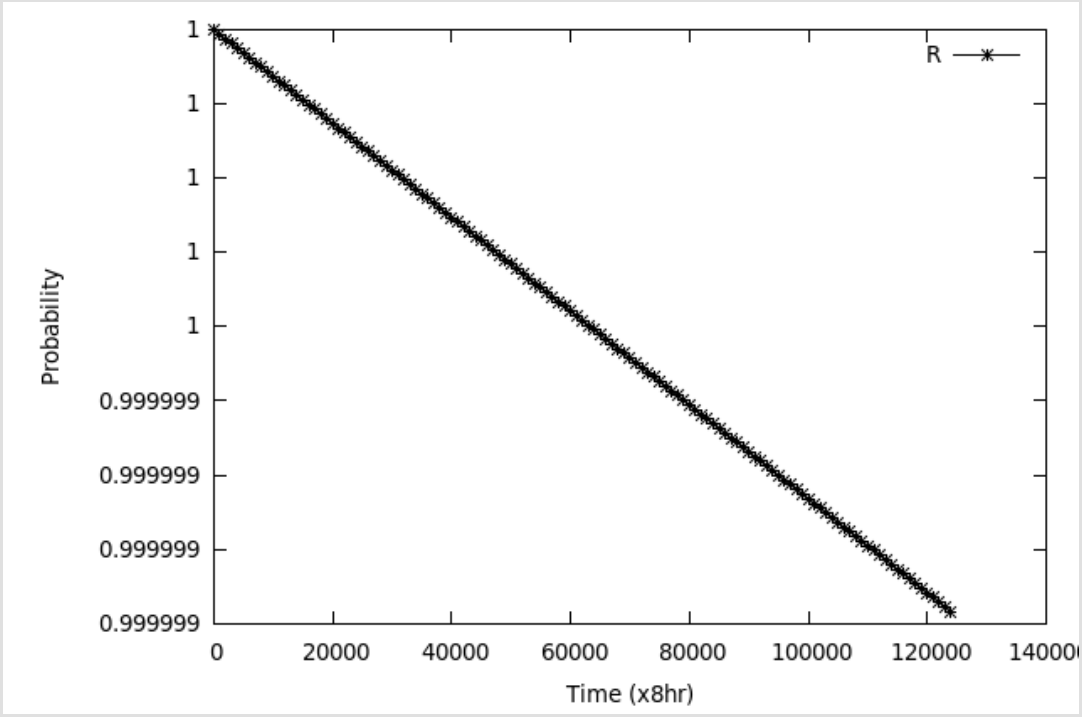

Figure A.16. R(t) Model E2: Mesh network model with switches = 3

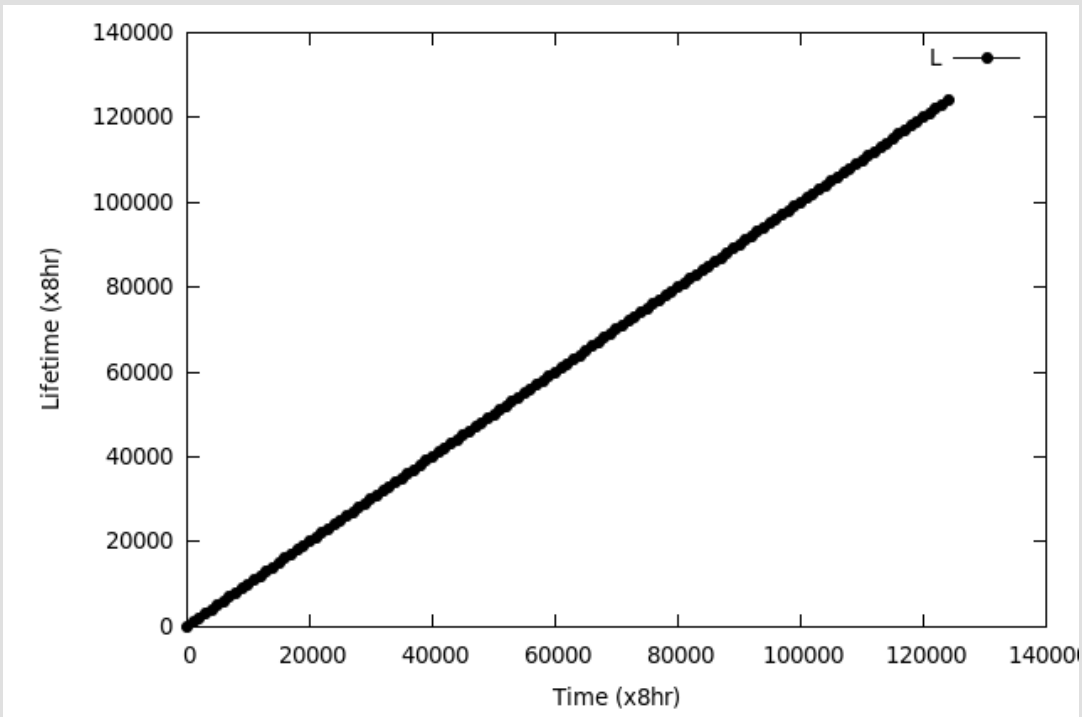

Figure A.17. L(t) Model E2: Mesh network model with switches = 3

A.7. Model F1

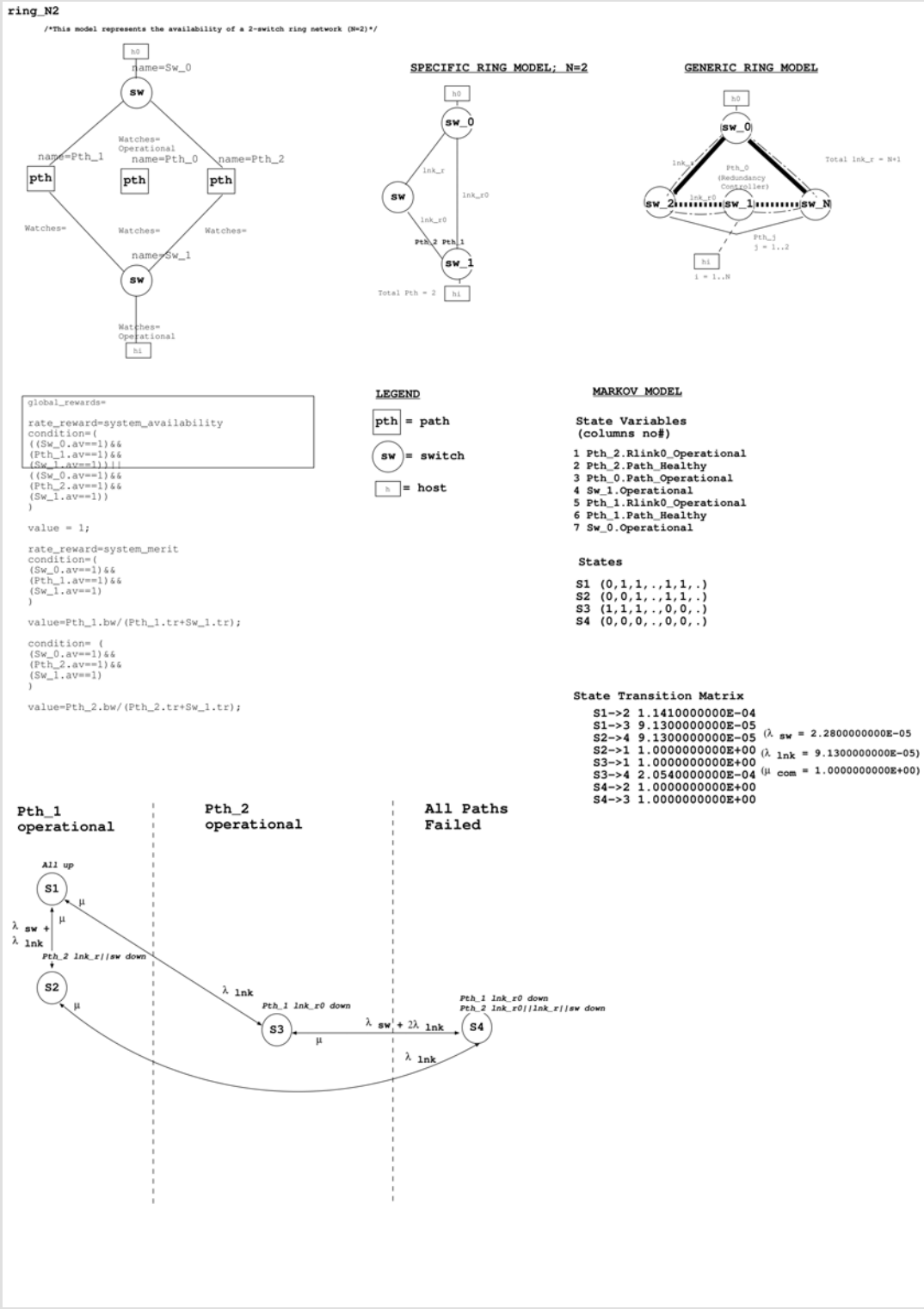

Figure A.18. Analytical Model F1: Ring network model with switches = 2

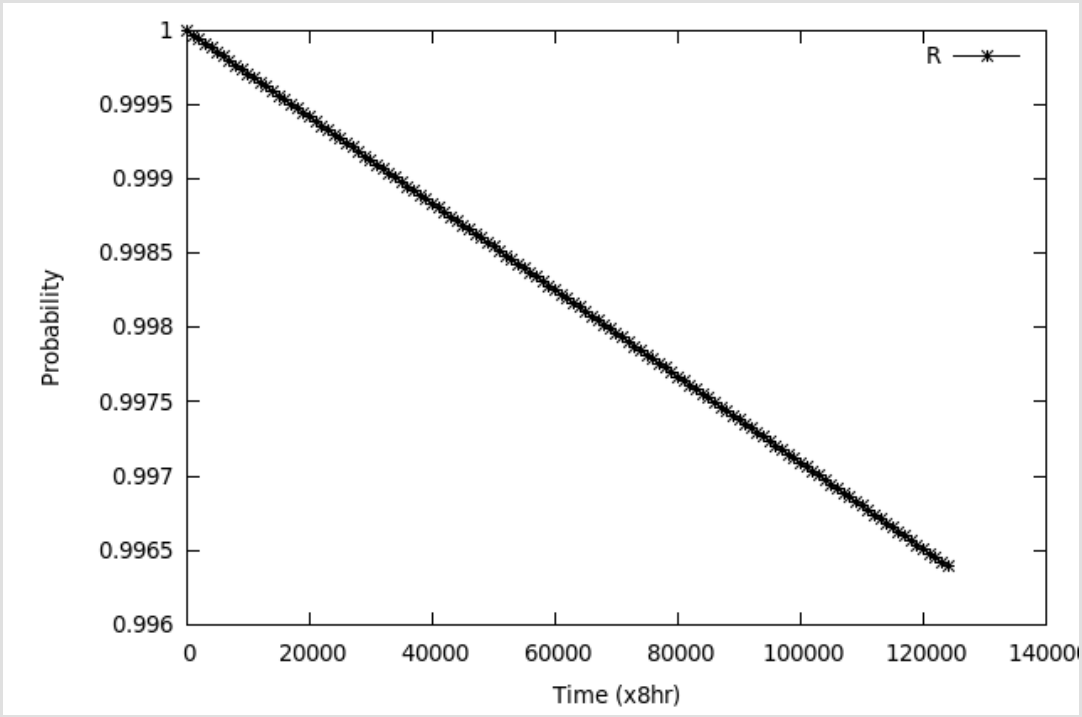

Figure A.19. R(t) Model F1: Ring network model with switches = 2

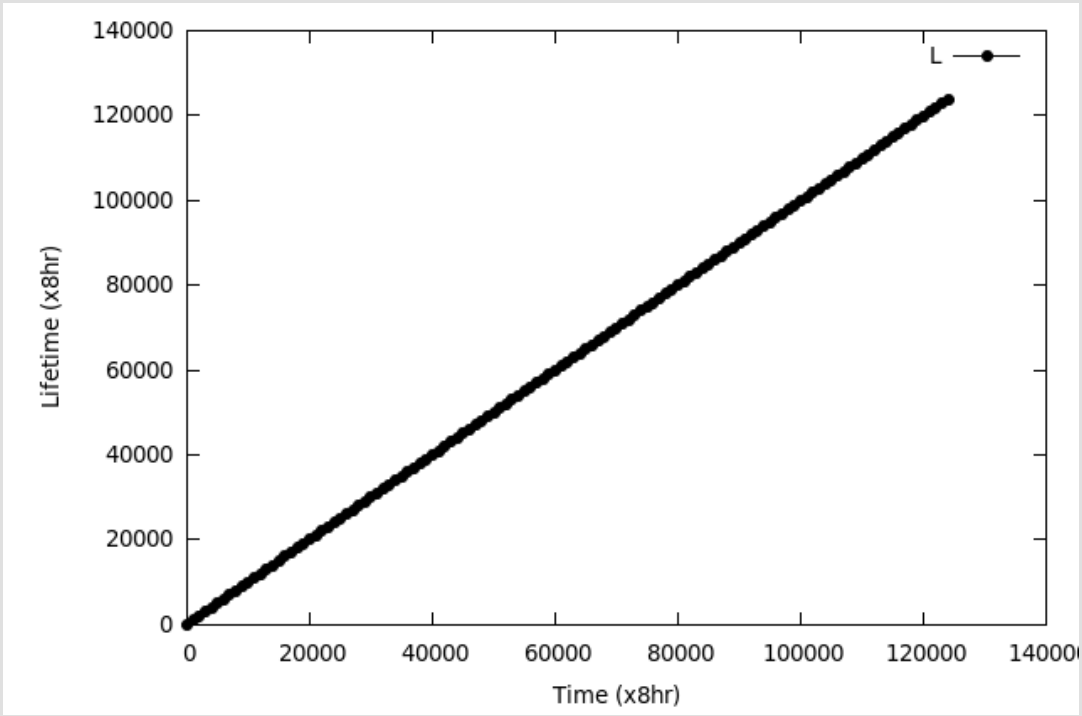

Figure A.20. L(t) Model F1: Ring network model with switches = 2

A.8. Model F2 - F19

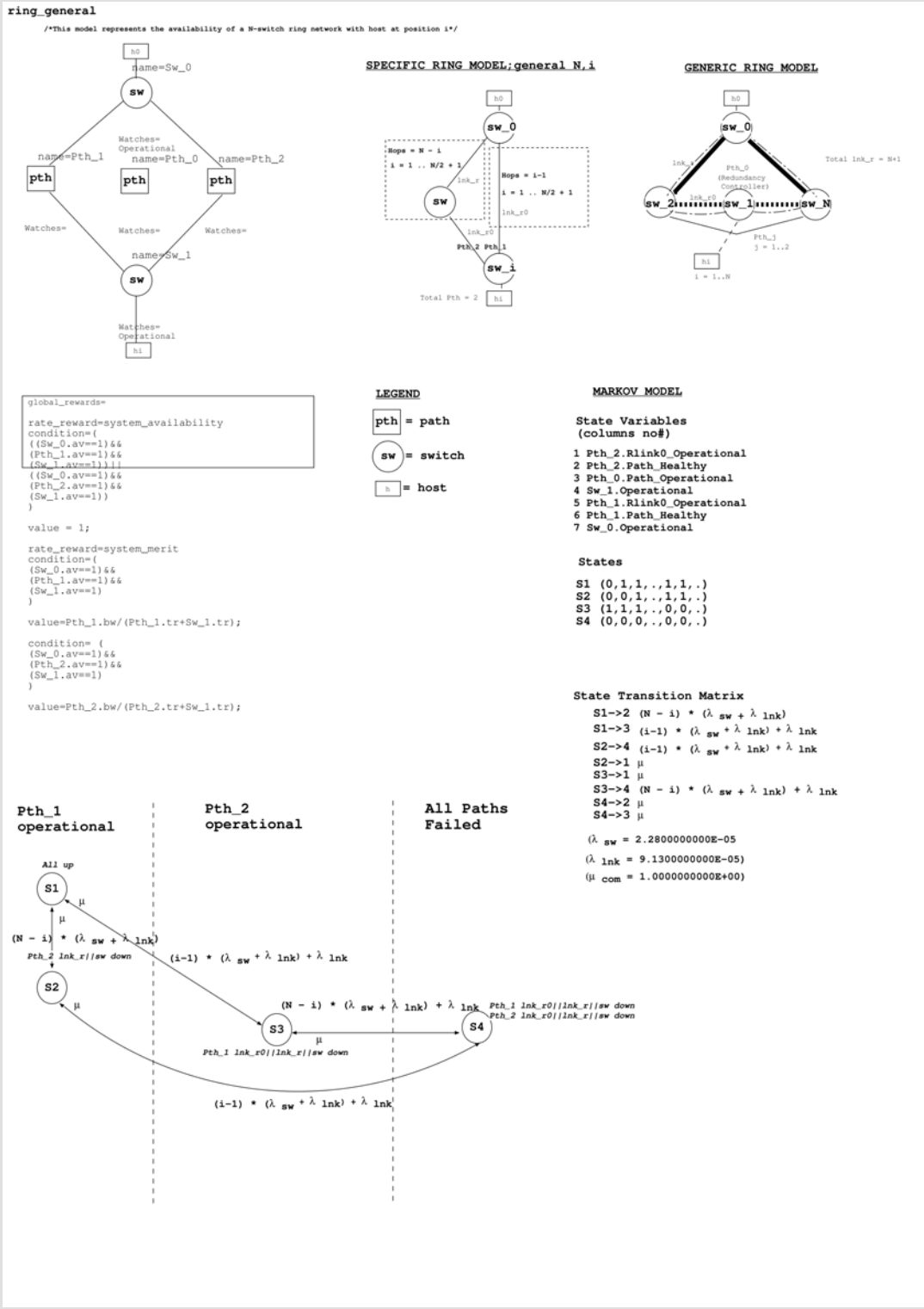

Figure A.21. Analytical Model F20: Generic ring network model with switches = N, i = 1 to N/2 + 1

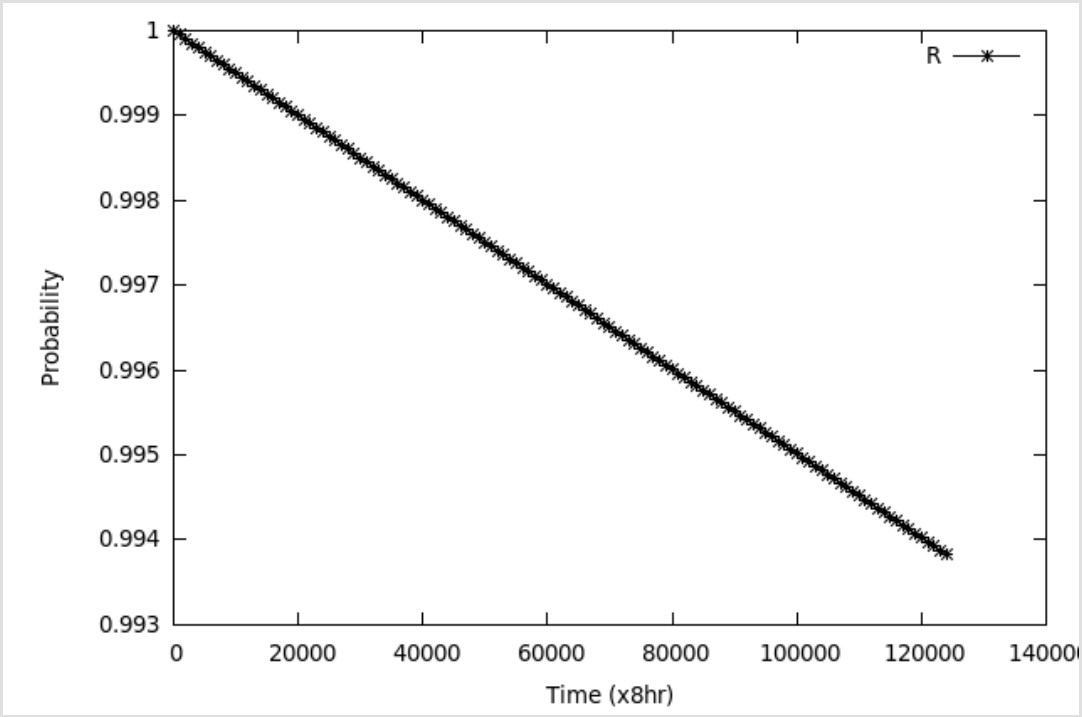

Figure A.22. R(t) Model F2: Ring network model with switches = 3, i = 1

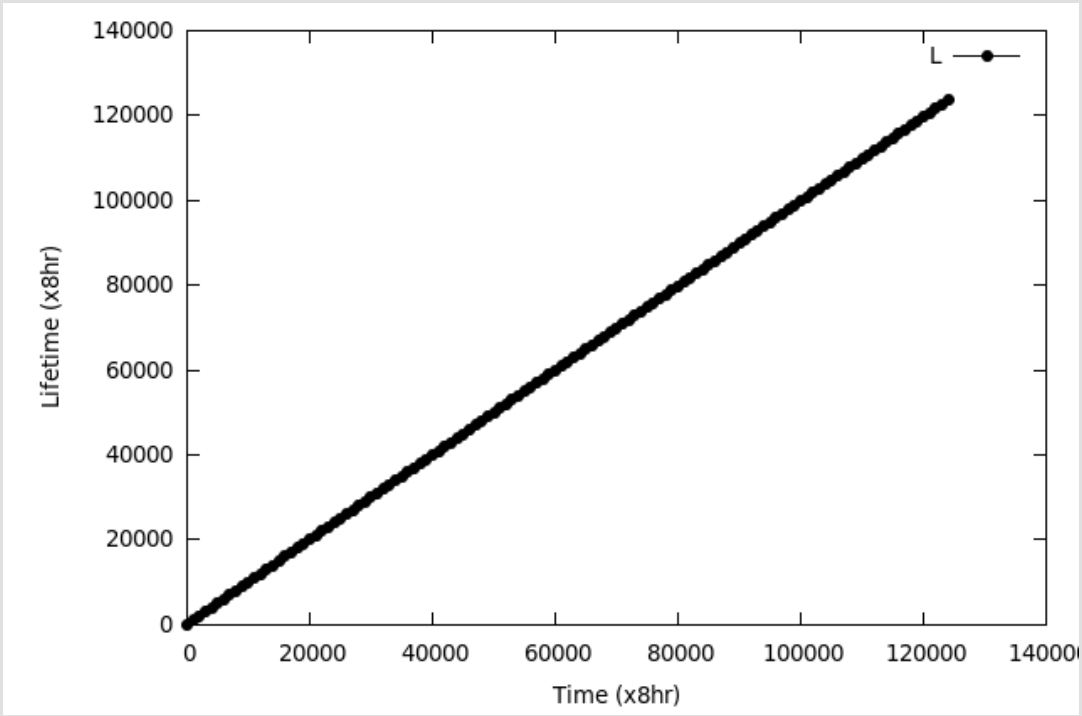

Figure A.23. L(t) Model F2: Ring network model with switches = 3, i = 1

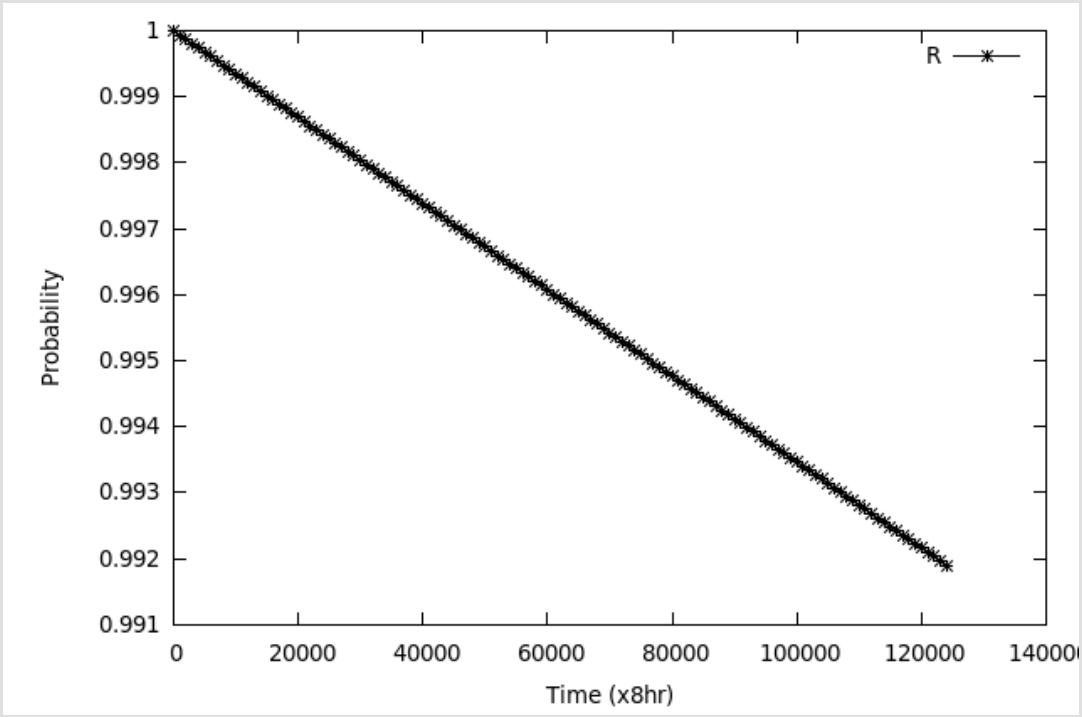

Figure A.24. R(t) Model F3: Ring network model with switches = 3, i = 2

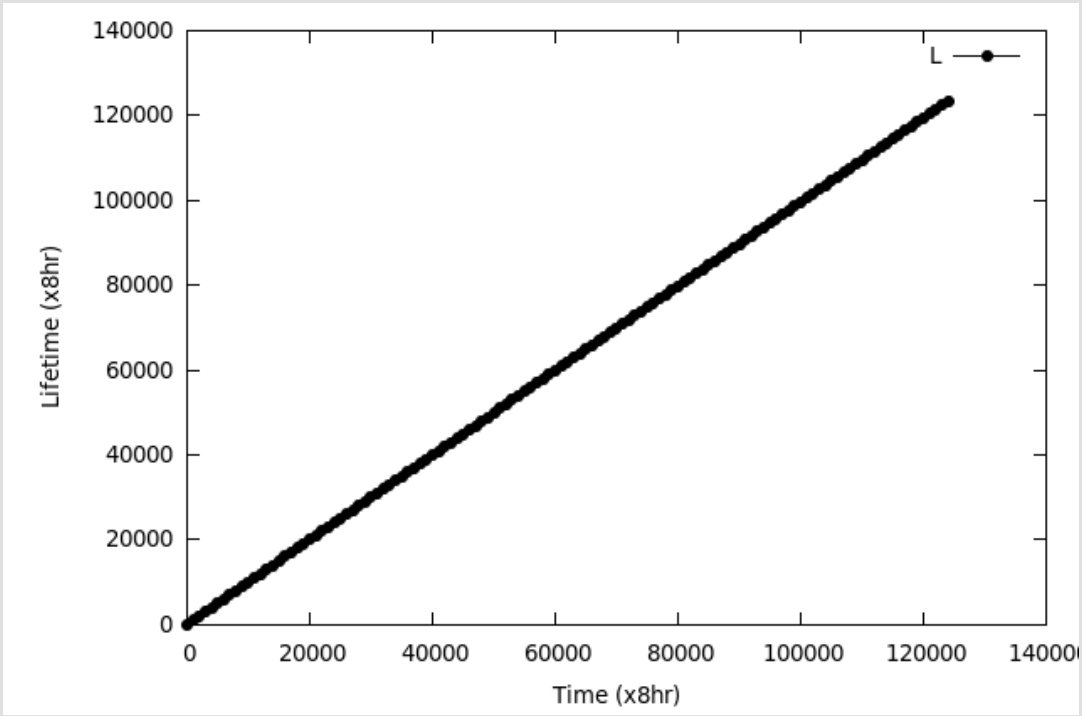

Figure A.25. L(t) Model F3: Ring network model with switches = 3, i = 2

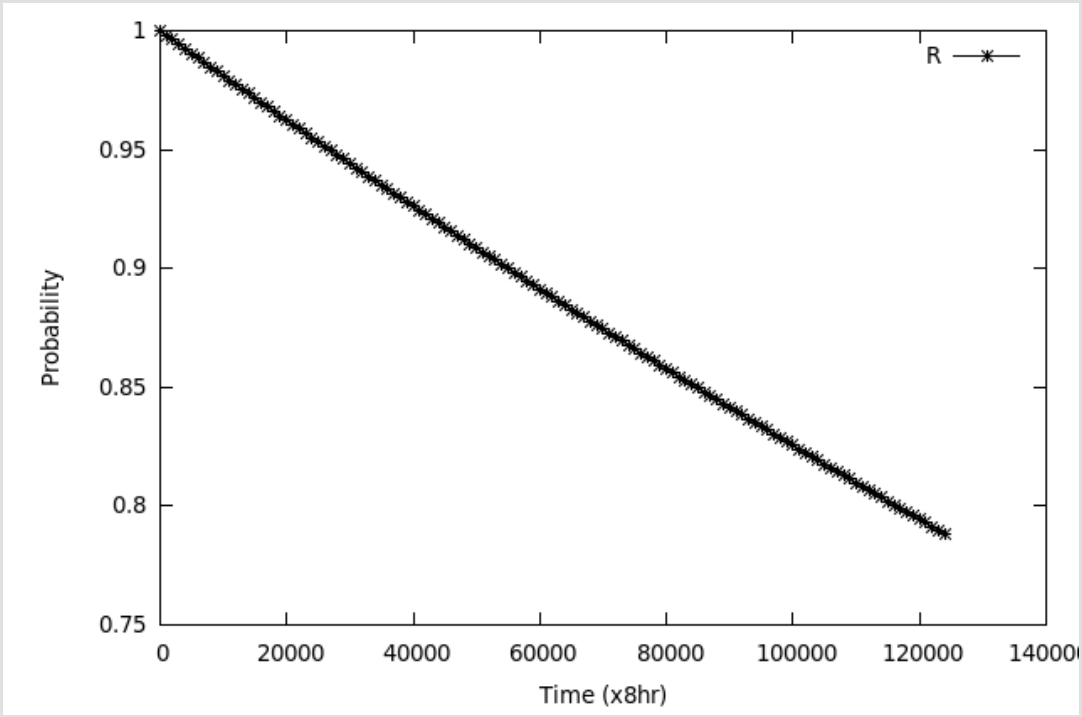

Figure A.26. R(t) Model F11: Ring network model with switches = 17, i = 9

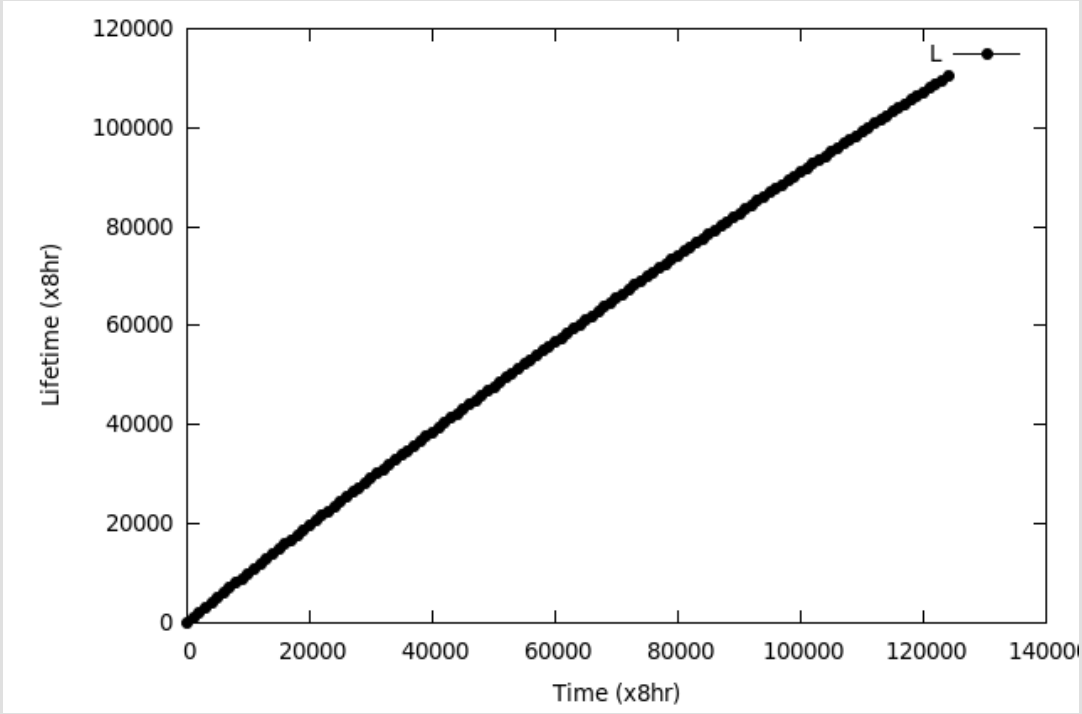

Figure A.27. L(t) Model F11: Ring network model with switches = 17, i = 9

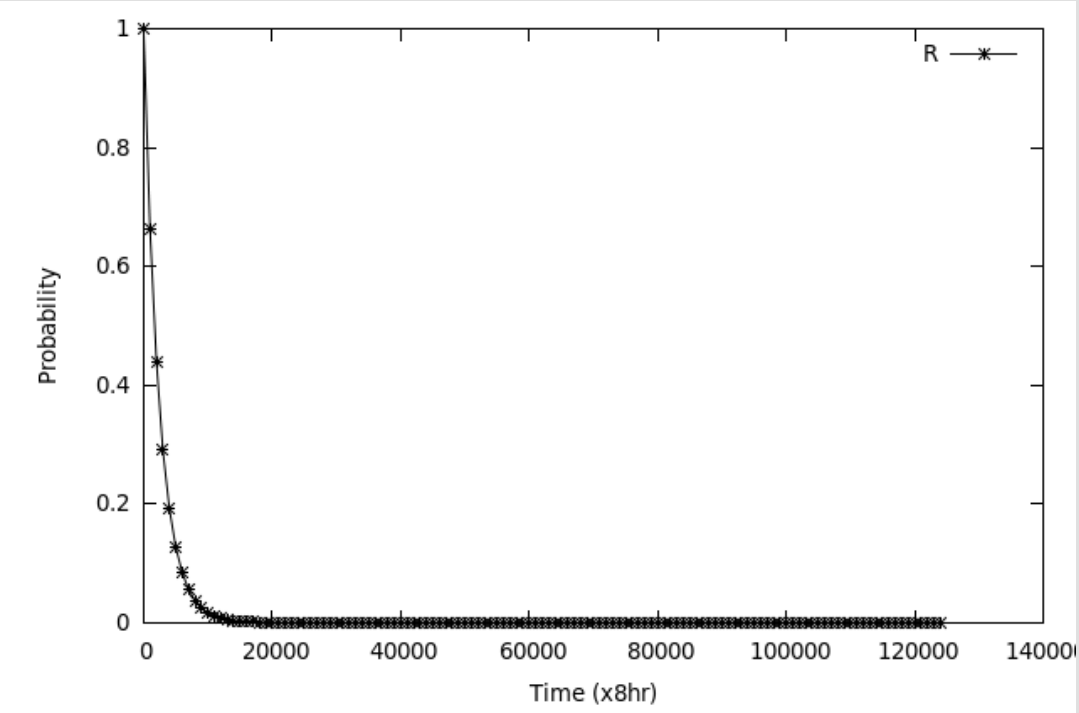

Figure A.28. R(t) Model F19: Ring network model with switches = 257, i = 129

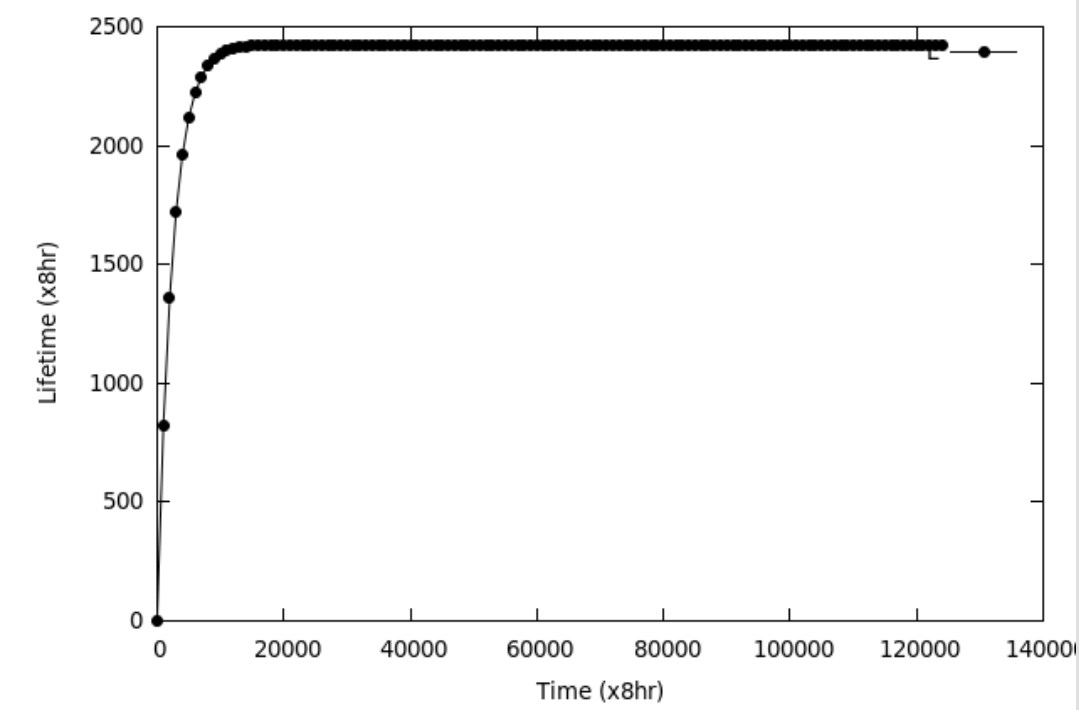

Figure A.29. L(t) Model F19: Ring network model with switches = 257, i = 129

A.9. Model G

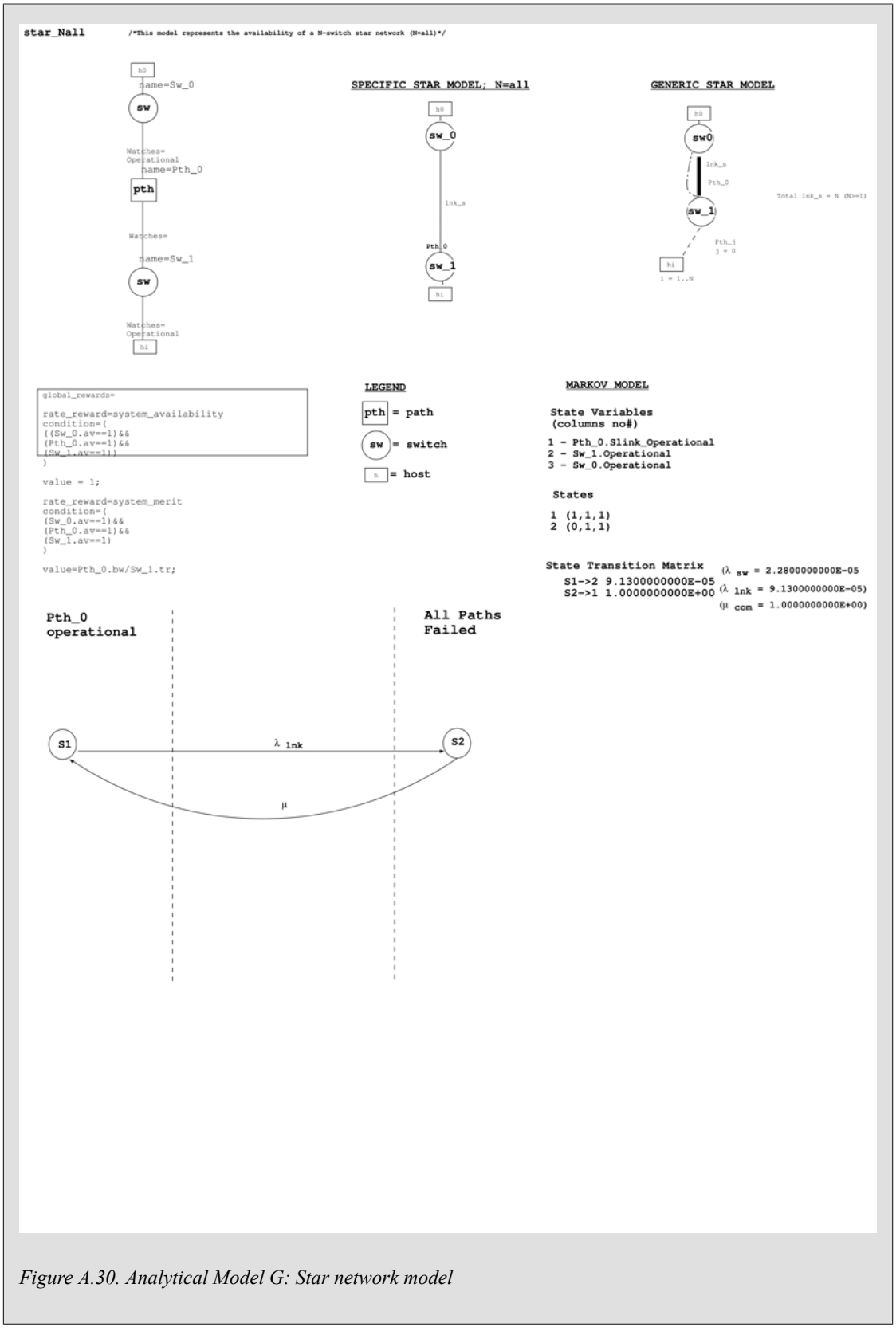

Figure A.30. Analytical Model G: Star network model

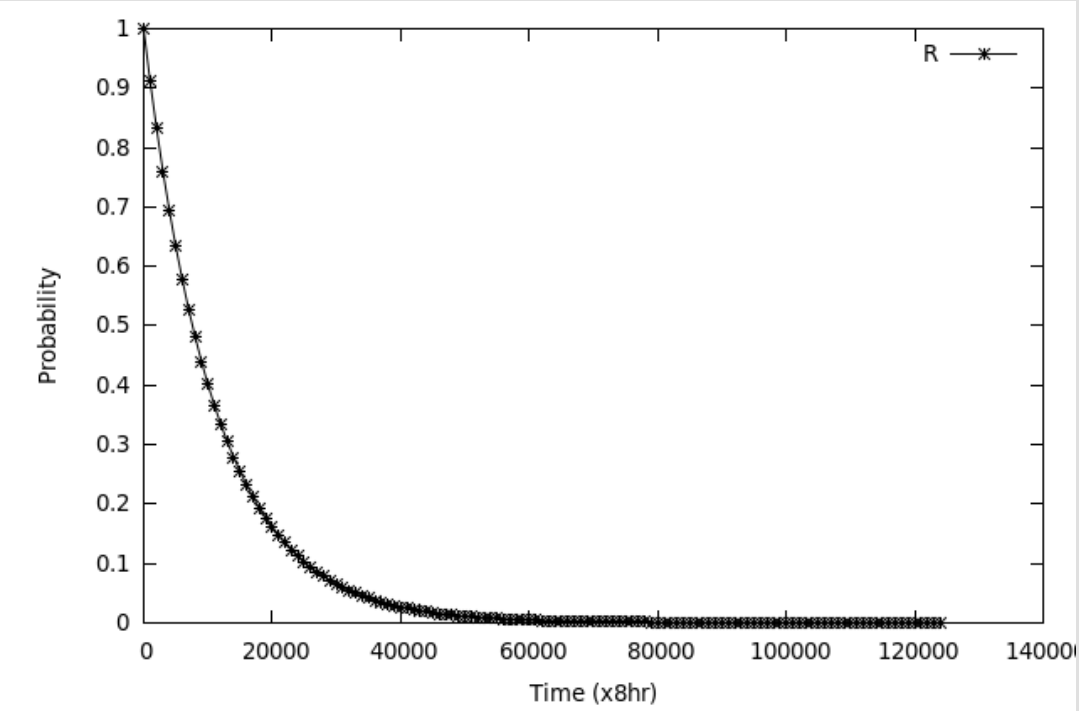

Figure A.31. R(t) Model G: Star network model

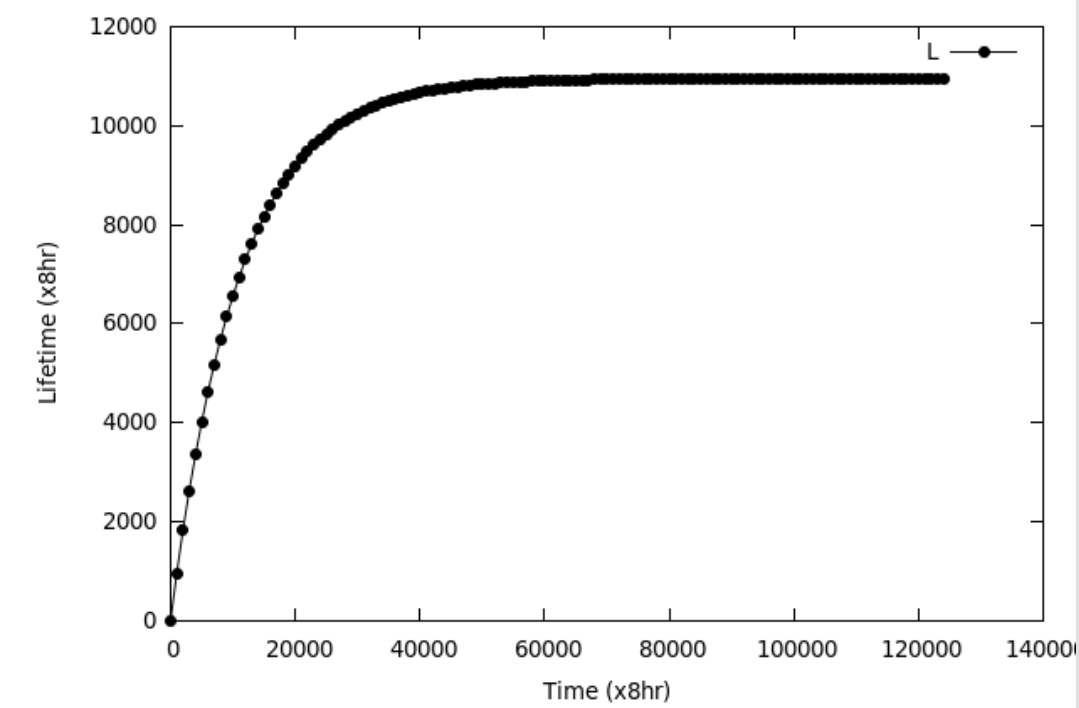

Figure A.32. L(t) Model G: Star network model

A.10. Model H

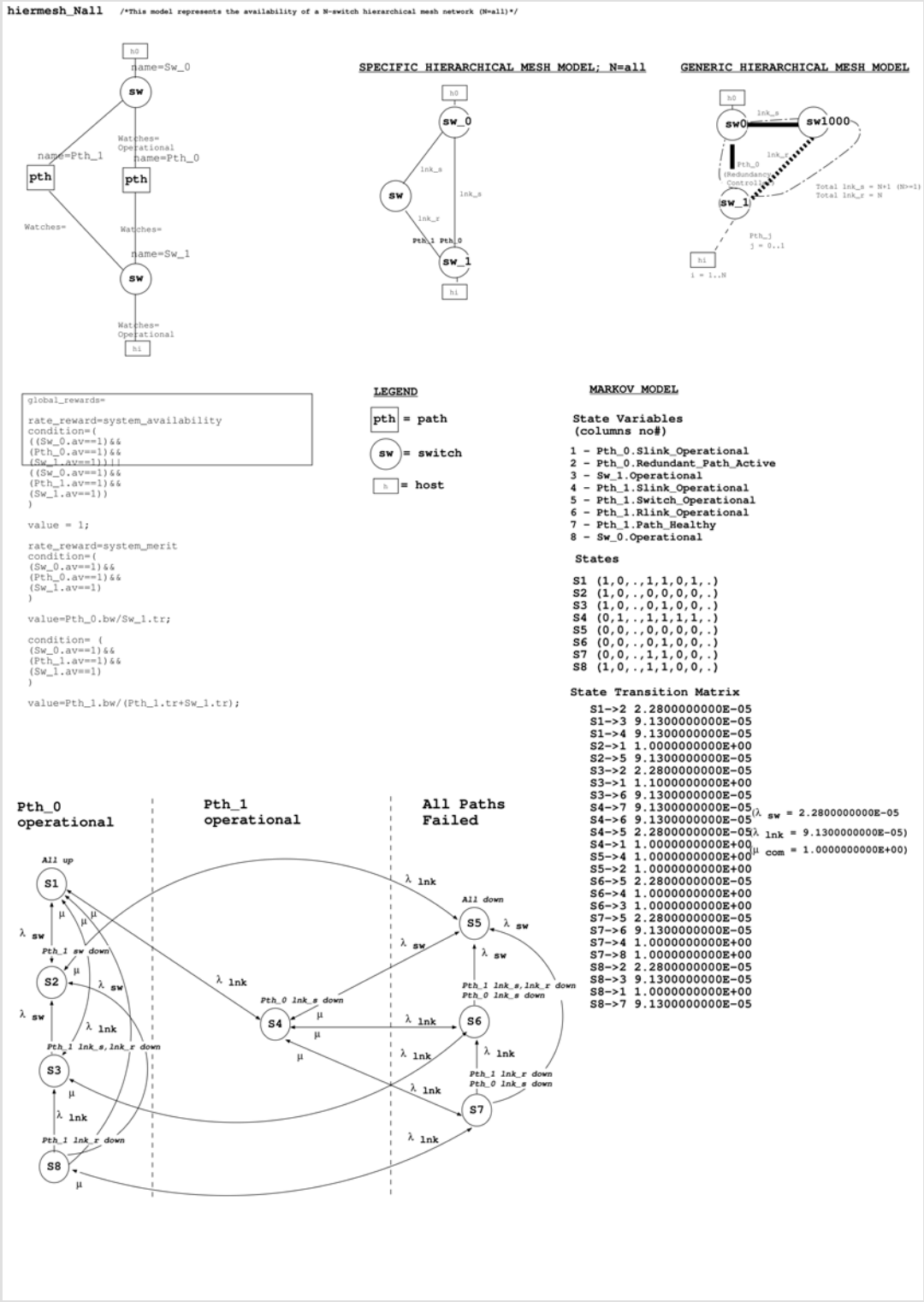

Figure A.33. Analytical Model H: Hierarchical mesh network model

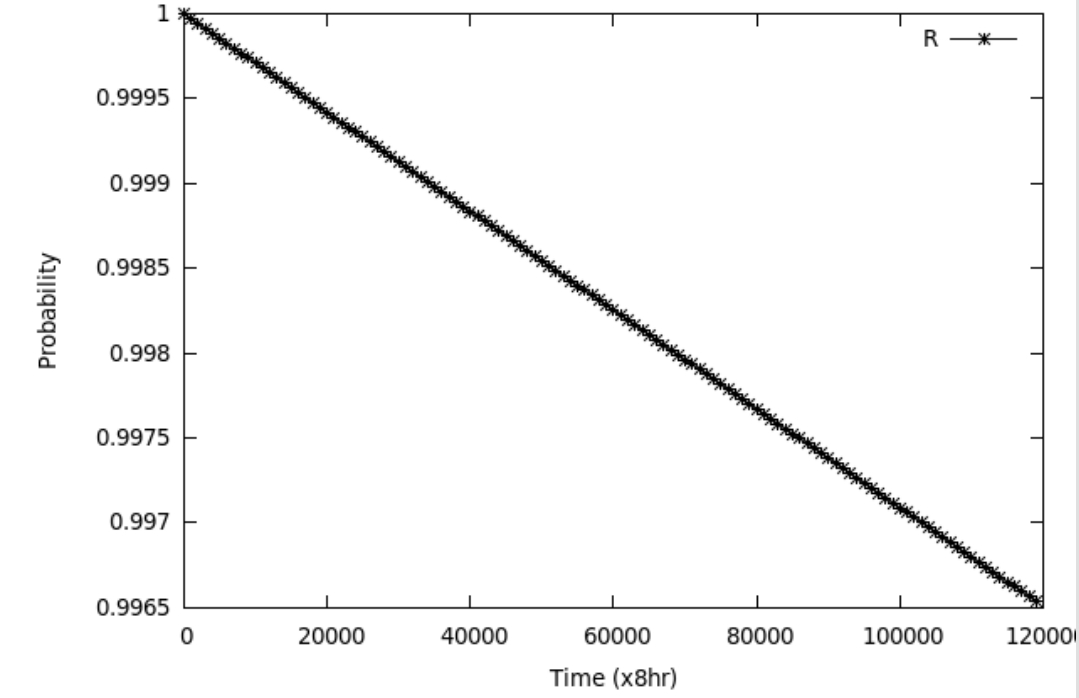

Figure A.34. R(t) Model H: Hierarchical mesh network model

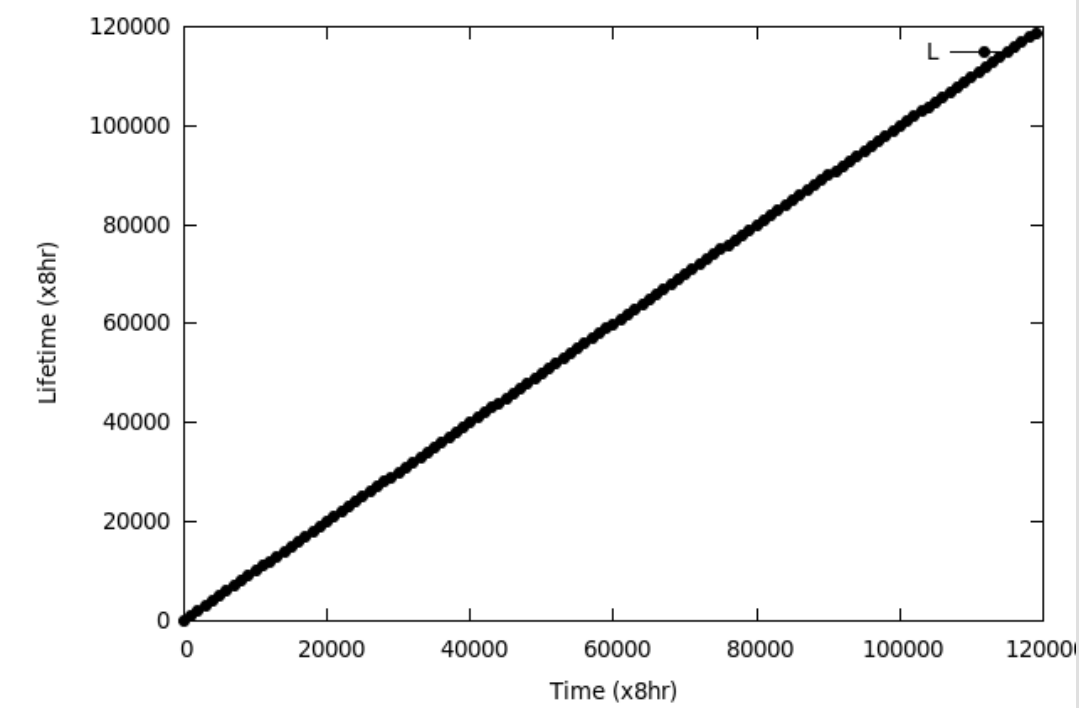

Figure A.35. L(t) Model H: Hierarchical mesh network model

Appendix B. Tangram model: Example C source code

model_2.user_code.c

```
//-----------------------------------------------------------------------------
//        T A N G R A M - I I      << . >>      L A N D - U F R J - B R A Z
//-----------------------------------------------------------------------------
//            This file is generated automatically by the Tangram tool.
//        It contains the user code for actions, messages and expressions.
//-----------------------------------------------------------------------------

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <math.h>
#include <signal.h>

void action_code_alarm_handler(int); /* handler for action code alarm */
#include <user_code.h>

//-----------------------------------------------------------------------------
/* execution context data structure */
extern t_context context;
//-----------------------------------------------------------------------------
static char *all = "*";
int CAN_GENERATE_CHAIN = 1;
int CAN_SIMULATE = 1;


//-----------------------------------------------------------------------------
//  user code for object: Nic_1
//-----------------------------------------------------------------------------

//-----------------------------------------------------------------------------
TGFLOAT Nic_1_Eve_Fail_EXP_rate (Object_State *obj_st)
{
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";

 return ( (TGFLOAT) ( FAILURE_RATE ) );
}
//-----------------------------------------------------------------------------
TGFLOAT Nic_1_Eve_Fail_cond (Object_State *obj_st, Simulator *simulator)
{
```

```
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";
 char *ev_name_param = "Fail";

 return ( (TGFLOAT) ( Operational )==(1) );
}
//-------------------------------------------------------------------------
TGFLOAT Nic_1_Eve_Fail_act_1 (Object_State *obj_st, Simulator *simulator)
{
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";

{
           ;set_st("Operational",0);
       };

 /* the new state var values must be updated by the user */
  return ( (TGFLOAT) (0) );
}
//-------------------------------------------------------------------------
TGFLOAT Nic_1_Eve_Repair_EXP_rate (Object_State *obj_st)
{
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";

 return ( (TGFLOAT) ( REPAIR_RATE ) );
}
//-------------------------------------------------------------------------
TGFLOAT Nic_1_Eve_Repair_cond (Object_State *obj_st, Simulator *simulator)
{
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";
 char *ev_name_param = "Repair";

 return ( (TGFLOAT) ( Operational )==(0) );
}
//-------------------------------------------------------------------------
TGFLOAT Nic_1_Eve_Repair_act_1 (Object_State *obj_st, Simulator *simulator)
{
```

```
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";

{
           ;set_st("Operational",1);
        };

 /* the new state var values must be updated by the user */
  return ( (TGFLOAT) (0) );
}
//-------------------------------------------------------------------------
TGFLOAT Nic_1_Rew_nic_availability_cond_1 (Object_State *obj_st, Simulator *
{
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";

 return ( (TGFLOAT) ( Operational )==(1) );
}
//-------------------------------------------------------------------------
TGFLOAT Nic_1_Rew_nic_availability_act_1 (Object_State *obj_st, Simulator *s
{
 /* declaration and initialization of object variables reference */
 int Operational = obj_st->show_st_var_int_value("Operational");
 const TGFLOAT FAILURE_RATE = 3.040000e-05;
 const TGFLOAT REPAIR_RATE = 1.000000e+00;
 char *nic_availability = "nic_availability";

 return ( (TGFLOAT) (1) );
}
//-------------------------------------------------------------------------
TGFLOAT GlobalReward_system_availability_cond_1 (System_State *sys_st)
{

 return ( (TGFLOAT) ((((sys_st->show_reward_val("Nic_1.nic_availability")))=
}
//-------------------------------------------------------------------------
TGFLOAT GlobalReward_system_availability_act_1 (System_State *sys_st)
{

 return ( (TGFLOAT) (1) );
}
//-------------------------------------------------------------------------
TGFLOAT GlobalReward_system_bandwidth_cond_1 (System_State *sys_st)
{
```

```
 return ( (TGFLOAT) ((((sys_st->show_reward_val("Nic_1.nic_availability")))=
}
//------------------------------------------------------------------------
TGFLOAT GlobalReward_system_bandwidth_act_1 (System_State *sys_st)
{

 return ( (TGFLOAT) (100) );
}
void action_code_alarm_handler(int signum)
{
  context.act_time_count+=USER_CODE_MAX_TIME;
  fprintf( stderr, "Warning: %d seconds spent executing action code in Objec
  fprintf( stderr, "         Continuing... (To stop press <Cancel>) \n");
  alarm(USER_CODE_MAX_TIME);
 }
 //------------------------------------------------------------------------
TGFLOAT function_handler(int fd, Object_State *obj_st, Simulator *simulator,
{
 TGFLOAT status = -1;

 context.user_code = 1;
 context.act_time_count=0;
 struct sigaction alarm_handler; // for installing action code alarm handler
 sigset_t alarm_mask; // to set mask for action code alarm handler
 alarm_handler.sa_handler = action_code_alarm_handler; // name of action coo
 alarm_handler.sa_flags = SA_RESTART; // flag is just RESTART
 sigemptyset(&alarm_mask); // clear all bits of blocked set
  sigaddset(&alarm_mask, SIGALRM);
 alarm_handler.sa_mask = alarm_mask; // set this empty set to be the mask
 if ( sigaction(SIGALRM, &alarm_handler, NULL) == -1 ) // try to install the
 {
  perror("sigaction");
  return (1);
 }
 alarm(USER_CODE_MAX_TIME);

 switch (fd) {
  case 1 : status = Nic_1_Eve_Fail_EXP_rate (obj_st); break;

  case 2 : status = Nic_1_Eve_Fail_cond (obj_st, simulator); break;

  case 3 : status = Nic_1_Eve_Fail_act_1 (obj_st, simulator); break;

  case 4 : status = Nic_1_Eve_Repair_EXP_rate (obj_st); break;

  case 5 : status = Nic_1_Eve_Repair_cond (obj_st, simulator); break;

  case 6 : status = Nic_1_Eve_Repair_act_1 (obj_st, simulator); break;

  case 7 : status = Nic_1_Rew_nic_availability_cond_1 (obj_st, simulator); b
```

```
  case 8 : status = Nic_1_Rew_nic_availability_act_1 (obj_st, simulator); br

  case 9 : status = GlobalReward_system_availability_cond_1 (sys_st); break;

  case 10 : status = GlobalReward_system_availability_act_1 (sys_st); break;

  case 11 : status = GlobalReward_system_bandwidth_cond_1 (sys_st); break;

  case 12 : status = GlobalReward_system_bandwidth_act_1 (sys_st); break;

  default : fprintf ( stderr, "ERROR: Invalid function descriptor\n"); statu
 }
 context.user_code = 0;
 alarm(0);

 return(status);
}
//------------------------------------------------------------------------
```

Appendix C. OPNET Failure-Recovery process model: c source code

Included in attached DVD.

oms_failrec_support_devel_2.h

oms_failrec_support_devel_2.c

Appendix D. Simulation data processing programs: Python source code

Included in attached DVD.

data_crunch_av.py:

data_crunch_m.py:

DES_crunch.py:

calc_from_failures_av.py:

Appendix E. Simulation result outputs

E.1. Model B validation test outputs

PCM calculation of A output with sim. time = 150 thousand seconds:

```
SAMPLE SIMULATION NAME = host to host, 150 thou, 32 runs - seeds

TOTAL SIMULATION TIME = 150000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 3
SAMPLES RUNS= 32
STP NOISE LEVEL = 0.0
TOTAL VALUES PER RUN = 50000


RUNS CALCULATED MEAN AVAILABILITY = 0.99972125

MODEL PREDICTED MTTF = 30052.6
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.99973387
ERROR SIMULATION MEAN AVAILABILITY = -0.001263%

*****
```

E.2. Model F8 validation test outputs

PCM calculation of M output with sim time = 2.4 million seconds:

```
TOTAL SIMULATION TIME = 2400000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 6
SAMPLES RUNS = 32
CALC TRUNCATE TIME = 10308.0
TOTAL VALUES PER RUN = 400000


RUNS CALCULATED MEAN MERIT = 0.200

MODEL PREDICTED MERIT = 0.200
M SCALING FACTOR = 94.745
ERROR SIMULATION MEAN MERIT = 0.046%
*****
```

E.3. Model F9 validation test outputs

PFM calculation of A output with sim. time = 60 million seconds:

```
run no = 1 fail pair_no = 0 fail pair_no = 1
```

```
 total fail duration in run = 17.0
 last recovery time in run = 54667249.0

run no = 2 fail pair_no = 0 fail pair_no = 1
fail pair_no = 2 fail pair_no = 3

 total fail duration in run = 26.0
 last recovery time in run = 50852085.0

run no = 3 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 8.0
 last recovery time in run = 55525678.0

run no = 4 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5 fail pair_no = 6

 total fail duration in run = 29.0
 last recovery time in run = 47278555.0

run no = 5 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 9.0
 last recovery time in run = 53171894.0

run no = 6 fail pair_no = 0 fail pair_no = 1

 total fail duration in run = 18.0
 last recovery time in run = 43272459.0

run no = 7 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 25.0
 last recovery time in run = 42221514.0

run no = 8 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5 fail pair_no = 6

 total fail duration in run = 32.0
 last recovery time in run = 56136235.0

run no = 9 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 8.0
 last recovery time in run = 30983920.0

run no = 10 fail pair_no = 0 fail pair_no = 1
```

```
 total fail duration in run = 7.0
 last recovery time in run = 48339372.0

run no = 11 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 9.0
 last recovery time in run = 52317816.0

run no = 12 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5 fail pair_no = 6

 total fail duration in run = 43.0
 last recovery time in run = 56396770.0

run no = 13 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 10.0
 last recovery time in run = 17289808.0

run no = 14 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 22.0
 last recovery time in run = 51972288.0

run no = 15 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4

 total fail duration in run = 10.0
 last recovery time in run = 54303891.0

run no = 16 fail pair_no = 0

 total fail duration in run = 2.0
 last recovery time in run = 23764848.0


PING SUMS SENT PER RUN
60000000.0 60000000.0 60000000.0 60000000.0 60000000.0 60000000.0
60000000.0 60000000.0 60000000.0 60000000.0 60000000.0 60000000.0
60000000.0 60000000.0 60000000.0 60000000.0
PING SUMS RECEIVED PER RUN
59999983.0 59999974.0 59999992.0 59999971.0 59999991.0 59999982.0
59999975.0 59999968.0 59999992.0 59999993.0 59999991.0 59999957.0
59999990.0 59999978.0 59999990.0 59999998.0
AVERAGE AVAILABILITY PER RUN
0.999999716667 0.999999566667 0.999999866667 0.999999516667
0.99999985 0.9999997 0.999999583333 0.999999466667 0.999999866667
0.999999883333 0.99999985 0.999999283333 0.999999833333 0.999999633333
0.999999833333 0.999999966667
```

```
*****
CALC SIMULATION NAME = ring n=9 i=5, 60 mil seconds,
3 runs - 3|51|61|110|6|49|96|123|68|66|10|97|111|114|9|27

TOTAL SIMULATION TIME = 60000000
CALC PING PERIOD = 1.0
SAMPLE UNITS = seconds
SAMPLES RUNS= 16


RUNS CALCULATED MEAN AVAILABILITY = 0.99999971

MODEL PREDICTED MTTF = 14569467.9
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.99999945
ERROR SIMULATION MEAN AVAILABILITY = 0.000026%

*****
```

PCM calculation of M output with sim time = 2.4 million seconds:

```
TOTAL SIMULATION TIME = 2400000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 6
SAMPLES RUNS = 32
CALC TRUNCATE TIME = 6240.0
TOTAL VALUES PER RUN = 400000


RUNS CALCULATED MEAN MERIT = 0.200

MODEL PREDICTED MERIT = 0.200
M SCALING FACTOR = 94.745
ERROR SIMULATION MEAN MERIT = 0.031%
*****
```

E.4. Model F10 validation test outputs

PFM calculation of A output with sim. time = 94 million seconds:

```
run no = 1 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5 fail pair_no = 6

 total fail duration in run = 37.0
 last recovery time in run = 90129259.0

run no = 2 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 15.0
```

```
 last recovery time in run = 92605295.0

run no = 3 fail pair_no = 0 fail pair_no = 1

 total fail duration in run = 16.0
 last recovery time in run = 55122873.0

run no = 4 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 30.0
 last recovery time in run = 67727747.0

run no = 5 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 12.0
 last recovery time in run = 75740553.0

run no = 7 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 27.0
 last recovery time in run = 68466114.0

run no = 8 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 28.0
 last recovery time in run = 74971426.0

run no = 9 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 12.0
 last recovery time in run = 83825407.0

run no = 10 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 13.0
 last recovery time in run = 61327359.0

run no = 11 fail pair_no = 0 fail pair_no = 1

 total fail duration in run = 2.0
 last recovery time in run = 92788286.0


PING SUMS SENT PER RUN
94000000.0 94000000.0 94000000.0 94000000.0 94000000.0 94000000.0
```

```
94000000.0 94000000.0 94000000.0 94000000.0 94000000.0
PING SUMS RECEIVED PER RUN
93999963.0 93999985.0 93999984.0 93999970.0 93999988.0 94000000.0
93999973.0 93999972.0 93999988.0 93999987.0 93999998.0
AVERAGE AVAILABILITY PER RUN
0.999999606383 0.999999840426 0.999999829787 0.999999680851
0.99999987234 1.0 0.999999712766 0.999999702128 0.99999987234
0.999999861702 0.999999978723
*****
CALC SIMULATION NAME = ring n=17 i=1, 94 mil seconds, 11 runs

TOTAL SIMULATION TIME = 94000000
CALC PING PERIOD = 1.0
SAMPLE UNITS = seconds
SAMPLES RUNS= 11


RUNS CALCULATED MEAN AVAILABILITY = 0.99999981

MODEL PREDICTED MTTF = 23481947.2
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.99999966
ERROR SIMULATION MEAN AVAILABILITY = 0.000016%

*****
```

PCM calculation of M output with sim time = 600 thousand seconds:

```
TOTAL SIMULATION TIME = 600000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 3
SAMPLES RUNS = 3
CALC TRUNCATE TIME = 6240.0
TOTAL VALUES PER RUN = 200000


RUNS CALCULATED MEAN MERIT = 0.075

MODEL PREDICTED MERIT = 0.111
M SCALING FACTOR = 94.745
ERROR SIMULATION MEAN MERIT = -32.819%
```

E.5. Model F11 validation test outputs

PFM calculation of A output with sim. time = 17 million seconds:

```
run no = 1 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 18.0
 last recovery time in run = 16498774.0
```

```
run no = 2 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3

 total fail duration in run = 13.0
 last recovery time in run = 16927843.0

run no = 3 fail pair_no = 0 fail pair_no = 1

 total fail duration in run = 7.0
 last recovery time in run = 13329936.0

run no = 4 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4

 total fail duration in run = 13.0
 last recovery time in run = 16094594.0

run no = 5 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 4.0
 last recovery time in run = 13550032.0

run no = 6 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 5.0
 last recovery time in run = 10166295.0

run no = 7 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 26.0
 last recovery time in run = 13072043.0

run no = 8 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5 fail pair_no = 6

 total fail duration in run = 20.0
 last recovery time in run = 15596878.0

run no = 9 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4

 total fail duration in run = 22.0
 last recovery time in run = 15811264.0

run no = 10 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 31.0
 last recovery time in run = 12249274.0

run no = 11 fail pair_no = 0 fail pair_no = 1
```

```
 total fail duration in run = 21.0
 last recovery time in run = 7557878.0

run no = 12 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 38.0
 last recovery time in run = 15750485.0

run no = 13 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 15.0
 last recovery time in run = 12511416.0

run no = 14 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2
fail pair_no = 3 fail pair_no = 4 fail pair_no = 5

 total fail duration in run = 16.0
 last recovery time in run = 14643702.0

run no = 15 fail pair_no = 0 fail pair_no = 1 fail pair_no = 2

 total fail duration in run = 14.0
 last recovery time in run = 4809774.0

run no = 16 fail pair_no = 0 fail pair_no = 1

 total fail duration in run = 14.0
 last recovery time in run = 12362734.0


PING SUMS SENT PER RUN
17000000.0 17000000.0 17000000.0 17000000.0 17000000.0 17000000.0
17000000.0 17000000.0 17000000.0 17000000.0 17000000.0 17000000.0
17000000.0 17000000.0 17000000.0 17000000.0
PING SUMS RECEIVED PER RUN
16999982.0 16999987.0 16999993.0 16999987.0 16999996.0 16999995.0
16999974.0 16999980.0 16999978.0 16999969.0 16999979.0 16999962.0
16999985.0 16999984.0 16999986.0 16999986.0
AVERAGE AVAILABILITY PER RUN
0.999998941176 0.999999235294 0.999999588235 0.999999235294
0.999999764706 0.999999705882 0.999998470588 0.999998823529
0.999998705882 0.999998176471 0.999998764706 0.999997764706
0.999999117647 0.999999058824 0.999999176471 0.999999176471
*****
CALC SIMULATION NAME = ring n=17 i=9, 17 mil minutes, 16 runs

TOTAL SIMULATION TIME = 17000000
CALC PING PERIOD = 1.0
SAMPLE UNITS = seconds
```

```
SAMPLES RUNS= 16


RUNS CALCULATED MEAN AVAILABILITY = 0.99999898

MODEL PREDICTED MTTF = 41685045.8
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.99999808
ERROR SIMULATION MEAN AVAILABILITY = 0.000090%

*****
```

PCM calculation of M output with sim time = 600 thousand seconds:

```
TOTAL SIMULATION TIME = 600000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 3
SAMPLES RUNS = 3
CALC TRUNCATE TIME = 6240.0
TOTAL VALUES PER RUN = 200000


RUNS CALCULATED MEAN MERIT = 0.075

MODEL PREDICTED MERIT = 0.111
M SCALING FACTOR = 94.745
ERROR SIMULATION MEAN MERIT = -32.819%
*****
```

E.6. Model G validation test outputs

PCM calculation of A output with sim. time = 360 thousand seconds:

```
SAMPLE SIMULATION NAME = star n 9 all, 360 thou, 32 runs - seeds

TOTAL SIMULATION TIME = 360000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 3
SAMPLES RUNS= 32
STP NOISE LEVEL = 0.0
TOTAL VALUES PER RUN = 120000


RUNS CALCULATED MEAN AVAILABILITY = 0.99990651

MODEL PREDICTED MTTF = 87623.2
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.99990871
ERROR SIMULATION MEAN AVAILABILITY = -0.000220%
```

```
*****
```

PCM calculation of M output with sim time = 2.4 million seconds:

```
TOTAL SIMULATION TIME = 2400000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 6
SAMPLES RUNS = 32
CALC TRUNCATE TIME = 2400000
TOTAL VALUES PER RUN = 400000


RUNS CALCULATED MEAN MERIT = 1.001

MODEL PREDICTED MERIT = 1.000
M SCALING FACTOR = 94.745
ERROR SIMULATION MEAN MERIT = 0.095%
*****
```

E.7. Model H validation test outputs

PCM calculation of A output with sim. time = 120 million seconds:

```
run no = 1 fail pair_no = 0

 total fail duration in run = 7.0
 last recovery time in run = 98151546.0

run no = 2 fail pair_no = 0

 total fail duration in run = 7.0
 last recovery time in run = 58900639.0

run no = 3 fail pair_no = 0

 total fail duration in run = 0.0
 last recovery time in run = 15249201.0


PING SUMS SENT PER RUN
120000000.0 120000000.0 120000000.0
PING SUMS RECEIVED PER RUN
119999993.0 119999993.0 120000000.0
AVERAGE AVAILABILITY PER RUN
0.999999941667 0.999999941667 1.0
*****
CALC SIMULATION NAME = hiermesh n=9 i=5, 120 mil seconds, 3 runs

TOTAL SIMULATION TIME = 120000000
CALC PING PERIOD = 1.0
SAMPLE UNITS = seconds
```

```
SAMPLES RUNS= 3


RUNS CALCULATED MEAN AVAILABILITY = 0.99999996

MODEL PREDICTED MTTF = 274352564.9
MODEL ASSUMED MTTR = 8.0
MODEL PREDICTED MEAN AVAILABILITY = 0.99999997
ERROR SIMULATION MEAN AVAILABILITY = -0.000001%

*****
```

PCM calculation of M output with sim time = 2.4 million seconds:

```
TOTAL SIMULATION TIME = 2400000
SAMPLE PING PERIOD = 1.0
SAMPLING UNITS = seconds
SAMPLING PERIOD = 6
SAMPLES RUNS = 32
CALC TRUNCATE TIME = 2400000
TOTAL VALUES PER RUN = 400000


RUNS CALCULATED MEAN MERIT = 1.000

MODEL PREDICTED MERIT = 1.000
M SCALING FACTOR = 94.745
ERROR SIMULATION MEAN MERIT = -0.027%
```