Framework for Systematic Evaluation of Protocol Performance with Respect to Out-of-Sequence Packet Arrivals

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde
eines Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

vorgelegt von

Sebastian Gunreben

geb. in Erlangen

Hauptberichter:	Prof. em. Dr.-Ing. Dr. h. c. mult. Paul J. Kühn
1. Mitberichter:	Prof. Dr. Josep Solé-Pareta, Universitat Politècnica de Catalunya
2. Mitberichter:	Prof. Dr.-Ing. Andreas Kirstädter
Tag der Einreichung:	10. Dezember 2009
Tag der mündlichen Prüfung:	22. Dezember 2010

Institut für Kommunikationsnetze und Rechnersysteme
der Universität Stuttgart

2010

Abstract

Protocols of packet-based networks rely on the paradigm that the network maintains the sending order of the packets. Networks realize this paradigm by forwarding packets along shortest paths to destination nodes. In case of unsymmetrical traffic matrices, this leads to unequal load distributions in these networks. Consequences are congestion, which cause packet loss and increase delays. Multi-path routing represents a paradigm shift. It aims for an optimal load distribution in a network to avoid congestion. The major disadvantage of multi-path routing is the potential out-of-sequence packet arrivals at the destination. As protocols rely on in-sequence packet arrivals, they require special mechanisms to cope with out-of-sequence packet arrivals.

A second paradigm of packet-based networks is the individual forwarding of each packet. Networks realize this paradigm by processing the header of each individual packet separately. Increasing traffic volumes and increasing line rates lead to even higher packet rates in metro and core networks. This increases complexity and power-consumption of the network devices. Packet assembly alleviates this problem. At the network edge, it assembles multiple packets of the same class to containers. The network forwards these containers individually maintaining the packet switching principle. Consequently, packet assembly reduces the absolute number of packets to process. One disadvantage of packet assembly is the change in the traffic characteristic of the assembled packets, when disassembling them. Additionally, if containers arrive out-of-sequence, the carried packets may arrive out-of-sequence, too. The same consequences as above apply.

This thesis addresses the problem of out-of-sequence arrivals with respect to the paradigm shifts of above. It proposes a framework, which enables protocol engineering with respect to out-of-sequence packet patterns. This thesis includes an analytic reordering model, which shows two major applications. It enables worst-case estimations on the amount of out-of-sequence arrivals for any traffic model with respect to a predefined network delay distribution. Additionally, it provides the parameterization of a network emulation environment, which allows engineering of real protocol implementations on predefined reordering patterns. Besides the analytic model, this thesis presents an equivalent simulation model as well as a coherent testbed for protocol engineering.

For a substantial background, Chapter 2 introduces and classifies the concepts of multi-path routing as well as packet assembly strategies. The classification criterion for both schemes is the network layer that implements multi-path routing. There are network technologies, which implement multi-path routing as well as packet assembly on the transport,

networking and MAC layer. Examples for network technologies with multi-path routing capabilities are optical burst switching networks for the MAC layer, IP multi-path routing for the networking layer as well as new TCP flavours on the transport layer. Additionally, mobile ad hoc networks also show multi-path routing properties on the MAC layer to increase network availability. Examples for network technologies showing packet assembly on the MAC layer are wireless network technologies, e. g., WiMax and WLAN as well as wired high-speed networks, e. g., optical burst switching and frame switching networks.

Changes in the packet order require special metrics to classify and differentiate reordering patterns. Literature provides a large set of different but inconsistent metrics. Chapter 3 firstly introduces requirements of reordering metrics. Secondly, it introduces and classifies the metrics proposed in the literature. A comprehensive discussion of the proposed metrics leads to the IETF metrics most suitable for the analysis of reordering patterns. Among others, they include the definition of in-sequence and out-of-sequence arrivals, the reordering ratio, the extent metric and a metric to estimate the TCP performance. The remainder of this thesis, especially the reordering model implements these metrics to analyse reordering pattern analytically in a simulation as well as in a testbed.

The analysis on the packet order requires a reordering model to reflect these changes. Literature proposes several queuing theoretical models to analyse packet reordering. Chapter 4 classifies the general ideas of these approaches in contrast to the work presented in this thesis. This chapter introduces the analytic reordering model of this thesis, which shows a discrete delay distribution. The model represents a set of parallel $\cdot$/D/1 models. Each of the $\cdot$/D/1 models shows a different constant delay. Chapter 4 presents for this model the formal analysis for deterministic as well as Poisson traffic models. Additionally to the findings of single layer reordering, the chapter also presents the expressions for hierarchical packet reordering. For selected scenarios, this chapter verifies the findings of the analytic model with equivalent simulation studies.

Chapter 5 applies the analytic reordering model in two different scenarios. The first application enables worst-case estimations on the amount of out-of-sequence arrivals. Section 5.2 provides the formal proof for this worst-case estimation and illustrates these findings for selected scenarios. It further shows that this worst-case property still holds if the mean traffic rate increases, while the disordering network remains in its original settings. The second application (Section 5.4) enables studies of real protocol implementations in testbeds. On basis of the reordering model, this thesis also implements a network emulation module, which changes the packet order according to predefined reordering pattern. The module bases on the Linux kernel using the network emulation module of the Linux Foundation. For the reordering module, this chapter includes a description of the module's state machine as well as a discussion on the representation of random numbers in the Linux kernel.

The parameterization of this module requires the network delay distribution of an observed reordering pattern, i. e., the individual delay of the $\cdot$/D/1 models. Re-engineering a predefined reordering pattern leads to a non-linear constraint optimization problem. Section 5.3 formulates this optimization problem and applies a solver algorithm to obtain the network delay distribution. This probability distribution should lead to the same predefined reordering pattern. A verification of the module and comprehensive comparison

of the findings of the testbed, simulation and analytic model conclude this chapter.

Chapter 6 considers the reordering model and its application in network emulation scenarios. It evaluates the quality of the solution of the optimization problem, which re-engineers the network delay distribution from a given reordering pattern. Thereby, the reordering patterns result from an optical burst switching network simulator. For the evaluation process, this chapter applies hypothesis tests and graphical methods. This chapter shows that the parameterized module creates the same reordering pattern as the simulation results. An additional subject for evaluation is the complexity of the solver. The complexity of the solver in terms of computation time considers two factors. One is the number of iterations the solver requires for a solution and the other is the time to compute the functions of the analytic model. As it was not feasible to determine the complexity of these functions analytically, measurements provide the basis for the analysis of complexity.

Summarizing, this thesis provides a concise framework for comprehensive studies of protocol performance with respect to out-of-sequence packet patterns obtained from measurements or simulations.

Kurzfassung

Protokolle in paketbasierten Netzen bauen auf das Paradigma, dass die Netze die Paketreihenfolge unverändert wiedergeben. Diese Netze realisieren dieses Paradigma indem sie alle Pakete auf dem kürzesten Pfad zum Zielknoten weiterleiten. Ist die Verkehrsmatrix in einem solchen Netz allerdings unsymmetrisch, wird auch der Verkehr innerhalb eines Netzes ungleichmäßig verteilt. In der Konsequenz entstehen Überlastbereiche, die zu Packetverlusten und Verzögerungen führen. Mehrwegevermittlung (multi-path routing) bricht mit diesem Paradigma. Es zielt darauf ab, die Last innerhalb eines Netzes optimal zu verteilen, um Überlastbereiche zu vermeiden. Der große Nachteil der Mehrwegevermittlung sind mögliche Reihenfolgeänderungen am Zielknoten. Da die Protokolle allerdings auf der ursprünglichen Reihenfolgeannahme basieren, benötigen sie Mechanismen, um mit Paketen außerhalb der Reihenfolge umgehen zu können.

Ein zweites Paradigma in paketbasierten Netzen ist die individuelle Vermittlung jedes einzelnen Paketes. Netze realisieren dieses Paradigma indem sie jeden einzelnen Protokollkopf separat verarbeiten. Steigendes Verkehrsvolumen und steigende Datenraten führen zu immer höheren Paketraten in Metro- und Kernnetzen. Dies wiederum erhöht die Komplexität und den Energiebedarf der Netzknoten. Eine Paketaggregation verringert dieses Problem. Sie aggregiert am Netzrand mehrere Pakete der gleichen Klasse zu größeren Containern zusammen. Diese Container werden vom Netz wieder einzeln prozessiert, so dass die ursprünglichen Vorteile der Paketvermittlung erhalten bleiben. Damit reduziert die Paketaggregation die absolute Anzahl von Paketen im Netz. Der große Nachteil von Paketaggregation ist die Änderung der Verkehrscharakteristik der aggregierten Pakete. Weiter geraten aggregierte Pakete außer Reihenfolge, wenn der Container außer der Reihenfolge am Zielknoten ankommt. Damit ergeben sich für die Protokolle dieselben Konsequenzen wie oben beschrieben.

Diese Arbeit beschäftigt sich mit dem Problem der veränderten Paketreihenfolge unter der Annahme obiger Paradigmenwechsel. Sie beinhaltet ein Vorgehensmodell zur Untersuchung von Protokollen im Hinblick auf eine veränderte Paketankunftsreihenfolge. Der Kern ist dabei ein analytisches Modell für eine veränderte Paketreihenfolge. Dieses Modell dient zwei Hauptanwendungsgebieten. Es ermöglicht den Anteil der Pakete abzuschätzen, die außerhalb der Reihenfolge ankommen. Weiter ermöglicht das Modell eine Netzemulation, die es gestattet, Protokolle im Hinblick auf eine bestimmte Reihenfolgeänderung zu untersuchen. Neben dem analytischen Modell stellt diese Arbeit ein äquivalentes Simulationsmodell und eine damit verbundene Experimentierplattform vor.

Kapitel 2 stellt verschiedene Konzepte der Mehrwegevermittlung sowie der Aggregationsmechanismen vor und klassifiziert diese nach der Schicht in der sie implementiert werden. Verschiedene Netztechnologien implementieren Mehrwegevermittlung und Packetaggregation auf Transport, Netz- oder MAC-Schicht. Beispiele für Netztechnologien, die Mehrwegevermittlung implementieren, sind optische aggregatvermittelnde Netze auf der MAC-Schicht und verschiedene Protokolle auf Transport- und Netzschicht. Zusätzlich implementieren mobile ad hoc Netze Mehrwegevermittlung, um die Netzverfügbarkeit zu erhöhen. Beispiele für Netztechnologien, die Packetaggregation implementieren, sind kabellose Netze wie WiMax und WLAN und kabelgebundene Hochgeschwindigkeitsnetze wie zum Beispiel optische aggregatvermittelnde Netze sowie rahmenvermittelnde Netztechnologien.

Änderungen in der Reihenfolge bedürfen besondere Metriken um Reihenfolgemuster zu unterscheiden und zu klassifizieren. Die Literatur schlägt dabei eine Reihe von verschiedenen, leider inkonsistenten Metriken, vor. Zunächst führt Kapitel 3 die Anforderungen an solche Metriken ein. Als zweites stellt es die in der Literatur vorgeschlagenen Metriken vor und klassifiziert diese im Hinblick auf die Anforderungen. Eine vergleichende Übersicht der Metriken führt zu den von der IETF vorgeschlagenen Metriken, die am Besten für die Untersuchungen von Reihenfolgeänderungen geeignet sind. Unter anderem definieren diese Metriken die Begriffe in Reihenfolge und außerhalb der Reihenfolge. Weitere Metriken bestimmen den Anteil der Pakete, die außerhalb der Reihenfolge ankommen, quantifizieren den Versatz eines Pakete oder schätzen den Einfluss auf das Transportprotokoll TCP ab. Diese Arbeit mit ihrem analytischen Modell implementiert diese Metriken, um eine Reihenfolgeänderung zu bewerten.

Die Analyse der Reihenfolgeänderung bedarf eines geeigneten Modells. Viele verkehrstheoretische Modelle zur Reihenfolgeänderung werden in der Literatur vorgeschlagen. Für eine erste Klassifikation fasst Kapitel 4 diese Ideen und Ergebnisse zusammen und grenzt diese zu dem Modell dieser Arbeit ab. Dieses Kapitel stellt im Weiteren das eigentliche Model zur Reihenfolgeänderung vor. Das Modell entspricht in den Grundzügen mehreren parallelen $\cdot/D/1$ Modellen. Die Verzögerung, die ein Paket darin erfährt, folgt damit einer diskreten Verteilung. Kapitel 4 präsentiert die formale Analyse für deterministischen Verkehr sowie für Poisson-Ankünfte. Zu den Ergebnissen der Betrachtung von einer Schicht (Container) kommen die formalen Ergebnisse für mehrschichtige Betrachtungen (aggregierte Pakete). Für ausgewählte Szenarien zeigt das Kapitel die Ergebnisse der Simulation sowie der formalen Analyse.

Kapitel 5 wendet das analytische Modell in zwei verschiedenen Szenarien an. Die erste Anwendung ermöglicht eine Abschätzung des Anteils der Pakete, die außerhalb der Reihenfolge ankommen. Abschnitt 5.2 zeigt den formalen Beweis auf und illustriert die Ergebnisse für ausgewählte Szenarien. Weiter zeigt das Kapitel, dass die Abschätzung auch gültig bleibt, wenn die mittlere Verkehrslast zunimmt und die Paketverzögerungen des Models gleich bleiben. Die zweite Anwendung (Abschnitt 5.4) ermöglicht Studien von echten Protokollimplementierungen in Experimentierplattformen. Auf Basis dieses Modells wurde innerhalb dieser Arbeit eine Netzemulation implementiert, welche die Paketreihenfolge nach einem vordefinierten Muster verändert. Das Modul der Netzemulation wurde im Linux Kernel implementiert und verwendet die Netzemulation der Linux Foun-

dation. Dieses Kapitel beinhaltet die Beschreibung des Zustandsautomaten des Moduls sowie die Diskussion der Generierung von Zufallszahlen innerhalb des Linux Kernel, die einer bestimmten Verteilung folgen.

Die Parametrisierung des Moduls bedarf dem Wissen über die Verteilung der Paketverzögerung, um eine bestimmte Reihenfolgeänderung zu erzielen. Die Parametrisierung ausgehend von einer bestimmten Reihenfolgeänderung führt zu einem nicht-linearen Optimierungsproblem mit Randbedingungen. Abschnitt 5.3 stellt das Optimierungsproblem auf und wendet einen geeigneten Lösungsalgorithmus an, um die Verteilung der Paketverzögerung zu erhalten. Diese Verteilung der Paketverzögerung soll dann zum gleichen Reihenfolgemuster wie vorgegeben führen. Eine Verifikation des Moduls und ein abschließender Vergleich der Ergebnisse von Experimentierplattformen, Simulation und analytischem Model schließt das Kapitel.

Kapitel 6 betrachtet das analytische Model zur Reihenfolgeänderung und seine Anwendung in der Netzemulation. Es bewertet die Qualität der Lösung des Optimierungsproblems, die es ermöglicht, eine vordefinierte Reihenfolgeänderung nachzubilden. Die Reihenfolgemuster kommen dabei von einem Simulator für optische aggregatvermittelnde Netze. Für eine Bewertung der Lösung, verwendet das Kapitel zwei Hypothesentests und zwei graphische Methoden. Das Kapitel zeigt an mehreren Beispielen, dass die Lösung geeignet ist, um vorgegebene Reihenfolgeänderungen nachzubilden. Ein weiterer Punkt des Kapitels ist die Bewertung des Lösungsalgorithmus. Die Komplexität des Lösungsalgorithmus hängt dabei von zwei Faktoren ab. Der eine beinhaltet die Anzahl der Schritte bis eine Lösung gefunden wird, der andere die Zeit, die benötigt wird, um die Metriken für einen Punkt auszurechnen. Da es nicht möglich ist, die Komplexität der Funktionen anhand der mathematischen Beschreibung zu erfassen, wird bei der Komplexitätsabschätzung auf Zeitmessungen zurückgegriffen.

Das in der Arbeit vorgestellte Vorgehensmodellsmodell ermöglicht es nachweislich, Protokolle im Hinblick auf vorgegebene Reihenfolgeänderungen, die von Messungen oder Simulationen stammen, zu untersuchen.

Contents

List of Figures

List of Tables

Abbreviations and Symbols

Abbreviations

ACK	Acknowledgement
API	Application Programmable Interface
ARQ	Automatic Repeat Request
ATM	Asynchronous Transfer Mode
BRM	Basic reference model
CCDF	Complementary Cumulative Distribution Function
CDF	Cumulative Distribution Function
CSMA	Carrier Sense Multiple Access
DARPA	Defense Advanced Research Projects Agency
DCCP	Data Congestion Control Protocol
DCF	Distributed Coordination Function
DCN	Data Control Network
DIFS	DCF InterFrame Space
ECN	Early Congestion Notification
FDL	Fibre Delay Line
FEC	Forwarding Equivalent Class
FIFO	First-In First-Out
FPGA	Field Programmable Gate Array
GFP	Generic Framing Procedure
GII	Global Information Infrastructure

GMPLS	Generalized Multiprotocol Label Switching
GPP	Generation Partnership Program
GSM	Global System for Mobile communication
IAT	Inter-Arrival Time
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IMS	IP Multimedia Subsystem
ISDN	Integrated Services Digital Network
ITU-T	International Telecommunication Union - Telecommunication Standardization Sector
JET	Just Enough Time
JIT	Just In Time
LAN	Local Area Network
MAC	Medium Access Control
MPLS	Multiprotocol Label Switching
MTU	Maximum Transmission Unit
NGN	Next Generation Network
NIST	National Institute of Standards and Technology
OBS	Optical Burst Switching
OPS	Optical Packet Switching
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
OTN	Optical Transport Network
PBB	Provider Backbone Bridge
PDF	Probability Density Function
PDU	Protocol Data Unit
PMRA	Packet Reservation Multiple Access Protocol
PSTN	Public Switched Telephone Network
RAM	Random Access Memory

RED	Random Early Drop
RFC	Request For Comments
RIP	Routing Information Protocol
RTP	Real-time Transport Protocol
RTT	Round-Trip Time
SACK	Selective Acknowledgement
SCTP	Stream Control Transmission Protocol
SDH	Synchronous Digital Hierarchy
SDU	Service Data Unit
SIFS	Short InterFrame Space
SIP	Session Initiation Protocol
SMSS	Sender Maximum Segment Size
SONET	Synchronous Optical Networking
TAG	Tell and Go
TAW	Tell and Wait
TCP	Transmission Control Protocol
TDM	Time Division Multiplex
UDP	User Datagram Protocol
VCI	Virtual Channel Identifier
VLAN	Virtual Local Area Network
WDM	Wavelength Division Multiplexing
WLAN	Wireless LAN

Symbols

B	Random variable on the number of burst occurrences
D_k	Delay per abstract line k
D_l	Random variable of the delay of the located burst
d_l	Delay distance of the located burst
D_t	Random variable of the delay of the test burst
d_t	Delay distance of the test burst
$\mathsf{E}[T]$	Expected value of random variable T
E_B	Random variable of the burst extent
e_B	Burst extent value
E_P	Random variable of the packet extent
e_P	Packet extent value
F	Random variable of the position of the located burst
f	Position of the located burst
$f(p)$	Packet per burst distribution
H_B	Random variable of the number of proceeding bursts
H_P	Random variable of the number of packets
K	Random variable of the number of packet in the disordering network, $s > 0$
l_k	Abstract link k
L	Random variable of the number of packet in the disordering network, $s < 0$
m	Number of additional abstract links
n_r	n_r-reordering metric value
N_B	Random variable of the burst n_r-reordering metric
n_B	n_r-reordering value of bursts
N_P	Random variable of the packet n_r-reordering metric
N_{ppb}	Random variable of the number of packets per burst
n_P	n_r-reordering value of packets
n_{ppb}	Number of packets per burst

$\tilde{N}_P$	Number of reordered packets
$\tilde{N}_B$	Number of reordered bursts
n_C	Maximum packet count for packet count assembly
p_k	Probability to follow abstract link k
p_B	Burst reordering probability
p_P	Packet reordering probability
P	Random variable of the packet position in a burst
s	Sequence number
$s[i]$	Sequence number of packet i
$s'[i]$	Next expected sequence number at the destination
T	Random variable of the inter-arrival time
T_{IJ}	Random variable of the inter-arrival time between packets I and J
t_C	Inter-arrival time for deterministic traffic
t_O	Timeout value for timer based assembly strategy
$\mathsf{VAR}\,[T]$	Variance random variable T
v	Average number of flows per burst
W	Random variable of the waiting time in the disordering network
X_j	Random variable of the number of occurrences within slot j
x_j	Number of occurrences within slot j
$\vec{X}^{(i)}$	Vector of random variables of length i
$\vec{x}^{(i)}$	Vector of length i

α	Parameter of Pareto traffic model
Δ	Basic delay unit
γ	Fraction of reordered packets
κ	Mean number of packets per burst
k	Parameter of Pareto traffic model
λ	Traffic mean rate
φ	Convex function
ϱ	Parameter of the random assembly process

1 Introduction

Since the opening of the Internet for commercial and non-research users around 1990, the number of offered services and participating users exploded. After 20 years, more than 1.5 billion people participate in the Internet [2]. This corresponds to a growth rate of more then 300% since 2000. These users use a large number of services; for instance, file sharing, Voice over IP (VoIP) or instant messaging applications. In addition, video platforms, photo galleries and online communities become more and more popular. At the beginning on the Internet, centralized services, e. g., World Wide Web (WWW), dominated. After this initial growth, peer-to-peer applications, especially for file sharing, emerged. Now, with the popularity of social networks, centralized servers with some peer-to-peer traffic, e. g., instant messaging, regain dominance. Regardless of the changing service organization structure, the Internet traffic now grows about 50% per year [3]. This growth rate demands flexible, high performance, scalable and high-speed networks to handle these traffic masses.

Next Generation Networks propose parallel processing as one solution to handle further increasing traffic. Parallel processing refers to simultaneous handling of individual packets (datagram) locally in a node, as well as forwarding streams of information on different paths to the destination. Consequently, parallelism occurs within nodes and within networks. As a drawback, parallelism may change the original sending order of packets. For in-order processing, any parallelism requires mechanisms to maintain the original datagram order. If network technology does not foresee these mechanisms or if these mechanisms fail, parallel processing may change the original order. As applications and transport protocols assume in-order arrivals, they may misinterpret this change in the order as an indication of congestion and underperform. Besides intended packet reordering[1], network malfunction may also cause packet reordering.

The next sections widen the scope of packet reordering and give a clear distinction of pathological and intended reordering. Common to both reasons is the consequence on the protocols facing reordering, which discusses a separate section. Further sections give an overview on the methods proposed in the literature to evaluate the protocol performance facing packets arriving out-of-sequence. In general, these methods are insufficient to establish a direct link between the specific out-of-sequence arrival pattern and the experienced degradation of any protocol performance. This thesis closes this gap and provides in a dedicated section a new method to analyze protocol performance under packet reordering conditions. The last section provides the structure of the thesis.

[1] the term unintended reordering describes the process to force packets out-of-order

1.1 Packet Reordering Phenomena

In high-speed networks, packets arrive out-of-order because of several reasons. In general, these reasons belong to one of the following two classes: *pathological reordering* and *technology inherent* reordering. The first class exhibits reasons, where out-of-order arrivals occur as the result of malfunctioning devices or miss-configured network protocols, e. g., flapping routes. The second class includes technologies, which draw benefits and performance improvements from out-of-order arrivals. This section introduces both classes in depth and closes with an overview on the consequences of out-of-sequence arrivals on transport layer protocols and applications.

1.1.1 Pathological Packet Reordering

This section provides the background for unintended reordering in packet-based networks, like the Internet. Unintended refers to operations within the network, which were not planned nor foreseen, mostly related to some erroneous network functions. An out-of-sequence occurs, if a packet arrives later than another packet, although it departed earlier.

Jaiswal et al. classify in [4] packet reordering in three different types: packet retransmissions, packet duplication and in-network reordering. *Packet retransmission* occurs, if the network drops or severely delays a packet. In this case, protocols that offer a reliable connection service (e. g., transmission control protocol, TCP) may resend the missing packet. Although in this case, the packet is not lost, it may trigger resending the packet due to timeouts. The retransmission becomes unnecessary and wastes bandwidth. Besides the protocol performance of TCP, lower layers also show packet reordering. For instance, wireless networks show in general a worse connection quality than fixed networks. For reliability, they implement an ARQ (automatic repeat request) mechanism in the data link layer to retransmit erroneous packets. This retransmission may lead to out-of-sequence arrivals with respect to the upper layer protocols. Koga et al. as well as Arthur et al. study this impact in detail in [5, 6].

Today, routers operate at very high speeds providing complex functions like forwarding, policing, encrypting, etc. In most cases hardware but also software (e. g., management) realize this functionality. Both implementations may include bugs with respect to the agreed functionality. In rare cases, miss-configured or buggy router functions corrupt packets or release the same packet multiple times. The latter case leads to *packet duplication*, where at the destination node, the second packet triggers an out-of-sequence arrival [7]. Besides this, some routers are able to stop forwarding if they process routing updates. Buffered packets and new arrivals may interspersed, which may also change the order [8].

The internal structure of modern routers in general provides parallel processing stages to cope with high speed packet rates [9]. Govind et al. studies in [10] the packet reordering phenomena of network processors. They study the parallel microengines of the Intel IXP 2400 network processor, which share a common transmit buffer. Because of the parallel microengines and the efficient access to the shared buffer (without mutual exclusions)

packet reordering may occur. They find 60% of the 64 B and 14% of the 512 B packets reordered without any mechanism to maintain the order. Introducing mechanisms to maintain the packet order decreases these reordering rates to 33% and 2%, respectively. The cost of this reduction is degradation in packet throughput, due to the limited access efficiency to the buffer.

Bare et al. survey in [11] the reasons of parallelism in routers. The main reason is the increasing gap between link and processing speeds. The parallel processing within these high-speed architectures may cause packet reordering. Also per-packet load balancing increases the number of reordered packets [12]. In principle, they simulate the main architecture of Govid et al., with several parallel microengines accessing the same buffer. They apply the reordering-density metric of Piratla et al. of [13] and found that about 10% of the packets leave the router in a different order on an OC-384 (20 Gbit/s) interface. On an OC-768 (40 Gbit/s) interface, this amount increases to 17%.

Kandula et al. propose in [14] an alternative load balancing mechanism, which avoids any packet reordering. They assume parallel links between two devices, where the devices know about the different delays on these links (e. g., by measurement). With this knowledge, the load balancing device is able to distribute fractions of flows (flowlets) among these different paths without producing out-of-order arrivals at the destination. The device takes into account the relative distance between consecutive arriving flowlets as well as the link of the last flowlet. With this splitting algorithm, the authors ensure in-order arrivals at the destination.

In general, these parallel structures provide mechanisms to maintain the original packet sequence, although this is not possible at any time. In case of head-of-line blocking within a pipeline, subsequent packets may be processed out-of-sequence to regain throughput. Additionally to the local processing, neighbouring nodes may share parallel links for load-balancing issues. Here, reordering may happen, due to neglecting flow properties of the packets. Additionally, flapping routes from interior and exterior routing protocols are also reasons for temporal packet reordering [15]. Forwarding routes oscillate between certain forwarding destinations due to bursty traffic and load dependent weights of the routing protocol [8]. According to Jaiswal, the term *in-network reordering* summarizes these effects.

1.1.2 Technology Inherent Packet Reordering

Besides unintended pathological packet reordering, there exists a series of network technologies and protocols accepting packet reordering to achieve certain traffic engineering goals. Especially, the Future Internet initiative focuses on some of these proposals. In most cases, traffic engineering aims to improve the overall network performance on a certain network layer or the quality of service of individual or all traffic flows in a network. Multi-path routing represents one method to achieve better network performance but may exhibit packet reordering. Chapter 2 introduces this concept and the related consequences in detail. Network technologies may first assemble packets to larger containers before forwarding them to decrease packet rates (elaborated in Section 2.2). If the network technology reorders these containers, the carried packets also arrive reordered at

the destination. Thereby, the reordering pattern of packets and containers may differ. This scenario refers to a hierarchical reordering, i.e., reordering on multiple layers.

1.1.3 Effects of Packet Reordering

In general, the receiving node processes the received packets usually in the same order than theses packets leave the sender. The receiving protocol instance usually forwards the packets to the application layer, either in correct order including necessary retransmissions or in the order of arrival. In both cases, either the application layer or the transport protocol layer has to provide countermeasures to re-gain the original packet sequence. Mechanisms are re-sequencing buffers or congestion control algorithms of reliable transport protocols. In any case, this may have an impact on the node design as well as on the performance of the application layer protocol. Consequently, it is necessary to study the impact of packet reordering on upper layer protocols in hierarchical as well as non-hierarchical network scenarios. The next two sections discuss both aspects of packet out-of-sequence delivery.

1.1.3.1 Buffer Requirements at the Receiver

In case of out-of-sequence arrivals, transport or application protocols have to provide mechanisms to re-gain the original sequence for further processing. Any packet delay, causes a *head-of-line blocking* at the receiver. Although a number of packets have already arrived, processing cannot continue due to a missing delayed packet. In here, we assume no packet losses.

A general mechanism to cope with the arrival of out-of-sequence packets is to store the intermediate arriving packets in a re-sequencing buffer. This buffer stores all arriving packets and forwards the datagram to the application in correct order. Thereby, it searches the buffer for the next in-sequence packet. If this packet is missing or has not arrived yet, the buffer waits until its arrival. To avoid starvation and deadlocks in case of missing packets, a timer or packet counter defines the maximum time to wait for out-standing packets. For time-critical and interactive applications like voice and video streaming, this timer is inherently required to target the required service properties, e.g., parameters agreed by the ITU-T Y.1540 and Y.1541 [Y.1540, Y.1541].

For the node design, it is important to know the dimensions of the buffer size in an expected reordering scenario. A too short buffer size would cause unnecessary packet losses, while a too large buffer causes extra hardware costs. Another parameter is the initial value of the timer or packet counter. Usually, the application defines the timeout value to cover its needs and service requirements. Both, the buffer size and the parameterization of the timer, define the service quality experienced by the application.

1.1.3.2 Window Based Congestion Control

The Transmission Control Protocol (TCP) holds the largest share of transport protocols in the Internet. It offers a reliable communication service for applications and implements a congestion control algorithm. Besides other mechanisms, the dominant algorithm for congestion control is window based. The next paragraphs shortly introduce the congestion control algorithm and its relation to out-of-sequence arrivals.

The TCP standard of [RFC 2581] includes four different congestion control algorithms: slow start, congestion avoidance, fast retransmit and fast recovery. The slow start and congestion avoidance algorithms apply window based congestion control. They include a sender window (congestion window) and an advertised receiver window (receiver window). The minimum of both determines the amount of data, which the sender is allowed to send within one round. Starting with an initial window size, the sender increases the congestion window by every acknowledged packet until it reaches a threshold (slow start threshold). Then it enters the congestion avoidance phase and increases the congestion window linearly by one segment per round trip time.

The receiver acknowledges each incoming segment with the next expected segment number. Upon an unexpected or out-of-sequence arrival, it immediately re-acknowledges the outstanding segment. The sender receives this duplicate acknowledgement (dup-ack) and enters the fast retransmit state after reception of the third duplicate acknowledgement (IETF [RFC 2581] proposes three). In the fast retransmit state, the sender retransmits the missing segment and changes the sender window to the size of the slow start threshold.

The TCP congestion mechanism reacts with the fast retransmit and fast recovery mechanism, if the sender receives a certain number of dup-acks. The requested packet may be lost or severely delayed by several inter-departure times. If the packet still arrives after the last dup-ack, the sender has already taken countermeasures and decreased the congestion window. The standard implementation of TCP continues sending with a smaller window size, regardless of the non-loss situation. Besides this, the delayed packets also impact the measure of the round trip time, which enables TCP to detect changed network conditions and to set the retransmission timeout [RFC 1323].

Up to this point, both, the sender and the receiver are not able to distinguish the difference of both events, *packet loss* and *out-of-order packet arrival*. The resulting behaviour of TCP is comparable in both situations. The quantification of the latter one depends on the dup-ack threshold and the traffic characteristic of the sending station and the intermediate queuing network. Several countermeasures are available to make TCP robust against out-of-sequence packets. One is to increase the duplicate acknowledgement threshold to a larger value. Measurements and learning capabilities of the sender may determine the new threshold depending on the extent[2] of the out-of-sequence arrivals. It is a trade-off between the time to react on packet losses and the time to wait for late packets. The parameterization of both is either static or determined dynamically at runtime. The following improvements to TCP exploit the dynamic adaptation of the dup-ack threshold.

[2] difference between actual and original packet position in a packet flow

1.1.3.3 Out-of-sequence Aware TCP Versions

In the literature, many researchers have addressed the problem of TCP facing reordered packets. Due to the reasons introduced in Section 1.1, research tries to make the most important transport protocol TCP robust with respect to unexpected out-of-sequence arrivals. Among others, literature proposes three major extensions to TCP to fight packet reordering, reordering robust TCP (RR-TCP, [16]), reordering detecting TCP (RD-TCP, [17]) and reordering persistent TCP (TCP-RP, [18]). The next paragraphs introduce the basic concepts, while Ka-Cheong Leung et al. survey in [8] further extensions for TCP facing reordering.

The reordering robust TCP maintains a scoreboard at the sender for every sent packet and their acknowledgements. These records provide the information on outstanding packets, as each missing packet produces a gap in the sequence of acknowledgements. The sender monitors these gaps of outstanding acknowledgements using a histogram. Based on the histogram data, the sender adapts the dup-ack threshold to compensate a defined amount of reordered packets.

RD-TCP includes intermediate gateways to distinguish lost packets from reordered packets. Each gateway monitors the received and dropped packets. If these gateways receive an acknowledgement packet, they check if they had dropped this packet earlier. If so, they indicate this by setting a special drop bit in the acknowledgement. Otherwise, packets arrive out-of-order. The special drop bit indicates the sender that a packet is dropped and that the sender can resent the missing packet immediately. If there is no drop bit indication, the packet will arrive later because of reordering. In this case, the sender waits for even more duplicate acknowledgements than the original threshold.

TCP-PR avoids the misinterpretation of duplicate acknowledgements by not considering them at all. TCP-PR only considers timestamps to detect reordering and to distinguish them from packet losses. The sender keeps two lists, a list indicating packets for sending and a list of unacknowledged packets. The arriving acknowledgements remove the corresponding items from the sending list. To detect lost packets, the sender keeps a timestamp of the sending time. It considers unacknowledged packets beyond the threshold as lost and triggers their retransmission. Unacknowledged packets below the threshold may arrive later because of severe delays. The authors propose an algorithm to adapt the threshold to the actual network conditions. The principle mechanism compares to RR-TCP where the criteria of lost packets is segment based, while in TCP-PR it is time based.

The Future Internet initiative also discusses extensions to TCP. Their focus is on a micro-flow level, which actually represents a part of a flow. Each micro-flow keeps the packet order on its path. Different micro-flows may follow different paths, not necessarily maintaining any order between them. As the implementations are in an early state or involve the network elements, there a doubts for an early deployment of these implementations. The discussions of the Future Internet are still ongoing. There, the outcome is not clear.

1.2 Methodologies to Study Packet Reordering

The previous section showed that packet reordering violates the paradigm of in-order sending and processing of packets. More severe, packet reordering may harm the protocol performance or requires additional mechanisms to re-gain the original sequence. The studies in the literature on packet reordering are therefore twofold. One branch classifies and quantifies the reordering pattern by measurements using different reordering metrics. The other branch tries to evaluate the effect of packet reordering on different protocols, mainly TCP and UDP. This section introduces the simulation, formal and experimental methodologies to determine the protocol performance of the second branch. The drawbacks of these methodologies build the motivation for the proposed methodology of this thesis. Chapter 3 refers to the first branch, the reordering metrics.

1.2.1 Simulation Studies

Integrated simulation studies dominate the research to study protocol performances with respect to out-of-order packet arrivals. These studies mostly apply a model or propose a model for both, the underlying network as well as the protocol to study. In most cases this includes optical burst switched networks and the transmission control protocol [RFC 2581]. They model the underlying OBS network as well as TCP or create a simulation model within a simulation environment. Performance metrics are the throughput/goodput of TCP, in rare cases also more detailed metrics on the number of acknowledges or window sizes. The main driver for these studies is the impact of packet and burst losses on the TCP performance [19–24].

Literature does not study the impact of burst reordering on TCP and other upper layer protocols in such detail. Callegati et al. introduce in [22,25] a burst reordering framework for a WDM network, but their reordering definition misses the exact link between the reordering characteristics and the related TCP mechanisms. In [26], Perelló et al. quantify by simulation the impact of contention resolution schemes on optical burst reordering and estimated the TCP performance. They measure the amount of optical burst reordering in the same order of magnitude as the burst loss probability. These results emphasize the necessity for a detailed investigation on reordering in optical burst switching networks. Schlosser et al. analyze in [27] the impact of burst deflection by intensive simulations. They apply an integrative TCP over OBS network model including only a single alternative path. Thus, the results are not representative for a network wide analysis with a different delay distribution between source and destination node.

The number of parameters of this kind of simulations is in general very large as OBS as well as TCP shows a large number of parameters and configurations. TCP has several options [RFC 2581] as well does OBS allow a wide flexibility. Section 2.4.2 will introduce the main parameters of OBS networks. Besides this, general parameters like network topology, network load and network dimensioning further increase the parameter space.

As the applied simulation models integrate protocol and network technology, it is in general hard to obtain the packet sequence and to distinguish the impact of out-of-sequence

arrivals from other network properties, the protocols may depend on, e. g., round-trip time. They lack the direct relationship between the packet reordering sequence and the protocol performance.

1.2.2 Analytic Studies

In contrast to the simulation studies, literature shows only very few analytic studies on the impact of out-of-sequence arrivals on the protocol performance. The dominating research field is the investigation on the TCP performance with respect to packet losses. The analytic TCP models are highly relevant for these kinds of studies. At first place, one has to mention the work of Padhye et al., who propose in [28] an analytic TCP model, which takes into account the timing constraints of TCP. They show that their model reproduces real TCP Reno connections very well. Applying Padhye's model, Detti et al. study in [19, 29] the interactions between optical burst switching networks and TCP. They give results for the burstification delay with respect to burst losses. They optimize TCP throughput and find aggregating several connections per burst beneficial. Besides this work, Du et al. propose in [30] an acknowledgement mechanism for optical bursts. The TCP performance should recognize and benefit from this acknowledgement for an improved throughput performance. Their analytic model matches the simulation results well.

Analytic models on the protocol performances of TCP or other transport protocols are very rare in the literature. The major reason is the huge complexity of these protocols and their large number of flavours. Consequently, any analytic model provides basic statements with respect to selected scenarios. These analytic models mostly focus on selected mechanisms, where they neglect the interaction to other mechanisms for simplification. They enable the understanding of principle interrelations, but can not evaluate the whole system with its environment at once.

1.2.3 Experimental Studies

In [31], Bennett et al. study carefully the impact of reordering on TCP. Therefore, they first quantify the amount of reordering in the network and second try to conclude on the TCP performance. They measure packet reordering in a backbone network and classify packet reordering in forward and backward direction. They trace the packet sequence numbers and analyze the trace offline. Additionally, they remove lost packets by renumbering the remaining packets in the stream. Besides the quantification of reordering in the network, they provide a detailed discussion on the TCP behaviour facing the measured packet sequence. This work provides detailed findings on the occurrence of reordering in the network but the conclusions on TCP base on simulations and standard documents rather than on real protocol implementations.

Additionally, Paxson measures in [15] the amount of reordering, loss, duplication, delay and corruption in extensive studies. Based on theses measurements, he evaluates the performance on TCP. For evaluation, he refers to the IETF standard documents for

TCP. As the IETF standards in general leave flexibility in implementation, the resulting implementations may show wanted or unwanted side effects. The same argument as before applies. Standards and real implementations may differ. To evaluate the whole system, the system itself needs to be analysed.

1.2.4 Proposed Pethodology

The approach proposed in this thesis enables a comprehensive protocol performance evaluation facing out-of-sequence arrivals. The approach clearly distinguishes between experienced out-of-sequence pattern and the protocol performance. Consequently, it enables a deep understanding of protocol mechanisms resulting from reordering pattern. The proposed methodology to study the performance of protocols covers the following steps (cf. Figure 1.1). The numbers in the figure correspond to the number in the enumeration. The right part of the figure denotes per step the applied methodology.

Step 1 depicts the problem statement: the focus of interest is the end-to-end protocol performance with respect to out-of-sequence packet arrivals. Thereby, the protocol may be any transport or application layer protocol in the end systems. The out-of-sequence packet arrivals occur because of special network features that Chapter 2 discusses in detail. These special network features may be present in real or future networks.

Step 2 models the expected traffic characteristics of the protocols in the end systems. It further models the network conditions that show packet reordering. Simulations obtain the reordering patterns of these network conditions, which enable packet reordering. In case of packet assembly, it obtains reordering patterns for both layers, flat (burst reordering) and hierarchical (packet reordering). This step applies reordering metrics to quantify the experienced reordering pattern. Chapter 4 introduces the applied reordering metrics in detail and discusses the advantages and disadvantages with respect to other metrics proposed in the literature.

Step 3 applies the obtained reordering pattern together with an analytical reordering model to reproduce the reordering pattern obtained from simulations. Chapter 4 introduces the novel analytic reordering model. The model is able to re-engineer the experienced network delay distribution starting from a given reordering pattern. This step leads to an optimization problem described in Section 5.3.1. The re-engineered network delay distribution serves as an input parameter for a network emulation scenario.

Step 4 shows the network emulation scenario, which emulates the desired packet reordering using a novel network emulation module. Therein, the analytic reordering model provides the parameters for this module. The network emulation extents a Linux kernel module and manipulates the outgoing buffer to reflect the destined reordering pattern. Chapter 5 describes this application scenario for the reordering model including the testbed and the module.

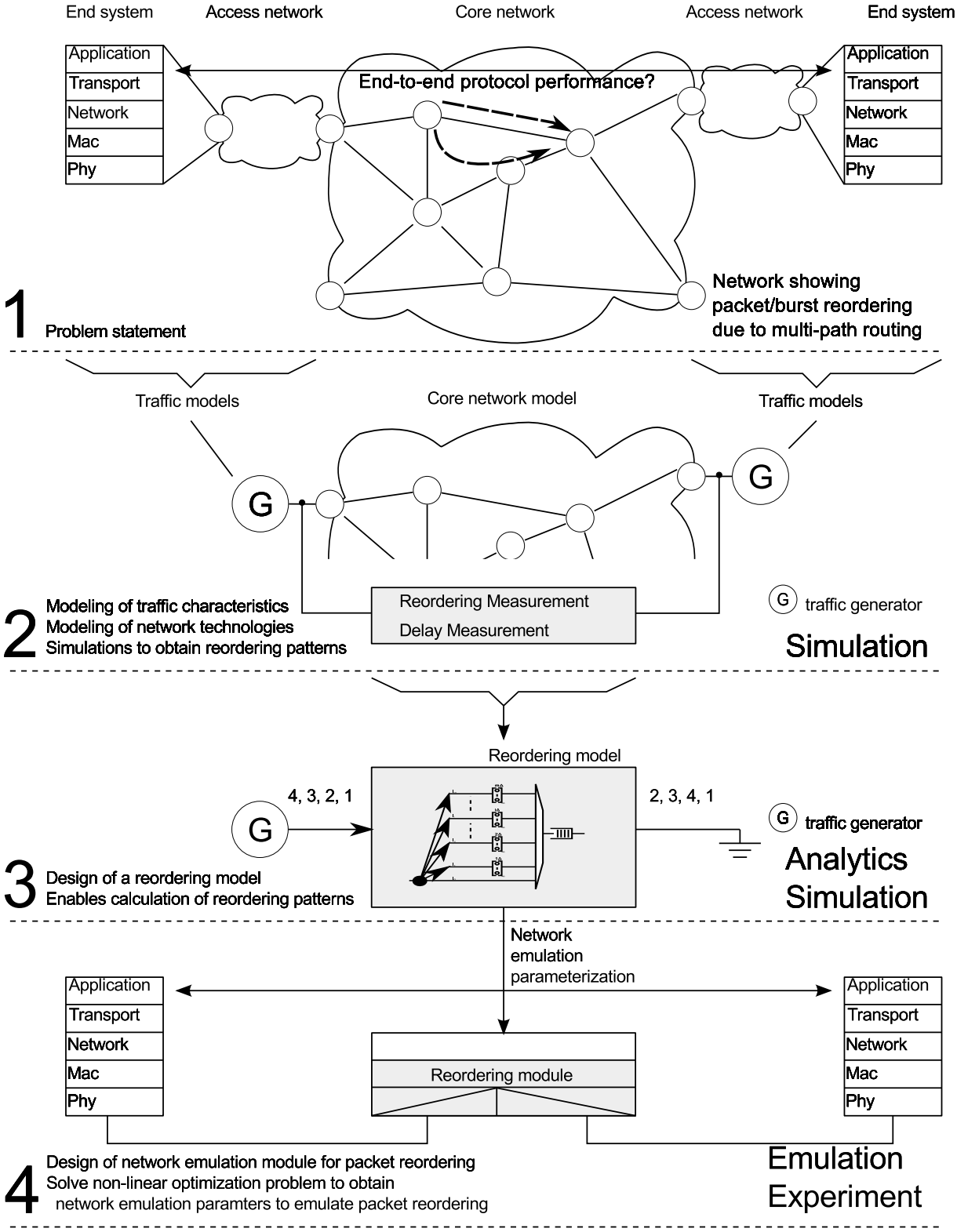


Figure 1.1: Proposed methodology

1.3 Structure of the Thesis

Each of the following chapters is dedicated to one of the previous steps. The individual steps of the proposed methodology (cf. Figure 1.1) require several sub-steps to solve. These provide the structure of this thesis.

Chapter 2 introduces the properties and concepts of Next Generation Networks and their relation to the global information infrastructure project proposed by the ITU-T. Besides this general introduction, this chapter also provides an overview on current trends in these networks. Two selected features of Next Generation Networks are multi-path routing and packet assembly. Chapter 2 motivates both concepts and imposes network architectures implementing these features. One representative of these Next Generation Networks implementing multi-path routing and packet assembly is optical burst switching. One section introduces this architecture as a representative for Next Generation Networks. Besides the potential increase in network performance, both features cause packet reordering either on a single layer or on different layers at the same time.

Recording of out-of-sequence packets at the destination requires metrics to quantify the amount and characteristic of packet reordering. Chapter 3 provides an overview, evaluation and the selection of reordering metrics provided in the literature. It discusses the individual approaches with respect to robustness, implementation complexity and relevance for protocol investigations. The closing evaluation chooses the reordering metrics of the IETF because of their easy implementation, low complexity and standardized approach. The reordering metrics include the definition of reordering, a measure for the reordering extent and metrics indicating the impact on the Transmission Control Protocol (TCP).

Chapter 4 introduces an analytic reordering model to abstract the reordering pattern of a certain scenario. It provides the background to determine reordering metrics for two selected traffic characteristics in an arbitrary delay environment. One traffic characteristic shows constant inter-arrival times, the other shows Poisson arrivals. Further, Chapter 4 provides two worst-case approximations. The first approximation shows for hierarchical packet reordering that the reordering pattern is worst, if there is exactly one packet in every container burst. The second approximation shows that constant inter-arrival times may lead to the largest number of reordering packets at the destination compared to any other traffic characteristic. This chapter formally proves that the reordering pattern of constant traffic represents an upper bound for any other traffic model. Besides this, it introduces a simulation environment, which backs-up the findings.

The application of the reordering model in a network emulation environment requires the inverse functions of the equations of the analytic reordering model. This leads to an optimization problem solved in Chapter 5. The solution of the optimization problem leads to parameters of an extended network emulation tool. This chapter shows the principle architecture and concepts of the widely deployed Linux traffic control suite, which enables network emulation on IP layer. This thesis developed a new module, which realizes the exact predefined reordering pattern. Chapter 5 highlights the characteristics of the reordering module and shows its principle operation. Thereby, the reordering model

provides both, analysis of protocols showing closed-loop behaviour (TCP) or open loop (UDP) characteristics. Besides the network emulation, the reordering model also provides the parameters for further protocol simulations.

Chapter 6 evaluates the proposed methodology to study protocol performance facing packet reordering. Starting from the reordering pattern from simulated OBS networks, the solution of the optimization problem provides the parameters for the network emulation and simulation setup. This chapter evaluates experienced reordering patterns from optical burst network simulations with respect to the results obtained by the new approach. It applies statistical methods for a comprehensive evaluation and measurement techniques to evaluate the complexity of the solver for the optimization problem. Chapter 6 shows that the proposed methodology including the analytic reordering model provides a reasonable methodology for investigations on packet reordering in network emulation environments.

The concluding Chapter 7 provides a summary of this thesis. It reviews shortly, the contribution of this thesis regarding the methodology, the analytic reordering model as well as network emulation framework. Further studies may concentrate on the improvement of the solver as well as the reordering module of the testbed environment.

2 Selected Features of Next Generation Networks

Due to the dominance of the IP technology, future networks remain and will be packet-based in the networking layer. A common term for these networks is *Next Generation Networks*. This section introduces Next Generation Networks and includes the terminology and definition as well as the background of this term proposed by the ITU-T. It further imposes packet- and circuit-switched networks and highlights their major differences. The section covers the ideas of the global information infrastructure project and the relation to the definition and architecture of Next Generation Networks. The last section to this introduction highlights trends and challenges of current networks towards Next Generation Networks.

One challenge of Next Generation Networks is the steadily increasing data volume. Additionally, the data rate increases and as the packet sizes remain constant the packet processing rate increases. For this challenge, literature discusses two major concepts to sustain high network quality of service. Thereby, network quality of service includes low jitter, losses and no congestion. The two concepts are multi-path routing and packet assembly. Multi-path routing spreads the traffic along several parallel paths to the destination and consequently reduces the probability of congestion. Packet assembly reduces the packet rate and network load in these networks by assembling packets to larger containers. The remaining sections motivate and introduce both concepts in detail. Additionally to these concepts, each section gives examples of network technologies applying one or the other concept. This section also presents the concepts as well as the network architecture of optical burst switching as a representative for Next Generation Networks implementing multi-path routing and packet assembly.

2.1 Introduction to Next Generation Networks

This section introduces the Global Information Infrastructure project of the ITU-T and its relation to Next Generation Networks (NGN). This section also imposes the reference model for NGN and provides an overview on the capabilities of NGN. A short reference to multi-layer networks, which are an inherent property of NGN, as well as trends within these networks end this section.

2.1.1 The Global Information Infrastructure Project

The Global Information Infrastructure project (GII) of the ITU-T is an evolutionary concept towards a global information infrastructure [Y.100, Y.110]. This information infrastructure enables people to use, share and manage securely a set of information services, which supports a boundless number of services and embraces all modes of information (voice, video and data). The infrastructure will include interactive, broadcast and other multimedia delivery mechanisms. Thereby, the GII is neither a single architecture nor a reference network but bases upon a seamless federation of interconnected, interoperable communication networks and information processing equipment. The communication networks themselves may have own reference networks, of which the ITU-T defines the interfaces for interconnection.

From an industrial view point, the GII is the centre of convergence of different networks, namely the computer/information (e. g. Internet), the consumer/entertainment (e. g. cable TV) and the telecommunication networks (e. g. telephone). A user sees the GII as a common platform, which offers services previously offered by separate dedicated networks. The GII provides *triple-play* capabilities.

2.1.2 Next Generation Networks

This section gives a short and condensed definition of Next Generation Networks defined by the ITU-T in [Y.2001, Y.2011] and their relation to the GII. As the NGN proposes packet-based networks in the networking layer, the first section introduces the basic differences of packet-switched and circuit-switched networks.

2.1.2.1 Multiplexing Paradigms

This section briefly discusses the different multiplexing schemes in packet- and circuit-switched networks. In general, there are two types of multiplexing schemes, i. e., channel multiplexing and block multiplexing.

Channel multiplexing associates each connection with a corresponding transmission channel. The transmission channel shows fixed bandwidth or bitrate. Sharing of this transmission channel follows the channel multiplexing principle with the following flavours: space division, frequency division, time division, wavelength division and code division multiplex. These multiplexing schemes are highly beneficial for connections, which show a constant bitrate.

As the largest amount of data in the networks belongs to variable bitrate connections, the application of block multiplexing is more beneficial. In block multiplexing, the signal consists of blocks of constant or variable size, e. g., packets, cells, frames. Each block holds the destination address or a common channel identifier in the header, e. g., shim header in MPLS or Virtual Channel Identifier (VCI) in ATM. These blocks may be organized connectionless or connection-oriented. They share a common channel, e. g., interface, in the order of arrival. Access is granted to the common channel statistically. Literature

refers to the conjunction of traffic model, buffer occupancy, interface capacity and number of sources as *effective bandwidth* with the following selected publications [32–36]. The theory of effective bandwidth enables for a given traffic model to determine the third parameter if two are known.

Summarizing, if the information occurs in blocks with variable bitrate, statistical multiplexing of blocks leads to efficient resource utilization, i. e., local and metropolitan area networks. If information occurs with constant bitrate, channel multiplexing is more beneficial, e. g., core networks with aggregated data.

2.1.2.2 Definition of Next Generation Networks

While the GII defines in a non-technical way the communication platform of the future, it leaves out implementation aspects. The Next Generation Networks are a first step towards the realization of the GII project. In a concise way, inter-operable NGN are the technical realizations of the GII [Y.2001, Y.2011].

The major task of the NGN activity is to ensure interoperable network elements to support applications globally across the NGN. According to the ITU-T, Next Generation Networks are packet-based and provide telecommunication services, which are independent of the underlying transport network technology. The user may choose the service from competing providers. Further, NGN may support a generalized mobility, which enables users to access the NGN on changing locations or technical environment with or without service continuity. Fundamental characteristics of NGN include converged fixed and mobile networks, support of multiple access technologies, and interworking with legacy network technologies using open interfaces. The total list of NGN characteristics provides [Y.2001].

2.1.2.3 Capabilities of Next Generation Networks

NGN shall provide the capabilities to enable creation, deployment and management of all kind of services. These include any kind of media (audio, visual, audiovisual) sharing all kinds of encoding schemes and data services to realize any kind of service. This includes conversational, unicast, multicast and broadcast services requiring a different quality of service, e. g., delay restrictive, delay tolerant, real-time, non-real time, best effort. Besides QoS, these services may use different kinds of bandwidth requirements beginning from kbit/s up to bandwidth hungry applications of Mbit/s and Gbit/s. Another aspect related to services is service customization, i. e., enabling users to customize services by application programmable interfaces (API) to support management, provisioning and support of services.

An inherent property of NGN is the decoupling of service and transport network infrastructures. With this separation, both infrastructures can evolve separately and independently. Therefore, NGN clearly distinguish between functions for services and functions for the transport network. NGN allows the provisioning of existing and new services independently of the network infrastructure.

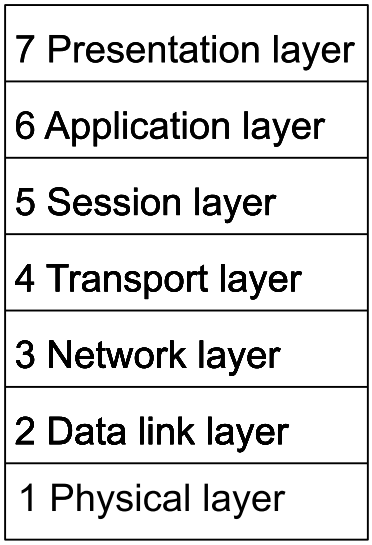


Figure 2.1: OSI BRM

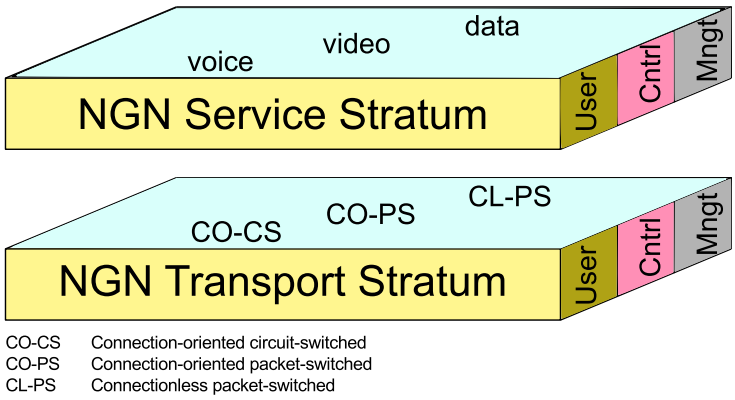


Figure 2.2: NGN BRM

The tasks of the function entities within NGN are to control policies, sessions, media, resources and service delivery. These functions may be distributed in the network using open interfaces for communication. For inter-operation between different network operators or existing networks like ISDN (Integrated Services Digital Network), GSM (Global System for Mobile communication) and PSTN (Public Switched Telephone Network), NGN require interworking functionality, i. e., gateways. For NGN, this includes the support of different terminal equipment like telephone, fax, mobile, etc. This aspect also includes the migration from the voice networks to NGN maintaining QoS, security and interoperability. For the wired access technologies, [Y.1001] provides migration strategies towards an IP based NGN platform.

In the last years, the Internet protocol became popular in local as well as wide area networks. IP offers a thin interface between the transport layer technology and the upper layer protocols, i. e., transport and application layer protocols. Because of this reason, the ITU-T recommends IP as the adaptation protocol to legacy application and services. Consequently, the Internet protocol will further dominate the networks at least as an adaptation layer.

2.1.3 Architecture of NGN

The ITU-T defines and standardizes the architecture of a NGN in [Y.2011]. This includes a generic reference model of two strata, the service and the transport stratum. Vertical planes for data, control and management subdivide each of these strata. The special case of multiple layers in the transport stratum requires special focus and mechanisms for inter-operation. This section introduces the NGN architectural model at first and second discusses briefly multi-layer issues, an inherent property of future transport networks.

2.1.3.1 Basic Reference Model

Originally, the ITU-T X.200 proposed the original seven layer OSI Basic Reference Model (OSI BRM, cf. Figure 2.1, [X.200]). It became a synonym for layered architectures and shows generic design principles for open systems. A set of developed OSI protocols match this layering, but did not succeed in wide applicability. With the rise of packet-based network technologies and the future requirements to these networks, the OSI BRM became inflexible for a reference model to the NGN [Y.2011]. The major arguments are: the number of layers is not seven, the functionality per layer does not match, and the protocols are not OSI compatible. This means that the OSI BRM applies for all layered architectures, but the functionality per layer may mismatch or is distributed among different layers. To provide more flexibility, the ITU-T proposed a new reference model for NGN. The model distinguishes inherently between the functionality for services and transport of data, i. e., the *service stratum* and the *transport stratum* (cf. Figure 2.2).

The transport stratum represents a set of functions to transport any digital information from one geographic point to another. With respect to the OSI BRM, this corresponds to the functionality of layers 1 to 3. This includes connectivity in terms of user/user, user/service and service platform/service platform. Network technologies within the transport stratum may apply connection-oriented circuit-switched (CO-CS), connection-oriented packet-switched (CO-PS) or connectionless packet-switched (CL-PS) technologies. In NGN, different network technologies may coexist forming multi-layer networks. Section 2.1.3.2 discusses this aspect in more detail. ITU-T Y.2011 [Y.2011] proposes IP as the preferred protocol to support legacy, present and future services. The application functions of the service stratum include any user services such as voice (e. g., telephone, fax), video (e. g., TV, movie, video clips) or data services (e. g., web, file, email) or any combination of these.

The structure within each stratum shows a user, control and management plane orthogonal to the horizontal layering (cf. Figure 2.2). The user plane represents the user in the service stratum and the data transport in the transport stratum. Functions of the *control plane* in the service stratum include, among others, user authentication, identification, service admission control, and application server functions. Functions in the transport stratum may be network admission control, network resource/policy control and dynamic connectivity provisioning. Functions of the management plane, according to ITU-T M.3050.x and M.3400, classify the tasks in fault, configuration, accounting, performance and security management [M.3050.0, M.3400]. This functionality usually applies for the transport stratum. According to the ITU-T, the relation to the service stratum is for further study.

2.1.3.2 Multi-layer Networks

One inherent property of the transport stratum is the flexibility to interconnect heterogeneous transport network technologies, i. e., several network technologies may coexist at the same time. Examples of interconnecting networks are OSI and G.805/G.809 [G.805, G.809] based networks as well as access and core networks. Besides this, network technologies

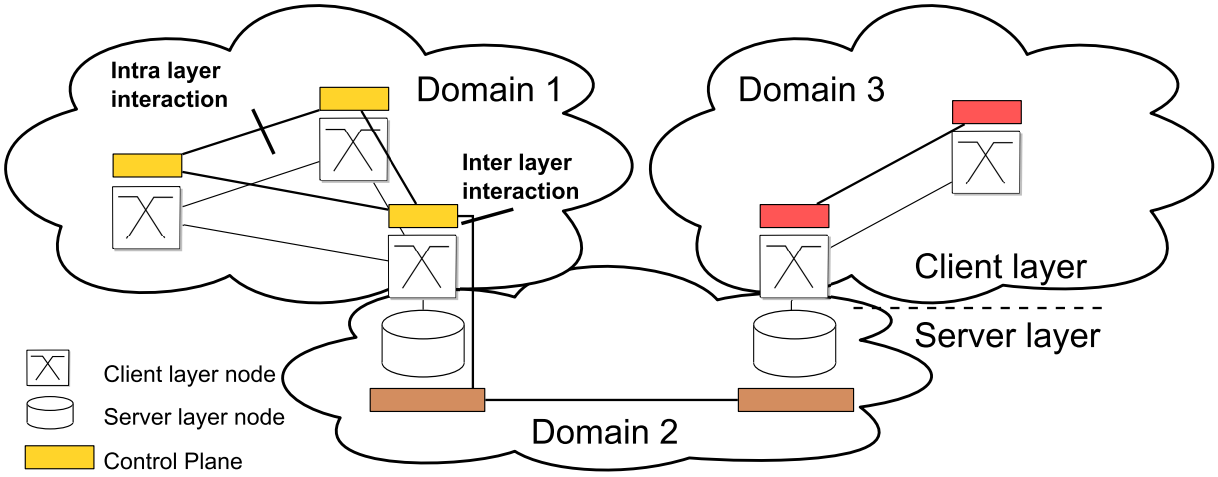


Figure 2.3: Multi-layer multi-domain scenario

may show multi-layer characteristics, e. g., IP/WDM, IP/Eth/WDM, IP/SDH/WDM or IP/Eth/SDH/WDM. This coexistence of different network technologies requires coordination. Within this section, the focus shifts to the control of these networks rather than the technical or economical advantages or disadvantages of these solutions.

In general, cooperating networks require one controlling entity per network. A distributed control plane may serve the purpose of the controlling entity. Both control planes cooperate by exchanging information, which may include reachability, resource and topology information. Figure 2.3 depicts the coordination activities of a typical multi-layer network. In the upper part, the data plane consists of a client layer (e. g., IP or Ethernet). The lower part shows the server layer, e. g., Synchronous Digital Hierarchy (SDH, [G.803]) or Wavelength Division Multiplex (WDM, [G.694.2]). It depicts the control planes and the interaction of both networks. Each network node holds a control plane module, which is inter-connected to other control plane nodes forming a data control network (DCN). The figure also highlights the inter-layer and intra-layer interaction. The former includes the definition of external interfaces, which requires the extension of existing protocols. The latter includes internal interfaces to enable the existence of multiple networks at another layer.

2.1.4 Trends in Next Generation Networks

This section highlights some of the challenges of future networks. It restricts itself to three objectives derived from ongoing discussions within the IETF, ITU-T and European projects. This list is neither complete nor exhaustive but represents major discussions in the research community motivating the studies presented in this thesis.

The first objective of network operators is to decrease the number of IP routers in the networks for scalability and cost reasons. The German project 100 GET as well as the European project Strongest exploit this idea. The second objective enhances network feedback for the end-systems to exploit network resources in a more efficient way. Besides this, the IETF proposes several standards to realize more intelligent networks [RFC 4782,

RFC 3168,37]. As a last objective, the control of these networks becomes more important. Reasons for this objective are mainly the cost reduction by automated network control and simple management. The German project Viola, as well as the European projects Terena, Nobel and Strongest focuses on these control aspects. Dragon is one example for a US project concentrating on the same issues. The next sections discuss each of these objectives in detail and provide the background for the scenario of this thesis.

2.1.4.1 Decrease of IP Routing Functionality

Current networks rely on IP technology in the networking layer because of the dominance of the Internet and the IP based LAN technology. As the end-systems and access networks apply IP technology, core networks migrated towards an IP enabled routing and switching. Rising line rates and meshed topologies require powerful IP routers with multiple gigabit line rates and several tens of line cards. The realization of these devices became extraordinary difficult. This significantly raised the capital and operational expenditures (i. e., power consumption). The operators try to escape from this cost-increasing spiral by reducing the number of IP routers in their networks. One option is to move routing functionality to the network edge, where the packet rates are lower and the devices are cheaper. Additionally, even in the metro network, current trends try to replace IP technology and its functionality by appropriate solutions on layer 2. Examples of an extensive research on layer 2 technologies are Optical Packet/Burst Switching (OBS, [38]), Multi-Protocol Label Switching (MPLS, MPLS-TE, MPLS-TP, [RFC 3031]) and the Ethernet based Provider Backbone Bridge (PBB, PBB-TE, [802.1ah, 802.1ay]) proposal.

2.1.4.1.1 Optical Packet/Burst Switching

In parallel to electronic packet-switching, optical packet-switching (OPS, [39]) realizes the same idea on the optical layer (cf. Section 2.1.2.1). Optical packets may enable highly flexible network architectures with very low power consumption and very high data rates. As the realization of OPS is currently not feasible, optical burst switching (OBS, [38]) decouples header processing from payload switching. It processes packet headers electronically, while switching payload optically. Section 2.4.2 dedicates itself to this topic and gives detailed insights.

2.1.4.1.2 Multi-Protocol Label Switching

Multi-protocol label switching (MPLS, [RFC 3031]) technology resides on layer 2.5 between IP and the underlying link layer protocol. It adds shim headers to the packet, which represent the basis for switching decisions in MPLS routers. The header includes a label, which serves as an identifier for packets belonging to the same path. MPLS capable network nodes are able to add, remove, switch or stitch labels providing a hierarchy of label switched paths.

MPLS implements a label-switching algorithm and does not require a longest matching prefix algorithm as IP. Thus, the implementation of the switching algorithm is cheaper. Besides this, its connection-oriented property enables traffic-engineering capabilities.

Traffic engineering extensions add a -TE to the abbreviation and denote the capability to include traffic-engineering objectives in routing and signalling protocols (MPLS-TE, [RFC 3564]). As the IETF proposes MPLS for the Internet community, the ITU-T adapted MPLS to the requirements of network operators. In a joint study group, IETF and ITU-T focus on these -TP (transport profile) extensions (ITU-T study group 15). The major changes include an adaptation to the already deployed SDH and OTN platforms and provide an interface to already deployed MPLS networks and devices.

2.1.4.1.3 Provider Backbone Bridge

In addition to the activities on the IP/MPLS layer, the IEEE focuses on an efficient Ethernet based solution for metro and core networks. They foster the provider backbone bridge (PBB, [802.1ah]) initiative, which enables Ethernet for application beyond local area networks. The limitations of Ethernet in the core network are mainly scalability issues of the flat address space, the inflexible routing functionality of the spanning tree protocol [802.1D] and the lack of appropriate management interfaces. Besides this, in Ethernet, each end-system owns an address, on which the provider has no influence. The distinction of customer addresses and provider addresses is an inherent problem of Ethernet.

One solution to this is the separation of customer from provider address spaces. In PBB, the Ethernet frame format shows additional address fields within the operator network (i. e., MAC in MAC encapsulation). The PBB edge devices are responsible to encapsulate the customer packets for the provider network. Besides this, they are also responsible to provide the information, where which device is located to avoid unnecessary broadcast information in the provider network. PBB with traffic engineering extensions -TE solve this problem. PBB-TE adds packet-switching circuit-oriented capabilities to the original PBB.

The IEEE addresses in [802.1ag] the issue of network management of these Ethernet-based networks. It defines special control messages to check link connectivity as well as the identification of bridges along the path. The survey of Sommer, Gunreben et al. provides in [40] a deep understanding of the evolution and variations of Ethernet in access, metro and core networks.

2.1.4.2 Enhanced Network Feedback

IP based networks, especially the Internet, are grown systems, which migrated from the DARPA net of 1972. This network showed small capacity links and a rather small number of users with low-performance end-systems. For this environment, researchers designed the protocols. As in general, the network belongs to a single authority, i. e., network provider, the hardware in the core changed with time. Besides this, network providers

enhanced their networks with additional mechanisms. In contrast to this, the protocols run in end-systems under the authority of each individual user. Consequently, it is difficult to disruptively change these protocols and adapt, e. g., to new network features. This especially relates to the TCP protocol, which shows different flavours inhomogeneously distributed in the network.

Given this scenario, the communication process between end-system protocols and network property relies on a very basic signalling mechanism. In the past and in today's networks, the mechanism is still the same, i. e., dropping packets to indicate congestion. This network operation shows two principle problems. First, the network is only able to resolve congestion instead of avoiding it in advance. Second, the protocols in the end-systems have no information on the available capacity in the network and adapt to the optimal utilization in a long-lasting procedure, which always ends in congestion, cf. Section 1.1.3.2.

Current discussions in the community consider a signalling mechanism to avoid the first problem, i. e., early congestion notification (ECN, [RFC 3168] and Re-ECN, [37]). One potential solution of the second problem is Quick-Start TCP, enhances the network to indicate available bandwidth to the end-system [RFC 4782]. As both concepts involve the protocols in the end-systems, the challenging question remains how to migrate current networks and protocols.

2.1.4.3 Integrated Control Plane

The heterogeneity of the network infrastructure requires a common control framework for easy interoperation and cooperation among different networks and network technologies. While the strict layer separation of the past favoured also a strict separation of the control layer, ongoing discussions in the research community show promising signals for an integrated control plane based on the Generic Multi-Protocol Label Switching (GMPLS, [RFC 3945]) framework. This framework extents the MPLS framework towards optical, TDM and L2 technologies and provides capabilities for efficient interworking of signalling and routing tasks.

The routing task provides information on the network topology and represents the basis for any path calculation. The signalling task reserves resources in a network and signals to the intermediate nodes along the path. Additionally to this basic functionality, the GMPLS framework leaves room for specialised tasks, e. g., path computation [RFC 4655, RFC 5657] and link management [RFC 4202]. The GMPLS framework does not imply selected protocols to realize this functionality, nor does it exclude any network technology. It provides the functional specifications and location of interfaces for a smooth operation. Summarizing, GMPLS represents an integrated control plane, which suits the requirements for NGN for heterogeneous transport networks.

One representative for a control plane for the service stratum is the IP Multimedia Subsystem (IMS, [41]). The IMS is a framework originally from 3GPP (thrid Generation Partnership Program) to enable multimedia services, i. e., voice and video, over IP technology, i. e., the Internet. For easy adaptability to IP, IMS reused IP protocols from

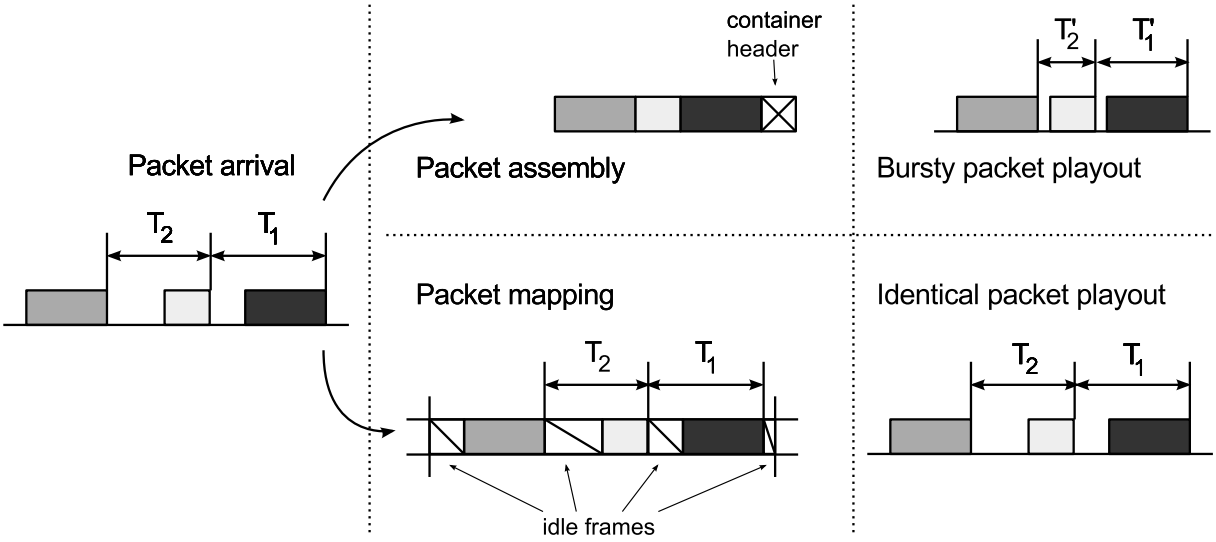


Figure 2.4: Packet assembly and mapping of packets in TDM slots

the IETF, i. e., SIP (Session Initiation Protocol, [RFC 3261, 42]) for signalling or Diameter [RFC 3588] for authentication, authorization and accounting (AAA). Besides the protocols, IMS introduces various new network elements, i. e., media and application servers as well as various gateways to access the signaling as well as the data stream. Between this network elements and at the border to other network control plane frameworks, IMS defines interfaces. As IMS relies in the service stratum, the interconnection to a potential underlying GMPLS infrastructure is limited.

2.2 Packet Assembly

Next Generation Networks face a steadily increasing packet-processing rate due to the increasing data volumes in these networks. One solution to reduce the packet rate is packet assembly. This section introduces the benefits of this concept in present and future networks. This section firstly introduces the generic packet assembly process in packet-switched networks. Secondly, it motivates the application of packet assembly in packet-based networks and provides a classification of assembly schemes in a separate section. In a last section, it classifies and introduces network technologies implementing packet assembly schemes.

2.2.1 Definition

Packet assembly or packet aggregation refers to a process including in principle three steps. The first step classifies packets and selects forward equivalent classes (FEC) for assembly. Packets of the same FEC belong together on a higher order, e. g., they show the same destination or network address or may belong to the same packet flow. The second step gathers packets of one FEC together in the assembly buffer. The third step forms a container, which includes the assembled packets. Literature refers to these containers as

bursts [38] or frames [43] depending on the underlying technology. Thereby, the container again is a packet and the original packet-switching paradigm of the network remains after the assembly process. Note that any information on the inter-arrival time vanishes in the buffering process.

The packet assembly process is different from the similar mapping process of TDM (Time Division Multiplex) systems like Synchronous Digital Hierarchy (SDH, [G.803]) or Optical Transport Networks (OTN, [G.872]). These technologies show circuit-switched characteristics, mapping individual packets into dedicated slots of defined duration. This mapping maintains the original traffic characteristic and inserts special frames indicating the inter-arrival gap of the incoming traffic. The destination node discards these special frames in the playout buffer and recreates the original traffic pattern.

Figure 2.4 illustrates the difference between packet assembly and the packet mapping. In the left part of the figure, three packets of the same FEC arrive. The arrival process shows different inter-arrival times. In the upper part in the middle, the assembly process forms a container and adds a container header. This header includes information on the payload structure of the aggregate packet. Releasing the packets from the container at the right side vanishes the original inter-arrival time of these packets. The packets nearly follow back-to-back. The lower part in the middle of the figure depicts the mapping process where idle frames fill the inter-arrival time. These idle frames are technology dependent, in case of OTN these may be idle frames of GFP (Generic Framing Procedure, [G.7041/Y.1303, 44]). At playout, the destination node discards the idle frames and restores the original packet sequence including the inter-arrival time.

In contrast to the packet assembly process, the mapping process is fully transparent for the packet network. This may be desirable because of policies between different network operator and the assembled arrival at the destination.

2.2.2 Motivation and Application

Section 2.1.2.1 discusses the principles of packet assembly, which provide an option to alleviate high packet processing rates, while sharing resources efficiently. The principle idea is to have large packets for a reduced packet rate, in addition to the exploitation of the statistical multiplexing gain of packet networks.

Increasing line rates in packet-switched networks require high performance nodes to process each individual packet. In a 10 Gbit/s system, the network nodes need to process a minimum sized packet (84 B on the wire including inter-framing gap) within 67.2 ns. Currently, next generation transmission capacities of 100 Gbit/s are in the standardization process [802.3ba]. Within these networks, network nodes need to process a packet every 6.7 ns. Modern Field Programmable Gate Arrays (FPGA) operate on a clock frequency in the order of 100-200 MHz which translates to 10-5 ns. Consequently, 100 Gbit/s capable network nodes need to finish processing a packet within every clock cycle of the FPGA. Modern network node architectures apply deterministic pipeline architectures with efficient pipeline stages to fulfil these timing requirements. One example of these high-speed network node architectures is the Frame Aggregation Unit operating at 10 Gbit/s [43,45].

The hard timing constraints are relevant in case of minimum sized packets. Larger packets relax these timing constraints significantly. The required processing effort reduces reciprocally with the packet size. Considering a packet size of 1500 B (Ethernet MTU, [802.3]) or the non-standard implementation of Ethernet jumbo frames of about 9000 B increases the minimum packet inter-arrival time to 18.8 ns and 100 ns respectively. This relaxes the burden on the node hardware implementation and reduces the required switching/processing effort in the networks [43]. Packet assembly decreases the packet rate and maintains the possible statistical multiplexing per interface.

Besides these advantages, packet assembly shows the disadvantage of creating bursty traffic at the disassembly unit. If the destination node releases a container, it usually forwards the packets back to back or with minimum possible delay. If several paths share a common outgoing link, this may lead to bursty traffic at the receiver node, producing unintended overload situations. Lautenschläger argues in [43] that this scenario is possible but not very likely. Additionally, several countermeasures may relax this potential bottleneck. For example, the assembly process may record the inter-arrival time, while the receiver implements a playout buffer, observing these times.

The arguments of above define the location of packet assembly. Networks usually implement packet assembly at the network edge. The edge shows in general a decline in the packet rate, the packet rate in the network is higher than outside the network. Within the network, traffic load is high and packets arrive back-to-back. This is in general beneficial for circuits as the gain for packet assembly is low. In the access network, traffic load is low and the packet rate is low, too. Any further decrease in the packet rate is not efficient. At the border of a network, packet assembly aggregates several flows of low rates together. Because of the reasons above, packet assembly is an option for metropolitan area networks.

2.2.3 Classification of Assembly Schemes

A classification of the most common assembly schemes proposes three different classes: time based, threshold based or probability based. Besides these three classes, there exists a variety of other assembly mechanisms adapted to special network technology requirements. This thesis skips details on them and restricts itself to one selected example. For the three major classes, Vega et al. give in [46] an overview on the algorithm and mechanisms proposed in literature. The assembly schemes have in common a buffer to collect packets and a control unit to decide whether a burst is ready for sending.

Time based assembly schemes start a timer on the first packet reaching an empty buffer. Until the timeout, the assembly unit aggregates all packets arriving in the meantime. At timeout, it assembles the packets in the buffer to one burst and forwards it to the transmission section. The buffer remains empty until the arrival of the next packet. This assembly scheme only focuses on the time aspect of any assembly. Consequently, it limits the maximum delay a packet receives in the assembly process.

Assembly schemes may also consider the *size of packets and bursts*. For this scheme, the assembly unit implements a size threshold, defining the maximum size of a burst. Again

starting from an empty buffer, the assembly process gathers the arriving packets until their total amount exceeds the size threshold. Then, the assembly unit forwards the burst to the transmission section. This mechanism is beneficial, if the assembled packet faces size limitations.

A variation of the threshold based assembly is the implementation of a *packet counter*. This scheme keeps a counter reflecting the number of necessary packets until sending the burst. The first packet arrival at an empty buffer initializes the packet counter. Each additional packet reduces the counter by one until reaching zero, which determines the time to send.

As a third option, the assembly process may implement a *random experiment* to determine the time to sent a burst. One example of this scheme is the *random selection assembly scheme*. At each packet arrival, it decides with a certain probability, whether the burst is ready for sending or not. The selection of the probability defines the size distribution of the burst. If the assembled traffic shows Poisson characteristics, the burst traffic characteristic shows Poisson characteristics, too.

A special case of probability based packet assembly is the aggregated FIFO discipline by Tounsi et al. in [47]. They assume several distinct classes in the assembly buffer and process the packets in the buffer in FIFO discipline. The first packet in service defines the class all packets in the burst will have. Then the assembly unit picks all packets of this class out of the buffer and assembles them to one burst. This assembly mechanism requires special buffer hardware for random access and a database to maintain the packet class and position.

Besides these pure implementations, any combination of these basic schemes exists. For instance, the combination of threshold/counter and time based assembly avoids starvation of packets in the buffer in temporary low load situations. The time defines a maximum waiting time, while the size threshold defines a maximum burst size.

2.2.4 Packet Assembly Implementations

The idea of packet assembly at the network edge to reduce the switching effort is visible in several network technologies. Table 2.1 classifies the individual proposals according to network layer and network technology. The first column gives the reference in the literature, while the second lists the assembly strategy. The last column describes the network scenario. The remainder of this section introduces each technology and points out the applied assembly scheme.

On the application layer, Pentikousis et al. apply in [48] packet assembly for VoIP packets in a fixed IEEE 802.16 WiMax scenario. They implement packet assembly on the application as well as on the MAC layer. On application layer, several voice samples share one RTP packet (Real-Time Transport Protocol, [RFC 3550]), while on MAC layer several IP packets share one larger container. They found, that both assembly mechanisms significantly reduce the transmission overhead with respect to the original ITU-T

Reference	Assembly strategy	Scenario
Application layer		
Pentikousis et al., [48]	Timer/size based	802.16 WiMax
Network layer		
Tounsi et al., [47]	Aggregated FIFO	IP/MPLS networks
Kanda et al., [49]	Size based	IP networks
Wang et al., [50]	Size based	Ultra wideband networks
Zhang et al., [51]	Size based	Cellular networks
MAC layer		
Pentikousis et al., [48]	Timer/size based	802.16 WiMax
Tao et al., [802.16, 52]	Size based	802.16 WiMax
Castro et al., [53]	Timer/size based	802.11 WLAN
Leligou, Mutter, Lautenschläger et al., [43, 45, 54]	Timer/size based	Ethernet networks

Table 2.1: Classification of network technologies using packet assembly

standard [G.723.1], which transports one voice sample per frame only. Their proposal further allows more data flows at the same time without performance degradation.

Several other authors addressed the small voice over IP (VoIP) packets as the major driver for any packet assembly. In 2001, Tounsi et al. propose in [47] and in 2004, Kanda et al. discuss in [49] packet aggregation at the network edge to reduce the number of voice packets in the core. In their proposal, packet aggregation takes place on the networking layer. As an assembly strategy, Tounsi et al. apply an aggregated FIFO discipline. Kanda et al. apply a packet count assembly strategy and vary the number of packets in a burst. Both authors study the impact of the assembly process on the packet layer and the network performance. They find that with reasonable assembly strategies, the impact on packet layer is negligible for reasonable scenarios. At the same time, the overall packet load decreases, this relaxes the effort in routers. Their studies focus on the benefits of aggregation and left out any discussion on the resulting container format on IP.

In [51], Zhang et al. propose packet aggregation in cellular networks using the packet reservation multiple access protocol (PMRA, originally proposed by Wong in [55]). This protocol operates on a shared medium but provides a low throughput because of small voice packets. They provide a Markov chain model of the assembly process and derive the delay and throughput of it. Their model allows a precise design of the assembly process considering the trade-off between delay and throughput.

In [50], Wang et al. also consider packet aggregation in wireless networks on MAC layer. They focus on an ultra wide band scenario and identify the short packets in the fieldbus control system for channel acquisition as a limiting factor. They propose a size based aggregation algorithm to assemble these small packets and transmit them as a whole. Their modified MAC algorithm outperforms the classical CSMA/CA algorithm and shows the advantage of packet assembly in this scenario.

Castro et al. propose in [53] also packet assembly for IEEE 802.11 wireless mesh networks on a hop-by-hop basis. They implement packet assembly on the MAC layer and apply

a combination of size and timer based assembly schemes. They also find that packet assembly reduces the load in the network leading to a better network performance. The reason is that larger packets on the MAC layer save time in terms of SIFS[1] and DIFS[2] periods. They identify the assembly strategy as the central parameter for optimum network performance.

IEEE 802.16 WiMax technology itself offers the capabilities for packet aggregation on the MAC layer. The standard defines in [802.16] the packing procedure, which allows the aggregation of multiple MAC SDU (Service Data Units) into a single MAC PDU (Protocol Data Unit). It further distinguishes if the content consists of ARQ (Automatic Repeat Request) or non-ARQ packets and if the aggregated packets have a fixed size. Because of the fixed frame size, the mechanism corresponds to a size based assembly algorithm. In [52], Toa et al. provide the performance evaluation for this assembly mechanism in WiMax and show efficient link utilization and reduced overheads.

On layer 2, Frame Switching [54] performs packet assembly. Frame Switching assembles packets at the network edge on the MAC layer and forwards them along a pre-established path to the destination node. For the establishment of the path, Frame Switching applies a modified GMPLS (Generalized Multi-Protocol Label Switching, [RFC 3945]) control plane. In the original proposal, Frame Switching uses G.709 containers from the Optical Transport Network (OTN, [G.709]) as containers. Newer studies show that even Ethernet frames are able to serve as containers. In their studies, Mutter and Lautenschläger apply in [43, 45] Ethernet Jumbo Frames as container technology. Besides the technological feasibility, they also show that in their scenario the assembly timer has only limited effect on the assembled packet stream.

2.3 Multi-path Routing

Another challenging problem is network congestion due to increasing traffic on dedicated paths in the network. Some links show heavy congestion, while other links remain underutilized. One solution to alleviate this problem is multi-path routing. Multi-path routing increases network resource efficiency and reduce network congestion by distributing packets to the same destination on several alternative paths. This section first gives a definition and a common understanding of multi-path routing. It refers to standards, which originally define multi-path routing on the network layer. Second, it motivates the application of multi-path routing mechanisms in current and future networks and argues its benefits for improved network efficiency with less congestion. The last section introduces network technologies, which implement multi-path routing algorithms, i. e., deflection routing in Optical Burst Switching.

[1] Short InterFrame Space
[2] DCF InterFrame Space, Distributed Coordination Function (DCF)

2.3.1 Definition

Protocols in communication networks rely on the paradigm that the network maintains the order of messages of the same flow. Thereby, a flow consists of packets sent from a particular host to any address. It may consist of all packets of a specific transport connection or media stream. Usually, a 5-tuple of source/destination address, source/destination port and protocol identifier defines a flow [RFC 3697].

Communication networks try to maintain this in-order requirement and send messages of the same flow along the same path. Herein, the same path refers to the same network elements and order of links and nodes from source to destination. In connection-oriented networks, messages inherently follow the same pre-reserved path. An exception is the LCAS protocol for SDH (Link Capacity Adjustment Scheme, [G.7042/Y.1305]), which enables to bundle individual connections to one logical one. In this case, the individual connections may follow different paths, but LCAS provides mechanisms to re-establish the original packet order. In connection-less networks, network elements are able to forward messages individually but usually on the same path to the destination. Static routing and dynamic routing protocols (e. g., Open Shortest Path First, OSPF, [RFC 1247] and Routing Information Protocol, RIP, [RFC 1732]) realize consistent topology information and provide exactly one next hop for forwarding.

Multi-path routing breaks with this paradigm and forwards packets to the same destination on different alternative paths. Therefore, each node maintains a list of alternative next hops to a certain destination node. On every packet arrival, the node decides anew which next hop to use. The next hop itself may own several subsequent next hops for the next destination. The procedure recursively starts from new. The decision on the next hop bases on load balancing mechanisms, which are subject of the next section.

2.3.2 Motivation and Mechanisms

In general, multi-path routing serves three major objectives: network engineering to decrease maximum link load, network availability and security.

The term load balancing refers to the first objective. With a suitable distribution of packets on alternative links, it is feasible to minimize maximum link load. Literature refers to this topic as adaptive multi-path routing. Selected publications contain the work of Abrahamsson and Karsson in [56, 57], respectively. Other objectives besides link load include performance metrics regarding the minimization of packet loss or delay.

The second argument for multi-path routing refers to the increased availability of an end-to-end connection. As the network element provides several alternative next hops for the same destination, it increases the probability for a working path in case of network errors. The switchover from the failure path to the working paths is quite simple, as the information to alternative next hops is already present. In case of a link failure, the node may skip the erroneous link and forward the traffic to the remaining hops. Without a hot standby of alternatives, the routing protocols first have to propagate the link failure and each node is required to update its topology and forwarding information.

The last argument for multi-path routing is increased security with respect to taping the messages of a flow. If these messages follow the same path, it is sufficient for an attacker to access one device on this path to get access to these messages. If these messages follow different paths, an attack requires access to different distributed devices.

In a multi-path routing environment, there are several algorithms to determine the next-hop for a packet (besides the destination address). The simplest mechanism performs a round robin or random selection algorithm on the alternative next hops. This spreads the packets among the alternative next hops. This may result in equally loaded links with respect to the traffic matrix of a network. As a drawback, this approach does not maintain the packet order within a packet flow. As TCP reacts severely on changes in the packet order, the IETF proposes in [RFC 2991] and [RFC 2992] hash based algorithms to determine the same next hop for packets of the same flow. The routers perform a hash operation on the packet header or at least on parts of the flow relevant header to determine the next hop. If the hash function leads to the uniform distribution, each next hop gets nearly the same share of packets.

This approach keeps the packet order as packets of the same flow follow the same path. Besides this advantage, this approach shows several disadvantages. First, the routers need to perform the hash operation on every packet. Consequently, the hash function needs to be resource efficient in high-speed networks. Second, dominating and long-lasting flows, still use the same path leading to an unequal distribution of traffic in the network. Third, the assumption on the uniform distributed hash value is only correct for a large number of flows, which typically only holds for large networks. In smaller networks, the flows become unequally distributed in the network.

Summarizing, multi-path routing has an ambivalent character. On one side, it shows a higher degree of freedom for efficient network engineering by distributing the traffic within the network. On the other side, it may change the packet order in the network, wherefrom protocols may suffer. Maintaining the packet order requires effort within the network, which increases network complexity and may lead to undesired results of unequally distributed traffic. For network simplicity, the network itself may not implement mechanisms to maintain the packet order. Then, the protocols in the end-systems like TCP need to handle out-of-order packets as described in the previous Section 1.1.3.2.

2.3.3 Multi-path Routing Implementations

A series of technologies try to improve network performance by forwarding packets to the same destination on different paths. Multi-path routing is present on transport and networking layer as well as on layer 2. Table 2.2 classifies the application fields of multi-path routing with respect to the layer. The columns show the reference in the literature in the first column and the field of application in the second column. The following paragraphs discuss selected network technologies with respect to their multi-path forwarding property.

On the transport layer, TCP dominates. For instance, Dong et al. propose in [58] a modified TCP version to handle multiple paths to the same destination node. The objec-

Reference	Application
Transport layer	
IETF [RFC 2960]	SCTP
Dong, Rojviboonchai, Han et al., [58–60]	Multi-path TCP
Network layer	
IETF, He, Chim et al., [RFC 2991, RFC 2992,61,62]	Equal-cost multi-path routing in IP networks
MAC layer	
Qiao et al., [38]	Contention resolution schemes in OBS and OPS networks (cf. Section 2.4.2)
Mueller et al., [63]	Wireless mobile ad hoc networks

Table 2.2: Classification of network technologies using multi-path routing

tive is an equal load distribution among several paths to experience minimum congestion. They propose and implement a backward compatible version, which handles the reordering problem on parallel paths. Their starting point is a multi-homed environment, where a scheduler determines the outgoing interface for the next segment. The decision on the interface follows a weighted round robin, where the weights depend on the round trip time and packet loss probability of each path. Rojviboonchai et al. follow the same research direction in [59]. They propose a TCP version, which is able to handle different paths on the transmission layer. The distinction relies on the different IP addresses of a multi-homed device. They solve the problem of reordering by adapting the dup-ack threshold as in the RR-TCP version of [16]. Han et al. provide in [60] the analytic base for a multi-homed TCP version exploiting multi-path routing. They focus on the analytic controller modelling, which is able to distribute traffic on multiple paths in-order to minimize congestion. Their major contribution is the proof regarding the linearized stability.

Besides TCP, there is SCTP [RFC 2960], which also supports multi-path routing in multi-homing environments. SCTP creates an association (i. e., connection) between source and destination instance. Within an association, it maintains several streams (i. e., flows) in parallel. Each stream maintains the packet order, while the streams among each other do not necessarily maintain any order. Furthermore, in multi-homing scenarios, SCTP is able to exploit different interfaces within the same association. Consequently, SCTP represents one alternative for TCP with respect to reliable transport and exhibits multi-path functionality. As the original intention of SCTP was to transport telephony signalling messages (Signalling System No. 7) over IP, there are only a few applications implementing SCTP (Diameter [RFC 4740], Reliable Server Pooling [RFC 5351]).

On the network layer, the multi-path routing feature of IP enables efficient resource utilization. While the IETF standards [RFC 2991] and [RFC 2992] provide the technical specification of any multi-path capable routing protocol, different proposals are available to distribute load among alternative routes. The crucial property for any load-balancing algorithm is the maintenance of the packet order within one flow. One representative reference to load-balancing algorithms maintaining the packet order within a flow is [62]. The authors propose a hash-based algorithm for load-balancing purpose. They evaluate

the additional effort in the router and show the benefit in both, distribution of load as well as out-of-sequence arrivals. Their algorithm was able to distribute the packets among alternative paths nearly equally with an acceptable amount of out-of-order arrivals with respect to transport protocols, e. g., TCP.

In the context of the Future Internet initiative, the topic on multi-path routing also came up. There, the community sees multi-path IP routing as an enabler for flexible traffic engineering in future networks. He and Rexford survey in [61] the different multi-path routing strategies, relying on round-robin or hash based procedures. They conclude that the trade-off between additional overhead for storing different routes and the potential traffic engineering impact justifies further studies on this topic. Additionally, Gojmerac proposes in his dissertation an adaptive multi-path algorithm for an equal distribution of load among alternative paths [64]. Optimized multi-path routing algorithms rely on flooding mechanisms to distribute the information on congested links in the network. As this produces a non-deterministic amount of signalling messages, Gojmerac proposes a modified signalling architecture to distribute the congestion indication only locally to a congested link. The proposal achieves both, a reduced amount of signalling messages as well as traffic engineering of multi-path routing.

Mobile ad hoc networks show infrastructure-less properties, provide interconnection for resource constraint nodes and exhibit high dynamics in the topology. Additionally, the wireless connections are unreliable and the environment of these networks usually is unpredictable. These networks provide any node inter-connection in an ad hoc mode and apply routing protocols specialized for these environments. For reliability issues and load-balancing purposes, the routing protocols apply multi-path routing mechanisms on layer 2. Mueller surveys in [63] on these issues. The discussion includes two different routing algorithms and their performance with respect to minimizing the end-to-end delay and to satisfy bandwidth demands. This work focuses on the routing aspect but leaves out the discussion on the impact on upper layer protocols such as TCP.

As an additional technology on layer 2, optical burst switching networks also exhibit multi-path mechanisms to avoid congestion. In this context, the terminology deflection routing applies. As OBS serves as a representative for network technologies implementing packet assembly as well as multi-path routing, Section 2.4.2 introduces this network architecture as well as the proposed multi-path routing mechanisms.

2.4 Examples for Next Generation Networks

This section introduces network technologies of present and future, which implement packet assembly and multi-path routing in one of its flavours. The first part gives an overview and classifies the network technologies according to the network layer. The second part picks the OBS network architecture as an example, which implements both features. By means of OBS, this section shows the principle scenario necessary to evaluate out-of-sequence arrival pattern on burst and packet level.

Network architecture	Multi-path routing	Packet assembly
Network layer		
IP-based Internet, [RFC 2991, RFC 2992]	Y	N/Y[a]
MAC layer		
Frame Switching, [43, 45]	N/Y[b]	Y
Optical Burst Switching, [38]	Y	Y

[a] Extensions of Kanda [49] and Tounsi [47] enable packet assembly in IP networks
[b] Connectionless mechanisms, e. g., IP routing, enable multi-path routing.

Table 2.3: Network technologies showing multi-path routing and packet assembly

2.4.1 Overview and Classification

This section classifies selected network technologies with respect to the implemented features of multi-path routing together with packet aggregation. It summarizes the findings of the previous two sections. Table 2.3 gives an overview on three network technologies and classifies them with respect to the network layer.

The Internet architecture applies the Internet Protocol (IP) on the networking layer. With support of appropriate routing protocols, IP inherently offers the possibility of multi-path routing as described in [RFC 2991, RFC 2992]. Native IP does not foresee packet assembly, but literature proposes several extensions to IP to support packet assembly. As already introduced, this includes the proposals of Tounsi, Kanda and Pentikousis, [47–49]. As the future Internet initiative currently discusses this issue, any estimation on the implementation is too early at this point of time.

Frame Switching resides on layer 2 [54]. Frame Switching assembles packets at the network edge on the MAC layer and forwards them along a pre-established path to the destination node. Consequently, it shows packet assembly properties but does not enable multi-path routing. As the forwarding of aggregates is only a matter of applied technology, multi-path routing extensions are possible if the applied technology supports it. In the studies of Mutter and Lautenschläger, Ethernet Jumbo Frames serve as container technology [43,45]. Applying IP based forwarding realizes the same task but also enables multi-path routing on the networking layer.

The most important network technology showing packet assembly together with multi-layer routing is Optical Burst/Packet Switching (OBS/OPS, layer 2). OBS is an example for other technologies, which also show multi-path routing as well as packet assembly capabilities. The next section provides a description of this network architecture.

2.4.2 Optical Packet/Burst Switching Networks

This section highlights the two principle properties of multi-path routing as well as packet assembly exemplarily in optical burst/packet switching networks. This section restricts itself to the basic operation and properties of OBS as the description of any flavour would not contribute with any new knowledge. Besides this section, Battestilli and Gauger give

an in depth introduction to OBS in [65] and [66], respectively. The remainder of this section firstly introduces the principle burst switching mechanism and secondly presents the network architecture with its network elements. The last section covers the contention resolution schemes, which relate to multi-path routing strategies.

2.4.2.1 Optical Packet Switching Principle

As packet processing on the electrical layer reaches its limits (cf. Section 2.2), optical packet switching (OPS) promises on the optical layer to realize both, flexible resource sharing like packet-switched networks together with energy-efficient switching of optical devices. The idea of OPS in principle tries to enable the same benefit in optical networks as IP or Ethernet provides for electrical networks.

OPS networks assume fibres with several wavelengths interconnecting neighbouring nodes. For transmission, the node occupies a wavelength and maps the optical packet onto the wavelength. The receiving node receives the optical signal, processes and switches the packet transparently to the next node or delivers it locally via an electrical interface. In transparent switching, the destination address in the optical packet header field determines the outgoing interface. The switching information in the optical packet header compares to the IP packet header. Literature refers to this technology as Optical Packet Switching (OPS). The fundamental operation is similar to electrical packet processing except of the more difficult optical header processing. Therein, OPS shows three major limitations: optical processing, optical buffering and switching speed within a node.

In case of contention, i.e., on an occupied interface, the node needs to buffer packets until the wavelength is available again. As optical RAM (random access memory) is not available, fibre delay lines (FDL) are an option to buffer optical packets. FDL represent fibres of certain length, which delay a congested packet. The length of the fibre determines the duration of the delay. For a delay of 1 ms the FDL shows a length of 200 km, which requires long fibres and large components. Because of the discrete buffer time, this workaround is inflexible and space consuming. Summarizing, optical RAM buffers are not available and FDL are only a workaround for this limitation.

The second drawback of optical packet switching relates to the switching speed of intermediate switching devices. On arrival of optical packets, the node has to configure the switching elements in time to avoid unnecessary delays, buffering and losses. Current hardware is not able to detect and process the optical header as fast as required to configure the switching matrix. An offset between optical header and optical payload as well as optical buffering alleviates this drawback as Klonidis et al. show in [67].

The third limitation is the optical processing of messages. There, research is in its early years. A workaround is to convert the optical signal in an electrical signal for further processing. After processing, the signal is converted to an optical signal again for transmission. This double signal conversion refers to optical-electrical-optical (o/e/o) conversion. In case of switching optical packets this process refers to opaque switching.

2.4.2.2 Burst Switching Principle

The previous section identified the switching speed as well as the lack of buffer mechanisms as the major limitations for optical packet switching. Optical burst switching alleviates the limitations of OPS partly. As optical packet processing is not feasible, Optical Burst Switching (OBS) separates header processing, i. e., forwarding information, and payload forwarding. This separation within OBS introduces a control wavelength in parallel to the wavelength channel for data transport.

In general, a burst consists of a burst header and a burst body. OBS splits both parts on different wavelengths. The burst header occupies the control wavelength and after a small gap, the burst body follows on the wavelength for data transport. On the control channel, the burst header packet announces the switching requirements of the corresponding data burst in advance. The control information equals the information in the header of optical packets. Besides others, the burst header packet includes information on the destination address. As the burst header packet precedes the burst data packet, the burst header processing and the configuration of the switching matrix of the next node may be feasible within the gap time. Besides the configuration of the switching matrix, the next hop processes the burst header packet electronically and reserves the next wavelength if necessary.

Literature shows several studies dealing with the reservation mechanisms for optical burst switching networks, e. g., [68–71]. For the optimization of the gap between burst header and burst payload, there are the mechanisms Tell and Go (TAG), Just-In-time (JIT) and Just-Enough-Time (JET). Rodrigues et al. provide in [68] a detailed comparison on these reservation schemes. Additionally, Dolzer et al. survey these algorithms in [70]. Besides this, each of the algorithms shows a scheduler organising the occupation of the connected wavelengths. It is highly related to the previous mechanisms. Consequently, literature proposes several schedules for the resource management of optical wavelength resources, e. g., [72–76]. In general, there are two major classes of schedulers, horizon based algorithms and void filling algorithms. The first class maintains the time instance when a wavelength becomes free again. In contrast to this, the second class maintains the time instances of the wavelength occupation including beginning and ending. The latter class is able to schedule additional bursts between the already reserved ones and thus shows significant advantage with respect to flexibility of scheduling. Among these general classes, several variations exist. For instance, a burst may be pre-empted or the occupation of alternative wavelengths follows special scheduling mechanisms (round robin, earliest free wavelength or least occupied wavelength). Li and Qiao provide a detailed overview on these scheduling mechanisms in [76].

2.4.2.3 Node Architecture

This section introduces a generic OBS node architecture highlighting the functional parts of an OBS node. Figure 2.5 depicts one generic OBS node model. It consists of optical fibre attachments on both sides. Each fibre carries a certain number of wavelengths, depending on the interface capabilities of an OBS node. At the ingress of the node, the

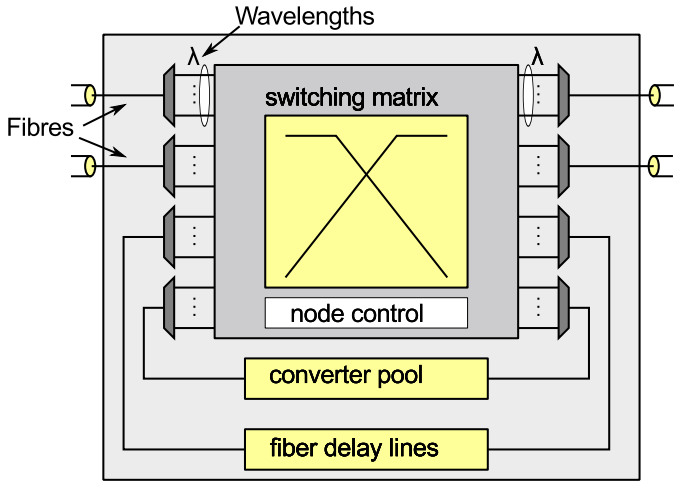


Figure 2.5: Generic OBS node architecture

fibre demultiplexes the individual wavelength channels. At the egress of the node, the wavelength channels are multiplexed again in the outgoing fibre. In the centre of the node, the switching matrix resides together with the node control entity. The switching matrix forwards the optical bursts from one input port to the foreseen output port. Thereby, the node control is responsible to configure the node for transparent switching. As the gap between the control packet and the burst is rather small, optical switching with more than four ports is hardly achievable today. El Bawab provides in [77] a comprehensive comparison of different realization candidates.

The control unit maintains a calendar scheduling the burst arrivals for the output wavelength. The generic model also foresees mechanisms to convert the data on one wavelength onto another wavelength. Wavelength converters in the lower half of the figure manage this task. These boxes represent the converters connected by fibres to the switching matrix. In case of wavelength conversion, the controller configures the matrix to forward the burst to a wavelength converter station. Literature also studied the amount of wavelength converters in [78–80]. Increasing the number of converters improves the network performance regarding the number of burst losses. This property holds on until saturation defined by the scenario.

Besides wavelength converters, the control unit also maintains the optional pool of fibre delay lines (lower right part of the node). Additionally to the wavelength converter pool, the control unit manages these resources and keeps free/busy lists. Dimensioning of the fibre delay line pool, as well as its number and length was subject of [81]. Therein, Gauger discusses the impact on the burst loss probability with respect to the number of FDL, the length of a FDL and the strategy for reservation. For his simulation scenario, he found a small number of FDL enough as well as the FDL length in the order of the burst length.

Thereby the consecutive application of wavelength converters as well as fibre delay lines is also possible in the generic model. The switching matrix together with the control unit may realize this application. This functionality increases the flexibility in case of contention but adds an additional burden on the controllers' complexity.

2.4.2.4 Contention Resolution/Avoidance Strategies

This section discusses the commonly used strategies for contention resolution and avoidance schemes. Thereby, a classification of both schemes according to the following definition is possible: *Contention resolution schemes* resolve a contention, while *contention avoidance schemes* try to minimize the number of contention situations in advance.

In [82], Rahbar et al. provide an in depth survey on the different schemes. They discuss the different schemes for all-optical packet-switched networks, which also hold for burst-switched networks. The next two sections introduce common contention avoidance and resolution schemes in detail.

2.4.2.4.1 Contention resolution schemes

Contention resolution schemes try to resolve a contention, where two or more bursts request the same resource, i.e., the same outgoing wavelength, at the same time. In principle, one burst succeeds this contention, while the node may apply countermeasures for the other bursts. Contention resolution provides three different classes: *shift in time*, *shift in space* and *shift in wavelengths*. The first one includes fibre delay lines, the second deflection routing and the third wavelength conversion.

A *shift in time* corresponds to an additional delay of the burst. The node delays the burst for a certain time interval to grand access to the resource after this time. Any delay mechanism requires some buffer. However, as optical buffers are not feasible today, fibre delay lines provide buffer mechanisms to delay bursts for a certain time interval. Thereby, the burst follows a fibre of a certain length. Due to the limited speed of light, the burst needs some time to travel. After this delay, the node again tries to forward the bursts on the original wavelength.

Shifting the burst *in space* reflects the possibility to forward the burst on an alternative path to the destination. This includes the network wide mechanisms of deflection routing. Applying deflection routing, the node determines an alternative path to the destination and consequently an alternative outgoing wavelength. If this wavelength is free, the node forwards the burst to this new outgoing interface. One may argue that deflection routing uses the network as a buffer. Additionally, this scheme may require mechanisms to avoid loops (if desired) for unlimited travelling of bursts, e.g., a hop counter. In case of contention, the forwarding principle on alternative paths realizes a similar idea of multi-path routing of Section 2.3.

The last option *shifts* the burst onto a *different wavelength*, using wavelength converters. The idea is to convert the wavelength on a different, free wavelength. Wavelength converters are organized in pools. The organisation of the converter pools may follow one of the following structures. Each interface may have its dedicated converter pool, or several interfaces share individual converters. Besides this, the wavelength converters may operate on the full range of wavelengths or only on parts of if.

2.4.2.4.2 Contention avoidance schemes

Contention avoidance schemes try to reduce the number of contentions or to avoid them in total. In principle, literature proposes two distinct mechanisms to realize this task. Multi-path routing may reduce the probability of contention, while any two-way reservation scheme is able to avoid resource conflicts in advance.

Multi-path routing distributes the traffic within the network on purpose, not as a reaction on any contention. As introduced before, it may try to optimize network performance, e. g., to minimize the maximum burst loss ratio over all links. Multi-path routing may take into account several network measures, for instance the switch load or link occupancy to achieve these goals. Thereby, the distribution of bursts over the network may take into account higher layer information, e. g., flow information or decide for each burst individually, where to forward it. The latter one allows the highest flexibility, but may degrade higher layer protocol performance. One algorithm implementing these concepts represents the work of Lu et al. in [83]. It routes flows on the same path avoiding out-of-sequence bursts/packets by minimizing the maximum link load.

An alternative solution to avoid any contention requires the reservation of resources in advance. As OBS shows on the fly reservation by the burst control packet, any two-way reservation scheme breaks with this paradigm. Nevertheless, literature proposes the tell and wait reservation algorithm, which reserves wavelengths on a path in advance [76]. The destination node acknowledges the reservation and after successful reception of the acknowledgements, the source node transmits the burst. The data burst releases instantaneously the reservation in the intermediate nodes. This mechanism relates to a circuit-oriented network technology, where the holding times are rather small. The advantages of a lossless network pay network performance and signalling complexity.

Besides this, literature proposes another control plane extension to implement a two-way signalling scheme to reserve capacity on the path from source to destination. In this case, intermediate nodes signal if wavelength occupation is possible or reject the connection request. This scheme avoids burst contention but may lead to an increased waiting time at the network edge. The wavelength-routed optical burst switching architecture of Düser et al. in [84] represents an example for this kind of architecture.

3 Packet Reordering Metrics

In packet-based data communication networks, sender and receiver exchange one or multiple packets. If the sender commits more than one packet to the network, these packets form a stream of packets, i. e., they belong to the same packet flow. The stream of packets has an initial order at the sender. This sequence of packets may change in the network due to network anomalies. The receiver may experience a different order. In this case, reordering has happened in the transport network. The precise analysis and differentiation of out-of-order arrivals requires reordering metrics and especially the definition of out-of-sequence and in-sequence arrivals. Reordering metrics measure and classify reordering pattern.

Besides changes in the packet order, the receiver may detect other changes within the packet sequence. For instance, the network may delay, drop or corrupt packets. Consequently, any analysis of the packet order requires robustness with respect to these network anomalies. As briefly introduced, the reordering metrics face strict requirements for a reasonable application. The next section imposes additional requirements and the second section classifies the reordering metrics proposed in the literature. This major contribution of this chapter is the introduction and evaluation of the related work on reordering metrics with respect to the imposed requirements. The last section discusses the selection of the metrics applied in this thesis.

3.1 Requirements on Packet Reordering Metrics

This section derives the requirements on metrics describing and quantifying out-of-sequence packet arrivals. Most of these metrics do not rely on specific implementation details of communication networks, but evaluate the sequence of a finite set of arbitrary discrete elements showing an initial order. In communication networks, this includes packets, cells, frames and bursts. The following sections consistently use the term packet for one element out of the set of discrete elements.

The common scenario to analyse the packet sequence assumes a communication relationship between a sender and a receiver. The sender sends a fixed number of packets in a certain sequence to the network. By definition, the packet sequence at the sender is in-order. This sequence of packets forms a packet flow. In this thesis, the term flow denotes a set of packets belonging together according to any superior order, e. g., belonging to the

same application. In this case, the 5-tuple of source/destination IP addresses and ports as well as the protocol identifier characterize a flow.

The sender sends the packet sequence to the network within a certain time interval. The receiver receives these packets one by another and stores them in a buffer according to some ordering criteria. The receiver always delivers packets in-order to the service user, i. e., an application. The receiver makes the decision to skip missing packets based on elapsed time or intermediate packet arrivals. The comparison of the original packet sequence and the packet sequence at the receiver may reveal the following changes in the packet sequence.

Changed inter-arrival time: the time between subsequent packets may vary compared to the sending inter-arrival time.

Changed time interval: the time from receiving the first packet until the last packet of the sequence may differ from the sending process.

Duplicated packets: at the receiver, a packet of the sequence arrives multiple times.

Lost packets: packets may not reach the receiver within a defined time interval. These packets are either lost or may reach the receiver after forwarding later packets.

Changed packet order: the sequence arrives in a different order than the sending order.

Misinserted packets: the arriving sequence may show a packet from another packet flow due to changes in the packet header.

Literature refers to the above list as network anomalies. Besides these *pure* network anomalies, any combination of these may occur. While these anomalies do not change the packet structure, networks may also corrupt packets due to malfunctioning network elements. In modern communication networks, corrupted packets may translate to lost packets as the data-link layer may discard them.

Based on findings by Piratla et al. in [85], the following sections derive the requirements on metrics to detect and classify changes in the packet sequence. These requirements include the presence of a sequence indicator, a definition of in-order and out-of-order, robustness, minimal complexity, and application specific claims. Each section details one requirement.

3.1.1 Sequence Indicator

Detection of a changed packet order firstly requires a classification criterion to specify some order. The classification criterion has to fulfil the following formal requirement: If packet i departs earlier than packet j, then the value of the classification criterion of packet i is less than the value of the classification criterion of packet j. If $C[i]$ denotes the value of the classification criterion of packet i, then the following holds: $C[i] < C[j]$.

In general, there are two options for the classification criterion: *Sequence numbers* and *timestamps*. Applying a sequence number, the sender assigns an integer number to each packet. This number increases by one for every packet, i. e., $C[i+1] = C[i] + 1, \quad i > 0, C[1] = 1$. With the timestamp option, the sender assigns the sending timestamp with a reasonable resolution to each packet.
Both identifiers increase monotonically, enabling the receiver to restore the original sequence. The next two paragraphs discuss both alternatives with respect to their ability to identify packet duplication and loss.

A packet duplicate is an identical copy of a packet at the receiver. The detection of duplicates requires a unique identifier for each packet. The sequence number as well as the timestamp fulfil this requirement. The following implementation example illustrates this property for the detection of packet duplicates. The receiver remembers the value of the classification criterion of the last packet forwarded to the client service layer. If this value is larger or equal than the corresponding value of the current packet, the packet is either a duplicate or late. In both cases, the receiver may safely discard it. In any other case, the receiver queues the packet in the buffer and detects any duplicates later, before their delivery.

A packet loss is the non-arrival of a packet at the receiver within a certain time interval. This definition includes both, packets arriving outside the considered time interval and packets discarded by the network. The detection of losses also requires a unique identifier for each packet, but it should additionally enable the receiver to determine the number of the missing packets. Here, the timestamp option and the sequence number differ. A gap in the received sequence numbers allows the receiver detecting the number of missing packets. With the sequence of received timestamps, the receiver is not able to detect missing packets without assuming a special traffic pattern, e. g., deterministic inter-arrival times.

Summarizing the above arguments, the sequence numbers enable the reliably detection of packet losses, packet duplicates and changes in the packet order. The remaining part of this chapter and the other requirements on the metrics assume sequence numbers to identify out-of-order packets.

3.1.2 Definition of Reordering

This section derives the requirements on the definition of in-sequence and out-of-sequence considering packet losses. For instance, the sender commits the following sequence to the network: 1,2,3,4,5. The sequence 1,2,3,5,4 arrives at the receiver. The first three packets of both sequences correlate, but packet 4 arrives after packet 5 although it departed before. Consequently, there are two interpretations possible, either packet 4 arrives too late or packet 5 arrives too early. The last interpretation allows no distinction from packet losses. The receiver would interpret packet 5 as reordered independently of a later arrival of packet 4. Consequently, the definition of reordering should only include late packets, which is consistent with the definition by Paxson in [15].

3.1.3 Robustness

An important criterion for any metric is its robustness against any of the other network anomalies. Thereby, robustness means that a lost or duplicate packet shall not disturb the calculation of the metric of further arrivals. The calculation of the metric should be stable and converge to the same value as without the lost or duplicate packet.

An example may illustrate this requirement. For instance, there are two arriving sequences 1,2,4,5,6,7,8 and 1,2,4,5,3,6,7,8. In the first sequence, packet 3 does not arrive at the receiver. It is lost. In the second sequence, packet 3 arrives after packets 4 and 5. Packets 1,2 and 4 as well as the sequence after packet 5 are the same in both cases. For these packets, any metric should determine the same value for both sequences. The only affected packets are 5 and 3, which should receive different values for both scenarios.

The same applies for the occurrence of reordering in combination with packet duplication. Consider for instance the sequences 1,2,3,4,5,3,6,7,8 and 1,2,4,5,3,6,7,8. In the first sequence, the network duplicates packet 3, which arrives after packet 5 the second time. The receiver experiences an in-order packet arrival after discarding the duplicate packet 3. In the second sequence, packet 3 arrives only once, but after packet 5. The receiver faces an incomplete set until the arrival of packet 3. Again, packets 6,7, and 8 should receive the same reordering values for both scenarios to guarantee a robust metric calculation.

In addition to these network anomalies, reordering metrics also face the wrap around problem. As digital systems limit the representation of the sequence numbers, they may wrap around from the largest value to the smallest value during a long-lasting flow. When a wrap around occurs, the receiver faces large and small sequence numbers at the same time, e. g., 65534,65535,0,1,2. A robust reordering metric has to cope with this behaviour not falsely determine 0,1,2 as reordered. The problem of wrap around is not unique to sequence numbers for reordering metrics, but applies in any kind of sequence number application, e. g., TCP and RTP. In general, protocols apply additional mechanisms to cope with the wrap-around problem. For instance, TCP uses timestamps to detect sequence number wrap-arounds [RFC 1323]. Perkins proposes in [86] for RTP (Real Time Transport Protocol, [RFC 3550]) an alternative. He proposes to hide the wrap-around of the sequence number to the application by extending the sequence number at the receiver. For this purpose, the receiver maintains a counter to record the wrap-around. As none of the following metrics considers the wrap-around problem in detail, all metrics implicitly rely on an infinite sequence number space. An actual implementation of sequence numbers requires one of the above mechanisms.

Summarizing, reordering metrics face the problem of network anomalies while observing changes in the packet order. Thus, these metrics have to identify changes in the packet order while being robust to any other network anomalies.

3.1.4 Application Specific Requirements

Packet reordering is crucial to applications relying on in-sequence communication. Given this aspect, reordering metrics may provide information on the potential impact of packet

reordering on these applications. For instance, receivers observing packet reordering may re-sequence these packets before delivering them to the application. Re-sequencing needs buffer space and delays delivery by a certain time. Reordering metrics indicating the required time and buffer space allow a dimensioning of the receiver device and an estimation of the application performance.

One example to illustrate this requirement is the impact on reliable connection services of the transport protocol layer, e. g., the effect on the Transmission Control Protocol (TCP, [RFC 793]) and the Stream Control Protocol (SCTP, [RFC 3286]). These services need to cope with packet reordering and to provide mechanisms to regain the original sequence. However, these mechanisms may degrade the provided services. Consequently, reordering metrics have to provide means to quantify this degradation.

3.1.5 Metric Complexity

The last requirement on reordering metrics is low complexity in terms of computational effort and memory consumption. In general, there are two application scenarios for reordering metrics. The first application scenario is an online scenario where some network devices update the reordering metrics online for each packet, i. e., *on the fly*. Examples are network information centres, which monitor these metrics as an indicator for quality of service. In this case, the evaluation of the reordering metrics in the network nodes should consume minimal resources and leave computational power to more important tasks, e. g., processing of router updates. Other examples are network simulations, where the evaluation of metrics requires additional time. Offline scenarios represent the second application scenario for reordering metrics. They apply the reordering metrics offline on packet traces.

While the first scenario requires an efficient computation of the reordering metrics, minimizing memory and computational power, the second scenario slightly relaxes this condition. Nevertheless, it is also desirable in the second scenario to obtain the results from the reordering metrics in an efficient way. Summarizing, obtaining metrics information must be easy implementable and resource efficient.

The Landau notation (big O-notation) denotes the limiting behaviour of a function if the argument tends towards a particular value or infinity, e. g., $O(n^2)$ denotes squared complexity of a function for increasing n. Thereby, n may be the number of function calls or items to process. n relates from the context of investigation. For the evaluation of reordering metrics, a limiting scenario often is the arrival of packets in reverse order, i. e., 5,4,3,2,1. As this scenario is very unlikely, the following evaluation of the complexity focuses on an *average effort* for calculating the metrics. Nevertheless, the evaluation also considers limiting scenarios.

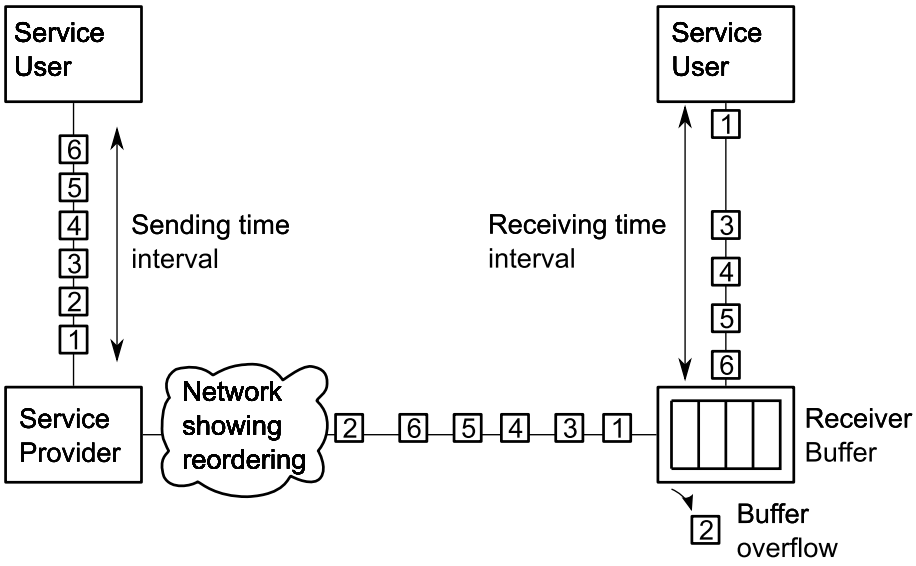


Figure 3.1: Application scenario for reordering metrics

3.2 Classification of Reordering Metrics

This section classifies the related work on reordering metrics. Most of the related work belongs to one of the following three classes: queuing analysis on reordering buffers and waiting times, studies on protocol performance with respect to packet reordering, and measurement and classification of reordering patterns.

The first class studies the characteristics of a reordering buffer facing out-of-order arrivals of packets (cf. Figure 3.1). Applying queuing theory, the major objectives are the waiting time characteristic and buffer size requirements. These studies relate to the reordering model of Chapter 4. Therefore, the later Section 4.1.3 provides an overview on these studies and classifies them with respect to the model presented in this thesis.

The second class studies the properties of application or transport protocols facing packet reordering. In most of the studies, either the packets of the IP layer [87] or the underlying OBS layer [22, 27] are reordered and the impact on the transport and application layer is measured. Thereby, the focus lies on the protocol performance without considering the different reordering pattern in too much detail. Consequently, most of the work manipulates the packet order and measures the protocol performance. They do not provide a detailed characterization of reordering with respect to any metric. Section 1.2 already introduced these studies and classified them with respect to the proposal of this thesis.

This section classifies the studies related to the third class, the measurement and classification of reordering events. Table 3.1 summarizes the work described in literature. The first column shows the name and the reference of the metric and the second column provides a short description of the metric. The following two columns indicate the robustness of the metrics facing packet loss and duplication. The next two columns indicate the effort required to obtain the metrics in terms of computational power and memory consumption in Landau notation. The last column indicates the feasibility of an online measurement application.

The classification of the metrics yields to four different types. The first type comprises metrics, which provide a precise definition of in-sequence and out-of-sequence arrivals, i. e., a definition of reordering. The second class characterizes the buffer dimensions at the receiver side to re-sequence packets. The third class includes the work of several authors studying the performance of TCP with respect to out-of-sequence arrivals. These authors mostly apply one of the reordering definitions of the first class and propose TCP related measurement or performance analysis. For completeness, the last class provides additional reordering metrics from the IETF.

		Robustness		Complexity		
	Description	Loss	Dupl.	Comp.	Memory	Online
Reordering definitions						
IETF Reordering definition, [RFC 4737]	Defines packet reordering	Y	Y	O(1)	O(1)	Y
IETF Reordering ratio, [RFC 4737]	Reordering indicator	Y	Y	O(1)	O(1)	Y
Piratla 2005, [88]	Reordering density	Y[1]	Y[1]	$O(1+\gamma W)$[2]	O(1)	Y
Gharai 2004, [89]	Defines packet reordering	-[3]	-[4]	O(1)	O(1)	N
Buffer size estimation						
IETF Reordering extent, [RFC 4737]	Buffer size estimation (packets)	Y	Y	$O(C)$[5]	$O(C)$	Y
IETF Byte-offset metric, [RFC 4737]	Buffer size estimation (byte)	Y	Y	$O(C)$[5]	$O(C)$[5]	Y
IETF Latetime metric, [RFC 4737]	Buffer size estimation (time)	Y	Y	$O(C)$[5]	$O(C)$[5]	Y
Piratla 2007, [13]	Buffer-occupancy density	Y[1]	Y	O(1)	O(1)	Y
TCP related metrics						
IETF n_r-reordering, [RFC 4737]	TCP dup-ack estimation	Y	Y	$O(C)$[5]	$O(C)$[5]	Y
Gharai 2004, [89]	TCP dup-ack estimation	-[6]	-[6]	-	-	N
Bellardo 2002, [90]	Reordering detection	-	-	-	-	N
Luo 2005, [91]	TCP reordering measurement	-[6]	-[6]	-[6]	-[6]	N
Jaiswal 2002, [4]	Classification of reordered packets	Y	-[4]	-	-	N
Other reordering metrics						
IETF Gap metric, [RFC 4737]	Reordering frequency	Y	Y	$O(C)$[5]	$O(C)$[5]	Y
IETF Freerun metric, [RFC 4737]	Reordering characteristic	Y	Y	O(1)	O(1)	Y

[1] requires renumbering of packets
[2] constant effort plus additional effort if packet is reordered, windowsize W
[3] they removed losses from their trace
[4] no reports on duplicates
[5] linear with the number of recorded discontinuity events C
[6] not applicable, measurement technique

Table 3.1: Classification of reordering metrics

Packet seq. nr.	1	2	3	5	4	6	7
Counter	1	2	3	4	5	6	7

Table 3.2: Example for the reordering density metric

The next sections review the related work on reordering metrics summarized in Table 3.1. They introduce the metrics and evaluate their applicability with respect to robustness and complexity. The latter one defines their applicability in online measurement scenarios.

3.2.1 Reordering Density

In [88, 92], Piratla et al. propose a formal and comprehensive reordering metric based on the work of [93]. They propose the *reordering density* metric to indicate and to quantify the displacement of packets in a stream. Their metric assumes sequence numbers that increase by one for each packet. At the destination node, they assume a counter, which increments by one for each arriving packet. If the packets arrive in-order without any loss, the value of the arriving sequence number corresponds to the counter value. Table 3.2 shows an example of the counter values. The first packet with sequence number 1 corresponds to the counter value of 1. The same applies for packets 2 and 3. Packet 5 is the fourth packet and packet 4 is the fifth packet arriving. At these two positions, both values differ. These differences are the basis for the definition of a reordering event.

3.2.1.1 Definition of a Reordering Event

The authors define a reordering event by the difference of the counter and the packet sequence number. If the counter value is greater than the sequence number, the packet is *late* by definition. The packet is *early* by definition, if the counter value is less than the sequence number. In both cases a *reordering event* occurred. If the sequence number equals the counter value, the packet is in-order. For a quantitative evaluation of packet reordering in the network, they depict the difference between the counter value and the sequence number in a histogram, e. g., Figure 3.2 depicts an example. Consequently, different reordering pattern show different histograms.
For deterministic traffic, it is possible to derive the metrics analytically. Piratla et al. show in [87] the analytic expression of their metric for deterministic traffic and verified their findings in an emulation scenario with two paths of different delays.

3.2.1.2 Robustness

Packet losses are considered first. As the counter for the expected packets increases continuously, counter and sequence number diverge in case of packet losses. This results in a constant offset for the reordering event (cf. Table 3.3). To remove this constant offset for further packet arrivals, the authors apply a window mechanism. The window size defines the maximum number of subsequent packet arrivals until the arrival of the missing packet. Otherwise this packet is declared lost. If the missing packet arrives within the window,

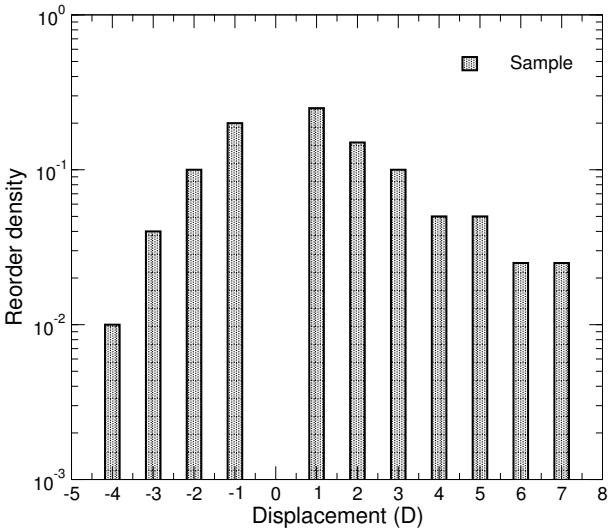


Figure 3.2: Sample histogram for the reordering density

the reordering event calculation follows the procedure as stated above. Otherwise, the receiver skips the lost packet sequence number and assigns the next smallest sequence number, which has already arrived, to the counter value.

Table 3.3 shows an example for packet reordering and packet loss. The network dropped packet 3 as visible in the first row. The second row gives the original counter value and the third row gives the derived reordering event value, respectively. With a window size of three packets, the receiver classifies packet 3 as lost after packet 6 arrived. It skips the sequence number 3 and assigns 4 to the counter value (counter*), as 4 is the smallest sequence number that already arrived. With the new counter value, the receiver calculates the new reordering event values leading to a zero offset for further arrivals. Summarizing, for a stable metric calculation in case of packet loss, the receiver has to adapt the metric values. Otherwise, a singular event is visible at every later arrival.

The case of packet duplication is simpler to handle. In the receiving buffer, the smallest sequence number indicates the next packet to be transferred to the upper layer. Additionally, the receiver records the last sequence number transferred to the upper layer. If a packet sequence number is less or equal than the sequence number already transferred to the upper layer, the packet is severely delayed or a duplicate. In both cases, the receiver may safely discard this packet.

3.2.1.3 Evaluation of Packet Density Metric

The packet density metric provides a measure to quantify the displacement of packets within a packet stream. It classifies arriving packets as late or early and provides a robust procedure to calculate the metric with respect to packet loss and packet duplication. For illustration and quantification, this metric is visualized in histograms (cf. Figure 3.2). The crucial part of any implementation is the receiving buffer. The following considerations on the complexity assume a buffer size of W places.

Packet seq. nr.	1	2	5	4	6	7	8	9
Counter	1	2	3	4	5	6	7	8
Reordering event	0	0	-2	0	-1	-1	-1	-1
Counter*	1	2	4	5	6	7	8	9
Reordering event*	0	0	-1	1	0	0	0	0

Table 3.3: Reordering density metric in case of loss

A newly arriving packet can face two different situations. In the first case, there are places in the buffer available and the receiver queues the packet. It calculates the metric value on the difference of two integer values, the counter and the sequence number. This process shows a constant effort per packet, i. e., O(1). In the second case, the packet may fill-up the buffer and trigger the detection of lost packets and the re-calculation of the counter value for the packets in the buffer. The effort for this step depends on the buffer size W, which results in an overall complexity of O(W). The worst-case complexity for N packets in a stream would be O(NW), which happens only if packets arrive in reverse order. Therefore, the average effort is a more practical measure. Let γ denote the probability of a packet facing a full buffer, then the average complexity of calculating this metric results in O($N + \gamma NW$) for a packet stream of N packets. The first term denotes the calculation of the reordering event value, while the second term represents the re-assignment of the metrics value of the buffered packets in case of packet losses. The memory requirements for this metric define two parts, the buffer size and the histogram. Both remain constant during calculation. Summarizing, the required overall memory is small and constant, i. e., O(1).

On average, the calculation and storage complexity per packet is constant. Scenarios that show packet losses or severe delays increase the complexity slightly. In summary, the metric is suitable for online application. Furthermore, the metric classifies packets as late or early. As already introduced, this differentiation is unusual as reordering events may only occur because of additional delays in the network. Therefore, an early packet arrival is hard to justify.

3.2.2 Reordering Buffer-occupancy Density

In [13], Piratla et al. propose an additional metric to quantify the *buffer-occupancy density* at the receiver. The buffer-occupancy density indicates the occupancy frequency of a virtual buffer at the receiver side. The same authors compared in [85] their metric proposal with the IETF metrics of [RFC 4737] presented in Section 3.2.5. Their proposal is documented in an informational RFC [RFC 5236], but became no standard and is not obligatory.

3.2.2.1 Definition

The buffer-occupancy density calculation requires the packet sequence number and an *expected sequence number* at the receiver. If E_{i+1} denotes the expected sequence number

Packet seq. nr.	1	2	5	4	6	3	7
Expected seq. nr.	1	2	3	3	3	7	7
Buffer occupancy	0	0	1	2	3	0	0

Table 3.4: Example for the buffer-density metric

and the next arriving packet has sequence number S_{i+1}, the following equation determines the value of E_{i+1} based on the current arrival. Only if the expected value matches the arriving sequence number, the expected sequence number is incremented, otherwise it remains unchanged.

$$E_{i+1} = E_i + 1 \quad \text{if } S_i = E_i \tag{3.1}$$

Table 3.4 shows an example. After the arrival of packet 2, the expected sequence number is 3, indicating that packet 1 and 2 have already arrived. The value remains unchanged during the arrival of 5, 4, and 6 as no further packet continues the sequence. Consequently, the receiver buffers all arriving packets until the original sequence is complete or the buffer is full. At every packet arrival, the buffer-occupancy value determines the number of packets in the buffer waiting for late packets. If the packet sequence number is larger than the expected sequence number, the receiver queues the packet in the buffer. In the example, the receiver stores packet 5, 4 and 6. Based on this measure, a histogram illustrates the buffer-occupancy density. To avoid infinite buffering in case of loss, the receiver defines a maximum buffer size to skip lost packets as indicated in the previous section.

3.2.2.2 Robustness

The robustness of this metric is not critical. In case of a packet loss, the same procedure applies as in the previous section. At the full buffer state, the sequence number of the missing packet is skipped and the expected value is assigned the next larger sequence number. Consequently, the metric remains operable facing packet losses.

The detection of duplicate packets occurs at the receiver when forwarding the packets to the upper layer. If the packet sequence number is larger than the sequence number of the last packet forwarded, the packet arrives as a duplicate and it may be safely discarded. The procedure equals to the mechanisms of the previous section. As the duplicate packet already contributed to the metric, this contribution has to be cancelled to avoid errors in the metric values. In a proper implementation, duplicate packets may not affect the metric values.

3.2.2.3 Evaluation

For every packet arrival, the metric records the number of packets in the buffer. A simple counter implements this metric. The counter increments by one if the packet is queued in the buffer. If one or more packets get forwarded to the receiver, the counter decrements by this number of packets. Consequently, the complexity per packet arrival is constant,

i. e., O(1). The representation of this metric requires a histogram, reflecting the density of the buffer occupancy. The effort to calculate the bin values is rather small and the memory consumption depends linearly on the number of bins. As in the previous section, the overall memory consumption is constant, i. e., O(1).

Summarizing, the effort to calculate and to store the metric per packet is constant. The processing effort increases linearly with the number of packets, while the memory consumption remains constant independent of the number of packets in the stream.

3.2.3 Monotonic Increase and TCP-like Metric

In [89], Gharai et al. study the amount of packet reordering in backbone networks. They sent UDP (User Datagram Protocol, [RFC 768]) flows of different length, line rate and packet sizes through the Internet and traced the received packets at the destination. Their first aim was to establish a link between the increasing line rate and the amount of reordering. Their second aim was to evaluate the observed reordering and the expected TCP performance facing the violation of dup-ack thresholds. For this purpose, they proposed two different reordering metrics. The first metric defines out-of-sequence arrivals and the second metric quantifies the reordering pattern with respect to the impact on TCP. They applied both metrics on recorded traces and did not aim for online measurement application.

3.2.3.1 Definition

Like the previous metrics, Gharai et al. apply monotonically and continuously increasing sequence numbers to the packets. For the definition of reordering, they use three packets. Let i, j and k the arrival position of packets with the sequence numbers S_i, S_j and S_k. They define that packet S_k arrives out-of-order if the following expression is true:

$$(i < j < k) \wedge (S_i < S_j) \wedge (S_j > S_k) \quad \forall i, j, k \tag{3.2}$$

As the authors were interested in the impact of reordering on TCP, they additionally define a *TCP-like metric*. They focused on the duplicate acknowledgement threshold of TCP described earlier in Section 1.1.3.2. They assumed a duplicate acknowledgement threshold of three and defined a violation of the dup-ack threshold by packet i with sequence number S_i if the following expression is true:

$$(S_i < S_{i-1}) \wedge (S_i < S_{i-2}) \wedge (S_i < S_{i-3}) \tag{3.3}$$

Literally, if there are three packets with a larger sequence number than packet i arriving prior to packet i, then packet i violates the duplicate acknowledgement threshold and triggers the fast retransmit algorithm. For evaluation of TCP, they count the number of these events to conclude on the TCP performance.

3.2.3.2 Robustness

As the authors were only interested in reordered packets, they skipped lost packets in the traces and recalculated the sequence numbers of the remaining packets. Additionally, the authors do not report any packet duplication, which was obviously not a problem; either because duplicates were implicitly filtered or did not occur. Nevertheless, this section evaluates the robustness of this metric, although the authors did not elaborate on this topic. Their definition of reordering corresponds in its detail to the reordering definition of Section 3.2.4. The evaluation of this metric is analogue to Section 3.2.4.

For the TCP-like metric, things are different. TCP sends duplicate acknowledgements for missing packets, which may be either lost or reordered. If a packet is lost, TCP will respond with duplicate acknowledgements in any case. If none of the packets is lost, the metric definition above applies. Both network anomalies have similar impacts on TCP, but the reordering metric covers only the reordering anomaly. The authors neglect the impact of packet losses. Consequently, the conclusion on the TCP performance due to packet reordering may not be precise enough.

3.2.3.3 Evaluation

The authors apply the above metrics offline on a recorded trace. Therefore, the complexity of the actual metric is only a minor issue in their scenario. Nevertheless, this section studies these metrics with respect to their potential applicability in an online scenario.

The reordering definition is quite simple. A packet is reordered if there is a packet arriving previously with a larger sequence number than the considered packet. This definition lacks the detailed information how to realize the comparison on the sequence numbers. One solution may be to introduce an expected sequence number as described next. This expected sequence number may represent the largest sequence number already received. A single comparison per packet may then classify a packet as out-of-sequence or in-sequence. The complexity of the evaluation of this metric is $O(N)$ if there are N packets per stream.

The complexity of the TCP-like metric is constant per packet. In any case, the receiver compares the sequence numbers of the three last arrived packets to the latest arrived sequence number. As this procedure is repeated per arriving packet, the complexity is $O(3N)$.

3.2.4 ITU-T Y.1540 Reordering Metrics

The ITU-T also addresses the packet reordering phenomenon on IP layer in its recommendation [Y.1540]. They provide two measures, a definition of reordering and a packet extent metric.

3.2.4.1 Definition of Reordering and Extent Metric

The ITU-T assumes sequence numbers and an expected sequence number counter at the receiver. The expected sequence number is the sequence number of the next expected packet. They define the expected sequence number and in-order and out-of-order packet arrivals the following way:

- A packet arrives *in-order*, if the sequence number of the arriving packet is equal or greater than the next expected sequence number. In this case, the receiver increments the next expected sequence number by one.
- A packet arrives *out-of-order*, if the sequence number of the arriving packet is lower than the next expected sequence number. The next expected sequence number remains unchanged.

Furthermore, they define the reordering ratio as the fraction of received reordered packets to the total number of received packets. Packets with a smaller sequence number than the expected value arrive reordered by definition. The ITU-T defines the displacement of the packet in relation to its original position as extent. Measures for the extent may be bytes, time, or packets. In addition to the ITU-T, the IETF standardizes this metric formally in [RFC 4737]. Section 3.2.5 will introduce it in detail.

3.2.4.2 Robustness

The counter for the expected sequence number increases only for packets with a larger or equal sequence number. Consequently, a lost packet results in a skipped sequence number in the counter sequence. The counter value requires no adaptation in case of losses; its implementation is quite simple and robust against future in-order arrivals.

The ITU-T does not elaborate on the arrival of duplicate packets, but the following arguments help to understand the problem. The detection of packet duplicates may follow the same principles as in Section 3.2.2.2. If the sequence number of the packet forwarded to the upper layer is larger than the considered sequence number, the packet is late or duplicate. In this case, the receiver may calculate the reordering metric also for duplicate packets in the same way as explained above. Then, the evaluation of the metric takes place before the delivery to the upper layer and the packet loss. Summarizing, the calculation of the metric with respect to packet duplicates is robust, but includes packets, which may be discarded later on. To avoid errors, the receiver may exclude duplicate packets from the metric evaluation. As the extent metric is similar to the metric proposed by the IETF in Section 3.2.5, its robustness is evaluated there.

3.2.4.3 Evaluation

The packet reordering definition of the ITU-T is quite simple. To determine if a packet arrives reordered, it requires only a single comparison with the next expected value counter

at the receiver. The effort to characterize a packet shows the constant complexity of O(1). Characterizing all packets of a stream requires O(N), if there are N packets in the stream. Consequently, this metric is applicable in an online measurement scenario. The evaluation of the complexity of the proposed extent metric is addressed in the Section 3.2.5.

3.2.5 IETF RFC 4737 Reordering Metrics

In [RFC 2330], the IETF defines a framework for IP performance metrics. Based on this framework, the IETF standardized metrics to classify out-of-sequence packets in IP networks in [RFC 4737]. This Internet standard includes a definition of packet reordering and provides a number of metrics to characterize out-of-sequence patterns in detail.

Like previous metrics, their metrics rely on two fundamental concepts: sequence numbers for each packet and an expected sequence number at the receiver. The sender assigns a monotonically increasing sequence number to each packet. The receiving node maintains an expected value of the next sequence number to arrive. With this framework, the IETF defines in-order and out-of-order arrivals as well as the following metrics: reordering ratio, reordering extent and related metrics, freerun metrics, and a TCP related metric. The next section introduces these metrics in detail.

3.2.5.1 Definition of Reordering

The IETF and the ITU-T share the same assumptions for sequence numbers and expected sequence numbers at the destination. The source node assigns each packet a *sequence number*. The sequence numbers increase monotonically. At the destination node a three tuple (i, $s[i]$, $s'[i]$) characterizes each packet arrival. Index i gives the arriving packet position at the destination. $s[i]$ denotes the sequence number and $s'[i]$ denotes the expected sequence number of the packet with index i. Like the ITU-T, they distinguish two cases for the value of $s'[i]$.

$s[i] < s'[i]$: packet i arrives reordered. $s'[i]$ remains unchanged, i. e., $s'[i+1] = s'[i]$.

$s[i] \geq s'[i]$: packet i arrives in-order and $s'[i+1] = s[i] + 1$.

Literally, a packet arrives out-of-sequence if there is at least one packet with a larger sequence number arriving prior to it. The first packet of a flow is always in-order by definition.

3.2.5.2 Reordering Ratio

The reordering ratio represents the ratio of the reordered packets to the total number of received packets. It indicates reordering in a network and may advise the evaluation of further reordering metrics. Let $\tilde{N}_P$ be the number of reordered packets at the destination and N_P the number of total arrivals, then the reordering ratio becomes $R = \tilde{N}_P / N_P$.

i	1	2	3	4	5	6	7	8
$s[i]$	2	4	6	7	3	5	8	9
$s'[i]$	2	3	5	7	8	8	8	9
Reordered	-	-	-	-	Y	Y	-	-
e_i	-	-	-	-	3	3	-	-
Buffer size	-	1	2	3	2	-	-	-

Table 3.5: IETF RFC 4737 reordering metrics

3.2.5.3 Reordering Extent Metric

The packet extent metric estimates the buffer places needed to restore packet order at the receiver. It equals the number of packet arrivals between the packets nominal in-order position and the packets actual arriving position. For an in-order packet, the reordering extent is undefined. Formally, the extent e_i for a reordered packet i is

$$e_i = i - \min_{j<i} \{j : s[j] > s[i]\} \tag{3.4}$$

Table 3.5 illustrates this metric. The first row gives the position in the arrival sequence. The second row gives the arriving packet sequence number. The third row indicates the next expected value and the fourth row indicates which packet arrives reordered according to the definition. In this example, only packets 3 and 5 arrive reordered as they face a larger expected value counter. The fifth row depicts the calculated reordering extent value for a reordered packet and the last row denotes the actual number of packets in the buffer, assuming a complete in-sequence delivery.

The reordering extent is only defined for reordered packets. Packets 2,4,6,7 are in-order per definition and show an undefined extent value. While for packet 5 the reordering extent metric determines a necessary buffer size of 3, the actual buffer size would be only two, as the receiver forwards packet 4 as soon as packet three has arrived. Consequently, the extent metric overestimates the required buffer size for certain arrivals. A modification of the reordering extent metric algorithm improves the algorithm and removes the overestimation. The new definition of the extent metric without overestimation becomes:

$$e'_i = \sum_{j=ie}^{i-1} \{s[j] \text{ where } s[j] > s[i]\} \tag{3.5}$$

Literally, the modified extent metric decrements the original extent metric by 1 for each sequence number larger than $s[i]$. This approach yields to the true buffer size for any reordering pattern at the receiver. It is comparable to the proposal by Piratla et al. in Section 3.2.2. Both show the same complexity in calculation. This topic is discussed in the evaluation section. The following metrics base on the reordering extent metric and derive related measures for out-of-order arrivals.

3.2.5.3.1 Latetime metric

While the extent metric calculates the offset of a packet in terms of packets, the *latetime metric* determines the extent in time. If $\mathrm{dt}(s[i])$ denotes the arrival time of packet $s[i]$ at

the destination, the following defines the latetime metric $\mathrm{lt}(s[i])$ of packet $s[i]$.

$$\mathrm{lt}(s[i]) = \mathrm{dt}(i) - \mathrm{dt}(i-e) \tag{3.6}$$

The time to wait until an in-order forwarding is possible equals the difference of the arrival instances of the packet determined for the reordering extent $i-e$ and the packet itself. The latetime metric estimates the waiting time in the buffer before forwarding. As this metric relies on the extent metric, it also overestimates the time for playout. Replacing the term $\mathrm{dt}(i-e)$ by the modified term of the extent metric leads to the correct buffer time.

3.2.5.3.2 Byte-offset metric

In parallel to the latetime metric and the extent metric, the byte-offset metric determines the receiver buffer size in terms of bytes. If $\mathrm{bo}(s[i])$ denotes the byte offset metric and $\mathrm{pl}(s[i])$ the payload of packet i, it is defined by the following equation:

$$\mathrm{bo}(s[i]) = \sum_{j=i-e}^{i-1} \{\mathrm{pl}(s[j]) \text{ where } s[j] > s[i]\} \tag{3.7}$$

Literally, the byte-offset accumulates the payload sizes from packets waiting in the buffer due to packet extents. This metric corresponds to the corrected extent metric, indicating the real buffer size in byte.

3.2.5.4 Gaps Between Discontinuities

A packet arrives in-order if $s[i] \geq s'[i]$. If the sequence number exceeds the next expected sequence number, the next expected sequence number increases depending on the arriving sequence number. For instance, in Table 3.4 the next expected value skips the value of 4 because of the in-order arrival of packet 5. This creates a *sequence discontinuity* in the sequence of arriving sequence numbers. The sequence discontinuity indicates a missing packet, which may be lost or will arrive later. The position of a discontinuity is marked with a $'$, i. e., i' and j'. In the following example $i' > j'$ holds.

The gap metric and the gaptime metric measure the interval between these discontinuity events in units of packets and time. The formal definition of the gap requires a discontinuity event for packet i' and packet j' without an intermediate discontinuity event. Then the next equations show the gap metric and the gaptime metric. To avoid confusions for the first packet, the equations distinguish two cases.

$$\mathrm{gap}(s[j']) = \begin{cases} i' - j' & \text{if } s[j'] > 0 \\ 0 \quad \text{otherwise} \end{cases} \tag{3.8}$$

$$\mathrm{gaptime}(s[j']) = \begin{cases} \mathrm{dt}(i') - \mathrm{dt}(j') & \text{if } s[j'] > 0 \\ 0 \quad \text{otherwise} \end{cases} \tag{3.9}$$

```
while(packets arrive with seq. nr. s)
{
  p++;
  /* s is in-order */
  if (s >= NextExp) then
          r++;
          a++;
  /* s is reordered */
  else
          q+= r*r;
          r = 0;
          x++;
}
```

r the run counter, the number of in-order packet arrivals between two successive out-of-order packets,

a the total number of in-order packets,

x the number of reordered packets,

q the sum of the squares of the run counter and

p the number of arrived packets ($p = a + x$).

Table 3.6: Pseudo code of the freerun metrics

Literally, the gap metric measures the number of packet arrivals between successive discontinuity events. The gaptime metric measures the time between these events. For evaluation purposes, a histogram of the gap and gaptime metric may illustrate the frequency of the gap values.

3.2.5.5 Freerun Metrics

Completing the metric proposals of the previous section, the *freerun metric* characterizes the number of successive in-order arrivals. This metric quantifies the variability of out-of-order packet arrivals. A small mean value indicates frequent reordering events, while a large mean value points to rare reordering events. Besides the mean value, the variation of this metric gives deep insights into the reordering pattern. The pseudo code in Table 3.6 defines the application of these variables. Each in-order packet arrival increments the run counter r. If the arriving packet is out-of-order, the square of r cumulates in q and r receives the value 0. Each out-of-order arrival increments the counter x, which represents the absolute amount of reordered packets. Based on these measures, the IETF proposes the following metrics. The percent of reordered packets is $(x/p) \cdot 100\%$. The average free run without an out-of-order packet arrival is $\bar{a} = a/x$. The counter q indicates the variation of the free run metric. The relation of q/a and $\bar{a}$, i. e., $\frac{q/a}{a/x}$, quantifies this variation.

The next example illustrates the necessity of the last metric to differentiate special reordering patterns. [RFC 4737] considers two packet arrival sequences of 36 packets each. Sequence 1 includes 36 packets with 3 runs of length 11, i. e., after 11 packets one reordered packet arrives. Sequence 2 includes 36 packets with 3 runs, 2 of them of length 1 and one of length 31, i. e., the reordered packets arrive with only one packet in between. Then a sequence of 31 in-order packets follows. The following two columns depict the

values of the reordering metrics of both sequences.

Sequence 1	Sequence 2
$p = 36$	$p = 36$
$x = 3$	$x = 3$
$a = 33$	$a = 33$
$q = 3 * (11 * 11) = 363$	$q = 1 + 1 + 961 = 963$
$\bar{a} = 11$	$\bar{a} = 11$
$\frac{q}{a} = 11$	$\frac{q}{a} = 29.18$
$\frac{q/a}{a/x} = 1.0$	$\frac{q/a}{a/x} = 2.65$

The reordering metrics differ in the 4th and the last two rows. The variability dominates the fraction q/a and its relation to a/x. The left column describes a harmonic and periodic reordering case, whereas the right column quantifies an asymmetric peak reordering event. The parameter r distinguishes both cases and serves as a parameter to study the influence on long lasting connections using, e. g., TCP. Long lasting TCP connections are relevant because of the slow-start or fast retransmit algorithms, TCP may apply. Then, TCP reduces its sending window, which reduces the throughput and decreases performance.

3.2.5.6 TCP Relevant Metric

The IETF TCP relevant metric quantifies the violation of the TCP duplicate acknowledgement threshold. It defines that a n_r-reordered packet triggers n_r dup-acks. If there is a set of n_r packets directly preceding packet i and $s[i]$ is less than the sequence number of each of these packets, n_r dup-acks are triggered. Formally, packet i is n_r-reordered if the following condition holds:

$$s[j] > s[i] \quad \forall j \in \{k : i - n_r \leq k < i, k \in \mathbb{N}\} \tag{3.10}$$

According to the definition in Eq. (3.10), it is obvious that every 2-reordered packet is also 1-reordered. For an illustration of this metric, a density function of the maximum possible n_r value may be applied, i. e., $\max\{n_r\}$ of (3.10). Table 3.7 depicts an example of this metric. The first row indicates the arrival sequence, while the second row depicts the sequence number. The third row indicates the maximum n_r-reordering metric. Packet 5 faces two packets 6, 7 with larger sequence numbers directly before it and consequently receives a n_r-reordering value of 2. Packet 3 faces three packets 6, 7, 5 with a larger sequence number and receives a value of 3. For packet 4 the situation is different. There is no packet with a larger sequence number directly before it.

The n_r-reordering metric indicates the number of possible duplicate acknowledgements for TCP as well as for SCTP and DCCP, [RFC 2960,RFC 4340]. The IETF drafts recommend a value of three for the duplicate acknowledgement threshold of these transport layer protocols. Thus, this metric provides a quantitative measure for possible retransmission. Note that this metric gives a hypothetical upper bound for TCP retransmissions, as the

i	1	2	3	4	5	6	7	8
$s[i]$	2	6	7	5	3	4	8	9
Reordered	-	-	-	Y	Y	Y	-	-
n_r	-	-	-	2	3	0	-	-

Table 3.7: IETF RFC 4737 n_r-reordering metric

equations do not consider the feedback mechanisms of TCP. The feedback mechanisms of TCP reduce the sending rate if it experiences packet loss or packet reordering. With a reduced sending rate, the experienced reordering may vanish due to changes in the traffic characteristic of the sender.

3.2.5.7 Robustness

This section studies the robustness of the above metrics with respect to the network anomalies of packet reordering and packet duplication. For the definition of reordering, the same arguments hold as in Section 3.2.4. A lost packet does not interfere with the definition of reordering, as the expected sequence number increments continuously and a packet is determined as reordered only on basis of the relation between actual and expected sequence number. Duplicate packets may change the metric values before they are discarded.

The remaining metrics are robust against packet losses since they only consider the arrival position rather than the actual sequence number. All dependencies on the sequence numbers are relative and do not rely on any specific sequence number. Consequently, these metrics are stable with respect to missing packets. The case of packet duplication again requires additional effort at the receiver side. The receiver needs to implement mechanisms to avoid erroneous reordering metrics due to duplicate packets. As introduced before in this section, the metric standardized by the IETF is robust against packet loss and duplication.

3.2.5.8 Evaluation

The IETF reordering metrics aim to quantify the amount of packet reordering in an online measurement scenario. Consequently, the metrics should enable efficient computation. The complexity to determine if a packet is reordered has already been discussed in the previous sections. The algorithm only requires a comparison of the sequence number with the next expected value.

The computation of this metric requires knowledge on the last sequence number arrived. As reordered packets are the results of discontinuity events, it is sufficient to store the last discontinuity events instead. At a discontinuity event, the arriving sequence number is larger than the expected sequence number.

The data stored for discontinuity events include the sequence number of each reordered packet, the arrival time and the number of arrived packets up to this moment. For

each reordered packet, the receiver searches the list of discontinuity events to find the appropriate reference sequence number to satisfy Eq. (3.4). The location of the sequence discontinuity provides the basis for the extent value, the latetime and the byte-offset metric. The overall complexity to calculate these metrics increases with the number of discontinuity events. If a packet arrives reordered and fills a gap of a discontinuity event, the implementation may remove the discontinuity event from the list. This reduces the average number of entries in this list.

The byte-offset metric requires detailed knowledge of the packets including their payload size. For this metric, the receiver has to maintain a buffer to store this information. For each reordered packet, the receiver searches the buffer and calculates the required buffer size necessary for re-sequencing. During the same operation, the receiver may also calculate the modified extent metric of Eq. (3.4). The procedure described here is equivalent to the techniques of Section 3.2.2. The complexity increases with the buffer size M and results in a complexity of $O(N + \gamma NM)$ for each packet of the flow size N. γ is the probability of a packet to face a full buffer.

For each discontinuity event, the gap metric requires the preceding discontinuity event. The calculation of this metric reduces to a simple difference in packet arrival numbers and timestamps. The memory consumption is constant, as the receiver only stores the last discontinuity event. In O-notation this results in O(1).

Table 3.6 shows the complexity of the freerun metric. It includes continuous updates and resets of counters. At the end of a stream, the receiver may calculate the relations of selected counters once. Summarizing, the memory consumption and the complexity in the evaluation are constant per packet arrival, i. e., O(1).

The remaining TCP-related metric requires an effort comparable to the byte-offset metric. There the receiver needs to store a certain number of the last packets arrived. If a packet is reordered, the receiver searches this list from the end until it finds a smaller sequence number. Consequently, the memory consumption grows linearly with the size of the list W, i. e., O(W). For searching a linear list, the effort is also about O(W).

3.2.6 TCP Specific Metrics

This section introduces other application specific reordering metrics proposed in the literature. They all focus on TCP, either determining its throughput or using TCP as measurement instruments to evaluate packet reordering in the network. In the following paragraphs, the term reordering metric broadens as the focus shifts towards the impact of the transport protocol TCP.

In [94], Laor et al. study the impact of packet reordering on TCP. For this, they reordered packets within a stream. Their reordering setup involved two parameters. The first parameter denotes the number of packets to skip, while the second denotes the number of packets to let pass. With this setup, they evaluated the TCP performance related to different parameterizations. They got insights into the TCP behaviour, but the setup

does not allow variable reordering patterns. They also do not provide a metrics definition to classify the reordering pattern precisely.

In [90], Bellardo et al. propose an active measurement technique to reliably estimate the amount of reordering in a network. They classify a TCP segment as reordered if it arrives later than expected. This is similar to the reordering definition defined earlier. However, they do not provide a formal description of this definition and lack a quantification of any other reordering characteristic. As a measurement tool, they use a TCP implementation. They study the TCP behaviour at the client side and quantify the amount of reordering using an arbitrary TCP server. Additionally, they distinguish forward and backward path reordering.

In addition to the work by Bellardo, Luo et al. propose in [91] four active measurement techniques using TCP. They are able to distinguish all four possible reordering events (out-of-order/in-order in forward/backward direction) while applying the reordering definition by Bellardo and others on the increasing sequence number. Thus, they provide no new findings on reordering metrics but focus their work on the application of TCP to detect reordering in backward and forward direction.

In [4], Jaiswal et al. classify packet reordering in backbone IP links. They point out three different reasons for retransmissions including retransmission timeout, fast retransmit, or packet duplication. They provide a methodology to identify the reason for out-of-sequence packets from a trace by using the header information of the IP and TCP protocols. As their focus has been on TCP only, they did not propose any metric to classify out-of-sequence patterns with respect to other protocols or properties.

In [95], Mellia et al. analyse network anomalies (packet reordering and duplication) with respect to a TCP trace. They logged the TCP/IP header of packets of a TCP connection in both directions to record data as well as acknowledgements. Their proposal includes a heuristic scheme to classify the expected anomalies in network duplicates, packet reordering, unnecessary retransmissions by the retransmission timeout, fast retransmit or flow control. This requires re-engineering the TCP stack of the end-systems. They assumed standard conform implementations, which is in general not applicable as each protocol version and operating system shows slight differences.

3.3 Concluding Remarks on Packet Reordering Metrics

Previous sections have introduced and classified the proposals for reordering metrics provided in literature. The large number of publications in this field indicates a necessity to classify and to standardize metrics for changes in the packet sequence in a concise way. As the standardization bodies ITU-T and IETF attended to this topic by recommendations and RFCs, they laid the bases for comprehensive studies. These standards enable protocol performance studies under reproducible conditions using a statistical description of reordering pattern.

This thesis uses the metrics as defined by the IETF, i. e., reordering definition, reordering

ratio, reordering extent and n_r-reordering metric to evaluate reordering in packet-based networks. The reordering extent metric determines the buffer size at the receiver needed to re-sequence packets. The n_r-reordering metric indicates the effect on the TCP protocol or any other protocol implementing TCP-like feedback mechanisms. This metric serves as an indicator for the expected protocol performance. The remaining metrics rely indirectly on the extent metric or the n_r-reordering metric. The detailed investigation on the remaining metrics is out-of-scope of this thesis. The next sections concentrate on the analytic derivation of both metrics with respect to certain traffic models.

4 A Novel Analytic Reordering Model

This chapter imposes a novel analytic reordering model for the description of burst and packet reordering patterns. This chapter firstly introduces the requirements of a reordering model and the relation between burst and packet reordering in networks showing packet assembly. Besides the model of this thesis, literature proposes a large number of queuing models to describe the reordering phenomenon. The first section presents a review on the related work on reordering models. It highlights the differences of the previous work and this thesis.

The second section introduces the principle structure of the queuing model proposed in this thesis. It represents a disordering network of parallel queuing systems, which is able to change the packet sequence. Besides the queuing model, this section also presents the equivalent simulation model, which validates the analytic findings. The introduction to the methodology to analyse the proposed model closes the second section.

The third section presents the major contribution of this chapter. It provides the formal analysis of the queuing model for two selected traffic models, i. e., deterministic and probabilistic, i. e., Poisson traffic. It derives the reordering ratio, reordering extent metric and the n_r-reordering metric for burst level. It turns out that the reordering model for deterministic traffic serves as an upper bound for the reordering ratio (cf. Section 5). This section illustrates the findings with the results of the simulation model.

The last section derives the packet reordering metrics in a network showing packet assembly, i. e., hierarchical reordering. These metrics depend on the burst reordering metrics as well as the packet per burst distribution. Additionally, it reviews the different packet per burst distributions depending on the assembly scheme. Also for hierarchical reordering, this section validates the findings with the simulation model.

4.1 Introduction

This section introduces the generic reordering scenario describing hierarchical packet reordering. Thereby, packet assembly enables hierarchical reordering if the underlying network technology implements multi-path routing. It introduces the network scenario as well as the requirements on the formal description of a reordering model.

4.1.1 Requirements on Formal Reordering Models

Chapter 2 showed manifold technical realizations, which may lead to packet reordering, e. g., multi-path routing. On an abstract view, an out-of-sequence arrival of two arbitrary packets may only occur if two network parameters correlate, i. e., the packet inter-arrival time and the packet delay. The next paragraph formulates the mathematical equivalent for this condition.

The formal analysis considers two packets, packet I with sequence number i and packet J with sequence number j where $i < j$. The random inter-departure time between packet I and packet J is T_{IJ}. In a reordering scenario, each packet receives a different delay D, where $d > 0$ on the path from source to destination. Packet I receives the random delay D_I, while packet J receives the random delay of D_J. According to the IETF definition of reordering in [RFC 4737] and Section 3.2.5, packet I arrives out-of-sequence with respect to packet J if the following equation holds:

$$d_I > d_J + t_{IJ}, \quad \text{where } d_I, d_J \geq 0, t_{IJ} > 0 \tag{4.1}$$

Literally, packet I arrives after packet J if its delay is larger than the delay of packet J together with the inter-departure time. Eq. (4.1) shows the fundamental principle of reordering in the network. It combines two network parameters, the network delay packets experience and the traffic characteristic (i. e., inter-arrival time). Only if both parameters correlate, an out-of-sequence arrival may occur. Summarizing, any reordering model requires both parameters for an analysis of the emerging reordering pattern.

4.1.2 Hierarchical Packet Reordering

This section introduces briefly the relation between burst reordering and packet reordering and the fundamental procedure to obtain both. Scenarios, which show packet aggregation, produce reordered packets, if the bursts carrying these packets arrive out-of-sequence. The remainder of this thesis denotes this process as *hierarchical packet reordering* as reordering occurs on different layers, i. e., burst and packet layer.

Hierarchical packet reordering may only occur if two conditions are fulfilled. First, an assembly process assembles packets into bursts and second the bursts arrive reordered at the destination. As the assembly process is independent of the subsequent burst reordering, it is possible to study both requirements independently of each other. The assembly process defines the number of packets per burst and the burst reordering defines the reordering pattern of the bursts. Taking both results together enables the calculation of the packet reordering metrics. As the packet reordering metrics represent a transformation of the burst reordering metrics, this thesis first studies the burst reordering metrics and second derives the packet reordering metrics in Section 4.4.1.

4.1.3 Related Work on Disordering Networks

The research community studies the problem of out-of-sequence packet arrivals since the first times of packet-based networks. They were motivated by the devices, which expect

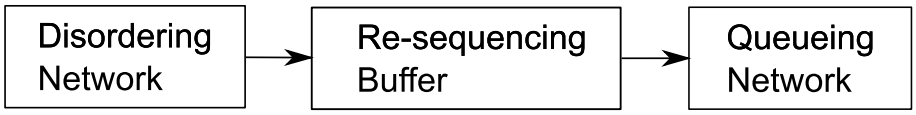


Figure 4.1: Generic model for re-sequencing

packets in-order but face out-of-order arrivals due to any communication network. For an in-order delivery they introduce a re-sequencing buffer. Figure 4.1 depicts an abstract model of this environment. It is similar to the first figure in [96]. The disordering network represents the communication network, which may change the packet order. The re-sequencing buffer stores these packets but forwards them only in-sequence to the subsequent queuing network. The challenging problem is the relation between the disordering network and the buffer. Subject of research are the additional waiting time in the buffer and the buffer size itself. Various studies apply methods of the queuing theory with special synchronization constraints, i. e., in-order delivery to the queuing network. The next paragraph highlights selected publications in this area.

Queuing theory models the disorder network as an A/B/C system (using Kendalls notation) with first-in first-out (FIFO) scheduling principle. Therein, A represents the arrival process, B the service process and C the number of parallel servers. The choice of A, B and C may produce out-of-sequence arrivals. Therein, a necessary condition for C is that it must be larger than 1, otherwise no reordering occurs[1]. An alternative is that several models operate in parallel. Then the number of parallel servers is also greater than 1. The major contribution of these studies is the analysis of the end-to-end waiting time and the waiting time in the re-sequencing buffer to satisfy synchronization constraints. Besides the time aspects, also the size of the re-sequencing buffer was subject to queuing theoretical studies. Table 4.1 provides an overview on these studies. The first column depicts the applied model, the second the reference and the third column gives a short description of the findings. As this list is neither complete nor exhaustive, the reader is referred to Baccelli et al., who provide in [96] a broad survey on these studies and their findings. Selected publications include the studies of Xia, Jean-Marie, Chowdhury, Harrus and Yum.

Xia et al. model in [97] the feedback mechanism of ARQ in case of reordering. They model the disordering network as a D/GI/∞ model. For the random service time distribution, they choose the negative exponential and Pareto distribution. Their model was able to determine the waiting time distribution as well as the required buffer size for both delay distributions.

Jean-Marie et al. analyse in [98] the sorjourn time and the re-sequencing delay of K parallel M/GI/1 systems. They found their model converging to a M/GI/∞ model if the number of parallel system increases and the probabilities to join one of the K models are equally distributed.

Chowdhury analyses in [99] the inter-dependence between the mean re-sequencing and total delay and the number of severs K of an M/H_2/K two-stage hyperexponential model.

[1] Non-FIFO service principles may also enable reordering with only one single server

He calculates the mean delay a reordered packet has to wait.

Harrus et al. analyse in [100] the properties of an M/G/∞ model. They focus on the reordering delay, the number of packets in the re-sequencing buffer and the total sojourn time. Thereby, they propose an approximation for the general model to obtain these measures. Later, Baccelli et al. propose in [101] an improved approximation avoiding bulk arrivals after leaving the re-sequencing buffer.

In [102], Yum et al. analyse an M/M/K model as a potential disorder network. They found the distributions for the total time and the waiting time to re-sequence the out-of-order packets. Additionally, they observe a larger coefficient of variation of the traffic leaving the re-sequencing buffer then entering the re-sequencing buffer.

In general, these studies focus on special queuing models of the disordering network and derive or approximate the relevant metrics of the waiting time and the buffer size for re-sequencing. These queuing theoretical analyses require a direct coupling between the arrival process and the service process. Thus, each combination of both processes requires a new queuing model and a new analysis. In protocol engineering, the packet arrival process mostly cannot be described by an accurate model (exceptions are constant bitrate traffic for instance). Consequently, engineering requires simple and robust reordering models, which enable broader statements and potential worst-case estimations independent of the traffic model. Besides this, the analysis on the waiting time and the buffer size is insufficient for protocol engineering to derive the impact on the protocol performance. As they approach the problem from a queuing theoretical approach, they miss the classification and application of special reordering metrics. The absence of metrics does not allow re-engineering the reordering patterns to parameterize any network emulation for packet reordering. Some protocol studies, i. e., TCP, require the exact description on the reordering pattern, e. g., n_r-reordering metric, for a comprehensive discussion on the performance.

The model presented in the next section closes this gap. It provides an analytic model to evaluate the exact reordering pattern for selected traffic models and enables worst-case estimations for any other traffic model. Starting from a given reordering pattern, the model is also able to derive the parameters for network emulation to reproduce the reordering pattern for protocol engineering. Thereby, the reproduced reordering pattern is independent of the packet arrival characteristic. The idea of the reordering model as well as some early findings have already been published by the author in [103–106].

4.2 Reordering Model

The previous sections introduced the three major parameters of a disordering network: the arrival process, the service process and in case of hierarchical reordering the packet per burst distribution. The novel model presented in this section introduces a new coupling of these three parameters and applies combinatorial mathematics to determine the reordering metrics of packets and bursts. The proposed model corresponds to $m + 1$ parallel

Disordering network model	Reference	Studies on
D/GI/∞	Xia et al., [97]	Buffer size and waiting time for neg. exp. and Pareto delay distribution.
K $\times$ M/GI/1	Jean-Marie et al., [98]	Buffer size, waiting time distributions
M/H_2/K	Chowdhury et al., [99]	Mean waiting time
M/M/∞	Kamoun et al., [107]	
M/G/∞	Harrus et al., [100], Baccelli et al., [101]	Buffer size, waiting time, system occupancy
M/M/K	Yum et al., [102]	Total and re-sequencing delay distributions

Table 4.1: Related work on Disordering Networks

$\cdot$/D/∞ systems, where m is the number of parallel queuing servers and $\cdot$ represents either deterministic or Poisson arrivals (cf. Figure 4.2). The service process is deterministic.

The first section introduces this theoretic queuing model for the disordering network. It shows the correlation between the inter-arrival time of the arrival process and a discrete service time. The second section introduces the corresponding simulation model of the queuing model. It serves as a reference for the analytic findings. The last section imposes the novel methodology to derive the reordering metrics on the novel disordering network.

The following sections use the term burst to identify a single datagram. The datagram may represent one block of data, i. e., packet, frame, aggregate. Later sections on hierarchical reordering differentiate the terms packet and burst.

4.2.1 Queuing Model of the Disordering Network

This section presents the disordering network, i. e., the reordering model. Figure 4.2 depicts the novel reordering model (disordering network) applied in this thesis. The model consists of $m+1, m \in \mathbb{N}_0$ alternative branches, i. e., abstract links l. Each abstract link l_k represents an abstract end-to-end path showing a different delay D_k. Besides this, l_0 shows no extra delay. In a network, this may relate to the shortest path from source to destination node. D_k accumulates delays resulting from different paths from source to destination, i. e., multi-path routing or contention resolution schemes in OBS. Starting from a continuous end-to-end delay distribution, each D_k discretises the distribution. Consequently, D_k is a discretisation of the experienced end-to-end delay in a network. On each abstract link, an infinite number of servers with a deterministic service time represent this delay. Summarizing, each abstract link represents a $\cdot$/D/∞ model, with different D per link, i. e., D_k.

At the beginning of the disordering network, a random splitting process decides for each burst, which path to take. Thereby, the probability to follow an abstract link differs. p_k

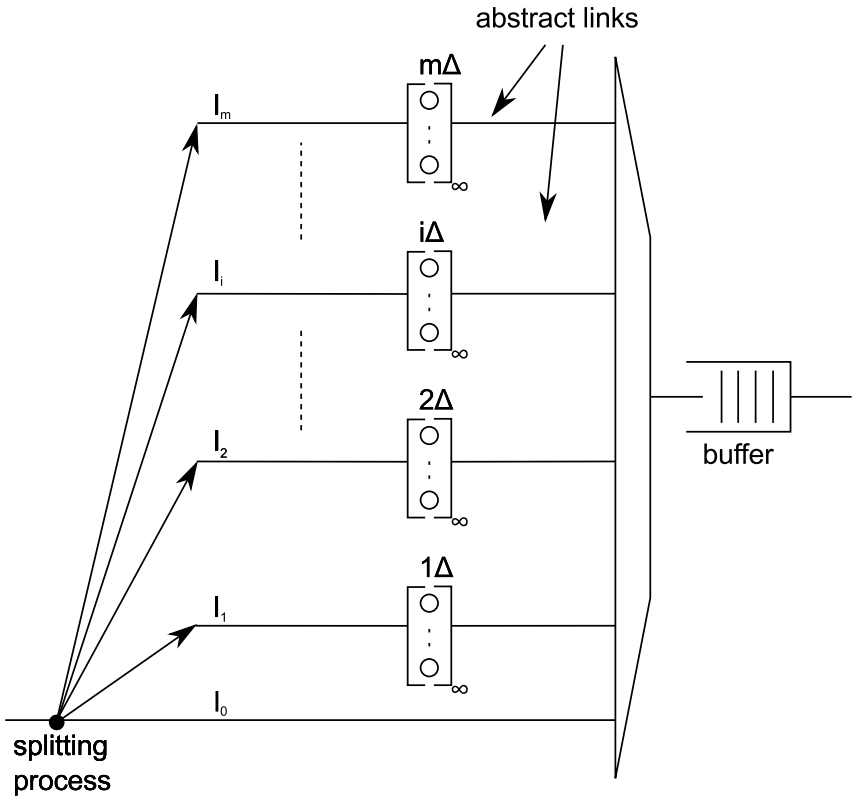


Figure 4.2: Disordering network

denotes this probability to follow abstract link l_k. Each burst follows independently one of these abstract links randomly. The probability reflects the distribution of the end-to-end delay of a certain end-to-end connection.

The arguments of the previous section (cf. Eq. (4.1)) yielded to the correlation of the inter-arrival time and the delay of two subsequent bursts. This queuing model obeys this correlation and defines a basic delay unit Δ. The basic delay unit Δ equals the mean inter-arrival time, i. e., $\Delta = \mathsf{E}[T_{IJ}]$. Further studies later in this thesis show that this assumption enables worst-case considerations as well as enables a suitable parameterization of any network emulation. The choice of Δ approximates the original end-to-end delay distribution. For small Δ, the model approximates the original end-to-end distribution.

Starting from abstract link 0 with no additional delay, the first abstract link delays a burst by Δ. The k^{th} abstract link delays the burst by $k\,\Delta$, i. e., $D_k = k\,\Delta$. Consequently, a 3-tuple $(k, p_k, k\,\Delta)$, $0 \leq p_k$, $0 \leq k \leq m$ characterizes each abstract link l_k: the link number k, the probability p_k to follow l_k and the burst delay $k\,\Delta$ as an integer multiple of the basic delay unit Δ. Further the law of total probability holds for all abstract links: $\sum_{k=0}^{m} p_k = 1$.

If network emulation implements this model, it is necessary to determine the latency and the buffer requirements of this model. This reordering model increases the latency because of the extra delays, i. e., the service time of an abstract link. The next paragraph determines the mean latency of this queuing model as well as the mean number of bursts in the system, i. e., total required buffer size. The mean inter-arrival time of the incoming

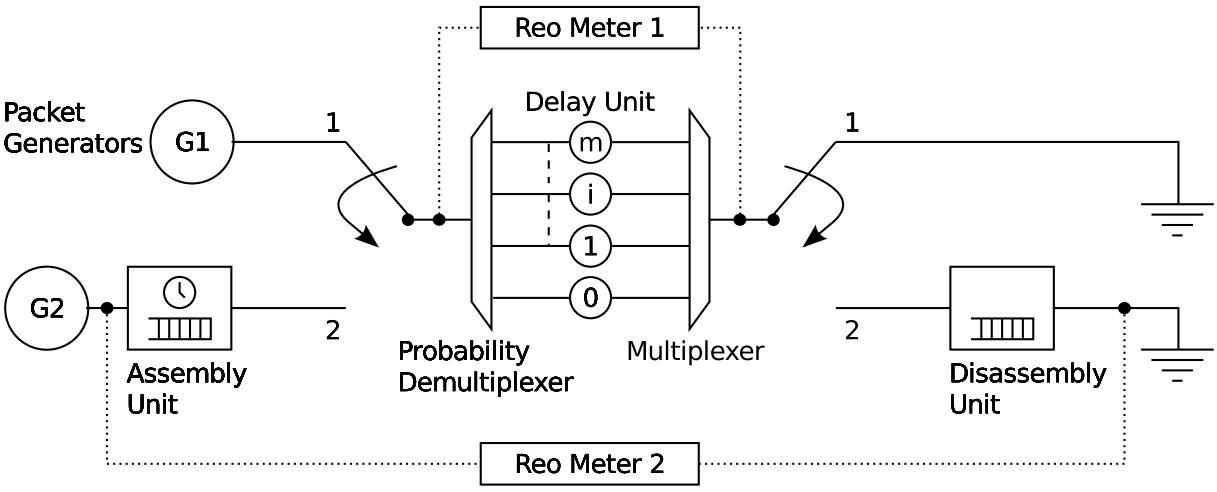


Figure 4.3: Simulation model

traffic is $\mathsf{E}[T] = 1/\lambda$ if λ is the mean arrival rate. A bursts following path i receives an additional delay of $i\,\Delta$ according to the definition of the reordering model. This bursts follows path i with probability p_i. Averaging over all paths the mean latency for m additional delay links, the expected mean value for the latency W results in

$$\mathsf{E}[W] = \Delta \sum_{i=1}^{m} p_i\, i \tag{4.2}$$

Little's Law ($\mathsf{E}[L] = \lambda\,\mathsf{E}[W]$) estimates the mean of the random number of bursts L in the model to:

$$\mathsf{E}[L] = \frac{\Delta}{\mathsf{E}[T_{IJ}]} \sum_{i=1}^{m} p_i\, i = \sum_{i=1}^{m} p_i\, i \tag{4.3}$$

The mean number of bursts in the system simplifies as the arrival rate and the basic delay unit correlate. Consequently, the number of bursts in the system only depends on the distribution of the probabilities p. Another aspect of the model is its stability with respect to the traffic rate. For stable operation of the system the following equation must hold: $\mathsf{E}[W] \,/\, \mathsf{E}[T_{IJ}] < 1$.

4.2.2 Reordering Simulation Model

This section briefly introduces a simulation model, which implements the disordering network of the previous section. This simulation model verifies the analytic findings of the queuing model. Additionally, the following sections refer to simulation results obtained with this model. Figure 4.3 depicts the simulation model. It consists of two packet generators at the left, the queuing model in the middle and sinks at the right. The model allows both, to simulate reordering on a single layer as well as hierarchical reordering including burst and packet reordering. The model visualizes this property by two switches on both sides of the queuing model with (1) and (2), respectively.

If both switches are in position (1), the simulation model enables single layer reordering studies. The packet generator G1 creates packets, which follow a defined traffic model (packet length and inter-arrival time). The generated packets first meet a probability demultiplexer, where the packets follow each output with a certain probability. The demultiplexer realizes the random splitting process of the model. Each port of the multiplexer connects to a server, which shows the same deterministic service time as in the model. The number of servers is infinite. After the delay unit, a multiplexer combines the different abstract links together to one output port. Finally, the multiplexer connects to a sink to terminate the packets. A reordering meter (reo meter 1) records the packet sequences before entering the abstract links and after the abstract links. It implements the IETF reordering metrics of Section 3.2.5 and its results serve as the major criteria for validating the analytical findings.

If both switches are in position (2), a study of hierarchical reordering is feasible, too. The packet generator G2 creates packets following a certain traffic model. These packets meet an assembly unit, which assembles packets to bursts. The actual number of assembled packets depends on the assembly strategy and the packet traffic characteristics. Besides others, it implements the assembly strategies based on time, size and the combination of both. Köhn and Hu describe in [108] the assembly unit with its functionality and basic mechanisms. Similar to the first scenario, the bursts meet the abstract links. At the end of the abstract links, they meet the disassembly unit releasing the assembled packets. For a precise study of the reordering pattern on the packet layer, a second reordering meter (reo meter 2) records the packet arrival pattern between the generator and after the burst disassembly unit. Finally, the packets terminate in the sink.

A classification of the simulation parameters leads to the three major parameters: traffic model for the generators, (optional) burst assembly strategy and parameterization of the queuing model. The traffic models include the parameterization and the configuration of the packet generator units, inter-arrival time and packet length distribution, respectively. The burst assembly process requires an assembly process and the parameters of it, timer and/or size thresholds, respectively. The parameters of the queuing network require the number of abstract links and the basic delay unit as well as the probability distribution among these links.

4.2.3 Applied Methodology

This section imposes the methodology to analyse the queuing model of Section 4.2.1. Figure 4.4 depicts the resulting reordering scenario of the disordering network with $m = 4$ alternative links. The bursts enter the disordering network from right to left. The arrows indicate the relative change of their position *at the destination*. The distance between the original and the new position indicates the delay of the abstract links. The evaluation of the reordering pattern considers one selected burst, i. e., the *test burst*. All considerations include the position of the test burst and all the other bursts. Thereby, any of the other bursts may serve as a test burst, too. The figure highlights the test burst at the splitting point. For clarity, the figure shows only the possible delays of the test burst, but any other burst may also follow certain abstract links and receive additional delays. This reordering

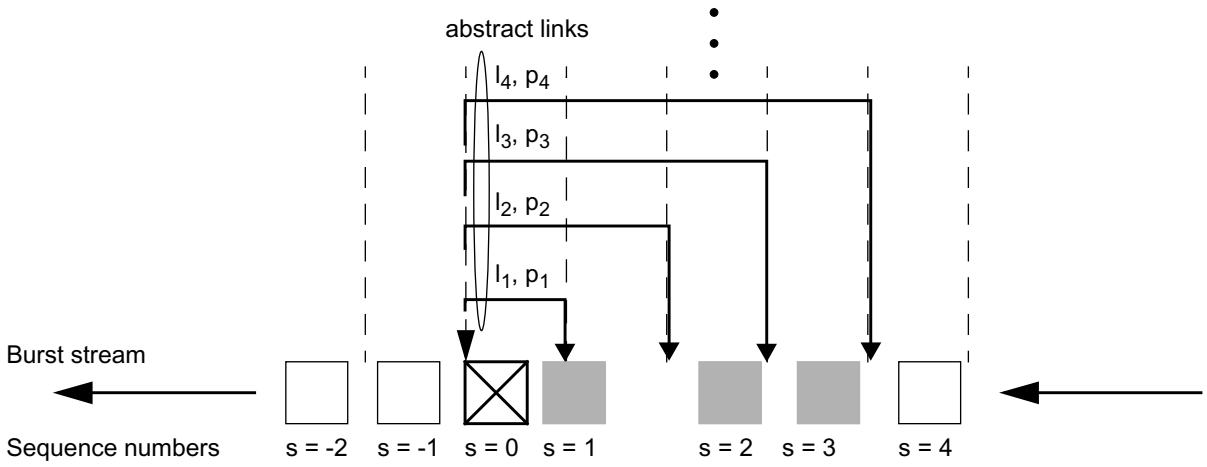


Figure 4.4: Reordering process

scenario introduces four kinds of bursts:

1. The test burst. Without loss of generality, its sequence number s is $s = 0$. The sequence number also serves as a reference number to identify the burst. The analytic model calculates the reordering metrics for the test burst, with respect to all other bursts.

2. Bursts departing later but arriving earlier than the test burst because of the delay of the test burst (grey).

3. Bursts departing and arriving earlier than the test burst and bursts departing and arriving later than the delayed test burst (white).

4. The located burst. It is a special burst, which shows a larger sequence number than the test burst. Besides this, it always arrives at the location earlier than the test burst. This property makes it a reference burst required for the reordering metrics reordering extent and n_r-reordering metric.

Depending on the arrival process of the bursts, it is possible to calculate explicitly the reordering metrics: reordering ratio, reordering extent and n_r-reordering of Section 3.2.5. The next section provides the analytic expression of these metrics for deterministic and Poisson traffic.

4.3 Application for Selected Traffic Models

Up to now, no assumptions were made on the traffic model of the reordering model except the existence of a mean value. The next two sections calculate explicitly the IETF reordering metrics of Section 3.2.5 for deterministic and Poisson traffic models, respectively. Section 4.3.2.2 elaborates on an extension towards generic traffic models. Chapter 5 will show that the deterministic traffic model serves as upper bound for the

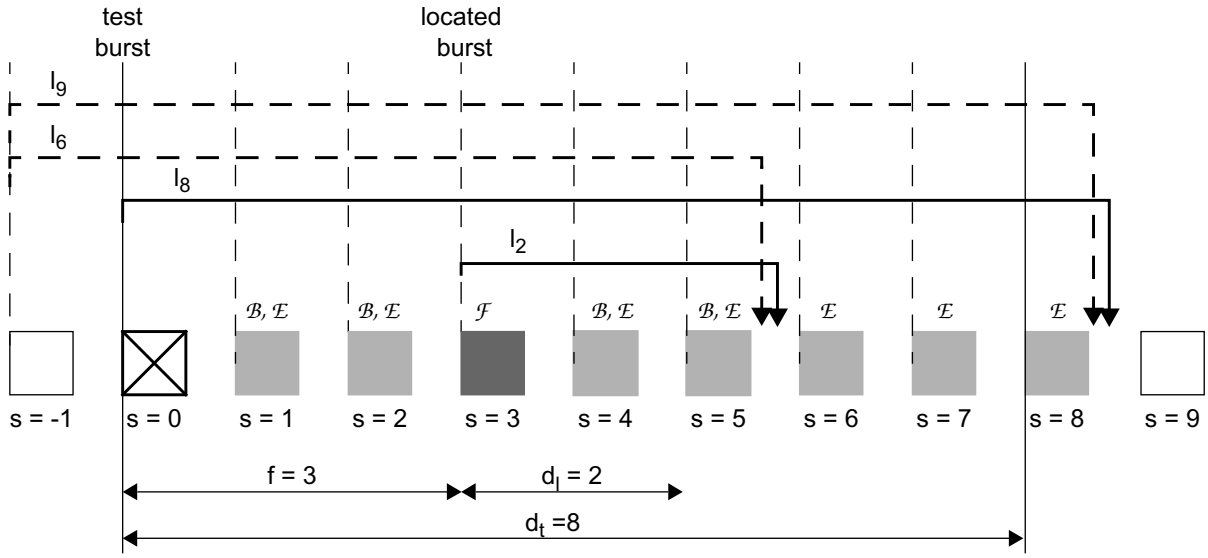


Figure 4.5: Deterministic traffic model

expected amount of reordering in a network. This finding enforces the necessity to consider the reordering model with respect to the deterministic traffic model.

4.3.1 Deterministic Traffic Model

This section analyses the queuing model of Section 4.2.1 and applies the terminology of Section 4.2.3. Inhere, the burst arrival process shows a constant inter-arrival time, i. e., $\mathsf{E}[T_{IJ}] = t_C = \Delta$. This may result from a time based assembly strategy if the packet arrival rate is sufficiently large [46]. In this scenario, the delay per abstract link l_k is $\Delta_k = k\,t_C = k\,\Delta$. The model identifies the bursts by their sequence number s. As the burst delay is proportional to the constant inter-arrival time Δ, the remainder of this chapter abbreviates a delay of $d\,\Delta$ time units by d bursts. The next three sections show the calculation of the reordering metrics of Section 3.2.5, i. e., the reordering probability, the reordering extent and the n_r-reordering metric.

4.3.1.1 Reordering Probability

According to Chapter 3 on reordering metrics, the test burst ($s = 0$) arrives out-of-sequence if the following condition holds: At the destination, there appears prior to the test burst at least one burst with sequence number $s > 0$. Consequently, the reordering probability is a conditioned probability of (a) that the test burst follows any abstract link $d_t > 0$ and (b) that there is at least one burst arrival with larger sequence number than zero before the test burst. Condition (a) assumes an arbitrary burst delay of d_t. Then, there are d_t candidate bursts, which may accomplish condition (b) if they are not delayed.

One may derive the probability of (b) by the complementary probability, that none of the d_t bursts arrives earlier than the test burst. The random variable of the test burst delay

is D_t. The random variable B denotes a burst arrival before the test burst. Then the probability that the candidate burst j, $0 < j \leq d_t$ does not accomplish condition (b) is $\Pr(B = 0|D_t = d_t, J = j) = \sum_{k=d_t-j+1}^{m} p_k$. The sum of probabilities represents all possible abstract links, which lead to a later arrival than the test burst. This sum considers the probabilities of the abstract link delays as well as the location of the burst j.

The joint probability that none of the candidate bursts accomplishes condition (b) at the same time is $\Pr(B = 0|D_t = d_t) = \prod_{j=1}^{d_t} \Pr(B = 0|D_t = d_t, J = j)$. The product consists of a joint probability of all bursts for a later arrival than the test burst. If $\Pr(B = 0|D_t = d_t)$ does not accomplish condition (b) (as all d_t bursts arrive later than the test burst), then due to the law of total probability the complementary probability does. The reordering probability for deterministic traffic $\mathfrak{C}$ results in

$$\begin{aligned} \Pr(\mathfrak{C}) &= \sum_{d_t=1}^{m} p_{d_t} \left(1 - \Pr(B = 0 \mid D_t = d_t)\right) \\ &= \sum_{d_t=1}^{m} p_{d_t} \left(1 - \prod_{j=1}^{d_t} \sum_{k=d_t-j+1}^{m} p_k\right) \end{aligned} \tag{4.4}$$

The outer sum considers all possible abstract links of the test burst with the corresponding probability. Within the brackets, the complementary distribution considers the joint probability of no burst arrival before the test burst. The reordering probability equals the reordering ratio, which denotes the ratio of bursts arriving out-of-sequence.

Figure 4.5 visualizes the procedure. The test burst follows abstract link 8 and arrives originally after burst 8. The test burst is considered only out-of-sequence if at least one burst from 1 to 8 arrives before the test burst. Contrary, if bursts 1 to 8 arrive later than the test burst, the test burst arrives in-order. The probability that burst 1 arrives later than the test burst is the sum of probabilities of abstract link 7 and beyond. Burst 2 arrives later than the test burst if it follows abstract links 6, 7, up to m. Burst 8 arrives later than the test burst if it does not follow abstract link 0. The product of these sums of probabilities results in Eq. (4.4).

4.3.1.2 Reordering Extent Metric

This section calculates the probability distribution of the reordering extent. The extent equals the number of burst arrivals between the *located burst* and the test burst. Formally, the extent e_i for a reordered burst at arrival position i at the destination follows from Eq. (3.4).

$$e_i = i - \min_{j<i} \{j : s[j] > s[i]\} . \tag{4.5}$$

Therein, the nominal in-order position is characterized by the smallest j, where the corresponding sequence number $s[j]$ is larger than the sequence number of the burst at position i. The name of this special burst j is *located burst* as it indicates the first burst out of a sequence of bursts, which have a larger sequence number than the test burst.

According to this definition, the sequence number of the located burst is $s = f$ where $0 < f \leq d_t$. The located burst may also follow an abstract link of length d_l, where the delay obeys the following inequality as the located burst arrives always before the test burst: $0 \leq d_l < d_t + f$. The bursts with sequence number $0 < s < f$ and the bursts with sequence number $f + 1 < s \leq f + d_l$, in case of a delayed located burst, arrive earlier than the located burst.

Figure 4.5 depicts this scenario for a test burst delay of $d_t = 8$ and the located burst $f = 3$. The latter receives an additional delay of $d_l = 2$. Note that the figure extends the actual delay to denote the order of the burst arrivals in case those two bursts arrive after the same burst. If $f = 3$ becomes the located burst, it has to satisfy the condition of a located burst according to Chapter 3. In this example, the bursts with sequence number 1 and 2 as well as 4 and 5 need to arrive later then burst 3: 1 and 2 because of their smaller sequence number; 4 and 5 because of the condition of the smallest burst index with a larger sequence number than the test burst. The evaluation of the burst extent requires the following three random events. The figure also shows these random events:

Random event $\mathfrak{F}$ applies to the located burst,

Random event $\mathfrak{E}$ applies to bursts arriving later than the located burst and prior to the test burst and thus define the extent,

Random event $\mathfrak{B}$ applies to bursts, which have to arrive later than the located burst due to the necessary condition of the located burst.

According to these random events, the following list classifies the bursts with respect to their sequence number (cf. Figure 4.5).

$s < 0$	bursts with sequence number smaller than 0 may arrive later than the located burst, and thus, contribute to the extent. Event $\mathfrak{E}$ applies.
$0 < s < f$	for bursts with sequence number smaller than f but larger than zero, both events $\mathfrak{E}$ and $\mathfrak{B}$ apply. These bursts may contribute to the extent but overall they arrive later than the located burst f, due to the condition of the located burst.
$f < s \leq f + d_l$	if the located burst f is delayed, too, the events $\mathfrak{E}$ and $\mathfrak{B}$ apply for the bursts between the located burst and the test burst.
$f + d_l < s \leq d_t$	bursts which originally arrive later than the located burst but prior to the test burst contribute to the extent, event $\mathfrak{E}$ applies.

The next sections evaluate the probabilities of these random events.

4.3.1.2.1 Random event $\mathfrak{E}$

Bursts with sequence number $s \leq d_t$ contribute to the extent if they arrive later than the located burst and prior to the test burst. The probability of a burst with sequence number s arriving later than the located burst and prior to the test burst depends on its location S, the delay of the test burst D_t, the located burst F and its possible delay D_l.

The probability for a burst with sequence number s, which arrives later than the located burst but prior to the test burst, is $\Pr(B = 1|S = s, D_t = d_t, F = f, D_l = d_l)$. Herein, the random variable B represents the burst arrival $B = 1$ prior to the test burst and after the located burst, otherwise $B = 0$. The shorter form $p_{1s}(d_t, f, d_l)$ abbreviates this probability. Eq. (4.6) defines it as sums of probabilities where each sum represents the probability to arrive later than the located burst but prior to the test burst.

$$p_{1s}(d_t, f, d_l) = \begin{cases} \sum_{\kappa=f-s}^{d_t-s} p_\kappa, & \text{if } s < 0 \text{ and } d_l = 0; \\ \sum_{\kappa=f-s}^{d_t-s-1} p_\kappa, & \text{if } 0 < s < f \text{ and } d_l = 0; \\ p_0 + \sum_{\kappa=1}^{d_t-s-1} p_\kappa, & \text{if } f < s \leq d_t \text{ and } d_l = 0; \\ \sum_{\kappa=f+d_l+1-s}^{d_t-s} p_\kappa, & \text{if } s < 0 \text{ and } d_l \neq 0; \\ \sum_{\kappa=f+d_l+1-s}^{d_t-s-1} p_\kappa, & \text{if } 0 < s \leq f + d_l \text{ and } d_l \neq 0; \\ p_0 + \sum_{\kappa=1}^{d_t-s-1} p_\kappa, & \text{if } f + d_l < s \leq d_t \text{ and } d_l \neq 0; \\ 0, & \text{otherwise.} \end{cases} \tag{4.6}$$

For instance, Figure 4.5 on page 72 considers the burst with sequence number $s = -1$. The probability that this burst arrives after the located burst $f = 3$, which follows l_{d_l} with $d_l = 2$ is $p_7 + p_8$. If burst $s = -1$ follows link l_6 it arrives prior to the located burst. If burst $s = -1$ follows link l_9 it arrives later than the test burst.

4.3.1.2.2 Random event $\mathfrak{B}$

Random event $\mathfrak{B}$ applies to bursts with $s > 0$, which originally arrive prior to the located burst. These bursts must not arrive prior to the located burst as a necessary condition of the located burst. With the application of the law of total probability, the probability of event $\mathfrak{B}$ is derived by its complementary $\Pr(\mathfrak{B}) = 1 - \Pr(\bar{\mathfrak{B}})$.

Therein, $\Pr(\bar{\mathfrak{B}})$ denotes the probability of a burst arrival for a specific burst prior to the located burst. This probability depends on the original location S of the burst and the located burst F and its delay D_l. The expression $Q(B = 1|S = s, F = f, D_l = d_l) = q_{1s}(f, d_l)$ denotes this. The random variable B indicates the burst arrival prior to the located burst. Eq. (4.7) shows the probabilities $q_{1s}(f, d_l)$ for all bursts which apply for random event $\mathfrak{B}$.

$$q_{1s}(f, d_l) = \begin{cases} 1 - \sum_{\kappa=0}^{f-s-1} p_\kappa, & \text{if } 1 \leq s < f \text{ and } d_l = 0; \\ 1 - \sum_{\kappa=0}^{f+d_l-s} p_\kappa, & \text{if } 1 \leq s \leq f + d_l \text{ and } d_l \neq 0; \text{ and } s \neq f \\ 1, & \text{otherwise.} \end{cases} \tag{4.7}$$

This again results in sums of probabilities indicating a non-arrival before the test burst.

4.3.1.2.3 Conditional random events $\mathfrak{B}$ and $\mathfrak{E}$

Event $\mathfrak{B}$ is a necessary condition for the bursts with a smaller sequence number than the located burst. These bursts also apply event $\mathfrak{E}$ as depicted in Figure 4.5. This results in a conditional probability for these bursts contributing to the extent. The probability that these bursts contribute to the extent (event $\mathfrak{E}$) is conditioned by event $\mathfrak{B}$. This conditional probability results in:

$$\Pr(\mathfrak{E}|\mathfrak{B}) = \frac{\Pr(\mathfrak{B},\mathfrak{E})}{\Pr(\mathfrak{B})} = \frac{\Pr(\mathfrak{E})}{\Pr(\mathfrak{B})} = \frac{\Pr(\mathfrak{E})}{1-\Pr(\bar{\mathfrak{B}})} \tag{4.8}$$

The joint probability $\Pr(\mathfrak{B},\mathfrak{E})$ is equal to the probability $\Pr(\mathfrak{E})$ as event $\mathfrak{E}$ includes random event $\mathfrak{B}$ as well. With the previous expressions $p_{1s}(d_t, f, d_l)$ (cf. Eq. (4.6)) and $q_{1s}(f, d_l)$ (cf. Eq. (4.7)), the conditional probability leads to: $p^*_{1s}(d_t, f, d_l) = \frac{p_{1s}(d_t,f,d_l)}{q_{1s}(f,d_l)}$.

4.3.1.2.4 Random event $\mathfrak{F}$

Each of the bursts with sequence number s in $0 < s \leq d_t$ may serve as the located burst. The sequence number of the located burst is f. The located burst receives a delay of d_l with probability p_{d_l}. The necessary condition for the located burst is the arrival of bursts with sequence number $0 < s < f$ later than the located burst f. This necessary probability depends on the position/sequence number F and the delay D_l of the located burst. The necessary condition for the located burst is:

$$\Pr(\mathfrak{F} \,|\, F = f, D_l = d_l) = \prod_{s=1}^{d_l+f} Q(B = 1 \,|\, S = s, F = f, D_l = d_l) = \prod_{s=1}^{d_l+f} q_{1s}(f, d_l) \tag{4.9}$$

This joint probability requires a later arrival of bursts, which departed earlier than the located burst. It considers all bursts with sequence numbers between 1 and the sequence number of the located burst plus its delay.

4.3.1.2.5 Reordering extent

With the above probability distributions for the various random events, the reordering extent distribution is easy to derive. $\Pr(E = e \,|\, D_t = d_t, F = f, D_l = d_l)$ denotes the probability of E burst arrivals between the located burst and the test burst: therein, the conditions are the delay of the test burst D_t and a located burst at position F with a delay D_l.

The next step considers the probability of every potential burst to contribute to the extent. The burst arrivals prior to the test burst and after the located burst are independent of each other. The composite of the number of burst arrivals forming the extent is a joint probability experiment. The discrete convolution of the probabilities of all bursts leads to the estimated probability of above. To calculate the convolution, the probability generating function (GF) is applied. $\Pr(B = 1|S = s, D_t = d_t, F = f, D_l = d_l)$ denotes

the probability that burst s contributes to the extent. The probability generating function for this distribution becomes:

$$\begin{aligned} G_{s,d_t,f,d_l}(z) &= \sum_{i=0}^{1} p_{is}(d_t, f, d_l)\, z^i \\ &= \begin{cases} p_{0s}(d_t, f, d_l) + p^*_{1s}(d_t, f, d_l)\, z & \text{if } 0 < s \leq f + d_l \\ p_{0s}(d_t, f, d_l) + p_{1s}(d_t, f, d_l)\, z & \text{otherwise} \end{cases} \end{aligned} \tag{4.10}$$

The product of the GF of all bursts arriving prior to the test burst determines the GF of the distribution of burst arrivals after the located burst and prior the test burst.

$$\begin{aligned} s_{min} &= \begin{cases} f + d_l - m & \text{if } d_l = 0 \\ f + d_l - m + 1 & \text{otherwise} \end{cases} \\ G_{d_t,f,d_l}(z) &= \prod_{s=s_{min}}^{d_t} G_{s,d_t,f,d_l}(z) \end{aligned} \tag{4.11}$$

The probability distribution function becomes $\Pr(E = e \,|\, D_t = d_t, F = f, D_l = d_l)$ by the derivation of the GF of the joint experiment:

$$\Pr(E = e \,|\, D_t = d_t, F = f, D_l = d_l) = \frac{1}{e!} \frac{\partial^e}{\partial z^e} G_{d_t,f,d_l}(z)\bigg|_{z=0} \qquad \forall e > 0 \tag{4.12}$$

The computational effort of the product in Eq. (4.12) and the derivation in Eq. (4.11) is relaxed by two less expensive steps. In Eq. (4.11) the eth derivation gives the eth coefficient of the polynomial $G_{d_t,f,d_l}(z)$. The Cauchy product defines this coefficient in Eq. (4.12). The probability distribution of the reordering extent considers every combination of the test burst delay D_t, the location of the located burst F and its delay D_l. Together they form a triple sum.

$$\begin{aligned} \Pr(E = e) = \sum_{d_t=1}^{m} \sum_{f=1}^{d_t} \sum_{d_l=0}^{(d_t-f-1)^+} & p_{d_t}\, p_{d_l}\, \Pr(\mathfrak{F} \,|\, F = f, D_l = d_l) \\ & \Pr(E = e - 1 \,|\, D_t = d_t, F = f, D_l = d_l) \end{aligned} \tag{4.13}$$

The outer sum represents the possible delay D_t of the test burst. The middle sum represents the position of the located burst F. The inner sum represents the delay D_l of the located burst. The three sums enclose a product of four factors. The first factor denotes the delay probability of the test burst. The second factor denotes the delay probability of the located burst. The third factor represents the conditional probability of the located burst Eq. (4.9). The last factor quantifies the probability of $e - 1$ burst arrivals between the located burst and the test burst Eq. (4.11). The located burst accounts to the overall extent e.

4.3.1.3 N_r-reordering Metric

This section derives the complementary cumulative distribution function (ccdf) of the n_r-reordering metric of Section 3.2.5. The test burst arrives n_r-reordered at the destination

if there are at least n_r subsequent burst arrivals with $s > 0$ prior to the test burst. This definition accounts for two conditions: (a) that the test burst receives an extra delay and (b) that the sequence of n_r burst arrivals with $s > 0$ at the destination excludes any arrival of bursts with sequence number $s < 0$.

The first burst of this sequence is the located burst with sequence number f. The located burst f receives a delay of d_l. The probability of $n_r - 1$ burst arrivals between the located burst and the test burst is $\Pr(B = n_r - 1 | D_t = d_t, F = f, D_l = d_l, S > 0)$. Note that only bursts with $s > 0$ contribute to the extent. The probability of no burst arrivals with sequence number $s < 0$ between the located burst and the test burst is $\Pr(B = 0 | D_t = d_t, F = f, D_l = d_l, S < 0)$.

The probability that a burst with sequence number s arrives later than the located burst but prior to the test burst depends on its location S and the delay of the test burst D_t and the located burst F and its delay D_l. Eq. (4.6) gives the individual probability. With help of Eq. (4.10), Eq. (4.12) and Eq. (4.11), one can calculate both probabilities.

$$\Pr(B = 0 \,|\, D_t = d_t, F = f, D_l = d_l, S < 0) = G_{s<0,d_t,f,d_l}(0) \tag{4.14}$$

$$s_{min} = \begin{cases} f + d_l - m & \text{if } d_l = 0 \\ f + d_l - m + 1 & \text{otherwise} \end{cases} \tag{4.15}$$

$$G_{s<0,d_t,f,d_l}(z) = \prod_{s=s_{min}}^{-1} G_{s,d_t,f,d_l}(z) \tag{4.16}$$

$$\Pr(B = n_r \,|\, D_t = d_t, F = f, D_l = d_l, S > 0) = \frac{1}{n_r!} \frac{\partial^{n_r}}{\partial z^{n_r}} G_{s>0,d_t,f,d_l}(z) \bigg|_{z=0} \tag{4.17}$$

$$G_{s>0,d_t,f,d_l}(z) = \prod_{s=1}^{d_t} G_{s,d_t,f,d_l}(z) \tag{4.18}$$

Eq. (4.14) gives the probability of no burst arrival between the located burst and the test burst for bursts with sequence numbers $s < 0$. Eq. (4.17) gives the probability of exactly $n_r - 1$ burst arrivals with a larger sequence number than the test burst $s > 0$ between itself and the located burst. Putting both results together, leads to the ccdf of the n_r-reordering metric of Eq. (4.19). The structure is similar to Eq. (4.13) except for the dependence on the sequence number of the burst arrivals.

$$\Pr(N_r > n_r) = \sum_{d_t=1}^{m} \sum_{f=1}^{d_t} \sum_{d_l=0}^{(d_t-f-1)^+} p_{d_t}\, p_{d_l} \Pr(B = n_r | D_t = d_t, F = f, D_l = d_l, S > 0) \Pr(B = 0 | D_t = d_t, F = f, D_l = d_l, S < 0) \tag{4.19}$$

The outer sum considers all possible delays of the test burst. The middle sum considers the possible positions of the located burst, while the inner sum refers to the potential delays of the located burst. The first two factors denote the probabilities to receive a certain delay, while the third factor assume a certain number of bursts with $s > 0$ before the test burst. The last factor prohibits any arrival of bursts with $s < 0$ in the considered interval of the test burst delay.

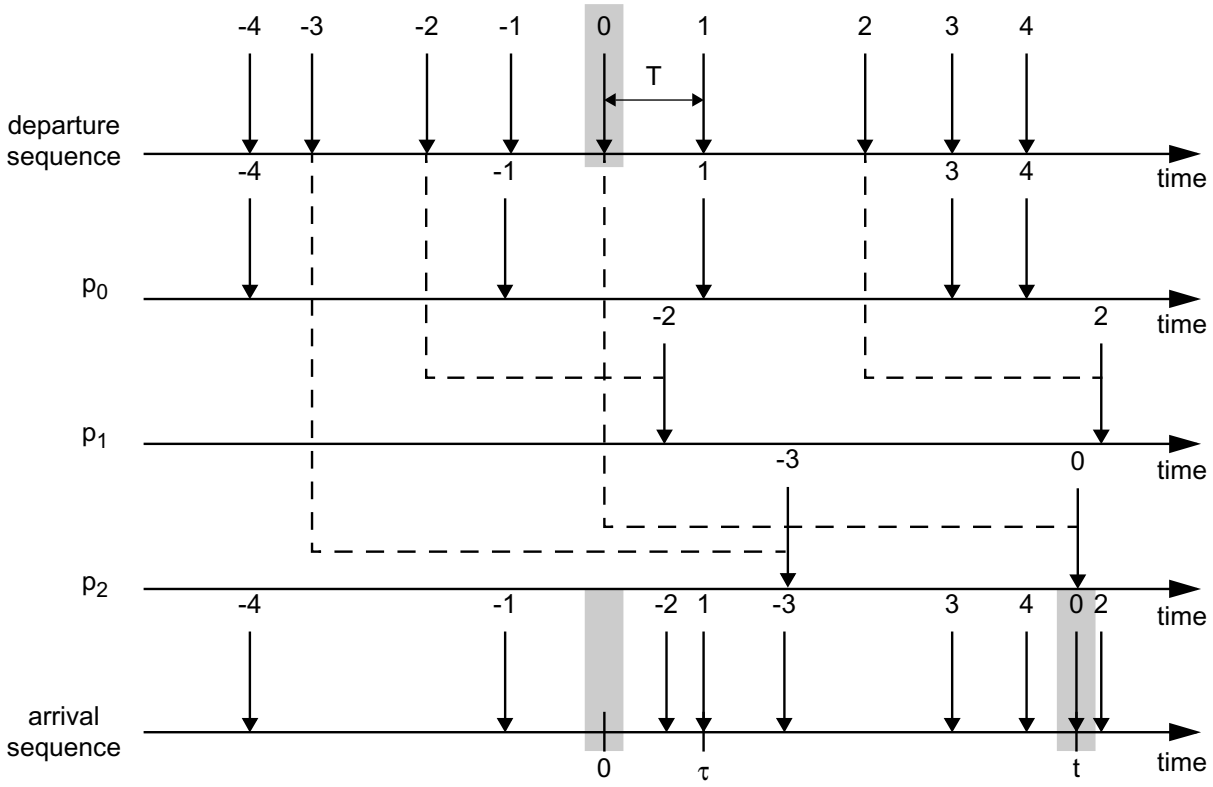


Figure 4.6: Poisson traffic, splitting process

4.3.2 Probabilistic Traffic Models

This section applies the reordering model on probabilistic traffic models. One probabilistic traffic model is the Poisson model with negative exponentially distributed inter-arrival times. For this traffic model, this section computes the IETF reordering metrics for deterministic traffic as the previous section. As this traffic model is rather simple and does not represent generic traffic conditions, the second section argues on an extension towards other probabilistic traffic models.

4.3.2.1 Poisson Traffic Model

This section calculates the reordering metrics of the previous section for Poisson arrivals. Poisson arrivals show a negative exponentially distributed inter-departure time. Figure 4.6 depicts the traffic at the splitting point of Figure 4.4 and the resulting traffic on the various abstract links. The scenario equals a point process of the burst arrivals at the splitting point. In the figure, the first time axis shows the origin burst departure process with the test burst $s = 0$ (highlighted). The random variable of the inter-departure time is T. The probability distribution function of this inter-departure time is $F(t) = 1 - \exp(-\lambda t)$, with mean rate $\lambda = 1/\mathsf{E}[T]$. The second time axis shows the burst arrival process of bursts following abstract link l_0, i. e., they do not receive any additional delay. The remaining two time axis depict the point process of bursts following abstract link l_1 or l_2, respectively. Bursts on these abstract links receive an additional delay and arrive later than originally scheduled (dashed lines). The superposition of abstract links $l_0 \ldots l_2$ leads

to the experienced arrival order at the destination on the bottom time axis.

Joining an abstract link l_j with probability p_j forms a random splitting process. As a result, the burst inter-arrival time T_j on link l_j is also negative exponentially distributed [109]. Consequently, the probability density function of the inter-departure time for link l_j becomes $f_j(t) = \lambda_j \exp(-\lambda_j t)$, where $\lambda_j = p_j \lambda$.

The calculation of the metric requires two special points in time on the time axis (bottom row of Figure 4.6). The first one, the origin, corresponds to the arrival of the test burst $s = 0$ at the destination. The second one is the arbitrary point in time t, $t > 0$. The sequence numbers of bursts departing before the test burst are $s < 0$, while the sequence number of bursts departing after the test burst are $s > 0$.

The interval $[0, t]$ is divided on the arrival axis as well as on the abstract links l_j into two intervals $[0, \tau)$ and $[\tau, t]$. The focus is on the probability distribution function of the random number X_j of burst arrivals on link l_j in the time interval $[\tau, t]$. Bursts originally departing in the interval $[\tau - j\,\Delta, t - j\,\Delta]$ arrive in the interval $[\tau, t]$ if they follow link l_j. The probability distribution of the number of burst arrivals of link l_j in $[\tau, t]$ follows a Poisson distribution:

$$p_j^{[\tau,t]}(x_j) = \frac{(\lambda_j(t-\tau))^{x_j}}{x_j!} e^{-\lambda_j(t-\tau)} \tag{4.20}$$

At the destination, the bursts on the individual abstract links are superposed together. The set $\mathbb{J}$ represents all abstract links superposing the bursts at the destination. The superposition of burst arrivals X_j within the interval $[\tau, t]$ forms a random experiment. The random variable of the number of burst arrivals X is a sum of random variables, i. e., $X = \sum_{j \in \mathbb{J}} X_j$. The probability distribution of X is the result of the discrete convolution of the probability distributions of all X_j. The probability generating functions help to apply the convolution on all X_j. The GF of X_j per abstract link and for the resulting random variable X becomes:

$$G_j^{[\tau,t]}(z) = \sum_{i=0}^{\infty} p_j^{[\tau,t]}(i)\, z^i \tag{4.21}$$

The GF of X becomes the product of the GF per abstract link:

$$G^{[\tau,t]}(z) = \prod_{j \in \mathbb{J}} G_j^{[\tau,t]}(z) \tag{4.22}$$

The successive derivation of the GF $G^{[\tau,t]}(z)$ identifies the probability distribution $p_x^{[\tau,t]}$ of the random variable X of the superposition:

$$p_x^{[\tau,t]} = \frac{1}{x!} \frac{\partial^x}{\partial z^x} G^{[\tau,t]}(z) \bigg|_{z=0}. \tag{4.23}$$

The derivation of the reordering metrics, especially, the n_r-reordering metric requires the distinction of the bursts according to their sequence number. In the observed interval $[\tau, t]$, bursts may arrive with sequence number $s > 0$ and $s < 0$ dependent on the abstract link l_j and the size of the interval $[\tau - j\,\Delta, t - j\,\Delta]$. There are three cases to distinguish:

1. If $(0 < \tau - j\,\Delta) \wedge (0 < t - j\,\Delta)$ bursts with sequence number $s < 0$ do not arrive in the interval $[\tau, t]$.
2. If $(0 > \tau - j\,\Delta) \wedge (0 < t - j\,\Delta)$ bursts with sequence number $s < 0$ as well as with $s > 0$ arrive in the interval $[\tau, t]$.
3. If $(0 > \tau - j\,\Delta) \wedge (0 > t - j\,\Delta)$ bursts with sequence number $s > 0$ do not arrive in the interval $[\tau, t]$.

The total number of burst arrivals in $[\tau, t]$ can be classified by the sequence number. For bursts with $s < 0$, there are L_j bursts, for bursts with $s > 0$, there are K_j bursts in $[\tau, t]$. The probability distributions of L_j and K_j for the previous three cases are with respect to Eq. (4.20):

$$p_{j,s<0}^{[\tau,t]}(L_j) = \begin{cases} p_j^{[\tau,t]}(L_j), & \text{if } (\tau - j\,\Delta < 0) \wedge (t - j\Delta < 0) \\ p_j^{[\tau - j\,\Delta, 0]}(L_j), & \text{if } (\tau - j\,\Delta < 0) \wedge (t - j\Delta > 0) \\ 0, & \text{otherwise} \end{cases} \tag{4.24}$$

$$p_{j,s>0}^{[\tau,t]}(K_j) = \begin{cases} p_j^{[\tau,t]}(K_j), & \text{if } (\tau - j\,\Delta > 0 \wedge t - j\Delta > 0) \\ p_j^{[0, t - j\,\Delta]}(K_j), & \text{if } (\tau - j\,\Delta < 0 \wedge t - j\Delta > 0) \\ 0, & \text{otherwise} \end{cases} \tag{4.25}$$

The probability distributions of L_j and K_j take into account the abstract link number and the position of τ and t. Consequently, the considered intervals of potential burst arrivals change (right side of the equation) when they apply Eq. (4.20). Applying the method of the probability generating function for the superposing event of multiple branches of set $\mathbb{J}$ for $s > 0$ and $s < 0$, the probability distributions of the number of burst arrivals within $[\tau, t]$ for bursts with $s < 0$ and $s > 0$ become:

$$p_{s<0}^{[\tau,t]}(L) = \frac{1}{L!}\frac{\partial^L}{\partial z^L} G_{s<0}^{[\tau,t]}(z)\bigg|_{z=0} = \frac{1}{L!}\frac{\partial^L}{\partial z^L}\left\{\prod_{j\in\mathbb{J}}\left(\sum_{i=0}^{\infty} p_{j,s<0}^{[\tau,t]}(i)\,z^i\right)\right\}\bigg|_{z=0} \tag{4.26}$$

$$p_{s>0}^{[\tau,t]}(K) = \frac{1}{K!}\frac{\partial^K}{\partial z^K} G_{s>0}^{[\tau,t]}(z)\bigg|_{z=0} = \frac{1}{K!}\frac{\partial^K}{\partial z^K}\left\{\prod_{j\in\mathbb{J}}\left(\sum_{i=0}^{\infty} p_{j,s>0}^{[\tau,t]}(i)\,z^i\right)\right\}\bigg|_{z=0} \tag{4.27}$$

The next sections apply these results to determine the values of the introduced reordering metrics.

4.3.2.1.1 Reordering probability

The probability of an out-of-sequence arrival is the probability of the compound event to follow link l_{d_t} (with probability p_{d_t}) and the probability of at least one burst arrival in the interval $[0, d_t\,\Delta]$. The complementary probability of no burst arrival applies Eq. (4.27) and calculates the latter probability. This leads to the reordering probability in case of Poisson traffic $\mathfrak{P}$:

$$P(\mathfrak{P}) = \sum_{d_t=1}^{m} p_{d_t}\,\left(1 - p_{s>0}^{[0,d_t\,\Delta]}(0)\right). \tag{4.28}$$

In Eq. (4.27), the set $\mathbb{J}$ of considered links l_j is $\mathbb{J} = \{o : 0 \leq o < d_t\}$. Bursts with $s > 0$ on links with a larger delay than $d_t\,\Delta$ do not arrive within $[0, t]$. In Eq. (4.28), the probability in the bracket corresponds to the 0th derivative of Eq. (4.27). The 0th derivative corresponds to the function itself, while $z = 0$ results in the first term of the sum. Taken both properties together simplifies Eq. (4.28).

$$\begin{aligned} P(\mathfrak{P}) &= \sum_{d_t=1}^{m} p_{d_t} \left(1 - \prod_{j=0}^{d_t-1} p_j^{[0,(d_t-j)\,\Delta]}(0)\right) = \sum_{d_t=1}^{m} p_{d_t} \left(1 - \prod_{j=0}^{d_t-1} e^{-\lambda_j \Delta (d_t-j)}\right) \\ &= \sum_{d_t=1}^{m} p_{d_t} \left(1 - \exp\left(-\lambda\Delta \sum_{j=0}^{d_t-1} p_j (d_t - j)\right)\right) \end{aligned} \tag{4.29}$$

4.3.2.1.2 Reordering extent metric

This section calculates the probability distribution of the reordering extent metric. According to the extent metric definition, the calculation of the reordering extent metric requires the following two steps: (a) identify the located burst with the smallest j as defined in Eq. (4.5), (b) count the number of burst arrivals between the located burst and the test burst.

The delay of the test burst is a necessary condition of (a). Without loss of generality, the test burst may follow link l_{d_t} receiving a delay of $d_t\,\Delta$. The located burst follows abstract link l_{d_l}, $0 \leq d_l < d_t$, and arrives at $\tau = d_l\,\Delta + t$. t is the time between the departure of the located burst and the departure of the test burst. Consequently, the co-domain of τ is $t + d_l\,\Delta \leq \tau \leq d_t\,\Delta$. The probability of a burst arrival at τ is a compound probability of a burst departing at t and the probability to follow link l_{d_l}: $\Pr(B = 1 | D_l = d_l, t < T \leq t + dt)$, with B indicating a burst arrival. The second part of this joint probability results in the instantaneous termination rate $\lambda = 1/\,\mathsf{E}[T]$.

$$\begin{aligned} \Pr(B = 1 | D_l = d_l, t < T \leq t + dt) &= p_{d_l} \frac{\Pr(t < T \leq t + dt)}{dt} dt \\ \lim_{dt \to 0} \frac{\Pr(t < T \leq t + dt)}{dt} &= \lim_{dt \to 0} \frac{1 - e^{-\lambda\, dt}}{dt} = \lambda \end{aligned} \tag{4.30}$$

The necessary condition of (a) restricts any burst arrival in $[0, \tau]$ with reference numbers smaller than the located burst, i. e., bursts departed in $(0, t)$. These bursts must not follow abstract link l_j with $j < t/\Delta + d_l$ as they would arrive earlier to the located burst. The probability of this restriction again is the probability of the compound event applying Eq. (4.27) with no burst arrival on the abstract links l_j with $j \in \{o : 0 \leq o \leq \hat{j} = \lfloor t/\Delta + d_l \rfloor\}$.

$$\Pr(B = 0 | 0 < T < \tau, S > 0) = p_{s>0}^{[0,\tau]}(0) = \prod_{j=0}^{\hat{j}} p_{j,s>0}^{[0,\tau - j\,\Delta]}(0) \tag{4.31}$$

An arbitrary number of burst arrivals form the extent in the interval $[\tau, d_t\,\Delta]$. The probability of this compound event bases on the probability distribution $p_j^{[\tau,t]}(x_j)$ (cf. Eq. (4.20))

of each abstract link j. As the located burst completes the extent value to e, Eq. (4.22) is applied for $e-1$ arrivals in the considered interval, with $\mathbb{J} = \{o : 0 \leq o \leq m\}$.

$$\Pr(B = e-1 | \tau < T \leq d_t\,\Delta) = p^{[\tau,t]}(e-1) = \frac{1}{(e-1)!}\,\frac{\partial^{e-1}}{\partial z^{e-1}} G^{[\tau,d_t\,\Delta]}(z)\bigg|_{z=0} \tag{4.32}$$

The compound of these necessary three conditions leads to the probability distribution of the extent. The following calculations consider each possible delay of the test burst and of the located burst. The located burst is considered at every possible point in time t. With help of Eq. (4.30), Eq. (4.31) and Eq. (4.32), the probability distribution of the burst extent metric becomes:

$$\begin{aligned}\Pr(E = e) = \sum_{d_t=1}^{m} p_{d_t} \sum_{d_l=0}^{d_t-1} \int_0^{(d_t-d_l)\,\Delta} & \Pr(B = 0 | 0 < T < \tau, S > 0) \\ & \times \Pr(B = 1 | t < T \leq t + dt)\ \Pr(B = e-1 | \tau < T < d_t\,\Delta)\end{aligned} \tag{4.33}$$

Within the integral, the first factor denotes the probability of the condition of the located burst with no burst arrival before it. The second factor gives the probability of the located burst arrival exactly at time t, while the last factor determines the probability for the required burst arrivals in the interval between located burst and test burst. With help of Eq. (4.32) and Eq. (4.31), the extent probability simplifies to:

$$\Pr(E = e) = \sum_{d_t=1}^{m} p_{d_t} \sum_{d_l=0}^{d_t-1} p_{d_l}\,\lambda \int_0^{(d_t-d_l)\,\Delta} p_{s>0}^{[0,\tau]}(0)\ p^{[\tau,t]}(e-1)\ dt \tag{4.34}$$

4.3.2.1.3 N_r-reordering metric

In contrast to the extent metric, the conditions for the n_r-reordering metric are different. After the located burst, the n_r-reordering metric requires consecutive arrivals of bursts, which have larger sequence numbers than the test burst. Again, the located burst may be at τ, while the test burst receives a delay of $d_t\,\Delta$.

The ccdf of the n_r-reordering metric depends on the joint probability of two events: (a) arrival of the located burst at τ, (b) arrival of $n_r - 1$ bursts with appropriate sequence number in the interval $[\tau,\, d_t\,\Delta]$. Condition (b) includes no burst arrival in $[\tau,\, d_t\,\Delta]$ with sequence number $s < 0$. From the previous section the probability of (a) is $\Pr(B = 1 | t < T \leq t+dt)$. With the help of the GF of the probability of the compound event, it is possible to derive the probability of the number of arrivals. This requires additionally Eq. (4.27) for $(s > 0)$ and Eq. (4.26) for $(s < 0)$:

$$\Pr(B = n_r | \tau < T \leq d_t\,\Delta, S > 0) = p_{s>0}^{[\tau,d_t\,\Delta]}(n_r) \tag{4.35}$$

$$\Pr(B = 0 | \tau < T \leq d_t\,\Delta, S < 0) = p_{s<0}^{[\tau,d_t\,\Delta]}(0) \tag{4.36}$$

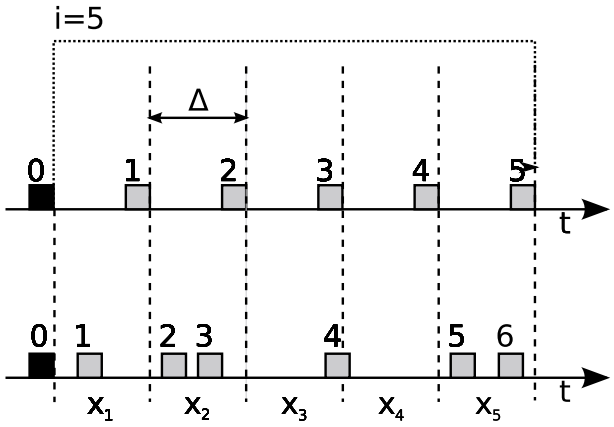


Figure 4.7: Deterministic traffic (top) and arbitrary traffic (bottom)

Similar to Eq. (4.33), the overall ccdf of the n_r-reordering metric becomes:

$$\Pr(N_r > n_r) = \sum_{d_t=1}^{m} p_{d_t} \sum_{d_l=0}^{d_t-1} \int_0^{d_t\,\Delta} \Pr(B = 0 \,|\, \tau < T \leq d_t\,\Delta, S < 0) \times \Pr(B = 1 \,|\, t < T \leq t + dt)\, \Pr(B = n_r \,|\, \tau < T \leq d_t\,\Delta, S > 0). \quad (4.37)$$

Applying Eq. (4.32) and Eq. (4.31) leads to the more compact version:

$$\Pr(N_r > n_r) = \sum_{d_t=1}^{m} p_{d_t} \sum_{d_l=0}^{d_t-1} \lambda p_{d_l} \int_0^{d_t\,\Delta} p_{s>0}^{[\tau, d_t\,\Delta]}(n_r)\;\; p_{s<0}^{[\tau, d_t\,\Delta]}(0) dt \quad (4.38)$$

The double sum considers the possible abstract links of the test burst as well as the located burst. The integral considers any possible arrival instance of the located burst. The first factors within the integral assures at least n_r burst arrivals with sequence numbers $s > 0$ before the test burst, while the second factor restricts any arrival of bursts with $s < 0$ within this sequence.

4.3.2.2 Extension Towards Generic Traffic Models

The last section applied the reordering model and corresponding methodology on one probabilistic traffic model. This section elaborates on the extensions of the reordering model and its methodology towards generic traffic models. These elaborations assume that the burst arrival process follows a renewal process, which shows iid (independent and identical distributed) properties. The model of the disordering network, i. e., discretised delays on each abstract link remains the same. The first section elaborates on the general difficulties on a potential extension, while the second section provides the findings for the reordering ratio for iid traffic models.

4.3.2.2.1 Metrics for iid traffic models

The reordering metrics of Section 3.2.5 rely on the number of arrivals between a test burst and the located burst or on the presence of any arrival before the test bust. As the disordering network shows discrete delays, the reordering metrics depend on the number of arrivals within certain intervals defined by the discrete delays (cf. Figure 4.7). These metrics require the knowledge on the number of arrivals within a certain interval. As the test burst defines the origin of the time axis, one can derive the probability distribution $p_{1,x}(\Delta)$ of arrivals x in the first interval according to the following equation:

$$p_{1,x_1}(\Delta) = \Pr\left(\sum_{k=1}^{x_1} T_k \leq \Delta, \sum_{k=1}^{x_1+1} T_k > \Delta\right) \tag{4.39}$$

$$= \int_0^{\Delta} \Pr\left(T_1 + T_2 + \cdots T_{x_1} \leq \Delta, T_{x_1+1} > \Delta - \tau_1\right) d\tau_1 \tag{4.40}$$

Literally, the sum of x_1 inter-arrival times is equal or smaller than the size of the interval Δ. The $x_1 + 1$st arrival exceeds the interval Δ. The solution of this equation leads to the probability distribution on the number of arrivals in the first interval. For deterministic traffic or if the arrival distribution follows a negative exponential distribution, the equation is easy to derive. For subsequent intervals, the number of arrivals depends on the position of the last arrival of the first interval. If τ_1 shows the location of the last arrival in the first interval, the number of arrivals in the second interval follows the next equation:

$$p_{2,x_2}(\Delta, \tau_1) = \Pr\left(\tau_1 - \Delta + \sum_{k=1}^{x_2} T_k \leq \Delta, \tau_1 - \Delta + \sum_{k=1}^{x_2+1} T_k > \Delta\right) \tag{4.41}$$

The probability distribution includes the memory of the last arrival within the first interval. These considerations also hold for subsequent intervals, which results in a inter-dependence of the number of arrivals of subsequent intervals $p_{j,x_j}(\Delta, \tau_1, \tau_2, \ldots, \tau_{j-1})$. For this correlation, it is in general hard to obtain a closed form solution. The proposed reordering model is not able to derive the reordering metrics extent and n_r-reordering explicitly for arbitrary traffic models. For these metrics, the model provides network engineers with an upper bound for the expected reordering in a network. Section 5.2 provides the background and the formal analysis for this upper bound scenario. For arbitrary but iid arrivals, the next section calculates the reordering ratio.

The arguments of above hold for iid traffic models. For inter-dependent arrivals, analytic expressions for the arrival distributions become even harder. Then, one has to consider conditional arrivals, too, which is out of scope of this thesis.

4.3.2.2.2 Reordering ratio for iid traffic models

This section determines the reordering ratio for a generic traffic model $\mathfrak{G}$ facing the same disordering network as introduced in Chapter 4. Figure 4.7 on page 84 depicts the scenario of iid traffic arrivals and deterministic traffic.

For deterministic traffic (top), at each interval Δ resides one burst, i. e., spacing of the abstract links. In contrast to this, the number of occurrences within one interval for iid traffic is randomly distributed. X_j is the random variable of the number of arrivals in interval j. The random variable $\vec{X}^{(i)} = (X_1, \ldots, X_i)^T$ is the vector of all random variable X_j. Therein, i is the length of the vector, i. e., the number of intervals. The length of the vector correlates with the considered branch i the test burst follows, i. e., 5 in Figure 4.7. The following equations summarize the properties of the iid arrival traffic model for the first interval with respect to the traffic mean value and the basic delay unit Δ, Therein, $p_n(t)$ denotes the probability distribution of the number of burst arrivals n within a time interval of t:

$$\sum_{n=0}^{\infty} n\, p_n(\Delta) = 1 \tag{4.42}$$

$$\sum_{n=0}^{\infty} p_n(\Delta) = 1 \tag{4.43}$$

The traffic mean value still correlates with the basic delay unit Δ. Eq. (4.42) summarizes this requirement. Eq. (4.43) gives the total probability of all possible arrivals within one interval. Extending these properties on i intervals, leads to Eq. (4.44) and Eq. (4.45).

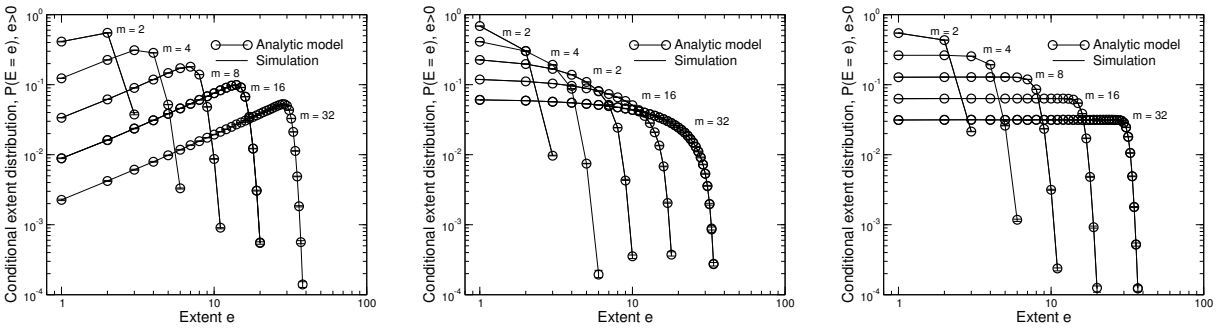
$$\sum_{n=0}^{\infty} \sum_{\vec{x}^{(i)} \in S_n^i} \Pr\left(\vec{X}^{(i)} = \vec{x}^{(i)} | N = n\right) = 1 \tag{4.44}$$

$$\sum_{n=0}^{\infty} \sum_{\vec{x}^{(i)} \in S_n^i} n \Pr\left(\vec{X}^{(i)} = \vec{x}^{(i)} | N = n\right) = i \tag{4.45}$$

Eq. (4.44) represents the total probability of all possible arrival distributions within the next i slots. Therein, S^i represents the co-domain of $\vec{X}^{(i)}$ holding all possible vectors $\vec{x}^{(i)}$ of length i. The subset S_n^i shows exactly n arrivals within the i slots. Eq. (4.45) fixes the condition of the traffic mean value, i. e., the traffic mean value corresponds to the basic delay unit Δ on the interval of all i slots. According to the definition of Section 4.3.1 this is i for i slots. In parallel to the reordering ratio of Section 4.3.1.1, the reordering probability for generic iid traffic $\mathfrak{G}$ becomes:

$$\begin{aligned}
\Pr(\mathfrak{G}) &= \sum_{i=1}^{m} p_i \left(1 - \sum_{n=0}^{\infty} \Pr(N_i = n) \sum_{\vec{x}^{(i)} \in S_n^i} \Pr(\vec{X}^{(i)} = \vec{x}^{(i)} | N_i = n) \prod_{k=1}^{i} \left(\underbrace{\sum_{j=i-k+1}^{m} p_j}_{q_k} \right)^{x_k} \right) \\
&= \sum_{i=1}^{m} p_i \left(1 - \sum_{n=0}^{\infty} \sum_{\vec{x}^{(i)} \in S_n^i} \Pr(\vec{X}^{(i)} = \vec{x}^{(i)}, N_i = n) \prod_{k=1}^{i} {q_k}^{x_k} \right)
\end{aligned} \tag{4.46}$$

The outer sum considers all possible branches i of the test burst and weights these probabilities with p_i. The bracket shows the complementary probability for at least one burst arrival with a larger sequence number. $\Pr(N_i = n)$ gives the probability of n arrivals



(a) Linear prob. dist. (b) Inverse linear prob. dist. (c) Uniform prob. dist.

Figure 4.8: Burst reordering extent for deterministic traffic, $p_0 = 0.9$

within the i intervals. $\Pr(\vec{X}^{(i)} = \vec{x}^{(i)} | N_i = n)$ gives the distribution of $\vec{x}^{(i)} = (x_1, \ldots, x_i)^T$ of these n arrivals among i slots. The last product gives the joint probability of all bursts x_k in interval k to arrive later than the test burst (equivalent to Eq. (4.4)). For convenience of reading, q_k abbreviates these individual factors.

4.3.3 Validation and Illustrative Results

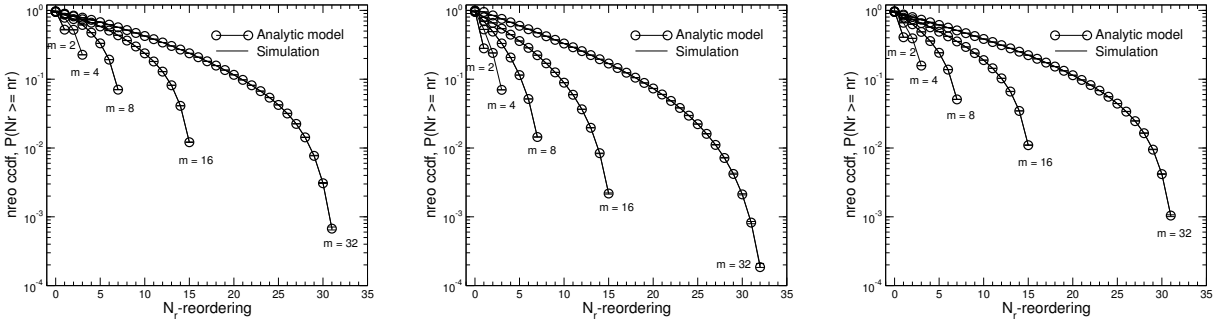
This section provides both, an illustration of the reordering metrics and a verification of the correctness of the obtained results for both traffic models, i. e., deterministic and Poisson traffic. The verification process applies the simulation model of Section 4.2.2.

In the reordering model, the distribution of the probabilities per abstract link is variable and subject to a random choice. For ease of comparison, three different distributions serve for a comprehensive and illustrative comparison. They all have in common the number of abstract links m and the branch probability $1 - p_0$, where p_0 is the probability to follow the abstract link l_0 with no additional delay. The three distributions of the probabilities consider linear, inverse linear and uniform delays. The following expressions show the resulting abstract link probability distributions of p_k for $m \geq k > 0$.

linear	inverse linear	uniform
$p_k = k\dfrac{2(1-p_0)}{m(m+1)}$	$p_{1+m-k} = k\dfrac{2(1-p_0)}{m(m+1)}$	$p_k = \dfrac{1-p_0}{m}$

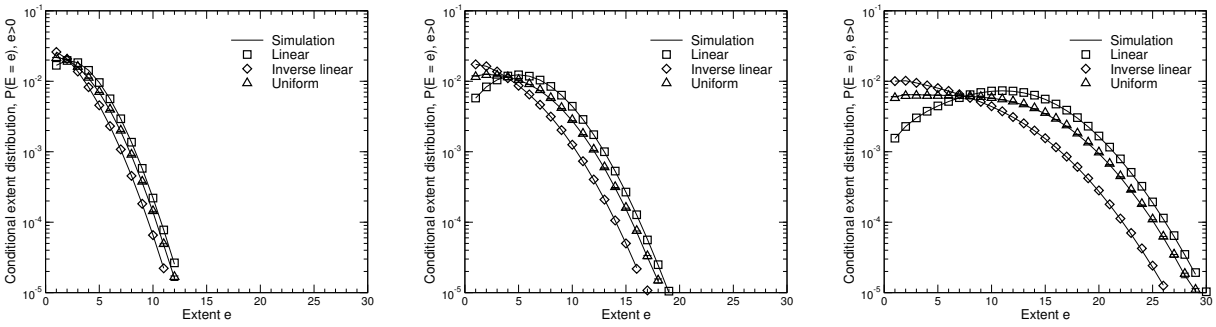
4.3.3.1 Deterministic Traffic

The following few graphs illustrate the two different reordering metrics in the introduced different delay environments. The graphs in Figure 4.8 visualize the probability distribution of the reordering extent for three delay distributions and a branch probability of 0.1 for the scenario with deterministic traffic. The branch probability is the sum of probability of any abstract link except l_0. The set parameter is the number of abstract links m, which is 2, 4, 8, 16 and 32. These graphs backup the analytic findings. The simulation model of Figure 4.3 supplied the values of the reordering metrics and a 95% confidence interval. In all graphs, the analytic findings reside within this confidence interval.



(a) Linear prob. dist. (b) Inverse linear prob. dist. (c) Uniform prob. dist.

Figure 4.9: Burst n_r-reordering metric for deterministic traffic, $p_0 = 0.9$

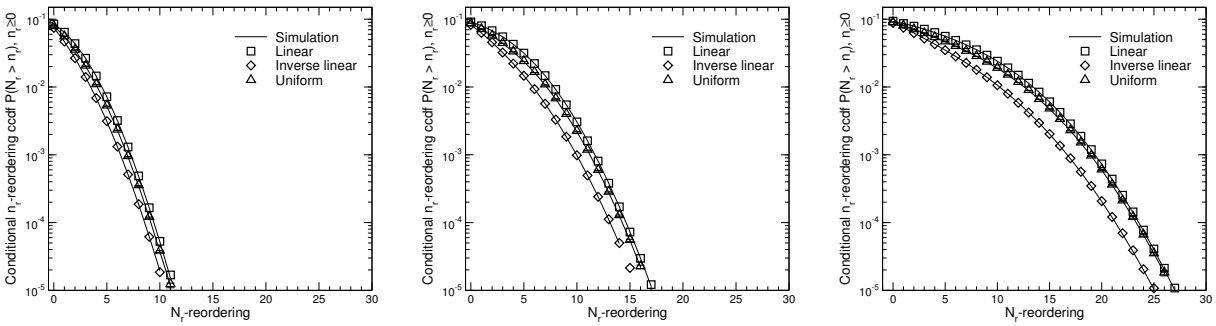


(a) $m = 4$ (b) $m = 8$ (c) $m = 16$

Figure 4.10: Reordering extent for Poisson traffic with $p_0 = 0.9$

The graphs show in logarithmic scale the extent value and the probability distribution of the extent value. The shape of the graphs reflects the distribution of delay lines. An increasing probability (linear) leads to an increasing extent value up to a certain maximum. After the maximum, the graphs decline rapidly towards zero that the theoretical maximum of the extent value is far below 10^{-4}. Similar considerations apply to the inverse linear and uniform distribution. In the inverse linear case, the probability declines steadily, while for the uniform case, the probability remains nearly constant. Both cases also decline very fast for extent values approaching the theoretic maximum.

The graphs in Figure 4.9 show the ccdf of the n_r-reordering metric for the same parameter set as above, i. e., $p_0 = 0.9$. Depending on the probability distribution of the abstract links, the graph decreases more or less rapidly. In here, the ccdf of all three distributions show a quite similar shape. The extension towards larger values differs as this depends on the number of abstract links m. The tail of the graph indicates the maximum value of the ccdf, which is limited in this scenario by the amount of delay lines m. Summarizing, the results obtained by simulation fit the analytic results well. The analytic results are always in the range of the 95% confidence intervals of the simulation. This proves the correctness of the analytic model.



(a) $m = 4$ (b) $m = 8$ (c) $m = 16$

Figure 4.11: N_r-reordering metric for Poisson traffic with $p_0 = 0.9$

4.3.3.2 Poisson Arrivals

This section shows the results for the same parameter set as before but applies Poisson arrivals with a negative exponentially distributed inter-arrival time. The scenario and the parameters are identical, namely the three probability distributions as well as the different number of delay lines that range from 4 to 16.

The graphs of Figure 4.10 depict the simulation and analytic results of the probability distribution for the extent metric. Simulation and analytic results perfectly fit. The maximum extent metric is proportional to the number of delay lines but the intensity is decreasing. With increasing number of delay lines, the probability distribution of the linear delay distribution changes also its shape. The peak extent values drift to the right, while both edges decline. With Poisson traffic, there is no strict theoretic maximum as the number of arrivals is a random process. These graphs illustrate this as they decay rapidly at a certain border value, which is larger than with deterministic traffic.

The graphs in Figure 4.11 depict the simulation and analytic results for the n_r-reordering metric. Again, the analytic findings fit the simulation ones. The maximum n_r-reordering metric is proportional to the delay lines. Because of the probabilistic characterization of this traffic, there is no strict absolute value but a rapidly decaying graph for larger values of n_r. The different delay distribution types do not differ much in shape.

4.4 Hierarchical Packet Reordering

This section considers hierarchical packet reordering, which may occur in networks, which implement packet assembly mechanisms. If these networks change the burst sequence, the assembled packets may also be reordered (cf. Figure 4.12). This section establishes the connection between observed burst reordering and packet reordering and calculates the reordering metrics of these assembled packets. This calculation requires the following assumptions:

- The assembly unit assembles packets in-order, i. e., packets in a burst are in-order by definition.

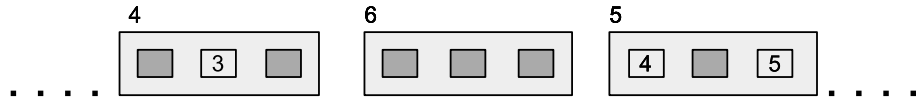


Figure 4.12: Packet and burst reordering

- The packet per burst probability density is $f(p)$. The number of packets per burst depends on the assembly scheme and the packet arrival characteristic.
- A burst arrives at the destination out-of-sequence with probability p_B.
- The probability distribution of the burst extent is $\Pr(E_B = e_B)$.
- The burst n_r-reordering metric is $\Pr(N_B = n_B)$.

For packets in a packet assembly network, the following three sections obtain the expressions for the reordering probability, the distributions for the reordering extent and the n_r-reordering metric. Thereby, an arbitrary packet per burst distribution is considered.

4.4.1 Packet Reordering Probability

The focus of this section is the packet reordering probability p_P. The packet reordering probability equals the ratio of reordered packets to the total number of received packets at the destination. The following theorem postulates the relationship between packet p_P and burst reordering probability p_B.

Theorem 4.1. *The reordering ratio of packets is always less than or equal to the burst reordering ratio:* $p_P \leq p_B$.

Proof of Theorem 4.1. The proof considers bursts, which arrive reordered at the destination. If out-of-sequence bursts carry packets, these packets also arrive reordered if at least one of the bursts also carries packets. Figure 4.12 illustrates this opposite case of in-sequence packets residing in out-of-sequence bursts. Although burst 5 is out-of-sequence, the assembled packets 4 and 5 arrive in-order as burst 6 does not carry any packet of the considered flow. With this observation the proof continues formally. The burst reordering ratio considers the amount of reordered bursts $\tilde{N}_B$ in relation to the total number of bursts N_B at the destination (cf. Eq. (4.47)).

$$p_B = \frac{\tilde{N}_B}{N_B} = \frac{\tilde{N}_B \sum_{i=1}^{\infty} f(i)\, i}{N_B \sum_{i=1}^{\infty} f(i)\, i} \geq \frac{\tilde{N}_P}{N_P} = p_P \tag{4.47}$$

The extension of the ratio by the average number of packets per burst $\sum_{i=1}^{\infty} f(i)\, i$ leads to the total number of arriving packets N_P in the denominator. The resulting number in the nominator is less or equal to the number of reordered packets $\tilde{N}_P$ at the destination. This is true as not necessarily any packet of a reordered burst arrives reordered (cf. Figure 4.12). Consequently, the packet reordering ratio is always less than the burst reordering ratio. □

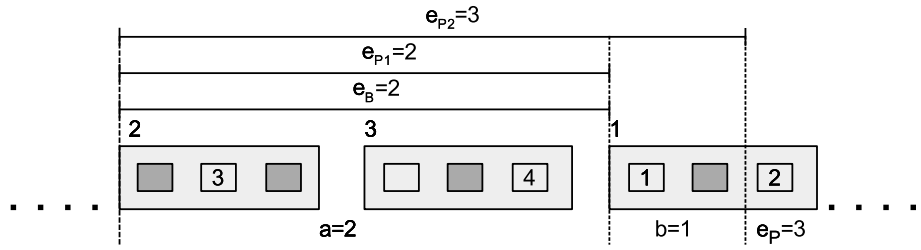


Figure 4.13: Packet extent metric

Additionally, if the packet per burst distribution $f(p)$ includes at least one packet per burst, i. e., $f(0) = 0$, then the packet reordering probability equals the burst reordering probability.

4.4.2 Packet Reordering Extent Metric

This section derives the packet reordering extent metric $\Pr(E_P = e_P)$ depending on a given burst extent metric $\Pr(E_B = e_B)$ and packet per burst distribution $f(p)$. A packet receives an extent e_P if the following necessary conditions hold:

- The packet belongs to an out-of-sequence burst.
- This out-of-sequence burst receives an extent of $e_B > 0$.
- There are exactly e_P packets preceding the considered packet.

The first two conditions depend on each other. Each burst, which arrives out-of-sequence receives a certain extent. Consequently, if a burst receives an extent, it arrives always out-of-sequence. The third condition is twofold. The e_P preceding packets represent a sum of two terms: $e_P = a + b$. a packets reside in the e_B preceding bursts, while b packets precede the considered packet in the same burst. Figure 4.13 illustrates this split. Burst 1 is reordered and receives an extent of 2. Within this burst, two packets reside. The packet with sequence number 2 receives an extent of 3 because of 2 packets in the preceding bursts of burst 1 and because of one packet (packet 1) in the same burst. Consequently, the probability of e_P preceding packets is a joint probability: $\Pr(H_P = a | H_B = e_B)$ denotes the probability of exactly a packets in e_B burst. It reflects a discrete convolution of e_B times the probability distribution $f(p)$.

$$\Pr(H_P = a | H_B = e_B) = \underbrace{(f(p) * \ldots * f(p))}_{e_B \text{ convolutions}}(a) \tag{4.48}$$

$\Pr(P = b+1)$ denotes the probability that the considered packet resides on position $b+1$ within the reordered burst. The probability to be the bth packet in a burst equals the ratio of the fraction of bursts with more than b packets to the average number of packets per burst:

$$\Pr(P = b) = \frac{\sum_{j=b}^{\infty} f(j)}{\sum_{j=1}^{\infty} f(j)\, j} \tag{4.49}$$

With these preconditions, the probability distribution of the packet extent becomes:

$$\Pr(E_P = e_P) = \sum_{e_B=1}^{e_P} \Pr(E_B = e_B) \sum_{a=e_B}^{e_P} \Pr(H_P = a | H_B = e_B) \Pr(P = e_P - a + 1) \quad (4.50)$$

For any burst extent value e_B, the second sum considers any combination of the packet position and the number of packets in the preceding bursts to account for the destined packet extent value of e_P.

4.4.3 Packet n_r-reordering Metric

For the packet n_r-reordering metric, similar considerations as for the extent metric apply. For the n_r-reordering metric, there are two different cases distinguishable: The packet may either receive a n_r-reordering value of $n_P = 0$ or $n_P > 0$. A packet receives a n_r-reordering value of $n_P > 0$ only if the following conditions apply:

- The burst belonging to that packet arrives out-of-sequence.
- The burst receives a n_r-reordering value of $n_B > 0$.
- The packet is the first packet within this burst, i. e., $\Pr(P = 1)$ (cf. Eq. (4.49)).
- Within the n_B preceding bursts n_P packets reside. The probability is $\Pr(H_P = n_P | H_B = n_B)$ (cf. Eq. (4.48)).

A packet receives a n_r-reordering value $n_P = 0$ if the following conditions apply:

- The burst belonging to that packet arrives out-of-sequence.
- The burst receives a n_r-reordering value of $n_B = 0$ and the considered packet is the first packet or the burst receives a n_r-reordering value of $n_B > 0$ and the packet is on any other position but not the first.

If $\Pr(N_B = n_B)$ denotes the probability distribution of the n_r-reordering metric of the burst, then the corresponding distribution for the packet $\Pr(N_P = n_P)$ is

$$\Pr(N_P = n_P) = \begin{cases} \sum_{a=1}^{\infty} \Pr(N_B = a)\ \Pr(H_P = n_P | H_B = a)\ \Pr(P = 1) & \text{for } n_P > 0 \\ 1 - \sum_{b=1}^{\infty} \Pr(N_P = b) & \text{for } n_P = 0 \end{cases} \quad (4.51)$$

The equation distinguishes two cases, $n_P > 0$ and $n_P = 0$. The probability for $n_P > 0$ consists of sum of all possible burst n_r-reordering values a. The sum consists of three factors. The first factor denotes the probability that the burst receives a n_r-reordering value of a. The second factor denotes the probability that there are exactly n_P packets in the a preceding bursts. The third factor denotes the probability that the considered packet is the first packet of the burst. The equation derives the probability for $n_P = 0$ from the complementary probability.

4.4.4 Packet per Burst Distributions

This section gives an overview on selected distributions of the number of packets per burst. These distributions depend on two major network parameters, the traffic characteristic and the burst assembly strategy. The traffic characteristic and the assembly strategy span a two-dimensional co-domain. One dimension reflects the assembly strategy and the other the traffic characteristic. The following paragraphs provide an overview on common packet per burst distributions $f(p)$ for selected traffic arrivals characteristics and assembly processes.

Size based assembly: the assembly process monitors a size threshold of the assembly stage. If the size counter reaches the threshold, the assembly stage forwards the burst. Thereby, the packet size distribution and the threshold limit determine the packet per burst distribution. As this requires further assumptions on the packet size distribution, a closed form solution is very hard to obtain. The reader is referred to the detailed description of the burst size distribution in [46].

Random selection: every packet arrival determines with probability ϱ if the burst leaves the assembly stage or not. The packet per burst distribution is thereby independent of the traffic characteristic, but the inter-arrival time is not. For the packet per burst distribution follows:

$$f(p) = \begin{cases} \varrho\,(1-\varrho)^{p-1}, & \text{if } p > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.52}$$

Packet count: the packet count assembly process guarantees bursts with a constant number of packets. A counter observes the actual number of packets in the assembly stage. If the counter reaches a threshold, the burst leaves the assembly stage. With a packet count limit of $n_C > 0$, the packet per burst distribution becomes:

$$f(p) = \begin{cases} 1, & \text{if } p = n_C \\ 0, & \text{otherwise} \end{cases} \tag{4.53}$$

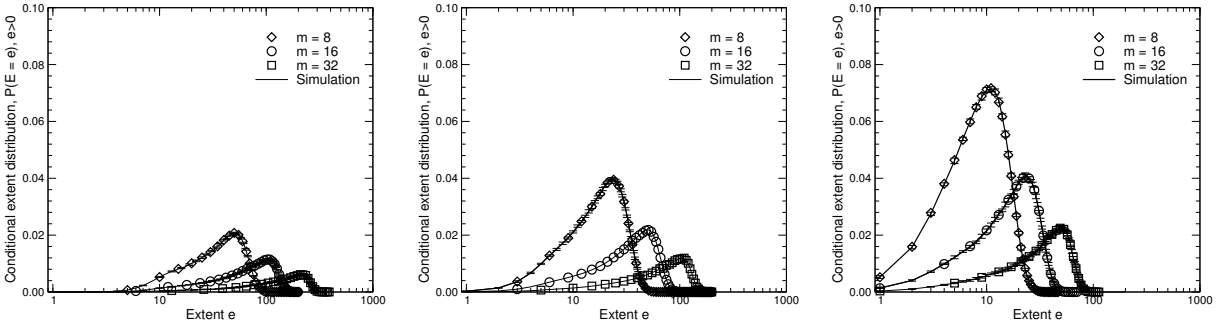
Timer based assembly: the assembly process starts a timer with the first arriving packet on an empty buffer. After reaching the timeout T, the burst leaves the assembly stage. The packet per burst distribution depends on the traffic characteristic of the arriving packets. For two widely applied traffic characteristics, a closed form solution is feasible.

- Constant inter-arrival time t_C.

$$f(p) = \begin{cases} 1, & \text{if } p = 1 + \left\lfloor \frac{T}{t_C} \right\rfloor \\ 0, & \text{otherwise} \end{cases} \tag{4.54}$$

- Poisson arrival process with mean arrival rate λ.

$$f(p) = \begin{cases} \frac{(\lambda T)^{p-1}\, e^{-\lambda T}}{(p-1)!}, & \text{for } p > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.55}$$



(a) $v = 8$ (b) $v = 4$ (c) $v = 2$

Figure 4.14: Packet extent distribution, linear network delay

Besides these basic assembly strategies, combinations out of several of them are feasible, too, e. g., size and threshold based. Then, the packet per burst distribution is in general hard to obtain in a closed form. As this is not the focus of this work, the reader my check beside others the work of Vega et al. in [46].

The previous findings assumed an aggregated traffic stream with mean rate λ. The next paragraphs concentrate on the packet per burst distribution of individual traffic flows rather than the aggregate. The assembly process applies a timer based assembly strategy. Thereby, the aggregate traffic rate λ represents a superposition of individual traffic flows with mean λ_j, i. e., $\lambda = \sum_j \lambda_j$. For each flow, the packet per burst distribution also follows a Poisson distribution, i. e.:

$$f_j(p) = \frac{(\lambda_j T)^p \; e^{-\lambda_j T}}{p!} \tag{4.56}$$

For sake of simplicity, all superposed flows show the same mean rate. If there are v flows, the average rate of each flow becomes $\lambda_j = \lambda / v$. Consequently, the packet per burst distribution for each individual flow becomes:

$$f_j(p) = \frac{(\lambda T / v)^p \; e^{-\lambda T / v}}{p!} \tag{4.57}$$

Therein, the domain of $\lambda T / v$ is $(0, \lambda T]$, depending on v. The lower bound reflects the scenario, with an infinite number of sources, while if there is only one source, the upper bound applies. At the same time, $\lambda T / v$ denotes the mean number of packets per burst. As the total amount of packets per burst is irrelevant, the product is normalized to 1, i. e., $1 = \lambda T$. The probability distribution simplifies to the following equation, with the only parameter v indicating the number of flows sharing one burst assembly unit: The domain of v is $[1, \infty]$.

$$f_j(p) = \frac{\left(\frac{1}{v}\right)^p \; e^{-\frac{1}{v}}}{p!} \tag{4.58}$$

4.4.5 Validation and Illustration

This section illustrates the reordering measures for the packet layer and illustrates by simulation that formal methods and simulation fit. The approach to validate the packet

reordering measures includes the following steps:

1. The simulation model of Section 4.2.2 obtains the reordering metrics on burst and packet level. The simulation varies the number of packet flows v, which correspond to the traffic load.

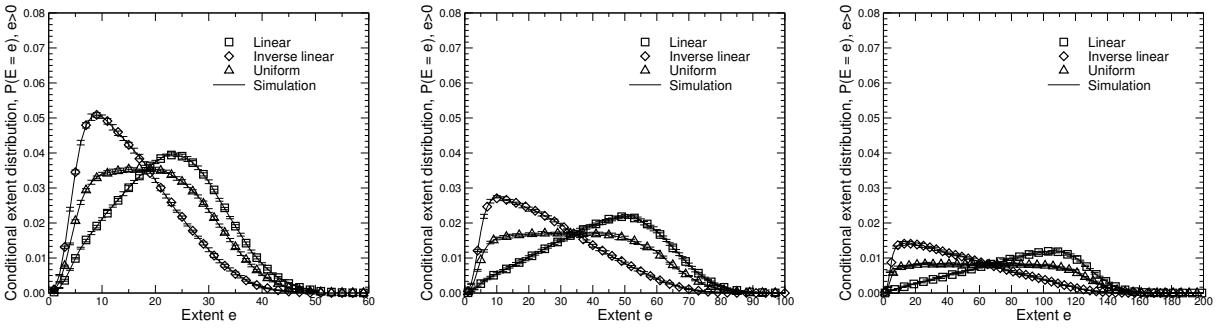
2. With help of the burst reordering metrics and the packet per burst distribution (depends on v and Eq. (4.56)), it is possible to calculate the packet reordering metrics of above.

3. The comparison of the analytic approach of Section 4.4 with the results obtained by simulation evaluates the findings.

The probabilities of the disordering network are linearly distributed. The assembly mechanism of the assembly unit is time based. The graphs of Figure 4.14 depict the packet extent distribution for various $v = 2, 4, 8$. A decreasing v decreases the absolute values of the extent, i.e., reordering or large extent values becomes unlikely. The increasing number of abstract links $m = 8, 16, 32$ in combination with the larger delay on these links favours reordering.

Figure 4.15 selects the scenario for $v = 4$ and depicts the packet reordering extent for selected network delay probability distributions and number of abstract links. The different network delay distributions show different extent values, whereby the shape of the curve of the network delay distribution corresponds to the shape of the graph of the extent distribution. The increasing number of abstract links further increases the absolute value of the extent but decreases the probability as well. Besides the general observations on these figures, all graphs show clearly the compliance of the results of the formal analysis and the simulation results. The values of the analytic findings always lie precisely within the confidence interval of the simulation.

Compared to the reordering metrics of the burst scenario (cf. Figure 4.9), the packet graphs show larger absolute extent values than the burst graphs. As the maximum extent value is $2m-1$ in the burst scenario, this value is multiplied by the mean number of packets per burst and reaches quickly much larger values with a highly decreasing probability. The graphs depicting the findings for the n_r-reordering metric show the same match of analytic as well as simulation. Consequently, these figures were skipped.



(a) $m = 8$ (b) $m = 16$ (c) $m = 32$

Figure 4.15: Packet extent distribution, $v = 4$

5 Application of Analytic Reordering Models

In this chapter, the analytic reordering model of Chapter 4 is applied to enable protocol performance studies with respect to packet reordering. The reordering model with respect to deterministic traffic allows two distinct applications. First, it enables a worst-case estimation of the amount of reordered packets for an arbitrary disordering network. This applies for scenarios with advance knowledge on the network delay characteristic, i. e., obtained from previous measurements. Second, testing real world applications in selected reordering scenarios requires the emulation of packet reordering pattern in real testbeds. Here, the analytic model with deterministic traffic supplies the parameters for network emulation. Additionally, the network emulation is able to create the required reordering pattern independent of the traffic and consequently allows any protocol study.

This chapter firstly introduces both application scenarios of the reordering model in more detail. Secondly it proves the worst-case estimation of the first application scenario and provides illustrative results. The third section introduces the methodology to obtain the parameters of the disordering network for a given reordering pattern. It requires the inverse of the functions of Chapter 4, which obtain the reordering metrics. This leads to a constraint non-linear optimization problem. As a last section, this chapter introduces the implemented network emulation environment, which enables packet reordering according to predefined reordering patterns.

5.1 Introduction to Application Scenarios

This section introduces in detail both application scenarios of the analytic reordering model showing deterministic traffic. The first section introduces the worst-case estimation. The second section imposes the application for network emulation.

5.1.1 Worst-case Estimation on the Reordering Ratio

This application starts from a measurement scenario, which provides the end-to-end delay distribution of a certain connection. The measurement data were obtained by recording the experienced delay from source to destination from every packet or burst. This distribution depends on network load, network resources, multi-path routing features, and

potential interplay of network and protocols, i. e., the protocols may react on network conditions.

Given these measurements, protocol engineers are interested in the amount of reordering an arbitrary traffic stream may produce under these delay conditions. Thereby, this study neglects the potential interplay of protocol and network and assumes an open loop traffic stream, i. e., the sender receives no information of the network conditions, e. g., streaming applications like video and voice. In such a scenario, the analytic model provides an upper bound on the amount of reordering of the arbitrary traffic stream.

The reordering model of Chapter 4 requires a discrete network delay distribution. Consequently, an end-to-end delay measurement has to be adapted to this model first. The step-size for the model corresponds to the traffic mean rate of the traffic stream, which should be evaluated. Formally, if $h(x)$ denotes the cdf of the measured end-to-end delay, the following equation derives the delay probabilities per abstract link for the model: Eq. (5.1) and Eq. (5.2) determine the probability for the abstract link 0, i. e., the abstract link which shows no additional delay. The probability is determined by the smallest value of $h(x)$, which is larger than zero. Eq. (5.3) and Eq. (5.4) discretize the distribution function in intervals of the basic delay unit Δ.

$$p_0 = h(u) \tag{5.1}$$

$$\min_u \{h(u)\} > \epsilon, \qquad \text{where } \epsilon \text{ is small} \tag{5.2}$$

$$p_k = h(u + k\Delta) - p_{k-1} \qquad \forall k \geq 1 \tag{5.3}$$

$$\min_m \{h((m+1)\Delta)\} > 1 - \epsilon, \qquad \text{where } \epsilon \text{ is small} \tag{5.4}$$

This discretization step approximates the original network delay distribution. With this network delay distribution, the model of Chapter 4 is able to determine the expected reordering metrics for deterministic and Poisson traffic. As the studied protocol may show a different traffic characteristic than the deterministic traffic, an extension of the model is necessary. Section 5.2 shows that deterministic traffic serves as an upper bound for any kind of traffic. Any other traffic model with the same mean value does not create more reordering than the traffic model with deterministic traffic.

5.1.2 Network Emulation

While the previous section expects the network delay distribution, this application scenario assumes that the reordering metrics (i. e., reordering extent, n_r-reordering) of a certain end-to-end connection are present. The reordering metrics may be the result of simulation or measurement studies (e. g., obtained from the simulation model of Section 4.2.2). The obtained values represent the measures of a selected scenario, with certain conditions, i. e., network load and multi-path routing. This application scenario of the reordering model reproduces the same reordering pattern than measured. It enables to study the protocol mechanisms and the protocol performance facing this reordering pattern.

Network emulation is one methodology to study these protocol mechanisms. Network emulation duplicates the functions of one system within a different system. In this case,

the functions represent the network properties (i. e., reordering pattern). The different system is a device or module, which is able to show the same network properties than the studied network. Network emulation enables the investigation of real protocols in the end-systems by emulating the network properties in between.

In this thesis, network emulation reproduces the reordering pattern. The input traffic stream originates from the studied protocol. The resulting output reordering pattern follows the predefined pattern and should be independent from the arriving traffic characteristic. Consequently, the network emulation should implement a delay mechanism, which does not rely on time, but on the number of packets passed by. The reordering model with deterministic traffic closes the gap between the reordering pattern and the delay-per-packet of the network emulation. This is possible as the deterministic traffic inherently provides the relation between the arrival order and the delay. Section 5.3.1 and Section 5.4 provide the details on this application.

5.2 Worst-case Estimation on Reordering

This section proves that the analytic reordering model with deterministic traffic serves as a worst-case traffic scenario for the amount of reordering with respect to any other traffic model showing the same traffic mean value. The first section introduces the definition of the worst-case scenario. The second section provides the proofs, while the last section verifies the correctness by simulation.

5.2.1 Definition of Worst-case Scenario

Comparing two reordering patterns requires the definition of a measure to determine which packet sequence shows more reordering than the other. A common indicator for any reordering is the reordering ratio, i. e., the amount of reordering in the network (cf. Section 3.1.2). The reordering ratio will serve as a comparison measure for two different reordering patterns.

Definition 5.1 (Reordering measure)**.** *Packet sequence S_1 has got a more severe impact on the network performance than packet sequence S_2 if the reordering ratio of S_1 is larger than the reordering ratio of S_2:* $\Pr(S_1) > \Pr(S_2)$.

With this definition, one can state the following theorem:

Theorem 5.1. *The reordering ratio of traffic stream $\mathfrak{G}$ showing an arbitrary inter-arrival time distribution with mean $\mathsf{E}[T_{IJ}]$ is less or equal than the reordering ratio of a traffic stream $\mathfrak{C}$ showing constant inter-arrival times t_C with the same mean inter-arrival time $t_C = \mathsf{E}[T_{IJ}]$ on the same disordering network. According to Eq. (4.4), this translates to the following proposition:*

$$\Pr(\mathfrak{G}) \overset{!}{\leq} \Pr(\mathfrak{C}) \tag{5.5}$$

5.2.2 Formal Proof for Deterministic Traffic

This section proves that deterministic traffic leads to the highest reordering ratio with respect to any other traffic model, which shows the same traffic mean rate than the deterministic traffic. First, this section introduces prerequisites, thereafter, the proofs for a disordering network with a single and multiple abstract links will follow.

5.2.2.1 Prerequisites for the Formal Proof

The formal proofs contrast deterministic traffic with an arbitrary traffic characteristic without further assumptions on the characteristic. Thereby, the proof neglects the size of the bursts as the focus remains on the order of the bursts. The formal proofs consider the test burst and its arrival instance relative to previous and later arriving bursts. For these considerations, the test burst marks the origin of a time axis as depicted in Figure 4.7 on page 84. At the top, the figure depicts the burst arrival instances of the deterministic traffic, while at the bottom it depicts arbitrary traffic arrivals.

The discrete disordering model shows a slotted time axis in intervals of Δ. For deterministic traffic, at each border of a slot exactly one packet resides. In case of arbitrary traffic, the number of packet arrivals depends on the traffic characteristic. The random variable of the number of packet arrivals within slot k is X_k (cf. Section 4.3.2.2.2). For a certain delay of the test burst i, $\vec{X}^{(i)} = (X_1, \ldots, X_i)^T$ denotes the random variable for the number of burst arrivals within all slots.

5.2.2.2 Formal Proof for a Single Alternative Link

This section proofs Theorem 5.1 for exactly one abstract link, i. e., $m = 1$. The mean rate of the generic traffic model and the mean rate of the deterministic traffic model correspond to each other, i. e., the mean value equals the basic delay unit $t_C = \mathsf{E}[T_{IJ}] = \Delta$. The reordering model consists of the analytic reordering model of Chapter 4 for deterministic traffic with $m = 1$ additional abstract link.

Proof of Theorem 5.1 for $m = 1$. For one alternative branch, the following inequality must hold to proof Eq. (5.5).

$$\underbrace{\Pr(\mathfrak{G})}_{\text{Eq. (4.46)}} \leq \underbrace{\Pr(\mathfrak{C})}_{\text{Eq. (4.4)}} \tag{5.6}$$

$$p_1 \left(1 - \sum_{n=0}^{\infty} p_n(\Delta)\, p_1^n\right) \leq p_1\,(1 - p_1) \tag{5.7}$$

The left side shows the reordering probability of an arbitrary traffic arrival stream (cf. Eq. (4.46)), while the right side represents the reordering probability for deterministic traffic as derived in Eq. (4.4). As there is only one alternative link, the test burst must follow this link for a potential out-of-sequence arrival. For a real out-of-sequence arrival

of the test burst there has to be at least one arrival in the subsequent interval of the test burst, which does not follow this alternative link. The complementary distribution within the brackets denotes this. Rearranging and simplifying the equations takes the following steps:

$$\begin{aligned} p_1 &\leq \sum_{n=0}^{\infty} p_n(\Delta)\, p_1^n \\ p_1 &\leq p_0(\Delta) + \sum_{n=1}^{\infty} p_n(\Delta)\, p_1^n \end{aligned} \tag{5.8}$$

Applying Eq. (4.43) leads to

$$\begin{aligned} p_1 &\leq 1 - \sum_{n=1}^{\infty} p_n(\Delta) + \sum_{n=1}^{\infty} p_n(\Delta)\, p_1^n \\ p_1 - 1 &\leq \sum_{n=1}^{\infty} p_n(\Delta)(p_1^n - 1) \\ 1 &\geq \sum_{n=1}^{\infty} p_n(\Delta)\frac{1-p_1^n}{1-p_1} \\ 1 &\geq p_1(\Delta) + p_2(\Delta)\frac{1-p_1^2}{1-p_1} + p_3(\Delta)\frac{1-p_1^3}{1-p_1} + \ldots \end{aligned} \tag{5.9}$$

A different representation of Eq. (4.42) is

$$1 = p_1(\Delta) + 2\,p_2(\Delta) + 3\,p_3(\Delta) + \ldots \tag{5.10}$$

Inserting Eq. (5.10) in Eq. (5.9) leads to:

$$p_1(\Delta) + 2\,p_2(\Delta) + 3\,p_3(\Delta) + \ldots \geq p_1(\Delta) + p_2(\Delta)\frac{1-p_1^2}{1-p_1} + p_3(\Delta)\frac{1-p_1^3}{1-p_1} + \ldots \tag{5.11}$$

The next two lines proof that each coefficient n of the left side is equal or larger than the corresponding coefficient on the right side.

$$\begin{aligned} n &\geq \frac{1-p_1^n}{1-p_1}, \quad n \geq 1 \\ n &\geq \underbrace{1 + p + p^2 + p^3 + \ldots + p^{n-1}}_{n \text{ summands each} \leq 1} \end{aligned} \tag{5.12}$$

As the sum of the right side consists of n terms, where each is smaller (except the first one) than 1 the sum is less than n. This proves Eq. (5.9) and consequently proves, that the constant inter-arrival time model reflects an upper bound for the reordering probability for any traffic model for one additional delay line ($m = 1$). □

5.2.2.3 Formal Proof for Multiple Abstract Links

This section proves Theorem 5.1 for multiple abstract links, i.e., $m > 1$. The following steps show parallelisms to the single abstract link case, but require additional mathematical methods.

Proof of Theorem 5.1 for $m \geq 1$. According to the reordering probability of Eq. (4.4), Eq. (4.46) formulates an equivalent equation to determine the reordering probability for arbitrary traffic $\mathfrak{G}$. Compared to deterministic traffic, the structure of the formula remains the same, although it changes the dimension of the vector and the interpretation of the slot sizes. According to the theorem, Eq. (5.5) must hold for any distribution of p. As a first step, the proposition demands that each individual term (i. e., each p_i) of Eq. (4.46) is smaller than the corresponding term of Eq. (4.4). If this is true, Eq. (5.5) is true, too. This restriction leads to the following inequality, where the expression of $\mathfrak{G}$ (from Eq. (4.46)) stands left and the expression of $\mathfrak{C}$ (from Eq. (4.4)) stands right (the inequality sign changed because of resolving the complementary probability):

$$\begin{aligned} &\sum_{n=0}^{\infty} \Pr(N_t = n) \sum_{\vec{x}^{(i)} \in S_n^i} \Pr(\vec{X}^{(i)} = \vec{x}^{(i)} | N_t = n) \prod_{k=1}^{i} {q_k}^{x_k} \geq \prod_{k=1}^{i} q_k \\ &\underbrace{\sum_{n=0}^{\infty} \Pr(N_t = n) \sum_{\vec{x}^{(i)} \in S_n^i} \Pr(\vec{X}^{(i)} = \vec{x}^{(i)} | N_t = n) \prod_{k=1}^{i} q_k^{x_k - 1}}_{\mathsf{E}\left[g(\vec{x}^{(i)})\right]} \geq \underbrace{1}_{g(\mathsf{E}\left[\vec{X}^{(i)}\right])} \end{aligned} \tag{5.13}$$

In Eq. (5.13) the double sum represents the expected value of the function $g(\vec{x}^{(i)}) = \prod_{k=1}^{i} q_k^{x_k - 1}$. The expectation of $\vec{x}^{(i)}$ is $\mathsf{E}\left[\vec{x}^{(i)}\right] = \vec{1}$, i. e., $x_k = 1, 1 \leq k \leq i$ (cf. Eq. (4.42), Eq. (4.45)). Consequently, the right side of the inequality is $1 = g(\mathsf{E}\left[\vec{x}^{(i)}\right])$. With these settings the right and left side of Eq. (5.13) can be simplified:

$$\mathsf{E}\left[g(\vec{x}^{(i)})\right] \geq g(\mathsf{E}\left[\vec{x}^{(i)}\right]) \tag{5.14}$$

With the inequality of Jensen [110] for convex functions φ the following relation holds:

$$\begin{aligned} \varphi\left(\frac{\sum a_i x_i}{\sum a_i}\right) &\leq \frac{\sum a_i \varphi(x_i)}{\sum a_i} \quad \text{where } a_i > 0 \\ \varphi\left(\mathsf{E}\left[\vec{x}^{(i)}\right]\right) &\leq \mathsf{E}\left[\varphi(\vec{x}^{(i)})\right]. \end{aligned} \tag{5.15}$$

Consequently, if $g(\vec{x}^{(i)})$ is a convex function, then Eq. (5.13) is true, which proves the upper bound characteristic of the reordering model. Section A.1 shows the detailed proof, which agrees with the initial assumption. □

The proof showed that deterministic traffic models always create more reordering than any other traffic model if they share the same traffic mean value and if the traffic mean value corresponds to the basic delay unit of the network delay distribution.

5.2.2.4 Extension Towards Changing Traffic Mean Values

This section considers the situation where the disordering network still shows discrete delays, i. e., the model remains the same, but the traffic mean rate does not correspond to the basic delay unit. This section shows that Theorem 5.1 also holds for increasing traffic mean rates. If the traffic mean rate is smaller than the basic delay unit, the bound does not hold any more.

Proof of Theorem 5.1 for $\mathsf{E}[T] < \Delta$ *and* $m \geq 1$. This proof starts at Eq. (5.13). For deterministic traffic, the random number of arrivals within one slot is now Y_k. The probability of y_k occurrences within one slot is 1 as it is deterministic traffic. Consequently, the number of arrivals in i intervals for deterministic and generic traffic in the mean is now $\sum_{k=1}^{i} \mathsf{E}[X_k] = \sum_{k=1}^{i} \mathsf{E}[Y_k] = \gamma\, i$, where γ is a positive real number $\gamma > 0$ (the mean rate of deterministic traffic and generic traffic are equal). If $0 < \gamma < 1$, than the inter-arrival time is larger compared to the original basic delay unit Δ. If $\gamma > 1$, than the inter-arrival time is smaller than the original basic delay unit.

The following paragraphs consider the reordering ratio for both traffic types. The structure of the left side keeps the same (with a different expression for $\Pr\left(\vec{X}^{(i)} = \vec{x}^{(i)}, N = n\right)$ as the mean rate changes), while the right side slightly changes.

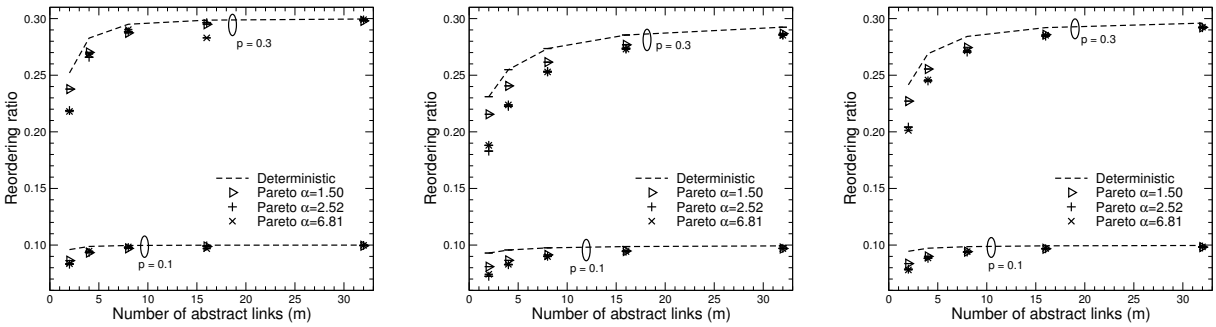
$$\begin{aligned} &\sum_{n=0}^{\infty} \Pr(N_t = n) \sum_{\vec{x}^{(i)} \in S_n^i} \Pr(\vec{X}^{(i)} = \vec{x}^{(i)} | N_t = n) \prod_{k=1}^{i} {q_k}^{x_k} \geq \prod_{k=1}^{i} q_k^{y_k} \\ &\underbrace{\sum_{n=0}^{\infty} \Pr(N_t = n) \sum_{\vec{x}^{(i)} \in S_n^i} \Pr(\vec{X}^{(i)} = \vec{x}^{(i)} | N_t = n) \prod_{k=1}^{i} {q_k}^{x_k - y_k}}_{\mathsf{E}\left[g(\vec{x}^{(i)} - \vec{y}^{(i)})\right]} \geq \underbrace{1}_{g(\mathsf{E}\left[\vec{x}^{(i)} - \vec{y}^{(i)}\right])} \end{aligned} \tag{5.16}$$

The sum of the left side of Eq. (5.16) again represents an expected value as in Eq. (5.13). The corresponding function is now $g(\vec{x}^{(i)} - \vec{y}^{(i)}) = \prod_{k=1}^{i} {q_k}^{x_k - y_k}$. According to Jensen's inequality $g(\cdot)$ needs to be convex. Additionally, the right side of the inequality should suffice $1 = g(\mathsf{E}\left[\vec{x}^{(i)} - \vec{y}^{(i)}\right])$, which is also true as Eq. (4.45) and Eq. (4.42) still hold. Consequently, Jensen's inequality holds. However, this is not fully true as the number of relevant intervals changes as shown in the following. The following paragraphs distinguish between a larger traffic mean rate $\gamma > 1$ and a smaller traffic mean rate $\gamma < 1$ compared to the original model, where the mean inter-arrival time equals the basic delay unit.

$\gamma \geq 1$**:** if the mean inter-arrival time is smaller compared to the original inter-arrival time the traffic mean rate increases. In this case, deterministic traffic still serves as an upper bound for the amount of reordering and Jensen's inequality holds. The findings of above apply.

$0 < \gamma < 1$**:** if the mean inter-arrival time is larger compared to the original mean, then traffic mean rate decreases. For deterministic traffic this means that there are inherently intervals without any arrival (due to the larger spacing of the arrivals compared to the delay unit). Especially, the first interval after the test burst does not contain any burst because the inter-arrival time is larger than the interval size of Δ. Consequently, the model changes as for deterministic traffic the first abstract link is meaningless. For the general traffic model, the first interval still is important as there is a probability that bursts occur in there, which may enable reordering.

Summarizing, deterministic traffic serves as an upper bound for reordering in a network, if the traffic mean rate is equal or larger than the corresponding basic delay unit Δ. In any other case, the upper bound approximation does not hold anymore. The following section illustrates this. □



(a) Linear prob. dist. (b) Inverse linear prob. dist. (c) Uniform prob. dist.

Figure 5.1: Illustration for Pareto traffic

5.2.3 Illustrative Results

This section illustrates Theorem 5.1 for two different scenarios. The first scenario shows that the amount of reordering reaches its maximum for deterministic traffic if the traffic mean rate is equal for all three scenarios. It validates that deterministic traffic serves as an upper bound. The second scenario shows that the deterministic traffic model serves as an upper bound even when the traffic mean rate increases. For decreasing traffic mean rates, the assumption on the bound is not valid anymore. The first section describes the scenario if the traffic mean rate equals the basic delay unit, while the second section describes the scenario if traffic mean rate and basic delay unit do not correspond.

For a comprehensive analysis, the analytic reordering model of Section 4 faces the traffic of three different traffic models: deterministic, Poisson and Parato traffic. The distribution function of Pareto traffic is

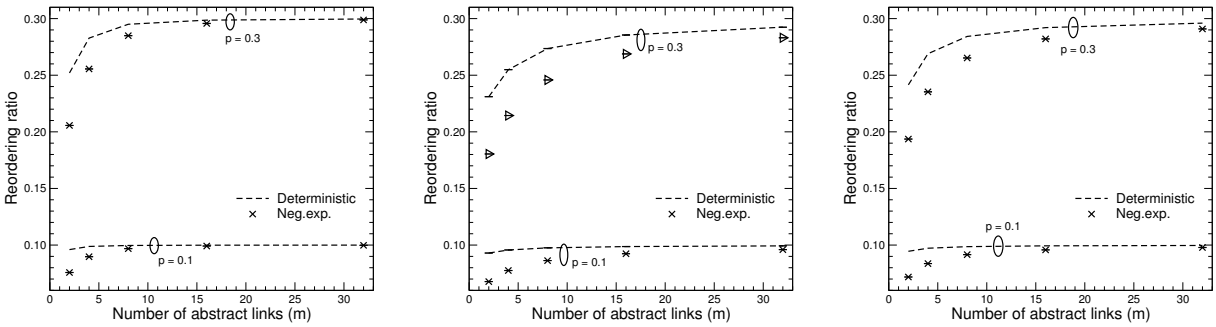
$$\Pr(X < x) = \begin{cases} \left(\frac{k}{x}\right)^{\alpha} & \text{for } x \geq k \\ 1 & \text{for } x < k \end{cases} \tag{5.17}$$

It includes two parameters k, the minimum value, and α, the Pareto index. The expected value of this distribution is $\mathsf{E}[X] = \frac{\alpha k}{\alpha - 1}$ for $\alpha > 1$. For $\alpha \leq 1$ the expected mean is infinite.

5.2.3.1 Traffic Mean Rate Equals the Basic Delay Unit

In this section, the traffic mean rate equals the basic delay unit of the network delay distribution. It considers Poisson and Pareto arrivals. In case of the negative exponentially distributed inter-arrival times, this directly relates to the parameter of the distribution. For the Pareto traffic model, this section shows three different parameterizations with the same mean rate: $\alpha_1 = 6.81, k_1 = 0.77$; $\alpha_2 = 2.52, k_2 = 0.59$ and $\alpha_3 = 1.5, k_3 = \frac{1}{3}$.

The graphs of Figure 5.1 show the calculated values of the reordering metric for deterministic traffic (dashed line) and the simulated values for the Pareto distributed traffic for an increasing number of abstract links (using the simulation model of Section 4.2.2).



(a) Linear prob. dist. (b) Inverse linear prob. dist. (c) Uniform prob. dist.

Figure 5.2: Illustration for Poisson traffic

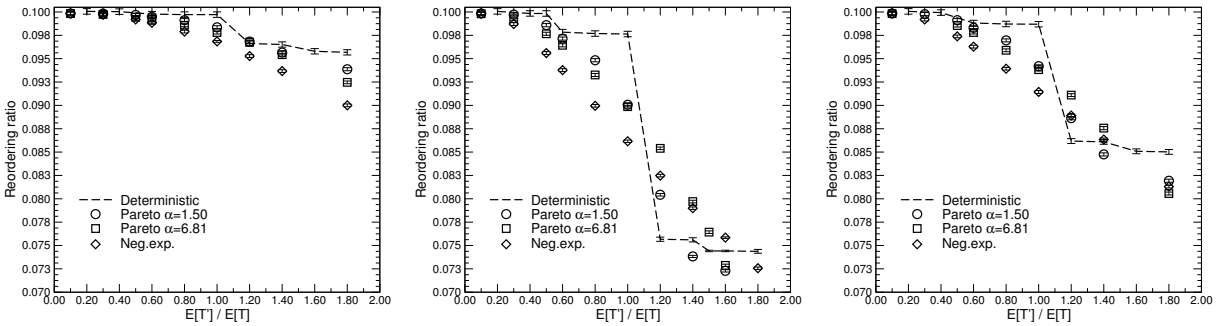
For reasonable results, the statistical values are obtained from ten batches each including at least one million burst arrivals (the figures also depict the confidence intervals). The figures illustrate the formal proof for probabilities of $p = 0.1$ and $p = 0.3$. Therein, the probability delay distribution of the disordering network is one of the following three according to Section 4.3.3: linear, inverse linear, uniform.

The figures show, that for a small number of abstract links, the reordering probability in all cases is much lower than the branch probability p. With the increasing number of links, the reordering ratio approximates the branch probability of 0.1 and 0.3, respectively. Regarding the Pareto traffic model, the reordering probabilities for different values of α is very similar if $\alpha > 2$. For $\alpha < 2$, the reordering ratio becomes larger as the variation increases. Nevertheless, the reordering ratio of all Pareto parameterizations is well below the corresponding value of the deterministic traffic. Thereby, the number of links has a significant impact on the reordering ratio while the network delay distribution has only minor influence.

The graphs of Figure 5.2 show the results for negative exponentially distributed inter-arrival times. Again, the figures depict the reference value for the deterministic traffic showing the same mean than the Poisson traffic. The findings for the Poisson traffic are equivalent to the findings of the Pareto traffic. The reordering ratio is always below the deterministic value as well as the reordering value approaches the branch probability for a large number of links. Comparing the reordering ratio to the corresponding value of the Pareto traffic highlights that Poisson traffic shows in general a smaller reordering ratio than Pareto traffic.

5.2.3.2 Traffic Mean Rate Unequal the Basic Delay Unit

This section demonstrates the findings for a different traffic mean rate. Inhere, the traffic mean rate does not correspond to the basic delay unit of the disordering network but is either smaller or larger. The graphs in Figure 5.3 illustrate that a deterministic traffic model creates maximum reordering ratio if the traffic mean rate is equal to or larger than the corresponding basic delay unit. The graphs illustrate the findings for the three different network delay distributions (linear, inverse linear and uniform). The scenario



(a) Linear prob. dist. (b) Inverse linear prob. dist. (c) Uniform prob. dist.

Figure 5.3: Illustration of upper bound with respect to changing traffic mean

shows $m = 8$ abstract links and a branch probability of $p = 0.1$. The dashed lines stand for deterministic traffic while the symbols stand for Pareto traffic and traffic showing Poisson characteristics. The x-axis shows the changes in the traffic mean rate by the fraction of the actual arrival rate $\mathsf{E}[T']$ and the basic delay unit $\mathsf{E}[T] = \Delta$.

These sample simulations back-up the findings of the previous section. For a ratio smaller than 1, i. e., higher traffic mean rate, deterministic traffic still serves as an upper bound. If the traffic mean rate decreases, the reordering ratio decreases in general, but the deterministic traffic model does not serve as an upper bound any more for all cases. Here, the difference network delay distributions (cf. Section 4.3.3) influence the overall characteristic. For instance, in case of a linear probability distribution, the reordering model still serves as an upper bound. This characteristic changes in case of a uniform or inverse linear probability distribution. The reordering model provides less reordering than expected by the traffic models. A reason for this is the inherent modification of the probability distribution for deterministic traffic. If the traffic mean rate decreases, the inter-arrival time gets larger. Consequently, the abstract link 1 does not have any impact on reordering as its delay is smaller than the inter-arrival time. In case of linear distribution this impact is rather small, while in both other distributions the impact on the overall branch probability is significant. With an increasing inter-arrival time, the lower number abstract links 2, 3 get idle and the branch probability further decreases.

5.3 Determining the Disordering Network

This section introduces the second application case for the analytic reordering model, i. e., network emulation. Thereby, the network emulation generates packet arrivals characteristics including out-of-sequence packets according to a given characteristic. Thereby, the out-of-sequence characteristic should follow predefined reordering patterns to analyse protocols under exactly these conditions. As only delayed packets create out-of-sequence arrivals at the destination, it is necessary to derive the network delay showing the same reordering pattern as predefined.

This section applies the findings for the analytic reordering model for deterministic traffic. This model provides the knowledge on how to delay the packets to receive a predefined

reordering pattern. These steps include an inversion of the equations of the reordering model leading to a non-linear constraint optimization problem. This section first introduces the optimization problem and the applied solver. Second, if imposes starting points for the solver algorithm.

5.3.1 Optimization Problem

This section applies the functions of the reordering model in reverse direction to obtain the distribution of the probabilities of the disordering network with respect to a given reordering pattern.

The functions to obtain the probability distributions of the reordering extent and the n_r-reordering are discrete, non-linear and in an open form (cf. Eq. (4.4), Eq. (4.13), Eq. (4.19)). In an abstract view, these functions map an n-dimensional input vector (probability distribution of the disordering network) onto an r-dimensional output vector (reordering metric distribution). Abstracting these functions by f, leads to the following generic equation:

$$\vec{y} = f(\vec{x}), \quad \text{where } \vec{x} \text{ is a probability vector} \tag{5.18}$$

Therein, $\vec{y}$ represents either the probability distribution of the reordering extent or the ccdf of the n_r-reordering metric according to the model. $\vec{x}$ represents in either case the probability distribution of the disordering network. If the vector degrades to a scalar, the function realizes the reordering ratio of Eq. (4.4).

A correct representation of the experienced reordering pattern requires the configuration of the parameter $\vec{x}$, i. e., probability distribution of the disordering network. This leads to the inversion of the reordering functions, i. e., $\vec{x} = f^{-1}(\vec{y})$, which is analytically hard. An alternative approach reformulates this problem to a constraint based optimization problem using the original functions for the reordering metrics. If $\vec{y'}$ denotes the estimated reordering vector, then Eq. (5.19) depicts the non-linear optimization problem with one single constraint:

$$\begin{aligned} &\min_{\vec{x}} ||\vec{y'} - f(\vec{x})|| \qquad \text{with the norm } ||\vec{a}|| = \sqrt{\sum_i a_i^2} \\ &\text{and constraint } 2 = \sum x_i + \sum |x_i| \end{aligned} \tag{5.19}$$

Eq. (5.19) describes how to find an $\vec{x}$, which minimizes the difference of the expected and the obtained reordering pattern. The second equation formulates the constraint of $\vec{x}$ being a probability vector.

For this kind of problems various solvers and tools exist. The Optimization Toolbox of Matlab [111] provides a suite of solvers. For the optimization problem of Eq. (5.19), this thesis applies the solver `fmincon`. The solver tries to find a minimum of a constrained non-linear multivariable function. The solver applies the active set algorithm, which the following literature describes in depth: [112–115]. This thesis skips the details on this algorithm as it is out of scope of this thesis. Chapter 6 evaluates the results of the solver

in detail. It introduces measures to quantify the match of $\vec{y}$ and $\vec{y'}$ and illustrates the findings at several examples.

The solver expects the function f with an initial starting point $\vec{x}_0$ and a set of constraints including scalar or matrix constraints. Additionally, parameters of the step size, the termination tolerance on the function value and the vector $\vec{x}$ configure the solver. This work configures the solver using the following set of parameters.

- Minimum step size of $\vec{x}$: 10^{-5}
- Termination tolerance of f: 10^{-8}
- Termination tolerance of the constraints: 10^{-8}

These parameters realize a resolution of $\vec{x}$ in the range of 10^{-4}, which is sufficient for this study. The choice of the starting point is topic of the next section.

5.3.2 Estimation of the Starting Point

The applied solver requires an initial starting point $\vec{x}_0$. This section provides a heuristic for an initial guess of the network delay distribution. The initial guess requires two parameters, the dimension of $\vec{x}$ and its initialization. The size of $\vec{x}$ and the maximum possible value of the reordering metrics (reordering extent, n_r-reordering) correlate. The relation of the vector length (m) of $\vec{x}$ and the maximum possible reordering extent $\hat{e}$ is: $\hat{e} = 2\,m - 1$. Consequently, the dimension of the network delay vector becomes $m = \left\lceil \frac{\hat{e}+1}{2} \right\rceil$. If the network offers $m + 1$ different paths from source to destination, then there is one path showing the minimum delay, while m paths delay the packet. The maximum value of the n_r-reordering metric $\hat{n}$ and the dimension of the vector also correlate: $\hat{n} = m$. Consequently, the dimension of the network delay vector becomes easily $m = \hat{n}$.

The above steps determined the size of the probability distribution vector. The starting point of the optimization algorithm requires an initial value of this vector. As the correlation between the network delay probability vector and the reordering metric is highly non-linear, a direct relation is hard to estimate. Therefore, for sake of simplicity, the initial starting point is proportional to the given reordering vector. The initial starting point takes the first m values of the distribution of the reordering metric (extent, n_r-reordering) and assigns them to $\vec{x}_0$ after normalization to 1. With this starting point, the solver starts its search to solve the optimization problem. In Section 6 the obtained solutions are discussed. The obtained solution serves as an input parameter for network emulation as discussed in the next section.

5.4 Network Emulation

Network emulation is the major application of the reordering model including the proposed methodology. To emulate reordering, network emulation may either delay arbitrarily

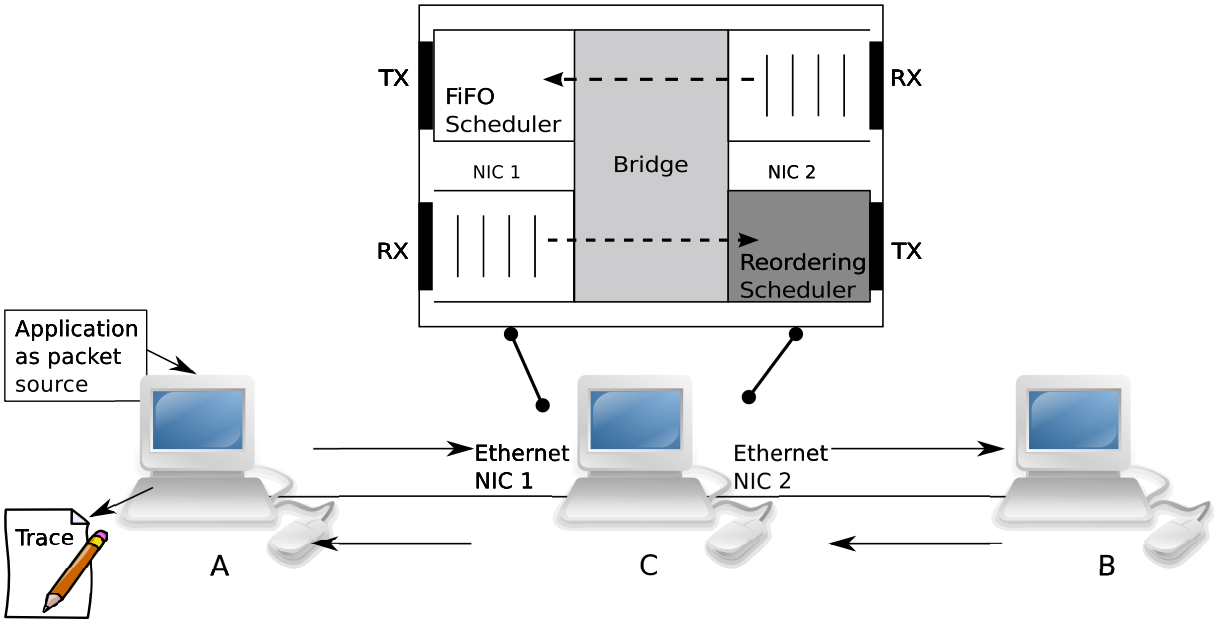


Figure 5.4: Network emulation setup

packets by either a predefined time interval or may implement special mechanisms that the out-of-sequence pattern at the destination node equals a predefined reordering pattern. The latter point is subject of this section.

Most of the commercial and free network emulation tools implement a timer mechanism to enable packet reordering, e. g., NetEm [116], NIST [117]. They delay packets randomly or by a certain predefined delay. They get out-of-order because of the delay and subsequent packet arrivals. These network emulation tools provide only unspecific packet reordering as the reordering depends on the correlation of both, the delay and the traffic characteristic. Consequently, these attempts exclude a structured and well defined method to quantify the impact of a predefined reordering pattern on an application.

None of the tools known by the author implement an algorithm which is able to produce an expected reordering pattern. The purpose of this section is to show the application of the reordering model in a network emulation tool. Thereby the network emulation tool is able to create any predefined reordering pattern. The objective of this approach is to study the influence of packet reordering on the protocol performance.

The next section imposes the network emulation setup, which has been setup for this thesis. The second section introduces basic mechanisms the Linux traffic control offers. The last section introduces in detail the network emulation module, which is responsible to delay the packets to experience a certain reordering pattern.

5.4.1 Network Emulation Setup

This section introduces the general network emulation testbed. Figure 5.4 shows the network emulation setup. It consists of three nodes interconnected by Ethernet [802.3].

The outer two nodes represent some personal computers, where A generates the packets for B and B replies the packets back to A. The node in the middle C is also a personal computer (PC). It is responsible for the network emulation, especially the reordering properties. For an easy setup, the nodes run Linux operating systems.

The network emulation PC in the middle shows two interfaces connected to A and B, respectively. They are locally bridged together forming a virtual layer 2 Ethernet switch [118], i. e., the network emulation is transparent to any IP communication. Each direction and interface provides virtual output queuing. The scheduler at each virtual output queue decides on the outgoing packet sequence according to the queuing discipline. In the network emulation scenario, the direction from B to A applies a simple FIFO scheduler, while the reverse direction from A to B applies the reordering scheduler. The remaining part of this section uses the term *packet shuffler* for the reordering scheduler at the right outgoing interface.

The Linux Traffic Control framework [119] is able to manipulate the scheduler at the outgoing interface and to apply a different queuing discipline than FIFO. This thesis applies this framework to realize the introduced packet shuffler. The following section gives a detailed introduction to this framework and the realization of the packet shuffler.

5.4.2 Linux Traffic Control

The Linux traffic control environment provides mechanisms and functions to enable traffic control within a router. Traffic control provides mechanisms to alter the traffic characteristic to achieve certain objectives. These objectives may include – among others – network performance improvements, network protection and quality of service. The following list describes typical mechanisms to realize these objectives [119]:

Shaping aims at changing the inter-arrival statistics of a packet flow by delaying individual packets. Tasks are to shape on a maximum or on an average rate. Common mechanisms implement token bucket and leaky bucket algorithms [120].

Scheduling arranges the order of the packets in a queue waiting to be served. The simplest model implements a FIFO principle. Other mechanisms are fair queuing mechanism, last-in-first-out, round robin or random selection.

Classifying applies selected criteria on packets and assigns them to common classes, which are identically processed. Methods for classification mostly check protocol header fields on any protocol layer, e. g., protocol type, packet length, port number.

Policing describes the decision principle to accept a packet or not. In most cases, policing checks if packets are aligned with a negotiated service level agreement. For instance, the service level agreement specifies a maximum rate. In general, policing is a yes/no decision on each data unit without any further distinction.

Dropping discards packets or entire flows. The decision to drop a packet may depend on the policing or the classification process.

Marking changes a packet for later identification. Marking may be the result of a previous classification or policing process.

The next section applies these basic concepts to design a network emulation module, which is able to create a predefined reordering pattern.

5.4.3 Reordering Module

This section firstly introduces the Linux traffic control environment of [119] and secondly shows its application for the packet shuffler. The focus on the latter part is the realization of the statistical nature of packet reordering. It provides an introduction to the mathematical background for random number generation and especially highlights the limitations of the Linux kernel.

5.4.3.1 Linux Kernel Reordering Module

This section gives a short introduction in the Linux kernel modules providing network emulation features. First, it introduces the network emulation module [116], which provides the basic framework and describes its features and drawbacks. The second section provides an overview on the realized reordering emulation module of this thesis.

5.4.3.1.1 Network emulation module

This section gives an overview on the realized kernel module implementing the packet shuffler of Section 5.4. It relies on the kernel module NetEm [116] and provides additional mechanisms to enable packet reordering. The NetEm module implements a scheduler and realizes the interfaces of the Linux traffic control environment introduced before. The NetEm module enables the following network emulation features: packet corruption, delay, drop, re-sequencing. The user may parameterize these features with mean and variation or even distributions for the packet delay. The re-sequencing (reordering) feature provides three different configurations:

- Delay every n^{th} packet by certain mean and variation.
- Delay a certain percentage of packets by certain mean and variation.
- Enable reordering by variation of the packet jitter.

This approach is able to produce a certain reordering pattern, which depends on the applied reordering method as well as the input traffic characteristic. As stated earlier, the combination of delay distribution and arrival characteristics generates reordering events. Because of this direct relation, any changes in the input traffic characteristic changes the output pattern. Consequently, it is hard to investigate a protocol with respect to a

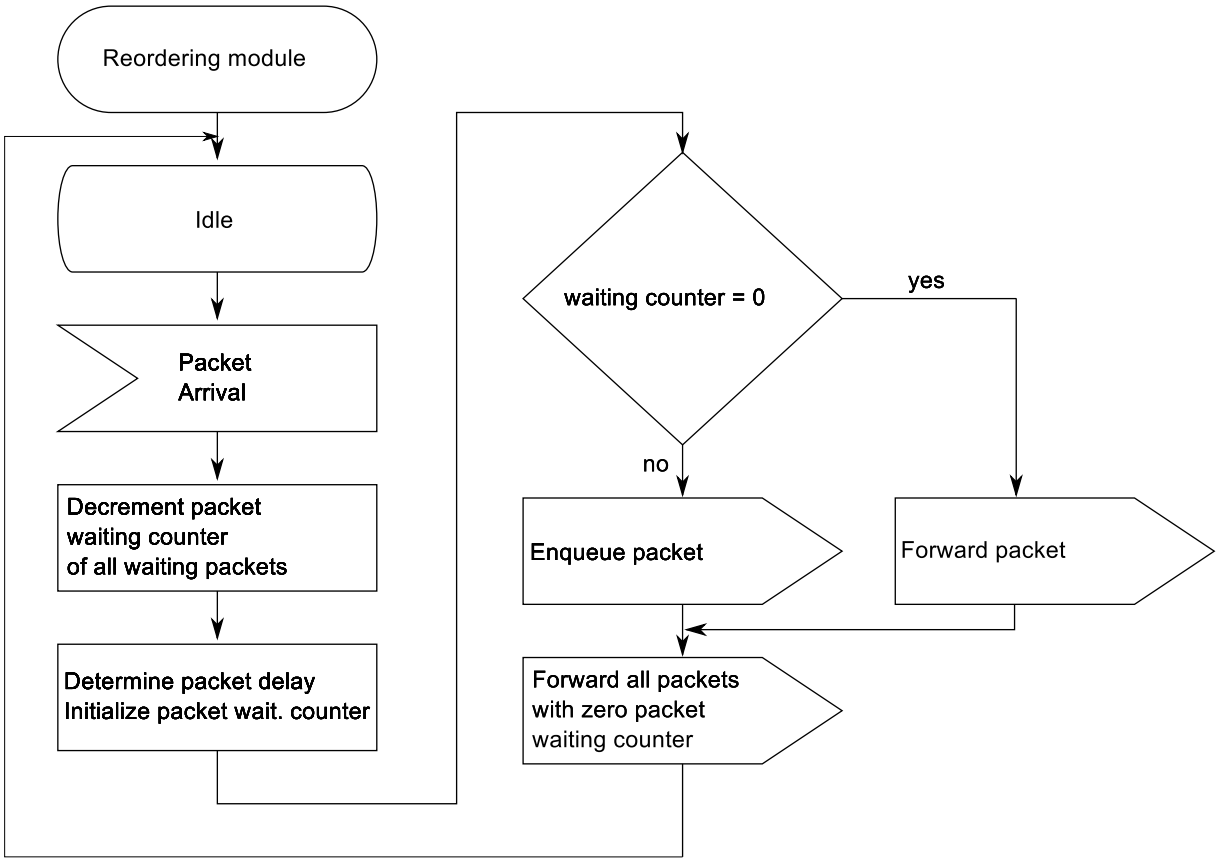


Figure 5.5: Process description of the reordering module

certain reordering pattern, as the traffic pattern is determined by the protocol itself and potentially changing.

Although this module is inflexible in this case, the individual packet handling is rather simple as its delay is time based. Additionally, the module is able to delay packets by the number of packets passed by. As this holds for any traffic, it excludes any randomness. In all three of the above modifications, a delayed packet receives an extra delay budget. The packet scheduler processes the packets in FIFO order and transmits them with zero delay budgets. Consequently, the state of each delayed packet depends only on its due time and not on the subsequent arrivals of packets. Any packet delay is independent of later arrivals and handled individually.

5.4.3.1.2 Reordering emulation module

The previous section showed the strict dependency of the reordering pattern on the arrival traffic pattern and the delay in the network. If the reordering pattern is fixed due to selected scenarios of reordering patterns, the network delay needs to depend on the number of subsequent packet arrivals rather than on a time budget. If the packet shuffler delays packets by a certain number of subsequent packets, this scenario equals the reordering model with deterministic traffic. As stated earlier, the number of packets passed and the delay correspond in the deterministic traffic scenario.

This realization requires the scheduler to maintain a packet counter for each packet. This counter indicates the number of packets to wait until the scheduler forwards this packet. For packet-based waiting, the data structure of the waiting packets has been extended to store the packet counter. Besides changes in the data structure, this also requires changes in the scheduler. The original time based scheduler now organizes the work based on the waiting counter of each packet. The scheduler reduces the packet counter of each packet by one at each packet arrival instance. Packets with zero counter value leave the queue, while other packets remain in the queue. Figure 5.5 depicts the process of the packet scheduler, which performs packet reordering according to a predefined reordering pattern. The next list provides the details on each individual step.

1. The process remains in idle state until a new packet arrives.
2. A new packet arrives.
3. The packet arrival triggers the process to decrement the packet waiting counter of all waiting packets by one. The scheduler iterates over the list of waiting packets and manipulates the packet counter.
4. For the new packet, according to the probability distribution of the disordering network, the kernel randomly chooses the number of packets to wait for.
5. If the random number equals zero, the packet is forwarded immediately. In any other case, the scheduler queues the packet in the buffer.
6. In the last step all packets in the queue with zero packet waiting counter are forwarded applying strict FIFO principle.

This packet shuffler is able to realize predefined reordering patterns. The solution of the solver of Section 5.3.1 defines the distribution of the random packets to wait for. It corresponds to the probability distribution of the analytical reordering model. The problem of this random choice lies in the implementation. This step requires a random number generator, which creates random numbers following a predefined distribution. In the Linux kernel, this is hard to realize as the following sections show. The next sections state the problem and introduce its solution.

5.4.3.2 Random Number Generation in the Linux Kernel

The reordering module requires emulating the splitting process of the disordering network (cf. Section 4). The splitting process determines the random delay of each packet in terms of packets to wait for. The random delay follows a discrete distribution.

For random number generation, the Linux kernel provides a 32 bit pseudo uniformly distributed random variable [121]. The following sections introduce the necessary steps to receive random numbers distributed according to a predefined discrete distribution. This requires firstly the inversion of the probability distribution function and secondly the generation of a random event.

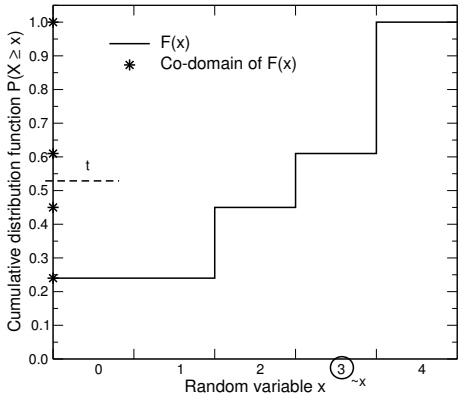


Figure 5.6: Visualzation of $F(x)$ and its co-domain

5.4.3.2.1 The inverse of the cumulative distribution function

The waiting time of a packet X (in packets) is a discrete random variable with the following cumulative distribution function:

$$F(x) = \Pr(X \leq x) = \sum_{i=0}^{x} p_i = \sum_{i=0}^{x} f(i) = t, \quad 0 \leq t \leq 1 \tag{5.20}$$

The inverse function of $F(x)$ is $G(t)$, where $G(t)$ may include holes as $F(x)$ is a sum of discrete values. Because of these holes, $G(t)$ yields to the smallest $\tilde{x}$, where $F(\tilde{x})$ is equal or larger than t. Eq. (5.21) shows the formal representation of this argument.

$$G(t) = \tilde{x} \qquad \text{where } \min_{\tilde{x}} \{F(\tilde{x}) \geq t\}, \quad 0 \leq t \leq 1 \tag{5.21}$$

Figure 5.6 visualizes this relation on an example. It depicts $F(x)$ and its co-domain with marked values (star) on the ordinate. The selected value of t (between 0.5 and 0.6) does not correspond to any value of the domain of x. In this example, $\tilde{x} = 3$ because $F(3) \geq t$ while $F(2) < t$, cf. Eq. (5.21).

5.4.3.2.2 Random event generation

This section introduces the generation of the uniform distributed value of t and the representation of G within a table of the Linux kernel as proposed by Chen in [122] and Bratley in [123]. The Linux kernel represents G in a table T with N rows. Therein, the ith entry of G holds $\tilde{x}$ as defined in Eq. (5.21). These preconditions formulate the following statement: Let U be a discrete random variable with a uniform distribution in domain $1 \ldots N$, then Y shows approximately the same distribution than X, with $Y = T(U)$. Thereby $H(x)$ is the cumulative distribution function of Y. The next paragraphs validate the above statement. The proof includes the probability distribution and the cumulative

distribution function of the newly created random variable Y.

$$\begin{aligned} H(x) =& P(Y \leq x) \\ =& i/N \quad \text{where} \quad \min_i \{ \underbrace{G(i/N)}_{\min_x\{F(x)\geq i/N\}} \} \geq x \quad \text{(cf. Eq. (5.21))} \\ =& \lfloor F(x)\, N \rfloor / N \quad \text{as } i \text{ is an integer} \end{aligned} \tag{5.22}$$

The second row determines the ith row, where the corresponding value of $F(x)$ is equal or larger than i/N. As i is an integer, $\lfloor F(x)\, N \rfloor$ defines the probable row. Therefore, the shape of $H(x)$ and $F(x)$ are similar except the floor function. With an appropriate table size N, the difference of $H(x)$ and $F(x)$ converges to zero by $\frac{1}{N}$.

5.4.3.3 Module Parameters

The approach of the last section requires the definition of the table size N. The table size N influences the quality of the approximation of the distribution of the original random variable X. From the previous section the absolute error between both distributions is less than $\frac{1}{N}$. Another criteria is the resolution of the original distribution function $F(x)$. Two neighboured values of X should lead to two different rows in the table to realize the corresponding random variable Y. The following relation determines the smallest distance of neighbouring values, i. e., the resolution of the distribution of X.

$$\min_x \left(F(x+1) - F(x)\right) = f(x+1) = \epsilon \qquad \text{where } \epsilon > 0 \tag{5.23}$$

The resolution of the table T is $1/N$. For a suitable resolution of the table, the following equation must hold:

$$1 \leq N\epsilon \tag{5.24}$$

As a result, the table size N should ensure $N \geq 1/\epsilon$. The experiment uses a table size of $N = 8192$ resulting in a resolution of about $\epsilon \approx 10^{-4}$. This corresponds to the resolution of the solver of Section 5.3.1. The following example illustrates the procedure with $\epsilon = 0.05$ and a corresponding table size of $N = 20$.

Example 5.1. *The next lines depict the probability distribution and the cumulative distribution function of the network delay distribution as well as the resulting table.*

$$\begin{aligned} f(0) &= 0.9 & F(0) &= 0.90 \\ f(1) &= 0.05 & F(1) &= 0.95 \\ f(2) &= 0.05 & F(2) &= 1.00 \end{aligned}$$

According to Section 5.4.3.2.2 the table holds the following values.

$$\begin{aligned} T(\ 1) &= G(\ 1/20) = 0 \\ T(\ i) &= G(\ i/20) = 0, \quad 1 < i < 19 \\ T(19) &= G(19/20) = 1 \\ T(20) &= G(20/20) = 2 \end{aligned}$$

A uniformly distributed random number applies the modulo operation with the table size N and chooses the appropriate table row. This leads to the desired distribution of X.

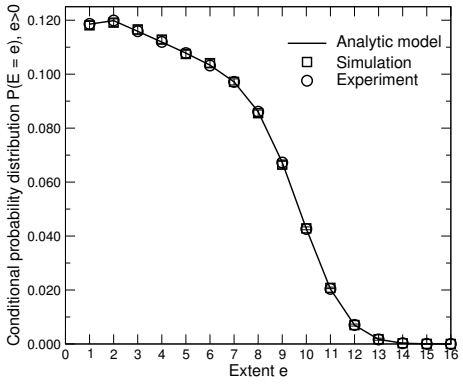


Figure 5.7: Validation of analytic model, reordering emulation and simulation

5.4.3.4 Module Verification

The testbed of Figure 5.4 including the reordering emulation module has been setup and tested. This section provides a comprehensive analysis of the testbed (cf. Figure 5.4) in comparison to the analytic model (cf. Section 4.3.1.2) and the values obtained from simulation (cf. Section 4.2.2). These different scenarios start with the same traffic model and the same disordering network. The reordering pattern is subject to comparison. In all three scenarios, the resulting reordering patterns should be the same.

The parameters of the common reordering scenario are the following: branch probability is $p_0 = 0.7$, $m = 10$ abstract links with a uniform network delay distribution, i. e., $p_i = 0.3/10, 1 \leq i \leq m$. The traffic model is deterministic where the inter-arrival time corresponds to the basic delay unit of the network delay environment. In the testbed, node A of Figure 5.4 sends 10^6 packets with inter-arrival time of $\Delta = 100\ \mu s$ to node B. Node B echoes all packets in reverse direction[1]. Node A traces the receiving packet sequence numbers and analyses the reordering pattern offline using the IETF metrics of Section 3.2.5. The reordering module in node C of Figure 5.4 delays packets according to the process of Figure 5.5. The simulation parameters of the simulation model in Section 4.2.2 and the analytic expression for the reordering extent in Section 4.3.1.2 apply the same parameters, i. e., $m = 10$, $p_i = 0.3/10, 1 \leq i \leq m$.

Figure 5.7 shows the values obtained by the testbed, the analytic model and the simulation model. The figure illustrates the probability distribution of the reordering extent metric of the packet layer. The three curves perfectly overlap. These findings back-up the analytic model and verify the correctness of the testbed implementation. Consequently, the testbed suits for any investigation on reordering of a given network delay distribution. As the results for any other configuration of the disordering network or the n_r-reordering metrics provide no additional findings, they were skipped.

[1]Linux ping command enables this functionality

5.4.3.5 Evaluation

This section shortly evaluates the Linux kernel reordering module and argues its drawbacks and potential limitations. In principle, the module shows two drawbacks. The first drawback addresses the changes of the traffic characteristic and the second drawback addresses some improvements to alleviate infinite waiting.

The reordering module delays packets according to a certain number of packets passed by (e. g., deterministic traffic model). These delays change the traffic characteristic, i. e., incoming and outgoing traffic characteristics show differences. In this case, the protocols face two different network anomalies, i. e., packet reordering as well as changes in the traffic characteristic. For a concise study of the out-of-sequence phenomenon, the changes in the traffic characteristic have to be alleviated. One solution to almost restore the original characteristic is to record the observed inter-arrival time at the input and provide a playout buffer releasing these packets following the recorded distribution. Depending on the resolution of the measurements, this approach may approximate the original arrival distribution. As the original traffic characteristic changes in any scenario of packet reordering, it is in general hard to differentiate between the impact of reordering or the impact of the changed traffic characteristic.

The second drawback addresses the delay in the module. As the module implements a packet-based waiting, a finite stream of packets may leave packets in the module waiting infinitely for subsequent packets. This occurs if a packet receives a delay, which is larger than the number of remaining packets of this finite stream of packets. To avoid infinite waiting, additionally to the packet-based waiting, a timeout value may be implemented. This timer guarantees that the packets leave the module after a maximum waiting time. The value of the timer should be large enough not to interfere with the general waiting to obtain a certain reordering pattern.

Summarizing, the proposed reordering module shows two minor drawbacks, which may be alleviated easily. Besides this, they do not interfere with the findings of the previous sections but show enhancements of the module for productive environments.

6 Evaluation of Analytic Reordering Models

The previous chapter applied the analytic reordering model of Chapter 4 in two different directions, i. e., for worst-case estimations on the amount of reordering and in network emulation scenarios to create predefined reordering patterns. The parameterization of the network emulation requires the solution of an optimization problem, which is defined by the predefined reordering pattern and the analytic reordering model for deterministic traffic (cf. Section 5.3.1). The solution of the optimization problem represents a probability distribution of the disordering network of Section 4. This disordering network together with any deterministic traffic should create the same statistical reordering than a predefined reordering pattern. This section quantifies the quality of the solution with respect to the original predefined reordering pattern.

The reordering metrics represent statistical measures, i. e., probability distributions (extent) and distribution functions (n_r-reordering). An evaluation of the goodness of these measures requires special statistical methods for comparison. Statistics refers to these methods as goodness-of-fit tests, which include hypothesis tests as well as graphical methods. The first section introduces both methods for the applied goodness-of-fit tests. The evaluation process requires a predefined reordering pattern. For reasonable data and as an example for networks showing inherent reordering, this thesis analyses the end-to-end reordering of an optical burst switching network. The quantification of reordering in these networks as well as the influence of contention resolution schemes is subject to the second section. The third section evaluates the solution of the optimization problem, for reordering pattern of selected end-to-end connections of the OBS network. The selection includes burst and packet reordering patterns as well as different network load scenarios. The evaluation shows that the solution approximates the desired reordering pattern very well. The last section evaluates the complexity to solve the optimization problem. This includes the computation time to obtain the reordering metrics analytically as well as the time the solver needs for a reasonable solution.

6.1 Evaluation Method

Reordering metrics characterize specific reordering patterns. These reordering metrics represent statistical properties, i. e., probability distributions and distribution functions of reordering extent and n_r-reordering metric, respectively. This section introduces the

statistical methods to evaluate the statistical consistency of two empirical distributions. The results obtained from the analytic model (with the parameterization of the solver) and the results of the simulation need to fit. Literature calls this a *goodness-of-fit* problem. Statistics provides two distinct classes of methods to evaluate if two random variables show the same distribution. The first class allows a visual evaluation, while the second class uses hypothesis tests. Examples of the first class are scatter plot, bar chart, histogram and quantile/quantile plot [124]. In general, hypothesis tests of the second class include the following steps: (a) propose a null or alternative hypothesis, (b) check the assumptions, e. g., independence of the samples, (c) compute the test statistic, (d) decide on the acceptance of the hypothesis by comparing the value of the test statistic to critical values obtained from tables.

The goodness-of-fit problem distinguishes two application cases. The first application case tests if an empirical distribution origins from a known distribution. The second application case tests if two empirical distributions originate from the same parent distribution.

For the first application case, a large number of hypothesis tests inspect sample distributions with respect to normality, i. e., the Normal distribution. Selected tests include the Student's t-test [125]. The Student's t-test is applied to test if the deviation of a random variable follows the Normal distribution. Fisher's Z-test [125] tests the hypothesis that a sample distribution can be approximated by a normal distribution. The application case is similar to Student's t-test but requires a larger number of samples.

The second application case applies for the problem within this thesis. Both distributions obtained by simulation and analytics should match, while their original distribution is unknown. A non-exhaustive list of hypothesis tests (goodness-of-fit tests) for distributions originating from an unknown distribution include the test of Kolmogorov-Smirnov [126], the χ^2-test [125] and the Anderson-Darling test [127]. The next sections introduce the applied quantile/quantile plot for visual comparison, the Kolmogorov-Smirnov and the χ^2-test in detail. As the Anderson-Darling test is similar to the Kolmogorov-Smirnov test with extra weights on the distribution tail, the reader is referred to the original publication [127].

6.1.1 Quantile/quantile Plot

The quantile/quantile-plot (qq-plot, [124]) enables a visual interpretation, if two sample sets follow the same distribution. Therein one distribution acts as a reference distribution while the other represents the sample distribution originating from measurements or simulations. This method requires the cumulative distribution function of both samples. The sample cdf may be $F_X(x)$ and the reference cdf may be $F_R(x)$.

A qq-plot entitles both axis with the random variable X or the cdf value. The points in the graph map $x \to y$, where $x \to F_X(x)$ and $F_X(x) = F_R(y) \to y$. For comparison, the qq-plot includes a straight 45-degree line. If both distributions match, the dots lie close to the straight line. Otherwise, the qq-plot shows shifts and scales compared to the reference distribution. Figure 6.3 on page 128 shows an example of a qq-plot.

6.1.2 χ^2-Test

This section introduces the χ^2-test as it is applied in this thesis. For an in depth introduction in the wide range of applicability of this test, the reader is referred to [125]. The χ^2-test tests the null hypotheses if a given sample distribution and a theoretic or other sample distribution originate from the same distribution. As a measure of goodness-of-fit, the test statistic includes the deviation of both probability distributions.

The χ^2-test requires the probability distribution of the sample random variable X and the probability distribution of the reference random variable R, $f_X(x) = \Pr(X = x)$ and $f_R(r) = \Pr(R = r)$, respectively. With both probability distributions, the χ^2-test defines the test statistic χ^2:

$$\chi^2 = \sum_i \frac{(f_X(i) - f_R(i))^2}{f_R(i)} \tag{6.1}$$

The test statistic recommends a rejection of the null hypothesis if $\chi^2 > \chi(1-\alpha; m)$. Therein, $\chi(1-\alpha, m)$ is the χ-distribution with the significance level α, i. e., 95%, and the number of samples reduced by the number of degrees of freedom m. Otherwise, it accepts the null hypothesis. In this thesis m is 1 as the distribution values come from samples, which are constrained by their number. In this thesis, the χ^2-test applies a significance level of 95%.

One disadvantage of the χ^2-test is its requirement on a large number of samples [125]. If the number of samples is rather small, the Kolmogorov-Smirnov Test enables an alternative testing method. In this thesis, the number of samples forming the distribution was large enough to reach the limits of the required number of samples. Nevertheless, this thesis applies both hypothesis tests to provide profound significance. The next section introduces this additional hypothesis test by Kolmogorov and Smirnov.

6.1.3 Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov test (KS-test, [126]) tests the null hypotheses if the distribution of the empirical values of the random variable X follows the distribution of the reference random variable R. The KS-test requires the empirical cumulative distribution function of X and R, $F_X(x) = \Pr(X \leq x)$ and $F_R(r) = \Pr(R \leq r)$, respectively. The test statistic of the KS-test relies on the maximum distance of both distributions and the number of samples creating both distributions. If $F_X(x)$ relies on n_x samples and $F_R(x)$ relies on n_r samples, the test statistic D_{n_x,n_r} becomes:

$$D_{n_x,n_r} = \sup_x \left| F_X^{(n_x)}(x) - F_R^{(n_r)}(x) \right| \tag{6.2}$$

The KS-test accepts the hypothesis if $\sqrt{\frac{n_x\, n_y}{n_x+n_y}}\, D_{n_x,n_r} \leq K_\alpha$, where K_α is found in tables [126] for a certain significance level α and sample space: $\Pr(K \leq K_\alpha) = 1 - \alpha$. If the number of samples, each increases 35, $\delta = \sqrt{-\frac{1}{2}\ln\left(\frac{1}{2}(1-\alpha)\right)}$ approximates K_α

reasonable well [128]. With this approximation, one rejects the hypothesis if

$$\sqrt{\frac{n_x\, n_y}{n_x + n_y}}\, D_{n_x,n_r} > K_\alpha, \quad \text{if } n_x, n_r < 35$$
$$\sqrt{\frac{n_x\, n_y}{n_x + n_y}}\, D_{n_x,n_r} > \delta, \quad \text{if } n_x, n_r \geq 35$$

In any other case, one accepts the hypothesis. For large n_x and n_r the square root converges to zero, which leads to a general reject of the hypothesis until both distributions correspond exactly.

The above version of the KS-test considers two empirical distributions. Literature names this variant Two Sample KS-test. If the reference sample distribution follows a theoretic distribution, the number of samples grows to infinity $n_r \to \infty$ and the KS-test simplifies to $\sqrt{n_x}\, D_{n_x,n_r} > K_\alpha$ and $\sqrt{n_x}\, D_{n_x,n_r} > \delta$, respectively. In this thesis, the distributions obtained by simulation are tested to the analytically obtained distributions of the reordering model. The distributions from simulations are of empirical nature, while the analytic distributions represent theoretic distributions with an infinite number of samples. As the test is symmetric, the assignment of these distributions to F_R and F_X does not matter. The KS-test requires the number of samples of the empirical distribution function. The simulation records and provides these measures. In the following, the significance level of the KS-test is 95%.

6.2 Optical Burst Switching Network Simulation

This section introduces the optical burst switching (OBS) network simulation environment, which provides the measures of the reordering metrics introduced in Section 3 and calculated analytically in Section 4. The OBS network simulation applies the network simulator of Gauger's dissertation [66] without the optical burst transport network extensions. The simulator has been extended by the reordering metrics of Section 3.2.5 to record the burst reordering pattern. The first subsection gives a brief overview on the major characteristics of this simulator. This includes network and node models as well as implemented and applied contention resolution schemes. The second subsection introduces the set of simulation parameters.

6.2.1 Simulator

The OBS network simulator of this thesis relies on the work of Gauger [66]. It uses the event-driven simulation library SimLib [129] and models the basic properties of an OBS network introduced in Section 2.4.2. Details on the implementation of this simulator provides [66], while this thesis highlights the main properties of this simulator briefly. The network simulator requires dimensioning of network resources, i. e., link and node capacities. This section introduces firstly the network dimensioning process and highlights secondly the most important parameters of the node model.

6.2.1.1 Network Dimensioning

The dimensioning process starts from a population-based traffic demand model, i. e., the amount of data exchanged with the other nodes is relative to their size. These end-to-end demands together with a selected topology and a routing algorithm (e. g., Dijkstra's shortest path) define the required capacity per link between two connected nodes. The routing algorithm passes these demands within the given topology. On each link, the routed demands superpose. The dimensioning process now maps the required capacity onto fibres carrying a certain number of wavelengths. Here, two parameters determine the mapping process: the capacity of a single wavelength and the number of wavelengths per fibre.

The routing and wavelength assignment steps define the physical topology including the resources of the network for a certain load scenario. The scenario of above defines the maximum load scenario of 100%. This maximum load scenario places the demands ideally in the network. In general, performance studies others than 100% have two options to vary the network load. Either they reduce the traffic demand or they increase the available network resources, e. g., link capacity. The simulator of this thesis implements the second option and increases the network resources for load scenarios smaller than 100%. The parameter α describes the amount of overdimensioning the network shows, i. e., $\alpha = 1$ corresponds to the 100% scenario; if $\alpha > 1$, network load decreases.

6.2.1.2 Node Dimensioning

For burst forwarding, the nodes reserve capacity on the next link. Otherwise, if the node corresponds to the destination node of the burst, the node delivers the burst to the receiver. Section 2.4.2 already introduced the architecture of an OBS node in detail. The most important parameters and degrees of freedom of an OBS node implemented in this simulator include the number of input/output ports and the available contention resolution schemes. Thereby, a port is the attachment of one fibre.

The connectivity of a node and the initial dimensioning process defines the number of input/output ports. The required capacity per link together with the wavelength capacity and the number of wavelengths per fibre translates in the number of ports. Consequently, the result of the dimensioning process parameterizes the number of ports of an OBS node. Thereby, in this model the switching matrix does not represent a limitation regarding the scalability of the node.

The contention resolution schemes include wavelength converters, fibre delay lines as well as deflection routing. Thereby, the contention resolution schemes of wavelength conversion and the fibre delay lines are node-local schemes, whereas deflection routing is a contention resolution scheme. Consequently, the parameterization of the wavelength converters as well as the fibre delay lines is node specific. The wavelength converter pool reflects the number of wavelength converters for contention resolution. As wavelength conversion does not contribute to any burst reordering, the number of converters is unlimited to neglect its impact. The resource management of fibre delay lines may be dedicated per port or fibre or shared among all ports. Gauger provides in [66] a detailed classification.

This work assumes shared fibre delay lines, where the number of fibres and the number of wavelengths per fibre represent the major parameters. The different wavelengths of a single FDL may be used in parallel.

6.2.1.3 Contention Resolution Schemes

Section 2.4.2.4 already provided an in depth introduction in contention resolution schemes of OBS networks. This simulator implements wavelength conversion, burst buffering as well as deflection routing. The following list imposes the major parameters of these mechanisms.

Wavelength Conversion (conv): The number of wavelength converters and their organisation determine this parameter.

Fiber Delay Lines (Fdl): The number of FDL lines and the number of wavelengths per FDL define its capability.

Deflection Routing (Defl): The maximum number of alternative paths as well as the maximum number of hops determine this parameter.

Besides these individual mechanisms, the simulator allows combinations of contention resolution schemes. Thereby the order of the schemes reflects the sequence of alternative fallback variants, i. e., if contention cannot be resolved with the first scheme the next scheme applies etc. This thesis applies the following contention resolution schemes, where wavelength conversion *Conv* is always applied first.

Fdl Fibre delay lines

Defl Deflection routing

DeflFdl Deflection routing and fibre delay lines

FdlDefl Fibre delay lines and deflection routing

Besides wavelength conversion, all other contention resolution schemes may change the burst arrival order at the destination node. The simulations in Section 6.3 quantify the amount and characteristic of out-of-sequence arrivals per contention resolution scheme.

6.2.1.4 OBS Traffic Profile

The network load originates from a population-based model. The traffic itself neglects the burst assembly process at the edge but simulates the burst level. This restriction avoids non-scalable multi-timescale simulations on packet as well as on burst level. The simulator abstracts the assembly process by traffic sources. These traffic sources create bursts originating from an assembly process. These sources require three parameters,

Figure 6.1: Reference network, [1]

the burst length distribution, the inter-arrival time distribution as well as the destination distribution. The first two parameters model the burst assembly process, while the second parameter reflects the applied population based demand model.

6.2.2 Simulation Parameters

This section introduces the applied simulation parameters of this study. It differentiates between parameters of the dimensioning process, parameters of the nodes and parameters of the burst generation process. The applied topology follows the pan-European COST 266 reference network in [1]. Figure 6.1 depicts the reference network. The network dimensioning process assumes the following parameters:

- Wavelengths per link: 8
- Line rate per wavelength: 10 Gbit/s
- Link delay (national network): 1 ms
- Resource reservation: just enough time (JET, [69])
- Mean burst length: 12.500 B

The following parameterize the contention resolution schemes per node:

- Number of wavelength converters: unlimited
- Number of FDL per node: 1
- Number of wavelengths per FDL: 32
- Deflection routing: the number of alternatives paths considered at a node is 1, while the number of allowed hops per deflection path is unlimited.

As the number of converter pools is unlimited with shared FDL, the product of the number of wavelengths per FDL times the number of FDL gives the available capacity for contention resolution using FDL. The chosen single FDL with 32 wavelengths relies on previous work [130], which shows reasonable network performances. In total, there are 32 FDL buffer places available, which also reflect a boundary where an additional increase does not lead to significant improvements in the network performance [81].

The sources for optical bursts are parameterized with the following settings. The burst departure process follows a Poisson process with negative exponentially distributed inter-arrival times. This reflects a scenario of a large number of sources multiplexed together, i. e., metro and core networks scenario. The burst size distribution follows the parameters of the dissertation of Gauger described in [66]. The burst size is negative exponentially distributed. This distribution reflects bursts resulting from a tri-model Internet traffic characteristics as Vega et al. studies in [46]. Each created burst randomly destines one of the other nodes. The distribution of destination follows the empirical population based distribution.

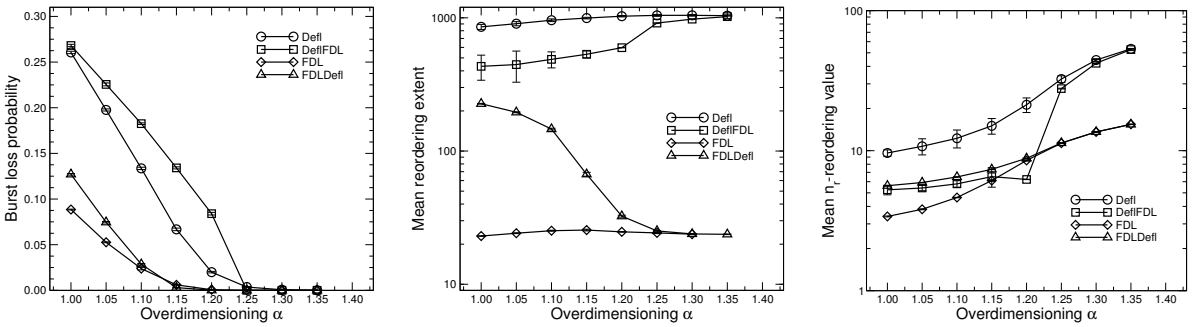
6.3 Comprehensive Results

This section evaluates the results from the optimization problem of Section 5.3.1. The solution parameterizes network emulations, which are able to create the same reordering pattern as the reordering pattern obtained from the OBS network simulation with the above parameter set. It evaluates single layer studies on the burst reordering pattern as well as the results obtained from hierarchical packet reordering. The evaluation process applies the qq-plot as well as the χ^2 and KS-tests introduced before.

The first section presents global findings of the reordering pattern in an optical burst switched network. Therein the reordering patterns per link have been averaged to give an overall impression of the present reordering in such networks. The second section evaluates the findings of the burst layer, while the third section focuses on the evaluation of the packet reordering metrics. The last section evaluates the complexity to solve the optimization problem.

6.3.1 General Observations

This section presents the simulation results of the simulator introduced before. The simulation obtains reordering results for the introduced contention resolution schemes: FDL, deflection routing and combinations of them. Each end-to-end connection records the experienced reordering. The following graphs give an overall impression on the fundamental relation between the mean reordering metrics and the contention resolution schemes for different load scenarios. Figure 6.2a) reviews the contention resolution schemes with respect to burst losses in different network load scenarios. With $\alpha > 1$, i. e., decreasing load, the burst loss probability decreases regardless of the applied contention resolution scheme. This is obvious as the reasons for contention reduce. The figure also depicts that



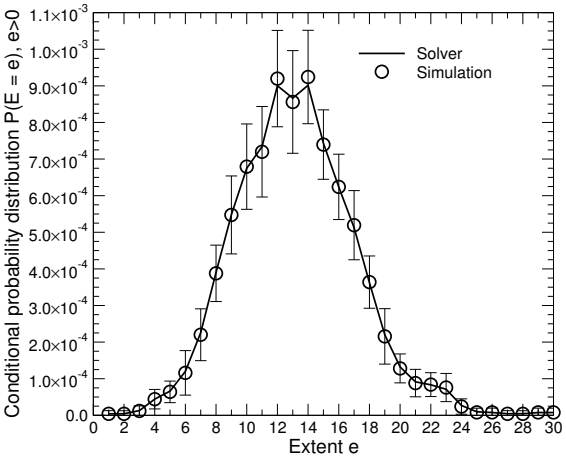
(a) Burst loss ratio (b) Mean reordering extent (c) Mean n_r-reordering

Figure 6.2: Global reordering pattern (mean values)

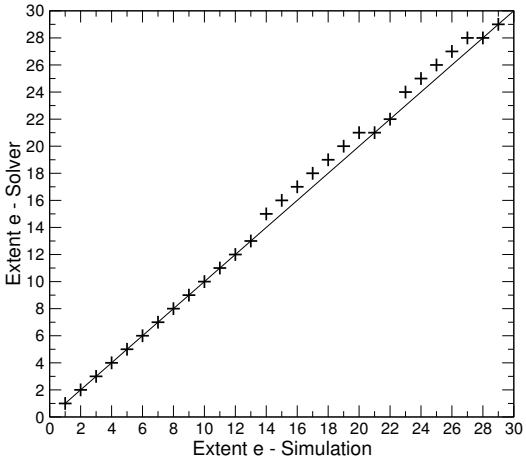
a combination of two schemes always leads to higher burst loss ratios as the corresponding single scheme, e. g., Defl shows less burst losses than DeflFDL. The reason is that the additional scheme occupies additional resources in high load situations and consequently increases the probability of contention. Another general observation is that deflection routing leads to a higher burst loss probability than FDL. This is obvious as deflection routing requires extra links, while FDL applies additional resources to buffer bursts.

Figure 6.2b) shows the results for the mean reordering extent. This graph represents earlier results obtained by Perelló and Gunreben et al. in [26]. If the contention resolution scheme only applies FDL, for the chosen scenario, the average extent value is quite low in the range of about 20-40 independent of network load. The quite small extent originates from the short fibre delay lines compared to the mean inter-arrival time of the burst. If the contention resolution schemes apply also deflection routing in any combination, for this scenario, the extent increases orders of magnitudes, as the experienced delay on alternative deflection paths is orders of magnitudes larger than the inter-arrival time. If the contention resolution scheme applies both, fibre delay lines and deflection routing (FDLDefl), the graph flaps from deflection routing curve towards the curve of pure fibre delay lines. This characteristic reflects the scenario that in high load situations, deflection, while in low load situation, fibre delay lines resolve contention. If the contention resolution scheme applies both schemes in reverse order, the situation changes. In this scenario, contention is resolved in high load situations by deflection routing and fibre delay lines. In load situations, deflection routing mainly resolves contention, i. e., the curve of DeflFDL approaches the curve of Defl.

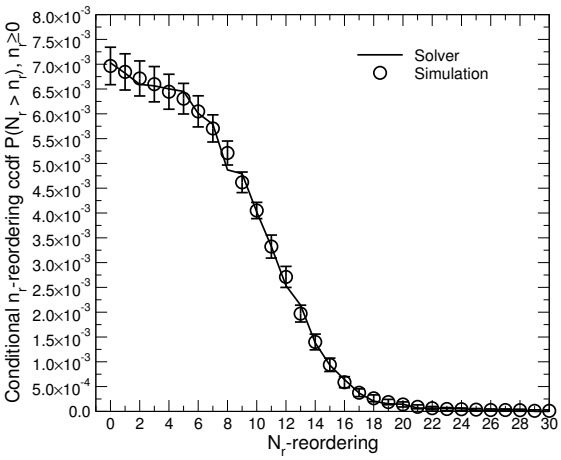
Figure 6.2c) shows the mean of the recorded n_r-reordering metric values. Note that the figure depicts the average values among out-of-sequence bursts. It shows two major differences compared to the mean extent value of Figure 6.2b). First, the average n_r-reordering value is increasing with decreasing load. The reason is that in low load scenarios reordering becomes unlikely. However, if reordering occurs it occurs for single bursts only and produce long series of in-order bursts, which results in large values of n_r (cf. Chapter 3). In high load scenarios, reordering occurs more likely but in these cases, trains of out-of-sequence bursts are rare, too. The findings for the contention resolution schemes DeflFDL and FDLDefl still hold for the n_r-reordering scenario.



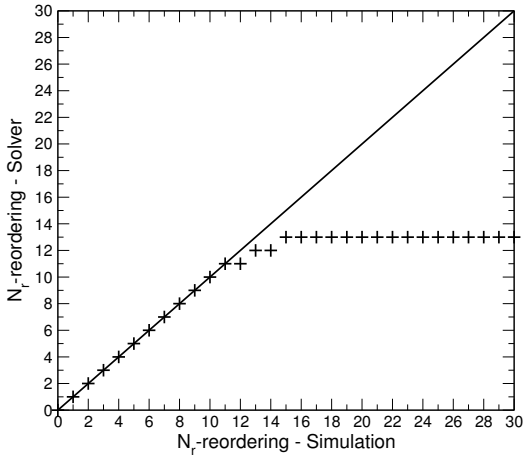
(a) Extent probability distribution



(b) Extent qq-plot



(c) N_r-reordering ccdf



(d) N_r-reordering qq-plot

Figure 6.3: Burst reordering analysis Paris – Rome, $\alpha = 1.35$

6.3.2 Burst Reordering

This section evaluates the results of the optimization problem. It chooses two selected connections from the simulation results as any other connection leads to similar results. The evaluation process includes four different criteria, the two graphical interpretations and the two hypothesis tests of Section 6.1. The graphical evaluations depict the reordering metrics from simulation and from the analytic model together with the solution of the solver in the same figure to give a visual impression on the quality of the solution. Besides this, the qq-plot visualizes the quantile of both distributions. The next paragraphs provide the results for connections Paris—Rome and Zagreb—Munich (cf. Figure 6.1 on page 125). Additionally to the burst results, Section 6.3.3 provides the results for the packet level.

Figure 6.3 depicts the simulation results for bursts of the connection from Paris to Rome at a load scenario of $\alpha = 1.35$. Figure 6.3a) depicts both, the conditional reordering extent distribution obtained from simulation as well as the conditional reordering extent distribution from the analytic model with the solution of the solver. In both and the

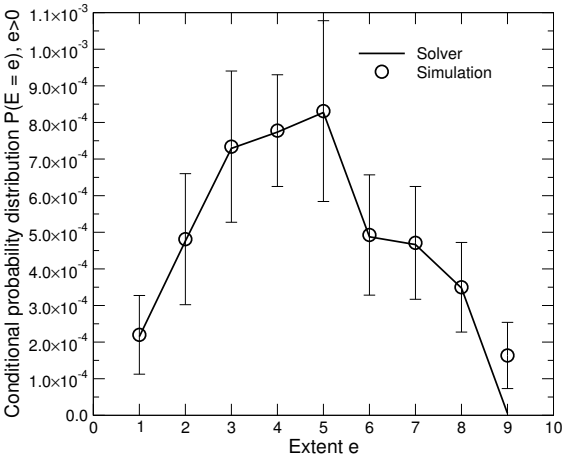
following cases the distribution is conditioned by $e > 0$. The visual expression is that both distributions match and originate from the same distribution. The hypothesis tests (χ^2, KS-test) both agree on this finding and recommend the acceptance of the hypothesis at a significance level of 95%. Additionally to the hypothesis tests, the qq-plot of Figure 6.3b) emphasizes this finding graphically. The values occupy the straight line, which indicates the correlation of both distributions. Besides this straight occupation, some points reside beside the line. The discrete values of the distribution and a small error in the solution create this small variation, i. e., one integer value beside the line still reflects a very good match.

Figure 6.3c) and Figure 6.3d) depict the findings for the n_r-reordering metric. The conditional ccdf of the n_r-reordering metric in the left visualizes the accurate solution of the optimization problem. The solution of the solver resides within the confidence intervals of the distribution and the values of the qq-plot again follow a straight line. Note that in the qq-plot, larger values than 12 follow a horizontal line. This occurs, if one distribution shows a larger upper bound than the other. In this case, the distribution value of one distribution always points to the maximum distribution value of the other distribution. In Figure 6.3d), the distribution from the simulation shows larger n_r-reordering values than the solution of the solver. As the corresponding value of this distribution is quite low, this does not affect the recommendation of the hypothesis tests. Both hypothesis tests, i. e., KS-test and χ^2-test, recommend an acceptance of the hypothesis that both distributions originate from the same distribution. Summarizing, the solution of the solver, i. e., the obtained parameterization of the disordering network for deterministic traffic suits the reordering pattern by simulation. Consequently, the findings of the solver may serve as parameter for the network emulation of Section 5.3.1 to create the same statistical reordering than obtained from simulations.

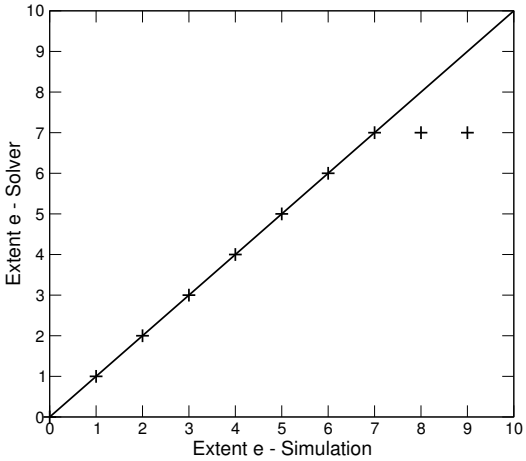
In addition to the findings of the previous link, Figure 6.4 depicts the burst reordering results of the connection Zagrab—Munich for a load of $\alpha = 1.3$. As Zagreb is a node with a rather small traffic volume (population-based traffic matrix) and as it is only two hops to Munich, the number of reordered bursts is small, too. The reordering extent in Figure 6.4a) as well as the n_r-reordering metric in Figure 6.4c) is limited up to values of 10. Also the amount of reordered bursts is very small in the order of $3 \cdot 10^{-3}$. As the event of a reordered burst occurs rarely, the confidence intervals are larger than in the first example. Especially for small n_r-reordering and extent values, the confidence intervals increase strongly. Nevertheless, Figure 6.4a) and c) depict both distributions. The solution of the solver and the simulation results perfectly match. The qq-plots of both metrics as well as both hypothesis tests indicate that both distributions originate from the same distribution. Summarizing with the results of Section 5.4.3.4, the solution of the solver will reproduce the same statistical reordering in network emulations as observed in this simulation.

6.3.3 Packet Reordering in Packet Assembly Networks

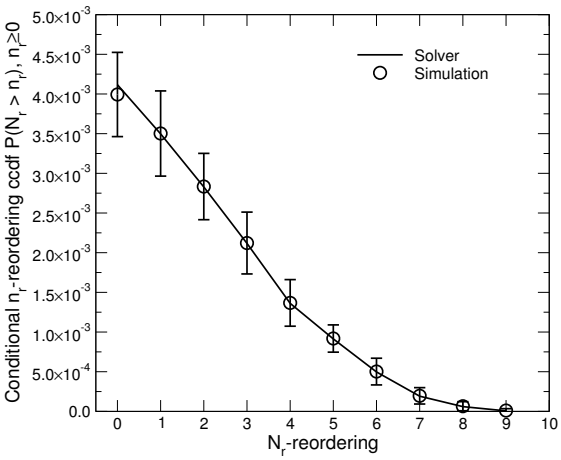
The last section showed exemplarily that the solver is able to obtain the required parameterization for burst reordering. The simulation results directly define the optimization



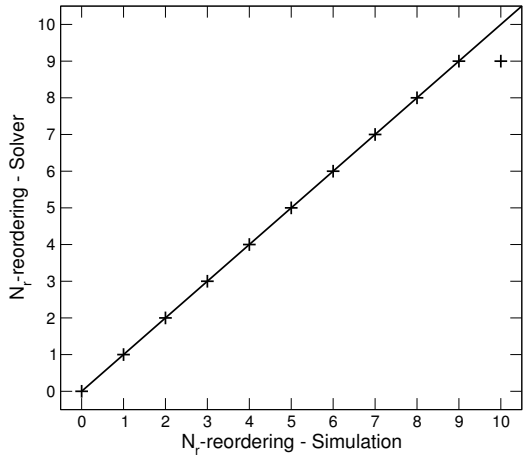
(a) Extent probability distribution



(b) Extent qq-plot



(c) N_r-reordering ccdf

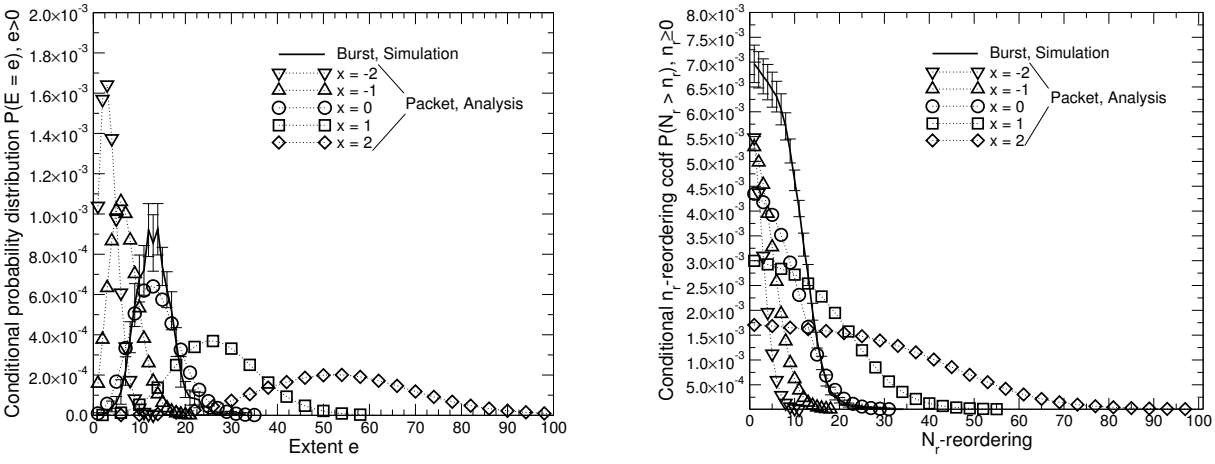


(d) N_r-reordering qq-plot

Figure 6.4: Burst reordering analysis Zagreb – Munich, $\alpha = 1.3$

problem. Besides burst reordering, it is more valuable to emulate the actual packet reordering if a network shows assembly mechanisms. Section 4.4.1 on page 90 presented already the fundamental mechanisms to obtain the packet reordering pattern form the original burst reordering pattern. The connection between both is the distribution on the number of packets per burst. This section applies these findings and derives the packet reordering patterns with respect to the burst reordering patterns. With the packet reordering pattern, the steps are equivalent compared to the burst reordering pattern.

This section undergoes the whole procedure for the packet reordering metrics: It starts with the burst reordering metrics obtained of the simulation. Together with a distribution on the number of packets per burst, this leads to packet reordering metrics. This last step serves as an input parameter for the solver. The remaining two steps correspond to the steps including only burst reordering patterns. The result of the solver in combination with the analytic reordering model should lead to the initial packet reordering metrics. The results from the solver and the goodness-of-fit of both distributions are subject of this section.



(a) Extent probability distribution (b) N_r-reordering ccdf

Figure 6.5: Theoretic packet reordering Paris – Rome, $\alpha = 1.35$

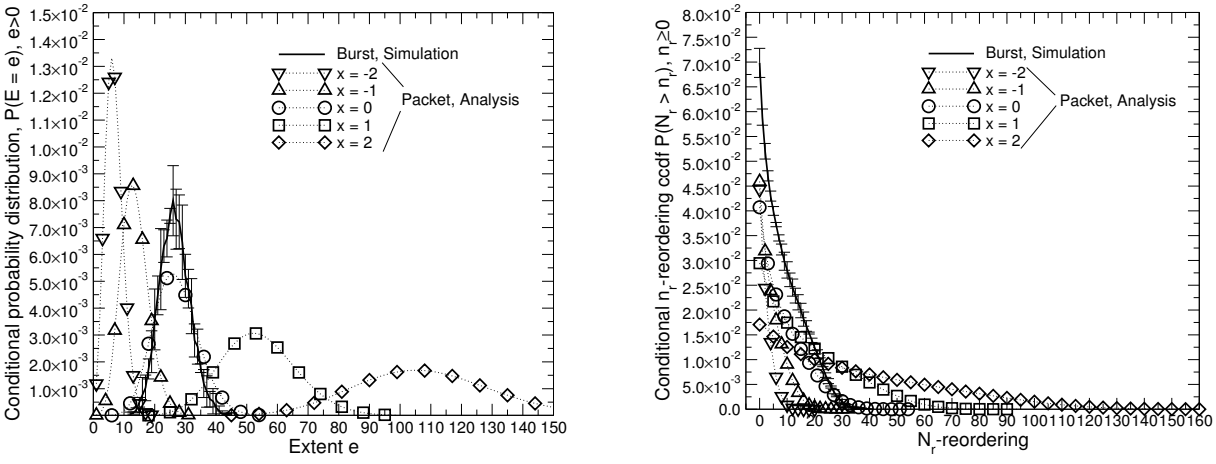
The next subsections firstly introduce a model for the distribution of packets per burst and secondly evaluate the solution of the solver for the packet reordering scenario.

6.3.3.1 Packet per Burst Distribution

This section studies packet reordering patterns with respect to the original burst reordering patterns. Therein the packet per burst distribution shows the major influence. In general, the packet per burst distribution varies, as the packet per burst distribution highly depends on the client packet characteristics, the overall load scenario and the burst assembly strategy. Consequently, it is not possible to identify a single distribution per application, protocol or load scenario. For a basic study on the effect of the packet per burst distribution on the reordering pattern, this thesis shows a simple theoretic distribution, which enables the basis for further studies on this topic.

In this study, the distribution of the number of packets per burst follows the Poisson distribution. It serves the requirements of a simple parameterization and allows basic findings. The Poisson distribution shows a single parameter, the expected number of packets per burst κ and per flow. For a first study, the expected number of packets per burst is varied with the following exponential series, i. e., $\kappa = 2^x, x \in \mathbb{N}$. In this thesis, $x = -2, -1, 0, 1, 2$, which leads to the following mean number of packets per burst $\kappa = 0.25, 0.5, 1, 2, 4$. These mean values for the number of packets per burst range in the order of the results found by Detti et al. in [29]. Therein, the left edge of the series denotes a low capacity flow while the right edge denotes a high capacity flow.

Figure 6.5 and Figure 6.6 depict the influence of the packet per burst distribution for two selected end-to-end connections of the considered network topology. Figure 6.5a) and b) consider the connection Paris–Rome at $\alpha = 1.35$, while Figure 6.6a) and b) consider the connection Hamburg–Amsterdam at $\alpha = 1.1$. The considered reordering metrics include the reordering extent in a) as well as the n_r-reordering metric in b). Each figure shows the original burst reordering metric obtained from simulation. Additionally, it shows the



(a) Extent probability distribution

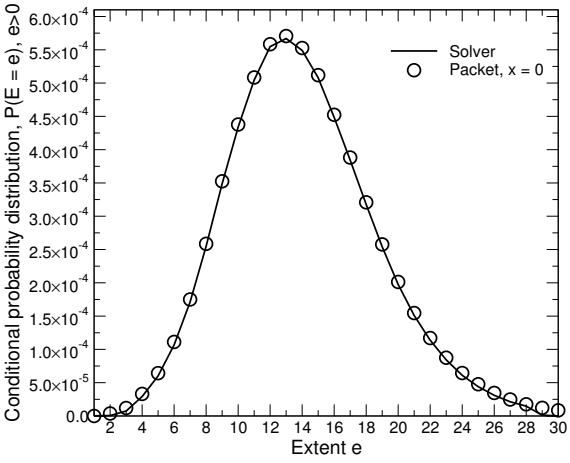
(b) N_r-reordering ccdf

Figure 6.6: Theoretic packet reordering Hamburg – Amsterdam, $\alpha = 1.1$

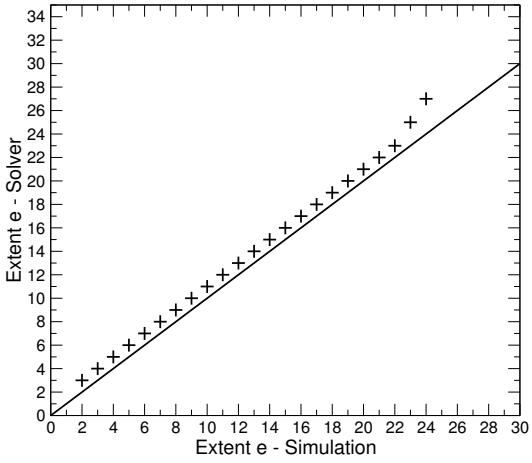
transformed reordering metrics for a different average number of packets per burst, i. e., different x. Note that the reordering metrics of the burst layer equals the scenario of *exactly* one packet per burst. The figure also shows the results for one packet per burst *on average*, i. e., $x = 0$. Both results are different, i. e., the curves do not match. The following observations on the graphs of the extent and the n_r-reordering metrics hold for both connections, i. e., Figure 6.5 and Figure 6.6. In this thesis, these two connections were selected, any other connection would not provide additional findings.

Figure 6.5a) and Figure 6.6a) depict the results of the extent probability distribution. The solid line represents the burst reordering extent, while the dotted lines show the results for different x, i. e., mean number of packets per burst. If the mean number of packets per burst increases $x > 0$, the mean extent also increases, i. e., the distribution moves to the right. Additionally, the distribution becomes broader and flatter compared to the original burst reordering extent distribution. The reason for this change is the increasing number of packets per burst. The packet extent value corresponds approximately to the product of the number of packets per burst and the burst extent value. If the number of packets per burst decreases below one packet per burst on average, i. e., $x \leq 0$, the distribution shifts to the left. Besides this, the distribution becomes thinner as large extent values become more unlikely. The reason for this shift is the small number of packets on average. If a burst arrives reordered and the number of packets within each burst is less than 1, large extent values become unlikely. If there are packets in the corresponding bursts, their number is quite small. The distributions at the left became thin and small. Although individual packet probabilities exceed the probabilities for the burst extent, the integral of these thinner distributions is smaller than the original distribution of the burst extent. The findings of Section 4.4.1 illustrate exactly this. They show that the burst reordering ratio reaches its maximum with respect to any packet per burst distribution.

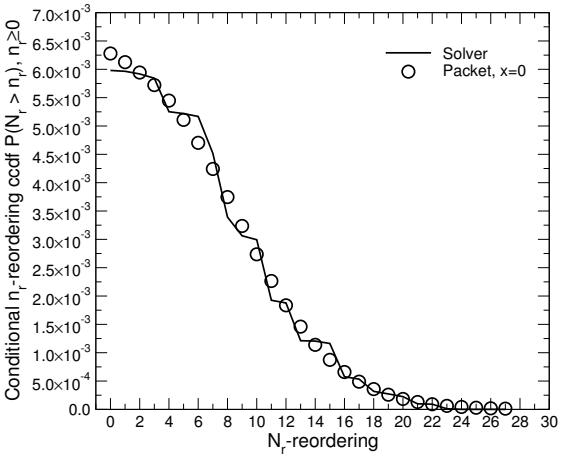
Figure 6.5b) and Figure 6.6b) depict the results for the ccdf of the n_r-reordering metric. Similar observations apply. An increasing average number of packets, i. e., $x > 0$, extents the ccdf towards higher n_r-reordering values. At the same time, the number of packets with $n_r > 0$ decrease as packets within one burst are always in-order. These packets



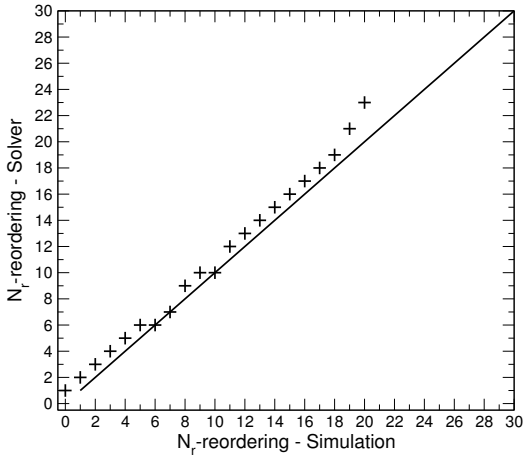
(a) Extent probability distribution



(b) Extent qq-plot



(c) N_r-reordering ccdf



(d) N_r-reordering qq-plot

Figure 6.7: Packet reordering analysis Munich – Brussels, $\alpha = 1.3$

receive a n_r-reordering value of $n_r = 0$. Additionally, the reordering ratio remains the same for $x > 1$ as shown in Section 5.2. For scenarios with a smaller average number of packets per burst, i. e., $x < 0$, the extension of the distribution is reduced as well as the ratio or reordered packets is decreased. The ccdf moves to the left and drops away. The reason for this is the decreasing probability of an out-of-sequence burst arrival as well as the decreasing probability of consecutive packets satisfying the strict order in the sequence number as introduced in Section 3.2.5. The relation between the average number of packets per burst and the n_r-reordering metrics totally agrees with the findings of Schlosser et al. in [27].

6.3.3.2 The Results of the Optimization Problem

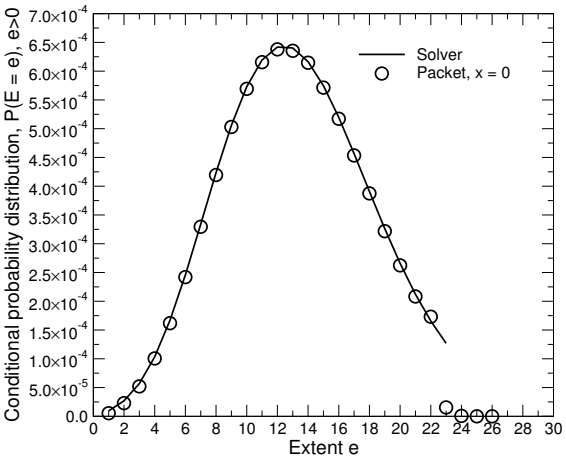
This section applies the solver of Section 5.3.1 on the transformed packet reordering metrics of the previous section to obtain the probability distribution of the disordering network. Hereby, the transformed reordering pattern considers the original burst reordering pattern and a suitable packet per burst distribution. For a quantitative evaluation,

this section again applies the methods of Section 6.1 including the visual interpretation and both hypothesis tests. For two selected sample connections, the following figures show the results of the solver and the original reordering pattern.

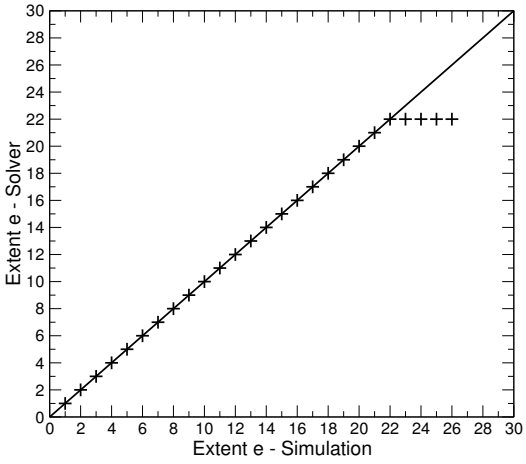
Figure 6.7 selects the connection between Munich and Brussels at a load level of $\alpha = 1.3$. Figure 6.7a) depicts the extent distribution for an average of one packet per burst, i. e., $x = 0$. Note that $x = 0$ does not equal the scenario if there is exactly one packet in every burst. While the latter scenario equals the burst reordering scenario, the former shows statistical variations from that. $x = 0$ was chosen as a trade-off for reasonable values of the distribution. The graph of the packet reordering does not contain any confidence intervals, as its values are derived from analytics (cf. previous section). The visual impression implies that the results from the solver together with the analytic model fit the transformed distribution from simulation. The qq-plot on the right side strengthens this impression, nearly all markers lie on the 45-degree line. Besides the qq-plot, also the KS-test as well as the χ^2-test recommend the hypothesis that both plots originate from the same distribution. The findings for the reordering extent distribution correspond to the findings obtained for the n_r-reordering metrics depicted in Figure 6.7c) and d). Again, the findings of the solver and the reference graph correspond. This includes the visual impression and the results of the qq-plot. Additionally, the hypothesis tests recommend an acceptance of the hypothesis that both distributions originate from the same parent distribution.

As a second example, Figure 6.8 depicts the results from the connection between Paris and Rome in a load scenario of $\alpha = 1.35$. Again, there is one packet per burst on average, i. e., $x = 0$. Concerning the reordering extent metric in Figure 6.8a), on a visual impression, both distributions match, although the solution of the solver does not span the whole range of the calculated distribution values, i. e., $e > 21$. In addition, the qq-plot confirms this impression and shows the missing values of the solvers for $e > 21$. The marks superpose on the 45-degree line or one position aside due to inaccuracy or discretization. Both hypotheses recommend an acceptance of the hypotheses that both distributions originate from the same parent distribution. In both cases, the number of samples provide enough confidence for a significant level of 95%. Additionally to the reordering extent metrics, the figure also depicts the results for the n_r-reordering metric. The visual impression as well as the qq-plot are even more promising, both curves perfectly fit. Also the KS-test and the χ^2-test also recommend an acceptance of the hypotheses that both distributions originate from the same parent distribution.

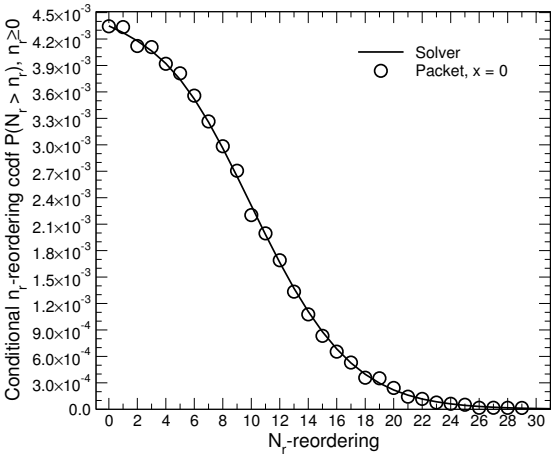
Summarizing, with the solution of the optimization problem, it is possible to re-engineer the disordering network for transformed reordering metrics, i. e., packet reordering metrics obtained from burst reordering metrics. On two selected samples including both metrics (reordering extent, n_r-reordering), the process to evaluate the solution of the solver was demonstrated. Any other packet per burst distribution leads to similar accurate results. As they provide no new findings, they were skipped. Re-engineering the transformed reordering packet metrics becomes very difficult for large absolute values of the extent and the n_r-reordering metric, cf. Figure 6.5a). Large absolute values result in large vectors, which increase the complexity of the solver. The next section addresses this issue and evaluates the complexity of the optimization problem.



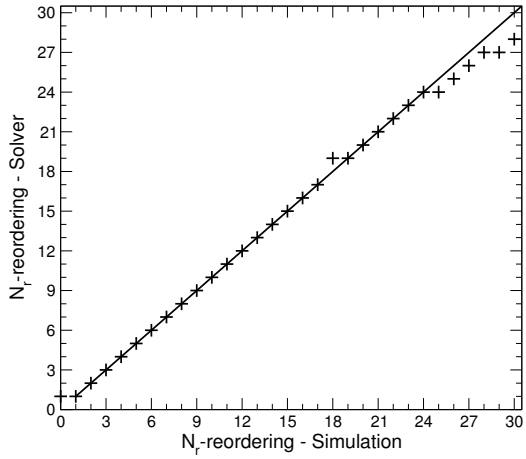
(a) Extent probability distribution



(b) Extent qq-plot



(c) N_r-reordering ccdf



(d) N_r-reordering qq-plot

Figure 6.8: Packet reordering analysis Paris – Rome, $\alpha = 1.35$

6.4 Complexity Analysis

The previous sections evaluated the solutions of the optimization problem of Section 5.3.1 with respect to their goodness-of-fit. In order to obtain the reordering metrics analytically, the evaluation step itself as well as the required effort of the solver were neglected. This section now quantifies the effort of the solver step itself and draws conclusions on the scalability of the optimization problem. Besides this, the complexity to evaluate the reordering metrics analytically is subject to this section.

For a comprehensive evaluation, this section applies Landau's O-notation already introduced in Section 3.1.5 on page 43, e. g., $O(n^2)$ denotes squared complexity of a function for increasing n. Thereby, n may be the number of function calls or items to process.

Solving the optimization problem includes in principle three steps: 1. Choice of an appropriate starting point and 2. evaluation of the reordering metrics at a certain point, i. e., certain network delay distribution, 3. selection of the next point, which minimizes the optimization problem. Steps 2. and 3. continue until a suitable point has been found

or the maximum number of iterations has been reached. Both steps together define the effort of the solver. This section addresses both steps and evaluates their complexity with respect to the computation time.

This section firstly studies the complexity to evaluate the functions of the reordering metrics of Section 3.2.5. The section secondly studies the complexity of the solver algorithm and the solution process in total. Both studies apply Landau's O-notation.

6.4.1 Model Complexity

This section evaluates the complexity to obtain the reordering metrics of reordering ratio, reordering extent and n_r-reordering metric for deterministic traffic. As the corresponding functions of Section 4 are non-linear and include convolution steps, the evaluation requires some computational time. This section studies the time to evaluate these functions and provides an estimate on the scalability of this evaluation step.

The evaluation of the reordering ratio is rather simple, it allows the estimation of the complexity by checking Eq. (4.4). The function shows a complexity of $O(m^3)$, if m is the number of abstract links. The first sum of Eq. (4.4) depends on m as well as the product and the second sum. If the evaluation time matters, an efficient implementation may reduce the effort to $O(m^2)$. This is possible as the second sum represents a sum of consecutive values, where the sum of two neighbouring k only differs by one term. This reduces the complexity of the second sum to $O(1)$.

For both other equations, i. e., Eq. (4.13) for the reordering extent and Eq. (4.19) for the n_r-reordering metric, it is not possible to give the complexity by analysing directly the equations. The functions for the reordering extent Eq. (4.13) and the equation for the n_r-reordering in Eq. (4.19) are far more complex. Therefore, the evaluation time of the functions were measured and fitted by a polynomial to derive the dominating complexity. For a concise evaluation, Figure 6.9 depicts the evaluation time of both metrics with respect to the number of abstract links m. The values were obtained from a reasonable up-to-date personal computer (Dual-Core AMD Opteron Processor 2220 with 2.8 GHz) to get an impression of the complexity. The star and the cross mark the corresponding measured values, while the solid line indicates a non-linear polynomial curve fitting. For both metrics, the fitting procedure determined a polynomial function with degree 4.3, i. e., $O(m^{4.3})$. The time to evaluate the functions for the reordering extent as well as the n_r-reordering metric, increases in the order of $m^{4.3}$. Both functions differ by their coefficient but the overall complexity remains in the same order of magnitude.

Summarizing, as m is the dimension of the probability vector of the disordering network, calculating the reordering extent and the n_r-reordering metric for large m produces a non-negligible computational effort.

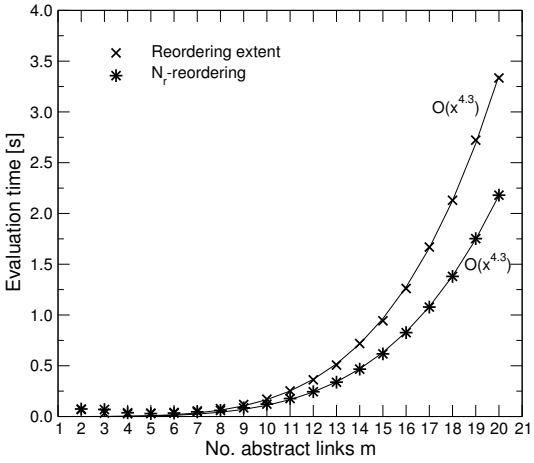


Figure 6.9: Evaluation time

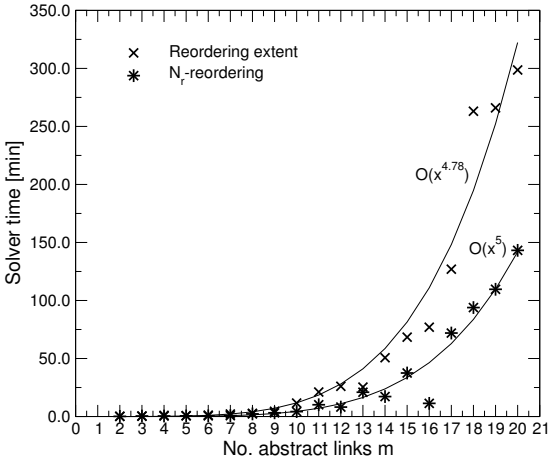


Figure 6.10: Solution time

6.4.2 Algorithm Complexity

As introduced in Section 5.3.1, the solver implements the active set algorithm of Powell et al. described in [112–115]. As literature provides a detailed analysis on the complexity of this solver, this section restricts itself to the findings of the experienced solving time of this thesis. A measure for the complexity of solver algorithms is the number of iterations a solver requires to obtain an optimum solution. The number of necessary iterations serves as a measure for comparison to other solver algorithms with respect to the same reference problem. The overall time a solver requires bases on two factors, the number of iterations as well as the time to evaluate the function describing the problem. The next paragraphs elaborate on this topic in detail.

Literature evaluates the applied solver with respect to these reference problems, which include at maximum 15 variables (corresponds to the dimension of $\vec{x}$) and at maximum 20 constraints. Literature also classifies the constraints of being linear and non-linear, which additionally adds complexity. For these reference problems of the literature, the applied solver requires at maximum 20 iterations. In contrast to this, the selected samples of this thesis shows at maximum 35 variables (maximum dimension of $\vec{x}$) with five non-linear constraints (constraints on the probability vector and its shape). The solutions of this thesis required about 150 to 250 iterations of the solver with the according number of evaluations of the optimization function. As the calculation of the reordering metrics requires the largest effort, the overall evaluation time depends on it. Furthermore, the solver has an upper limit of 250 iterations until termination to guarantee any result within reasonable time. As a result, the solutions of the selected samples in this thesis need about one day on a reasonable up-to-date personal computer (Dual-Core AMD Opteron Processor 2220 with 2.8 GHz).

For a quantitative analysis, the solver faced the reordering metrics from a linear network delay distribution (cf. Section 4.3.3) with $p = 0.1$ and different m. The time to determine the original network delay distribution was measured. Depending on the number of abstract links m, Figure 6.10 depicts the time required to obtain the solution. As already introduced, the number of abstract links m represents the scalability parameter of this

step. The figure depicts the measured samples with stars and crosses, while the solid lines represent the non-linear curve fitting. The curves were fitted on a polynomial with degree of 4.78 and 5, respectively. Consequently, the solver's complexity in O-notation is estimated to be about $O(m^5)$, i. e., time to obtain a solution from the solver scales with power of 5, with respect to the number of abstract links m.

The graph also shows some points, which do not lie directly on the non-linear curve fit. These peaks occur because of the search algorithm of the solver. In some cases, the solver follows the correct branch of the solution space immediately. Then, the solver obtains a solution in very short time. In other cases, a solution requires further steps. Thus, the actual number of iterations varies with the problem, which leads to some points with a smaller and a larger number of iterations with respect to the estimated polynomial.

Summarizing, the outcome of the solver is the network delay distribution, which is necessary for the parameterization of the network emulation of Section 5.4. The overall complexity to obtain this parameterization scales with $O(m^5)$ if m is the number of abstract links. This drawback is alleviated by the fact that the solver process is only required once for any investigation on transport and application layer protocols in a network emulation scenario. Different protocols and applications may share the same configuration of network emulations, i. e., the result serves for different studies. Besides this, the obtained solution may serve as a starting point to vary the probability distribution of the disordering network directly, which does not require any additional effort.

7 Conclusions and Outlook

Current trends of the future Internet initiative consider to deploy multi-path routing as well as packet assembly in future high-speed networks motivated this work. Both concepts break with the existing paradigms of shortest path routing and individual packet processing. Consequently, both trends have an impact on the packet order, which may influence the protocol performance. Metrics of the protocol performance are throughput and delay, while buffer sizes are relevant for the node architecture to re-sequence packets. Consequently, these trends need a careful analysis including testbeds for experiments. This thesis presented and analysed an analytic reordering model, which serves two major applications. It enables worst-case estimations on packet reordering in present and future networks and allows parameterizing a network emulation to perform protocol engineering on predefined reordering patterns. Besides the analytic model, this thesis presented an equivalent simulation model as well as a coherent testbed for protocol engineering. The findings of the analytic model, the simulation model as well the findings of the network emulation fit together and show consistent results.

For a substantial background, Chapter 2 introduces and classifies various concepts of multi-path routing as well as packet assembly strategies. The classification criterion for both schemes is the network layer. The transport, networking and MAC layer may implement multi-path routing as well as packet assembly depending on the network technology. Reasons for multi-path routing are the increased network performance by equal load distribution. Examples for multi-path implementations are optical burst switching networks on layer 2, IP on layer 3 as well as TCP extensions on layer 4. Additionally, mobile ad hoc networks also show multi-path routing properties to increase network availability. The reason for packet assembly is primarily the reduction of the header-processing rate. Examples for network technologies showing packet assembly only are some wireless network technologies, e. g., WiMax and WLAN as well as wired high-speed networks, e. g., optical burst switching and frame switching technologies.

Changes in the packet order require special metrics to classify and differentiate reordering patterns. Literature provided a large set of different inconsistent metrics. Chapter 3 firstly introduces a set of requirements of these metrics. The most important parameters are robustness, easy implementation and metric complexity. Secondly, it introduces and classifies the metrics proposed in the literature according to these requirements. A comprehensive discussion of the proposed metrics leads to the IETF metrics most suitable for the following analysis. The IETF metrics represent a standard, are easy to implement and show coherence to the metrics from other standardization bodies like ITU-T. Among others, they include the definition of in-order and out-of-order arrivals, the reordering ra-

tio, the extent metric and a metric to estimate the TCP performance. The remainder of this thesis, especially the reordering model implements these metrics to evaluate a certain reordering event.

The analysis on the packet order requires a model to reflect changes in the packet order. Literature considered these phenomena of packet-based networks with queuing theoretical approaches to determine the required buffer size and time to re-sequence reordered packets. Literature proposes several models to determine the specific reordering. For classification, Chapter 4 introduces the general ideas of these approaches in contrast to the work presented in this thesis. Any reordering model shows in principle two major parameters, the delay a packet receives and the inter-arrival time of the traffic model. Chapter 4 introduces the analytic reordering model of this thesis, which showed a discrete delay distribution. The discrete delay distribution may originate from an approximation of a measured end-to-end delay. Consequently, the model represents a set of parallel $\cdot$/D/1 models with different delays D. For an initial study, the discrete network delay and the traffic characteristic share a common basic delay unit. For this reordering model, Chapter 4 presents the formal analysis for deterministic as well as Poisson traffic models. Additionally to the findings of single layer reordering, the chapter also presents the formal reordering metrics for hierarchical packet reordering. In this case, the transporting bursts arrive reordered, which may also reorder carried packets. For selected scenarios, this chapter verified the findings of the analytic model with equivalent simulations.

Chapter 5 applied the analytic reordering model in two different scenarios. The first application of the analytic reordering model with deterministic inter-arrival time enables worst-case estimations of the reordering ratio with respect to any other traffic model with the same mean traffic rate. This chapter provides the formal proof for this worst-case estimation and illustrates these findings for selected scenarios. It further shows that this worst-case property still holds if the traffic mean increases, while the disordering network remains in its original settings. For lower mean traffic rates, the worst-case estimation is not valid any more. In this scenario, the disordering network degenerates as some of the alternative $\cdot$/D/1 models do not impact the reordering pattern anymore. However, this scenario decreases the reordering ratio in any case, which limits the negative impact on protocols and networks. Consequently, the scenario, where the inter-arrival time and the basic delay unit correlate, represents a border scenario.

The second application of the model enables studies of real protocol implementations in testbeds. On basis of the reordering model, this thesis also implemented a network emulation module, which changes the packet order according to a predefined statistical reordering pattern. This testbed emulates packet reordering of a certain connection. Thereby, the reordering patterns from this connection originate from simulation, analytic studies or measurements of investigated network technologies. The module bases on the Linux kernel using the network emulation module of the Linux Foundation. It realizes the analytic reordering model for deterministic traffic, delays packets with respect to subsequent packet arrivals, and thus produces destined reordering patterns. For the reordering module, this chapter included a description of the module state machine as well as a discussion on the representation of random numbers in the Linux kernel. The parameterization of this module requires the network delay distribution of an observed reordering

pattern, which is in general not obvious. The predefined reordering pattern together with the reordering model lead to a non-linear constraint optimization problem. The applied Matlab solver obtains a network delay distribution. This probability distribution should lead to the same predefined reordering pattern. The solver applies the model for deterministic inter-arrival times as only the arrival order than the traffic model is relevant for this scenario. A verification of the module and comprehensive comparison of the findings of the testbed, simulation and the analytic model conclude this chapter.

Chapter 6 evaluated the reordering model and its application in network emulation scenarios. It evaluated the quality of the optimization problem, which re-engineers the network delay distribution from a given reordering pattern. The reordering pattern came from simulations of a technology showing multi-path routing and packet assembly for hierarchical packet reordering, i.e., optical burst switching scenarios. The simulation results on the burst and packet reordering metrics served as input parameters for the solver. The solution of the optimization problem represents the corresponding network delay distribution, which serves as a parameter for the network emulation module of Chapter 5. As for the network emulation only the order is relevant, the solver applies the deterministic traffic model. The parameterized module creates the same reordering pattern as from the simulations, independent of the arriving traffic characteristic.

The comprehensive evaluation includes the comparison of the initial reordering pattern from simulations and the result from the solver. As both results are distributions, this chapter firstly introduces statistical evaluation methods for a quantitative evaluation. These methods include the qq-test, the Kolmogorov-Smirnov and χ^2 hypothesis tests. Secondly, it presents the optical burst switching network simulator and its parameters. The simulator provides the measurements of the reordering patterns. The investigated scenario includes a common European reference network, as well as different load situations. Finally, this chapter applies the evaluation methods and compares the simulated burst and packet-reordering pattern with the findings of the solver. The visual as well as the hypothesis test advise that the result of the solver reproduce the estimated reordering pattern very well.

An additional subject for evaluation is the complexity of the solver. The complexity of the solver in terms of computation time considers two factors. One is the number of iterations the solver requires for a solution and the other is the time to evaluate the functions of the analytic model. As it was not feasible to determine the complexity of these functions analytically, measurements provide the basis for the analysis of complexity. A non-linear curve-fitting algorithm fits the measurement results of the function evaluations to a curve of a polynomial of degree 4.3 with respect to the number of alternative paths of the reordering model. The solving process requires several steps to evaluate the same function with different parameterizations. The time to obtain a solution of the optimization problem was fitted with a polynomial of degree 5. The findings show that for a large number of abstract links, the solver requires improvements to speed up the solution process.

This thesis paves the way for further studies in three different directions. The first direction includes the improvement of the solver step to obtain results in shorter time. The second direction opens performance studies of protocols with respect to certain reordering patterns. The third direction enables improvements of the network emulation modules.

The improvement of the solver includes an optimization of the solver parameters as well as concise studies on the starting point to reduce the number of steps. Additionally, the precision of the solution is subject to discussions on the improvement of the solver step. Looser requirements on the correctness would speed up the solving process but may still reproduce to a large degree the characteristic reordering pattern. Besides optimizing the solver step, any optimization should also reduce the time to compute the reordering functions, too. The implementation of these functions leaves degrees of freedom to speed up calculation time, e. g., implementation specific improvements. Regardless of the computational effort of the solver, the proposed procedure only requires solving the optimization problem once for any application or protocol study in a network emulation scenario, which face the same reordering conditions.

This thesis provided the basis to study protocol performance with respect to any reordering pattern. The actual investigation of selected protocols is subject to further studies. Selected protocols to study may include applications, which base on either TCP or UDP transport protocols. Therein, the TCP algorithms, which resolve reordered packets, are of special interest. TCP implementations may be investigated and improved when facing certain reordering patterns. These studies may have influence on the dup-ack threshold or the receiving buffer size. Both may be adapted according to the experienced reordering pattern. In addition, applications relying on UDP also have to provide similar mechanisms than TCP. UDP based applications, e. g., voice and video, may experience degradation in the quality of service because of waiting too long for reordered packets. Studies on this topic may include adaptations of the playout buffer or re-sequencing algorithms. These future studies may include both, new protocols and applications facing reordering and already deployed applications and protocols with respect to new network conditions, i. e., multi-path routing and packet assembly. If network paradigms change towards multi-path routing and packet assembly, these studies become necessary for early worst-case estimations on the amount of reordering in these networks as well as investigations of already implemented protocols.

Regarding the reordering module, it delays packets with respect to subsequent arriving packets and therefore changes the original traffic characteristic. For a compensation of the changes in the traffic characteristic, an additional module may re-create the original traffic characteristic after changing the order of the packets. The additional module maintains the new reordered packet order but provides a playout buffer, which forwards packet according to the original traffic characteristic. To realize this, the reordering module may record the original packet inter-arrival time.
Another issue is the delay by packet implementation of the reordering module. Applying the reordering module together with closed-loop protocols, e. g., TCP, may lead to retransmission timeouts because of missing packet delayed in the reordering module. An improvement of the reordering module implements a timer in the reordering module to avoid infinite waiting. However, this timer also slightly changes the reordering pattern experienced at the destination. Further studies may quantify the changes in the reordering pattern due to an additional timer.

A Mathematical Proofs

A.1 Convexity of Function $g(\vec{x})$

Section 5.2.2.3 left open the proof for the convexity of function $g(\vec{x}) = \prod_{k=1}^{i} q_k^{x_k-1}$ where $i, x_k \in \mathbb{N}_0$ and $0 < q_k < 1$. The defintion of a convex function f is according to [131]: $f : \mathbb{N}^n \to \mathbb{N}$ is convex if $\vec{x} \in \mathbb{A}$ forms a convex set and if $f(\theta\vec{x}+(1-\theta)\vec{y}) \leq \theta f(\vec{x})+(1-\theta)f(\vec{y})$ for all $\vec{x} \in \mathbb{A}$ and $0 \leq \theta \leq 1$.

The first requirement is that $\mathbb{A}$ is a convex set. $\mathbb{A}$ is a convex set if $(1-t)\vec{x} + t\vec{y}$ is also in the set $\mathbb{A}$ for $0 \leq t \leq 1$ and $\vec{x}, \vec{y} \in \mathbb{A}$. In this case, the set $\mathbb{A}$ represents a set of points as $x_k \in \mathbb{N}_0$. But any of these points, which $(1-t)\vec{x} + t\vec{y}$ covers is included in the set. Consequently, $\mathbb{A}$ represents a convex set of points.

The second requirement is $f(\theta\vec{x} + (1-\theta)\vec{y}) \leq \theta f(\vec{x}) + (1-\theta)f(\vec{y})$. With the substitution of $f \to g$, this leads to:

$$\begin{aligned}
g(\theta\vec{x} + (1-\theta)\vec{y}) &\leq \theta g(\vec{x}) + (1-\theta)g(\vec{y}) \\
\prod_{k=1}^{i} {q_k}^{\theta x_k - 1 + (1-\theta)y_k} &\leq \theta \prod_{k=1}^{i} {q_k}^{x_k-1} + (1-\theta)\prod_{k=1}^{i} {q_k}^{y_k-1} \\
\prod_{k=1}^{i} {q_k}^{\theta x_k} \prod_{k=1}^{i} {q_k}^{(1-\theta)y_k} &\leq \theta \prod_{k=1}^{i} {q_k}^{x_k} + (1-\theta)\prod_{k=1}^{i} {q_k}^{y_k} \\
\left(\underbrace{\prod_{k=1}^{i} {q_k}^{x_k}}_{a}\right)^{\theta} \left(\underbrace{\prod_{k=1}^{i} {q_k}^{y_k}}_{b}\right)^{1-\theta} &\leq \theta \underbrace{\prod_{k=1}^{i} {q_k}^{x_k}}_{a} + (1-\theta)\underbrace{\prod_{k=1}^{i} {q_k}^{y_k}}_{b}, \quad 0 < a, b \leq 1 \\
a^{\theta} b^{1-\theta} &\leq \theta a + (1-\theta)b
\end{aligned} \tag{A.1}$$

Deviding both sides by b and defining $a/b = x$ results in:

$$\begin{aligned}
x^{\theta} &\leq \theta x + (1-\theta) \\
x^{\theta} &\leq \theta(x-1) + 1
\end{aligned} \tag{A.2}$$

The left side of Eq. (A.2) represents an exponential function. The right side of Eq. (A.2) represents a linear function. At the borders of θ, i. e., $\theta = 0$ and $\theta = 1$, both sides are equal. The following holds for exponential functions for $0 < \theta \leq 1$. If $x > 1$ it increases

more slowley than a linear function of Eq. (A.2). If $0 < x < 1$ it decays faster than a linear function of Eq. (A.2). If $x = 1$ both sides of Eq. (A.2) are equal. In all cases, the left side is always equal or smaller than the right side. Summarizing, g fulfils the requirements of a convex function.

Bibliography

[1] R. Inkret, A. Kuchar, B. Mikac, C.M. Gauger, and M. Köhn. Advanced Infrastructures for photonic Networks - Extended Final Report of COST Action 266. Technical report, Faculty of Electrical Engineering and Computing, University of Zagreb, 2003.

[2] World Internet Users and Population Stats, August 2009. http://www.internetworldstats.com.

[3] University of Minnesota. Minnesota internet traffic studies (mints). World Wide Web electronic publication, 2009. http://www.dtc.umn.edu/mints/home.php.

[4] Sharad Jaiswal, Gianluca Iannaccone, Christophe Diot, Jim Kurose, and Don Towsley. Measurement and classification of out-of-sequence packets in a tier-1 IP backbone. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 113–114, New York, NY, USA, 2002. ACM. doi:10.1145/637201.637218.

[5] C.M. Arthur, D. Girma, D. Harle, and A. Lehane. The effects of packet reordering in a wireless multimedia environment. *Wireless Communication Systems, 2004. 1st International Symposium on*, pages 453–457, Sept. 2004. doi:10.1109/ISWCS.2004.1407288.

[6] Hiroyuki Koga, Takeshi Ikenaga, Yoshiaki Hori, and Yuji Oie. Out-of-Sequence in Packet Arrivals due to Layer 2 ARQ and Its Impact on TCP Performance in W-CDMA Networks. In *SAINT '03: Proceedings of the 2003 Symposium on Applications and the Internet*, page 398, Washington, DC, USA, 2003. IEEE Computer Society.

[7] Vern Paxson. End-to-end Internet packet dynamics. *IEEE/ACM Transactions on Networking*, 7(3):277–292, 1999. doi:10.1109/90.779192.

[8] Ka-Cheong Leung, Victor O.K. Li, and Daiqin Yang. An Overview of Packet Reordering in Transmission Control Protocol (TCP): Problems, Solutions, and Challenges. *IEEE Transactions on Parallel and Distributed Systems*, 18(4):522–535, 2007.

[9] Rich Seifert. *The All-New Switch Book*. John Wiley & Sons, 2008.

[10] S. Govind and R. Govindarajan and Joy Kuri. Packet Reordering in Network Processors. In *International Parallel and Distributed Processing Symposium (IPDPS-07)*, 2007.

[11] A. Bare, A. Jayasumana, and N. Piratla. On Growth of Parallelism within Routers and Its Impact on Packet Reordering. In *In 15th IEEE Workshop on Local and Metropolitan Area Networks*. Princeton, NJ, 2007.

[12] Cisco. How Does Load Balancing Work? Technical Report 5212, Cisco, August 2009.

[13] Nischal M. Piratla, Anura P. Jayasumana, Abhijit A. Bare, and Tarun Banka. Reorder buffer-occupancy density and its application for measurement and evaluation of packet reordering. *Computer Communications*, 30(9):1980–1993, 2007. doi:10.1016/j.comcom.2007.03.001.

[14] S. Kandula, D. Katabi, S. Sinha, and A. Berger. Dynamic load balancing without packet reordering. *Communication Review*, 37(2), April 2007.

[15] Vern E. Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, EECS Department, University of California, Berkeley, Jun 1997. http://www.eecs.berkeley.edu/Pubs/TechRpts/1997/5498.html.

[16] Ming Zhang, B. Karp, S. Floyd, and L. Peterson. RR-TCP: a reordering-robust TCP with DSACK. *Network Protocols, 2003. Proceedings. 11th IEEE International Conference on*, pages 95–106, November 2003. doi:10.1109/ICNP.2003.1249760.

[17] Arjuna Sathiaseelan and Tomasz Radzik. RD-TCP: Reordering detecting TCP. In *HSNMC 2003 : high-speed networks and miltimedia communications*, 2003.

[18] S. Bohacek, J.P. Hespanha, Junsoo Lee, C. Lim, and K. Obraczka. A new TCP for persistent packet reordering. *Networking, IEEE/ACM Transactions on*, 14(2):369–382, April 2006. doi:10.1109/TNET.2006.873366.

[19] Andrea Detti and Marco Listanti. Impact of segments aggregation on TCP Reno flows in optical burst switching networks. In *Proc. IEEE INFOCOM*, 2002.

[20] Xiang Yu, Jikai Li, Xiaojun Cao, Yang Chen, and Chunming Qiao. Traffic statistics and performance evaluation in optical burst switched networks. *IEEE/OSA Journal of Lightwave Technology*, 22(12):2722–2738, 2004.

[21] S. Gowda, R.K. Shenai, K.M. Sivalingam, and H.C. Cankaya. Performance evaluation of TCP over optical burst-switched (OBS) WDM networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, volume 2, pages 1433–1437 vol.2, 2003.

[22] Franco Callegati, Walter Cerroni, and Carla Raffaelli. Impact of Optical Packet Loss and Reordering on TCP Performance. In *IEEE Global Telecommunications Conference (GLOBECOM '06)*, pages 1–5, November 2006. doi:10.1109/GLOCOM.2006.411.

[23] Jingyi He and S. H. Gary Chan. TCP and UDP performance for Internet over optical packet-switched networks. *Computer Networks*, 45(4):505–521, July 2004.

[24] J.P. Gelpke, M. Schlosser, E. Patzak, and H. Buchta. Assessment of tcp performance in obs networks with load dependent contention. In *Proceedings of the 9th International Conference on Transparent Optical Networks (ICTON)*, 2007.

[25] F. Callegati, G. Muretto, C. Raffaelli, P. Zaffoni, and W. Cerroni. A framework for performance evaluation of OPS congestion resolution. In *Proceedings of the IFIP Working Conference on Optical Network Design and Modelling (ONDM)*, pages 242–249, 2005.

[26] Jordi Perelló, Sebastian Gunreben, and Salvatore Spadaro. A Quantitative Evaluation of Reordering in OBS Networks and its Impact on TCP Performance. In *Proceedings of the IFIP Working Conference on Optical Network Design and Modelling (ONDM)*, March 2008.

[27] M. Schlosser, E. Patzak, and P. Gelpke. Impact of deflection routing on TCP performance in optical burst switching networks. In *Proceedings of the 7th International Conference on Transparent Optical Networks (ICTON)*, volume 1, pages 220–223, Barcelona, June 2005.

[28] J. Padhye, V. Firoiu, D. Towsley, and J. Krusoe. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 303–314, 1998.

[29] Andrea Detti and Marco Listanti. Amplification effects of the send rate of TCP connection through an optical burst switching network. *Optical Switching and Networking*, 2(1):49–69, May 2005.

[30] P. Du and S. Abe. TCP performance analysis of optical burst switching networks with a burst acknowledgment mechanism. In *Communications, 2004 and the 5th International Symposium on Multi-Dimensional Mobile Communications Proceedings. The 2004 Joint Conference of the 10th Asia-Pacific Conference on*, volume 2, pages 621–625 vol.2, Aug.-1 Sept. 2004.

[31] Jon C. R. Bennett, Craig Partridge, and Nicholas Shectman. Packet reordering is not pathological network behavior. *IEEE/ACM Transactions on Networking*, 7(6):789–798, 1999. doi:10.1109/90.811445.

[32] F. Kelly, S. Zachary, and I. Zeidins, editors. *Stochastic Networks: Theory and Applications*, chapter Notes on effective bandwidths, pages 141–168. Number 4. Oxford University Press, 1996.

[33] Damon J. Wischik. The output of a switch, or, effective bandwidths for networks. *Queueing Syst. Theory Appl.*, 32(4):383–396, 1999. doi:10.1023/A:1019107708752.

[34] D. Abendroth and U. Killat. Effective bandwidth shaping: a framework for resource dimensioning–extended version. *Computer Communications*, 28(15):1703–1710, September 2005. doi:10.1016/j.comcom.2004.12.014.

[35] Stefan Bodamer and Joachim Charzinski. Evaluation of effective bandwidth schemes for self-similar traffic. In *Proceedings of the 13th ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management*, pages 21.1–21.10, Monterey, CA, September 2000.

[36] Martin Köhn, Stefan Bodamer, Christoph M. Gauger, Sebastian Gunreben, Guoqiang Hu, and Detlef Sass. Comparison of IP/WDM transport network architectures for dynamic data traffic. In *Proceedings of the 11th European Conference on Networks and Optical Communications (NOC)*, Berlin, 2006.

[37] Robert Briscoe. *Re-feedback: Freedom with Accountability for Causing Congestion in a Connectionless Internetwork.* PhD thesis, UC London, 2009. http://www.cs.ucl.ac.uk/staff/B.Briscoe/pubs.html#refb-dis.

[38] C. Qiao and M. Yoo. Optical burst switching (OBS)—A New Paradigm for an Optical Internet. *Journal of High Speed Networks*, 8(1):69–84, January 1999.

[39] P. Gambini, M. Renaud, C. Guillemot, F. Callegati, I. Andonovic, B. Bostica, D. Chiaroni, G. Corazza, S.L. Danielsen, P. Gravey, P.B. Hansen, M. Henry, C. Janz, A. Kloch, R. Krahenbuhl, C. Raffaelli, M. Schilling, A. Talneau, and L. Zucchelli. Transparent optical packet switching: network architecture and demonstrators in the KEOPS project. *IEEE Journal on Selected Areas in Communications*, 16(7):1245–1259, 1998. doi:10.1109/49.725193.

[40] J. Sommer, S. Gunreben, A. Mifdaoui, F. Feller, M. Köhn, D. Sass, and J. Scharf. Ethernet - A Survey on its Fields of Application. *to appear in the IEEE Communications Surveys and Tutorials*, 2009.

[41] Miikka Poikselka, Aki Niemi, Hisham Khartabil, and Georg Mayer. *The IMS: IP Multimedia Concepts and Services.* Wiley, 2nd edition, March 2006.

[42] Alan B. Johnston. *SIP : understanding the Session Initiation Protocol.* Artech House, Boston, MA, 2nd edition, 2003.

[43] Wolfram Lautenschläger, Arthur Mutter, and Sebastian Gunreben. Frame aggregation in packet core networks – overview and experimental results. In *Proceedings of the 10. ITG Symposium on Photonic Networks*, Leipzig, May 2009.

[44] S.S. Gorshe and T. Wilson. Transparent generic framing procedure (gfp): a protocol for efficient transport of block-coded data through sonet/sdh networks. *Communications Magazine, IEEE*, 40(5):88–95, May 2002. doi:10.1109/35.1000218.

[45] Arthur Mutter, Martin Köhn, and Matthias Sund. A generic 10 Gbps assembly edge node and testbed for frame switching networks. In *Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom2009)*, page accepted for publication, 2009.

[46] M. de Vega Rodrigo and J. Goetz. An analytical study of optical burst switching aggregation strategies. In *Proceedings of the Third International Workshop on Optical Burst Switching (WOBS)*, San Jose, October 2004.

[47] H. Tounsi, L. Toutain, and F. Kamoun. Small packets aggregation in an ip domain. *Computers and Communications, 2001. Proceedings. Sixth IEEE Symposium on*, pages 708–713, 2001. doi:10.1109/ISCC.2001.935453.

[48] Kostas Pentikousis, Esa Piri, Jarno Pinola, Frerk Fitzek, Tuomas Nissilä, and Ilkka Harjula. Empirical evaluation of voip aggregation over a fixed wimax testbed. In *TridentCom '08: Proceedings of the 4th International Conference on Testbeds and research infrastructures for the development of networks & communities*, pages 1–10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[49] Toshikatsu Kanda and Kazunori Shimamura. Application of packet assembly technology to digital video and voip. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 392–395, New York, NY, USA, 2004. ACM. doi:10.1145/1027527.1027620.

[50] Jun Wang, Hong Wang, and Aidong Xu. A Packet-Aggregating MAC Protocol Based on Ultra Wide Band for Fieldbus Control System. *Control and Automation, 2007. ICCA 2007. IEEE International Conference on*, pages 3213–3216, June 2007. doi:10.1109/ICCA.2007.4376955.

[51] Qi Zhang, V.B. Iversen, and F.H.P. Fitzek. Throughput and Delay Performance Analysis of Packet Aggregation Scheme for PRMA. *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*, pages 1380–1384, April 2008. doi:10.1109/WCNC.2008.248.

[52] Z. Tao, K.H. Teo, and J. Zhang. Aggregation and concatenation in IEEE 802. 16j mobile multihop relay (MMR) networks. In *Mobile WiMAX Symposium, 2007. IEEE*, pages 85–90, March 2007. doi:10.1109/WIMAX.2007.348703.

[53] M.C. Castro, P. Dely, J. Karlsson, and A. Kassler. Capacity Increase for Voice over IP Traffic through Packet Aggregation in Wireless Multihop Mesh Networks. *Future generation communication and networking (FGCN 2007)*, 2:350–355, Dec. 2007. doi:10.1109/FGCN.2007.81.

[54] Helen C. Leligou, Gert Eilenberger, Lars Dembeck, Wolfram Lautenschlaeger, Stephan Bunse, A. Stavdas, J. Angelopoulos, and Christina T. Politi. Hybrid burst/packet switching architectures from IP NOBEL. In Benjamin B. Dingel, Ken ichi Sato, Werner Weiershausen, and Achyut K. Dutta, editors, *Optical Transmission Systems and Equipment for Networking V*, volume 6388, page 63880C. SPIE, 2006. http://link.aip.org/link/?PSI/6388/63880C/1, doi:10.1117/12.691610.

[55] W.C. Wong and D.J. Goodman. A packet reservation multiple access protocol for integrated speech and data transmission. *Communications, Speech and Vision, IEE Proceedings I*, 139(6):607–612, Dec. 1992.

[56] Henrik Abrahamsson, Bengt Ahlgren, Juan Alonso, Anders Andersson, and Per Kreuger. A Multi Path Routing Algorithm for IP Networks Based on Flow Optimisation. In *In Third COST 263 International Workshop on Quality of Future Internet Services, QoFIS*, pages 135–144. Springer, LNCS, 2002.

[57] Gunnar Karlsson and Mikhail I. Smirnov, editors. *Quality for All*, volume 2811 of *Lecture Notes in Computer Science*. Springer, October 2003.

[58] Yu Dong, Dingding Wang, N. Pissinou, and Jian Wang. Multipath load balancing in transport layer. In *Next Generation Internet Networks, 3rd EuroNGI Conference on*, pages 135–142, May 2007. doi:10.1109/NGI.2007.371208.

[59] K. Rojviboonchai, T. Osuga, and H. Aida. R-M/TCP: protocol for reliable multi-path transport over the Internet. In *Advanced Information Networking and Applications, 2005. AINA 2005. 19th International Conference on*, volume 1, pages 801–806 vol.1, March 2005. doi:10.1109/AINA.2005.289.

[60] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley. Multi-Path TCP: A Joint Congestion Control and Routing Scheme to Exploit Path Diversity in the Internet. *Networking, IEEE/ACM Transactions on*, 14(6):1260–1271, Dec. 2006. doi:10.1109/TNET.2006.886738.

[61] Jiayue He and Jennifer Rexford. Toward internet-wide multipath routing. *IEEE Network*, 22(2):16–21, 2008.

[62] Tat Wing Chim, Kwan L. Yeung, and King-Shan Lui. Traffic distribution over equal-cost-multi-paths. *Computer Networks*, 49(4):465–475, November 2005. doi:10.1016/j.comnet.2005.01.011.

[63] Stephen Mueller, Rosep. Tsang, and Dipak Ghosal. Multipath routing in mobile ad hoc networks: Issues and challenges. In *In Performance Tools and Applications to Networked Systems, volume 2965 of LNCS*, pages 209–234. Springer-Verlag, 2004.

[64] Ivan Gojmerac. *Adaptive Multi-Path routing for Internet Traffic Engineering*. Monographie, Technische Universität Wien, Technische Universität Wien, Fakultät für Elektrotechnik und Informationstechnik, April 2007.

[65] T. Battestilli and H. Perros. Optical burst switching: A survey. Technical report, North Carolina State University at Raleigh, Raleigh, NC, USA, 2002.

[66] C.M. Gauger. *Novel Network Architecture for Optical Burst Transport - Communication Networks and Computer Engineering Report No. 92.* PhD thesis, Universität Stuttgart, 2006.

[67] D. Klonidis, C.T. Politi, R. Nejabati, M.J. O'Mahony, and D. Simeonidou. OPSnet: design and demonstration of an asynchronous high-speed optical packet switch. *Lightwave Technology, Journal of*, 23(10):2914–2925, Oct. 2005. doi:10.1109/JLT.2005.856167.

[68] J.J.P.C. Rodrigues, M.M. Freire, and P. Lorenz. One-way resource reservation protocols for IP over optical burst switched mesh networks. In *Proceedings of Systems Communications*, pages 229–234, 2005.

[69] Myunsik Yoo and Chunming Qiao. Just-enough-time (JET): A high speed protocol for bursty traffic in optical networks. In *Proceedings of the IEEE/LEOS Summer Topical Meetings*, pages 26–27, Montreal, Canada, August 1997.

[70] K. Dolzer, C.M. Gauger, J. Späth, and S. Bodamer. Evaluation of reservation mechanisms for optical burst switching. *AEÜ International Journal of Electronics and Communications*, 55(1), January 2001.

[71] Sascha Junghans. *Realisierbarkeit von Scheduling-Modulen in Optical Burst Switching-Kernknoten*. PhD thesis, University of Stuttgart, 2007.

[72] Sascha Junghans. Pre-estimate burst scheduling (PEBS): An efficient architecture with low realization complexity for burst scheduling disciplines. In *Proceedings of the Fifth International Workshop on Optical Burst/Packet Switching (WOBS)*, 2005.

[73] A. Kaheel and H. Alnuweiri. Batch scheduling algorithms: a class of wavelength schedulers in optical burst switching networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, volume 3, pages 1713–1719, 2005.

[74] B. Kim, S. Lee, Y. Choi, and Y. Cho. An efficient preemption-based channel scheduling algorithm for service differentiation in obs networks. *Computer Communications*, 29(12):2348–2360, August 2006.

[75] E. Kozlovski, M. Duser, I. de Miguel, and P. Bayvel. Analysis of burst scheduling for dynamic wavelength assignment inoptical burst-switched networks. In *Proceedings of the 14th AnnualMeeting of the IEEE Lasers and Electro-Optics Society*, volume 1, pages 161–162 vol.1, 2001.

[76] J. Li, C. Qiao, and Y. Chen. Recent progress in the scheduling algorithms in optical-burst-switched networks. *Journal of Optical Networking*, 3:229–241, April 2004. http://www.osa-jon.org/abstract.cfm?URI=JON-3-4-229.

[77] Tarek S. El-Bawab. *Optical Switching*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[78] Zvi Rosberg, Andrew Zalesky, Hai L. Vu, and Moshe Zukerman. Analysis of obs networks with limited wavelength conversion. *IEEE/ACM Transactions on Networking*, 14(5):1118–1127, 2006. doi:10.1109/TNET.2006.882855.

[79] Christoph M. Gauger, M. Köhn, and J. Scharf. Performance of contention resolution strategies in OBS network scenarios. In *Proceedings of the 9th Optoelectronics and Communications Conference/3rd International Conference on the Optical Internet (OECC/COIN)*, Yokohama/Japan, July 2004.

[80] Christoph M. Gauger, M. Köhn, and J. Scharf. Comparison of contention resolution strategies in OBS network scenarios. In *Proceedings of the 6th International Conference on Transparent Optical Networks (ICTON)*, volume 1, pages 18–21, 2004.

[81] C.M. Gauger. Dimensioning of FDL buffers for optical burst switching nodes. In *Proceedings of the 6th IFIP Working Conference on Optical Network Design and Modelling (ONDM 2002)*. IFIP, February 2002.

[82] A.G.P. Rahbar and O.W.W. Yang. Contention avoidance and resolution schemes in bufferless all-optical packet-switched networks: a survey. *Communications Surveys & Tutorials, IEEE*, 10(4):94–107, 2008. doi:10.1109/SURV.2008.080408.

[83] Jia Lu, G. Mohan, and Kee Chaing Chua. Adaptive proportional flow routing in optical burst switching networks. *Communications, 2006. ICC '06. IEEE International Conference on*, 6:2581–2586, June 2006. doi:10.1109/ICC.2006.255168.

[84] M. Duser and P. Bayvel. Performance of a dynamically wavelength-routed optical burst switched network. *Photonics Technology Letters, IEEE*, 14(2):239–241, Feb 2002. doi:10.1109/68.980534.

[85] Nischal M. Piratla and Anura P. Jayasumana. Metrics for packet reordering—a comparative analysis. *International Journal of Communication Systems*, 21(1):99–113, 2008. doi:10.1002/dac.v21:1.

[86] Colin Perkins. *RTP: Audio and Video for the Internet*. Addison-Wesley Professional, 2003.

[87] N.M. Piratla and A.P. Jayasumana. Reordering of packets due to multipath forwarding - an analysis. *Communications, 2006. ICC '06. IEEE International Conference on*, 2:829–834, June 2006. doi:10.1109/ICC.2006.254810.

[88] N.M. Piratla, A.P. Jayasumana, and T. Banka. *Reorder Density (RD): A Formal, Comprehensive Metric for Packet Reordering*, volume 3462/2005 of *Lecture Notes in Computer Science*, pages 78–89. Springer Berlin/Heidelberg, 2005. doi:10.1007/11422778_7.

[89] L. Gharai, C. Perkins, and T. Lehman. Packet reordering, high speed networks and transport protocol performance. *Computer Communications and Networks, 2004. ICCCN 2004. Proceedings. 13th International Conference on*, pages 73–78, October 2004. doi:10.1109/ICCCN.2004.1401591.

[90] John Bellardo and Stefan Savage. Measuring packet reordering. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 97–105, New York, NY, USA, 2002. ACM. doi:10.1145/637201.637216.

[91] Xiapu Luo and Rocky K. C. Chang. Novel approaches to end-to-end packet reordering measurement. In *IMC '05: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 20–20, Berkeley, CA, USA, 2005. USENIX Association.

[92] N.M. Piratla, A.P. Jayasumana, and T. Banka. On reorder density and its application to characterization of packet reordering. *Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on*, pages 156–165, 15-17 Nov. 2005. doi:10.1109/LCN.2005.96.

[93] T. Banka, A.A. Bare, and A.P. Jayasumana. Metrics for degree of reordering in packet sequences. *Local Computer Networks, 2002. Proceedings. LCN 2002. 27th Annual IEEE Conference on*, pages 333–342, Nov. 2002.

[94] M. Laor and L. Gendel. The effect of packet reordering in a backbone link on application throughput. *IEEE Network*, 16(5):28–36, 2002. doi:10.1109/MNET.2002.1035115.

[95] M. Mellia, M. Meo, L. Muscariello, and D. Rossi. Passive analysis of TCP anomalies. *Computer Networks*, 52(14):2663–2676, 2008.

[96] F. Baccelli and A.M. Makowski. Queueing models for systems with synchronization constraints. *Proceedings of the IEEE*, 77(1):138–161, Jan 1989. doi:10.1109/5.21076.

[97] Ye Xia and David Tse. Analysis on packet resequencing for reliable network protocols. *Perform. Eval.*, 61(4):299–328, 2005. doi:10.1016/j.peva.2004.09.002.

[98] Alain Jean-Marie and Levent Gün. Parallel queues with resequencing. *J. ACM*, 40(5):1188–1208, 1993. doi:10.1145/174147.169748.

[99] S. Chowdhury. An analysis of virtual circuits with parallel links. *Communications, IEEE Transactions on*, 39(8):1184–1188, Aug 1991. doi:10.1109/26.134005.

[100] G. Harrus and B. Plateau. Queueing analysis of a reordering issue. *Software Engineering, IEEE Transactions on*, SE-8(2):113–123, March 1982.

[101] François Baccelli, Erol Gelenbe, and Brigitte Plateau. An End-to-End Approach to the Resequencing Problem. *J. ACM*, 31(3):474–485, 1984. doi:10.1145/828.1883.

[102] Tak-Shing Yum and Tin-Yee Ngai. Resequencing of messages in communication networks. *Communications, IEEE Transactions on*, 34(2):143–149, Feb 1986.

[103] S. Gunreben and G. Hu. A Multi-layer Analysis on Reordering in Optical Burst Switched Networks. *IEEE Communication Letters*, 11(12), December 2007.

[104] S. Gunreben. Multi-layer Analysis to Quantify the Impact of Optical Burst Reordering on TCP Performance. In *Proceedings of the 9th International Conference on Transparent Optical Networks (ICTON 2007)*, July 2007.

[105] S. Gunreben. An Optical Burst Reordering Model for a Time-based Burst Assembly Scheme. In *Proceedings of the International Workshop on Optical Burst/Packet Switching 2008 (WOBS 2008)*, September 2008.

[106] Sebastian Gunreben and Óscar González de Dios. Why deterministic traffic shows the highest reordering ratio. In *submitted to Workshop on Optical Burst Switching (WOBS)*, Madrid, 2009.

[107] F. Kamoun, L. Kleinrock, and R. Muntz. Queueing analysis of the reordering issue in a distributed database concurrency control mechanism. In *Proceedings of the Second International Conference on Distributed Computing Systems*, pages 13–23, Versailles, France, April 1981.

[108] M. Köhn and G. Hu. Evaluation of Packet Delay in OBS Edge Nodes. In *Proceedings of the 8th International Conference on Transparent Optical Networks (ICTON 2006)*, June 2006.

[109] P.J. Kühn. Approximate analysis of general queuing networks by decomposition. *IEEE Transactions on Communications*, 27(1):113–126, January 1979.

[110] Walter Rudin. *Real and complex analysis, 3rd ed.* McGraw-Hill, Inc., New York, NY, USA, 1987.

[111] MathWorks, Inc. *Optimization Toolbox - User's Guide*, 3 edition, 2004.

[112] M. J. D. Powell. *The Convergence of Variable Metric Methods For Nonlinearly Constrained Optimization Calculations.* Academic Press, 1978.

[113] M. J. D. Powell. *A Fast Algorithm for Nonlinearly Constrained Optimization Calculations*, volume 630. Springer Verlag, 1978.

[114] S. P. Han. A Globally Convergent Method for Nonlinear Programming. *Journal of Optimization Theory and Applications*, 22:297, 1977.

[115] P.E. Gill. *Practical Optimization.* Academic Press, 1981.

[116] The Linux Foundation. *Net:Em.* The Linux Foundation, May 2009. `http://www.linuxfoundation.org/en/Net:Em`.

[117] Mark Carson and Darrin Santay. NIST Net: a Linux-based network emulation tool. *SIGCOMM Comput. Commun. Rev.*, 33(3):111–126, 2003. `doi:10.1145/956993.957007`.

[118] Stephen Hemminger. *Net:Bridge.* The Linux Foundation, May 2009. `http://www.linuxfoundation.org/en/Net:Bridge`.

[119] Martin A. Brown. *Traffic Control HOWTO.* linux-ip.net, October 2006. `http://tldp.org/HOWTO/Traffic-Control-HOWTO/`.

[120] Anurag Kumar, D. Manjunath, and Joy Kuri. *Communication Networking: An Analytical Approach.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[121] Robert Love. *Linux Kernel Development, Second Edition.* Novell Press, January 2005.

[122] Hui-Chuan Chen and Yoshinori Asau. *On Generating Random Variates from an Empirical Distribution*, volume 6. Taylor & Francis, 1974. `http://www.informaworld.com/10.1080/05695557408974949`.

[123] Paul Bratley, Bennett L. Fox, and Linus E. Schrage. *A guide to simulation (2nd ed.).* Springer-Verlag New York, Inc., New York, NY, USA, 1987.

[124] J. M. Chambers, W. S. Cleveland, Beat Kleiner, and Paul A. Tukey. *Graphical Methods for Data Analysis*. Wadsworth, 1983.

[125] H. Cramér. *Mathematical methods of statistics*. Princeton: Princeton University Press, 1946.

[126] Marek Fisz. *Wahrscheinlichkeitsrechnung und Mathematische Statistik*, volume 40. VEB Deutscher Verlag der Wissenschaften, 1958.

[127] T. W. Anderson and D. A. Darling. Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *The Annals of Mathematical Statistics*, 23(2):193–212, 1952. `http://www.jstor.org/stable/2236446`.

[128] Josef Heinhold and Karl-Walter Gaede. *Ingenieur-Statistik*. R. Oldenburg Verlag, München, Wien, 1964.

[129] Stefan Bodamer, Klaus Dolzer, Christoph Gauger, Marc Barisch, and Marc Necker. *IKR Simulation Library 2.6 User Guide*. Institut für Kommunikationsnetze und Rechnersysteme, Universität Stuttgart, June 2004.

[130] C.M. Gauger, H. Buchta, E. Patzak, and J. Saniter. Performance meets Technology - an Integrated Evaluation of OBS Nodes with FDL Buffers. In *Proceedings of the 1st International Workshop on Optical Burst Switching*, October 2003.

[131] Marek Kuczma. *An introduction to the theory of functional equations and inequalities. Cauchy's equation and Jensen's inequality. Edited by Attila Gilányi. 2nd ed.* Basel: Birkhäuser. xiv, 2009.

[802.16] IEEE Computer Society. 802.16: IEEE Standard for Local and Metropolitan Area Networks – Air Interface for Fixed Broadband Wireless Access Systems, October 2004.

[802.1ag] IEEE Computer Society. 802.1ag: IEEE Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks, Amendment 5: Connectivity Fault Management, December 2007.

[802.1ah] IEEE Computer Society. 802.1ah: Draft Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks, Amendment 6: Provider Backbone Bridges, November 2007.

[802.1ay] IEEE Computer Society. 802.1ay: Draft Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks, Amendment: Provider Backbone Bridge–Traffic Engineering, December 2007.

[802.1D] IEEE Computer Society. 802.1D: IEEE Standard for Local and Metropolitan Area Networks–Media Access Control (MAC) Bridges, 2004.

[802.3] IEEE Computer Society. 802.3: IEEE Standard for Local and Metropolitan Area Networks–Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications, 2005.

[802.3ba] IEEE Computer Society. P802.3ba: IEEE Standard for Local and Metropolitan Area Networks–Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Amendment: Media Access Control Parameters, Physical Layers and Management Parameters for 40 Gb/s and 100 Gb/s Operation, 2008. `http://www.ieee802.org/3/ba/`.

[RFC 768] J. Postel. User Datagram Protocol. RFC 768, IETF, August 1980.

[RFC 793] J. Postel. Transmission Control Protocol. RFC 793, IETF, September 1981.

[RFC 1247] J. Moy. OSPF Version 2. RFC 1247, IETF, July 1991.

[RFC 1323] V. Jacobson, R. Braden, and D. Borman. TCP Extensions for High Performance. RFC 1323, IETF, May 1992.

[RFC 1732] M. Crispin. IMAP4 Compatibility with IMAP2 and IMAP2bis. RFC 1732, IETF, December 1994.

[RFC 2330] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis. Framework for IP Performance Metrics. RFC 2330, IETF, May 1998.

[RFC 2581] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC 2581, IETF, April 1999.

[RFC 2960] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson. Stream Control Transmission Protocol. RFC 2960, IETF, October 2000.

[RFC 2991] D. Thaler and C. Hopps. Multipath Issues in Unicast and Multicast Next-Hop Selection. RFC 2991, IETF, November 2000.

[RFC 2992] C. Hopps. Analysis of an Equal-Cost Multi-Path Algorithm. RFC 2992, IETF, November 2000.

[RFC 3031] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, IETF, January 2001.

[RFC 3168] K. Ramakrishnan, S. Floyd, and D. Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, IETF, September 2001.

[RFC 3261] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, IETF, June 2002.

[RFC 3286] L. Ong and J. Yoakum. An Introduction to the Stream Control Transmission Protocol (SCTP). RFC 3286, IETF, May 2002.

[RFC 3550] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550, IETF, July 2003.

[RFC 3564] F. Le Faucheur and W. Lai. Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering. RFC 3564, IETF, July 2003.

[RFC 3588] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, and J. Arkko. Diameter Base Protocol. RFC 3588, IETF, September 2003.

[RFC 3697] J. Rajahalme, A. Conta, B. Carpenter, and S. Deering. IPv6 Flow Label Specification. RFC 3697, IETF, March 2004.

[RFC 3945] Generalized Multi-Protocol Label Switching (GMPLS) Architecture. RFC 3945, IETF, October 2004.

[RFC 4202] Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS). RFC 4202, IETF, October 2005.

[RFC 4340] E. Kohler, M. Handley, and S. Floyd. Datagram Congestion Control Protocol (DCCP). RFC 4340, IETF, March 2006.

[RFC 4655] A. Farrel, J.-P. Vasseur, and J. Ash. A Path Computation Element (PCE)-Based Architecture. RFC 4655, IETF, August 2006.

[RFC 4737] A. Morton, L. Ciavattone, G. Ramachandran, S. Shalunov, and J. Perser. Packet Reordering Metrics. RFC 4737, IETF, November 2006.

[RFC 4740] Diameter Session Initiation Protocol (SIP) Application. RFC 4740, IETF, November 2006.

[RFC 4782] S. Floyd, M. Allman, A. Jain, and P. Sarolahti. Quick-Start for TCP and IP. RFC 4782, IETF, January 2007.

[RFC 5236] A. Jayasumana, N. Piratla, T. Banka, A. Bare, and R. Whitner. Improved Packet Reordering Metrics. RFC 5236, IETF, June 2008.

[RFC 5351] P. Lei, L. Ong, M. Tuexen, and T. Dreibholz. An Overview of Reliable Server Pooling Protocols. RFC 5351, IETF, September 2008.

[RFC 5657] L. Dusseault and R. Sparks. Guidance on Interoperation and Implementation Reports for Advancement to Draft Standard. RFC 5657, IETF, September 2009.

[G.694.2] ITU-T. Spectral grids for WDM applications: CWDM wavelength grid. Rec. G.694.2, ITU-T, December 2003.

[G.7041/Y.1303] ITU-T. Generic framing procedure (GFP). Rec. G.7041/Y.1303, ITU-T, August 2005.

[G.7042/Y.1305] ITU-T. Link capacity adjustment scheme (LCAS) for virtual concatenated signals. Rec. G.7042/Y.1305, ITU-T, February 2004.

[G.709] ITU-T. Synchronous multiplexing structure. Rec. G.709, ITU-T, March 1993.

[G.723.1] ITU-T. Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s. Rec. G.723.1, ITU-T, March 1996.

[G.803] ITU-T. Architecture of transport networks based on the synchronous digital hierarchy (SDH). Rec. G.803, ITU-T, March 2000.

[G.805] ITU-T. Generic functional architecture of transport networks. Rec. G.805, ITU-T, March 2000.

[G.809] ITU-T. Functional architecture of connectionless layer networks. Rec. G.809, ITU-T, March 2003.

[G.872] ITU-T. Architecture of optical transport networks. Rec. G.872, ITU-T, November 2001.

[M.3050.0] ITU-T. Enhanced Telecom Operations Map (eTOM) - Introduction. Rec. M.3050.0, ITU-T, July 2004.

[M.3400] ITU-T. TMN management functions. Rec. M.3400, ITU-T, February 2000.

[X.200] ITU-T. Information technology - Open Systems Interconnection - Basic Reference Model: The basic model. Rec. X.200, ITU-T, July 1994.

[Y.100] ITU-T. General overview of the Global Information Infrastructure standards development. Rec. Y.100, ITU-T, June 1998.

[Y.1001] ITU-T. IP framework - A framework for convergence of telecommunications network and IP network technologies. Rec. Y.1001, ITU-T, November 2000.

[Y.110] ITU-T. Global Information Infrastructure principles and framework architecture. Rec. Y.110, ITU-T, June 1998.

[Y.1540] ITU-T. Internet protocol data communication service - IP packet transfer and availability performance parameters. Rec. Y.1540, ITU-T, December 2002.

[Y.1541] ITU-T. Network performance objectives for IP-based services. Rec. Y.1541, ITU-T, May 2002.

[Y.2001] ITU-T. General overview of NGN. Rec. Y.2001, ITU-T, December 2004.

[Y.2011] ITU-T. General principles and general reference model for Next Generation Networks. Rec. Y.2011, ITU-T, October 2004.

Acknowledgement

This thesis finalises my six years stage at the Institute of Communication Networks and Computer Engineering (IKR) at the University of Stuttgart, Germany. The stage at the institute gave me the opportunity to develop myself in three different directions: Thematically: I got a broad and deep view into the fields of computer networks and network engineering. Methodically: I enriched myself with a large number of scientific research methods in my own research area and connected fields. Personally: I learnt how to work in and to organise public and private funded projects, to advise and to guide people. I also extended my inter-cultural skills because of my international project work and by travelling within Europe.

Prof. Paul Kühn gave me the opportunity for development by the position of a research staff member at his institute. I want to thank him for this chance and his confidence in me during my stage. Especially, in the last phase, Prof. Kühn provided profound assistance with respect to this thesis. Additionally, I would like to thank Prof. Josep Solé Pareta twice. The first time for hosting me in Barcelona within the e-Photone ONe project and the second time for reviewing this thesis.

Many people shared my ideas of this thesis, gave me input and discussed findings critically. First of all, my colleagues at the IKR and especially the members of the High Speed Networks group discussed with me the concepts and ideas of this thesis. Many thanks to Guoqiang Hu, Detlef Sass, Martin Köhn, Christoph Gauger, Arthur Mutter, Simon Hauger, Elisabeth Georgieva and special thanks to Frank Feller and Joachim Scharf for carefully checking this document. Last but not least, I thank Ulrich Gemkow for all the administrative issues, the thoughtful discussions and granted freedom at the institute.

Within the various projects, I worked together with many colleagues from all over Europe. The following list is neither complete nor exhaustive but represents a relevant subset of people I worked with. From NOBEL and NOBEL 2, I want to mention Salvatore Spadaro, Josep Sole Pareta, Jordi Perelló, Davide Careglio, Mirek Klinkowski, Frank Rambach, Stefan Herbst, Monika Jäger, Michael Düser, Erwin Patzak and Michael Schlosser. Within the e-Photone ONe I want to list Nail Akar, Franco Calegatti, and Mario Pickavet. Some of these joint activities were continued or newly established within the BONE NoE. Here, I want to refer to Ramon Casellas, Ricardo Martinez, Raul Muñoz, Oscar de Dios Gonzalez, and Juan Fernandez Palacios. Within German national and bilateral industrial projects, the most important contacts are Gert Eilenberger, Wolfram Lautenschläger, Lars Dembeck, and Jens Milbrandt.

Writing a thesis costs – among others – patience. The writer needs patience as well as does his social community. For this thesis, I want to thank Yvonne for her patience and continually support during the time of writing. The last words are dedicated to my family, especially my brothers and parents. They gave me the opportunity to decide for this academic way at the university and were always supportive during this time. I am very thankful and appreciate this chance very much.