Compensation-Aware Connection Provisioning in Transport Networks

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

Vorgelegt von

Tobias Enderle

geboren in Bietigheim-Bissingen

Hauptberichter	Prof. Dr.-Ing. Andreas Kirstädter
Mitberichterin	Prof.'in Dr.-Ing. Carmen Mas Machuca

Tag der mündlichen Prüfung 11. Juli 2024

Institut für Kommunikationsnetze und Rechnersysteme
der Universität Stuttgart

2024

Abstract

A network operator has to generate sustainable profits from the sale of network services in order to maintain a viable business. To this end, monetizing as much of the available network capacity as possible is crucial. At the same time, the traffic demand is ever-increasing, and the operator must invest in network upgrades constantly to provide enough capacity. Over the past decades, various new technologies have been introduced, mainly in the optical layer, e.g., wavelength-division multiplexing (WDM), erbium-doped fiber amplifiers (EDFAs), and coherent transmission. Each technological step increased the network capacity further. However, recent advances, like probabilistic constellation shaping (PCS), are approaching the physical capacity limits of the deployed network infrastructure. New approaches are needed to continue the steady increase in capacity. One such approach is more intelligent and, by that, more resource-efficient service provisioning. This thesis proposes a connection provisioning mechanism called *compensation-aware provisioning with surplus sharing* (*CAPSS*), with resource efficiency as one major goal.

A different aspect of communication networks is reliability. A network operator guarantees a certain level of reliability — typically in the form of a minimum monthly service availability — to its customers. Such reliability guarantees are stipulated in a service level agreement (SLA). SLAs are especially important for customers like companies, data center (DC) operators, or other network operators. A violation of the SLA results in compensation that the operator must pay the customer. The amount of compensation is determined by the compensation policy in the SLA. To avoid SLA violations, the operator typically deploys redundant network resources, i.e., resources that are unused most of the time. This contradicts the operator's first goal of monetizing the available capacity as much as possible. Therefore, an important task for an operator — at least theoretically — is to balance redundancy and the risk of SLA violation.

However, in practice, many operators prefer to minimize SLA violations by deploying much redundancy. There are different reasons for this behavior. Apart from the monetary risk SLA violations pose, an operator legitimately also fears that its reputation could be damaged. This might lead to increased customer churn and could jeopardize the operator's long-term profits. A second reason is that it is difficult to quantify the effort for redundancy and the risk of SLA violations in economic terms. Keeping the balance is then almost impossible. In this thesis, we argue that customers react differently to SLA violations. Some depend on a highly reliable service, of course. However, others implement their own redundancy anyway, e.g., by buying network services from different operators. They can tolerate SLA violations, especially if that means that the service itself is less expensive. Therefore, CAPSS allows SLA violations to a certain extent. In that way, it increases the network capacity.

To be more precise, we contribute a mathematical model for the probability distribution of a connection's SLA compensation. This model is a step toward characterizing a connection's reliability from an *economic perspective*. The provisioning mechanism we build on

top of this model, CAPSS, provisions a connection such that the incurred compensation exceeds an operator-defined limit only with an operator-defined probability. In other words, a connection complies with the operator-defined compensation limit with a probability we call the *compliance probability*. With this approach, an operator can limit the monetary risk associated with SLA violations.

For several reasons we discuss in this thesis, connections may perform better than required regarding the compliance probability. They generate *compliance surplus*. Therefore, an additional feature of CAPSS is *surplus sharing*. Surplus is shared among different connections to relax their respective reliability requirements. Eventually, this increases the network capacity because less redundancy is needed.

Overall, CAPSS combines compensation-aware and resource-efficient connection provisioning. It is one of only a few approaches in the literature that consider compensation. To the best of the author's knowledge, CAPSS is the only approach that incorporates the compliance probability into the provisioning process and combines it with a sharing mechanism.

We evaluate CAPSS in simulation studies with dynamically arriving connection requests and departures. We consider both realistic networks and synthetic networks with controlled properties. The simulations confirm that the probability model for the compensation is accurate in the scenarios it is designed for. Furthermore, they show that CAPSS, as a whole, works as expected, i.e., it meets the compliance probability targeted by the operator. To evaluate the performance, we compare CAPSS with a conventional provisioning mechanism that only considers the SLA availability, not the compensation, and does not employ any sharing mechanism. Moreover, we study the effect of surplus sharing, i.e., we compare CAPSS and a version of CAPSS without surplus sharing, named CAP. The corresponding simulation studies focus on the network capacity.

The comparison with the conventional provisioning mechanism shows that it depends on the scenario parameters whether CAPSS increases the capacity or not. The main factors of influence are the compensation policy and the contract period associated with the connection. A general advantage of CAPSS is that it consistently meets the operator's targeted compliance probability.

On the other hand, the comparison between CAPSS and CAP shows that surplus sharing increases the capacity in almost all scenarios. The only exceptions are networks with very high or very low overall reliability. Since the reliability is usually assumed to depend on the link length, this translates to networks with very small or very large geographical extent (< 10 km or $\gg 1000$ km average node pair distance). For all other networks, capacity increases are realized. For small networks, increases of more than 100 % are possible. In the realistic networks, capacity increases range from around 5 % to 60 %. Again, smaller networks benefit more because they are assumed to be more reliable and, thus, generate more surplus for sharing. Additionally, the capacity increase is influenced by the "strictness" of the operator's targets. That means if the operator allows high compensation and only requires a low compliance probability, high capacity increases can be expected.

Overall, CAPSS brings technical and economic aspects closer together by incorporating the probability distribution of the SLA compensation into the provisioning process. Furthermore, its surplus sharing mechanism enables resource-efficient provisioning. As a result, a network operator can better achieve and maintain sustainable profits by employing CAPSS.

Kurzfassung

Netzbetreiber müssen mit dem Verkauf von Netzdiensten nachhaltige Gewinne erwirtschaften, um langfristig wirtschaftlich erfolgreich zu sein. Zu diesem Zweck ist es entscheidend, die verfügbare Netzkapazität so gut wie möglich zu monetarisieren. Gleichzeitig nimmt die Verkehrsnachfrage in heutigen Netzen kontinuierlich zu und die Betreiber müssen ihre Netze ständig ausbauen, um genügend Kapazität bereitzustellen. In den vergangenen Jahrzehnten wurden verschiedene neue Technologien eingeführt, hauptsächlich in der optischen Übertragungstechnik. Beispiele hierfür sind Wellenlängenmultiplexing (WDM), Erbiumdotierte Faserverstärker (EDFAs) und kohärente Übertragung. Mit jedem technologischen Schritt wurde die Netzkapazität weiter erhöht. Die jüngsten Fortschritte, wie Probabilistic Constellation Shaping (PCS), stoßen jedoch an die physikalischen Kapazitätsgrenzen der vorhandenen Netzinfrastruktur. Daher werden neue Ansätze benötigt, um der stetig zunehmenden Verkehrsnachfrage gerecht zu werden. Ein solcher Ansatz ist eine intelligentere und damit ressourceneffizientere Bereitstellung von Netzdiensten. In dieser Arbeit wird ein Bereitstellungsmechanismus vorgeschlagen, der als *compensation-aware provisioning with surplus sharing* (*CAPSS*) bezeichnet wird und unter anderem auf Ressourceneffizienz abzielt.

Ein anderer Aspekt von Kommunikationsnetzen ist die Zuverlässigkeit. Ein Netzbetreiber garantiert seinen Kunden ein bestimmtes Maß an Zuverlässigkeit — normalerweise in Form einer monatlichen Mindestverfügbarkeit. Solche Zuverlässigkeitsgarantien werden in einem Service Level Agreement (SLA) festgelegt. SLAs sind insbesondere dann wichtig, wenn der Kunde ein Unternehmen, ein Rechenzentrumsbetreiber oder ein anderer Netzbetreiber ist. Eine Verletzung des SLA führt zu einer Entschädigung (compensation), die der Betreiber an den Kunden bezahlen muss. Die Höhe der Entschädigung wird durch Regelungen (compensation policy) im SLA bestimmt. Um SLA-Verletzungen zu vermeiden, setzt der Betreiber meist redundante Netzressourcen ein, d. h. Ressourcen, die die meiste Zeit ungenutzt sind. Dies widerspricht dem ersten Ziel des Betreibers, die verfügbare Kapazität so gut wie möglich zu monetarisieren. Eine wichtige Aufgabe des Netzbetreibers ist es deshalb, ein Gleichgewicht zwischen Redundanz und dem Risiko von SLA-Verletzungen herzustellen.

In der Praxis ziehen es jedoch viele Betreiber vor, SLA-Verletzungen durch den Einsatz von viel Redundanz zu minimieren. Für dieses Verhalten gibt es unterschiedliche Gründe. Abgesehen vom finanziellen Risiko, das SLA-Verletzungen mit sich bringen, fürchten Betreiber, dass ihr Ruf Schaden nehmen könnte. Dies könnte zu einer erhöhten Kundenabwanderung führen und die langfristigen Gewinne des Betreibers gefährden. Ein zweiter Grund ist, dass es schwierig ist, den Aufwand für Redundanz und das Risiko von SLA-Verletzungen ökonomisch zu quantifizieren. Das angesprochene Gleichgewicht einzustellen, ist dann

fast unmöglich. In dieser Arbeit argumentieren wir, dass Kunden unterschiedlich auf SLA-Verletzungen reagieren. Einige sind natürlich auf einen hochzuverlässigen Dienst angewiesen. Andere hingegen implementieren ohnehin ihre eigene Redundanz, beispielsweise durch den Einkauf von Diensten bei verschiedenen Netzbetreibern. Sie können SLA-Verletzungen tolerieren, vor allem wenn dies bedeutet, dass der Dienst selbst günstiger ist. Daher lässt CAPSS SLA-Verletzungen bis zu einem gewissen Grad zu und erhöht auf diese Weise die Netzkapazität.

In dieser Arbeit wird ein mathematisches Modell für die Wahrscheinlichkeitsverteilung der SLA-Entschädigung einer Netzverbindung entwickelt. Dieses Modell ist ein Schritt hin zu einer Charakterisierung der Zuverlässigkeit einer Verbindung aus *ökonomischer Sicht*. CAPSS baut auf diesem Modell auf. Der Mechanismus stellt eine Verbindung so bereit, dass die anfallende Entschädigung eine vom Betreiber festgelegte Grenze nur mit einer bestimmten, ebenfalls vom Betreiber festgelegten, Wahrscheinlichkeit überschreitet. Mit anderen Worten: Eine Verbindung hält die vom Betreiber festgelegte Entschädigungsgrenze mit einer Wahrscheinlichkeit ein, die wir als *Erfüllungswahrscheinlichkeit* (*compliance probability*) bezeichnen. Mit diesem Ansatz kann ein Betreiber das mit SLA-Verletzungen verbundene finanzielle Risiko begrenzen.

Aus verschiedenen Gründen, die wir in dieser Arbeit erörtern, können Verbindungen zuverlässiger sein als erforderlich. Sie erzeugen einen *Überschuss an Erfüllungswahrscheinlichkeit* (*compliance surplus*). Deshalb ist eine weitere Funktion von CAPSS die Verteilung dieser Überschüsse (*surplus sharing*). Überschüsse werden unter den verschiedenen Verbindungen aufgeteilt, um deren Zuverlässigkeitsanforderungen zu lockern. Letztlich erhöht sich dadurch die Netzkapazität, da weniger Redundanz erforderlich ist. Insgesamt stellt CAPSS einen Mechanismus zur ressourceneffizienten Verbindungsbereitstellung unter Berücksichtigung der SLA-Entschädigung dar. CAPSS ist einer von wenigen Ansätzen in der Literatur, der SLA-Entschädigungen berücksichtigt. Nach bestem Wissen des Autors ist CAPSS der einzige Ansatz, der die Erfüllungswahrscheinlichkeit in den Bereitstellungsprozess einbezieht und mit einem Sharing-Mechanismus kombiniert.

CAPSS wird in Simulationsstudien mit dynamisch eintreffenden Verbindungsanfragen bewertet. Wir betrachten sowohl realistische Netze als auch synthetische Netze mit kontrollierten Netzeigenschaften. Die Simulationen bestätigen, dass das Wahrscheinlichkeitsmodell für die Entschädigung in den Szenarien, für die es konzipiert ist, präzise arbeitet. Außerdem zeigen die Simulationen, dass CAPSS als Ganzes wie erwartet funktioniert, d. h. die vom Betreiber angestrebte Erfüllungswahrscheinlichkeit wird erreicht. Zur Bewertung der Leistungsfähigkeit vergleichen wir CAPSS mit einem konventionellen Bereitstellungsmechanismus. Dieser berücksichtigt nur die SLA-Verfügbarkeit, jedoch nicht die SLA-Entschädigung. Außerdem setzt er keinen Sharing-Mechanismus ein. Des Weiteren untersuchen wir die Auswirkungen der Überschussaufteilung, d. h. wir vergleichen CAPSS mit einer Version ohne Überschussaufteilung (CAP). Die Simulationsstudien konzentrieren sich dabei auf die Netzkapazität.

Der Vergleich mit dem herkömmlichen Bereitstellungsmechanismus zeigt, dass es von den Szenarioparametern abhängt, ob CAPSS die Kapazität erhöht oder nicht. Die wichtigsten Einflussfaktoren sind die Compensation Policy und die Vertragslaufzeit. Ein genereller Vorteil von CAPSS ist, dass die Erfüllungswahrscheinlichkeit, die der Betreiber anstrebt, immer erreicht wird.

Der Vergleich zwischen CAPSS und CAP hingegen zeigt, dass die Verteilung von Überschüssen die Kapazität in fast allen Szenarien erhöht. Die einzigen Ausnahmen sind Netze mit sehr hoher oder sehr geringer Zuverlässigkeit. Da meist davon ausgegangen wird, dass

die Zuverlässigkeit von der Linklänge abhängt, entspricht dies Netzen mit sehr geringer oder sehr großer geografischer Ausdehnung ($< 10\,\text{km}$ oder $\gg 1000\,\text{km}$ durchschnittliche Knotenpaarentfernung). Für alle anderen Netze werden Kapazitätssteigerungen realisiert. Bei kleinen Netzen sind Steigerungen von mehr als 100 % möglich. In den realistischen Netzen liegen die Kapazitätssteigerungen zwischen 5 % und 60 %. Auch hier profitieren kleinere Netze stärker, da davon ausgegangen wird, dass sie zuverlässiger sind und somit mehr Überschuss für die Verteilung erzeugt wird. Zusätzlich wird die Kapazitätssteigerung von der Zielsetzung des Betreibers beeinflusst. Wenn der Betreiber hohe Entschädigungszahlungen zulässt und nur eine geringe Erfüllungswahrscheinlichkeit anstrebt, sind hohe Kapazitätssteigerungen zu erwarten.

Durch die Einbeziehung der Wahrscheinlichkeitsverteilung der SLA-Entschädigung in den Bereitstellungsprozess bringt CAPSS technische und ökonomische Aspekte näher zusammen. Überdies ermöglicht das Verteilen von Überschüssen eine ressourceneffiziente Bereitstellung. Infolgedessen können Netzbetreiber durch den Einsatz von CAPSS besser nachhaltige Gewinne erzielen und diese aufrechterhalten.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

AC-GSP	availability-constrained general segment protection	48
ADT	allowed downtime	58
ANF	allowed number of failures	58
ASON	automatically switched optical network	49
CAGR	compound annual growth rate	23
CAP	compensation-aware provisioning	26
CAPSS	compensation-aware provisioning with surplus sharing	25
CC	cable cuts	106
CDF	cumulative distribution function	79
CI	confidence interval	106
CL	confidence limit	108
CO	central office	32
CVaR	conditional value at risk	55
CW	critical window	56
DC	data center	24
DSL	digital subscriber line	31
DWDM	dense wavelength-division multiplexing	109
ECMP	equal-cost multipath	75
EDFA	erbium-doped fiber amplifier	23
FRR	fast reroute	36
GMPLS	generalized multiprotocol label switching	36
i.i.d.	independent and identically distributed	75
IAT	inter-arrival time	104
IGP	interior gateway protocol	36
ILP	integer linear program	47
IoT	Internet of things	29
IP	Internet protocol	30
IS-IS	intermediate system to intermediate system	75
ISP	Internet service provider	24
JVM	Java virtual machine	104
MPLS	multiprotocol label switching	30
MPLS-TP	multiprotocol label switching – transport profile	32
MRC	monthly recurring charge	41
MTTF	mean time to failure	35
MTTR	mean time to repair	35

OSPF	open shortest path first	75
OTN	optical transport network	32
PCE	path computation element	75
PCS	probabilistic constellation shaping	23
PDF	probability density function	83
PMF	probability mass function	89
PON	passive optical network	31
PoP	point of presence	32
RAN	radio access network	99
RV	random variable	27
SDH	synchronous digital hierarchy	32
SDM	space-division multiplexing	23
SDN	software-defined networking	75
SLA	service level agreement	24
SLO	service level objective	42
SNCP	subnetwork connection protection	36
SONET	synchronous optical network	32
TCP	transmission control protocol	30
UL	urgency level	58
VaR	value at risk	55
WDM	wavelength-division multiplexing	23

Symbols

$\mathbf{1}_x$	Indicator function; equals 1 if x is a true statement and 0 otherwise	79
A	Experienced availability (RV)	39
$\mathbb{A}$	System state	79
a	Steady-state availability	38
α	Rate parameter of an exponential distribution	80
α_{SLA}	Monthly availability guaranteed in the SLA	43
$a(t)$	Point availability	37
$a^*(\Delta t)$	Interval availability	38
B	Cumulated time spent in state $\mathbb{B}$ during the time interval $[0, t_{\text{int}}]$ (RV)	79
$\mathbb{B}$	System state	79
β	Rate parameter of an exponential distribution	80
C	Compensation for the whole contract period (RV)	89
$\mathcal{C}_1$	Support of compensation C_j ; Range of compensation policy $g(u)$	87
$C_{1,i}$	Cumulated compensation for the first i billing cycles (RV)	89
$\mathcal{C}_i$	Support of compensation $C_{1,i}$	89
C_j	Compensation for the j-th billing cycle (RV)	87
$c_{\max}$	Allowed compensation	70
E	Set of edges in a graph	74
$\mathbf{E}(\cdot)$	Expectation of a random variable	81
$\mathbb{F}$	Failed state	75
$F_B(b)$	CDF of B	80
$F_{X_j}(x)$	CDF of X_j	85
G	Graph representing a network topology	74
$g(u)$	Compensation policy function	87
$\kappa_{\max}$	Maximum link capacity	112
$\kappa_{\min}$	Minimum link capacity	112
l	Offered load	137
λ	Constant failure rate	39
ℓ_e	Length of link e	51
$\lim_{x \nearrow y} f(x)$	Limit of function f if x approaches y from left or below	81
μ	Constant repair rate	39
$\mathbb{N}$	Natural numbers excluding zero	112
$\mathbb{N}_0$	Natural numbers including zero	96
$\mathbf{P}(\cdot)$	Probability	37
$\mathbf{P}(\cdot, \cdot)$	Joint probability of two events	80
$\mathbf{P}(\cdot \mid \cdot)$	Conditional probability	84
$\mathbf{P}(A < \alpha_{\text{SLA}})$	Probability of SLA or availability violation	49
$\mathbf{P}(C \leq c_{\max})$	Compliance probability	71

p_i	Computed compliance probability for connection i	71
$\bar{p}_n$	Average computed compliance over n connections	107
$\hat{p}_n$	Compliance ratio over n connections	107
$\Delta p_{\mathrm{s},i}$	Global compliance surplus after provisioning of connection i	72
p_t	Compliance target	71
$p^*_{\mathrm{t},i}$	Relaxed compliance target of connection i	72
$\mathbb{R}_{\geq 0}$	Non-negative real numbers	55
$\mathbb{R}_{>0}$	Positive real numbers	79
$T_{\mathbb{A},i}$	Uninterrupted time in state $\mathbb{A}$ (RV)	79
$T_{\mathbb{B},i}$	Uninterrupted time in state $\mathbb{B}$ (RV)	79
$T_{\mathbb{F},i}$	Uninterrupted time in the failed state $\mathbb{F}$ (RV)	75
$T_{\mathbb{W},i}$	Uninterrupted time in the working state $\mathbb{W}$ (RV)	75
t_a	Arrival time of a connection request	92
$t_{\mathrm{c},j}$	Duration of the j-th billing cycle	82
Δt_h	Contract period of a connection	70
t_int	Duration of observed time interval in the two-state system	79
$t_{\mathrm{s},j}$	Start of the j-th billing cycle	83
$\Delta t_{\mathrm{s},j}$	Duration between the start of a connection and its j-th billing cycle	84
U_j	Experienced unavailability in the j-th billing cycle (RV)	87
V	Set of vertices in a graph	74
$\mathbb{W}$	Working state	75
X_j	Cumulated downtime in the j-th billing cycle (RV)	82
ξ_j	Connection state at the beginning of the j-th billing cycle (RV)	84
$\xi(t)$	System or connection state at time t (RV)	79

1

Introduction

1.1 A Brief History of Communication Networks

In the history of communication networks, various disruptive changes regarding the employed technologies, the architectures, but also their purpose took place. We give a short overview partly based on [Muk+20, Ch. 1] and [Sim08, Sec. 1.1].

Early networks supported telephone calls only. The two parties talking to each other were directly connected by an electric circuit through the network. Electrical transmission was also used for the first computer networks. Computer networks enabled the exchange of data between computers instead of voice calls. This changed the purpose of communication networks fundamentally. It also changed the requirements, mainly in terms of the supported capacity. With the evolution of end-user applications and services, the amount of data transported in the networks grew and still grows exponentially. For example, [Tka10] report compound annual growth rates (CAGRs) for the Internet traffic in North America of around 151 % between 1990 and 2000 and 58 % between 2000 and 2010. Cisco [Cis19b] reports CAGRs for the global Internet traffic of 212 % between 1992 and 2002, 82 % between 2002 and 2007, and 37 % between 2007 and 2017. The global Internet data carried in 2017 in one second was estimated to be around 47 TB. The transportation of such massive amounts of data was made possible by another fundamental change, namely the introduction of fiber-optic communication technologies, such as wavelength-division multiplexing (WDM), erbium-doped fiber amplifiers (EDFAs), coherent transmission, and all-optical cross-connects.

For a long time, those technologies provided steady capacity increases at moderate costs. However, recent advances in digital signal processing and signal modulation, e.g., probabilistic constellation shaping (PCS) [CW19], are approaching the theoretical limits of a fiber's transmission capacity. To keep up with the growing traffic demand, researchers suggest different solutions. For example, adding another dimension of multiplexing, namely space [WN17]. Space-division multiplexing (SDM) uses multiple cores or multiple modes of a single fiber. It is expected to be more cost-efficient than simply adding parallel fibers

and upgrading network nodes accordingly. Another option is the extension of the optical spectrum from the C-band to the L-band [Ahm+21]. To handle increasing traffic in the short term, researchers like Pointurier [Poi17] suggest reducing system margins in the network. This allows postponing costly network upgrades and leads to higher network utilization in general. However, it also makes the network less reliable. This is a difficult tradeoff for network operators because, just like network capacity and resource efficiency, network reliability is a very important topic.

1.2 Network Reliability and SLAs

A network service is reliable if it always operates as expected. However, failures of network components which lead to service interruption cannot be prevented entirely. Therefore, a network operator has to employ recovery mechanisms, e.g., protection or restoration, to ensure uninterrupted operation even in the face of individual component failures. Such mechanisms typically require some form of redundancy in the network. As discussed above, network capacity is a precious good, and hence, redundant capacity is costly. Therefore, operators have to find a good balance between adding redundancy and risking service interruptions.

Several degrees of freedom exist for finding this balance. First, a plethora of recovery mechanisms have been introduced in the literature [Gro04]. Some are more resource-efficient than others, and also the implementation complexity varies. Hence, the network operator usually has some flexibility when selecting an appropriate recovery mechanism. Second, not all services have the same reliability requirements. For example, Internet services of residential customers are typically provided in a *best-effort* manner. They are not explicitly covered by recovery mechanisms and interruptions are tolerated by the operator. On the other hand, important customers, like large enterprise customers, data center (DC) operators, Internet service providers (ISPs), or other network operators, are offered recovery in various degrees and at the corresponding price. High-quality protection keeps interruption times below the threshold of 50 ms. Restoration mechanisms result in slightly longer interruptions.

The detailed reliability parameters are stipulated in a service level agreement (SLA) between the network operator and the customer buying the service [Xia08, Ch. 5]. Typical examples for such parameters are the minimum availability per month, the maximum cumulated downtime per month, or the maximum interruption time per outage. In addition to that, the SLA also contains a compensation policy, which specifies how the customer is compensated monetarily in case the stipulated reliability level is not met. Of course, compensation payments are not the only monetary aspect of SLA violation. Frequent and severe SLA violations can also affect the operator's reputation negatively, which may scare away potential customers. Nevertheless, the compensation policy is an important parameter because it bridges the gap between the technical and the economic perspectives on reliability to some extent.

With the compensation policy, the mentioned balance between redundancy and the risk of service interruptions can be interpreted on a monetary basis. Redundant network resources have to be bought, maintained, and operated. All of this comes at a cost. Furthermore, redundant resources are unused most of the time and do not generate revenue. On the other hand, violations of the reliability level specified in the SLA incur costs in the form of compensation as well. "Premium" operators typically invest a lot in redundancy to avoid SLA violation as much as possible. An alternative approach would be to tolerate a certain level of

SLA violation and the incurred compensation but save on investments for redundancy.

1.3 Research Questions and Contributions

As we have seen, a network operator has to constantly trade off network resources allocated to recovery against the resulting risk of SLA violation and the corresponding compensation. Often, this happens during service provisioning. In the following, we concentrate on end-to-end connections as one particular type of network service. When a connection is provisioned, the route through the network is determined, and also the recovery mechanism. Many provisioning schemes try to choose the route and the recovery mechanism such that the reliability guarantees in the SLA are fulfilled in (almost) every failure scenario. This typically entails a lot of costly redundancy and often overfulfills the SLA guarantees in the end.

In this context, we consider two main questions in this thesis. Both are concerned with the provisioning of connections:

1. How can the SLA overfulfillment be reduced and the network capacity increased?
2. How can the SLA compensation be integrated into the provisioning process?

The first question combines two aspects. By reducing the SLA overfulfillment, we expect to deploy less redundancy. Consequently, more network resources can be actively used to carry traffic and do not only serve as backup. The second question is related to the fact that typical provisioning schemes consider technical parameters only, e.g., the reliability level specified in the SLA. However, by integrating information about potential compensation payments into the provisioning process, we expect a better alignment between provisioning decisions and the operator's business models.

To answer the aforementioned two questions, this thesis provides the following three main contributions:

1. A mathematical model for the probability distribution of SLA compensation of a protected network connection.
2. A novel mechanism, named CAPSS, for dynamic connection provisioning that is compensation-aware and features surplus sharing to exploit SLA overfulfillment.
3. An evaluation of the model and the provisioning mechanism in a variety of transport network scenarios.

The model for the distribution of SLA compensation is the core of the mechanism. It describes the total compensation incurred for a connection during its contract period. It takes into account that a contract period comprises multiple billing cycles — a fact usually not considered in the literature. The proposed provisioning mechanism *compensation-aware provisioning with surplus sharing* (*CAPSS*) is built on top of the compensation model.

CAPSS has two goals: compensation-aware and resource-efficient provisioning. Compensation awareness is achieved by routing connections such that an operator-defined maximum compensation is not exceeded with a certain probability. This probability is defined by the operator as well. We assume that the operator can derive suitable values for both parameters from its business models. Resource efficiency is achieved by sharing SLA overfulfillment — we also call it *surplus* — among connections and relaxing their requirements. Sharing is a popular principle to achieve resource efficiency in networks. For example, many authors propose recovery mechanisms that share backup resources. As we will see throughout this thesis, our approach of sharing SLA surplus follows a slightly different philosophy.

A large part of this thesis is concerned with the evaluation of CAPSS in a variety of transport network scenarios. It will be shown that compensation-aware connection provisioning often behaves differently from conventional approaches. Furthermore, it will be shown that surplus sharing increases the network capacity significantly in many network scenarios. The increase typically ranges from 5 % to 10 %, depending on the exact scenario parameters. However, also much higher increases are observable, especially in networks that cover countries the size of Germany (23 % on average). The evaluation also shows that CAPSS meets the operator-defined probability of not exceeding a specific amount of compensation.

1.4 Outline

This thesis consists of six chapters, including this introduction.

Chapter 2 introduces fundamental concepts of transport networks that are necessary to understand and position this thesis. This includes today's network architectures, network outages, recovery mechanisms, and availability metrics. Furthermore, we discuss the relationship between a network operator and its customers and the role of SLAs. A focus of this chapter is the clarification of terminology we use throughout the thesis.

Chapter 3 puts the focus on dynamic connection provisioning that takes availability parameters in the SLA into account. CAPSS belongs to this class of provisioning mechanisms. After a precise problem statement, we present an extensive literature review on related models and mechanisms. We provide a categorization of those works, and we identify their shortcomings. Subsequently, we motivate our novel provisioning mechanism.

Chapter 4 finally introduces our provisioning mechanism CAPSS. We introduce the basic concept in two steps. The first step discusses compensation-awareness and leads to a mechanism called compensation-aware provisioning (CAP). In the second step, CAP is extended to CAPSS by adding the surplus sharing functionality. After this conceptual introduction, we derive the mathematical model for the probability distribution of SLA compensation. Subsequently, we introduce the algorithms that constitute CAPSS. We end the chapter with a discussion containing a categorization of CAPSS and remarks about its applicability.

Chapter 5 evaluates the functionality and performance of CAPSS in a variety of simulation studies. After a detailed introduction to the simulation methodology, we first evaluate the accuracy of the distribution model alone and CAPSS overall. Afterward, an illustrative example points out the main effects CAPSS has on the network. The subsequent section compares CAPSS with a conventional provisioning mechanism that is not compensation-aware and does not employ sharing of any kind. Subsequently, we evaluate the potential of the surplus sharing functionality by comparing CAPSS with CAP. We close the chapter with a summary and recommendations.

Chapter 6 summarizes the thesis and draws conclusions. Additionally, it gives an outlook on future research directions.

1.5 Notation

The mathematical notation in this thesis follows some conventions. We shortly introduce the conventions that apply globally. More specific conventions are introduced later on.

All variables are typeset in italics. In contrast, names or identifiers are typeset upright.

As an example, p is a variable and corresponds to a probability. In contrast, the "p" in Δt_p is upright and stands for "past." The variable Δt_p describes a time interval in the past. Euler's number is denoted by e, although it is not a variable but a constant.

A random variable (RV) is represented by an uppercase letter. For example, B is a RV for the cumulated downtime in a time interval. The realization of a RV is always lowercase, e.g., b. An exception to this convention are the RVs $\xi(t)$ and ξ_j in Section 4.4.1.1.

The variable t, possibly with an index, can describe both points in time and time intervals. However, some time intervals carry an additional Δ for clarity, e.g., Δt_p as mentioned above.

Tuples are surrounded by parentheses, e.g., (c_1, c_2, c_3). Closed interval endpoints are represented by square brackets, e.g., $[0, 1]$, while open interval endpoints are represented by parentheses, e.g., $(-\infty, \infty)$.

2

Transport Networks, Reliability, and Contracts

In this chapter, we introduce basic concepts and terminology of transport networks and related topics. We focus on areas that are relevant for this thesis. Therefore, not all details are provided for every topic. We first discuss fundamental terms of communication networks in Section 2.1. In Section 2.2, we introduce typical network architectures and technologies. Since the provisioning mechanism we propose in this thesis is related to availability, Sections 2.3 to 2.5 cover the topics network outages, recovery mechanisms, and different availability metrics. Lastly, Section 2.6 discusses network contracts and the relationship between network operator and customer.

2.1 Network Fundamentals

The fundamental purpose of a network is to enable communication between different endpoints. By saying that endpoints communicate, we mean that they exchange information or data. Those endpoints have long been humans — through telephones, computers, smartphones, etc. Nowadays, most communication happens between a human and a machine (e.g., video streaming) or two machines directly (e.g., Internet of things (IoT) or in data centers (DCs)) [Cis20]. Endpoints are attached to the network and use it for communication. However, we do not consider them a part of the network in the strict sense.

Nodes and links From an abstract perspective, the basic building blocks of a network are nodes and links. *Nodes* are responsible for receiving and forwarding or switching data from and to other nodes and the endpoints attached to the network. Nodes have complex internal structures. However, we mostly abstract from these internals and consider a node as a black box only. A node can only communicate with another node directly if both are connected by a *link*. As we will see in Section 2.2.2, links can be realized either physically or virtually. We

treat and visualize a network as a graph in which nodes correspond to vertices and links correspond to edges.

Paths and routes Data traverses a network along a sequence of nodes and links. This sequence is usually called a path or a route, interchangeably. However, in this thesis, we distinguish between a path and a route for clarity. We define a *path* as a linear sequence of nodes and links. That means a path can describe the actual trajectory data took on their way through the network. Also, a path can be the result of a shortest path algorithm like Dijkstra's algorithm. In contrast, we define a *route* as a combination of paths, e.g., a route between two nodes can consist of two parallel disjoint paths: a primary path and a backup path. The difference between a path and a route is important when we discuss protection mechanisms. If a pair of nodes can be connected on many different paths, we say there is a high *path diversity*. In contrast, if only one path exists between a pair of nodes, there is no path diversity.

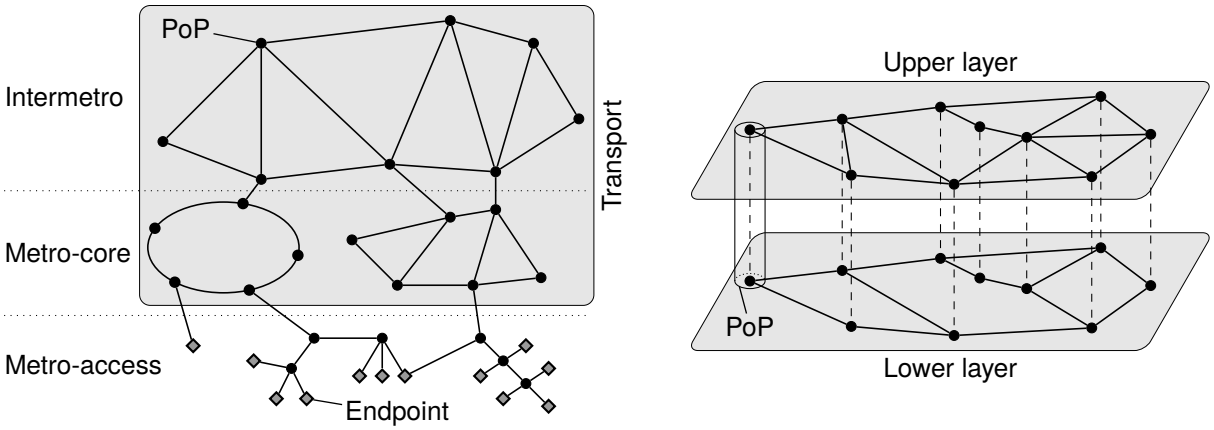
Connections An important task of the network is to ensure that data finds its way from the sending endpoint to the receiving endpoint. In modern networks, this can be achieved by two different methods [TW11, Sec. 1.3.3]. In the *connectionless* method, the network finds a suitable way to deliver the data for every single chunk of data the endpoints exchange. An example of a connectionless network protocol is the Internet protocol (IP). In the *connection-oriented* method, the network establishes an end-to-end *connection* between both endpoints once. The endpoints exchange data through this exclusive connection and close it once they have no more data to exchange. Examples are transmission control protocol (TCP) connections, multiprotocol label switching (MPLS) paths, or connections in optical networks, so-called *lightpaths* or *wavelengths*. A connection is always assigned a route. Therefore, we sometimes use the terms connection and route interchangeably, e.g., when we consider the availability of a connection or its route. An operator can also *reject* or *block* a connection request, e.g., if not enough network resources are available to establish the connection.

This thesis is mainly concerned with connection-oriented communication. Nevertheless, some parts can be applied to connectionless communication as well.

Traffic and capacity The entirety of data that is exchanged over the network is often called *data traffic* or *traffic* [Sim08, Sec. 1.6]. We define the amount of traffic the network can carry without being overloaded as the *network capacity*. The network capacity depends on different network parameters. For example, each node comprises a multitude of transmitter and receiver modules. The more modules, the higher the node's and thus also the network's capacity. If the network is short of capacity because the traffic grew over time, the capacity can typically be increased by upgrading existing modules to newer technologies or by installing additional modules. However, the capacity is also influenced by the way traffic is routed through the network and how much redundant network resources are reserved for the event of failures. This is where the contributions of this thesis come into play.

2.2 Network Architectures

Communication networks permeate all areas of our modern lives. To cope with their complexity, networks are usually organized in different hierarchies. Two such hierarchies are shown in Figure 2.1, namely three *network stages* in Figure 2.1a and two *network layers* in Figure 2.1b. The following explanations are based on [Muk+20, Sec. 11.1] and [Sim08, Ch. 1].



(a) Network stages

(b) Network layers – The dashed lines correspond to PoP-internal interlayer links.

Figure 2.1 Perspectives on the network architecture – Based on [Sim08, Sec. 1.2] and [Muk+20, Sec. 11.1.1]

2.2.1 Network Stages

Consider the network stages, also called segments or levels, in Figure 2.1a. Following [Muk+20, Sec. 11.1], we distinguish three stages, namely intermetro, metro-core, and metro-access. Network stages mainly differ in their geographical extent, the amount of traffic they carry, the employed technologies, and also in their function.

Communication endpoints, like private households, companies, or cellular base stations, are typically found in the *metro-access* stage. The metro-access stage collects and aggregates traffic from all connected endpoints and hands it over to the metro-core stage. In the opposite direction, it distributes traffic it receives from the metro-core stage and distributes it to the endpoints. A single metro-access network covers a few kilometers and up to hundreds of customers. The *metro-core* stage is responsible for transporting traffic between different metro-access networks. If the respective metro-access networks are directly connected to the same metro-core network, this transport happens inside the metro-core network itself. Otherwise, the traffic is again moved one stage up to the intermetro stage. Metro-core networks span up to hundreds of kilometers and transport traffic of thousands of endpoints. Finally, the *intermetro* stage connects different metro-core networks and is the highest stage. It covers up to thousands of kilometers, i.e., nations or even continents, and transports traffic of millions of endpoints.

While the function of the metro-access stage is the collection and distribution of traffic from and to endpoints, the metro-core and intermetro stages are only responsible for transporting the aggregated traffic. Therefore, the metro-core and intermetro networks are referred to as *transport networks*.

The stages also differ in their topologies and technologies. The metro-access stage usually forms tree topologies with the endpoints as leaves. Such topologies are not failure-resistant. Therefore, some endpoints can also connect to multiple metro-access networks. Typical technologies[1] are passive optical network (PON) over optical fiber, “cable” over coaxial cable, and digital subscriber line (DSL) over copper wires. As can be seen, the metro-access

[1]We list only the most prominent technologies for the different stages. However, we are aware that many legacy technologies are still present in today’s networks.

stage comprises different types of transmission media. In contrast, the transport stages contain only optical fiber today. Metro-core networks either form ring topologies or meshed topologies. Ring topologies have long been the standard together with technologies like synchronous digital hierarchy (SDH) and synchronous optical network (SONET). Nowadays, efforts are going in the direction of meshed topologies and technologies like metro Ethernet, multiprotocol label switching – transport profile (MPLS-TP), or optical transport network (OTN) and wavelength-division multiplexing (WDM). Intermetro networks are always meshed networks with a consolidated set of technologies, namely IP/MPLS, WDM, and often OTN.

2.2.2 Network Layers

The previous section mentions many networking technologies. In the metro-access stage, all mentioned technologies appear in parallel, i.e., each endpoint uses one of the technologies. In the transport stages, some of the technologies are used in parallel, but others are used on top of each other in a layered architecture. Figure 2.1b shows two such layers — an upper and a lower one — for illustrative purposes. Often, more than two layers are present, but the basic principle is the same: The upper layer uses the layer below to realize communication. To this end, nodes of the upper layer are attached to nodes of the lower layer by *interlayer links*. In the figure, those links are shown as dashed lines. Nodes connected in that way are usually located in the same building, called central office (CO) or, more abstractly, point of presence (PoP).

In the example, assume that the links of the lower layer correspond to optical fibers, and nodes comprise optical transmitters and responders, also known as *transponders*. Hence, the lower layer is capable of communicating via optical signals. The upper layer could then be an IP layer communicating with other IP nodes through the lower, optical layer.

Generally, a link in the upper layer is realized through the lower layer. Therefore, the links in the upper layers are "only" *virtual* links, whereas the links in the lowest layer are *physical* links, i.e., actual cables or fibers. Typically, upper layers provide more links than lower layers because a lower layer i can realize a virtual link of the upper layer $i + 1$ by communicating via intermediate nodes in its layer, i.e., over several contiguous links in layer i.

2.2.3 Packet-Optical Networks

As mentioned before, intermetro networks typically employ IP/MPLS, WDM, and, optionally, OTN. This results in a two- or three-layer architecture. WDM forms the lowest layer and provides enormous data rates per wavelength. To use those data rates efficiently, several connections with lower *sub-wavelength* data rates are *groomed*, i.e., bundled up. A popular choice for this task and for transporting various legacy technologies is OTN on top of the WDM layer. The top layer is formed by the packet-oriented IP/MPLS. This setup can be summarized by the term *packet-optical network* since the packet-oriented upper layer is transported over an optical bottom layer. Intermediate layers, like the OTN layer, can be present but are not considered explicitly in this abstraction. Even though the technologies in the metro-core stage differ slightly, the packet-optical abstraction still applies. Hence, the packet-optical architecture is representative of modern transport networks in general [Muk+20, Ch. 18.1].

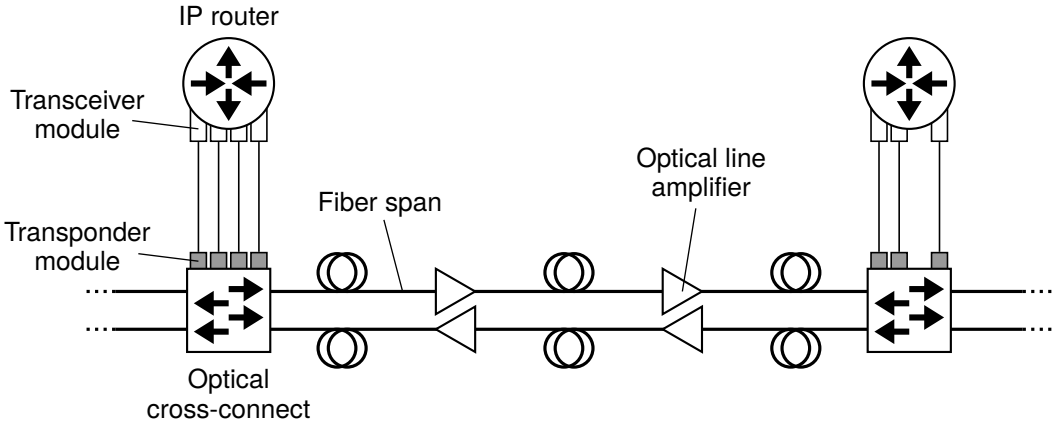


Figure 2.2 Simplified view of an IP over WDM network

To gain more insight into the components of a packet-optical network, Figure 2.2 shows a basic view of an IP over WDM system. The illustration is still very coarse but includes the most important parts and is sufficiently detailed for this thesis. The figure shows the IP layer at the top and the WDM layer at the bottom. Upper layers, like the IP layer, are sometimes referred to as *electrical* layers because the routing and switching is done electrically. In contrast, WDM switching can be done purely optical. An IP node consists of an IP *router* containing a number of *transceivers* (a combination of transmitter and receiver). A WDM node consists of an *optical cross-connect* capable of switching wavelengths either in the optical domain or by internally converting the optical signal to an electrical one, switching that to the desired output, and converting it back to an optical signal again. Attached to the optical cross-connect are transponders that convert short-reach signals used to communicate with the IP router — the interlayer links — to long-reach lightpaths. The combination of an IP router and an optical cross-connect corresponds to a PoP. Two optical cross-connects are physically connected by *fiber lines*. Technically, a fiber line consists of multiple *fiber spans* of around 80 km in length each [Muk+20, Ch. 4], joined by *optical line amplifiers*. The example shows two fiber lines, one for each communication direction. However, more fiber pairs can be deployed in parallel to increase the capacity. Also, an additional pair of transceiver and transponder modules can be installed in the router and cross-connect on the right. This increases the capacity as well.

2.3 Network Outages

Networks suffer from outages of different types and for various reasons. Most outages are the result of unplanned events like network component failures. However, maintenance events planned by the network operator can also lead to outage time, e.g., if software or hardware upgrades are required, and traffic cannot be redirected during that time.

Since an outage can be caused by a failure but also by a planned maintenance event, the terms *outage* and *failure* are not synonyms in a strict sense. Nevertheless, we mostly use both interchangeably in this thesis because we focus on failure-related outages.

2.3.1 Types of Failures and Their Causes

On a practical level, wired transport networks suffer from two fundamental types of failures, namely *cable cuts* and *device failures*. On an abstract level, either nodes or links can fail. Cable cuts correspond to link failures. In the case of a device failure, it depends on the particular device. For example, the failure of a complete router or a cross-connect corresponds to a node failure. On the other hand, a transponder or amplifier failure leads to a (partial) link failure.

Failures are often assumed to occur independently of each other. However, they can also occur simultaneously or cascaded, e.g., if the failure of one device triggers a software bug in another device, or if a broken link in the lower layer is used by multiple virtual links in the upper layer. For example, Markopoulou et al. [Mar+08] analyzed outages in the Sprint IP core network. They found that almost 30 % of all unplanned outages affect multiple IP links simultaneously. The remaining 70 % are independent single-link failures.

Buried cables are mostly damaged during digging activities. Aerial cables and their poles are much more exposed and are mostly damaged in storms, by vehicles, or by contact with power lines [Gro04, Sec. 3.1.2]. Device failures can be caused by power outages, software bugs, misconfiguration, or aging. The listed causes usually lead to independent failures. Simultaneous failures are often caused by natural disasters like earthquakes or floods, or can be the result of sabotage. For an in-depth discussion, consider [Muk+20, Sec. 11.2.1].

2.3.2 Characterization of Outage Behavior

If a network component or a connection, i.e., an entity, is operating as intended, we say it is *working* or *up*. Otherwise, if an outage is present, it is *down* or in the *failed state*. If the entity is down due to a failure, the operator has to take measures to eliminate the failure, e.g., splice a fiber, fix configuration parameters, reboot a device, or completely replace it. We refer to those measures by the single term *repair*. We assume that the entity is "as new" after it has been repaired.

We characterize the outage behavior of an entity by two durations. The first duration is the time it takes to eliminate the outage. For a planned outage, this corresponds to the time the operator takes to finish the maintenance. Otherwise, this corresponds to the repair time. Irrespective of this subtle difference, we refer to this duration as the *repair time*, *time to repair*, or *downtime*. The second duration is the time between the end of one outage and the occurrence of the next. We denote this duration as *time to failure* or *uptime*.

Both durations are random variables (RVs) that follow specific random distributions. Several authors have analyzed outage data of existing networks to identify the underlying distributions and approximate them with analytic ones. As mentioned above, Markopoulou et al. [Mar+08] analyzed the Sprint IP core network. They found that the Weibull distribution often approximates the time to failure well. González and Helvik [GH11] confirmed this for parts of the Norwegian academic IP network UNINETT. However, for long links, they found that the gamma distribution fits better. Uchida [Uch14] analyzed outages in Japan's networks. The author found that the time to failure is best modeled by an exponential distribution and the repair time by a Pareto distribution. Lourenço and Mello [LM12] compare exponentially and Weibull-distributed downtimes on a theoretical basis. They conclude that the type of distribution may impact the availability characteristics of a connection considerably. As can be seen, the most suitable distributions vary from network scenario to network scenario. Therefore, many authors assume exponential distributions for the sake of simplicity. However,

also Weibull, gamma, Pareto, or log-normal distributions are applied in the literature. More details will be discussed in Section 3.4. In particular, Tables 3.1 and 3.2 in Section 3.4.4 provide an overview of the distributions employed in failure models and network applications.

Regardless of the specific distributions, the mean durations, i.e., the *mean time to repair* (*MTTR*) and the *mean time to failure* (*MTTF*), are of importance in the following as well, e.g., to compute the availability.

2.4 Recovery Mechanisms

Once a failure occurs in the network, recovery mechanisms are responsible for steering traffic around the failure until it is repaired. In this context, we call the original path the traffic used until the failure occurred the *working path* or *primary path*. The alternative path assigned by the recovery mechanism is the *recovery path* or *backup path*. As mentioned before, we denote the combination of primary path and backup path as route.

In the following, we list important properties that help categorize recovery mechanisms. Subsequently, we describe important recovery mechanisms of different network layers and technologies. For further discussion, we refer to [VPD04; Muk+20]. Chołda et al. [Cho+07b] provide an excellent overview of recovery mechanisms. Zhang and Mukherjee [ZM04] review recovery in the WDM layer. Tipper [Tip14] concentrates on multi-layer aspects.

2.4.1 Basic Properties of Recovery Mechanisms

Recovery mechanisms can be categorized by several basic, technology-independent properties or modes (see, e.g., [VPD04, Sec. 1.5] and [Muk06, Sec. 11.4.1]). The most important properties for positioning this thesis are

- protection vs. restoration
- path vs. segment recovery
- dedicated vs. shared backup resources.

The exact definitions vary from author to author. Therefore, we provide the definitions we use throughout this thesis in the following.

Protection vs. restoration *Protection* means a preplanned recovery path is signaled to all network components potentially involved in a failure and its recovery *prior* to the failure. For example, when a new connection is provisioned, and a route is determined, this route already contains a backup path. Once a failure occurs, the recovery path is already established and can be used quickly. In the case of *restoration*, the recovery path is only signaled, i.e., established, once a failure occurs. This is slower but also provides more flexibility. Restoration can still employ preplanned recovery paths. However, recovery paths that are dynamically computed in the event of failure are more common.

Path vs. segment recovery *Path recovery* changes the whole affected path to an alternative end-to-end backup path. The backup path is often node- or at least link-disjoint to the affected path. However, failures usually affect some but not all network components of a path. Therefore, activating backup resources around these failed components only is often sufficient. We call this *segment recovery*. Segment recovery allows finer control over the backup resources and can be faster since only a segment of the primary path must be changed. In the extreme case, segment recovery only protects a single link or node. Segment recovery is also called *local recovery*. Path recovery is also called *global recovery*.

Dedicated vs. shared backup resources A *dedicated backup resource* is assigned exclusively to a specific working resource. In contrast, a *shared backup resource* is assigned to *multiple* working resources. If two or more working resources fail simultaneously, not all of them can be fully recovered. Also, shared backup is more complex to plan and evaluate than dedicated backup. On the other hand, shared backup is more resource-efficient.

2.4.2 Common Recovery Mechanisms

In this section, we list important recovery mechanisms implemented in today's technologies. We focus on mechanisms that recover path- or connection-like entities individually. Therefore, we omit mechanisms like generalized multiprotocol label switching (GMPLS) span recovery [FB06, Sec. 7.5], which recovers all connections traversing a span, e.g., a link, simultaneously. We also omit mechanisms for ring recovery since SONET/SDH ring networks are considered legacy.

Two very common protection types are called *1+1* and *1:1*. A general definition is given in [G.808.1]. Both *1+1* and *1:1* correspond to dedicated protection, i.e., exclusive backup resources are reserved. For 1+1, the backup resources carry a copy of the traffic on the primary resources all the time. The receiving node can autonomously switch to the backup copy if primary traffic is erroneous or missing. In contrast, 1:1 puts the traffic on the backup resources only in case of failure. Otherwise, the backup resources can carry other traffic. 1+1 and 1:1 protection are implemented in many technologies, like SONET/SDH [G.841], OTN [G.873.1], or GMPLS [RFC4872]. SONET/SDH and OTN allow the protection of an end-to-end path in the *trail protection* mode and the protection of a segment in the *subnetwork connection protection* (*SNCP*) mode. GMPLS defines segment recovery in [RFC4873].

The recommendations [G.841] and [G.873.1] also define the *1:n* mode for shared path protection for SONET/SDH and OTN. In this mode, one backup path is reserved for n distinct working paths connecting the same source and destination nodes. Another shared protection mode is called *shared mesh protection*. It allows sharing of backup resources among working paths that connect different nodes. It is defined in [G.808.3; G.873.3] for OTN. GMPLS defines a shared mesh *restoration* mode with preplanned recovery paths in [RFC4872].

Consider now the higher network layers. MPLS implements three recovery mechanisms, most importantly MPLS *fast reroute* (*FRR*) [RFC4090]. FRR is a segment protection mode in which the node directly in front of the failed node or link redirects the traffic to the backup path. In addition to that, there is also MPLS *path protection*, which is 1:1 path protection, and MPLS *global default restoration*, which realizes restoration of end-to-end paths. On the IP layer, the normal interior gateway protocol (IGP) reconvergence can be considered a restoration mechanism.

All recovery mechanisms listed above are single-layer mechanisms for a certain technology or protocol. However, as explained in Section 2.2.2, transport networks consist of multiple layers. Therefore, recovery also can be realized on multiple layers if desired. The simplest case is still recovery on a single layer. The most complex case is joint recovery over multiple layers under the assumption that a single network controller or control plane can coordinate recovery over all involved layers. Some examples can be found in [REB21; GH14; Gun+12]. The middle course is one recovery mechanism per layer with parameters finely tuned between the different mechanisms to avoid instabilities and race conditions. According to [Muk+20, Sec. 11.2], this middle course is the typical configuration today.

2.4.3 Segment Protection

We employ two protection mechanisms in this thesis, namely dedicated path protection and dedicated segment protection. Since both recovery modes are protection mechanisms, the backup resources are assigned when the connection is provisioned and do not change later on. The segment protection is inspired by [KMO08b]. However, our implementation only covers one segment of the primary path. The size of the segment is selected such that the protected route reaches a specific figure of merit. For example, assume that the route for a connection must provide a specific steady-state availability (see Section 2.5.1).

In our implementation, the primary path of the route is determined first. If the primary path alone provides the required figure of merit, no backup resources are added. Otherwise, the algorithm iteratively adds backup resources until the figure is reached or the whole primary path is protected, as in our dedicated path protection mechanism.

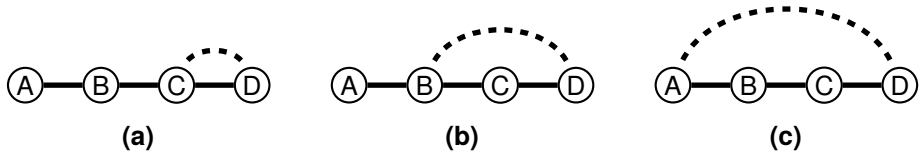


Figure 2.3 Iterations of the segment protection mechanism

Figure 2.3 shows an example. The route connects nodes A and D with a primary path via nodes B and C. In the first step, in Figure 2.3a, only the last link of the primary path between nodes C and D is protected. The backup is indicated by the dashed line. This dashed line represents a proper backup path, i.e., it generally consists of multiple nodes and links. If this first step is insufficient (e.g., in terms of the required steady-state availability), the penultimate link is protected as well, as shown in Figure 2.3b. This procedure is repeated until the required figure of merit is met or the primary path is fully path-protected, as shown in Figure 2.3c. A detailed algorithmic description is provided In Section 4.5.1.

2.5 Availability of Network Connections

2.5.1 General Availability Metrics

As discussed in Section 2.3, network components and, thus, also network connections are subject to failures and repair. At a specific point in time, a connection can be in one of two states: *working* or *failed*. In this context, the *availability* is an important concept. Intuitively, it describes the probability of finding a connection in the working state [Gro04, Sec. 3.12]. However, this is not the complete definition. According to [Lew95, Sec. 10.3], there exist different types of availability metrics that differ slightly in their definition and meaning.

Point availability As the basic building block, the *point availability* $a(t)$ is the probability of finding a connection in the working state *at a specific point in time* t, i.e.,

$$a(t) = \mathbf{P}(\text{“connection works at } t\text{”}). \tag{2.1}$$

Interval availability The average point availability for a time interval Δt is the *interval availability* $a^*(\Delta t)$. It is defined as

$$a^*(\Delta t) = \frac{1}{\Delta t} \int_0^{\Delta t} a(t)\,\mathrm{d}t. \tag{2.2}$$

The integral equals the total uptime in Δt. Hence, a more intuitive definition, given in [Gro04, Sec. 3.12], is

$$a^*(\Delta t) = \frac{\text{uptime in } \Delta t}{\Delta t}. \tag{2.3}$$

Since the connection can only be in the working or the failed state, the time interval Δt can be decomposed into the sum of the total uptime and downtime in Δt, i.e.,

$$a^*(\Delta t) = \frac{\text{uptime in } \Delta t}{(\text{uptime in } \Delta t) + (\text{downtime in } \Delta t)}. \tag{2.4}$$

Steady-state availability For long time intervals, the interval availability converges to the so-called *asymptotic* or *steady-state availability* a, defined as

$$a = \lim_{\Delta t \to \infty} a^*(\Delta t) \tag{2.5}$$

$$= \lim_{\Delta t \to \infty} \frac{1}{\Delta t} \int_0^{\Delta t} a(t)\,\mathrm{d}t \tag{2.6}$$

or, equivalently,

$$a = \lim_{\Delta t \to \infty} \frac{\text{uptime in } \Delta t}{\Delta t} \tag{2.7}$$

$$= \lim_{\Delta t \to \infty} \frac{\text{uptime in } \Delta t}{(\text{uptime in } \Delta t) + (\text{downtime in } \Delta t)}. \tag{2.8}$$

That means a connection's steady-state availability is the ratio of its total working time to the whole observed time interval when this time interval approaches infinity.

The individual up- and downtimes the connection experiences are RVs and, thus, of variable duration. However, given that the time interval approaches infinity and assuming that the failure and repair processes are stationary, we can assume that the up- and downtimes are well summarized by their corresponding means, namely the MTTF and the MTTR. By normalizing (2.8) with the number of failure events in Δt, $n(\Delta t)$, we obtain

$$a = \lim_{\Delta t \to \infty} \frac{\frac{\text{uptime in } \Delta t}{n(\Delta t)}}{\frac{\text{uptime in } \Delta t}{n(\Delta t)} + \frac{\text{downtime in } \Delta t}{n(\Delta t)}}. \tag{2.9}$$

From this, we get the well-known equation for the steady-state availability

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

which is also given in [Gro04, Sec. 3.12].

Experienced availability The interval availability, $a^*(\Delta t)$, is the average probability that the connection is operational at any point in time during Δt. Later on, we are interested in another related availability metric, which we call the *experienced availability*, following [XCW09; ZG05]. The experienced availability is a RV we denote by A. It describes the availability a specific connection *actually experienced* over a specific time interval, like a month or the whole contract period. That means it is a value that is measured during network operation.

Reliability The various types of availability introduced above apply to systems that get repaired once they fail and are fully operational again afterward. In contrast, if only the time until a failure occurs is of interest, e.g., because repair is not even possible, then the system is better described by the *reliability*. The reliability is the probability that a system does not fail during the time interval $[0, t]$ or, equivalently, that the failure only occurs at some point *after* t [Lew95, Sec. 6.2]. Systems with a high MTTF usually also have a high reliability. The reliability is only relevant for small parts of the discussion of related work in Chapter 3.

Terminology As we have seen, the concept of availability covers different metrics. The one most frequently used in the literature is the steady-state availability. Therefore, whenever we mention *availability* in the following, we either implicitly refer to the steady-state availability or we use the word *availability* as a qualitative description only. Furthermore, we use the word *reliable* almost exclusively to compare availability levels qualitatively. For example, "more reliable networks" means "networks with a higher availability." The strict definition of reliability is only rarely used in this thesis.

2.5.2 Availability for Exponential Failures and Repair

As mentioned before, the failure and repair processes of network connections are often assumed to follow exponential distributions. In this case, the MTTF is the inverse of the failure rate λ, and the MTTR is the inverse of the repair rate μ. Therefore, based on (2.10), the steady-state availability in the exponential case is

$$a = \frac{\mu}{\lambda + \mu}. \tag{2.11}$$

According to [Bir17, Ex. 6.1], the point availability under the additional assumption that the connection works at $t = 0$ is given by

$$a(t) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{\lambda + \mu} e^{-(\lambda+\mu)t}. \tag{2.12}$$

Since the connection is assumed to be working at $t = 0$, we have $a(0) = 1$.

Figure 2.4 shows the behavior of the point availability and the interval availability for $MTTR = 9\,\mathrm{h}$ ($\mu = 0.111\,\mathrm{h}^{-1}$), $a = 0.99955$, and, consequently, $MTTF = 19\,991\,\mathrm{h}$ ($\lambda = 5 \cdot 10^{-5}\,\mathrm{h}^{-1}$). It can be seen that the interval availability converges to the steady-state availability when the considered time interval, Δt, grows. Also, the point availability converges to the steady-state availability. The convergence of the point availability is exponential, with a time constant of $\tau = 1/(\lambda + \mu) = 8.996\,\mathrm{h}$ [Bir17, Sec. 6.2.4]. After a time span of $-\log(0.01) \cdot \tau \approx 5\tau$, the difference between $a(t)$ and a is less than 1 % of the original difference $a(0) - a$ at $t = 0$, i.e., $a(t)$ has practically converged. Typically, $\mu \gg \lambda$ and, hence, $\tau \approx 1/\mu = MTTR$. So, the point availability in the exponential case has converged after around $5 \cdot MTTR$.

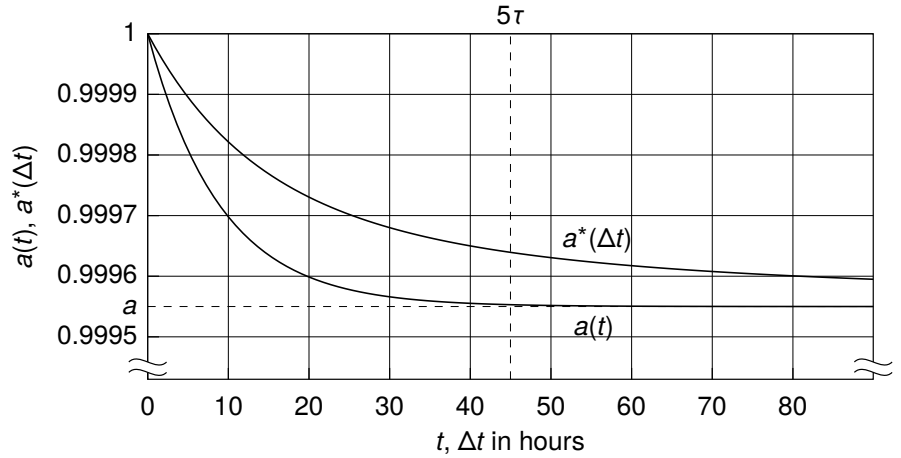
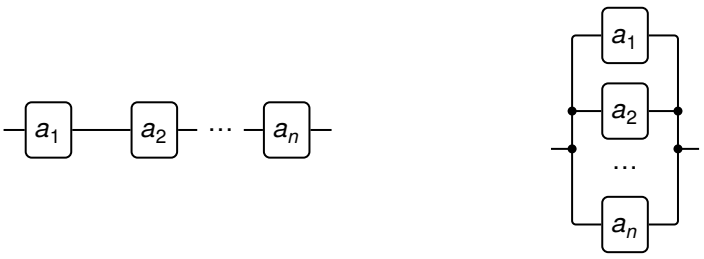


Figure 2.4 Behavior of point and interval availability for exponential failures and repair – $MTTR$ = 9 h, $a = 0.99955$, $MTTF$ = 19 991 h

2.5.3 Steady-State Availability of Composed Systems

Often, a system, e.g., a network connection, consists of several components in series or in parallel. Assume that each component can be assigned a steady-state availability. Then, availability-wise, the system can be transformed into an equivalent single-component system by (repeated) series or parallel reduction steps.



(a) System components in series **(b)** System components in parallel

Figure 2.5 Reliability block diagrams of fundamental system configurations

Figure 2.5a shows the *reliability block diagram* of a series system. Figure 2.5b shows a parallel system. Both systems consist of n components. Each component i has a certain steady-state availability a_i. According to [Gro04, Sec. 3.12], the steady-state availability of the equivalent single-component system in the *series* case is

$$a_{\text{series}} = \prod_{i=1}^{n} a_i. \tag{2.13}$$

In the parallel case, it is

$$a_{\text{parallel}} = 1 - \prod_{i=1}^{n} (1 - a_i). \tag{2.14}$$

More complex systems can be transformed by repeated series and parallel reduction steps [Gro04, Sec. 3.12.4].

2.6 Contracts and Economic Aspects of Network Connections

The topics introduced so far are of technical nature only. However, the provisioning mechanism we propose in this thesis aims to bring technical and economic aspects closer together. Therefore, we turn our attention to economic and contractual topics in this section.

2.6.1 Customers and Contracts

A network operator sells network connections to its customers. Throughout this thesis, we assume a single network operator that has multiple customers. Each customer can buy multiple independent connections, also between the same source and destination nodes. Since our focus is on transport networks, the customers are not private households but, e.g., companies, DC operators, other network operators, or Internet service providers (ISPs). A contract for a connection contains, among other things, the contract period, the monthly fee for the connection, i.e., the *monthly recurring charge* (*MRC*), and performance specifications. Some authors also refer to the contract period as *holding time*. Before we discuss performance specifications in more detail, we consider the life cycle of a contract and the corresponding connection.

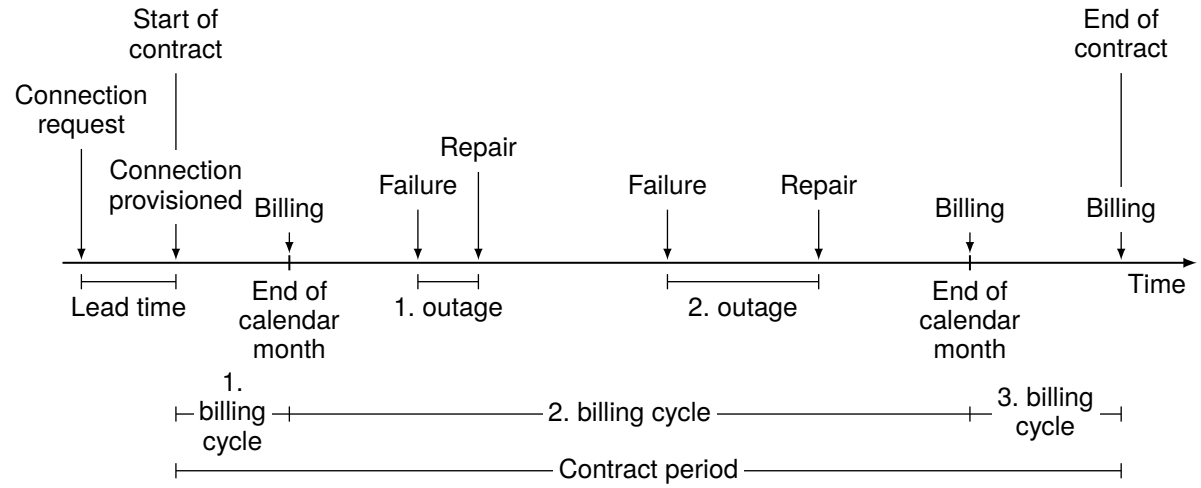


Figure 2.6 Life cycle of a network connection and its contract

Figure 2.6 shows an example life cycle. With the help of the figure, we introduce some more terminology. Some aspects are called and handled slightly differently from operator to operator. Nevertheless, we stick to the conventions we introduce here throughout this thesis.

When the customer orders a connection, the network operator receives a *connection request*. The operator then starts to plan and provision the connection. When the connection has been provisioned, the contract between the operator and the customer starts. The time between the reception of the connection request and the successful provisioning of the connection is called *lead time*. While the contract is active, the customer is billed regularly at the end of a calendar month. If the contract's start does not coincide with the end of a calendar month, the first billing cycle of the connection is shorter than a month. The same is true for the last billing cycle (the third one in Figure 2.6) if the end of the contract does not coincide with the end of a calendar month. With *billing*, we mean that the customer is charged the MRC for the connection and is credited the accrued compensation (see Section 2.6.2.2 below). If the end of the contract does not coincide with the end of a calendar

month, the customer is billed one last time when the contract ends. When the contract has ended, the network operator shuts down the connection and frees the corresponding network resources. We say the connection *departs* or *leaves the network*. In addition to those "regular" events, also failures can happen. The example shows two outages in the second billing cycle.

2.6.2 Service Level Agreements

Network contracts between operators or an operator and a company typically stipulate precise performance specifications for a connection, as well as compensation schemes for the case of their violation. The specifications — also called service level objectives (SLOs) — are contained in a service level agreement (SLA), which forms a part of the contract. In the following, we introduce typical performance parameters and compensation schemes found in existing SLAs. After a general introduction, we focus on availability specifications.

2.6.2.1 Performance Specifications

The following list contains typical performance parameters specified in an SLA:

- minimum monthly availability
- type of recovery mechanism
- maximum downtime per outage
- maximum end-to-end delay or latency
- maximum jitter
- minimum packet delivery ratio.

Note that the list is not exhaustive, and not all the parameters apply to all types of connections. The list is based on the SLA shown in [Xia08, Ch. 5] and the SLAs listed in Table 2.1.

Table 2.1 Evaluated SLAs with availability-related compensation policy properties

Reference	Company	Service type	Policy	Highest compensation (of *MRC*)
[Cen11]	CenturyLink	Wavelength	Staircase	50 %
[Glo11]	Global Crossing	Wavelength	Staircase	100 %
[Lum23]	Lumen Technologies	Various	Staircase	100 %
[Ora12]	Orange	Ethernet	Linear	100 %
[TEL23]	TELUS	Wavelength	Staircase	40 %
[Ver20]	Verizon	Wavelength	Staircase	100 %
[Win19]	Windstream	MPLS	Linear	15 %

Some SLAs specify the type of recovery mechanism, e.g., [Ver20]. Others only specify a minimum availability level. In the latter case, the operator is free to select appropriate recovery mechanisms to provide the specified availability. Most other parameters are formulated as guarantees, i.e., either a minimum or maximum level, that must be provided by the network operator. If the operator fails to provide the guaranteed level for a connection, we talk about *SLA violation* or *SLA underfulfillment*. If the operator provides a performance

that matches the guaranteed level or is even better, we talk about *SLA compliance* or *SLA fulfillment*. If the provided performance is strictly better than the requirement, we talk about *SLA overfulfillment*. Considering n different connections, we call the ratio of the number of fulfilling connections to all n connections the *SLA fulfillment ratio*.

2.6.2.2 Compensation

If the operator violates the SLA, the customer receives a *compensation*. Compensation usually means the customer's account is credited a certain amount or a fraction of the MRC is waived. Regardless of the actual mechanism in place, we simply talk about the *amount of compensation* or *compensation*. Usually, it is the customer's responsibility to open a trouble ticket once a performance parameter is violated. Without opening a ticket, the customer is not eligible for compensation (see, e.g., [Ver20; Win19]).

The amount of compensation is defined in the SLA and typically depends on the severity of the violation. Following [MN11b], we denote the relationship between severity and amount of compensation as *compensation policy*.

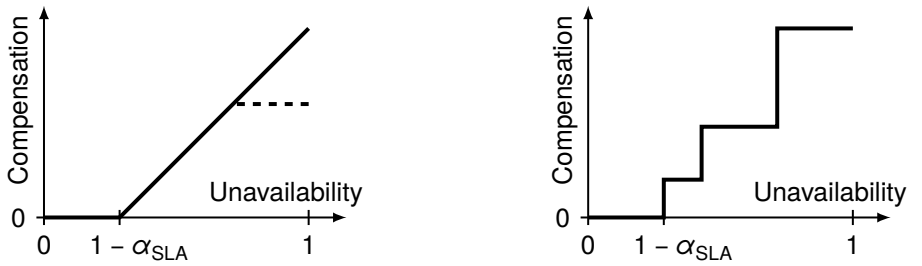
In the following, we focus on the *monthly availability* and do not consider other parameters in more detail. We denote the monthly availability level guaranteed in the SLA as

$$\alpha_{\mathrm{SLA}} \in [0, 1]. \tag{2.15}$$

We include the case of $\alpha_{\mathrm{SLA}} = 1$ because a few SLAs guarantee 100 % availability. However, in the vast majority of SLAs, we have $\alpha_{\mathrm{SLA}} < 1$. If we compare two SLAs, we call the SLA with the higher availability guarantee the *stricter* one.

Let A denote the experienced availability (see Section 2.5.1) at the end of a month, and $U = 1 - A$ the experienced unavailability. In the context of availability, we define *SLA violation* as $A < \alpha_{\mathrm{SLA}}$, *SLA compliance* or *fulfillment* as $A \geq \alpha_{\mathrm{SLA}}$, and *SLA overfulfillment* as $A > \alpha_{\mathrm{SLA}}$. The lower A or the higher U, respectively, the more severe the SLA violation is.

Technically, most operators only consider a connection unavailable once the customer opens a trouble ticket (e.g., [Lum23; Glo11]). Therefore, the monthly unavailability acknowledged by the operator can be shorter than the experienced unavailability, i.e., the actual failure-induced unavailability of the connection. However, we assume in this thesis that the difference is negligible, and both operator and customer assume the same unavailability.



(a) Linear compensation policy **(b)** Staircase compensation policy

Figure 2.7 Typical types of compensation policies for the monthly availability

The SLAs we evaluated employ two types of compensation policies, namely a linear function or a staircase function of the monthly unavailability. The basic shapes are shown in Figure 2.7. In both cases, a monthly unavailability of less than $1 - \alpha_{\mathrm{SLA}}$ does not lead to

compensation. For the linear policy type depicted in Figure 2.7a, any higher unavailability is linearly mapped to the amount of compensation. In the SLAs in [Xia08; Ora12; Win19], one hour of cumulative monthly downtime leads to a compensation of a thirtieth of the MRC, i.e., the share of the MRC that corresponds to one day. In [Ora12], the maximum amount of compensation is limited to 100 % of the MRC. Such a cap is indicated by the dashed line in Figure 2.7a. The other SLAs employ staircase-shaped policy functions with distinct compensation and unavailability levels, as depicted in Figure 2.7b. Table 2.1 shows the highest monthly compensation level for each of the SLAs.

Based on our investigations, the staircase policy is the most popular one in transport network SLAs. Therefore, this thesis concentrates on this particular type of compensation policy. SLAs usually define a plethora of additional clauses, e.g., to exclude the effects of force majeure events or to limit the compensation in case multiple performance parameters are violated. We do not consider such additional clauses in this thesis but restrict the calculation of compensation to the evaluation of the policy function.

3

Availability-Aware Dynamic Connection Provisioning

This chapter formulates the problem we solve in this thesis and discusses related work and its shortcomings. We identify three categories of availability-aware provisioning mechanisms and three major shortcomings of existing mechanisms. The structure of this chapter is based very much on those categories and shortcomings. First, Section 3.1 states the problem and briefly introduces the three categories of provisioning mechanisms. Section 3.2 discusses the first category of provisioning mechanisms, which we call conventional mechanisms. In Section 3.3, we derive three important shortcomings of those conventional mechanisms. Section 3.4 introduces related work that tackles those shortcomings. It includes provisioning mechanisms of all three categories but also the underlying mathematical models. Lastly, Section 3.5 motivates the development of our novel provisioning mechanism.

3.1 Problem Statement

Connections in transport networks are not static. Over time, customer requests for new connections arrive, and existing connections leave the network when their contracts end. Connection requests must be provisioned within relatively short time frames and cannot be postponed to regular network planning and upgrade cycles of several years [Muk06, Sec. 1.6].

To provision connection requests in a timely manner, the network operator has to employ a mechanism that we will refer to as a *dynamic provisioning mechanism*. There are other names for such mechanisms, e.g., in the context of optical networks, Mukherjee [Muk06, Sec. 7.3.2] calls it *dynamic lightpath establishment*. Simmons [Sim08, Sec. 1.7], more abstractly, refers to it as *real-time network planning*.

As depicted in Figure 3.1, a dynamic provisioning mechanism determines a suitable route in the network. It takes the available network resources and their occupancy by

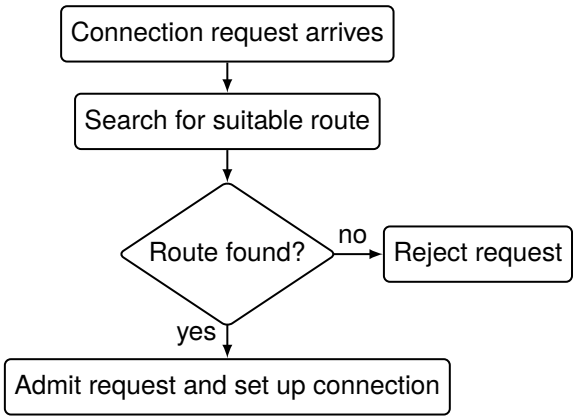


Figure 3.1 Basic structure of a dynamic provisioning mechanism

existing connections into account. Subsequently, it admits the connection to the network and initiates its deployment. If no route is found, e.g., because the network is short of capacity or connection requirements cannot be fulfilled, the connection request is rejected. A typical goal for a dynamic provisioning mechanism is resource efficiency, i.e., it should use a minimal amount of network resources to realize the connection. However, this is usually not the only goal, and compromises must be made.

A dynamic provisioning mechanism can also be extended to make decisions about accepting or rejecting a request independent of the presence of a route but based on additional criteria. We call this an *admission control mechanism*. For example, in [Das12; ZS15; Raz+19], connection requests are rejected if their estimated profit is too low, even if a route exists that is suitable in terms of capacity and availability requirements. In contrast, the mechanism in [Xia+09b] admits connections in a best-effort manner even if the required backup resources are *not* available at the time of provisioning. If backup resources become available later in time, they are added to the existing connections in a reprovisioning step. The provisioning mechanism developed in this thesis does not include such speculative functionality. Instead, whenever a suitable route exists, the connection is accepted. If no suitable route exists, the connection request is rejected.

If the connection request entails a service level agreement (SLA), the provisioning mechanism has to take the SLA specifications into account. If the mechanism considers SLA specifications related to the connection *availability*, we call it an *availability-aware dynamic provisioning mechanism*. The term subsumes all of the following mechanisms we discuss.

This thesis proposes a dynamic provisioning mechanism that considers the SLA compensation incurred when the SLA-guaranteed availability is not provided. Since the compensation policy is related to the SLA availability, the mechanism is an availability-aware dynamic provisioning mechanism. Its overall goal is to increase the network capacity while adjusting the amount of SLA compensation the operator has to pay its customer.

As we will see in the next sections, various availability-aware mechanisms exist in the literature. Nafarieh et al. [NRR15] present an initial overview of such mechanisms. We distinguish three categories of availability-aware provisioning mechanisms, namely mechanisms based on

1. the steady-state availability of a route
2. the probability of SLA violation
3. the SLA compensation.

Mechanisms of the first category only consider the availability guaranteed in the SLA, α_{SLA}, and find a route whose steady-state availability, a, satisfies this guaranteed availability. This is the most common approach. However, the steady-state availability assumes long observation intervals and cannot describe the probability of violating the guaranteed availability in a specific billing cycle properly. Therefore, in the second category, more sophisticated mechanisms take the probability of SLA violation into account. Some employ distribution-based models to compute the probability of SLA violation, $\mathbf{P}(A < \alpha_{\text{SLA}})$. Others use heuristic approaches to approximate the violation probability. In both cases, connections are usually provisioned such that the violation probability is minimized. Some mechanisms also involve reprovisioning of connections. Finally, some mechanisms even consider the amount of compensation that arises from SLA violations. For example, they balance the compensation and the revenue a connection generates and maximize the resulting profit. In the following sections, we will discuss existing mechanisms and models of the different categories.

3.2 Conventional Provisioning Mechanisms

In this section, we summarize a selection of publications that fit into the first, most basic category of provisioning mechanisms. That means they consider the SLA availability guarantee and the steady-state availability. We denote these mechanisms as *conventional* mechanisms because they are well established in the literature and, presumably, also used by network operators. In subsequent sections, we will discuss their shortcomings and more sophisticated mechanisms of the other categories that tackle those shortcomings.

As one of the first, Zhang et al. [Zha+03a] describe a heuristic algorithm that selects a route with suitable availability from a pre-computed set of protected or unprotected candidate routes, S. All routes in S have a steady-state availability larger than the availability guaranteed in the connection's SLA. Which of the routes in S is selected depends on the operator's strategy. It can be the *most reliable* route, the *minimal-cost* route, or the so-called *just-above-threshold* route. While the *most reliable* route is the one in S with the highest steady-state availability, the *just-above-threshold* route is the route in S with the lowest steady-state availability. Hence, the latter selection strategy provides a theoretical connection availability that is close to the availability level guaranteed in the SLA. In [Zha+03b], the same authors also present an integer linear program (ILP)-based provisioning mechanism that minimizes resource usage and ensures that connection steady-state availabilities satisfy their SLA requirements. A more detailed presentation of both mechanisms can be found in [Zha+07].

While the methods of Zhang et al. are designed for static connection requests, Huang et al. [Hua+04] present a dynamic connection provisioning mechanism in which the source node dynamically searches for a route whenever a connection request arrives. It first searches for an unprotected route. If the route's steady-state availability fulfills the SLA requirement, it is selected, and the connection request is accepted. Otherwise, a backup path is added, and the resulting steady-state availability is re-evaluated. If the availability is still too low, the request is rejected. The steady-state availability of a route is estimated by series and parallel reductions of individual network component availabilities, similar to the approach introduced in Section 2.5.3. A similar mechanism is introduced in [SZM05] and in [SZM07].

The routing procedure in [Mel+05] is similar to the aforementioned. However, the authors model the failure and repair processes of the network components as a continuous-time Markov chain and compute the steady-state availability of a connection with a matrix-based approach.

While the previous works all support dedicated or shared path protection, Kantarci et al. [KMO08b] present an availability-aware provisioning algorithm for segment protection. Their *availability-constrained general segment protection* (*AC-GSP*) algorithm first selects three potential working paths and their corresponding segment backups. Then, the steady-state availability of each of the three combinations is calculated. Out of those combinations that fulfill the requested SLA availability, the one with the highest availability is chosen to route the connection on.

The mechanisms discussed so far target optical networks only. Yao and Ramamurthy [YR04] and Tornatore et al. [Tor+12] introduce availability-aware provisioning for multi-layer networks that support sub-wavelength requests and traffic grooming.

The presented publications are a representative part of the existing literature. It can be seen that the mechanisms are very similar in their basic routing principle: Connection requests are only accepted if a route with a steady-state availability larger than or equal to the availability value stipulated in the SLA is found. Furthermore, in most of the mechanisms, the level of deployed protection is kept as low as possible to save resources, i.e., if an unprotected route's steady-state availability fulfills the SLA, no protection is added, and shared protection is preferred over dedicated protection. The different works target different use cases or network scenarios. Also, they use different methods to estimate a route's steady-state availability. However, the basic routing principle is always the same. Despite their popularity in the literature, the presented mechanisms have their shortcomings. The next section discusses them in detail.

3.3 Shortcomings of Conventional Mechanisms

This section identifies and discusses three major shortcomings of conventional availability-aware provisioning mechanisms:

1. They do not consider the probability of SLA violations.
2. They do not incorporate economic aspects into the provisioning process directly.
3. They result in overfulfillment of the SLA-guaranteed availability or, more generally, an availability-related, operator-defined figure of merit.

3.3.1 Probability of SLA Violation

As introduced in Section 2.6.2, SLAs typically define an availability level against which the experienced connection availability is compared at the end of a billing cycle — usually one month. If the experienced connection availability is below the SLA level, the network operator must compensate the customer. The amount of compensation is determined by the compensation policy in the SLA.

As shown in the previous section, many conventional provisioning mechanisms base their routing decision on the steady-state availability. They route a connection such that the steady-state availability of the route is larger than or equal to the availability level stipulated in the SLA. With this approach, the connection satisfies the SLA guarantee in the majority of billing cycles. Often, the experienced connection availability will even be higher than the required availability, e.g., if no outage occurred at all. However, the experienced connection availability is the result of a stochastic process of random connection up- and downtimes. Therefore, even though the route's steady-state availability is larger than the SLA guarantee,

billing cycles will occur in which the experienced availability is worse than the guaranteed level. Consequently, SLA violation due to a missed availability guarantee cannot be prevented entirely. Therefore, there will always be a non-zero probability that the SLA availability is violated. We call this probability *SLA violation probability* and denote it by $\mathbf{P}(A < \alpha_{\text{SLA}})$. Unfortunately, an analysis or routing based on the steady-state availability alone cannot capture this probability appropriately.

Zhou and Grover [ZG05] were among the first to study this issue in more detail in the context of communication networks. In simulation studies, they evaluate the unavailability connections experience over their whole contract period. The simulated periods reach up to 20 years. They show that the experienced unavailability is indeed different from the route's steady-state unavailability. For most simulated contract periods, the 0.95- and 0.99-quantiles of the experienced connection unavailability are much worse than the route's steady-state unavailability. That means that more than 5 % (or 1 % when the 0.99-quantile is considered) of all connections violate their SLA. Therefore, if the operator guarantees an availability in the SLA that corresponds to the route's steady-state availability, the probability of SLA violation can be quite high. To reduce the violation probability, the authors recommend the introduction of a so-called *safety margin*. The safety margin is a difference between the unavailability guaranteed in the SLA and the steady-state unavailability of the route. They emphasize that the safety margin should grow when the contract period decreases. For *very* short contract periods, their results suggest that a safety margin is superfluous again. However, this is not discussed further by the authors. In addition to this investigation, the authors also develop an approximate model for the probability distribution of the outage time a connection will experience on a specific route during its contract period. The model is based on a route's steady-state unavailability and mean time to repair (MTTR), and it assumes Poisson-distributed failure arrivals in the network. With this, a network operator can design the guaranteed availability in the SLA in such a way that an SLA violation will only occur with a certain, controllable probability, e.g., 1 %.

Clemente et al. [Cle+05] address the same question in a simulation study comprising a 38-node Italian automatically switched optical network (ASON). They also find that a significant safety margin is necessary. Furthermore, for a contract period of one month, the margin must be larger than for a contract period of a year. Xia et al. [XCW09], Mastroeni and Naldi [MN11b], and González and Helvik [GH12a] provide more detailed statistical models that also consider the distribution of the repair time. In particular, Mastroeni and Naldi [MN11b] show that the probability of SLA violation changes with the characteristics of the repair time distribution even though the steady-state availability is constant.

Overall, it becomes clear that the probability of SLA violation depends on the statistical characteristics of the connection's failure and repair process and the duration of the billing cycle or contract period, respectively. Routing based on the steady-state availability cannot capture and incorporate this probability and exposes the network operator to a nonquantifiable risk of SLA violation.

3.3.2 Economic Risks

Since conventional, steady-state-based provisioning algorithms do not consider the probability of SLA violations, it is also difficult to estimate and limit economic risks as part of the provisioning process. SLA violations entail various consequences. Current customers are dissatisfied with the service and might switch to a different network operator. Also, frequent

or severe SLA violations may damage the operator's reputation and drive away potential customers [FS17; GH12b]. Those consequences are quite subjective and, thus, challenging to model. However, another consequence is that the network operator has to compensate its customer. The amount of compensation is well-defined in the SLA itself and can be determined objectively. That means at least a part of the economic risks, namely the SLA compensation, can be modeled objectively. This, in turn, allows the operator to integrate economic parameters directly into the provisioning process. In that way, technical and economic aspects can be brought closer together. Conventional provisioning mechanisms miss out on this opportunity.

3.3.3 Overfulfillment of the SLA-Guaranteed Availability

A last shortcoming of conventional provisioning mechanisms we want to mention here is that they often *overfulfill* the SLA, i.e., they provide connections that perform significantly better than guaranteed in the SLA.

The problem is related to topological constraints imposed by the network itself. Potential routes must satisfy a certain SLA-derived figure of merit. This requirement is independent of whether the network operator employs the steady-state availability, the SLA violation probability, or the compensation as a decision criterion for the selection of routes. However, in most practical cases, it is impossible to find a route that matches the specified figure of merit *exactly*. Some routes perform worse than this figure, while others perform better. Using a conventional provisioning mechanism, the operator chooses a route that performs better than the specified figure of merit. If no such route exists, the connection request is rejected. As a result, all connections overfulfill the figure of merit. In other words, the connections are more reliable than required by the SLAs in place.

Overfulfillment alone is not a problem per se. Often, however, overfulfillment goes hand in hand with an increase in the amount of required network resources. This is particularly true when protection is deployed to increase the connection availability. Protection increases the connection availability at the cost of additional network resources. Again, it is almost impossible to find backup resources such that the required figure of merit is matched exactly. Consequently, the SLA is overfulfilled, and network capacity is "wasted." The finer-grained the protection mechanism is, the lower the overfulfillment and capacity waste potentially is. However, to the best of the author's knowledge, no protection mechanism exists that solves this problem entirely.

The problem of availability overfulfillment has been observed by other authors before. Both Tornatore et al. [Tor+08] and Xia et al. [Xia+09a] identify it as an opportunity to save network resources. We will discuss their approaches later in Section 3.4.3.

Illustration of overfulfillment But first, to illustrate the problem, we show how shortest paths lead to overfulfillment in the germany50 topology [Orl+10] (see Section 5.1.4.1) for different SLA availability levels. Note that we use shortest paths here because they are a standard choice in the solution of routing problems. However, overfulfillment is not a problem of shortest paths alone. Also, the use of shortest paths does not necessarily lead to overfulfillment in each and every scenario. Nevertheless, in the scenario we consider here, they do. Since this section discusses conventional provisioning mechanisms, we use the steady-state availability as SLA-derived figure of merit. However, the same qualitative observations apply to the SLA violation probability or the amount of compensation, which will be introduced later in Section 4.4.2.

The germany50 topology consists of 50 nodes. Consequently, the set of unordered node pairs N contains 1225 pairs. We assume one connection per node pair, and every connection has the same SLA requirement of α_{SLA}. We assume that nodes do not fail (see Section 4.2.1 for a justification). Hence, we compute a connection's steady-state availability from the steady-state availability of the links the connection traverses using the reduction technique introduced in Section 2.5.3. With values from Verbrugge et al. [Ver+05], we assume that the mean time to failure (MTTF) of a link e of length ℓ_e is $MTTF_e = 5.5 \cdot 10^6\,\text{km h}/\ell_e$, and the MTTR equals 9 hours. Then, with (2.10), the link availability is

$$a_e = \frac{5.5 \cdot 10^6\,\text{km h}}{5.5 \cdot 10^6\,\text{km h} + \ell_e \cdot 9\,\text{h}}. \tag{3.1}$$

We study the behavior for path protection and segment protection as introduced in Section 2.4.3. In both cases, for each node pair $(v, w) \in N$, we find the shortest path based on the link length and compute its steady-state availability. If the availability is less than α_{SLA}, we add protection to the primary path. In the case of path protection, we add a link-disjoint, end-to-end backup path. In the case of segment protection, we add a link-disjoint backup path to a variable-size segment of the primary path, as explained in Section 2.4.3. We assume infinite link capacities. Thus, all connections use the shortest path without the need for deviations.

Finally, we compute a ratio that quantifies the amount of overfulfillment. For each connection, we divide the allowed unavailability, namely $1 - \alpha_{\text{SLA}}$, by the connection's steady-state unavailability, $1 - a_{v,w}$. Then, we define the overfulfillment ratio as the average

$$o = \frac{1}{|N|} \sum_{(v,w)\in N} \frac{1 - \alpha_{\text{SLA}}}{1 - a_{v,w}}. \tag{3.2}$$

In Figure 3.2, we consider the interval $3.2 \cdot 10^{-6} < 1 - \alpha_{\text{SLA}} < 0.032$ because the overfulfillment is most prominent in this range. Even for the lowest allowed unavailability, all node pairs can be connected when path protection is deployed. That means $a_{v,w} \geq \alpha_{\text{SLA}} \forall (v, w) \in N$. With this, the overfulfillment ratio, o, is guaranteed to be larger than or equal to 1. Ideally, the ratio is exactly 1, which would mean that all node pair connections match the SLA requirement precisely. Higher values signify that the steady-state availability provided by the route of the connection is higher than required, i.e., availability is "wasted."

The solid lines in Figure 3.2 show the behavior of the overfulfillment ratio (left ordinate) over different SLA unavailability levels. The dashed line shows the share of connections that need protection of some form (right ordinate). In area 1, where the unavailability allowed in the SLA is lower than $4.2 \cdot 10^{-5}$, all connections need protection. As a matter of fact, even for the segment protection mode, all connections require full path protection in area 1. Protection of only a segment is not sufficient. Since the connections effectively receive the same protection in area 1, no matter which protection mode, they also use the same routes with the same overfulfillment. For this reason, the two curves for the overfulfillment ratio of path-protected and segment-protected connections are on top of each other in area 1. In area 3, where the allowed unavailability is higher than $1.5 \cdot 10^{-3}$, the two curves are on top of each other as well. However, this time, all connections are unprotected. Throughout area 1, the overfulfillment ratios increase. The reason is that even though the allowed unavailability, $1 - \alpha_{\text{SLA}}$, increases, all routes still need path protection throughout area 1. Hence, the ratio between the allowed unavailability and a connection's unavailability, $1 - a_{v,w}$, grows. At the transition between areas 1 and 2, the share of

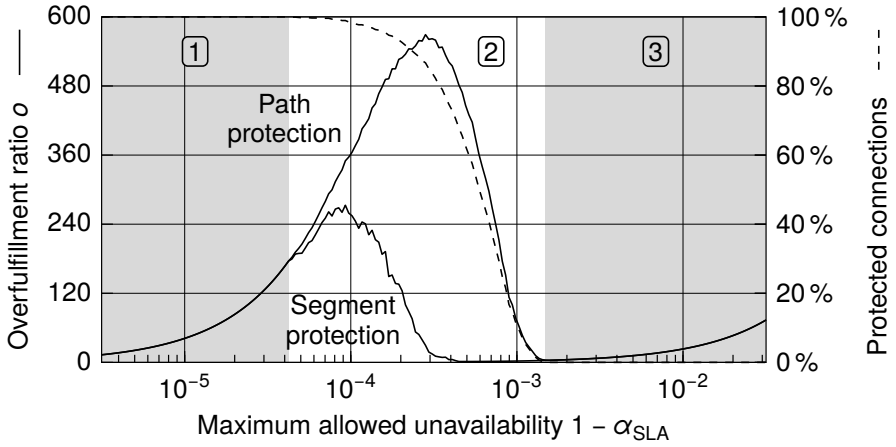


Figure 3.2 Availability overfulfillment in the germany50 topology using shortest paths with path protection and segment protection

protected connections starts to fall. The allowed unavailability is now high enough for some connections to be unprotected (in the path protection mode) or only partially protected (in the segment protection mode). Still, the overfulfillment ratios grow because only a few connections are eligible for such reduced protection. At an allowed unavailability of around $9 \cdot 10^{-5}$ in the segment protection mode and $3 \cdot 10^{-4}$ in the path protection mode, the overfulfillment ratios reach their respective maxima of 273 and 569. At those points, the vast majority of connections are still protected in some way, and the protection provides too much availability. For higher allowed unavailabilities, the overfulfillment ratios start to decrease. The respective minima are 1.5 and 3.6. In area 3, the overfulfillment ratios increase again. Similar to area 1, the routes, and therefore also the unavailabilities of the connections, do not change anymore because they are all unprotected. Considering all three areas, the overfulfillment ratio resulting from path protection is consistently higher or at least equal to that of segment protection. This indicates that a finer-grained protection scheme like segment protection can mitigate the problem of availability overfulfillment to some extent. However, in real-world scenarios, it cannot eliminate it completely, and the fundamental problem of overfulfillment remains.

Overfulfillment vs. safety margin Section 3.3.1 argued that the closer a route's steady-state availability is to the availability level guaranteed in the SLA, the higher the probability of SLA violation. Zhou and Grover [ZG05] suggest the introduction of a safety margin to adjust the violation probability. At first sight, simply accepting availability overfulfillment, as presented in this section, might appear to be a *natural solution* to adjust the probability of SLA violation. Why introduce an artificial safety margin to control the violation probability when there is availability overfulfillment anyway? The point is that both effects, the probability of SLA violation and the availability overfulfillment, are *uncontrolled* in conventional provisioning mechanisms. The amount of overfulfillment strongly depends on topological properties and does not automatically match the safety margin required to achieve the SLA violation probability the operator targets. Therefore, to realize a specific violation probability, a dedicated mechanism is required. One example is the safety margin approach proposed by Zhou and Grover [ZG05]. However, the next section will also show further approaches.

Similar to [ZG05], the provisioning mechanism proposed in this thesis uses a stochastic model to compute violation probabilities and select suitable routes. However, in contrast to

other approaches, it considers the SLA compensation instead of only the SLA availability. Additionally, our mechanism exploits overfulfillment to increase the network capacity. As a result of the above discussion, both features, the route selection based on the violation probability and the exploitation of overfulfillment, are decoupled. Each feature serves a dedicated purpose.

3.4 Existing Solution Proposals

The following sections introduce existing approaches from the literature that solve parts of the shortcomings discussed above. Section 3.4.1 summarizes studies and models concerned with the probability of SLA violation. Section 3.4.2 presents works about SLA compensation and economic risks of SLA violations. Section 3.4.3 introduces various dynamic provisioning mechanisms, most of them availability-aware and with the goal of resource-efficient SLA fulfillment or profit maximization. Lastly, Section 3.4.4 provides a summary and categorization of the introduced literature.

3.4.1 Properties and Models of the SLA Violation Probability

As mentioned in Section 3.3.1, the probability of SLA violation was first discussed in the communication network context by Zhou and Grover [ZG05] and Clemente et al. [Cle+05]. Both conclude that the steady-state availability of the connection must be much higher than the availability guaranteed in the SLA in order to comply with the SLA with a high probability. They highlight that the margin between both availability values must be significantly larger for short billing cycles on the order of months than for long billing cycles on the order of years. The reason is that the interval availability approaches the steady-state availability for long time intervals (Section 2.5.1).

Snow and Weckman [SW07] and Xia et al. [XCW09] demonstrate in simulation studies that the particular choice of MTTF and MTTR values greatly impacts the SLA violation probability as well. The steady-state availability of a connection, a, can be defined as a function of the connection's MTTF and MTTR, with $a = MTTF/(MTTF+MTTR)$. Therefore, an infinite number of MTTF and MTTR combinations have equal steady-state availabilities even though their probabilities of SLA violation in a finite time interval are different. The authors show that connections with a high MTTF are typically less prone to SLA violation because they have a high probability of not experiencing a failure at all during their lifetime.

Mastroeni and Naldi [MN11b] go one step further and compare different repair distributions, namely the negative exponential, the Weibull, and the log-normal distribution. In the presented scenarios, the differences in the resulting SLA violation probabilities appear negligible. However, they find that the violation probability decreases as the variance of the repair time grows while the MTTR is constant. This might seem counterintuitive. However, we argue that this is a plausible result and an important characteristic of the SLA violation concept. An SLA violation is of binary nature. The SLA is either violated or not, and there is no notion of severity. If a certain repair time violates the SLA, then an even longer repair time violates it as well. However, the fact that the repair time was longer is not reflected in the SLA violation. As the repair time variance around a constant MTTR grows, very long and very short repair times become more likely. Following the argument above, very long repair times do not increase the SLA violation probability further. However, very short repair times may lead to SLA fulfillment and a lower violation probability.

While many of the above results were found by simulation, there also exist various analytical methods to compute the probability of SLA violation. They all consider a network connection as a two-state system with sequential up- and downtimes. Zhou and Grover [ZG05], Xia et al. [Xia+10b; Xia+11a], Lucerna et al. [Luc+12], and Mastroeni and Naldi [MN11b] assume that the connection uptimes follow an exponential distribution and that the downtimes are negligibly short compared with the uptimes. Therefore, regarding the failure occurrences, they assume a pure Poisson process and ignore the additional time the repair process takes. In subsequent modeling steps, Zhou and Grover [ZG05], Xia et al. [Xia+11a], and Lucerna et al. [Luc+12] then assume constant downtime durations. Mastroeni and Naldi [MN11b] assume exponentially distributed downtimes. In [MN11b], the resulting expression for the probability of SLA violation contains an infinite summation covering all possible failure counts during a contract period. To evaluate this summation, the authors have to truncate it. That means they ignore high failure counts, which are irrelevant in practical scenarios anyway. In [ZG05], the approximation error is evaluated by simulation. The other authors do not evaluate the approximation error.

Like the above mechanisms, González and Helvik [GH12a] consider the connection downtimes negligibly short. Therefore, they also approximate the number of failures in a billing cycle with the number of events generated by the pure failure process. However, in contrast to the authors above, González and Helvik consider different distributions for the *uptimes*. They derive analytical expressions for the violation probability assuming exponentially, Weibull-, or gamma-distributed uptimes. The downtimes are assumed to follow an exponential distribution. The approximation error is evaluated by simulation. The authors also mention — but do not use — the precise, analytical solution for exponential up- and downtimes provided by Takács [Tak57]. The model developed in this thesis is based on results from Takács [Tak57].

Mello et al. [Mel+06] acknowledge that a network connection is not an atomic entity but consists of different network components. Specifically, they model a connection as a set of repairable links. Connections can be protected by dedicated or shared path protection. They assume exponential up- and downtimes of the individual links and model the network connection as a Markov chain. Each state in the Markov chain corresponds to a specific failure scenario, e.g., no link failures, failure of a certain single link, or failure of multiple links at once. To keep the model tractable, they consider up to two simultaneous link failures. The probability of SLA violation is computed with the approximation technique presented in [SG86]. With this technique, it is possible to analytically bound the resulting approximation error. Since the last step of computing the actual probability distribution is computationally expensive, Mello et al. also provide an adapted mechanism in [MWQ11] that only provides analytical bounds on the violation probability. The model devised by Zhao and Subramaniam [ZS15] also relies on the approximation technique in [SG86].

Mello and Lourenço [ML13] derive analytical expressions for the violation probability of shared path-protected connections. They assume exponentially distributed uptimes and exponentially or Weibull-distributed downtimes. Similar to the previous Markov-based method, the approach requires an enumeration of all link failure scenarios that lead to a connection downtime, i.e., all double-link failures. Furthermore, the authors assume that no more than one connection downtime occurs during a billing cycle. This makes the model particularly suitable for connections with short billing cycles and high reliability. In practice, this means that spatially small networks are better covered by the model than large networks.

3.4.2 Models and Applications of SLA Compensation

SLAs typically contain a compensation policy that relates SLA violations to compensation or penalty payments, respectively.

Mastroeni and Naldi [MN11a] extend the previously introduced analysis of the SLA violation probability [MN11b] to the compensation that results from those SLA violations. While the previously discussed literature sometimes uses the term *risk* as a synonym for *probability*, Mastroeni and Naldi use the term *risk* in the sense of the value at risk (VaR), a measure often used in finance [Ale09]. The VaR with respect to the SLA compensation is the amount of compensation that is not exceeded with a certain probability. That means if $C \in \mathbb{R}_{\geq 0}$ is a random variable (RV) representing the compensation that accumulates for a connection during its lifetime, then the VaR with a probability of $\eta \in [0, 1]$ (e.g., $\eta = 0.99$) is defined by [MN11a] as

$$VaR_\eta(C) = \inf\{c \in \mathbb{R}_{\geq 0} : \mathbf{P}(C > c) \leq 1 - \eta\}. \tag{3.3}$$

VaR_η is equivalent to the η-quantile. The authors study the VaR for three different compensation policies, namely a policy that compensates each failure with a fixed penalty, a policy that compensates only long outages with a fixed penalty, and a policy that compensates proportionally to the cumulated downtime in a time interval. It is shown that the choice of compensation policy impacts the VaR considerably. Also, the VaRs differ for different policies even when the average compensations are equal. This emphasizes that a treatment of SLA compensation only based on expected values ignores important information.

Chołda et al. [CGR14] also study the VaR as a risk measure for SLA compensation. In contrast to Mastroeni and Naldi, they consider the total compensation over *all* connections in the network. In that context, they discuss known shortcomings of the VaR and evaluate its suitability for practical network scenarios. One shortcoming is that the VaR is generally not *subadditive*, i.e., if C_i $(i = 1, 2, 3, \ldots)$ represents the compensation of connection i, then

$$VaR_\eta\left(\sum_i C_i\right) \leq \sum_i VaR_\eta(C_i) \tag{3.4}$$

does *not hold generally*. However, the authors show through extensive simulation that the VaR behaves like a subadditive risk measure for a very large set of practical network scenarios. Thus, in many network scenarios, the upper bound on the VaR of multiple connections can be computed by simply adding the VaRs of the individual connections. The authors also investigate the conditional value at risk (CVaR) as an alternative. The CVaR is defined as the expectation of all losses exceeding VaR_η, i.e., $CVaR_\eta(C) = \mathbf{E}(C|C \geq VaR_\eta(C))$. However, while the CVaR is subadditive in general, it is much more complex to compute. Therefore, the authors favor the VaR-based bound in (3.4) and do not recommend the more complex CVaR-based bound. In [CRG16], the same authors derive an upper bound for the total compensation that is more effective than the bound in (3.4). Finally, in [RGC16], they extend their previous two works by providing effective upper bounds for the total SLA compensation for four different compensation policies. While the model for calculating the upper bound assumes exponentially distributed up- and downtimes of the individual network components, the authors show in a simulation study that the bound is effective for Pareto, Weibull, and log-normal distributions as well.

Følstad and Helvik [FH16] present a detailed mathematical model for the network operator's profit. They assume that the operator generates revenue from the sale of connections. In

contrast, there are costs for the deployment and operation of the connections. Furthermore, the operator has to pay SLA compensation when one or more SLA guarantees are violated. The SLA guarantees considered are the maximum number of failures, the maximum number of long outages, and the maximum cumulated downtime. The authors derive the joint probability distribution for the violation of at least one of those guarantees. By combining the revenue, costs, and violation probability, they obtain the operator's profit, which they maximize in various network scenarios. One important result is that longer billing cycles lead to higher profits. A second important finding is that the highest profit is not obtained by deploying the highest possible availability because the deployment costs for high-availability connections exceed the expected compensation. This result supports our hypothesis that availability overfulfillment is not necessarily helpful from an economic perspective. Like in all presented mechanisms, the model includes several approximations and assumptions. One assumption is that the connection uptimes follow an exponential distribution with a constant failure rate. To judge the strictness of this assumption, the authors simulate a network with multiple cyclic and trend effects in the failure rate. The results show that the model of the SLA violation probability is mostly insensitive to these changes.

While the previous works mainly focus on the modeling and evaluation of compensation-related measures, the following publications also employ such measures for network design and dimensioning.

Meusburger and Schupke [MS08] find the ideal point in time to upgrade network protection in a multi-period planning study. They assume that operators have to upgrade their networks regularly in order to adapt to growing traffic demand and to provide enough protection capacity. Neglecting such protection upgrades leads to increased SLA compensation. By equating compensation with network upgrade costs that decrease over time, they find the optimal time to upgrade.

González and Helvik [GH12b] present an optimization approach to allocate a set of connections to network resources. The authors assume that, in addition to SLA compensation, connection outages also lead to a bad reputation for the operator. The authors model the overall costs as a piecewise linear function of the connection downtime, with different pieces representing different impacts on reputation and compensation costs. Using a two-stage stochastic program that supports protection, they find connection allocations that maximize the operator's profit.

3.4.3 Dynamic Provisioning Mechanisms

The previous section already introduced two static provisioning or planning approaches related to availability and compensation. In this section, we focus on *dynamic* provisioning mechanisms. Most of them are availability-aware; some even consider the SLA violation probability or compensation. Except for the works of Kantarci et al. [KMO08a; KMO08c; KMO08d; KMO09], all following approaches assume that the holding time of a connection, i.e., the contract period, is already known once the connection request arrives.

Xia et al. [Xia+10a; Xia+11b] and Sevilla et al. [Sev+11] assume that connections have time-varying protection requirements, meaning that they require protection only during certain time intervals called *critical windows* (*CWs*). A connection's CWs are known when the connection request arrives. Therefore, two or more connections can share the same backup path as long as their CWs do not overlap. That means they share backup resources but in a time-differentiated way, making them effectively dedicated. The authors present and

evaluate a local provisioning algorithm for individual requests as well as a global algorithm that provisions multiple connections at once and is more efficient due to its global knowledge. A similar mechanism is devised in [Wei+08]. Instead of CWs that require protection, Wei et al. [Wei+08] assume that connections require different levels of availability throughout their contract period, and the network operator has to reprovision the connections accordingly. Therefore, the algorithm of Wei et al. [Wei+08] dynamically adjusts the number of backup paths of connections. The simulation results show improvements in the rejection ratio and resource usage compared with scenarios with static availability requirements.

Tornatore et al. [Tor+08] acknowledge one of the fundamental problems of conventional provisioning mechanisms, namely the problem of availability overfulfillment and the resulting waste of network resources, as discussed in Section 3.3.3. To provide a remedy, the authors propose a novel shared-path-based provisioning algorithm that adapts the sharing degree of backup resources over time. For this to work, the authors assume that connections have a known contract period or holding time Δt_h together with the guaranteed SLA availability, α_SLA. In the seldom scenario with $\alpha_\mathrm{SLA} = 1$, the mechanism does not provide any advantages. Therefore, assume $\alpha_\mathrm{SLA} < 1$ here. A connection is provisioned on a route whose estimated steady-state availability, a, (including shared backup) is larger than or equal to α_SLA, i.e., $a \geq \alpha_\mathrm{SLA}$. Assume, at an arbitrary time t, a connection has been in the network for a time span Δt_p and has a duration Δt_f to go until the end of its holding time, i.e., $\Delta t_\mathrm{h} = \Delta t_\mathrm{p} + \Delta t_\mathrm{f}$. If the connection did not experience any outages during Δt_p, its availability for that time span is $a_\mathrm{p} = 1$. Then, according to the authors, the connection can be downgraded availability-wise for the remaining time Δt_f because it "did better" than required ($a_\mathrm{p} > \alpha_\mathrm{SLA}$) during Δt_p. That means the connection can be routed on a route with a lower steady-state availability. Instead of α_SLA, the route's availability now only needs to satisfy

$$\alpha' = \frac{\alpha_\mathrm{SLA} \cdot \Delta t_\mathrm{h} - \Delta t_\mathrm{p}}{\Delta t_\mathrm{f}}. \tag{3.5}$$

Since $\alpha' < \alpha_\mathrm{SLA}$, the sharing degree of the connection's backup path can be increased. In that way, more connections can be accommodated in the network, and the availability overfulfillment is reduced. The authors only increase the sharing degree if no outages occurred in Δt_p. However, as they mention themselves, if an outage occurs, it might be necessary to decrease the sharing degree again. This scenario is covered by the authors in [Luc+09], which is an extended version of [Tor+08]. The authors now distinguish the two cases *credit* and *debit*. *Credit* means the network operator provides too much availability (discussed in [Tor+08]), while *debit* means the operator provides too little availability. The *debit* situation can arise either due to connection outages or deliberately to accept more connections when it is predictable that this debit will dissolve with future connection departures. The simulation results show significant reductions in the rejection ratio compared with a conventional availability-aware provisioning mechanism. However, the availability overfulfillment is not reduced significantly. Finally, [Luc+12] is yet another slight extension in this series of publications. The authors introduce a model for the probability of SLA violation. They describe how the violation probability can replace the steady-state availability to make their provisioning mechanism more precise. However, they do not present any evaluation for this updated algorithm.

Kantarci et al. [KMO08a; KMO08c; KMO08d; KMO09] devise another provisioning algorithm based on a variable sharing degree for shared path protection. They assume that a connection belongs to one of several availability classes. They regularly evaluate the average steady-state unavailability and resource usage per availability class. Further, they assume

that a high resource usage corresponds to many deployed protection resources and, hence, low unavailability. In contrast, a low resource usage is expected to lead to high unavailability. The authors formulate this tradeoff as an equation that the provisioning algorithm regularly minimizes with respect to the sharing degrees of all links. Afterward, the link weights used for routing are updated based on the resulting sharing degrees. As a result, subsequent connection requests are routed with adapted shared path protection. Compared with a conventional provisioning algorithm, the rejection ratio and resource usage are considerably improved.

Similarly, the authors of [Cav+07] adapt link weights before a new connection is provisioned. The link weights are used for pathfinding during routing, and they represent the link's steady-state availability and the backup sharing degree. The authors assume that the departure times of existing connections are known. Consequently, they know when these connections will release network resources. Every time a departing connection releases network resources, the sharing degree of the occupied links changes. Therefore, when a new connection is provisioned, the algorithm first computes link weights that take future changes in the sharing degrees into account. More specifically, a link's weight is a time-weighted average of the link's estimated future sharing degrees and its steady-state availability. In this way, the most reliable paths can be found using a shortest path algorithm. Finally, the connection is routed without protection, with shared path protection, or with dedicated path protection so that its SLA availability guarantee is fulfilled.

Chen et al. [Che+15] present a mechanism that is inspired by [Luc+09]. The protection mode of connections — no protection, shared path protection, or dedicated path protection — is upgraded or downgraded depending on their failure evolution. The authors present a proof-of-concept implementation specific to software-defined elastic optical networks and, by that, demonstrate the practical feasibility of such approaches.

In [Xia+08] and [Xia+09b], Xia et al. devise a provisioning and reprovisioning algorithm that maximizes revenue by reducing SLA violations. SLA violations are reduced because connections can preempt the backup resources of lower-priority connections during initial provisioning or regular reprovisioning. A connection's priority at time t is based on its remaining allowed number of failures (ANF). The ANF estimates how many more outages the connection can tolerate before violating its SLA. It is based on the connection's allowed downtime (ADT), which is defined in terms of the contract period, Δt_h, as $ADT = (1 - \alpha_\mathrm{SLA}) \cdot \Delta t_\mathrm{h}$. Furthermore, let x denote the total connection downtime that accumulated until time t. Then, the ANF at time t is defined as

$$ANF = \left\lfloor \frac{ADT - x}{MTTR} \right\rfloor. \tag{3.6}$$

The ANF is comparable to a dynamic estimation of the SLA violation probability. However, it is non-probabilistic because it only takes the expectation of the downtime (MTTR) into account. Connections with no more allowed failures ($ANF = 0$) and connections that already violated their SLA ($ANF < 0$) have the highest priorities. Connections that can tolerate additional downtime have lower priorities. The final connection priority also takes the potential amount of compensation into account and is summarized by the so-called urgency level (UL). Since the mechanism allows regular reprovisioning of connections, connection requests are accepted even if no backup path is available. Connections accepted in this "best-effort" manner are more prone to SLA violation. However, they can be protected at a later point in time during reprovisioning. The authors evaluate their mechanism by simulation and show significant improvements in terms of rejection ratio, resource requirements, SLA

fulfillment ratio, and revenue compared with static mechanisms.

Dikbiyik et al. [DTM15] also employ the UL as a priority metric that characterizes the SLA violation probability. They use network excess capacity, i.e., network capacity that is currently not used by connections, for the effective protection of existing connections. Depending on how much excess capacity is available, connections receive dedicated link protection, 1+1 protection, or shared path protection. If no excess capacity is available, the algorithm reprovisions existing connections to free capacity, or it suggests capacity upgrades to the network operator. Like in [Che+15], connections with a high UL, i.e., a high priority, receive improved protection while the protection of low-priority connections is downgraded during reprovisioning. Other approaches compute the connection availability analytically, e.g., as a function of the underlying network links. Dikbiyik et al. [DTM15] estimate the availability with an exponentially weighted moving average over existing connections' experienced availabilities. Overall, the algorithm effectively delays costly link capacity upgrades.

The algorithm in [Xia+09a] is again based on the ADT. Connections whose cumulated downtime is close to their ADT can preempt other connections that can tolerate additional downtime. However, in this mechanism, connections are grouped in so-called *service clusters*. Connections can only preempt connections of their own service cluster. In [Xia+09a], a service cluster groups connections with the same source and destination node. Generally, however, clusters can realize other groupings as well. The authors assume that connections approximately fulfill the required SLA availability with a primary path only. Therefore, the provisioning algorithm does not add protection explicitly. Instead, the connections inside a service cluster are virtually protected by the other connections in their cluster through the preemption mechanism described above. In order not to overload a service cluster, connection requests can be rejected. To decide whether a new connection can be accepted, the algorithm evaluates the "protection capacity" of the respective cluster. To this end, it estimates the *excess allowed downtime* each existing connection will have at the end of its contract period. For a connection with a remaining holding time of Δt_f, this excess allowed downtime at time t is

$$e = ADT - x - (1 - a) \cdot \Delta t_\mathrm{f}. \tag{3.7}$$

Here, the difference $ADT - x$ is the remaining allowed downtime. The term $(1 - a) \cdot \Delta t_\mathrm{f}$ is the expected additional downtime the connection will experience until the end of its contract period on its route with steady-state availability a. If e is negative, the connection is expected to violate its SLA. If it is positive, this excess downtime can be used to recover failed connections in the cluster. A new connection is accepted into a specific cluster only if the sum of all excess allowed downtimes of the connections in that cluster and of the new connection is non-negative, i.e., if

$$\sum_{i=1}^{n+1} e_i \geq 0 \tag{3.8}$$

where e_i is the excess allowed downtime of connection i, n is the number of existing connections in the particular cluster, and e_{n+1} is the excess allowed downtime of the new connection. Compared with shared path protection, the simulation results show similar SLA fulfillment ratios but greatly reduced rejection ratios.

Nafarieh et al. [Naf+11] present a rather drastic approach that drops connections as soon as their remaining holding time is less than their remaining allowed downtime, i.e., $\Delta t_\mathrm{f} < ADT - x$. While this approach prevents overfulfillment completely, we find it rather

inappropriate to *deterministically* shut down connections prior to the end of their contract. The authors also assign a risk level — the *path risk factor* — from 0 to 3 to existing connections and their routes based on their remaining allowed downtime. The lower the remaining allowed downtime is, the more "risky" a connection is. When a new connection is provisioned and a suitable route has to be selected, this risk level is incorporated into the decision process. This means that the algorithm makes use of the history of existing connections. Assuming constant and equal downtimes of links, Nafarieh et al. use the failure rate of a link as its weight while retrieving shortest primary paths. In this way, they obtain paths with minimal SLA violation probability.

In [Xia+10b] and [Xia+11a], Xia et al. extend this approach to unequal downtimes by transforming the failure rates. The transformation is only an approximation. Nevertheless, if the link failure rates and downtimes across different links do not vary too much, the approximation error is bearable. Thus, in typical network scenarios, the method is able to find the path with minimal SLA violation probability using Dijkstra's shortest path algorithm. Xia et al. employ this technique in a provisioning algorithm that minimizes a connection's SLA violation probability. They compare it with an algorithm that maximizes a connection's steady-state availability and show in a simulation that the resulting SLA violation ratio is significantly lower than that of the availability-maximizing algorithm. This confirms that the steady-state availability is a non-optimal decision criterion when the SLA violation probability should be adjusted or minimized (see discussion in Section 3.3.1).

Das [Das12] presents a provisioning mechanism that maximizes the operator's profit by routing on paths with low SLA violation probability and by rejecting connection requests that are not profitable. The algorithm first finds an unprotected and a protected route with minimal SLA violation probability. Then, it estimates the expected profit for both routes. The profit comprises the costs incurred by deploying the connection, the revenue, and the expected compensation. The route with the higher expected profit is considered for provisioning. However, the connection request is only accepted if the expected profit exceeds an operator-defined minimum threshold. Depending on the threshold, more or less connection requests are accepted. Interestingly, the simulation results show that accepting all or most requests doubles the profit compared with accepting only high-profit requests. Unfortunately, no details are provided about how the SLA violation probability is calculated.

Zhao and Subramaniam [ZS15] devise another profit-oriented provisioning mechanism. They compute the SLA violation probability like Mello et al. [Mel+06]. From the violation probability and a staircase-shaped compensation policy, they derive the expected compensation. The revenue generated by a connection is based on its holding time, the distance between the source and destination node, and the SLA availability. The algorithm selects either the route with minimum expected compensation or the route with minimum resource costs. In both cases, only routes whose ratio of expected compensation to revenue does not exceed an operator-defined threshold are accepted. In contrast to most other mechanisms, Zhao and Subramaniam take the problem of regenerator placement and many physical layer parameters of optical networks into account.

3.4.4 Summary and Categorization

The following three tables summarize and categorize the presented literature. Table 3.1 contains stochastic models related to the experienced availability of a connection and the probability of SLA violation (Section 3.4.1). Table 3.2 contains stochastic models related

to the amount of compensation that arises due to SLA violations (Section 3.4.2). Lastly, Table 3.3 contains the dynamic provisioning mechanisms presented in Section 3.4.3.

In Tables 3.1 and 3.2, the columns *uptime* and *downtime* contain the distributions of up- and downtimes of the basic components considered by the models. Some of the listed distributions are classified as "simulation only." Those distributions are analyzed by the authors in simulation studies only. The other distributions are covered by analytical models. In the simpler models, a single basic component corresponds to the whole connection. For the more complex models, a connection consists of multiple network components, e.g., links and nodes. In this case, a basic component corresponds to a network component. Whether the model supports the composition of multiple network components is shown in the column *composite*. The column *supported protection* lists protection modes the model can handle.

In Table 3.2, the column *target statistic* describes the final statistic the authors are interested in. The modeled distribution serves to compute this statistic. The columns *basis* and *function* describe the compensation policies the models are based on. If *basis* is *billing cycle*, the policy considers all outages in a billing cycle jointly to determine the amount of compensation. An example would be a policy that is based on the cumulated downtime per billing cycle. If *basis* is *individual outage*, a compensation is determined for every outage individually.

In Table 3.3, the column *mechanism* summarizes the core mechanism exploited in the provisioning approach. Among others, the column distinguishes between preemptive reprovisioning, denoted by *preemption*, and non-preemptive reprovisioning, simply termed *reprovisioning*. The column *measures* contains availability-related measures that are incorporated into the provisioning. The measures reflect the category of the mechanism (steady-state availability, violation probability, or compensation as listed in Section 3.1). Some mechanisms combine aspects of multiple categories. The column *online* is checked if the mechanism needs to know the past downtime or availability of existing connections at arbitrary points in time. This requires the online collection of outage data. The column *reprovisioning* is checked if the routing of existing connections is changed during their lifetime. This might be either planned as in [Wei+08] or unplanned to react to state changes in the network. The column *protection* shows the general types of protection the provisioning mechanism can be used with *beneficially*. Here, *any* means primarily dedicated path protection or shared path protection. However, presumably, other protection mechanisms can be used as well. The important point is that the provisioning mechanism benefits from the fact that some kind of protection is present. Without protection, the respective mechanisms do not work. *Shared* means that only protection mechanisms that share resources are suitable. The reason is that the core mechanism of the respective provisioning mechanisms is the adjustment of the sharing degree of backup resources. *Unprotected* means that the provisioning mechanism provides advantages even with unprotected connections.

Table 3.1 Stochastic models related to the experienced connection availability

Reference	Modeled distribution	Uptime	Downtime	Composite	Supported protection
Zhou and Grover [ZG05]	Number of failures	Exponential	· Constant · Exponential†	×	–
Xia et al. [XCW09]	Cumulated downtime	Exponential	Normal	×	–
Mastroeni and Naldi [MN11b]	Cumulated downtime	Exponential	· Exponential · Weibull† · Log-normal†	×	–
Xia et al. [Xia+10b; Xia+11a]	Number of failures	Exponential	· Constant · Normal† · Uniform† · Weibull†	✓	Shared
Lucerna et al. [Luc+12]	Number of failures	Exponential	Constant	✓	Shared
González and Helvik [GH12a]	Cumulated downtime	· Exponential · Gamma · Weibull	Exponential	×	–
Mello et al. [Mel+06]	· Cumulated downtime · Individual downtime	Exponential	Exponential	✓	· Dedicated · Shared
Mello et al. [MWQ11]	Cumulated downtime	Exponential	Exponential	✓	· Dedicated · Shared
Zhao and Subramaniam [ZS15]	Cumulated uptime	Exponential	Exponential	✓	Dedicated
Mello and Lourenço [ML13]	Cumulated downtime	Exponential	· Exponential · Weibull	✓	· Dedicated · Shared

†Simulation only

Table 3.2 Stochastic models related to the amount of SLA compensation

Reference	Modeled distribution	Target statistic	Compensation policy		Uptime	Downtime	Composite	Supported protection
			Basis	Function				
Mastroeni and Naldi [MN11a]	Compensation per connection	VaR	Billing cycle	· Prop.* to num. of failures · Prop.* to num. of long outages · Prop.* to cumulated downtime	Exponential	Exponential	×	–
Chołda et al. [CRG16]	Total compensation over all connections	Upper bound for VaR	Individual outage	· Constant · Prop.* to downtime	Exponential	Exponential	✓	–
Rusek et al. [RGC16]	Total compensation over all connections	Upper bound for VaR	Individual outage	· Constant · Prop.* to downtime · Prop.* to downtime with constant offset · Square of downtime	· Exponential · Weibull	· Exponential · Pareto · Log-normal	✓	Dedicated
Følstad and Helvik [FH16]	Compensation per connection	Profit	Billing cycle	Constant if max. num. of failures, max. num. of long outages, or max. cumulated downtime exceeded	Exponential	Gamma	×	–

*Proportional

Table 3.3 Availability-aware dynamic provisioning mechanisms

Reference	Overall goal	Solution approach	Mechanism	Measures	Online	Reprov.‡	Protection
Sevilla et al. [Sev+11], Xia et al. [Xia+10a; Xia+11b]	Resource-efficient protection	Connections with non-overlapping CWs share backup resources	Time-differentiated resource sharing	–	×	×	Any
Wei et al. [Wei+08]	Resource-efficient fulfillment of time-varying SLA availability	Adapt protection to meet time-varying SLA availability	Reprovisioning	Steady-state availability	×	✓	Any
Lucerna et al. [Luc+09; Luc+12], Tornatore et al. [Tor+08]	Resource-efficient fulfillment of SLA availability	Adapt residual availability requirements over time	Parameter adaptation	· Experienced availability · Steady-state availability · SLA violation probability	✓	×	Shared
Kantarci et al. [KMO08a; KMO08c; KMO08d; KMO09]	Resource-efficient fulfillment of SLA availability	Adapt link sharing degrees to minimize unavailability-resource product	Parameter adaptation	Steady-state availability	×	×	Shared
Cavdar et al. [Cav+07]	Resource-efficient fulfillment of SLA availability	Time-weighted link weights	Parameter adaptation	Steady-state availability	×	×	Shared
Chen et al. [Che+15]	Resource-efficient fulfillment of SLA availability	Adapt residual availability requirements and reprovision protection	Reprovisioning	· Experienced availability · Steady-state availability	✓	✓	Any

‡Reprovisioning

↓ Continued on next page

Table 3.3 (continued)

Reference	Overall goal	Solution approach	Mechanism	Measures	Online	Reprov.‡	Protection
Xia et al. [Xia+08; Xia+09b]	Profit maximization	Urgent connections preempt others to reduce SLA violation and compensations	Preemption	· Experienced downtime · Compensation	✓	✓	Any
Dikbiyik et al. [DTM15]	Resource-efficient fulfillment of SLA availability	Reprovision backup based on urgency level and excess capacity	Reprovisioning	· Experienced downtime · Experienced availability	✓	✓	· Any · Unprotected
Xia et al. [Xia+09a]	Resource-efficient fulfillment of SLA availability	Connections inside a cluster preempt each other to avoid SLA violation	Preemption	· Experienced downtime · Steady-state availability	✓	✓	· Any · Unprotected
Nafarieh et al. [Naf+11]	Resource-efficient fulfillment of SLA availability	· Estimate route availability from history · Drop connection once SLA is fulfilled	Parameter adaptation	· Experienced downtime · Steady-state availability	✓	×	· Any · Unprotected
Xia et al. [Xia+10b; Xia+11a]	Minimization of SLA violation probability	Normalize link failure rates to find reliable path with Dijkstra	Parameter adaptation	SLA violation probability	×	×	· Any · Unprotected
Das [Das12]	Profit maximization	Admit connections only if expected profit exceeds threshold	Admission control	Expected compensation	×	×	· Any · Unprotected
Zhao and Subramaniam [ZS15]	Profit maximization	Admit connections only if expected profit exceeds threshold	Admission control	Expected compensation	×	×	· Any · Unprotected

‡Reprovisioning

3.5 Motivation for a Novel Provisioning Mechanism

In Section 3.3, we discussed three major shortcomings of conventional availability-aware provisioning mechanisms:

1. Connection failures and repairs are random processes, but the steady-state availability can capture their properties only partially. In particular, the probability of SLA violation is not considered in conventional mechanisms.
2. SLAs define not only technical performance parameters but also compensation policies for the case the performance parameters are not satisfied. Conventional mechanisms do not consider these policies and, hence, do not integrate economic aspects into the provisioning process.
3. Conventional provisioning mechanisms often overfulfill availability requirements because it is almost impossible to find routes that match the required availabilities exactly. This availability overfulfillment ties up network resources.

Section 3.4 introduced existing provisioning mechanisms that tackle those shortcomings. However, there is no mechanism that tackles all three shortcomings jointly. For this reason, we propose a new provisioning mechanism in this thesis. However, before introducing it in the next chapter, we first explain how it relates to the existing mechanisms.

Most of the presented works target the fulfillment of SLA availability in a resource-efficient way. That means they try to solve the problem of excessive resource usage and, if only implicitly, the problem of availability overfulfillment. They attempt this by reprovisioning protection, preemption of other connections, or by adapting weights to steer the routing of future connections. Both reprovisioning and preemption change the network state. Too many changes, both in the optical and electrical layers, may render the network unstable [Cug+13; CJH16]. Therefore, we do not employ such techniques in the provisioning mechanism we propose in this thesis. Instead, we only provision a new connection once with a route that does not change as long as the connection is in the network. This results in high network stability and low complexity of the network management. Furthermore, it relaxes the technological requirements imposed on the network devices because they do not need to support frequent reconfiguration.

Some of the presented mechanisms assume to know the experienced downtime or availability of existing connections at arbitrary points in time. This requires intensive signaling and bookkeeping of connection states. To avoid this, our approach does not depend on failure data of existing connections at all. In fact, it only requires knowledge about the remaining link capacities and the operational state of network components to ensure that new connections are provisioned on a working route. Usually, this information is readily available in modern network management systems [Nok19; Hua23].

Most authors employ the steady-state availability as a decision criterion, like in conventional mechanisms. Exceptions are Lucerna et al. [Luc+12], Xia et al. [Xia+10b; Xia+11a], Das [Das12], and Zhao and Subramaniam [ZS15], which consider the probability of SLA violation stochastically. Stochastic models contain more information than models based on expected or steady-state values. They can represent the real behavior of network connections better. Therefore, we also use a stochastic model, both for the cumulated connection downtime and the amount of SLA compensation (see Section 4.4).

Xia et al. [Xia+08; Xia+09b], Das [Das12], and Zhao and Subramaniam [ZS15] are the only ones to incorporate economic aspects directly into the provisioning mechanism. They estimate the revenue from the sale of a connection and the amount of compensation the

operator has to pay due to an SLA violation. Das [Das12] also considers the deployment costs. The provisioning mechanisms then try to maximize the resulting profit. Xia et al. [Xia+08] and Das [Das12] apply a constant compensation if the SLA availability is violated. In [Xia+09b] and [ZS15], the compensation policy is a staircase function with multiple levels of compensation. As explained in Section 2.6.2.2, compensation policies in real SLAs are often staircase functions. For this reason, we also employ staircase functions in our model. Xia et al. [Xia+09b] do not model the compensation stochastically. Zhao and Subramaniam [ZS15] model the distribution of the cumulated uptime, but for the compensation, they only consider the expected value. We go one step further and incorporate the distribution of the SLA compensation directly into our provisioning mechanism. To be more precise, one goal of our approach is to avoid, with a certain probability, that the compensations exceed a pre-defined maximum. This is comparable to the VaR concept. However, we compute the probability for a fixed maximum compensation while the VaR is the quantile for a fixed probability.

The compensation is a deterministic function of the connection downtime. Using the compensation policy defined in the SLA, its computation is straightforward. On the other hand, it is much less clear and generalizable how connections are priced and what deployment and operational costs they incur. Choosing the wrong assumptions here may render any evaluations of the resulting provisioning mechanism and their conclusions invalid. Therefore, instead of modeling all those terms and maximizing the operator's profit, we focus on the SLA compensation alone. It will be the operator's responsibility to define an acceptable amount of compensation based on its knowledge about costs incurred and connection pricing.

Sections 3.4.1 and 3.4.2 discuss different models for the SLA violation probability or cumulated downtime, respectively, and the corresponding compensation. The models in [CRG16] and [RGC16] compensate every outage individually. We target SLAs that compensate based on the cumulated downtime during a billing cycle. Therefore, these models are not applicable. Both Mastroeni and Naldi [MN11a] and Følstad and Helvik [FH16] compensate based on the cumulated downtime. However, neither of them uses a staircase function as compensation policy. Furthermore, they base their approximate models on the assumption that the downtimes are negligibly short compared with the uptimes. While this assumption is usually valid in carrier-grade transport networks, it still introduces an approximation error that may impact the final results of the provisioning mechanisms. In order to avoid such uncertainties, we employ a more precise model for the cumulated downtime based on the work of Takács [Tak57]. The work of Takács was already mentioned by González and Helvik in [GH12a]. However, they do not use it in their approach. With this, we derive the distribution of the amount of compensation that arises during the contract period of a network connection. As another extension to existing mechanisms, we take into account that the connection performance is typically evaluated per billing cycle and not only once at the end of the whole contract period. Therefore, to estimate the amount of compensation for the whole contract period, one has to aggregate the compensation terms of all contained billing cycles. The detailed derivation is presented in Section 4.4.

Lastly, we tackle the problem of overfulfillment, or, put differently, we increase the network capacity by *exploiting* overfulfillment. Our idea is inspired by the works we presented. Authors like Lucerna et al. [Luc+12] and Chen et al. [Che+15] relax the availability requirements of connections over time. As a result, those connections need less protection. In the case of dedicated protection, the freed network resources can be used by other connections. For shared protection, the sharing degree can be adjusted. Essentially, excess availability is

shared among all connections to improve the network capacity. Xia et al. [Xia+09a] share excess allowed downtime among connections in a cluster. They do not adapt connection requirements over time but only accept new connections if enough excess allowed downtime is available in a cluster. Our provisioning mechanism implements a similar sharing idea. However, since we consider the SLA compensation stochastically during provisioning, we share "excess compensation probability" among connections. The next chapter introduces the concept and its terminology precisely.

4

Compensation-Aware Provisioning with Surplus Sharing (CAPSS)

This chapter introduces *compensation-aware provisioning with surplus sharing* (*CAPSS*), the provisioning mechanism proposed in this thesis. CAPSS considers the service level agreement (SLA) compensation during provisioning. Furthermore, it increases the network capacity by sharing SLA overfulfillment among different network connections. Parts of the mechanism and related concepts have already been published by the author of this monograph in [End22; EK21; End21].

Section 4.1 introduces the basic concept of the mechanism. To introduce the details of CAPSS concisely, Section 4.2 presents several assumptions about the network and the connections. Section 4.3 introduces an important building block of the whole mechanism, namely a procedure to compute the failure and repair rate of a route from the underlying network components' rates. Subsequently, Section 4.4 derives the mathematical model for the distribution of SLA compensation. This model is the mathematical core of CAPSS. Based on this model, Section 4.5 explains the actual provisioning algorithm. Lastly, with the whole mechanism introduced, Section 4.6 classifies CAPSS based on the categories of Section 3.4.4. Furthermore, it discusses the applicability of CAPSS in use cases other than transport networks.

4.1 Concept of the Provisioning Mechanism

As discussed in the previous chapter, this thesis proposes a novel availability-aware dynamic provisioning mechanisms. Before introducing the details, this section explains the basic concept of the mechanism.

The previous chapter has shown that conventional provisioning mechanisms suffer from several shortcomings. They are based on the steady-state availability, which only describes long time intervals properly. Therefore, the network operator is unaware of the probability

of SLA violation in a billing cycle. Furthermore, conventional provisioning mechanisms only incorporate technical parameters, like the steady-state availability and the availability level guaranteed in the SLA. They ignore economic parameters like the compensation policy entirely, even though they are present in the SLA. Lastly, the conventional route selection leads to overfulfillment of availability.

Section 3.4.3 presented several approaches that tackle those shortcomings. Some of the presented approaches employ the probability of SLA violation or even the expected SLA compensation or profit. Many of them adopt the principle of *sharing* to increase network capacity and reduce availability overfulfillment. For example, network resources are shared cooperatively among connections using shared path protection. Other approaches share network resources by reprovisioning. In that way, connections that are not at risk of violating their SLA give capacity to other connections. Figuratively, availability overfulfillment is shared among connections.

CAPSS combines ideas from existing literature and augments them with novel solutions. In the following, we will introduce the concept of CAPSS in two steps. In the first step, we present a compensation-aware provisioning mechanism, named compensation-aware provisioning (CAP), that does not employ any sharing. In the second step, CAP is extended to the full CAPSS mechanism, including the sharing of overfulfillment. The mechanism without overfulfillment sharing, CAP, will also serve as a reference approach in the evaluation chapter.

4.1.1 Compensation Awareness

Conventional provisioning mechanisms, and also most of the mechanisms in Section 3.4.3, employ the steady-state availability as a decision criterion for routing. In our approach, we employ the SLA compensation instead. We model the compensation in a billing cycle as a function of the cumulated downtime a connection experiences. The downtime is mapped to the compensation with the help of the compensation policy of the SLA. Previous results have shown that the expected value of the SLA compensation only contains limited information [MN11a]. To improve on this, we incorporate the whole probability distribution of the SLA compensation. That means when a connection is routed, the route is selected such that the resulting compensation exceeds an operator-defined limit with an operator-defined probability only. Using the distribution of SLA compensation has two advantages. First, it helps to enforce the network operator's economic goals since it can be related to its business models more directly than availability parameters like the steady-state availability or the SLA violation probability. Second, the distribution of SLA compensation is modeled based on the random failure and repair process of the connection and, hence, captures its probabilistic nature. The steady-state availability cannot capture such details.

Let $C \in \mathbb{R}_{\geq 0}$ be a random variable (RV) describing the total compensation the network operator has to pay for a connection over the whole contract period, Δt_{h}. We assume that the network operator is able to define a maximum amount of compensation, $c_{\max} \in \mathbb{R}_{\geq 0}$, it can tolerate. As discussed earlier, failure-related metrics cannot be limited deterministically in a reasonable manner. Of course, for most compensation policies, a worst-case compensation can be determined, which cannot be exceeded (see Table 2.1). However, practically relevant compensation levels (or generally everything lower than the worst case) will be exceeded with a *non-zero* probability. Therefore, the goal of the proposed provisioning mechanism is to provision connections such that the probability $\mathbf{P}(C \leq c_{\max})$ of not exceeding the

compensation limit, $c_{\max}$, is greater than or equal to an operator-defined target probability p_t. We denote the probability $\mathbf{P}(C \leq c_{\max})$ as *compliance probability* or simply *compliance*. Further, we call the probability p_t *compliance target*.

Whenever a connection request i ($i = 1, 2, \ldots$) arrives and needs to be routed, CAP finds a route for which the *computed compliance probability* p_i satisfies the compliance target, i.e.,

$$p_i \geq p_t. \tag{4.1}$$

The compliance probability is computed using a mathematical model that will be introduced in Section 4.4. If no suitable route is found, the connection request is rejected. Note that we distinguish between the *true* compliance probability, $\mathbf{P}(C \leq c_{\max})$, and the *computed* compliance probability, p_i. The algorithm can only take the computed compliance into account. However, since p_i is computed with a mathematical model, it may differ from the true compliance probability. This issue is evaluated in Section 5.2.

With the described provisioning algorithm, the operator tolerates a compensation above $c_{\max}$ only with a probability of $1 - p_t$. As long as a connection can potentially fail and $c_{\max}$ is less than the worst-case compensation — which we do assume for real-world scenarios — $\mathbf{P}(C \leq c_{\max})$ is strictly less than 1. Therefore, if the network operator sets $p_t = 1$, the provisioning mechanism does not find routes at all. As a consequence, p_t is required to be less than 1. That means a residual probability of exceeding the self-defined compensation limit is inevitable. Such a residual probability is also present in conventional mechanisms, but it is not quantifiable there. Our mechanism makes it quantifiable and, in that way, also controllable.

The parameters $c_{\max}$ and p_t are the two main parameters of the provisioning mechanism. In the following, we assume that both are defined globally and apply to all provisioned connections. However, extensions are conceivable that allow different parameter values for different classes of connection requests. If we compare different values for $c_{\max}$, we call the *lower* value the stricter one. In contrast, when comparing different values for p_t, we call the *higher* value the stricter one.

A special parametrization is $c_{\max} = 0$. If the operator does not allow compensation, a connection has to fulfill the SLA availability. Consequently, the compliance probability, $\mathbf{P}(C \leq c_{\max})$, boils down to the complementary SLA violation probability, i.e.,

$$\mathbf{P}(C \leq 0) = \mathbf{P}(A \geq \alpha_{\text{SLA}}) \tag{4.2}$$

$$= 1 - \mathbf{P}(A < \alpha_{\text{SLA}}). \tag{4.3}$$

CAP is then comparable to existing mechanisms that focus on the SLA violation probability.

So far, CAP is similar to conventional mechanisms in that it finds a route with a performance parameter no worse than a pre-specified figure of merit. However, employing the compensation as a decision criterion also changes the perspective significantly. Providing the availability level guaranteed in the SLA is a direct contract fulfillment and is mainly *customer-focused*. In contrast, selecting routes in such a way that the SLA compensation is "limited" does not necessarily mean that the SLA availability level is fulfilled. It can be violated as long as the resulting amount of compensation is within the operator's tolerable limit. Therefore, this approach is *operator-focused*. The goal is no longer the fulfillment of the SLA alone. Instead, the SLA with the guaranteed availability level and, in particular, the compensation policy is a mere parameter for the operator's business models.

Another similarity the approach still has with conventional mechanisms is that it suffers from overfulfillment. Connections are only accepted if (4.1) is satisfied. As argued in

Section 3.3.3, it is highly unlikely that a route is found that matches the targeted probability exactly ($p_i = p_t$). Therefore, the compliance target will eventually be overfulfilled, i.e., $\mathbf{P}(C \leq c_{max}) > p_t$. We argue that it is preferable to avoid overfulfillment. After all, the operator parametrizes c_{max} and p_t intentionally based on its business models. A compliance probability higher than intended might seem beneficial because it means lower compensation payments. However, it comes at the price of increased network resource usage and, hence, higher connection request rejection rates. Therefore, the extended mechanism, CAPSS, introduced in the next section, tries to fulfill the operator's parameters as precisely as possible, i.e., without any overfulfillment.

As we will see later on, the algorithmic realization of CAPSS and CAP is very similar. Therefore, we will not introduce details about CAP separately. Instead, Section 4.5 will describe how the realization of CAP differs from that of CAPSS.

4.1.2 Surplus Sharing

The compensation-aware mechanism of the previous section, CAP, has already changed the perspective from customer-focused to operator-focused. In order to avoid overfulfillment and increase network capacity, CAPSS goes one step further and relaxes the condition for route selection in (4.1). Individual connections can be provisioned with a lowered compliance probability as long as the *average* compliance probability over all connections matches the operator-defined compliance target. This is a major difference from conventional provisioning mechanisms and also most of the mechanisms presented in Section 3.4.3, in which the relevant figure of merit, e.g., the SLA availability, is enforced for every connection separately.

When the i-th connection ($i = 1, 2, \ldots$) is provisioned with CAP, the connection overfulfills the compliance target by the probability $p_i - p_t \in [0, 1)$. Assuming that the provisioning mechanism can memorize this overfulfillment, we make use of it by relaxing the compliance target for the subsequent connection request. That means the request $i+1$ can be provisioned on a route with a compliance probability larger than or equal to $p_t - (p_i - p_t) \leq p_t$. We refer to the amount of overfulfillment as *compliance surplus* or simply *surplus*. We denote the compliance surplus resulting from the provisioning of the i-th connection by $\Delta p_{s,i} \in \mathbb{R}_{\geq 0}$. Further, we denote the *relaxed compliance target* of the i-th connection request by $p^*_{t,i} \in (-\infty, 1)$ and define it as

$$p^*_{t,i} = p_t - \Delta p_{s,i-1} \tag{4.4}$$

with $\Delta p_{s,0} = 0$. The compliance surplus after the i-th connection has been provisioned is defined as

$$\Delta p_{s,i} = p_i - p^*_{t,i}. \tag{4.5}$$

The new condition to accept a route during provisioning (an adaptation of (4.1)) is

$$p_i \geq p^*_{t,i}. \tag{4.6}$$

A flowchart of CAPSS is depicted in Figure 4.1. It is an extension of the flowchart in Figure 3.1.

As can be seen in (4.4) and (4.5), surplus is passed on from connection to connection. This is how the sharing principle is realized in CAPSS. The compliance target is relaxed by the amount of surplus other connections generate. Assuming that there are always connection requests that can "absorb" the generated surplus during relaxation, the average

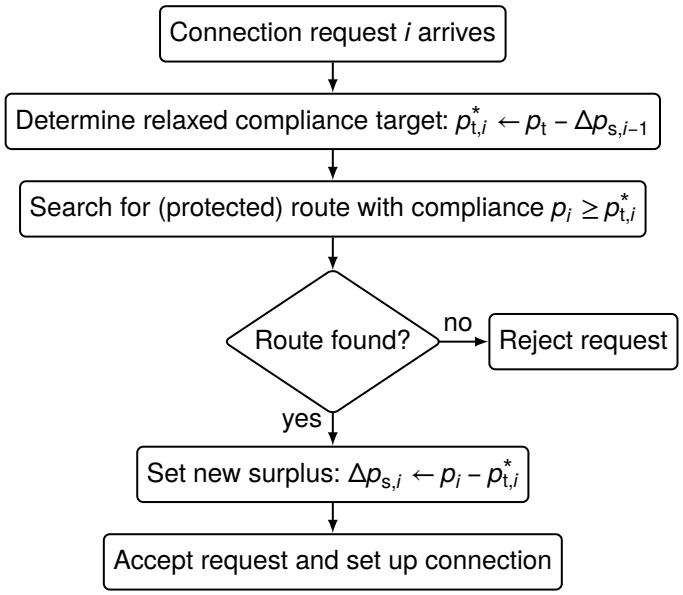


Figure 4.1 Basic structure of CAPSS

route compliance over all connections approaches the original operator-defined compliance target, i.e.,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} p_i = p_t. \tag{4.7}$$

That means, under the stated condition, CAPSS prevents overfulfillment.

Table 4.1 Example sequence of connection requests and surplus sharing – The compliance target is $p_t = 0.99$, and the initial surplus is $\Delta p_{s,0} = 0$.

Request number i	Relaxed target compliance $p^*_{t,i} = p_t - \Delta p_{s,i-1}$	Compliance of selected route p_i	Compliance surplus $\Delta p_{s,i} = p_i - p^*_{t,i}$
1	0.990	0.995	0.005
2	0.985	0.986	0.001
3	0.989	0.991	0.002

Table 4.1 presents an example sequence of connection requests. The compliance target is set to $p_t = 0.99$. The third column contains compliance probabilities of hypothetical routes. For the first connection request, no surplus is available and, hence, $p^*_{t,1} = p_t$. Suppose that the corresponding route has a computed compliance probability of $p_1 = 0.995$. Then, the resulting surplus is $\Delta p_{s,1} = 0.005$. This surplus is used to relax the compliance target of the second request. Therefore, the relaxed compliance target of the second request is only $p^*_{t,2} = 0.985$. Assume that the corresponding route has a compliance probability of $p_2 = 0.986$. Then, the new cumulated surplus is only $\Delta p_{s,2} = 0.001$. A major part of the surplus has been used to provision the second connection on a route with relaxed requirements. This route might not have been possible without surplus sharing. Assume that the route for the third connection has a compliance probability of $p_3 = 0.991$. Since $p_3 > p_t$, the connection does not make

use of the present surplus. As a result, the cumulated surplus grows to $\Delta p_{s,3} = 0.002$. The average compliance over all three connections is $(p_1 + p_2 + p_3)/3 = 0.9907$. Without surplus sharing, the lowest possible compliance probability for the second connection would be 0.99, which would lead to an average compliance of $0.992 > 0.9907$.

As can be seen, surplus sharing reduces compliance overfulfillment in this short example. More importantly, though, relaxing a connection's compliance requirement increases the number of suitable routes. On the one hand, protection can be reduced or even omitted entirely. On the other hand, longer routes can be selected. While less protection relieves the resource usage, the inclusion of longer paths increases the overall path diversity. Both effects potentially lead to an increased network capacity.

Now that the basic concept of CAPSS has been presented, we introduce the details about the system model, the mathematical model for the compliance probability, and the final provisioning algorithm itself.

4.2 Model Assumptions

Chapter 2 introduced the fundamentals of transport networks and connections and already made some assumptions. In this section, we reiterate the most important assumptions and add some more to allow a concise presentation of our provisioning mechanism. However, some of those assumptions can be lifted with little effort. Consequently, the mechanism can be extended in different ways, e.g., to support other protection schemes or to model network components in greater detail. Those examples are possible directions for future research.

4.2.1 Network Model

As introduced in Section 2.2, a communication network consists of a plethora of network components, like routers, switches, amplifiers, power supplies, and the like. Without loss of generality, we consider a compressed view of the network, modeling it as a set of nodes and links, where each node and each link comprises several of those network components. We denote both nodes and links as *network elements*. Often, nodes either have a high internal redundancy or are duplicated entirely and together form a single point of presence (PoP). For example, Cisco [Cis19a, Ch. 3] provides detailed information about the redundancy features of their ASR 9000 series routers. Gunkel and Horneffer [GH14] show an example of node duplication. As a consequence, a typical assumption in the literature is that nodes do not fail [Che+15; CGR14; KMO09]. While there are exceptions, e.g., [ZS15; Pán+06], we also assume that nodes *do not fail*. Therefore, the only network elements relevant for the availability analysis are the links. This assumption streamlines the derivation of the following mathematical models. However, an extension that considers node failures as well is easily possible. Mathematically, we represent the network as an undirected graph $G = (V, E)$, where nodes correspond to vertices in V, and links correspond to edges in E.

CAPSS provisions connections *end-to-end*, and for the surplus sharing functionality, the accumulated compliance surplus must be memorized. Furthermore, once a connection request arrives, CAPSS needs to know the remaining link capacities in order not to overload links. Also, CAPSS ensures that a connection works the moment it is deployed. Hence, once a connection request arrives, CAPSS needs to know which network elements are

working and which elements are under repair. Therefore, we require that CAPSS be run in a *logically* centralized entity. This could be, for example, a path computation element (PCE) [Muk+20, Sec. 15.2] or the provisioning module of a software-defined networking (SDN) controller [Muk+20, Sec. 15.3]. It is not possible to implement CAPSS with distributed routing protocols like intermediate system to intermediate system (IS-IS) or open shortest path first (OSPF) because they only allow a single end-to-end path between a node pair or multiple *equivalent* ones in the case of equal-cost multipath (ECMP) [MR07, Sec. 6.2.7]. However, CAPSS provisions connections on diverse routes because the compliance target is relaxed for every connection request individually. Different connections may have different delays this way. If it is required that different connections have the same or similar delays, additional restrictions must be added to the route search in CAPSS.

Network elements are subject to failures because the comprised hardware or software may fail. Generally, failure detection takes some time. However, we do not model this time explicitly. That means we assume that the network operator notices the failure the moment it occurs and immediately starts the repair process, in which the failed element is either repaired or replaced. Subsequently, the element is in working condition again and as good as new. We do not consider any intermediate states like partial performance degradation due to aging [BRV20]. Consequently, a network element can either be in the working state $\mathbb{W}$ or in the failed state $\mathbb{F}$, in which the repair process is active. Since element failures occur randomly, we assume that the duration of a working phase, i.e., the time between a successful repair or the beginning of network operation and the i-th element failure, can be modeled by a RV $T_{\mathbb{W},i} \in \mathbb{R}_{\geq 0}$. In the following, we denote $T_{\mathbb{W},i}$ as *uptime*. Similarly, we model the duration of the i-th repair phase as the RV $T_{\mathbb{F},i} \in \mathbb{R}_{\geq 0}$. We denote $T_{\mathbb{F},i}$ as *downtime*. As a result, the evolution of an element's state over time can be represented as in Figure 4.2.

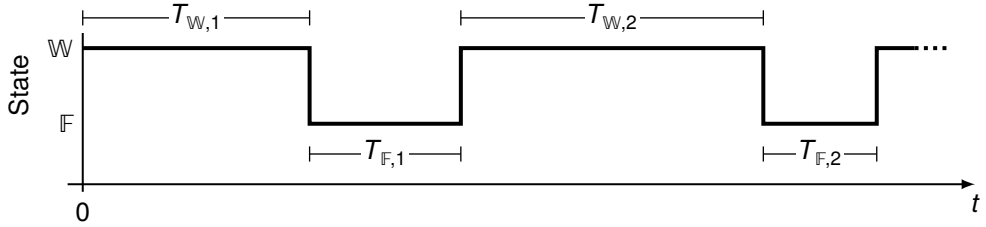


Figure 4.2 Temporal evolution of a network element's operating state

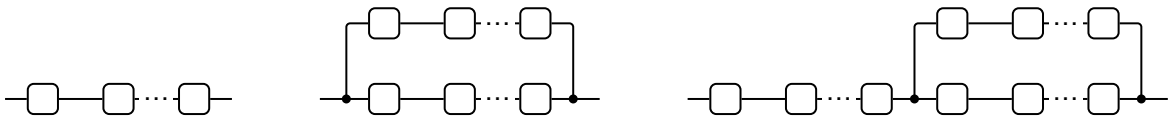
Like most authors (see Tables 3.1 and 3.2 in Section 3.4.4), we assume that all downtimes, $T_{\mathbb{F},i}$, of a particular network element j are independent and identically distributed (i.i.d.) and follow a negative exponential distribution with a repair rate μ_j. Likewise, we assume that the uptimes, $T_{\mathbb{W},i}$, of this network element are i.i.d. and follow a negative exponential distribution with a failure rate λ_j.

4.2.2 Connection Model

A network connection connects two nodes $v, w \in V$, $v \neq w$, bidirectionally with a certain data rate. It must fulfill certain performance characteristics, which are specified in the SLA between the network operator and the customer buying the connection. In the following, we consider as performance characteristic the guaranteed availability per billing cycle, α_{SLA}. The availability can be equivalently represented by the unavailability or the cumulated downtime per billing cycle. Since network elements can experience downtimes, network

connections experience downtimes as well. Based on the experienced unavailability in a billing cycle, the customer receives compensation payments according to the compensation policy specified in the SLA. As mentioned in Section 2.6.2.2, most operators only consider a connection unavailable once the customer opens a trouble ticket. However, we ignore this detail and assume that the unavailability the operator acknowledges equals the unavailability the customer actually experienced.

In the simplest case, a connection corresponds to a single unprotected path that traverses intermediate nodes and links to connect v and w. However, as discussed in Section 2.4, a connection can also be protected. We assume that the recovery time, i.e., the time needed to switch to the backup path, is negligible and does not add to the experienced downtime. In the following, we employ either dedicated path protection or dedicated segment protection, as introduced in Section 2.4.3. In either case, the primary path and the backup path are *link-disjoint*. Accordingly, we assume that a network connection can be represented by one of the block diagrams depicted in Figure 4.3. A block corresponds to a repairable link. We require that the block diagram be composed only of series and parallel structures.



(a) Unprotected **(b)** Path-protected **(c)** Segment-protected

Figure 4.3 Reliability block diagram representations of protected and unprotected connections – A block corresponds to a link.

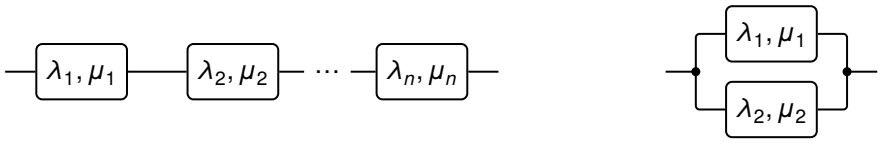
Section 2.6.1 mentions the lead time, i.e., the time it takes to provision a connection once the connection request has been issued. The provisioning mechanism we propose in this thesis does not involve time-consuming operations like mathematical optimizations for routing or reprovisioning of existing connections. Therefore, we assume that the lead time is negligible and a connection is provisioned or rejected the moment the connection request is issued by the customer.

4.3 Connection Failure and Repair Processes

A connection comprises multiple network elements. In order to describe the failure and repair process of a connection, we combine the failure and repair processes of the comprised network elements into a single, equivalent process.

In Section 4.2.1, we assumed that the up- and downtimes of a network element j follow negative exponential distributions with failure and repair rates λ_j and μ_j. Therefore, a connection can be modeled as a time-homogeneous Markov process [Bir17, p. 503]. Using the theory of such Markov processes, we can reduce a connection's complex reliability block diagram to an equivalent single-element block diagram with combined failure and repair rates λ and μ. This allows the calculation of the compliance probability using the methods that will be introduced in Section 4.4. In the following, we derive the steps for the block diagram reduction. We incorporate results from [Bir17]. In order for these results to be applicable, two more assumptions about the failure and repair processes will be made in the following.

As explained in Section 2.5.2, for the exponential case, the steady-state availability of the



(a) Series network elements **(b)** Parallel network elements

Figure 4.4 Reliability block diagrams of fundamental network element configurations

j-th network element depends on its failure and repair rate and is given by

$$a_j = \frac{\mu_j}{\lambda_j + \mu_j}. \tag{4.8}$$

For a series of n network elements, like in Figure 4.4a, we assume that no concurrent failures occur. Then, according to [Bir17, Sec. 6.3.1], the series can be reduced to a single element whose up- and downtimes follow exponential distributions again. The combined failure rate, λ_{series}, equals the sum of the individual failure rates [Bir17, Sec. 6.3.1 and Tab. 6.9], i.e.,

$$\lambda_{\text{series}} = \sum_{j=1}^{n} \lambda_j. \tag{4.9}$$

The combined steady-state availability is the product of the individual availabilities (see Section 2.5.3), i.e.,

$$a_{\text{series}} = \prod_{j=1}^{n} a_j. \tag{4.10}$$

Since (4.8) holds for the combined network element as well, we obtain for the combined repair rate

$$\mu_{\text{series}} = \lambda_{\text{series}} \frac{a_{\text{series}}}{1 - a_{\text{series}}}. \tag{4.11}$$

For two parallel elements, like in Figure 4.4b, we assume that only one repair crew is available. That means if both elements are down concurrently, the one that failed second can only be repaired after the first one is fixed. According to [Bir17, Ex. 6.6 and Tab. 6.9], the combined mean time to failure (MTTF) of the two elements is

$$MTTF_{\text{parallel}} = \frac{(\lambda_1 + \mu_2)(\lambda_2 + \mu_1) + \lambda_1(\lambda_1 + \mu_2) + \lambda_2(\lambda_2 + \mu_1)}{\lambda_1\lambda_2(\lambda_1 + \lambda_2 + \mu_1 + \mu_2)}. \tag{4.12}$$

The up- and downtimes of the combined element are *not* exponentially distributed. However, according to [Bir17, Ex. 6.6 and p. 199], they behave approximately exponentially distributed if $\lambda_1 \ll \mu_1$ and $\lambda_2 \ll \mu_2$. Since this condition is typically fulfilled in large transport networks, we assume in the following that the parallel structure has exponentially distributed up- and downtimes. Following [Bir17, Tab. 6.9], we set the constant failure rate to

$$\lambda_{\text{parallel}} \equiv \frac{1}{MTTF_{\text{parallel}}} \tag{4.13}$$

$$= \frac{\lambda_1\lambda_2(\lambda_1 + \lambda_2 + \mu_1 + \mu_2)}{(\lambda_1 + \mu_2)(\lambda_2 + \mu_1) + \lambda_1(\lambda_1 + \mu_2) + \lambda_2(\lambda_2 + \mu_1)}. \tag{4.14}$$

With Section 2.5.3, the combined availability is

$$a_{\text{parallel}} = 1 - (1 - a_1)(1 - a_2). \tag{4.15}$$

Similar to the series case, for the combined repair rate, we get

$$\mu_{\text{parallel}} = \lambda_{\text{parallel}} \frac{a_{\text{parallel}}}{1 - a_{\text{parallel}}}. \tag{4.16}$$

With repeated application of the presented transformations, a network connection's failure and repair rate can be calculated. For unprotected connections, a single series reduction of the involved network elements is sufficient. The route of a path-protected connection consists of two parallel series of elements. According to Section 2.4.3, the route of a segment-protected connection consists of an unprotected part represented by a series of elements and a protected part represented by two parallel series of elements (see Figure 4.5a).

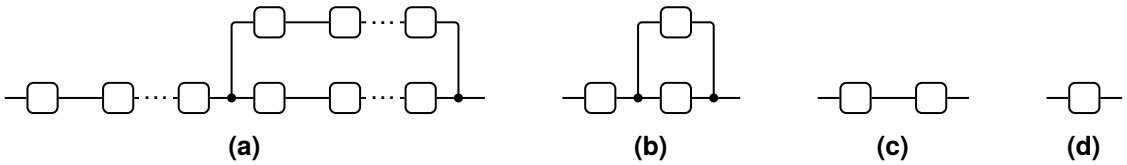


Figure 4.5 Reliability block diagram reduction of a segment-protected connection – **(a)** Initial block diagram **(b)** After reduction of all series structures **(c)** After reduction of the parallel structure **(d)** After reduction of the last series structure

An example of segment protection is shown in Figure 4.5. After reducing all series elements to a single element (Figure 4.5b), the parallel part is reduced to a single element as well (Figure 4.5c). Eventually, the two resulting series elements are reduced to the final equivalent element (Figure 4.5d). The process for path protection is mostly equal, except that only two parallel parts are involved. Whenever parallel structures are involved, the result is an approximation. Section 5.2.1 provides a simulative evaluation of the approximation error.

The presented reduction procedure is a core element of CAPSS. Prior to the computation of the compliance probability, the block diagram of a route is always reduced to an equivalent single-element diagram first. For this reason, the mathematical models we derive in the following only consider a single element with a failure rate λ and a repair rate μ.

4.4 Probability Distribution of a Connection's Compensation

CAPSS is a compensation-aware provisioning mechanism. As such, it computes the compliance probability of potential network routes during connection provisioning. For this computation, we need to know the distribution of the SLA compensation. Its derivation is the goal of this section. Since the compensation is a function of the connection (un)availability or downtime, respectively, we first derive the distribution of the *cumulated connection downtime* per billing cycle. Then, we extend this distribution to the distribution of the *SLA compensation* per billing cycle and per contract period.

4.4.1 Cumulated Connection Downtime

We first introduce general results about cumulated sojourn times from [Tak57]. Subsequently, we map those results to the use case at hand, namely the connection downtime per billing

cycle. Amongst other things, the mapping involves additional results from the theory of alternating renewal processes.

4.4.1.1 Cumulated Sojourn Time of an Alternating Two-State System

We first consider a two-state system, similar to that in Section 4.2.1. However, we assume general states $\mathbb{A}$ and $\mathbb{B}$ here. One might be tempted to map $\mathbb{B}$ to $\mathbb{F}$ from Section 4.2.1 and $\mathbb{A}$ to $\mathbb{W}$. However, this interpretation is not entirely correct for our use case, as we will see in this section.

We follow lines of argument and results from [Tak57]. Let $\xi(t)$ describe the state the system is in at time t. The system is in state $\mathbb{A}$ initially, i.e., $\xi(0) = \mathbb{A}$, and then alternates between the two states. Let $T_{\mathbb{A},i} \in \mathbb{R}_{\geq 0}$ $(i = 1, 2, 3, \ldots)$ denote the RVs that describe the uninterrupted durations the system spends in state $\mathbb{A}$. Assume that all $T_{\mathbb{A},i}$ are i.i.d. and share the same cumulative distribution function (CDF) $G(t)$. Similarly, let $T_{\mathbb{B},i} \in \mathbb{R}_{\geq 0}$ $(i = 1, 2, 3, \ldots)$ denote the RVs that describe the uninterrupted durations the system spends in state $\mathbb{B}$. Assume that all $T_{\mathbb{B},i}$ are i.i.d. and share the same CDF $H(t)$. An example of the temporal evolution of such a system is shown in Figure 4.6.

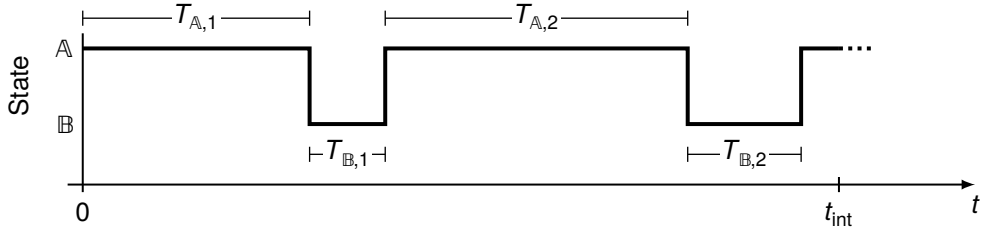


Figure 4.6 Temporal evolution of a two-state system's state with initial state $\mathbb{A}$

We are now interested in the cumulated amount of time the system spends in state $\mathbb{B}$ during the time interval $[0, t_{\text{int}}]$ with $t_{\text{int}} \in \mathbb{R}_{>0}$. This amount of time is represented by the RV B with

$$B = \int_0^{t_{\text{int}}} \mathbf{1}_{\xi(t)=\mathbb{B}} \, \mathrm{d}t \tag{4.17}$$

where $\mathbf{1}_x$ is the indicator function, which equals 1 if x is a true statement and 0 otherwise. In the following, we consider the distribution of B, i.e., $\mathbf{P}(B \leq b)$.

In Figure 4.6, we have two full sojourns in state $\mathbb{B}$. Consequently, the total time in state $\mathbb{B}$ is $B = T_{\mathbb{B},1} + T_{\mathbb{B},2}$. However, in general, the number of sojourns in state $\mathbb{B}$ is random, and, unlike in Figure 4.6, the system can also be in state $\mathbb{B}$ at t_{int}. In the latter case, only a part of the last sojourn time in state $\mathbb{B}$ is relevant for B.

Let us denote the sum of the first k sojourn times in state $\mathbb{B}$ as $S_{\mathbb{B},k} = \sum_{i=1}^{k} T_{\mathbb{B},i}$, and the sum of the first k sojourn times in state $\mathbb{A}$ as $S_{\mathbb{A},k} = \sum_{i=1}^{k} T_{\mathbb{A},i}$. If the system is in state $\mathbb{A}$ at t_{int} (as in Figure 4.6), the time interval contains n $(n = 0, 1, 2, \ldots)$ *full* $\mathbb{B}$ sojourns, and we have

$$S_{\mathbb{A},n} + S_{\mathbb{B},n} \leq t_{\text{int}} < S_{\mathbb{A},n+1} + S_{\mathbb{B},n}. \tag{4.18}$$

In this case, the event $B \leq b$ corresponds to $S_{\mathbb{B},n} \leq b$.

If, on the other hand, the system is in state $\mathbb{B}$ at t_{int}, the time interval contains m ($m = 1, 2, 3, \ldots$) full $\mathbb{A}$ sojourns, and it holds that

$$S_{\mathbb{A},m} + S_{\mathbb{B},m-1} \leq t_{\text{int}} < S_{\mathbb{A},m} + S_{\mathbb{B},m}. \tag{4.19}$$

The event $B \leq b$ then corresponds to $t_{\text{int}} - S_{\mathbb{A},m} \leq b$, which is equivalent to $S_{\mathbb{A},m} \geq t_{\text{int}} - b$. If all possible values of n and m are considered, then, according to [Tak57], the probability of $B \leq b$, i.e., the CDF of B, is

$$\begin{aligned} \mathbf{P}(B \leq b) = &\sum_{n=0}^{\infty} \mathbf{P}(S_{\mathbb{B},n} \leq b, S_{\mathbb{A},n} + S_{\mathbb{B},n} \leq t_{\text{int}} < S_{\mathbb{A},n+1} + S_{\mathbb{B},n}) + \\ &\sum_{m=1}^{\infty} \mathbf{P}(S_{\mathbb{A},m} \geq t_{\text{int}} - b, S_{\mathbb{A},m} + S_{\mathbb{B},m-1} \leq t_{\text{int}} < S_{\mathbb{A},m} + S_{\mathbb{B},m}) \end{aligned} \tag{4.20}$$

$$\equiv F_B(b). \tag{4.21}$$

In (4.20), the probabilities are *joint probabilities* with the individual events separated by a comma. It is also shown in [Tak57], that (4.20) is equivalent to

$$F_B(b) = \sum_{n=0}^{\infty} \mathbf{P}(S_{\mathbb{B},n} \leq b) \cdot \mathbf{P}(S_{\mathbb{A},n} < t_{\text{int}} - b \leq S_{\mathbb{A},n+1}) \tag{4.22}$$

$$= \sum_{n=0}^{\infty} H_n(b) \cdot \big(G_n(t_{\text{int}} - b) - G_{n+1}(t_{\text{int}} - b)\big) \tag{4.23}$$

where $G_n(t)$ and $H_n(t)$ are the respective n-th iterated convolutions of the CDFs $G(t)$ and $H(t)$ with themselves, e.g., $G_n(t) = \int_{\tau=0}^{t} G_{n-1}(t-\tau)\,\mathrm{d}G(\tau)$.

If all sojourn times follow negative exponential distributions, i.e.,

$$G(t;\alpha) = \begin{cases} 1 - e^{-\alpha t} & \text{for } t \geq 0 \\ 0 & \text{for } t < 0 \end{cases} \tag{4.24}$$

with rate parameter $\alpha \in \mathbb{R}_{>0}$, and

$$H(t;\beta) = \begin{cases} 1 - e^{-\beta t} & \text{for } t \geq 0 \\ 0 & \text{for } t < 0 \end{cases} \tag{4.25}$$

with rate parameter $\beta \in \mathbb{R}_{>0}$ then, according to [Tak57], the CDF for $b \geq 0$ becomes

$$F_B(b; t_{\text{int}}, \alpha, \beta) = e^{-\alpha(t_{\text{int}}-b)}\left(1 + \sqrt{\alpha\beta(t_{\text{int}} - b)} \cdot \int_0^b e^{-\beta y} y^{-\frac{1}{2}} I_1\left(2\sqrt{\alpha\beta(t_{\text{int}} - b)y}\right)\mathrm{d}y\right) \tag{4.26}$$

where $I_1(x)$ is the modified Bessel function of the first kind of order 1. Since B is a period of time, we have $F_B(b) = 0$ for $b < 0$.

To summarize, $F_B(b; t_{\text{int}}, \alpha, \beta)$ in (4.26) is the CDF of the RV B for the exponential case. B is the cumulated time the system spends in state $\mathbb{B}$ during a time interval $[0, t_{\text{int}}]$ when the system starts in state $\mathbb{A}$. The times between state changes follow exponential distributions with rate parameters α and β. If the parameters are not of special interest, we denote (4.26) by $F_B(b)$ without the parameter list in the following.

4.4.1.2 Properties of the Cumulated Sojourn Time

As mentioned before, B is a period of time and cannot be negative. Therefore,

$$\mathbf{P}(B < 0) = 0. \tag{4.27}$$

We assume that the system is in state $\mathbb{A}$ at the beginning of the considered time interval. It is possible that no state transition occurs during the whole time interval $[0, t_{\text{int}}]$. In that case, the cumulated time in state $\mathbb{B}$ is zero, i.e., $B = 0$. The event of no state transition during the time interval $[0, t_{\text{int}}]$ corresponds to the event that the first transition occurs at the end of the time interval or at an arbitrary point in time thereafter. Therefore, we have

$$\begin{aligned}\mathbf{P}(B = 0) &= \mathbf{P}(T_{\mathbb{A},1} \geq t_{\text{int}}) && (4.28)\\ &= 1 - \mathbf{P}(T_{\mathbb{A},1} < t_{\text{int}}) && (4.29)\\ &= e^{-\alpha t_{\text{int}}} && (4.30)\end{aligned}$$

where the step from (4.29) to (4.30) is a consequence of $T_{\mathbb{A},1}$ following an exponential distribution. As can be seen, $\mathbf{P}(B = 0)$ is non-zero and, therefore, represents a point mass. As a result, the RV B is a mixed RV consisting of a discrete part at $b = 0$ and a continuous part for $b \in (0, t_{\text{int}}]$. Consequently, since a CDF is always right-continuous [CK04, Prop. 4.30], $F_B(b)$ has the following properties:

- $\lim\limits_{b \nearrow 0} F_B(b; t_{\text{int}}, \alpha, \beta) = 0$
- $F_B(0; t_{\text{int}}, \alpha, \beta) = e^{-\alpha t_{\text{int}}}$.

The notation $\lim_{b \nearrow 0}$ denotes the limit as b approaches 0 from the left or from below, respectively. An alternative notation of the first property is $F_B(0^-) = 0$. The second property is also evident from (4.26), namely,

$$\begin{aligned}F_B(0; t_{\text{int}}, \alpha, \beta) &= e^{-\alpha t_{\text{int}}}\left(1 + \sqrt{\alpha\beta t_{\text{int}}} \cdot \int_0^0 e^{-\beta y} y^{-\frac{1}{2}} I_1\left(2\sqrt{\alpha\beta t_{\text{int}} y}\right) \mathrm{d}y\right) && (4.31)\\ &= e^{-\alpha t_{\text{int}}} && (4.32)\end{aligned}$$

since the integral is zero.

As an example, Figure 4.7 shows six different parametrizations of the CDF. The parameter values are given in terms of the rate parameter α. The interval duration, t_{int}, takes values in $0.1/\alpha$, $1/\alpha$, and $10/\alpha$. The rate β equals either α or 9α. Note that the parameter values serve to illustrate the properties of the CDF and do not necessarily correspond to values found in practical scenarios. The x-axis is normalized to the respective interval duration. For values on the x-axis below 0 and above 1, all curves coincide.

The jump discontinuities at $b = 0$ illustrate the mixed nature of the distribution. It can be seen that the probability mass is increasingly concentrated around the point $b/t_{\text{int}} = \alpha/(\alpha+\beta)$ (the two dashed vertical lines in Figure 4.7) as the interval duration grows from $0.1/\alpha$ to $10/\alpha$. In the asymptotic case, i.e., for $t_{\text{int}} \to \infty$, the distribution degenerates to a deterministic distribution with the single value of $t_{\text{int}} \cdot \alpha/(\alpha+\beta)$. It can also be seen that the distributions for $t_{\text{int}} = 0.1/\alpha$ are very similar for both $\beta = \alpha$ and $\beta = 9\alpha$. The reason is that it is likely that the system stays in state $\mathbb{A}$ during the entire time interval because, with $t_{\text{int}} = 0.1/\alpha = 0.1 \cdot \mathbf{E}(T_{\mathbb{A},1})$, the expected time in state $\mathbb{A}$ is much higher than the interval duration itself. The actual probability for this to happen is $\mathbf{P}(B = 0) = e^{-0.1} = 0.905$, as can also be seen in Figure 4.7. Consequently, the rate parameter β of state $\mathbb{B}$ does not impact the distribution much in this particular scenario.

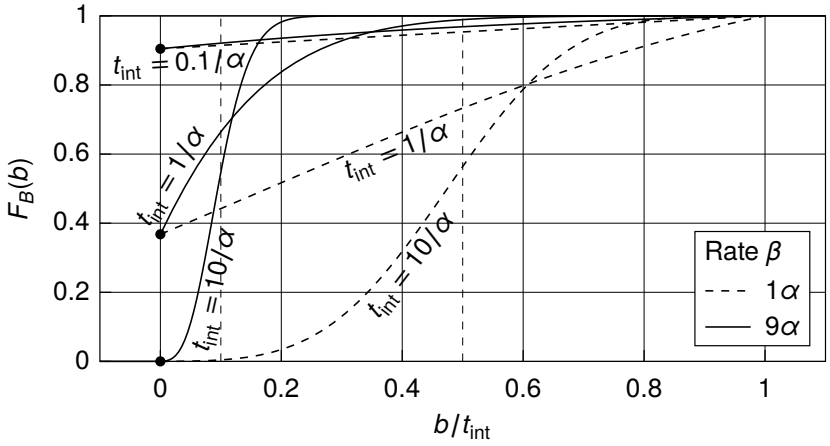


Figure 4.7 Illustration of the CDF $F_B(b)$ – The parameter values are $t_{\text{int}} \in \{0.1/\alpha, 1/\alpha, 10/\alpha\}$ and $\beta \in \{\alpha, 9\alpha\}$. For values on the x-axis below 0 and above 1, all curves coincide.

4.4.1.3 Cumulated Connection Downtime per Billing Cycle

We now use the results presented in Section 4.4.1.1 as a basis to describe the cumulated downtime per billing cycle of a network connection as introduced in Sections 2.6 and 4.2.2. While Section 4.4.1.1 mostly reproduced the work of Takács [Tak57], the following sections show the work of this monograph's author.

We assume that a connection is in the network for the duration of several billing cycles, which make up the contract period, Δt_{h}. As an example, Figure 4.8 shows the whole contract period of a connection. Except for the first and the last billing cycle, we assume that a billing cycle is always aligned with a calendar month. Since the connection may start and end at an arbitrary point in time, the first and the last billing cycle may be shorter than a month. In Section 4.2.1, we denoted the working state by $\mathbb{W}$, the failed state by $\mathbb{F}$, and the corresponding times in those states by $T_{\mathbb{W},i}$ and $T_{\mathbb{F},i}$. In the following, we use the same notation but refer to a whole connection, not only a network element.

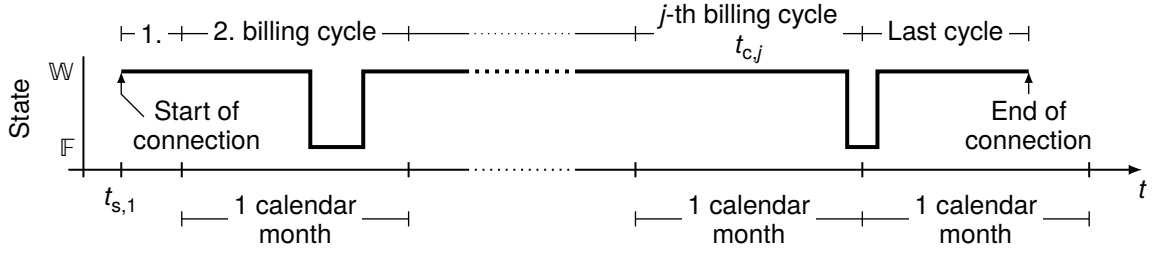


Figure 4.8 Temporal evolution of a network connection's state – The contract period is j months and contains $j + 1$ billing cycles.

In the following, we study an arbitrary billing cycle, namely the j-th. We denote the duration of this billing cycle by $t_{\text{c},j} \in \mathbb{R}_{>0}$. We are interested in the distribution $\mathbf{P}(X_j \leq x)$ of the RV $X_j \in \mathbb{R}_{\geq 0}$, which describes the cumulated downtime in the j-th billing cycle. An approximated solution is easily obtained by mapping the states $\mathbb{A}$ and $\mathbb{B}$ of the introduced two-state system to the states $\mathbb{W}$ and $\mathbb{F}$ of the network connection and the random variable B to X_j. However, this mapping overlooks several details. To get a more precise solution, some additional reasoning as well as some adjustments are necessary and will be discussed in the following.

For (4.26) to hold, two important conditions must be fulfilled:

1. All sojourn times $T_{\mathbb{A},i}$ of state $\mathbb{A}$ must be identically distributed. Similarly, all sojourn times $T_{\mathbb{B},i}$ of state $\mathbb{B}$ must be identically distributed. In particular, the first sojourn time in a billing cycle must follow the same distribution as subsequent sojourn times of the respective state.
2. The system must be in state $\mathbb{A}$ at the beginning of the considered time interval.

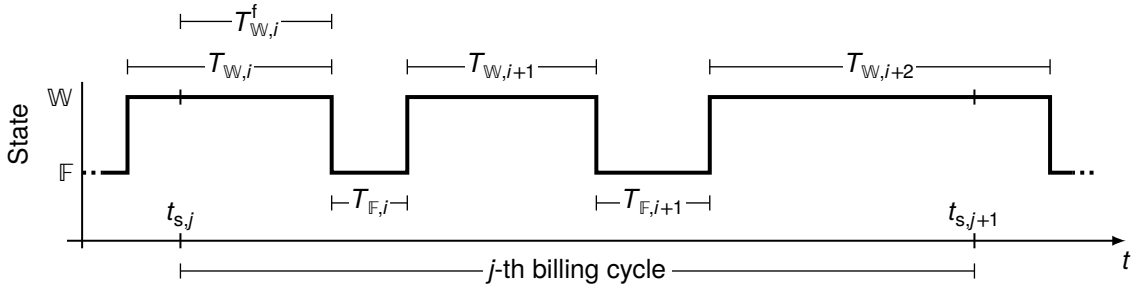


Figure 4.9 Connection up- and downtimes in a billing cycle

For the first point, consider Figure 4.9, which shows a possible evolution of a connection for the j-th billing cycle. At the beginning of the billing cycle, at time $t_{\mathrm{s},j}$, the connection is in state $\mathbb{W}$. However, it has already been in state $\mathbb{W}$ in the previous billing cycle. Hence, the time until the next failure occurs, as seen from the beginning of the billing cycle, is not $T_{\mathbb{W},i}$, but only a part of it. Denote this part by $T^{\mathrm{f}}_{\mathbb{W},i}$ (see Figure 4.9). It needs to be shown that $T^{\mathrm{f}}_{\mathbb{W},i}$ follows the same distribution as $T_{\mathbb{W},i}$. To that end, notice that the connection model corresponds to an *alternating renewal process* [Bir17, A7.3] in which $T^{\mathrm{f}}_{\mathbb{W},i}$ corresponds to the *forward recurrence time*, i.e., the residual time until the next failure. Assume that the CDF of $T_{\mathbb{W},i}$ is $F(t)$. Then, according to [Bax81], the probability density function (PDF) $f_{\mathrm{f}}(t)$ of $T^{\mathrm{f}}_{\mathbb{W},i}$ is

$$f_{\mathrm{f}}(t) = \frac{1 - F(t)}{\mathbf{E}(T_{\mathbb{W},i})}. \tag{4.33}$$

As stated above, we assume that $T_{\mathbb{W},i}$ follows a negative exponential distribution with CDF $F(t) = 1 - e^{-\lambda t}$, PDF $f(t) = \lambda e^{-\lambda t}$, and $\mathbf{E}(T_{\mathbb{W},i}) = 1/\lambda$. Therefore, we get

$$f_{\mathrm{f}}(t) = \lambda e^{-\lambda t} \tag{4.34}$$

$$= f(t). \tag{4.35}$$

Hence, $T^{\mathrm{f}}_{\mathbb{W},i}$ and $T_{\mathbb{W},i}$ share the same distribution. The result can be transferred to the failed state $\mathbb{F}$ and $T_{\mathbb{F},i}$ as well. Consequently, the first condition is fulfilled.

The second condition requires that the system is in state $\mathbb{A}$ at the beginning of the considered time interval, which is the duration of a billing cycle, $t_{\mathrm{c},j}$, now. A naive approach that simply maps $\mathbb{A}$ to $\mathbb{W}$, $\mathbb{B}$ to $\mathbb{F}$, and B to X_j falls short because it is not guaranteed that a connection works at the beginning of a billing cycle. Our provisioning algorithm will ensure that a network connection works at the beginning of the first billing cycle when the connection is newly established, tested, and then handed over to the customer. However, for subsequent billing cycles, the connection might be broken during the transition from one billing cycle to the next. This is illustrated in the last billing cycle in Figure 4.8. Therefore, in the first step, we consider two separate cases, which we unify in the second step. The first

case assumes that, indeed, the connection works at the beginning of the billing cycle. We indicate this by $\xi_j = \xi(t_{\mathrm{s},j}) = \mathbb{W}$. We map $\mathbb{A}$ to $\mathbb{W}$, $\mathbb{B}$ to $\mathbb{F}$, and t_{int} to $t_{\mathrm{c},j}$, and obtain

$$\mathbf{P}(X_j \leq x | \xi_j = \mathbb{W}) = \mathbf{P}(B \leq x) \tag{4.36}$$
$$= F_B^{\mathbb{W}}(x) \tag{4.37}$$

where the $\mathbb{W}$ in $F_B^{\mathbb{W}}(x)$ signifies that the state $\mathbb{W}$ is used as state $\mathbb{A}$. The second case assumes that the connection does *not* work at the beginning of the billing cycle. Consequently, the connection is in state $\mathbb{F}$. In this case, we map $\mathbb{A}$ to $\mathbb{F}$, $\mathbb{B}$ to $\mathbb{W}$, and again t_{int} to $t_{\mathrm{c},j}$. With this mapping, B describes the cumulated *uptime* in the billing cycle. Since the sum of up- and downtime is equal to the full billing cycle duration, the downtime is $t_{\mathrm{c},j} - B$. Consequently, we have

$$\mathbf{P}(X_j \leq x | \xi_j = \mathbb{F}) = \mathbf{P}(t_{\mathrm{c},j} - B \leq x) \tag{4.38}$$
$$= \mathbf{P}(B \geq t_{\mathrm{c},j} - x) \tag{4.39}$$
$$= 1 - \mathbf{P}(B < t_{\mathrm{c},j} - x). \tag{4.40}$$

This can be expressed in terms of the CDF of B as

$$\mathbf{P}(X_j \leq x | \xi_j = \mathbb{F}) = 1 - \lim_{y \nearrow (t_{\mathrm{c},j} - x)} F_B^{\mathbb{F}}(y) \tag{4.41}$$
$$= 1 - F_B^{\mathbb{F}}\left((t_{\mathrm{c},j} - x)^-\right) \tag{4.42}$$

As before, the $\mathbb{F}$ in $F_B^{\mathbb{F}}(y)$ signifies that the state $\mathbb{F}$ corresponds to the state $\mathbb{A}$. The limit in (4.41) is necessary because a CDF is always right-continuous, i.e., $F_B(b) = \mathbf{P}(B \leq b)$ corresponds to the probability that B is *less than or equal to* b. However, in (4.40), we only ask for the probability that B is *less than* $t_{\mathrm{c},j} - x$, without the possibility of equality.

To unify both cases, we use the law of total probability and obtain

$$\mathbf{P}(X_j \leq x) = \sum_{i \in \{\mathbb{W}, \mathbb{F}\}} \mathbf{P}(X_j \leq x | \xi_j = i) \cdot \mathbf{P}(\xi_j = i) \tag{4.43}$$
$$= \mathbf{P}(X_j \leq x | \xi_j = \mathbb{W}) \cdot \mathbf{P}(\xi_j = \mathbb{W}) + \mathbf{P}(X_j \leq x | \xi_j = \mathbb{F}) \cdot \mathbf{P}(\xi_j = \mathbb{F}). \tag{4.44}$$

To evaluate this total probability, we need to find the probabilities $\mathbf{P}(\xi_j = \mathbb{W})$ and $\mathbf{P}(\xi_j = \mathbb{F})$. As stated before, we ensure that the connection works at time $t_{\mathrm{s},1}$ when it is deployed (see Figure 4.8). Consequently, $\mathbf{P}(\xi_1 = \mathbb{W}) = 1$. A subsequent, j-th billing cycle is started at time $t_{\mathrm{s},j}$ (see Figure 4.9). The probability of starting the j-th billing cycle in the working state, i.e., $\mathbf{P}(\xi_j = \mathbb{W})$, corresponds to the *point availability*. In Section 2.5.1, we defined the point availability $a(t)$ for a system starting operation at $t = 0$. Here, the connection only comes into existence at $t_{\mathrm{s},1}$. At $t_{\mathrm{s},j}$, it is only $\Delta t_{\mathrm{s},j} = t_{\mathrm{s},j} - t_{\mathrm{s},1}$ "old." Therefore, we consider the point availability relative to $t_{\mathrm{s},1}$ and denote it by $a(\Delta t_{\mathrm{s},j})$. Given that the connection has been working at the time of its deployment and that the up- and downtimes are exponentially distributed, the point availability is (see Section 2.5.2)

$$a(\Delta t_{\mathrm{s},j}) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{\lambda + \mu} e^{-(\lambda+\mu)\Delta t_{\mathrm{s},j}}. \tag{4.45}$$

The applicability of (4.45) is, once again, justified by the fact that the forward recurrence time at $t_{\mathrm{s},1}$ follows a negative exponential distribution with rate parameter λ, and, hence, is equal to the distribution of the subsequent connection uptimes.

For the first billing cycle, $a(0) = 1$, as expected. The limit of the point availability is the steady-state availability, namely

$$\lim_{\Delta t_{\mathrm{s},j} \to \infty} a(\Delta t_{\mathrm{s},j}) = \frac{\mu}{\lambda + \mu}. \tag{4.46}$$

Hence, for a connection with a long contract period and many billing cycles, the probability of finding a connection working at the beginning of a billing cycle converges to the steady-state availability of the connection over time.

Since the states $\mathbb{W}$ and $\mathbb{F}$ are mutually exclusive,

$$\mathbf{P}(\xi_j = \mathbb{F}) = 1 - \mathbf{P}(\xi_j = \mathbb{W}) \tag{4.47}$$

$$= 1 - a(\Delta t_{\mathrm{s},j}). \tag{4.48}$$

Hence, for the CDF of X_j, we obtain

$$\mathbf{P}(X_j \leq x) = a(\Delta t_{\mathrm{s},j}) \cdot \mathbf{P}(X_j \leq x | \xi_j = \mathbb{W}) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \mathbf{P}(X_j \leq x | \xi_j = \mathbb{F}) \tag{4.49}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot F_B^{\mathbb{W}}(x) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - F_B^{\mathbb{F}}\big((t_{\mathrm{c},j} - x)^-\big)\Big). \tag{4.50}$$

If $\mathbb{A}$ is mapped to $\mathbb{W}$, as in $F_B^{\mathbb{W}}(x)$, then α corresponds to λ, and β corresponds to μ. In contrast, if state $\mathbb{A}$ is mapped to $\mathbb{F}$, as in $F_B^{\mathbb{F}}\big((t_{\mathrm{c},j} - x)^-\big)$, then α corresponds to μ, and β corresponds to λ. Therefore,

$$\mathbf{P}(X_j \leq x) = a(\Delta t_{\mathrm{s},j}) \cdot F_B(x; t_{\mathrm{c},j}, \lambda, \mu) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - F_B\big((t_{\mathrm{c},j} - x)^-; t_{\mathrm{c},j}, \mu, \lambda\big)\Big) \tag{4.51}$$

$$\equiv F_{X_j}(x; t_{\mathrm{c},j}, \lambda, \mu, \Delta t_{\mathrm{s},j}). \tag{4.52}$$

To summarize, X_j is the cumulated downtime in the j-th billing cycle of a connection. $F_{X_j}(x)$ is its CDF. The billing cycle duration, $t_{\mathrm{c},j}$, is typically one month but can be shorter for the first and last billing cycles. The parameter λ is the constant failure rate of the connection, and μ is the constant repair rate. Lastly, $\Delta t_{\mathrm{s},j}$ denotes the time between the start of the j-th billing cycle and the start of the connection.

4.4.1.4 Properties of the Cumulated Connection Downtime

Since X_j is based on B, both share similar properties. X_j is a mixed RV. However, in contrast to B, X_j has *two* point masses, one at $x = 0$ and one at $x = t_{\mathrm{c},j}$. In the following, we derive and illustrate those point masses from a probabilistic perspective. A derivation directly based on the CDF $F_{X_j}(x)$ in (4.52) can be found in Appendix A.1.

The point mass at $x = 0$ corresponds to the probability that the connection is working at the beginning of the billing cycle and does not fail during the cycle. As mentioned above, the probability of starting in the working state corresponds to the point availability, i.e., $\mathbf{P}(\xi_j = \mathbb{W}) = a(\Delta t_{\mathrm{s},j})$. According to Section 4.4.1.2, the probability of no failure, i.e., no state transition, given that the connection starts the billing cycle in the working state, is $\mathbf{P}(X_j = 0 | \xi_j = \mathbb{W}) = e^{-\lambda t_{\mathrm{c},j}}$. In contrast, the probability of no downtime, given that the connection already starts the billing cycle in the failed state, is zero, i.e., $\mathbf{P}(X_j = 0 | \xi_j = \mathbb{F}) = 0$. Hence, with the law of total probability, we obtain

$$\mathbf{P}(X_j = 0) = a(\Delta t_{\mathrm{s},j}) \cdot e^{-\lambda t_{\mathrm{c},j}}. \tag{4.53}$$

Since $\mathbf{P}(X_j = 0) > 0$, the CDF has a jump discontinuity at $x = 0$, and with $\mathbf{P}(X_j < 0) = F_{X_j}(0^-) = 0$ we get

$$F_{X_j}(0) = \mathbf{P}(X_j = 0) \tag{4.54}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot e^{-\lambda t_{\mathrm{c},j}}. \tag{4.55}$$

The point mass at $x = t_{\mathrm{c},j}$ corresponds to the probability that the connection does *not* work at the beginning of the billing cycle and does not get repaired during the cycle. The probability of starting in the failed state corresponds to the *point unavailability*, i.e., $\mathbf{P}(\xi_j = \mathbb{F}) = 1 - a(\Delta t_{\mathrm{s},j})$. The probability that the repair process is not finished during the billing cycle, i.e., no state transition occurs, given that the connection starts the billing cycle in the failed state is $\mathbf{P}(X_j = t_{\mathrm{c},j} | \xi_j = \mathbb{F}) = e^{-\mu t_{\mathrm{c},j}}$. In contrast, the probability of no uptime, given that the connection already starts the billing cycle in the working state, is zero, i.e., $\mathbf{P}(X_j = t_{\mathrm{c},j} | \xi_j = \mathbb{W}) = 0$. Hence, with the law of total probability, we obtain

$$\mathbf{P}(X_j = t_{\mathrm{c},j}) = \left(1 - a(\Delta t_{\mathrm{s},j})\right) \cdot e^{-\mu t_{\mathrm{c},j}}. \tag{4.56}$$

As before, since $\mathbf{P}(X_j = t_{\mathrm{c},j}) > 0$, the CDF has a jump discontinuity at $x = t_{\mathrm{c},j}$. Since $\mathbf{P}(X_j > t_{\mathrm{c},j}) = 0$, we have

$$F_{X_j}(t_{\mathrm{c},j}) = 1. \tag{4.57}$$

Consequently,

$$F_{X_j}(t_{\mathrm{c},j}^-) = 1 - \mathbf{P}(X_j = t_{\mathrm{c},j}) \tag{4.58}$$

$$= 1 - \left(1 - a(\Delta t_{\mathrm{s},j})\right) \cdot e^{-\mu t_{\mathrm{c},j}}. \tag{4.59}$$

In Section 2.5.2, it was explained that the point availability converges exponentially to the steady-state availability with a time constant of $\tau = 1/(\lambda + \mu) \approx 1/\mu = MTTR$. After a time span of about $5 \cdot MTTR$ — which is much less than a month in practical scenarios — the point availability has converged almost completely. Consequently, the probability of starting the second billing cycle of a connection in the working state is already well described by the steady-state availability. Therefore, we approximate the point availability, $a(\Delta t_{\mathrm{s},j})$, with the steady-state availability, $a = \mu/(\lambda + \mu)$, in the following example.

Similar to Figure 4.7, Figure 4.10 shows six different parametrizations of the CDF. The parameter values are given in terms of the failure rate λ. The interval duration, $t_{\mathrm{c},j}$, takes values in $0.1/\lambda$, $1/\lambda$, and $10/\lambda$. The repair rate μ equals either λ or 9λ. As before, the values serve to illustrate the properties of the CDF and do not represent values found in practical scenarios. The x-axis is normalized to the respective interval duration. As can be seen, the CDF has jump discontinuities at $x = 0$ and $x = t_{\mathrm{c},j}$. For a repair rate $\mu = \lambda$, the CDF is point-symmetric around the point $(0.5, 0.5)$ because $F_B^{\mathbb{W}}(x)$ and $F_B^{\mathbb{F}}(x)$ are effectively parametrized equally. However, this is only the case if we approximate the point availability with the steady-state availability because then $\mathbf{P}(\xi_j = \mathbb{W}) = \mathbf{P}(\xi_j = \mathbb{F}) = a = 0.5$. In real-life scenarios with $\mu \gg \lambda$, the point mass at $x = 0$ is typically much larger than that at $x = t_{\mathrm{c},j}$. If, additionally, $t_{\mathrm{c},j}$ is not too small, the point mass at $x = t_{\mathrm{c},j}$ tends to vanish.

4.4.2 Compensation

As explained in Section 2.6.2.2, the amount of compensation a network operator has to pay at the end of a billing cycle is a function of the connection unavailability or, equivalently,

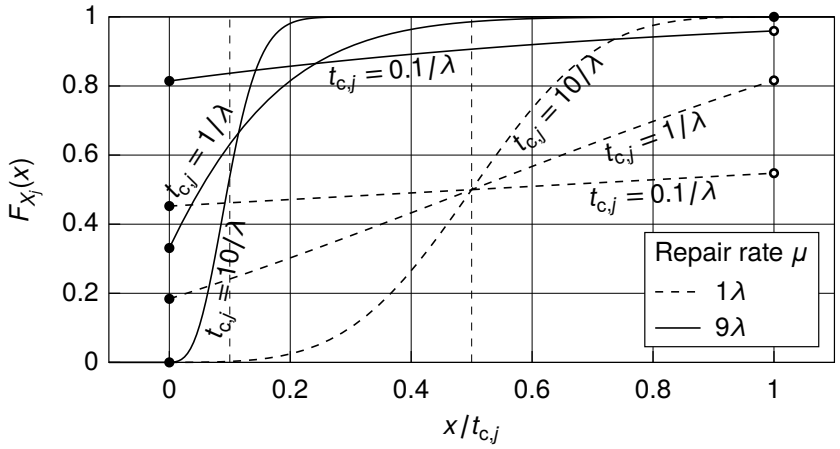


Figure 4.10 Illustration of the CDF $F_{X_j}(x)$ – The parameter values are $t_{c,j} \in \{0.1/\lambda, 1/\lambda, 10/\lambda\}$ and $\mu \in \{\lambda, 9\lambda\}$. For values on the x-axis below 0 and above 1, all curves coincide. Solid dots represent included values, while open dots represent excluded values.

the cumulated connection downtime during that billing cycle. In the previous section, the distribution of the cumulated downtime per billing cycle was derived. In the following, we derive the distribution of the compensation for the staircase compensation policy introduced in Section 2.6.2.2. We first consider the case of a single billing cycle and then extend it to a full contract period. This last extension yields the final model for the distribution of SLA compensation over a connection's contract period.

4.4.2.1 Compensation per Billing Cycle

Considering a single network connection, we define the RV C_j to describe the amount of compensation the network operator has to pay its customer at the end of the j-th billing cycle. The compensation is a function of the cumulated downtime, X_j. The mapping is realized by the compensation policy defined in the SLA. As mentioned earlier, a connection may start and end at an arbitrary point in time. Therefore, the first and the last billing cycle may be shorter than a month. In order to keep the compensation policy independent of the actual billing cycle duration, $t_{c,j}$, we model the policy function as a function of the unavailability in the billing cycle. The unavailability is defined as

$$U_j = \frac{X_j}{t_{c,j}}. \tag{4.60}$$

Further, we assume that the compensation policy is a staircase function with a discrete range $\mathcal{C}_1 \equiv \{0, c_1, c_2, \ldots, c_n\} \subset \mathbb{R}_{\geq 0}$ consisting of the compensation levels defined in the SLA. Therefore, we define the policy function as $g : [0,1] \to \mathcal{C}_1$ with

$$g(u) = \begin{cases} 0 & \text{for } u \leq u_1 \\ c_i & \text{for } u_i < u \leq u_{i+1} \qquad 1 \leq i \leq n \end{cases} \tag{4.61}$$

and $u_{n+1} = 1$. An example policy is depicted in Figure 4.11. In total, we have the relationship

$$C_j = g(U_j) \tag{4.62}$$

$$= g\left(\frac{X_j}{t_{c,j}}\right). \tag{4.63}$$

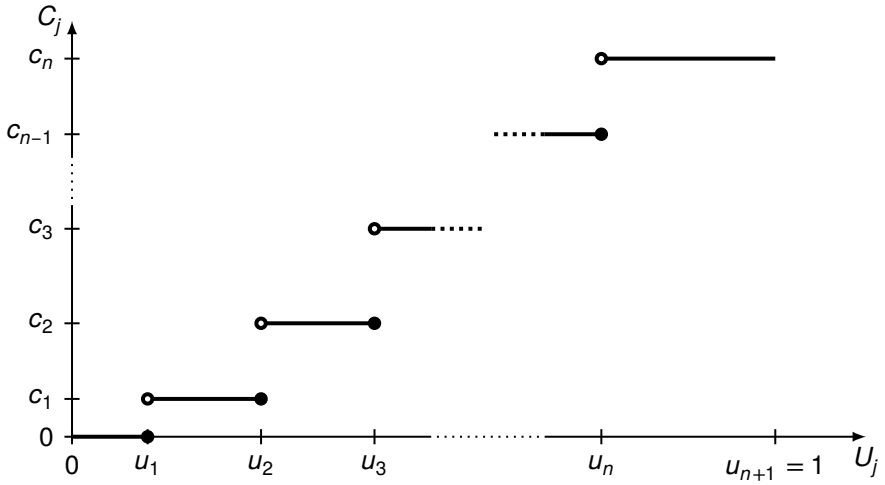


Figure 4.11 Generic SLA compensation policy function $g(u)$

The support[1] of C_j is $\mathcal{C}_1$. Since $\mathcal{C}_1$ is a finite set, C_j is a discrete RV. The individual probabilities $\mathbf{P}(C_j = c)$ can be derived from the distribution of the cumulated downtime. Generally, we get

$$\mathbf{P}(C_j = c) = \mathbf{P}\Big(g\Big(X_j/_{t_{c,j}}\Big) = c\Big) \tag{4.64}$$

$$= \mathbf{P}\Big(X_j/_{t_{c,j}} \in g^{-1}(c)\Big). \tag{4.65}$$

Here, $g^{-1}(c)$ is the *preimage* of the policy function, i.e., $g^{-1}(c) = \{u \in [0,1] : g(u) = c\}$. The policy function has discontinuities at the unavailabilities u_i $(1 \leq i \leq n)$. The function jumps to the compensation level $c_i > 0$ at the unavailability u_i. Since g is left-continuous at the discontinuities, the preimage $g^{-1}(c_i)$ for a *non-zero* compensation level equals the half-open interval $(u_i, u_{i+1}]$. Consequently, we get

$$\mathbf{P}(C_j = c_i) = \mathbf{P}\Big(X_j/_{t_{c,j}} \in (u_i, u_{i+1}]\Big) \tag{4.66}$$

$$= \mathbf{P}\Big(u_i < X_j/_{t_{c,j}} \leq u_{i+1}\Big) \tag{4.67}$$

$$= \mathbf{P}\big(u_i \cdot t_{c,j} < X_j \leq u_{i+1} \cdot t_{c,j}\big). \tag{4.68}$$

The last probability can be computed with the CDF of X_j, namely

$$\mathbf{P}(C_j = c_i) = F_{X_j}(u_{i+1} \cdot t_{c,j}) - F_{X_j}(u_i \cdot t_{c,j}). \tag{4.69}$$

For $C_j = 0$, the preimage corresponds to the closed interval $[0, u_1]$ because we have to include the case of no downtime, $X_j = 0$. Hence, for $C_j = 0$, we obtain

$$\mathbf{P}(C_j = 0) = \mathbf{P}\Big(X_j/_{t_{c,j}} \in [0, u_1]\Big) \tag{4.70}$$

$$= \mathbf{P}\Big(0 \leq X_j/_{t_{c,j}} \leq u_1\Big) \tag{4.71}$$

$$= \mathbf{P}(0 \leq X_j \leq u_1 \cdot t_{c,j}) \tag{4.72}$$

$$= F_{X_j}(u_1 \cdot t_{c,j}) - \lim_{x \nearrow 0} F_{X_j}(x). \tag{4.73}$$

[1]The support of a RV C is the smallest closed set $\mathcal{C} \in \mathbb{R}$ such that $\mathbf{P}(C \in \mathcal{C}) = 1$.

The limit is necessary because we seek the probability of X_j being greater than *or equal to* 0. Since $\lim_{x \nearrow 0} F_{X_j}(x) = 0$, we obtain

$$\mathbf{P}(C_j = 0) = F_{X_j}(u_1 \cdot t_{\mathrm{c},j}). \tag{4.74}$$

We can now describe the distribution of the SLA compensation for arbitrary billing cycles. In the next section, we will derive the distribution for a connection's entire contract period, Δt_{h}.

4.4.2.2 Compensation per Contract Period

Let the contract period, Δt_{h}, consist of m billing cycles. Then, the cumulated compensation for the whole contract period is

$$C = \sum_{j=1}^{m} C_j. \tag{4.75}$$

In order to find the CDF of C, we define the RV $C_{1,i}$ as the sum of the cumulated compensation of the connection's first i billing cycles, i.e.,

$$C_{1,i} = \sum_{j=1}^{i} C_j \qquad \text{for } i > 0. \tag{4.76}$$

Since each summand in (4.76) is supported on $\mathcal{C}_1$, the support of $C_{1,i}$ is

$$\mathcal{C}_i = \left\{ \sum_{k=1}^{i} c_k \,:\, (c_1, c_2, \ldots, c_i) \in \mathcal{C}_1^i \right\} \subset \mathbb{R}_{\geq 0}. \tag{4.77}$$

As an example, consider a hypothetical compensation policy that compensates either 50 % or 100 % of the monthly recurring charge (MRC), depending on how long the connection was down. Then, in each billing cycle, the amount of compensation, C_j, takes values in $\mathcal{C}_1 = \{0, 0.5 \cdot MRC, 1 \cdot MRC\}$. Now, consider the cumulated compensation for the *first three* billing cycles. Ideally, all three billing cycles comply with the SLA, and no compensation arises. In the worst case, full compensation is necessary in each billing cycle. Hence, the possible cumulated compensation ranges from 0 to $3 \cdot MRC$. More precisely, we have $\mathcal{C}_3 = \{0, 0.5 \cdot MRC, 1 \cdot MRC, 1.5 \cdot MRC, 2 \cdot MRC, 2.5 \cdot MRC, 3 \cdot MRC\}$.

Using an approximation, the probability mass function (PMF) of $C_{1,i}$ for $i > 1$ can be calculated iteratively. As a first step, we have

$$\mathbf{P}(C_{1,i} = c) = \mathbf{P}(C_{1,i-1} + C_i = c). \tag{4.78}$$

The sum $C_{1,i-1} + C_i$ equals c only if $C_{1,i-1} = y$, $C_i = z$, and $y + z = c$. Therefore, the right side in (4.78) can be expressed as a sum of joint probabilities, namely

$$\mathbf{P}(C_{1,i-1} + C_i = c) = \sum_{(y,z) \in \mathcal{C}_{i-1} \times \mathcal{C}_1 \,:\, y+z=c} \mathbf{P}(C_{1,i-1} = y, C_i = z). \tag{4.79}$$

By assuming that $C_{1,i-1}$ and C_i are independent — which is the approximation mentioned above — we finally obtain

$$\mathbf{P}(C_{1,i} = c) \approx \sum_{(y,z) \in \mathcal{C}_{i-1} \times \mathcal{C}_1 \,:\, y+z=c} \mathbf{P}(C_{1,i-1} = y) \cdot \mathbf{P}(C_i = z). \tag{4.80}$$

This means that $\mathbf{P}(C_{1,i} = c)$ can be computed iteratively from the PMF of C_j defined in (4.64) and (4.70). Generally, the PMF of the sum of two independent RVs is the convolution of their PMFs. Replacing y with $c - z$ inside the sum of (4.80) yields $\mathbf{P}(C_{1,i-1} = c - z) \cdot \mathbf{P}(C_i = z)$, which shows that this operation is indeed a convolution. A simulative evaluation of the approximation error is provided in Section 5.2.

With this result, the cumulated compensation for the whole contract period is

$$C = C_{1,m} \tag{4.81}$$

with the CDF

$$F_C(c) = \mathbf{P}(C_{1,m} \leq c) \tag{4.82}$$

$$= \sum_{y \in \mathcal{C}_m : y \leq c} \mathbf{P}(C_{1,m} = y). \tag{4.83}$$

At this point, we are finally able to compute the compliance probability over a connection's entire contract period, namely

$$\mathbf{P}(C \leq c_{\max}) = F_C(c_{\max}). \tag{4.84}$$

This model is a core element of CAPSS. The next section discusses two aspects of the derived distribution that are of interest for the implementation details of the actual CAPSS algorithm.

4.4.2.3 Properties of the Compensation

We highlight two properties of the compensation distribution. The first property is related to series systems. The steady-state availability of a series system is the product of the individual availabilities [Gro04, Sec. 3.12.2]. This is a useful property, e.g., for the system reduction shown in Section 4.3 or for finding the path with the highest steady-state availability using Dijkstra's algorithm. Unfortunately, the compensation distribution does not possess this property. To be more precise, consider the network in Figure 4.12 consisting of the three nodes A, B, and C, two links, and three connections.

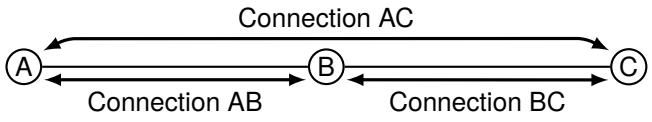


Figure 4.12 Two-link linear network

Connection AC traverses the two links sequentially, and we want to compute its compliance probability, $\mathbf{P}(C_{\mathrm{AC}} \leq c_{\max})$. Assume that we know the compliance probability for connection AB, $\mathbf{P}(C_{\mathrm{AB}} \leq c_{\max})$, and for connection BC, $\mathbf{P}(C_{\mathrm{BC}} \leq c_{\max})$. Connection AC can be considered a serial composition of connections AB and BC. However, we have

$$\mathbf{P}(C_{\mathrm{AC}} \leq c_{\max}) \neq \mathbf{P}(C_{\mathrm{AB}} \leq c_{\max}) \cdot \mathbf{P}(C_{\mathrm{BC}} \leq c_{\max}). \tag{4.85}$$

A simple example proves that taking the product as in (4.85) is not correct. Assume all connections have a contract period of one year, spanning exactly 12 full calendar months. Furthermore, assume that the SLA for all connections guarantees an availability of 0.9999. A worse monthly availability will lead to a compensation of 50 % of the MRC. A monthly

availability that is even below 0.999 will result in a compensation of 100 % of the MRC. Hence, the compensation policy is given by

$$g(u) = \begin{cases} 0 & \text{for } u \leq 0.0001 \\ 0.5 \cdot MRC & \text{for } 0.0001 < u \leq 0.001 \\ 1 \cdot MRC & \text{for } u > 0.001 \end{cases}. \tag{4.86}$$

Assume that we have two identical links, each with a steady-state availability of 0.99995 and a mean time to repair (MTTR) of 1 h. The probability that the total compensation (over the whole contract period of one year) for a connection over *one* of the links does not exceed $c_{\max} = 1 \cdot MRC$ is $\mathbf{P}(C_{\mathrm{AB}} \leq c_{\max}) = \mathbf{P}(C_{\mathrm{BC}} \leq c_{\max}) = 0.954$. According to the naive multiplication approach, the probability for connection AC over both links would then be

$$\mathbf{P}(C_{\mathrm{AB}} \leq c_{\max}) \cdot \mathbf{P}(C_{\mathrm{BC}} \leq c_{\max}) = 0.91. \tag{4.87}$$

However, the correct way of computing this probability is by first reducing the series system both links form to a single-component system, as described in Section 4.3. Then, the probability we are looking for is

$$\mathbf{P}(C_{\mathrm{AC}} \leq c_{\max}) = 0.85 \tag{4.88}$$

$$\neq \mathbf{P}(C_{\mathrm{AB}} \leq c_{\max}) \cdot \mathbf{P}(C_{\mathrm{BC}} \leq c_{\max}). \tag{4.89}$$

More details on the calculation of this specific compliance probability can be found in Appendix A.2.

The second property we want to discuss here is the fact that a higher steady-state availability does not necessarily go hand in hand with a higher compliance probability. To demonstrate this, consider the scenario from above again and focus on connections AB and BC. This time, we assume that the link between B and C, and thus also connection BC, has a steady-state availability of 0.9999 and an MTTR of 10 h. Connection AB still has a steady-state availability of 0.99995 and an MTTR of 1 h.

Table 4.2 Connection parameters

Index i	Availability a_i	$MTTR_i$	Failure rate λ_i	$\mathbf{P}(C_i \leq c_{\max})$
AB	0.99995	1 h	$5 \cdot 10^{-5}\,\mathrm{h}^{-1}$	0.954
BC	0.9999	10 h	$1 \cdot 10^{-5}\,\mathrm{h}^{-1}$	0.997

Since the MTTR and availability values for the two connections are fixed, the failure rates and the compliance probabilities are determined as well. All values are listed in Table 4.2. As can be seen, connection AB has a higher steady-state availability than connection BC. Nevertheless, connection AB's compliance probability is lower than that of connection BC. This shows that there are scenarios in which the connection with the higher steady-state availability still has the lower compliance probability.

The main reason here is the difference in the failure rates. Connection AB has a higher failure rate than connection BC. Hence, connection AB experiences failures more often, even though it has a higher steady-state availability. With an SLA-guaranteed monthly availability of 0.9999, the allowed downtime per month is roughly four minutes. Longer downtimes violate the SLA and result in 50 % compensation. A cumulated downtime of more than 43

minutes also violates the 0.999 threshold in the SLA and results in the compensation of the full MRC. With MTTR values of 1 h or 10 h, respectively, already a single failure is likely to lead to an SLA violation. Therefore, the failure rate is the deciding factor in this example. Since connection AB has the higher failure rate, its compliance probability is lower.

4.5 Provisioning Algorithm

The previous section has laid the mathematical foundations for CAPSS. This section completes the methodology by introducing the algorithmic realization. It consists of one procedure and several small functions. Section 4.5.1 presents the procedure HANDLECONNECTIONREQUEST and the function FINDROUTE, which are responsible for the routing of connection requests. Section 4.5.2 describes the functions COMPLIANCEPROBABILITY, PMFCOMPENSATION, and SUMPMF, which are responsible for the calculation of the compliance probability and follow the model derived in Section 4.4.

The two mechanisms CAPSS and CAP are mostly equal, except for the surplus sharing functionality. Therefore, the algorithms also describe CAP if line 4 in Algorithm 4.1 is replaced by $p_t^* \leftarrow p_t$.

The procedure and functions are represented as pseudocode. A *function* in the pseudocode is similar to a mathematical function, i.e., it returns one (or more) value for the provided arguments. It cannot access or modify any global state. In contrast, a *procedure* does not return anything but can access and modify the global state. For example, Algorithm 4.1 modifies the globally accumulated compliance surplus, Δp_s, and sets up connections. To improve clarity, variable indices that denote the connection index or the billing cycle index (i and j) are often omitted. The pseudocode is Python-inspired in some parts. In particular, we use a *conditional expression*[2] in Algorithm 4.3, line 20, and the value *None*[3] where other programming languages use *null* or *nil*. Furthermore, we use variables of type *dictionary*[4] (or *dict* for short) to represent associative arrays that map keys to values. An example can be found in Algorithm 4.3, line 3. The special function *values*() returns an array of only the values stored in the dictionary. When we iterate over a dictionary, every key-value pair is accessed as a tuple (this corresponds to the *items*() function in Python). For an example, see Algorithm 4.5, line 4.

4.5.1 Connection Routing

The basic framework of the provisioning mechanism is shown in Algorithm 4.1. The procedure **HANDLECONNECTIONREQUEST** handles arriving connection requests. It tries to find a suitable route for the request and accepts it if a route is found. Otherwise, it rejects the connection request. Algorithm 4.1 is based on the flowchart in Figure 4.1. A connection request consists of a source node s, a destination node d, a data rate h, a compensation policy g, an arrival time t_a, and a contract period Δt_h. Since we assume that the lead time is negligible, the arrival time, t_a, and the start of the connection, $t_{s,1}$, coincide. This also means that the connection is deployed instantly if a route is found. Therefore, in order to hand over a working connection to the customer, potential routes may only include links that are working at the time of the request's arrival. Additionally, a suitable route must provide a

[2] https://docs.python.org/3/reference/expressions.html#conditional-expressions
[3] https://docs.python.org/3/library/constants.html#None
[4] https://docs.python.org/3/tutorial/datastructures.html#dictionaries

Input (Global)

- $G = (V, E)$ . . . . . Network graph
- $p_t \in [0, 1)$ . . . . . Compliance target
- $c_{max} \in \mathbb{R}_{\geq 0}$ . . . . . Allowed compensation
- $\Delta p_s \in \mathbb{R}_{\geq 0}$ . . . . . Cumulated compliance surplus
- m_{prot} . . . . . Protection mode (path, segment, or no protection)

Input (Arguments)

$r = (s, d, h, g, t_a, \Delta t_h)$ . . . Connection request with source $s \in V$, destination $d \in V$, data rate $h \in \mathbb{R}_{>0}$, compensation policy $g : [0, 1] \to \mathcal{C}_1$, arrival time $t_a \in \mathbb{R}_{\geq 0}$, and contract period $\Delta t_h \in \mathbb{R}_{>0}$

```
1  procedure HandleConnectionRequest(r)
2      G_w ← subgraph of G including only working links
3      G_c ← subgraph of G_w including only links with free capacity ≥ h
4      p_t^* ← p_t − Δp_s                    ▷ For CAP, replace this line with p_t^* ← p_t
5      route, p ← FindRoute(G_c, r, c_max, p_t^*, m_prot)
6      if route ≠ None then
7          Set up connection on route
8          Δp_s ← p − p_t^*                  ▷ Update global compliance surplus
9      else
10         Reject connection request
```

Algorithm 4.1 Basic provisioning framework of CAPSS

data rate of h. The two requirements are handled in lines 2 and 3 by creating the subgraph G_c that only contains valid links.

A suitable route must also provide a sufficiently large compliance probability. The target level for the compliance probability, p_t, is specified by the network operator. CAPSS makes use of surplus sharing, i.e., the compliance target is relaxed by the accumulated compliance surplus, Δp_s (line 4). We denote the resulting relaxed compliance target by p_t^*. For CAP, on the other hand, surplus sharing is not available, and therefore, the compliance target is not relaxed. Consequently, $p_t^* = p_t$ for CAP.

In line 5, the function FindRoute (Algorithm 4.2) is invoked to find a suitable route in the adapted graph G_c. If a route is found, the connection request is accepted, and the connection is set up. Furthermore, in line 8, the globally accumulated compliance surplus is updated with the surplus of the selected route. If no route is found, the connection request is rejected.

The function **FindRoute** in Algorithm 4.2 finds a route for a connection request. It supports unprotected routes as well as routes with path or segment protection. It always starts by searching for an unprotected route. If its compliance surplus is sufficiently high, this unprotected route is returned. Otherwise, an additional backup path is determined. In the case of segment protection, this is done iteratively. First, a backup path that protects only the last link of the unprotected path is determined, and the resulting route is checked for its compliance probability. If the compliance is still too low, the backup path is extended to cover the last *two* links of the unprotected path. This is repeated until the route's compliance is sufficiently high. However, if even path protection is not enough, FindRoute returns *None*.

FindRoute is supposed to find a route with a compliance probability $\mathbf{P}(C \leq c_{max})$ larger than or equal to p_t^*. However, finding a path with a specific compliance probability or even

Input (Arguments)
- G . . . . . Adapted network graph; links work and provide sufficient capacity
- $r = (s, d, h, g, t_a, \Delta t_h)$ . . . Connection request
- $c_{max} \in \mathbb{R}_{\geq 0}$ . . . . . Allowed compensation
- $p_t^* \in (-\infty, 1)$ . . . . . Relaxed compliance target
- m_{prot} . . . . . Protection mode (path, segment, or no protection)

Output Route including primary path and backup path, and computed compliance probability

```
1  function FindRoute(G, r, c_max, p_t^*, m_prot)
2      primaryPath ← DijkstraShortestPath(G, s, d)
3      if primaryPath = None then
4          return None
5      G_b ← subgraph of G excluding links in primaryPath
6      σ ← d                                   ▷ Node at which backup path starts
7      backupPath ← "empty" path from d to d   ▷ Equivalent to no backup path but ≠ None
8      loop
9          if backupPath ≠ None then
10             λ, μ ← compute route's failure and repair rate from primaryPath and backupPath
11             p ← ComplianceProbability(g, t_a, Δt_h, λ, μ, c_max, p_t^*)
12             if p ≠ None then                 ▷ p ≠ None implies p ≥ p_t^*
13                 return (primaryPath, backupPath), p
14         if m_prot = path protection and σ ≠ s then
15             σ ← s
16         else if m_prot = segment protection and σ ≠ s then
17             σ ← predecessor node of σ in primaryPath
18         else
19             return None
20         backupPath ← DijkstraShortestPath(G_b, σ, d)   ▷ Link-disjoint with primaryPath
```

Algorithm 4.2 Function FindRoute

the path with the maximum compliance probability is not straightforward. In our scenarios, it cannot be found reliably using a greedy algorithm. For example, Xia et al. [Xia+11a] have already shown for the SLA violation probability that their link weight transformation maximizes the violation probability of the whole path only approximately. Also, the first example in Section 4.4.2.3 has shown that the compliance probability of a path cannot be decomposed into a product of the included links' compliance probabilities. Ideally, we would find a route with a compliance as close to p_t^* as possible to reduce overfulfillment already during pathfinding. However, this is even more demanding than maximizing the compliance. Instead, since we have the surplus sharing mechanism to handle overfulfillment, we keep the algorithm simple here. We search for the route with the highest steady-state availability. That route is likely to have a very high compliance probability as well.

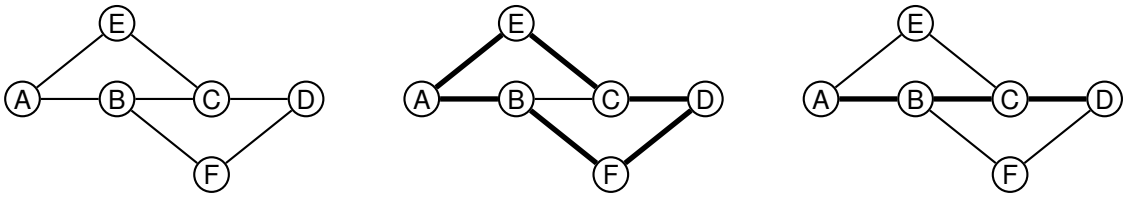
To this end, we employ the method introduced by Zhang et al. [Zha+03b]. They find the *most reliable path*, i.e., the path with the highest steady-state availability, using Dijkstra's shortest path algorithm and specifically crafted link weights. Dijkstra's algorithm finds the path with the lowest total weight. The total weight of a path is the sum of the contained links' weights. The steady-state availability of a path is the product of the steady-state availabilities of the contained links, a_e, i.e., $a = \prod_e a_e$. To transform this product into a summation, Zhang et al. apply the logarithm and obtain $\log(a) = \sum_e \log(a_e)$. A path that maximizes $\log(a)$

also maximizes a because the logarithm is a monotonic function, and a is positive ($a = 0$ is theoretically possible but irrelevant in practice). Since Dijkstra's algorithm *minimizes* the path weight but the goal is to find the *maximum* availability, Zhang et al. assign the *negative* logarithm, i.e., $-\log(a_e)$, as link weights. Eventually, they obtain the path with the maximum steady-state availability.

As mentioned above, this is only an approximate approach. The second example in Section 4.4.2.3 has shown that a path with the maximum steady-state availability does not necessarily have the maximum compliance probability too. Nevertheless, for most practical scenarios, this heuristic is sufficient to identify paths with the highest or at least a very high compliance probability. Since we assume static links, we compute and assign the link weights only once at the beginning of a simulation. Therefore, this assignment step is not part of the algorithms we present here.

Dijkstra's algorithm is first called in line 2 to find a primary path. If no path is found, then, apparently, the network does not provide enough remaining capacity, and FINDROUTE returns without a route. Otherwise, the function enters a loop, in which it first checks the compliance probability of the found path (line 12) using the COMPLIANCEPROBABILITY function. In the first loop iteration, the backup path is empty (line 7). That means the first iteration checks an unprotected route. If the compliance probability is sufficient, FINDROUTE returns this unprotected route. Otherwise, a protection path is added for the next iteration in line 20. Protection paths are determined using Dijkstra's algorithm in a graph G_b that does not include the links contained in the primary path. Consequently, the backup path is always link-disjoint from the primary path. The backup path starts at the node σ. For path protection, σ equals the start node, s. For segment protection, σ moves from the destination node, d, to the start node, s, along the nodes of the primary path. In that way, the backup path is iteratively increased until the compliance probability is high enough.

In sparse topologies or when the network is heavily loaded, the graph G might be a "trap topology" [DGM94]. In a trap topology, the presence of a link-disjoint backup path strongly depends on the choice of the primary. Figure 4.13a presents an example of such a topology from [DGM94].



(a) Topology **(b)** Primary and backup path **(c)** Backup path blocked

Figure 4.13 Example trap topology from [DGM94]

Figure 4.13b shows that it is possible to find link-disjoint primary and backup paths to connect A to D. However, if the path A–B–C–D is chosen as the primary path, no link-disjoint backup path exists (Figure 4.13c). Since CAPSS selects primary and backup paths sequentially, it is possible that such "traps" arise. In that case, FINDROUTE starts over with the next shortest primary path. If no route is found after an adjustable number of restarts, FINDROUTE returns without a route. The restart mechanism adds another loop to the FINDROUTE function. For reasons of clarity, this loop has been omitted in Algorithm 4.2.

4.5.2 Calculation of the Compliance Probability

The function **CompliancePROBABILITY** in Algorithm 4.3 computes the compliance probability, $\mathbf{P}(C \leq c_{\max})$, of a specific route and checks whether it fulfills the targeted compliance level of p_t^*. If the probability is sufficient, i.e., $\mathbf{P}(C \leq c_{\max}) \geq p_t^*$, CompliancePROBABILITY returns this probability. If the probability is too low, the exact value is not needed in the overall methodology. Therefore, CompliancePROBABILITY simply returns *None* in case the probability is too low. This allows shortcuts in the algorithm and, hence, reduces the running time.

Input (Arguments)

- $g : [0,1] \to \mathcal{C}_1$ . . . Compensation policy function
- $t_a \in \mathbb{R}_{\geq 0}$ . . . . . Arrival time of the connection request
- $\Delta t_h \in \mathbb{R}_{>0}$ . . . . Contract period of the connection
- $\lambda \in \mathbb{R}_{>0}$ . . . . . Failure rate of the route
- $\mu \in \mathbb{R}_{>0}$ . . . . . Repair rate of the route
- $c_{\max} \in \mathbb{R}_{\geq 0}$ . . . . Allowed compensation
- $p_t^* \in (-\infty, 1)$ . . . . Relaxed compliance target

Output $\mathbf{P}(C \leq c_{\max})$ if this probability is larger than or equal to p_t^*; None otherwise

```
1  function CompliancePROBABILITY(g, t_a, Δt_h, λ, μ, c_max, p_t^*)
2      t_{c,1} ← compute duration of first billing cycle from t_a and Δt_h
3      f : dict ℝ_{≥0} → [0, 1]
4      f ← PMFCOMPENSATION(g, λ, μ, t_{c,1}, 0, c_max)                ▷ PMF of C_1
5      if ∑ values(f) < p_t^* then                                    ▷ Is P(C_1 ≤ c_max) < p_t^*?
6          return None
7      Δt_s ← t_{c,1}
8      n ∈ ℕ_0 ← compute number of month-long billing cycles from t_a and Δt_h
9      for i = 1, ..., n do
10         f_{i+1} ← PMFCOMPENSATION(g, λ, μ, 1 month, Δt_s, c_max)   ▷ PMF of C_{i+1}, 1 ≤ i ≤ n
11         f ← SUMPMF(f, f_{i+1}, c_max)                              ▷ PMF of C_{1,i+1}
12         if ∑ values(f) < p_t^* then                                ▷ Is P(C_{1,i+1} ≤ c_max) < p_t^*?
13             return None
14         Δt_s ← Δt_s + 1 month
15     t_{c,n+2} ← compute duration of last billing cycle from t_a and Δt_h
16     if t_{c,n+2} > 0 then
17         f_{n+2} ← PMFCOMPENSATION(g, λ, μ, t_{c,n+2}, Δt_s, c_max) ▷ PMF of C_{n+2}
18         f ← SUMPMF(f, f_{n+2}, c_max)                              ▷ PMF of C_{1,n+2}
19     p ← ∑ values(f)                                                ▷ P(C ≤ c_max)
20     return p if p ≥ p_t^* else None
```

Algorithm 4.3 Function CompliancePROBABILITY

The function is based on the mathematical model introduced in Section 4.4.2, i.e., it computes the compliance probability for the whole contract period iteratively from the underlying billing cycles' compensation PMFs. Since we want to compute the probability of $C \leq c_{\max}$, all functions invoked in CompliancePROBABILITY, namely PMFCOMPENSATION and SUMPMF, only compute PMFs in the domain $[0, c_{\max}]$. Probabilities for compensation values above $c_{\max}$ are not required and, therefore, also not computed. This becomes apparent in Algorithm 4.4, line 4, and in Algorithm 4.5, lines 3 and 6.

Input (Arguments)
- $g : [0,1] \to \mathcal{C}_1$ . . . Compensation policy function
- $\lambda \in \mathbb{R}_{>0}$ . . . . . Failure rate of the route
- $\mu \in \mathbb{R}_{>0}$ . . . . . Repair rate of the route
- $t_c \in \mathbb{R}_{>0}$ . . . . . Duration of the j-th billing cycle
- $\Delta t_s \in \mathbb{R}_{\geq 0}$ . . . . . Duration between connection start and start of j-th billing cycle
- $c_{max} \in \mathbb{R}_{\geq 0}$ . . . . Allowed compensation

Output PMF of compensations up to c_{max} for the j-th billing cycle

```
1 function PmfCompensation(g, λ, μ, t_c, Δt_s, c_max)
2     f ← empty dictionary
3     p_last ← 0
4     for each c ∈ C_1 : c ≤ c_max and in ascending order do
5         u ← sup{g^{-1}(c)}
6         p ← F_X(u · t_c; t_c, λ, μ, Δt_s)
7         f[c] ← p − p_last
8         p_last ← p
9     return f
```

Algorithm 4.4 Function PmfCompensation

The function ComplianceProbability starts by computing the duration of the first billing cycle, $t_{c,1}$, from the connection's arrival time, t_a, and its contract period, Δt_h. We denote the PMF of the total compensation by f. The PMF is realized as a dictionary that maps compensation values to probabilities, i.e., f : dict $\mathbb{R}_{\geq 0} \to [0,1]$.

In line 4, the PMF for the first billing cycle is computed using the function PmfCompensation shown in Algorithm 4.4. **PmfCompensation** realizes the mathematical description in (4.69) and (4.74). In lines 8 to 14 of Algorithm 4.3, the PMF is iteratively augmented for every month-long billing cycle that follows the first billing cycle. The function **SumPmf** is shown in Algorithm 4.5. It is responsible for computing the PMF of the sum of two RVs according to the mathematical description in (4.80).

In lines 15 to 18 of Algorithm 4.3, the PMF f is augmented by the PMF for the last

Input (Arguments)
- $f_{1,i-1}$: dict $\mathbb{R}_{\geq 0} \to [0,1]$ . . . PMF of the compensation RV $C_{1,i-1}$
- f_i : dict $\mathbb{R}_{\geq 0} \to [0,1]$ . . . . PMF of the compensation RV C_i
- $c_{max} \in \mathbb{R}_{\geq 0}$ . . . . . Allowed compensation

Output PMF of $C_{1,i-1} + C_i$ up to a compensation of c_{max}

```
1 function SumPmf(f_{1,i-1}, f_i, c_max)
2     f ← empty dictionary with a default value of 0 for unknown keys
3     for each (y, p_{1,i-1}) ∈ f_{1,i-1} : y ≤ c_max do
4         for each (z, p_i) ∈ f_i do
5             c ← y + z
6             if c ≤ c_max then
7                 f[c] ← f[c] + p_{1,i-1} · p_i
8     return f
```

Algorithm 4.5 Function SumPmf

billing cycle, which has a duration of $t_{c,n+2}$. Finally, in line 19, the compliance probability, $\mathbf{P}(C \leq c_{\max})$, is computed. Since f only contains compensations up to $c_{\max}$, summing over all (probability) values of f yields the desired probability $\mathbf{P}(C \leq c_{\max})$.

To improve the running time, CompliancePROBABILITY can return early in lines 6 and 13 if it becomes apparent that the required compliance target p_t^* is missed. The rationale is as follows. With every additional billing cycle, the probability that C will be less than or equal to $c_{\max}$ decreases because every additional billing cycle brings with it additional potential compensation. Therefore, if the cumulated compliance probability of the first i billing cycles, $\mathbf{P}(C_{1,i} \leq c_{\max})$, is already less than p_t^*, it is safe to say that the probability of the final compensation, $\mathbf{P}(C \leq c_{\max})$, will be less than p_t^* as well. Consequently, CompliancePROBABILITY returns *None* to indicate that the route does not satisfy the compliance target p_t^*. The example in Appendix A.2 illustrates this. In particular, the last column of Table A.2 shows that the compliance probability decreases from billing cycle to billing cycle.

4.6 Discussion

This chapter presented the compensation-aware provisioning mechanism CAPSS. It mainly consists of a mathematical model for the probability distribution of SLA compensation and the actual provisioning algorithm that also features surplus sharing.

Categorization CAPSS provisions connections based on the compliance probability, $\mathbf{P}(C \leq c_{\max})$. As mentioned before, if the network operator sets $c_{\max} = 0$, CAPSS is essentially reduced to a mechanism that is only aware of the probability of availability violation, $\mathbf{P}(A < \alpha_{\text{SLA}})$. In Section 3.4.3, we discussed several mechanisms proposed in the literature that provision connections based on the violation probability. With $c_{\max} = 0$, CAPSS is directly comparable to them. Allowing no compensation is the strictest possible setting for $c_{\max}$. However, since CAPSS *does allow* non-zero compensation, it can be considered a less stringent version of the mechanisms that are based on the violation probability.

Apart from that, the model for the compensation distribution can be categorized like the models in Table 3.2. The respective column names are emphasized in the following. The *modeled distribution* is the compensation over a connection's contract period. The *target statistic* is the compliance probability for an allowed compensation of $c_{\max}$. The compensation policy uses a billing cycle as *basis*, and the policy *function* is a staircase function of the cumulated downtime or unavailability in the billing cycle. The *up- and downtimes* of the network elements are assumed to follow exponential distributions. Our model for the compensation distribution is a *composite* model because a connection or its route, respectively, consists of multiple network elements. Supported modes of *protection* are dedicated path and segment protection.

The whole CAPSS mechanism can be categorized like the mechanisms in Table 3.3. The *overall goal* is the resource-efficient fulfillment of the compliance probability targeted by the network operator. Our *solution approach* is the sharing of compliance surplus among connections to relax the requirements on their routes. Hence, the underlying *mechanism* is a parameter adaptation, namely the adaptation of the compliance target. The key *measure* used in CAPSS is the compliance probability. CAPSS does not need *online* information about other connections. Also, it does not *reprovision* connections. Both properties make CAPSS much less complex in its operation than other mechanisms that require online information or reprovision connections. As we will see in the evaluation in Chapter 5, the

surplus sharing mechanism included in CAPSS increases the network capacity considerably compared with CAP. This is achieved mainly by relaxing protection requirements. However, CAPSS also improves the capacity by increasing the path diversity. So, regarding *protection*, CAPSS provides advantages for dedicated path- or segment-protected connections but also for unprotected connections. It is questionable whether CAPSS also provides benefits in combination with shared protection since shared protection *already is a sharing mechanism*, just like surplus sharing. This question is left for future research.

Applicability CAPSS is designed for dynamic connection provisioning in transport networks. It can be employed together with various connection-oriented technologies, e.g., optical connections in the wavelength-division multiplexing (WDM) layer, optical transport network (OTN) channels, or paths in multiprotocol label switching (MPLS) and multiprotocol label switching – transport profile (MPLS-TP). Our implementation supports dedicated path protection and dedicated segment protection. Both protection modes are found in most modern technologies, as has been discussed in Section 2.4.2.

Apart from that, our method, in particular the model of the compensation distribution, can also be adapted to suit other use cases. For example, in the context of cloud computing [GH12c; MMN19; Nal17] and 5G radio access network (RAN) slicing [Ela+19], service outages, redundancy, and SLA compensation are an important topic as well. Generally, our mathematical model can be applied to a wide range of availability-related use cases, even outside the field of networks and computing.

In the following chapter, we evaluate CAPSS in transport networks with a tendency toward optical layer scenarios.

5

Evaluation

This chapter is dedicated to the evaluation of CAPSS. We focus on two main topics. First, we evaluate the accuracy of CAPSS and the underlying mathematical model. Second, we study the performance with respect to the network capacity and the request rejection ratio. To this end, we compare CAPSS with CAP and a conventional, availability-based provisioning mechanism. The chapter is structured as follows. Section 5.1 introduces the methodology we employ for the various evaluation studies. Section 5.2 evaluates the accuracy of the model and CAPSS. Section 5.3 provides a small-scale example illustrating the behavior of CAPSS and surplus sharing in particular. This helps to better understand the subsequent sections. Sections 5.4 and 5.5 evaluate the performance of CAPSS. To this end, Section 5.4 compares CAPSS with a conventional mechanism, while Section 5.5 compares it with CAP. Finally, Section 5.6 provides a summary and recommendations.

5.1 Methodology

The evaluation in this chapter is based on simulation studies. The studies share the basic evaluation methodology we introduce next. Section 5.1.1 introduces the two provisioning mechanisms that are used as reference approaches in the simulations. Section 5.1.2 explains the simulation procedure. Section 5.1.3 introduces the most important statistics we evaluate. Lastly, Section 5.1.4 describes the employed network scenarios comprising synthetic and realistic networks.

5.1.1 Reference Mechanisms

In this chapter, we evaluate different aspects of CAPSS. The evaluation of the model accuracy and the overall accuracy of CAPSS in Section 5.2 does not require any reference mechanisms. However, we also evaluate the performance of CAPSS with respect to the network capacity and the request rejection ratio. To this end, we employ two reference mechanisms. The

first reference is a conventional, availability-based provisioning mechanism. The second reference is CAP. CAP serves to study the benefits surplus sharing provides.

5.1.1.1 Conventional Provisioning (CONV)

The first reference mechanism is a conventional mechanism, as discussed in Section 3.2. It provisions connections such that the connection's steady-state availability fulfills the service level agreement (SLA) availability, α_{SLA}. The value of α_{SLA} is extracted from the compensation policy. Apart from that, the mechanism does not incorporate any further information about the compensation policy. Also, it does not employ any sharing functionality.

Algorithms 5.1 and 5.2 describe the mechanism in detail. In the following, we call this specific implementation *CONV*. Algorithm 5.1 is similar to its respective counterpart in CAPSS and CAP, HANDLECONNECTIONREQUEST. However, no surplus sharing is involved, and the compliance target has been replaced with the SLA availability, α_{SLA}. Algorithm 5.2 is similar to the CAPSS version as well. However, the algorithm only supports unprotected and path-protected connections since we do not evaluate segment protection in the studies involving CONV. Like in CAPSS's FINDROUTE function (Algorithm 4.2), the outer loop to handle trap topologies is omitted in Algorithm 5.2 for clarity.

Input (Global)
$G = (V, E)$ . . . . Network graph
Input (Arguments)
$r = (s, d, h, g)$ . . . Connection request with source $s \in V$, destination $d \in V$, data rate $h \in \mathbb{R}_{>0}$, and compensation policy $g : [0, 1] \to \mathcal{C}_1$

```
1  procedure HandleConnectionRequestConv(r)
2      G_w ← subgraph of G including only working links
3      G_c ← subgraph of G_w including only links with free capacity ≥ h
4      u_1 ← sup{g^{-1}(0)}                    ▷ Max. unavailability without compensation
5      α_SLA ← 1 − u_1                         ▷ Min. availability without compensation ≡ SLA availability
6      route ← FindRouteConv(G_c, r, α_SLA)
7      if route ≠ None then
8          Set up connection on route
9      else
10         Reject connection request
```

Algorithm 5.1 Basic provisioning framework of CONV

The input to the procedure HANDLECONNECTIONREQUESTCONV in Algorithm 5.1 is a subset of the input to CAPSS's HANDLECONNECTIONREQUEST in Algorithm 4.1. That means all three mechanisms, CAPSS, CAP, and CONV, can be exchanged without any further modifications to the surrounding evaluation framework.

5.1.1.2 Compensation-Aware Provisioning (CAP)

The second reference mechanism is CAP, as described in Section 4.1.1. CAP is a compensation-aware mechanism *without surplus sharing*. Due to its similarity to CAPSS, both mechanisms share major parts of their algorithmic realization. Therefore, we refer to Section 4.5 for the details of CAP's implementation. By comparing the performance of CAPSS with that

Input (Arguments)
- G . . . . . Adapted network graph; links work and provide sufficient capacity
- $r = (s, d, h, g)$ . . . Connection request
- $\alpha_{\text{SLA}} \in [0, 1]$ . . . SLA availability

Output Route including primary path and backup path

```
1  function FindRouteConv(G, r, α_SLA)
2      primaryPath ← DijkstraShortestPath(G, s, d)
3      if primaryPath ≠ None then
4          a ← compute steady-state availability from primaryPath
5          if a ≥ α_SLA then
6              return primaryPath
7          G_b ← subgraph of G excluding links in primaryPath
8          backupPath ← DijkstraShortestPath(G_b, s, d)        ▷ Link-disjoint with primaryPath
9          if backupPath ≠ None then
10             a ← compute steady-state availability from primaryPath and backupPath
11             if a ≥ α_SLA then
12                 return (primaryPath, backupPath)
13     return None
```

Algorithm 5.2 Function FindRouteConv

of CAP, we gain insights into the effects and the advantages and disadvantages of surplus sharing.

5.1.2 Simulation

The following studies use simulation to evaluate CAPSS. While each study has its own scenarios and parameters, the basic simulation methodology is the same and will be explained next. In case a study uses a modified approach, the details are explained in the description of the study.

5.1.2.1 Simulator Architecture and Implementation

Our simulator models a network topology, connection requests, and connections. The topology comprises nodes and links. While links can fail, we assume that nodes cannot. Furthermore, links typically have a limited capacity.

The simulator consists of four main modules. The *network module* holds the network topology, including the link capacities and the current states the links are in. Furthermore, it holds the provisioned connections. The *traffic generation module* creates connection requests with pre-configurable parameters. The *provisioning module* implements the CAPSS algorithm as well as the reference algorithms. It is responsible for provisioning or rejecting connection requests. This also includes the calculation of the compliance probability of potential routes. The operator-defined parameters $c_{\max}$ and p_{t} are global constants throughout a simulation. Lastly, the *bookkeeping module* records connection downtimes when failures occur and the resulting compensation at the end of each billing cycle or when a connection leaves the network. The simulator also records a variety of other statistics, such as the resource utilization, the backup overhead, or the request rejection ratio.

The simulator employs discrete-event simulation [CL21, Ch. 10], i.e., each state change is marked by an event that occurs at a specific point in time. Our simulation model knows five types of events, namely, a *link failure*, a *finished link repair*, the *arrival of a connection request*, the *departure of a connection* at the end of its contract period, and the *end of a billing cycle*.

The *end of a billing cycle* event occurs regularly at the end of each calendar month. It triggers the calculation of the compensations for the ending billing cycle in the bookkeeping module. Events for arriving connection requests are generated by the traffic generation module. The time between two consecutive connection request events, i.e., the inter-arrival time (IAT), follows an exponential distribution in most studies. That means the arrival process is a Poisson process. For each connection request, the traffic generation module uniformly selects a data rate, a contract period, and a compensation policy from a pre-configured set of options. The node pair is also selected uniformly from all node pairs in the network[1]. Once a connection request has been accepted by the provisioning module, an event for the connection departure is scheduled to occur at the end of the connection's contract period. Link failure and repair events are generated by the network module. Each link alternates between the up and down state independently of the other links' states. More details about the connection request generation and the link failures and repairs are explained in the following sections.

Section 4.5.1 explains that in trap topologies, a badly chosen primary path can make it impossible to find an additional backup path. This may lead to a request rejection even though enough capacity is available in principle. To mitigate this problem, CAPSS tries to provision the connection request again using a different primary path. In the following simulations, CAPSS tries up to five different primary paths if a potential trap situation is present. Only after five unsuccessful attempts, the connection request is rejected.

The simulator software is written in Kotlin and runs on the Java virtual machine (JVM). The event and time management, as well as the recording of statistics, is handled by the IKR SimLib [SS10], a discrete-event simulation library for the JVM. All graph algorithms are based on the graph library JGraphT [Mic+20]. The calculation of the compliance probability relies on the Apache Commons Mathematics Library[2] for the numerical integration and CERN's Colt library[3] for the Bessel function. To simplify time-related calculations and comparisons, we assume a month always has 30 days. Consequently, a year has 360 days.

5.1.2.2 Connection Request Generation and Infeasible Requests

The provisioning algorithms in this thesis accept a connection only if a suitable route is found. They can deploy protection if required. However, the maximum level of protection we allow in our studies is path protection with one backup path. In some situations, this might not be enough, which is why some connections may be *infeasible*.

To be more precise, an *infeasible connection request* is a request between nodes s and d that cannot be routed, even with unlimited link capacities, because the availability or compliance requirement, α_{SLA} or p_{t}, is too strict, even for path protection. That means no suitable route between s and d exists.

Using surplus sharing, a connection's requirement could be relaxed. In that way, a connection that is infeasible under CAP could be made feasible with CAPSS. However, that

[1] In some studies, connection requests are only generated for specific node pairs, not for all.
[2] https://commons.apache.org/proper/commons-math
[3] https://dst.lbl.gov/ACSSoftware/colt/index.html

would mean that all connections between s and d would always have relaxed requirements. This, in turn, would mean that those connections would systematically be worse off than other connections. While this is, of course, a possible mode of operation for an aggressive network operator, we opt against such an approach. Instead, we filter out infeasible connection requests during the request generation.

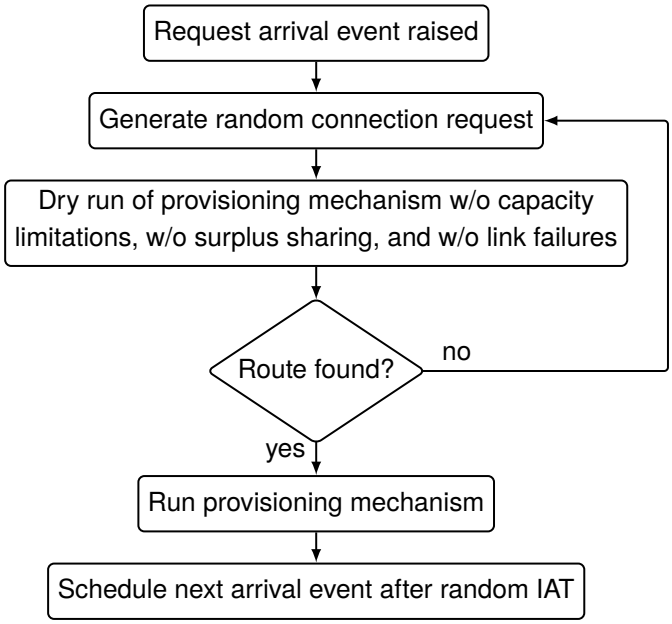


Figure 5.1 Connection request generation loop in the simulation – Infeasible requests are not passed on for provisioning but result in the generation of a new request.

Figure 5.1 shows the basic approach for the generation of connection requests in the simulation. We filter by doing a dry run of the provisioning mechanism without considering link capacity constraints and current link failures and without surplus sharing to see if a route can be found in principle. If this is not the case, we directly generate a new connection request with different source and destination nodes.

5.1.2.3 Link Failures and Repairs

Section 4.4 derives the mathematical model for the compliance probability. The model employs existing results from the literature. These existing results enforce several assumptions. One is that series network elements do not suffer from concurrent failures. Another is that parallel elements, i.e., primary and backup paths of a protected connection, have only one repair crew. In the simulation, we do not adhere to these assumptions because they are not realistic in the network context. However, we do not expect a significant degradation of the model performance when using slightly modified assumptions. Section 5.2.1 evaluates how far this expectation is justified. What we actually assume, though, is the independence of individual link failures. That means the simulator models all link uptimes as independent of each other. This is a very common assumption in the literature [ML13; ZS15; RGC16], even though, in reality, failures are sometimes correlated [Gon+10]. Likewise, we assume independent link downtimes, i.e., the number of repair crews equals the number of links in the network. Limiting the number of repair crews for protected connections to one is not possible in reality because the repair crew does not repair the connection itself but the underlying link. The link, however, is shared by different connections. That means even if

there was a direct mapping between a repair crew and a connection, connections could still be fixed by another crew working on the same link.

The up- and downtimes in the simulated scenarios follow exponential distributions. We set the same mean time to repair (MTTR) for all links in a network. Following [Ver+05], we model the mean time to failure (MTTF) of a link $e \in E$ as a function of its length ℓ_e. An important factor in this function is the *cable cuts* (*CC*) parameter. CC describes the average link length that experiences one cable cut per year. With this, the MTTF of link e is

$$MTTF_e = \frac{CC \cdot 360\,\text{days} \cdot 24\,\text{h/day}}{\ell_e}. \tag{5.1}$$

Remember that we define a year to have 360 days in our simulations. We set the same CC value for all links in a network. Unless otherwise stated, we use the parametrization $CC = 628\,\text{km}$ and $MTTR = 9\,\text{h}$ from [Ver+05]. These values are representative of buried fiber links.

5.1.3 Important Statistics and Quantities

As mentioned before, a plethora of statistics are recorded during a simulation. The most important ones are introduced in the following. Other statistics will be added later on in the respective sections. Additionally, Section 5.1.3.5 discusses how quantities are compared in the following studies. Lastly, Section 5.1.3.6 explains the calculation and visualization of confidence intervals (CIs) and box plots.

5.1.3.1 Rejection Ratio and Infeasible Request Ratio

During a simulation run, many connection requests are generated. Each request can either be accepted or rejected by the provisioning mechanism, or, as explained in Section 5.1.2.2, it can be considered infeasible and ignored. As explained in Sections 4.1 and 5.1.1, the mechanisms we consider accept a connection request if a suitable route is found. A suitable route has sufficient capacity and a sufficient figure of merit, like the compliance probability in the case of CAPSS and CAP or the steady-state availability in the case of CONV. In contrast, a rejection may have two, possibly coincident, reasons. First, the provisioning algorithm might just have stopped searching too early. In our scenarios, this is unlikely but nevertheless possible. Second, there might not be enough network resources available on an otherwise suitable route.

To quantify the different outcomes, let j denote the number of infeasible requests, and let m denote the number of feasible connection requests generated during a simulation run. Of the m feasible requests, n will be accepted and provisioned, while $m - n$ will be rejected. With this, we define the *ratio of infeasible connection requests* as $j/(m + j)$ and the *request rejection ratio* as $(m - n)/m$.

Since we select the node pair for a connection request uniformly, the ratio of infeasible connection requests can often be interpreted as the *share of infeasible node pairs*, i.e., the share of node pairs for which no connection can be offered. However, this is only possible if only one SLA configuration is present, i.e., if parameters like the compensation policy and the contract period are equal for all simulated connection requests. In mixed scenarios, e.g., with various contract periods and both GOLD and SILVER requests, the interpretation is not valid. In any case, though, a positive ratio of infeasible requests implies at least one infeasible node pair for at least one SLA configuration.

5.1.3.2 Carried Load

One of the key performance indicators of a provisioning mechanism is the number of connections or the amount of traffic the network can carry under the mechanism's provisioning strategy.

In many of the following scenarios, all connection requests have the same data rate requirement. For the sake of simplicity, we then assume that each connection request asks for one abstract unit of traffic, and one such unit equals one connection. That means the number of *carried connections* equals the number of carried traffic units. In other scenarios, connection requests ask for different data rates, and hence, the *carried traffic* differs from the number of carried connections.

We use the term *carried load* as a generalization of the two terms *carried connections* and *carried traffic*. In particular, if the connections' data rates are equal, we use the terms carried load and carried connections interchangeably. Otherwise, carried load is meant as a synonym for carried traffic.

5.1.3.3 Network Capacity

When comparing different provisioning mechanisms, the change in network capacity is an important performance indicator. In Section 2.1, we defined the network capacity as the amount of traffic the network can carry without being overloaded. For the remainder of this thesis, we assume that a network is overloaded if the request rejection ratio exceeds a certain threshold value.

A network operator has to monitor the rejection ratio carefully so that the network is neither under- nor overloaded. If the rejection ratio is too high, the operator has to upgrade link and node capacities. If it is too low, the operator could potentially accept more traffic. Therefore, we assume that an operator tries to reach and maintain a certain rejection ratio. This is the desired operating point of the network. According to various authors, e.g., [Zho+16; Mus+19], a rejection ratio of 0.01, i.e., 1 %, is an acceptable value for large-scale transport networks. Therefore, whenever the network capacity is of interest in the following studies, we consider the network for a request rejection ratio of 0.01. Section 5.1.4.5 explains how the desired rejection ratio is calibrated in the simulations.

5.1.3.4 Average Computed Compliance and Compliance Ratio

For each accepted connection i ($i = 1, 2, \dots, n$), CAPSS computes the compliance probability, p_i, of the assigned route during the provisioning process. We define the *average computed compliance* over all n accepted connections as

$$\bar{p}_n = \frac{1}{n} \sum_{i=1}^{n} p_i. \tag{5.2}$$

On the other hand, we estimate the real compliance probability by evaluating each connection's total amount of compensation, c_i, when it leaves the network. The resulting *compliance ratio* based on all n accepted connections is

$$\hat{p}_n = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{c_i \leq c_{\max}} \tag{5.3}$$

where $\mathbf{1}_{c_i \leq c_{\max}}$ is the indicator function, which equals 1 if $c_i \leq c_{\max}$ and 0 otherwise. For large n, $\hat{p}_n$ converges to the *real* compliance probability, i.e.,

$$\lim_{n \to \infty} \hat{p}_n = \mathbf{P}(C \leq c_{\max}). \tag{5.4}$$

If the mathematical model for the compliance probability is accurate, $\overline{p}_n$ converges to the real compliance probability as well. This accuracy is evaluated in Section 5.2.1.

5.1.3.5 Absolute and Relative Differences

Whenever statistics are compared between different parametrizations or provisioning mechanisms, we consider absolute or relative differences or both. Assume that r is a statistic measured in a simulation using a reference mechanism and z is the same statistic under CAPSS. Then, the *absolute difference* or *absolute change* is $z - r$. The *relative difference* or *relative change* is $(z - r)/r = z/r - 1$.

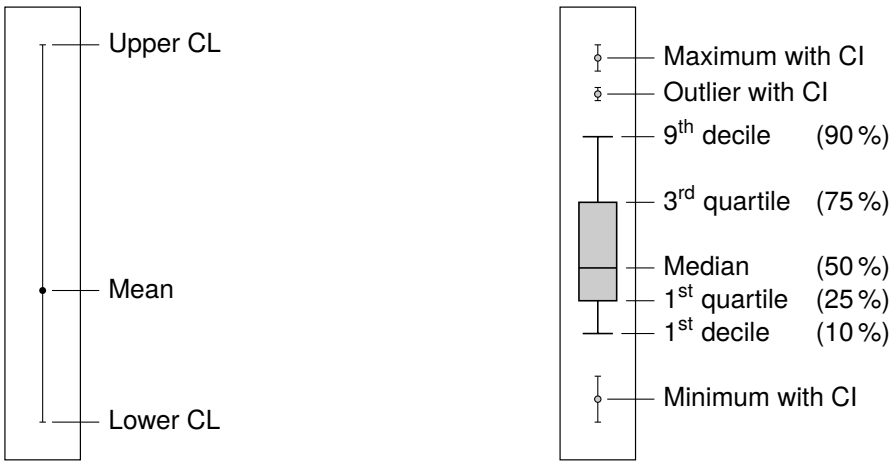
As an example, to determine the increase in network capacity when using CAPSS instead of CAP, we run one simulation for CAPSS and one for CAP and compute the relative difference of the simulated means of the carried traffic.

We always express a relative change as a percentage, i.e., as $100\,\% \cdot (z/r - 1)$. A relative change of 0 % means the reference and CAPSS perform equally.

5.1.3.6 Confidence Intervals and Box Plots of Statistics

All simulations use either the *batch means* method or the method of *independent replications* [CL21] to provide a CI for the mean of a statistic. The warm-up phase of a simulation is not considered for the recording of a statistic. In that way, the network is only evaluated in the steady state and not in the transient state.

All simulation results are plotted with 95 % CIs. However, in some cases, the interval markers are not visible because the intervals are small. Figure 5.2a shows an illustration of a CI consisting of a lower and an upper confidence limit (CL) around the mean.



(a) Mean value with asymmetric CI delimited by upper and lower CL

(b) Box plot with outliers

Figure 5.2 Illustration of CI and box plot

CIs can be symmetric or asymmetric, depending on what type of statistic they describe. For mean values recorded during the simulation, symmetric CIs are computed using Student's *t*-distribution [CL21, Sec. 10.7.2]. However, to compare different provisioning mechanisms, we need to evaluate differences and ratios of statistics. To provide accurate CIs for such *derived* values, we employ the MOVER-D and MOVER-R methods from [Wan+23]. MOVER-D is designed for differences, while MOVER-R handles ratios. CIs computed by MOVER-R are asymmetric. More details about both methods can be found in Appendix A.3.

We use box plots to visualize the results of different simulation runs in a single plot, e.g., for different topologies or different scenario parametrizations. From each simulation run, we obtain the mean value and CI for a certain statistic, e.g., the carried traffic. Those mean values constitute a box plot. An example box plot is shown in Figure 5.2b. The box covers the data between the first and the third quartile. Furthermore, it shows the median of the data, i.e., of all included mean values. The whiskers show the first and the ninth decile, i.e., 10 % and 90 %, respectively. All data points not covered so far are shown as outliers together with their CI. If the number of data points per box plot inside one figure varies, the widths of the box plots in x-direction are scaled linearly with the number of included data points.

5.1.4 Network Scenarios

We evaluate CAPSS in various network scenarios. Those scenarios are characterized by different parameters like contractual parameters and traffic properties, parameters of the routing mechanisms, and network topologies. While most of the following studies employ synthetic topologies in which we can fix certain properties, we also consider realistic topologies to confirm our findings in real-world scenarios. Also, the properties of the synthetically generated topologies are derived from the realistic topologies to some extent.

Even though CAPSS has not been designed for a specific technology, some of the scenario parameters are inspired by technologies and infrastructure parameters of the fiber-optic layer. For example, the network topologies we use in the following correspond to fiber topologies. Also, the link MTTF and MTTR values are typical values in the fiber layer [Ver+05]. A link capacity of 80 connections may be interpreted as 80 wavelengths in a 50 GHz fixed-grid dense wavelength-division multiplexing (DWDM) system [G.694.1][Sim08, Sec. 1.6]. A link capacity of 10 Tbit/s roughly corresponds to 96 channels of 100 Gbit/s each in a DWDM system in the extended C-band [Muk+20, Sec. 21.2.2]. Despite this bias toward optical layer parameters, CAPSS is still applicable to other network layers and technologies as well.

5.1.4.1 Realistic Network Topologies

We consider nine realistic transport network topologies that cover the US, Europe, Germany, and Italy. The topologies differ in various properties, most notably in their spatial sizes and the number of nodes and links. Table 5.1 shows a selection of properties for the networks together with a mapping between their original names and the identifiers we use.

The Italian network *IT/IBN* is from [EPP12] and the US network *US3/CORONET* is from [Von+15]. All other networks are taken directly from the SNDlib [Orl+10] or are slight modifications of networks thereof. The latter applies to the networks *US1* and *EU1*. US1 is based on *abilene*, but the node ATLAM5 and its incident link have been removed to obtain a 2-edge-connected topology that allows for protection between all node pairs. Likewise, EU1 is based on *geant*, a European research network. The *geant* topology in [Orl+10] also has links to the US with a node in New York. However, this node and its incident links have been

Table 5.1 Properties of realistic network topologies – Minimum and maximum values are highlighted. Topologies marked with * are an adaption of the original reference.

ID	Original name	Reference	Nodes	Diameter in hops	Avg. node degree	Avg. link length in km	Avg. node pair distance in km
DE1	nobel-germany	[Orl+10]	17	6	3.06	143	**311**
DE2	germany50	[Orl+10]	50	9	**3.52**	**101**	321
IT	IBN	[EPP12]	31	9	3.35	128	430
EU1	geant*	[Orl+10]	21	6	3.24	752	1263
EU2	nobel-eu	[Orl+10]	28	8	2.93	416	1059
EU3	cost266	[Orl+10]	37	8	3.08	438	1211
US1	abilene*	[Orl+10]	**11**	**5**	**2.55**	**993**	**1971**
US2	janos-us	[Orl+10]	26	8	3.23	601	1761
US3	CORONET	[Von+15]	**75**	**17**	2.64	330	1810

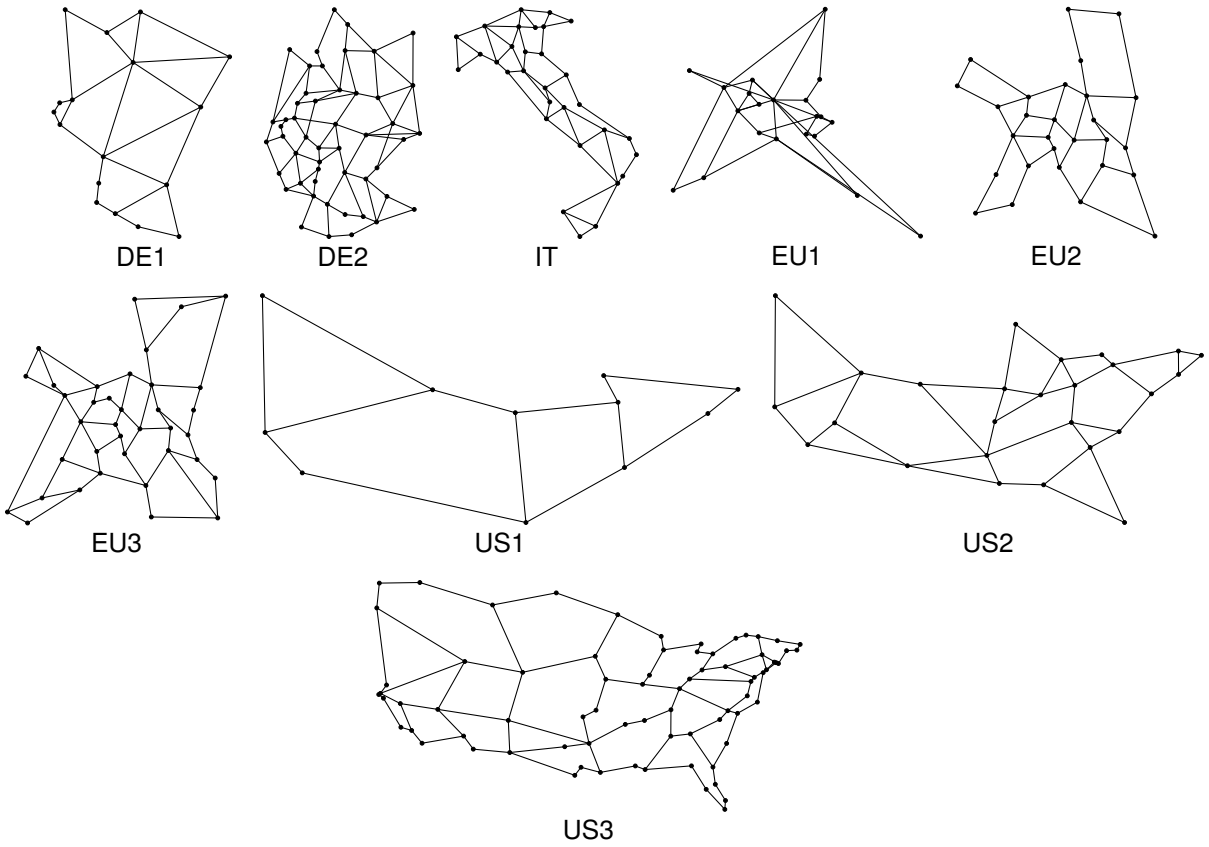


Figure 5.3 Fiber topologies of realistic networks

removed to avoid too strong an imbalance in link lengths. The fiber topologies are visualized in Figure 5.3.

For each network, we know in which cities the nodes are located. However, we do not know the exact cable runs of the links. Therefore, we take the straight-line distance between the incident nodes as the link length. Likewise, the *average node pair distance* given in Table 5.1 is computed from the straight-line distances between all node pairs.

5.1.4.2 Synthetic Network Topologies

The realistic topologies help to illustrate the behavior of different provisioning mechanisms in real-world scenarios. However, they all have different topological properties, and hence, it is difficult to assess the individual impact of each property on the provisioning mechanisms. In order to control those properties, we have created a set of synthetic network topologies in a randomized, simulated annealing-inspired generation procedure [KGV83].

All topologies have 15 nodes and an average link length of 100 km. Furthermore, all generated topologies are biconnected. The nodes are positioned randomly by a force-directed graph drawing algorithm [FR91]. The two properties we control are the average node degree and the diameter. Of the introduced realistic topologies, US1 with 11 nodes and DE1 with 17 nodes have a similar number of nodes. Their diameter is 5 and 6 hops, respectively, and their average node degree is 2.55 and 3.06. Therefore, we have selected a diameter of 5 hops and an average node degree of 2.8 as the *base parametrization* of the synthetic topologies. Additionally, we have generated topologies with an average node degree of 2.8 and a diameter of 4, 6, or 7 hops, and topologies with a diameter of 5 hops and an average node degree of 2.4, 3.2, 3.6, 4, or 4.4. This amounts to nine different topology parametrizations.

To increase the meaningfulness of our results, we have generated ten different topology instances per parametrization with different node adjacencies and node locations. All instances are pairwise non-isomorphic. In total, this amounts to 90 topology instances.

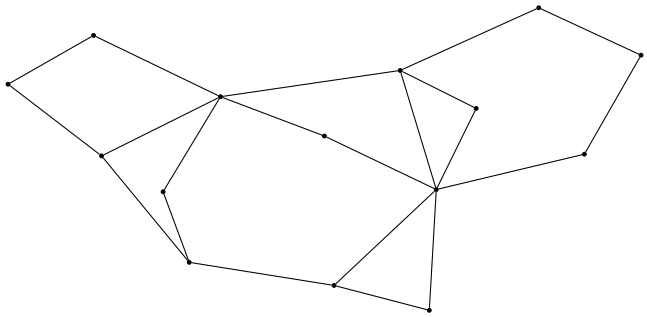


Figure 5.4 Synthetic, randomly generated topology instance BASE

Figure 5.4 depicts the first of the ten topology instances created with the base parametrization, i.e., a diameter of 5 hops and an average node degree of 2.8. We call this particular instance *BASE*. It will be employed in many of the following studies. The other synthetic topologies are shown in Appendix A.4.

5.1.4.3 Compensation Policies

Up to now, we have considered the compensation policy function of an SLA in a general way, only assuming it is a staircase function. However, for the following evaluation studies, we need specific functions. To this end, we derived two functions from existing compensation policies found in wavelength service SLAs of three large network operators [Lum23; Cen11; Ver20]. The functions are defined in Tables 5.2a and 5.2b.

We call the two resulting compensation policies *GOLD* and *SILVER*. Since the GOLD policy guarantees a higher availability than the SILVER policy, we consider the GOLD policy stricter than the SILVER policy. While the SILVER policy is compensation-free down to a monthly availability of 0.995, the GOLD policy only allows an availability down to 0.9999 without compensation. Both policies have seven compensation levels. The minimum

Table 5.2 Compensation policies derived from [Lum23; Cen11; Ver20]

(a) GOLD policy

Experienced unavailability	Experienced availability	Compensation (of *MRC*)
≤ 0.0001	≥ 0.9999	0 %
≤ 0.001	≥ 0.999	10 %
≤ 0.003	≥ 0.997	20 %
≤ 0.005	≥ 0.995	30 %
≤ 0.01	≥ 0.99	50 %
≤ 0.015	≥ 0.985	75 %
≤ 1	≥ 0	100 %

(b) SILVER policy

Experienced unavailability	Experienced availability	Compensation (of *MRC*)
≤ 0.005	≥ 0.995	0 %
≤ 0.01	≥ 0.99	10 %
≤ 0.015	≥ 0.985	20 %
≤ 0.02	≥ 0.98	30 %
≤ 0.025	≥ 0.975	50 %
≤ 0.035	≥ 0.965	75 %
≤ 1	≥ 0	100 %

compensation is no compensation at all. The maximum is 100 % of the monthly recurring charge (MRC). That means, in the worst case, the network operator has to pay the whole monthly rate back to the customer.

5.1.4.4 Link Capacity Dimensioning

In most studies, we assume that the network has a limited amount of network resources, i.e., each link has a certain capacity. Prior to the simulation, we dimension those link capacities. Since the node pairs of connection requests are selected uniformly, and we use shortest path routing, links that are more central in the network are more in demand than those further out. Therefore, we dimension the capacities based on the betweenness centrality of the links. According to [Bra08], the betweenness centrality of a link $e \in E$ is

$$c_{\mathrm{B},e} = \sum_{s,d \in V} \frac{\sigma(s,d|e)}{\sigma(s,d)} \tag{5.5}$$

where $\sigma(s,d)$ is the number of shortest paths between s and d, and $\sigma(s,d|e)$ is the number of those paths that also traverse the link e. We compute shortest paths based on the link length. Assuming that the more central a link, the more traffic it has to carry, we linearly map the centralities to link capacities between a minimum and a maximum number of traffic units $\kappa_{\min} \in \mathbb{N}$ and $\kappa_{\max} \in \mathbb{N}$, respectively. That means the capacity of link e is

$$\kappa_e = \left\lfloor \left(c_{\mathrm{B},e} - \min_{e \in E}\{c_{\mathrm{B},e}\} \right) \cdot \frac{\kappa_{\max} - \kappa_{\min}}{\max_{e \in E}\{c_{\mathrm{B},e}\} - \min_{e \in E}\{c_{\mathrm{B},e}\}} + \kappa_{\min} \right\rfloor. \tag{5.6}$$

We employ the floor operator because we work with integer-valued capacities only. In that way, the link with the lowest centrality has a capacity of $\kappa_{\min}$ traffic units, while the link with the highest centrality has a capacity of $\kappa_{\max}$ traffic units.

5.1.4.5 Calibration of a Specific Request Rejection Ratio

In the following studies, we compare the performance of different provisioning mechanisms. While some studies focus on accuracy or consider different network load situations, most

of the studies compare the mechanisms with respect to the achievable network capacity. As already explained in Section 5.1.3.3, we define the network capacity as the amount of carried load in the network at a request rejection ratio of 0.01.

The rejection ratio is a quantity measured during the simulation. It depends on the parameters of the connection requests, the network properties, and the provisioning mechanism in a complex way. Therefore, a simulation cannot be parametrized for a rejection ratio of 0.01 directly. Instead, we calibrate the rejection ratio by iteratively running simulations and adapting the mean IAT with an exponential curve fitting on the rejection ratios and mean IATs of the previous runs.

With this procedure, we can control the accuracy of the calibrated rejection ratios. However, we cannot guarantee that the ratios match 0.01 *exactly*. Consequently, when we compare different provisioning mechanisms or different network scenarios, we always compare them in slightly shifted operating points. To ensure that the simulation results are not significantly influenced by these shifts, we require an accuracy of 0.5 % for the calibrated rejection ratios. That means the final rejection ratios range from 0.00995 to 0.01005. This is a tradeoff between simulation time and solution quality. The simulation results in Section 5.5.1 show that this accuracy is sufficient for our evaluations.

5.2 Evaluation of the Accuracy

A core element of CAPSS is the calculation of the compliance probability for potential routes. If these calculations are not accurate, connections are provisioned on unsuitable routes that either under- or overfulfill the operator's compliance target. Therefore, the accuracy of the mathematical model for the compliance probability is crucial and will be evaluated in Section 5.2.1.

An additional feature of CAPSS is the sharing of compliance surplus. Ideally, the sharing mechanism eliminates the overfulfillment that is present in conventional provisioning mechanisms and in CAP. However, whether CAPSS succeeds in completely eliminating the overfulfillment again depends on the accuracy of the mathematical model and also on the network scenario. Thus, Section 5.2.2 evaluates the capability and accuracy of CAPSS as a whole. Section 5.2.3 summarizes the whole accuracy evaluation.

5.2.1 Accuracy of the Compliance Model

Parts of the model for the compliance probability have already been introduced by the author of this monograph in [End21; EK21]. In particular, a simpler version for the distribution of cumulated downtime per billing cycle and the expected amount of SLA compensation have been derived in [End21]. A simulative evaluation shows that the modeled expectations match the true means very well. This is already a useful result. However, two major differences to the approach in this thesis exist. First, as already mentioned, [End21] considers the *expected compensation* instead of the compliance probability. Second, it only considers an arbitrary billing cycle in a theoretically infinite contract period. In the thesis at hand, we consider finite contract periods. Therefore, the model derived in Section 4.4 incorporates a finite number of consecutive billing cycles.

In this section, we evaluate the accuracy of the model. To this end, we determine the compliance probability in different scenarios with the help of simulation. Then, we compare the simulated compliance probability with the one calculated with the model.

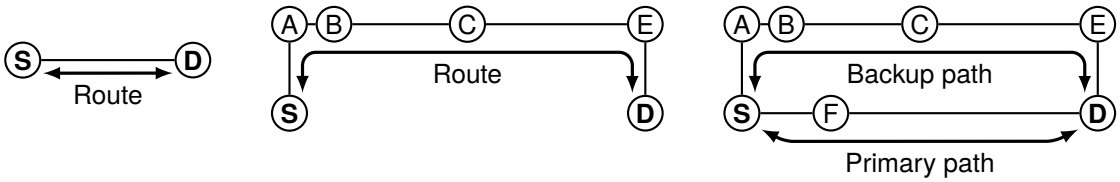
The model contains two approximations. First and foremost, the approximation that the compensations of consecutive billing cycles are independent of each other. Second, the approximation that a network element, formed by two parallel elements in the reliability block diagram, has exponential up- and downtimes. We put a special focus on the effects the two approximations have on the model accuracy.

Furthermore, the model assumes only one repair crew for parallel elements. As mentioned in Section 5.1.2.3, this assumption is not realistic, and therefore, the simulation does not adhere to it. We evaluate the effect this deviation has on the accuracy as well.

We do not evaluate the accuracy in scenarios in which the preconditions differ fundamentally from the preconditions we assumed in Section 4.2. For example, we do not consider *non*-exponential up- and downtimes of the underlying network elements or correlated failures due to natural disasters. Such studies are left for future work.

5.2.1.1 Evaluation Parameters

We evaluate the accuracy for three types of connections, namely a single hop connection, an unprotected connection, and a connection with path protection. With unprotected and protected connections, we evaluate the impact of the series and parallel block diagram reductions on the compliance probability. The single-hop scenario evaluates the fundamental compliance model without any other transformations.



(a) Single-hop **(b)** Unprotected **(c)** Protected

Figure 5.5 Evaluated network connections

All three connections are shown in Figure 5.5. The connections always connect nodes S and D. The route of the protected connection consists of a primary path via node F and a link-disjoint backup path (Figure 5.5c). Note that, unlike in our provisioning mechanisms, the backup path is enforced here.

As mentioned before, the assumption that consecutive billing cycles are independent of each other is an approximation. To study the impact on the model accuracy, we consider contract periods of lengths 1, 3, 6, 12, and 24 months. Further, we vary the allowed compensation, $c_{\max}$, between 0 and $4 \cdot MRC$. The compensation policy selected is the GOLD policy. In terms of connection availability, we consider two scenarios. Scenario A in which the connection (including protection) has $MTTF = 10^5$ h and $MTTR = 5$ h, and a less reliable scenario B with $MTTF = 10^4$ h and $MTTR = 25$ h. Scenario A covers compliance probability values around $\mathbf{P}(C \leq c_{\max}) = 0.85$ and above. Scenario B covers a broader range of compliance probabilities down to $\mathbf{P}(C \leq c_{\max}) \approx 0.2$.

The given MTTF and MTTR values apply to the whole connection. The corresponding CC and MTTR values for the underlying links are derived by inverting — numerically to some extent — the series and parallel reductions of Section 4.3.

5.2.1.2 Simulation Setup

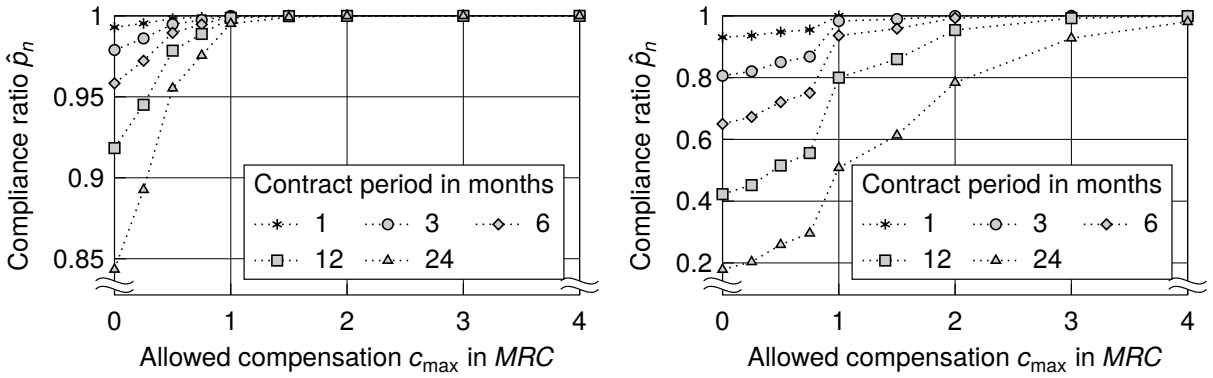
The simulation methodology is based on the methodology introduced in Section 5.1. Each simulation run covers one particular combination of the parameters discussed in the previous section and simulates a large number of connections. To avoid dependence between different connections, they are simulated back-to-back, i.e., only once the current connection has left the network does the next connection request arrive. In other words, the IAT between consecutive connection requests is constant and equals the contract period.

A single simulation run covers 10^7 years. Consequently, the maximum number of simulated connections with a two-year contract is 10^7 years/2 years $= 5 \cdot 10^6$. For one-month contracts, the maximum number of simulated connections is 10^7 years/(1/12 years) $= 1.2 \cdot 10^8$. The actual numbers are slightly lower because connection requests that arrive while the route is broken cannot be accepted. To provide CIs, we use the method of independent replications. With a confidence level of 95 %, we simulated between 10 and 30 replications in order to obtain sufficiently small intervals.

In a single simulation run, all connections share the same compliance probability. Also, all computed compliance probabilities, p_i $\forall i$, are the same. Nevertheless, we refer to the computed compliance as p_i, including the index. We compare p_i with the compliance ratio, $\hat{p}_n$, which serves as an estimator for the true compliance probability.

5.2.1.3 Results

Compliance ratio Figure 5.6 shows the simulated compliance ratio, $\hat{p}_n$, over the allowed compensation, $c_{\max}$, for the single-hop connection. The left figure corresponds to scenario A, while the right figure corresponds to scenario B. The values for the unprotected and protected connections equal those of the single-hop connection. Therefore, they are not shown here. The largest 95 % CI for the compliance ratio is $1.9 \cdot 10^{-4}$. Hence, the CI markers are too small to be visible in Figure 5.6. Also, since the CIs are so small, $\hat{p}_n$ approximates the true compliance probability, $\mathbf{P}(C \leq c_{\max})$, very well in these simulations. Therefore, we use the terms *compliance ratio* and *true compliance probability* interchangeably throughout the discussion of this study.



(a) Scenario A **(b)** Scenario B

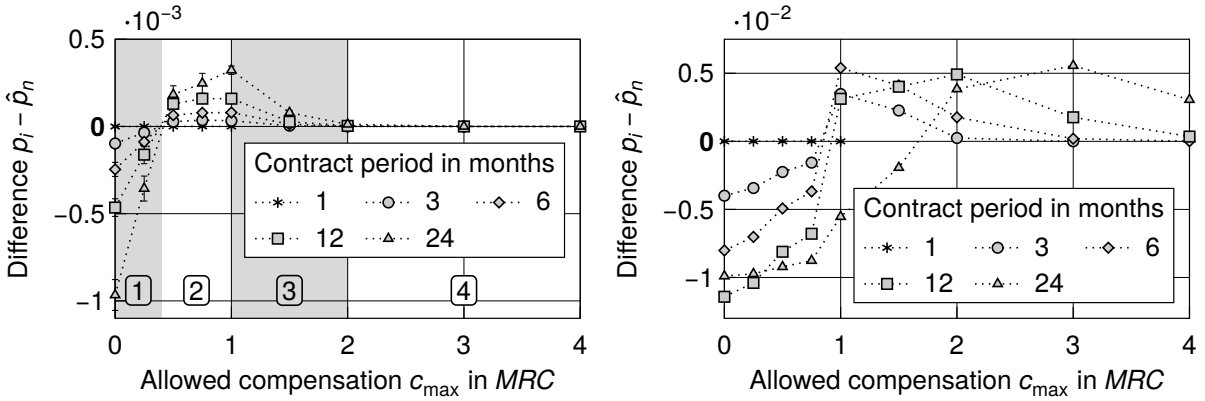
Figure 5.6 Simulated compliance ratio, $\hat{p}_n$, for the single-hop connection

For the contract periods of one and three months, the simulation values for $c_{\max} > 1 \cdot MRC$ and $c_{\max} > 3 \cdot MRC$, respectively, are missing. The reason is that the maximum monthly compensation in the employed compensation policy GOLD is 100 % of the MRC. Therefore,

the total compensation of a connection with a contract period of one month cannot exceed $1 \cdot MRC$. Likewise, the total compensation of a connection with a contract period of three months cannot exceed $3 \cdot MRC$. Consequently, the compliance probability, $\mathbf{P}(C \leq c_{\max})$, is trivially 1 for $c_{\max} \geq 1 \cdot MRC$ or $c_{\max} \geq 3 \cdot MRC$, respectively. For this reason, parameter combinations with higher $c_{\max}$ values have not been simulated.

As can be seen in Figure 5.6, the compliance ratio increases when the allowed compensation increases. Also, it increases when the contract period decreases because less compensation accumulates with a shorter contract period.

Single-hop accuracy Figure 5.7 shows the difference or error, respectively, between the computed compliance probability and the simulated compliance ratio for the single-hop connection. Negative values, e.g., in area 1 of Figure 5.7a, signify an underestimation of the true compliance probability. The CIs correspond to the CIs of the compliance ratio because the compliance probability computed by the model, p_i, is a constant. When comparing Figures 5.7a and 5.7b, note the different y-axis scalings.



(a) Scenario A, single-hop

(b) Scenario B, single-hop

Figure 5.7 Difference between the computed compliance probability, p_i, and the simulated compliance ratio, $\hat{p}_n$, for the single-hop connection – Note the different y-axis scalings.

A first important result is that the compliance model is accurate for a contract period of one month. In fact, the minimum and maximum difference for this case is $-2.4 \cdot 10^{-7}$ and $7.1 \cdot 10^{-6}$, respectively. In this evaluation study, a one-month contract consists of a single billing cycle only.[4] Therefore, the approximation that consecutive billing cycles are statistically independent of each other is irrelevant. Consequently, no approximation error arises. For longer contract periods, i.e., multiple billing cycles, a significant difference between the model and the simulation is visible. For low values of $c_{\max}$ (e.g., area 1 of Figure 5.7a), the difference is negative, i.e., the actual probability is higher than the computed value. In CAPSS, this underestimation leads to a conservative behavior that is characterized by overfulfillment of the targeted compliance probability. When $c_{\max}$ increases from 0, the *absolute* approximation error decreases at first (area 1). It then grows again with an overshoot-like behavior (area 2) before it finally approaches zero (areas 3 and 4). A positive difference means the compliance probability is overestimated. In CAPSS, this leads to aggressive behavior that is characterized by underfulfillment of the targeted compliance probability.

[4]Generally, a one-month contract contains two billing cycles except if the beginning of the contract coincides with the beginning of the calendar month, as is the case in this study.

In scenario A (Figure 5.7a), the contract period impacts the severity of the approximation error. However, it has no significant impact on the ranges of $c_{\max}$ in which the error is negative, positive, or converges to zero. In scenario B (Figure 5.7b), which represents a less reliable connection, the error is approximately one order of magnitude more severe than in scenario A. Also, the contract period *does have* an impact on the ranges in which the error is negative, positive, or converges to zero. The longer the contract period, the higher $c_{\max}$ has to be in order for the error to vanish. Overall, that means that the error of the computed compliance probability is more severe in a wider range of parametrizations for connections with low reliability and long contract periods.

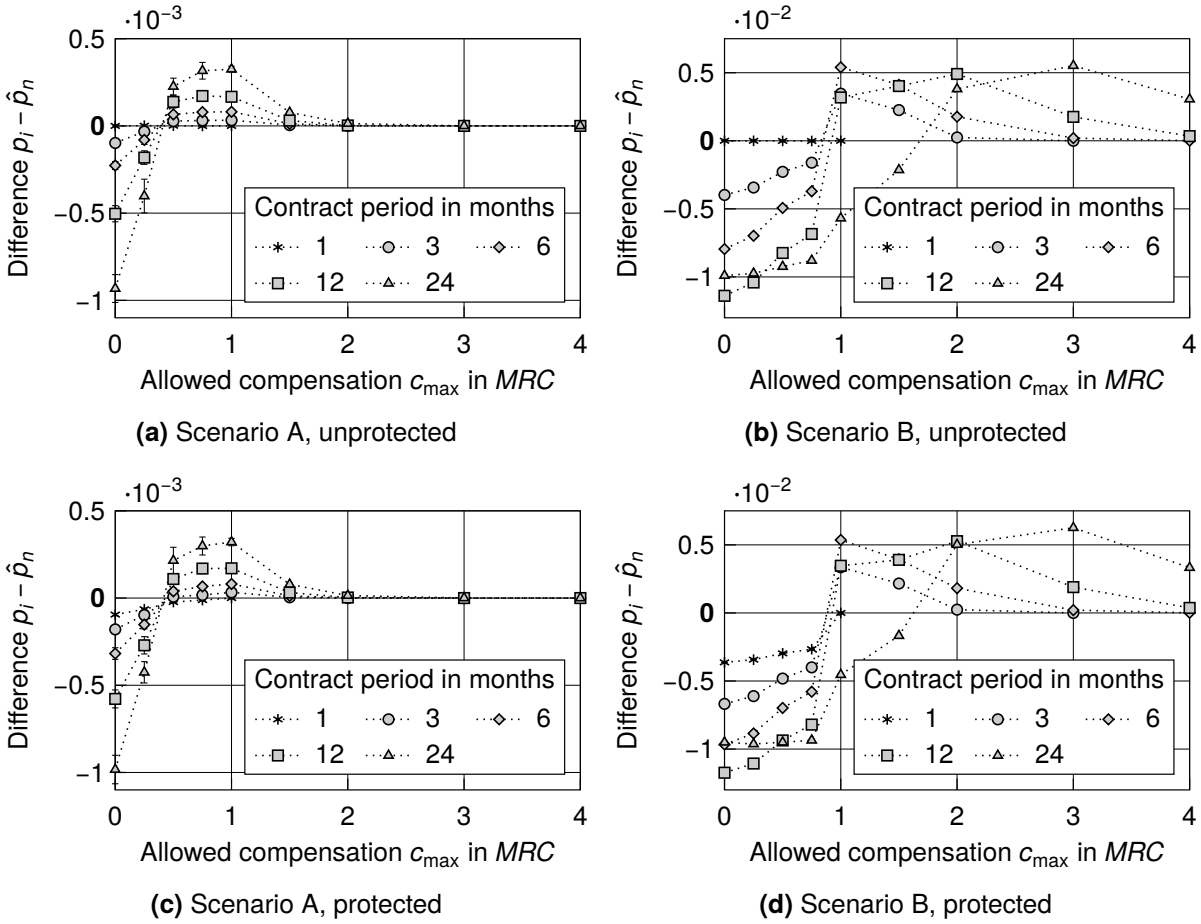


Figure 5.8 Difference between the computed compliance probability, p_i, and the simulated compliance ratio, $\hat{p}_n$, for the unprotected connection (top row) and the protected connection (bottom row) – Note the different y-axis scalings.

Unprotected accuracy Figures 5.8a and 5.8b show the approximation error for the unprotected connection. There is virtually no difference from the approximation error of the single-hop connection. This suggests that the series reduction in the reliability block diagram itself does not introduce any additional error, even though the simulation violates the assumption of no concurrent link failures. However, it must be said that more than 99.8 % of all link failures were single-link failures in the simulations. Therefore, conclusions about the concurrency assumption are only of limited significance.

Protected accuracy Lastly, Figures 5.8c and 5.8d show the approximation error for the protected connection. A major difference from the previous results is that an approximation

error exists also for the one-month contract period. The reason for this is the parallel reduction in the reliability block diagram. In this parallel reduction, we assume that the resulting block's up- and downtimes follow exponential distributions again. However, this is only an approximation. Another assumption the model is based on is that only one repair crew is available per parallel structure in the block diagram. In the simulation, each link has its own repair crew. Consequently, the downtimes are shorter than anticipated by the model, and the true compliance probability is higher than the compliance probability computed by the model. The error is visible for longer contract periods as well. However, compared with the error that arises from the independence assumption of consecutive billing cycles, the parallel reduction error is less severe. Also, it loses significance when $c_{\max}$ grows above $1 \cdot MRC$.

5.2.1.4 Summary

We evaluated the accuracy of the model for the compliance probability in two scenarios with different availability levels, for different allowed compensations, and for various contract periods. The results show that the model is accurate for one-month contract periods and unprotected connections. In contrast, multi-month contracts and protection result in small errors. Those errors are well expected because they are caused by a deliberate violation of assumptions and by approximations in the model.

Generally, protected low-availability connections with long contracts experience the worst errors with absolute magnitudes on the order of 10^{-2}. For low levels of allowed compensation — which we consider the default — the model underestimates the compliance probability. In CAPSS, this leads to conservative behavior and does not expose the network operator to the risk of exceeding its compliance target.

5.2.2 Accuracy of CAPSS

A core functionality of CAPSS is the prevention or at least reduction of compliance overfulfillment. In this section, we evaluate this functionality in a network simulation. While Section 5.2.1 only evaluated the quality of the compliance model, this section considers the whole CAPSS algorithm, including the sharing of compliance surplus among connections. In particular, we study how well the average computed compliance probability, $\overline{p}_n$, and the compliance ratio, $\hat{p}_n$, match the compliance target, p_t.

5.2.2.1 Simulation Setup and Evaluation Parameters

The behavior is evaluated in the BASE topology depicted in Figure 5.4. It consists of 15 nodes and is biconnected. It has an average node degree of 2.8, a diameter of 5 hops, and an average link length of 100 km.

We simulate a variety of parameters in order to cover a broad range of network and algorithm scenarios with the evaluation. We study MTTR values of 1, 5, 20, and 50 hours, and CC values of 100 km and 1000 km. Connection requests arrive following a Poisson distribution with a mean IAT of 12 hours. The contract period is set to 1, 3, 6, 12 or 24 months. The allowed compensation ranges from 0 to $2 \cdot MRC$, and the compliance target is either 0.9 or 0.99. The compensation policy is either GOLD or SILVER. In total, we simulate 1280 parameter combinations. We simulate ten batches of 5000 years each for every parameter combination. This yields around $3.6 \cdot 10^6$ simulated connections per combination.

We allow segment protection, i.e., the connections can be unprotected, partially protected, or fully path-protected. Furthermore, we assume infinite link capacities. Consequently, connection requests are only rejected if all feasible routes are broken at the time of request arrival.

5.2.2.2 Results

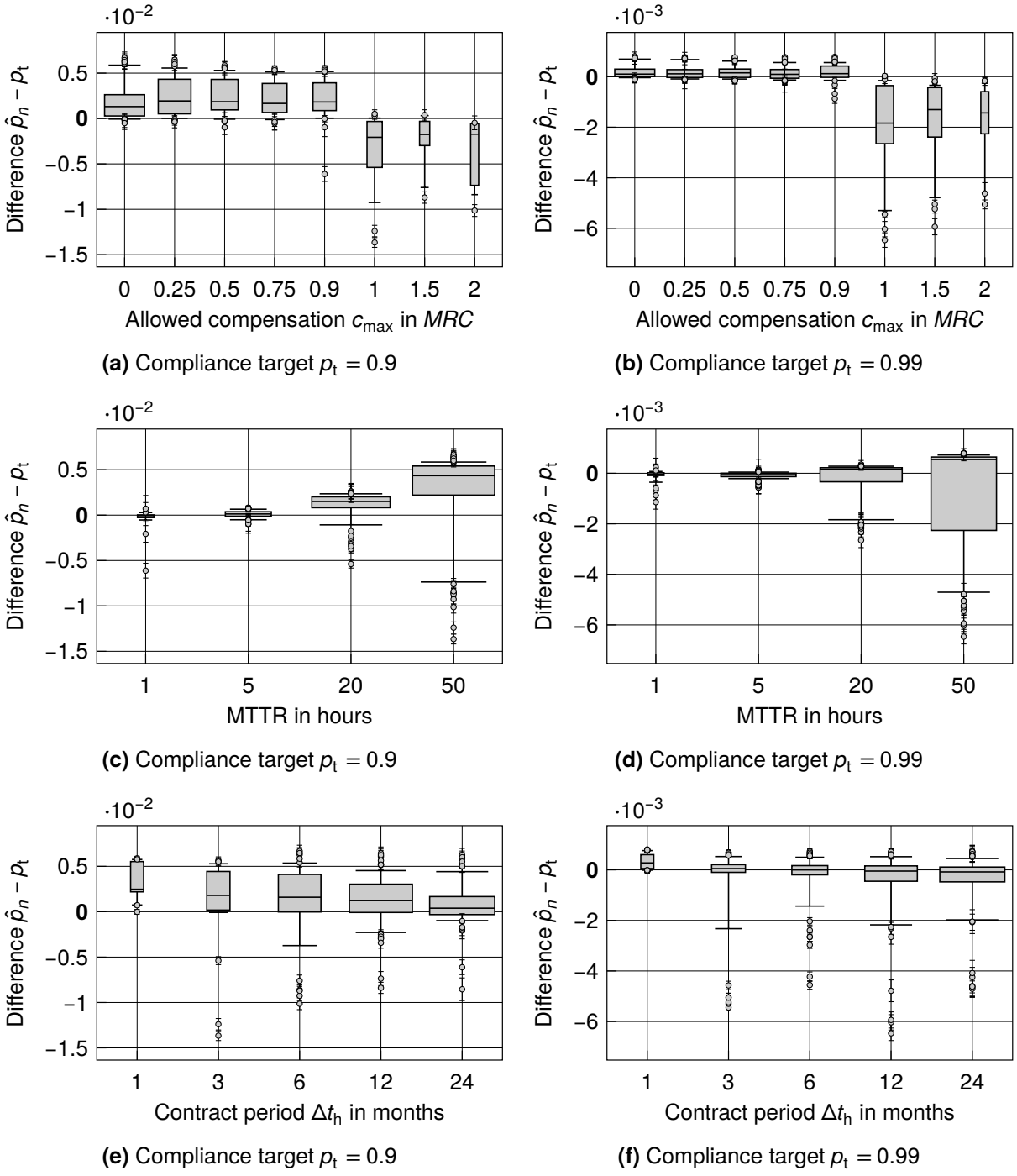
Applicability For 70 combinations with low availability ($MTTR \in \{20\,\text{h}, 50\,\text{h}\}$ and $CC = 100\,\text{km}$) and strict compensation targets ($c_{\max} < 1 \cdot MRC$ and $p_\text{t} = 0.99$), the ratio of infeasible requests is above 50 %. That means only node pairs with a short distance can be connected, and most long-distance connections are not possible. Since this is not representative, we ignore those parameter combinations in the following presentation of results.

On the other hand, for 616 parameter combinations, i.e., in 48 % of the cases, CAPSS is not able to prevent compliance overfulfillment entirely. The reason is that in those cases, none of the node pairs requires a protected connection to satisfy the targeted compliance. An unprotected connection is sufficient. Since no protection is needed — and connections can always be routed on the shortest path due to infinite link capacity — connections cannot be relaxed. Consequently, compliance surplus accumulates, but CAPSS has no way to reduce it. This is a clear limitation of the whole approach in terms of overfulfillment reduction. However, from an operator's perspective, such a scenario is anything but bad because it is resource-optimal. In this study, we focus on evaluating the functionality of the surplus sharing mechanism. Since the sharing mechanism is not used in the described parameter combinations, we do not show them in the following presentation of results.

This leaves us with 594 parameter combinations with an infeasibility ratio below 50 % and the need for protection. The need for protection means connections can "absorb" compliance surplus. This is the theoretical precondition for successful overfulfillment prevention (see Section 4.1.2). Two statistics are important for the evaluation of CAPSS's functionality, namely the compliance ratio, $\hat{p}_n$, and the average computed compliance probability, $\overline{p}_n$. The compliance ratio estimates the "real" compliance probability that is effectively realized. On the other hand, the average computed compliance probability only comprises the computed probabilities CAPSS "sees." The best CAPSS can do is prevent any difference between the average computed compliance and the compliance target. However, even if this is the case, the final compliance ratio can still deviate from the compliance target, e.g., due to approximation errors in the compliance model identified in Section 5.2.1.

Average computed compliance Let us consider the average computed compliance probability, $\overline{p}_n$, first. If $\overline{p}_n = p_\text{t}$, CAPSS works as expected. Therefore, for every parameter combination, the average computed compliance, $\overline{p}_n$, and its corresponding 95 % CI have been measured in the simulations. If the compliance target, p_t, is outside the CI, the difference $\overline{p}_n - p_\text{t}$ is significant on the 95 % confidence level. This is the case for 31.3 % of the parameter combinations. Consequently, the differences are *not* significant in 68.7 % of the cases. The worst absolute difference, $|\overline{p}_n - p_\text{t}|$, over both groups — significant difference or not — is $5.5 \cdot 10^{-8}$ for $p_\text{t} = 0.9$ and $1.4 \cdot 10^{-8}$ for $p_\text{t} = 0.99$. Therefore, even though some significant differences exist, these differences are negligible in practice. Overall, the results suggest that the *surplus sharing mechanism* itself works as intended.

Compliance ratio As already mentioned, the fact that the average computed compliance matches the compliance target does not necessarily mean that the compliance ratio also matches the target. To shed more light on this, Figure 5.9 shows box plots for the difference



(a) Compliance target $p_t = 0.9$

(b) Compliance target $p_t = 0.99$

(c) Compliance target $p_t = 0.9$

(d) Compliance target $p_t = 0.99$

(e) Compliance target $p_t = 0.9$

(f) Compliance target $p_t = 0.99$

Figure 5.9 Difference between the compliance ratio, $\hat{p}_n$, and the compliance target, p_t – Note the different y-axis scalings.

$\hat{p}_n - p_t$.

Each figure on the left, i.e., Figures 5.9a, 5.9c, and 5.9e, contains the differences of all 249 parameter combinations with $p_t = 0.9$. However, each figure groups the data points differently, either by the allowed compensation, the MTTR, or the contract period. The other varied parameters do not impact the difference significantly and are not shown. On the right, each figure covers the 345 parameter combinations for which $p_t = 0.99$.

Each figure contains several box plots. The number of data points per box plot varies

because the ignored parameter combinations are not distributed evenly. The width of a box plot shows its relative number of data points compared with the other box plots in the same figure. When comparing the left and the right column, note the different y-axis scalings. For $p_t = 0.9$, the 95 % CIs of the underlying data points range from $2.8 \cdot 10^{-4}$ to $2.9 \cdot 10^{-3}$. For $p_t = 0.99$, they range from $6.1 \cdot 10^{-5}$ to $1.0 \cdot 10^{-3}$.

A positive difference, $\hat{p}_n - p_t$, corresponds to residual compliance overfulfillment CAPSS was not able to prevent. On the other hand, a negative difference corresponds to compliance underfulfillment. As mentioned above, the average computed compliance, $\bar{p}_n$, matches the compliance target very well, with maximum deviations on the order of 10^{-8}. Consequently, any differences of higher magnitude are either uncertainty resulting from the finite simulation time or are caused by the approximation errors in the compliance model. Since most CIs are comparatively tight, large differences can only be explained by approximation errors.

As can be seen in Figures 5.9a and 5.9b, the allowed compensation has a considerable impact on the difference. For $c_{max} < 1 \cdot MRC$, the compliance target is overfulfilled by almost all parameter combinations. For $c_{max} \geq 1 \cdot MRC$, the difference turns into underfulfillment with a higher spread and potentially a higher absolute value. The behavior can be explained by the findings in Section 5.2.1. There, it was shown that the compliance probability model underestimates the compliance probability for small c_{max}, leading to overfulfillment in CAPSS. For larger c_{max}, the model overestimates the probability, which leads to underfulfillment.

Figures 5.9c and 5.9d show the impact of the MTTR on the difference. As can be seen, the spread in values grows with the MTTR, i.e., the ability of CAPSS to meet the compliance target deteriorates. Both over- and underfulfillment become more severe. However, the median increases with the MTTR, making compliance overfulfillment more likely than underfulfillment. Again, this behavior is in line with the findings in Section 5.2.1, where the less reliable scenario B with a higher MTTR caused more severe approximation errors.

Figures 5.9e and 5.9f show the impact of the contract period on the difference. In both figures, the spread in values for a one-month contract period is lower than for longer contract periods. This can be explained by the fact that the model for the compliance probability assumes independence of consecutive billing cycles. However, this independence is not given in reality and in this simulation. The fact that even the one-month compliance ratio deviates has two explanations. First, the connection requests arrive randomly in this simulation. It is unlikely that the arrival of a connection request coincides with the beginning of a calendar month. Consequently, most one-month contracts comprise two billing cycles. This may already introduce an approximation error. Second, the connections in this simulation can be protected. This results in another error component (see Section 5.2.1).

In Figure 5.9e, i.e., for $p_t = 0.9$, the box plot whiskers, which mark the spread of 80 % of the data points, are furthest apart for a contract period of 6 months. For longer periods, the spread decreases again. On the other hand, both the median and the interquartile range monotonically decrease and approach zero while the contract period grows from 1 to 24 months. For $p_t = 0.99$ (Figure 5.9f), the contract periods 3, 6, 12, and 24 months do not differ significantly.

Lastly, we consider Figure 5.9 column by column. The differences in the left column ($p_t = 0.9$) are on the order of 10^{-2}. In the right column ($p_t = 0.99$), they are smaller, on the order of 10^{-3}. That means, in the presented scenarios, CAPSS is accurate at least down to the number of decimal places the compliance target has.

5.2.2.3 Summary

This study considered CAPSS's overfulfillment prevention capability. It was first shown that CAPSS cannot reduce compliance overfulfillment in network scenarios with infinite link capacity and no need for protection because compliance surplus cannot be spent. In other scenarios, the evaluation of the average computed compliance probability showed that the surplus sharing mechanism implemented in CAPSS works as intended. The deviations of the *average computed compliance probability* from the compliance target are negligible. However, due to approximation errors in the mathematical model, the *compliance ratio* suffers from larger deviations. Both over- and underfulfillment of the compliance target occur. In particular, CAPSS overfulfills the compliance target for $c_{\max} < 1 \cdot MRC$ and underfulfills it otherwise. Furthermore, the deviation becomes worse when the MTTR increases, when the contract period exceeds one month, or when the compliance target decreases. Overall, the highest observed deviations are on the order of 10^{-2}, which is in line with the results in Section 5.2.1.

5.2.3 Summary of the Accuracy Evaluation

In the previous sections, we evaluated the accuracy of the compliance probability model and of CAPSS overall. The results for the model alone show small errors for multi-month contracts or if protection is used. Those errors are expected since they are caused by the approximations the model includes and by the violation of model assumptions in the simulation. The worst errors are on the order of 10^{-2}. However, the majority of observed errors are much smaller.

Concerning the surplus sharing mechanism implemented in CAPSS, the results show that it works as intended. The differences between the average computed compliance and the compliance target are negligible. However, due to the approximation errors in the compliance model, the compliance ratio suffers from deviations up to around 10^{-2} as well. Both over- and underfulfillment of the compliance target occur. However, low values for the allowed compensation and a low MTTR result in small deviations and conservative behavior of CAPSS.

Overall, we consider both the model and CAPSS as sufficiently accurate for network operation. Therefore, the following sections will not discuss the accuracy again. Instead, they focus on the effects CAPSS has on the network. Furthermore, they evaluate the performance with respect to the network capacity and the rejection ratio.

5.3 Illustration of Compliance Surplus Sharing

Section 5.2 showed the accuracy of the developed mathematical model and the surplus sharing mechanism. In this section, we go through an illustrative example in order to make the surplus sharing mechanism intuitively accessible in different network situations. This will help to better understand the evaluations in the subsequent studies. To illustrate the effects of surplus sharing, we compare CAPSS and CAP because their only difference is precisely the surplus sharing feature.

Section 4.1.2 describes two main effects surplus sharing can have on the network, namely lowered protection requirements and the possibility to select longer, less reliable routes. Both effects will be observable in the following example.

5.3.1 Simulation Setup and Evaluation Parameters

We use the three-node network depicted in Figure 5.10. Since one of CAPSS's goals is to increase the network capacity, we study the behavior of the two provisioning algorithms for an offered load that greatly exceeds the total link capacity. Furthermore, we vary the reliability level of the network. To that end, we keep the MTTR of the links constant at 9 hours [Ver+05], but we vary the CC value between 10 km and 7000 km.

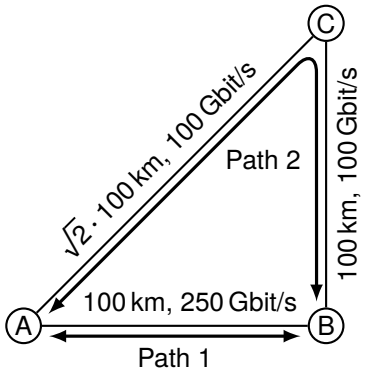


Figure 5.10 Three-node network topology with link lengths and link capacities

We assume that connections are requested between nodes A and B only. A connection can be routed on path 1, on path 2, or on both paths in case path protection is needed. Path 1 has a capacity of 250 Gbit/s, while path 2 has a capacity of 100 Gbit/s. For the purpose of demonstration, let each connection request a data rate of 1 Gbit/s. Also, let each connection stay in the network for one year. Assume that the network operator allows a compensation of $c_{\max} = 0.25 \cdot MRC$ with a compliance target of $p_t = 0.95$. As compensation policy, we use the GOLD policy. As can be seen in Figure 5.10, the links A–B and B–C each have a length of 100 km, while the link A–C has a length of $\sqrt{2} \cdot 100$ km. As an example, with a value of $CC = 2000$ km, the compliance probability of a connection routed on path 1 would be

$$p_1 = 0.96. \tag{5.7}$$

On path 2, it would be

$$p_2 = 0.91 \tag{5.8}$$

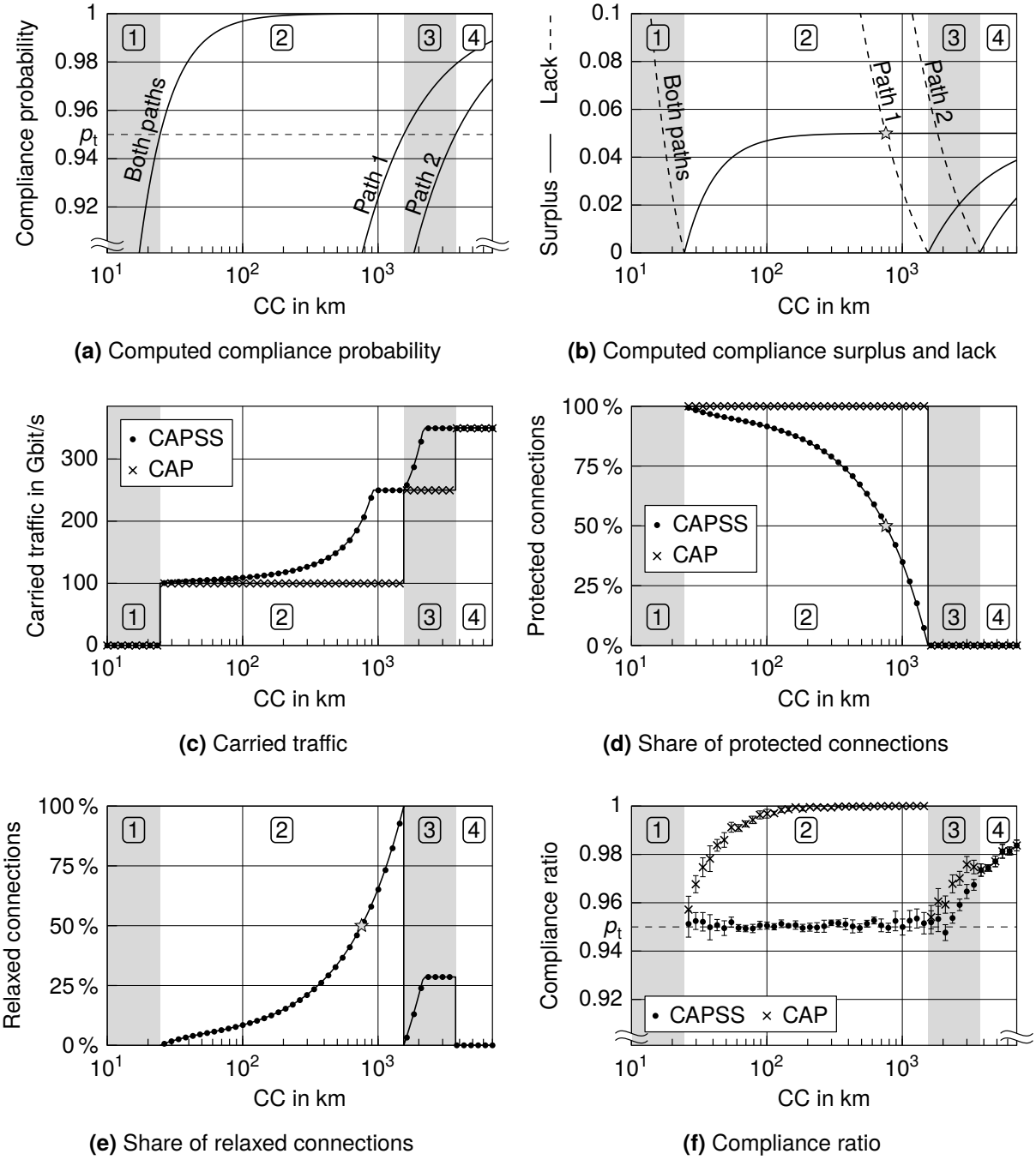
and using both paths in parallel in the mode of path protection, it would be

$$p_{12} = 0.999992. \tag{5.9}$$

Clearly, with a compliance target of $p_t = 0.95$, routing on path 2 without surplus sharing is not possible because $p_2 < p_t$. On the other hand, routing on path 1 or using path protection provides enough probability. In the following, we will see how the routing changes when the link reliability changes and when surplus sharing is activated. We will derive analytic expressions for the behavior of various statistics and verify them by simulation. In the simulation, connection requests are generated according to a Poisson process. The offered load, i.e., the product of the average request arrival rate and the contract period, is set to 1000 in order to fill the network. One simulation run with surplus sharing (CAPSS) and one without (CAP) was conducted for each CC value. A single simulation run comprises a startup phase of 1000 years followed by ten batches of 1000 years each. The result plots that follow show mean values and CIs computed from the ten batches.

5.3.2 Analytical and Simulated Results

Figure 5.11 shows various statistics that are either simulated, computed analytically, or both. Except for Figure 5.11f, the 95 % CIs are too small to be visible.



(a) Computed compliance probability

(b) Computed compliance surplus and lack

(c) Carried traffic

(d) Share of protected connections

(e) Share of relaxed connections

(f) Compliance ratio

Figure 5.11 Analytically computed values (lines) and simulated mean values (marks) for the three-node example

Figure 5.11a shows the compliance probability of the three existing routes over CC values from 10 km to 7000 km. As *CC* grows from left to right, the links become more reliable. The values p_1, p_2, and p_{12} from the example above for $CC = 2000$ km can be identified in the figure. The dashed line indicates the operator's compliance target, p_t. Without surplus

sharing, a route is suitable only when its compliance probability is at least p_t. In area ①, no route is suitable. At $CC = 24.6$ km, the border between area ① and ②, the compliance probability of the protected route equals p_t. Starting from that value, the protected route, and hence the network, can carry connections. In area ③, the compliance probability of path 1 exceeds p_t, and, finally, in area ④, the probability of path 2 follows suit.

Figure 5.11b shows the compliance surplus and the *compliance lack*. While compliance surplus is the difference between a route's compliance probability, p_i, and the compliance target, p_t, when $p_i \geq p_t$, compliance lack is the difference between the compliance target and a route's compliance probability when $p_i < p_t$. The values in Figure 5.11b are derived from the values in Figure 5.11a. Surplus is represented by a solid line, while lack is shown as a dashed line. The star marks the point at which the surplus of the protected route equals the lack of the route using path 1. This point will be discussed later after the other figures have been introduced. Figures 5.11a and 5.11b illustrate again the problem introduced in Section 3.3.3, namely the fact that, usually, there is no route that matches the targeted compliance probability exactly. As we will see in the following, the result is overfulfillment and a waste of resources.

Figure 5.11c shows the routed traffic with and without surplus sharing, i.e., using CAPSS and CAP, respectively. Figure 5.11d shows the corresponding shares of protected connections. The marks correspond to the simulated mean values, while the lines correspond to analytically computed values, which we derive in the following. We consider the case *without* surplus sharing (CAP) first. In area ①, no route provides sufficient compliance surplus, so no connections can be routed. In area ②, connections can be routed using path protection, i.e., using paths 1 and 2 in parallel. Consequently, the share of protected connections is 100 %, as shown in Figure 5.11d. The accommodated traffic rises to 100 Gbit/s. Since path protection occupies both paths equally, both path 1 and path 2 each carry 100 Gbit/s. While in that way, path 2 is fully utilized, the utilization of path 1, whose capacity is 250 Gbit/s, only reaches 40 %. Clearly, path 2 is a bottleneck throughout area ②. In area ③, routing on path 1 alone, i.e., without protection, becomes feasible. Consequently, the protection ratio drops to 0 %. All connections are now routed on path 1 only. Path 1 is fully exhausted, and the total traffic in the network equals the capacity of path 1, namely 250 Gbit/s. Finally, in area ④, routing on path 2 becomes feasible as well. Therefore, in addition to all the connections on path 1, path 2 is filled, too. The total traffic reaches 350 Gibt/s, i.e., the network is fully utilized.

Next, let us consider the behavior when surplus sharing is active. CAPSS behaves like CAP in area ① because no route provides sufficient compliance probability, and therefore, also no surplus can be accumulated for sharing. In area ②, connections are routed with protection, which yields compliance surplus. This surplus is used to relax subsequent connection requests, which can then be routed without protection. The more surplus provisioned connections generate and the less lack alternative routes have, the more subsequent connections can be provisioned without protection. Figure 5.11d shows how the share of protected connections decreases when *CC* — and, consequently, also the surplus — increases. Both the share of protected connections and the amount of routed traffic can be estimated from the ratio

$$\gamma_2 = \frac{p_{\text{lack},1}}{p_{\text{surplus},12}} \tag{5.10}$$

where $p_{\text{lack},1}$ is the compliance lack of path 1 and $p_{\text{surplus},12}$ is the compliance surplus of the protected route. Both $p_{\text{lack},1}$ and $p_{\text{surplus},12}$ are functions of *CC* here. The ratio γ_2 describes the number of connections that need to be routed in order to relax one additional connection

request. As an example, consider the points marked by the star in Figures 5.11b and 5.11d. In Figure 5.11b, the star marks the point at which the compliance lack of the route using path 1 equals the compliance surplus of the protected route, i.e., $\gamma_2 = 1$. At this point, routing one connection on the protected route accumulates just enough surplus to route the next connection without protection, i.e., on path 1 only. Consequently, the share of protected connections is 50 %, as can also be seen in Figure 5.11d. For lower CC values, more than one protected connection is needed to accumulate sufficient surplus. For higher CC values, one protected connection generates more surplus than is needed to relax a single connection.

A metric closely related to the share of protected connections is the share of *relaxed* connections. It is depicted in Figure 5.11e. In area ②, the higher *CC*, the more connections are relaxed. For the same reasons as above, for $\gamma_2 = 1$, the share of relaxed connections is 50 %.

Generally, in area ②, the share of protected connections is

$$\frac{\gamma_2}{\gamma_2 + 1} \tag{5.11}$$

because for γ_2 protected connections, one additional connection without protection can be accommodated on path 1. Similarly, the share of relaxed connections equals

$$\frac{1}{\gamma_2 + 1}. \tag{5.12}$$

The total traffic of 100 Gbit/s, which was achieved without surplus sharing, has now increased by a factor of $(\gamma_2 + 1)/\gamma_2$. Since the total traffic is also limited by the first path's capacity of 250 Gbit/s, the total traffic in area ② is

$$\min\left(\frac{\gamma_2 + 1}{\gamma_2} \cdot 100\,\text{Gbit/s}, 250\,\text{Gbit/s}\right). \tag{5.13}$$

In area ③, there are no protected connections, no matter if surplus sharing is active or not. Nevertheless, some connections are relaxed, as can be seen in Figure 5.11e. In contrast to area ②, where a relaxation of the compliance target leads to the avoidance of protection, in area ③, relaxation allows the use of path 2 for unprotected routing. That means surplus sharing increases the path diversity and, in that way, makes available additional network capacity. As can be seen in Figure 5.11c, the total traffic grows to the maximum of 350 Gibt/s long before CAP reaches this value in area ④. In area ③, the number of connections that need to be routed in order to relax one additional connection request is

$$\gamma_3 = \frac{p_{\text{lack},2}}{p_{\text{surplus},1}}. \tag{5.14}$$

With this, and since the capacity of the second path is only 100 Gbit/s, the share of relaxed connections is

$$\min\left(\frac{1}{\gamma_3 + 1}, \frac{100\,\text{Gbit/s}}{350\,\text{Gbit/s}}\right). \tag{5.15}$$

For the total traffic in area ③, we obtain

$$\min\left(\frac{\gamma_3 + 1}{\gamma_3} \cdot 250\,\text{Gbit/s}, 350\,\text{Gbit/s}\right). \tag{5.16}$$

Finally, in area ④, path 2 is feasible even without surplus sharing. Therefore, surplus sharing does not provide any additional benefit, and both CAPSS and CAP behave equally.

Figure 5.11f shows the measured compliance ratio. With CAP, the compliance target, p_t, is exceeded in almost all simulation points. The only exceptions are the very beginning of areas ② and ③ because here, the routes' compliance probabilities are close to p_t anyway. With CAPSS, the compliance ratio stays around p_t in area ② and at the beginning of area ③. For higher CC values, i.e., at the end of area ③ and in area ④, the compliance ratio starts to grow as well. The reason is that the links become more and more reliable, but the additional surplus cannot be used to further increase the amount of routed traffic because the network is already at the capacity limit.

5.3.3 Summary

In this illustrative example, we compared CAPSS and CAP to demonstrate the major effects of surplus sharing, namely lowered protection requirements and a more diverse set of potential routes. Both effects increase the amount of carried traffic. Thus, the network is used more efficiently. The example also shows that surplus sharing is not possible for very high network availabilities because, with high availability, also CAP can use the full path diversity and does not need protection. Hence, there is no more capacity to gain using CAPSS.

Since the example considers connections between a single pair of nodes only, the two effects appear separated from each other in different ranges of the CC parameter. Also, it is possible to analytically model the behavior of the algorithm or the network, respectively. For more complex networks with a multitude of connected node pairs, the effects cannot be separated that clearly, and it is not possible to capture the behavior analytically. Nevertheless, the fundamental observations can still be transferred to more complex network scenarios.

5.4 Comparison of CAPSS and Conventional Provisioning

CAPSS is a compensation-aware provisioning mechanism. As such, it is implicitly also an availability-aware provisioning mechanism. Even though the perspectives and goals of CAPSS and conventional availability-aware mechanisms like CONV are very different, it is still important to compare those mechanisms. In particular, network operators that consider switching mechanisms need to know the behavior of CAPSS compared with their current provisioning strategy. To this end, we compare CAPSS and CONV in the following sections. In Section 5.4.2, we discuss the simulation results in synthetic network scenarios to identify and separate important effects. In Section 5.4.3, we compare the two mechanisms in the realistic networks with mixed connection requirements. Section 5.4.4 summarizes the results.

5.4.1 Simulation Setup and Evaluation Parameters

The goal of CONV is to select a route whose steady-state availability fulfills the SLA requirement. In contrast, CAPSS selects routes that fulfill the operator's compliance target. Since the two goals are very different, the mechanisms cannot be compared in a straightforward way by simply parametrizing each of them to reach a single, shared goal. Instead, we run both mechanisms in what we consider "standard" parametrizations. CONV does not have

any adjustable parameters anyway. For CAPSS, we adjust the allowed compensation, $c_{\max}$, and the compliance target, p_t.

We compare the mechanisms in the synthetic BASE topology depicted in Figure 5.4 and in the nine realistic topologies introduced in Section 5.1.4.1. Sections 5.4.2.1 and 5.4.2.2 consider the synthetic topology. In Section 5.4.2.1, we vary $c_{\max}$. The contract period, Δt_h, is one year. In Section 5.4.2.2, we vary the contract period, Δt_h, while $c_{\max} = 0.25 \cdot MRC$. Additionally, we vary the compliance target and the compensation policy in the synthetic topology. We selected those parameters for variation because they have the biggest impact when comparing CAPSS and CONV. In the synthetic topology, each connection requests a data rate of one abstract traffic unit, i.e., one traffic unit equals one connection. The average straight-line distance between any two node pairs is set to 500 km. This is achieved by scaling all link lengths in the BASE topology with a factor of 2.25.

Section 5.4.3 considers the realistic networks. Here, we use a mixed traffic scenario, i.e., connection requests with varying parameter values appear in a single simulation run. Connections request data rates of 1, 5, 10, or 50 Gbit/s and either the GOLD or SILVER policy. The contract period is 3, 6, 12, or 24 months. We only consider a compliance target of $p_t = 0.95$ in the realistic topologies.

In all scenarios, we set the MTTR of a link to 9 hours and the CC value to 628 km [Ver+05]. Both provisioning mechanisms can deploy path protection if necessary but not segment protection. Connection requests arrive following a Poisson process. As described in Section 5.1.4.5, the mean IAT is adjusted prior to every simulation run such that the rejection ratio is 0.01. In that way, the operating points are comparable across different provisioning mechanisms and parameter configurations. Furthermore, we dimension the link capacities based on the betweenness centrality, as described in Section 5.1.4.4. In the synthetic topology, the minimum and maximum link capacities are set to $\kappa_{\min} = 40$ and $\kappa_{\max} = 80$ connections. In the realistic topologies, we set $\kappa_{\min} = 5$ Tbit/s and $\kappa_{\max} = 10$ Tbit/s.

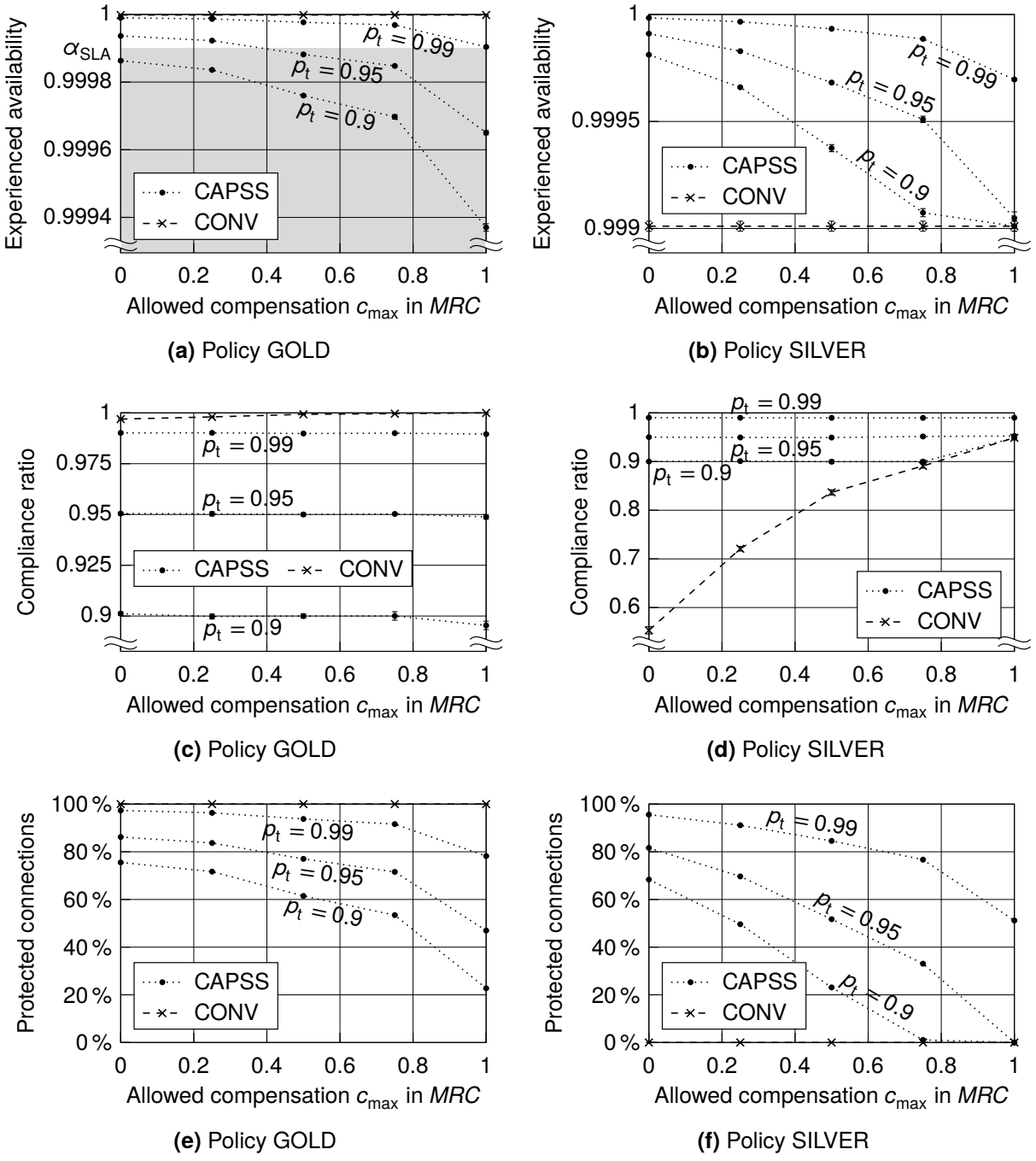
5.4.2 Results in the Synthetic Topology

The presentation of the results for the synthetic topology is split into two sections. Section 5.4.2.1 concentrates on the behavior for different levels of allowed compensation, different compliance targets, and the two compensation policies. Section 5.4.2.2 considers the impact of different contract periods.

In all figures, the left column shows simulations for the compensation policy GOLD, while the right column corresponds to the SILVER policy. The dotted curves correspond to CAPSS with different compliance targets, p_t. The dashed curve corresponds to CONV. Since CONV does not consider the compliance target for route selection, the results are insensitive to a variation of the compliance target.

5.4.2.1 Behavior for Different Levels of Allowed Compensation

Figure 5.12 shows the experienced availability per billing cycle, the compliance ratio, and the share of protected connections. Additionally, Figure 5.13 depicts the carried load. The x-axis shows the allowed compensation, $c_{\max}$. Regarding the experienced availability, the share of protected connections, and the carried load, CONV is insensitive to changes in the allowed compensation. The reason is that CONV does not incorporate the allowed compensation into the provisioning process, and thus, a variation has no impact on the routing decisions. In other words, CONV always routes connections the same way, independent of $c_{\max}$. In



(a) Policy GOLD **(b)** Policy SILVER

(c) Policy GOLD **(d)** Policy SILVER

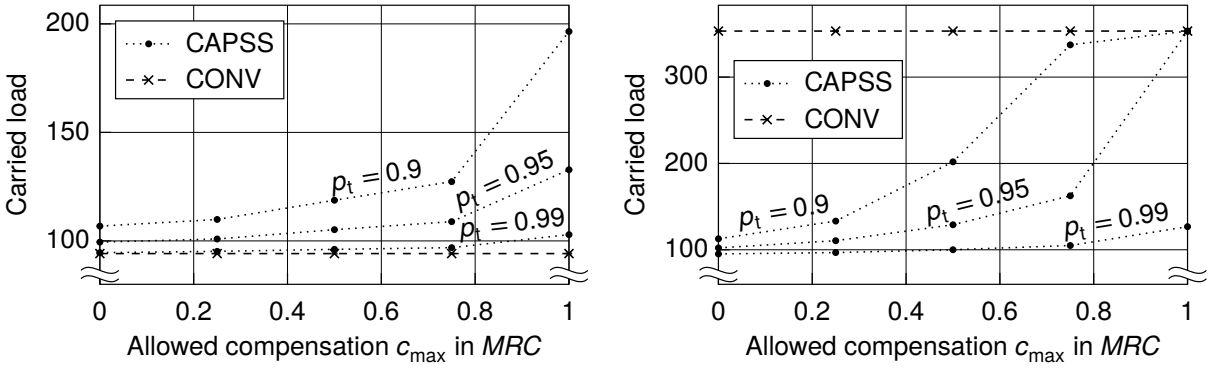
(e) Policy GOLD **(f)** Policy SILVER

Figure 5.12 Comparison of CAPSS and CONV in the synthetic topology with respect to the allowed compensation

contrast, the compliance ratio achieved with CONV changes when c_{max} changes because the compliance ratio is a function of c_{max} (see Section 5.1.3.4).

In the following, we will first consider the experienced availability and compare it with the compliance ratio. This step will confirm the differences between CONV and CAPSS we discussed in the previous chapters. Subsequently, we will consider the carried load, and we will see that no mechanism is universally better than the other in terms of network capacity.

Consider the experienced availability depicted in Figures 5.12a and 5.12b. The experienced availability is a key performance indicator for CONV because the goal of CONV



(a) Policy GOLD **(b)** Policy SILVER

Figure 5.13 Comparison of CAPSS and CONV in the synthetic topology with respect to the allowed compensation

is to provision connections such that their experienced availability fulfills the availability level defined in the SLA. The shaded area in Figure 5.12a describes availabilities worse than the availability level required in CONV. In the case of the GOLD policy, the required availability is $\alpha_{\text{SLA}} = 0.9999$. For the SILVER policy, it is $\alpha_{\text{SLA}} = 0.995$, which is not visible in Figure 5.12b. As can be seen, CONV overfulfills the availability requirement consistently. This has already been identified as a general shortcoming in Section 3.3. For CAPSS, the experienced availability increases when the compliance target, p_{t}, or the allowed compensation, $c_{\max}$, become stricter. The reason is that CAPSS maintains a constant compliance ratio. When p_{t} or $c_{\max}$ become stricter, CAPSS has to select more reliable routes to achieve the same compliance ratio. For the SILVER policy (Figure 5.12b), CAPSS provides even higher availabilities than CONV. In contrast, for the GOLD policy (Figure 5.12a), CAPSS provides lower availabilities than CONV. In the less strict parametrizations, i.e., low p_{t} and high $c_{\max}$, the availability even falls below α_{SLA} (into the shaded area).

The experienced availability considered here is an average value. The fact that CONV overfulfills the required availability level overall does not mean that every individual connection overfulfills its SLA in every billing cycle, either. In fact, the probability of SLA violation, i.e., the experienced availability is below the required availability in a billing cycle, is 0.02 % for the GOLD policy and 4.79 % for the SILVER policy (independent of $c_{\max}$). Especially for the SILVER policy, this probability is non-negligible, and even worse, it is not controllable in CONV. This issue has already been identified as another shortcoming of conventional provisioning in Section 3.3.

Next, we consider the compliance ratio depicted in Figures 5.12c and 5.12d. Contrary to the experienced availability, the compliance ratio is a key performance indicator for CAPSS. Figures 5.12c and 5.12d confirm that CAPSS provides a compliance ratio that matches the respective compliance target. There are two exceptions to this. First, for the GOLD policy (Figure 5.12c), $c_{\max} = 1 \cdot MRC$, and $p_{\text{t}} = 0.9$, the compliance ratio is lower than 0.9 due to the approximation errors identified in Section 5.2.1. Second, for the SILVER policy (Figure 5.12d), $c_{\max} = 1 \cdot MRC$, and $p_{\text{t}} = 0.9$, the compliance ratio exceeds 0.9 because all connections can be provisioned unprotected, as can be seen in Figure 5.12f. Hence, CAPSS has no real way of spending compliance surplus (routing on longer, less reliable paths is typically quite limited). However, overall, we see that CAPSS provides the desired compliance ratio. Contrary to this, CONV either under- or overfulfills the compliance targets. For the GOLD policy (Figure 5.12c), CONV consistently overfulfills even the strictest

compliance target of $p_t = 0.99$. For the SILVER policy (Figure 5.12d), it mostly underfulfills the compliance target, even though the experienced availability is far above the requirement of 0.995 (Figure 5.12b). This shows once again that matching the steady-state availability with the SLA availability does not automatically match the actual compliance probability with the targeted compliance probability and vice versa.

Now, we consider the resulting carried load. For the SILVER policy, depicted in Figure 5.13b, CAPSS reduces the network capacity. Figure 5.12d shows that CONV leads to compliance ratios that are much lower than those targeted by CAPSS. A lower compliance ratio results mainly from less protection. Less protection, in turn, requires less redundant network resources and, hence, allows more carried load. Therefore, the reduced network capacity with CAPSS is the price the network operator has to pay for achieving the compliance target. In contrast, CAPSS increases the network capacity for the GOLD policy (Figure 5.13a). Figure 5.12c shows that CONV provides compliance ratios that are higher than those targeted by CAPSS. That means CAPSS requires less reliable routes and, thus, increases the network capacity.

As a rule of thumb, if CONV *underfulfills* the compliance target, it provides *more* network capacity than CAPSS. If CONV *overfulfills* the compliance target, it provides *less* network capacity than CAPSS. In the presented scenario, when switching from CONV to CAPSS, the network capacity can be increased when the GOLD policy is used. However, for the SILVER policy, the network capacity decreases. It is important to note that the network capacity is only one performance indicator. CAPSS might not be able to increase the network capacity generally. However, CAPSS always fulfills the desired compliance target and, in that way, provides a general advantage over conventional provisioning.

To understand better why CONV overfulfills the compliance target for the GOLD policy but underfulfills it for the SILVER policy, we consider two main points. First, consider Figures 5.12e and 5.12f again. For the SILVER policy, CONV routes all connections *without* protection. For the GOLD policy, all connections are protected. Path protection increases both the steady-state availability and the compliance probability of a route substantially. This has already been discussed in Section 3.3.3. In the present scenario, the protection increases the compliance probability so much that it overfulfills the compliance target significantly.

A second important difference lies in the compensation policies themselves. Both policies have the same seven compensation levels, from 0 % to 100 % of the MRC. However, the corresponding unavailabilities are different. The first two compensation steps in the GOLD policy are at the unavailabilities 0.0001 and 0.001, with a factor of 10 in between. For the SILVER policy, the first two steps are at 0.005 and 0.01 and, hence, only separated by a factor of 2. That means the SILVER policy generally allows higher unavailabilities, but the rise in compensation is steeper. From a steady-state availability point of view, the SILVER policy allows unprotected connections between all node pairs in the present scenario. However, from a compliance probability point of view — which takes the whole policy into account — protection is required in order to cope with the steep compensation increase in the SILVER policy. For the GOLD policy, the situation is "inverted." The unavailability at the first compensation step is strict, but the subsequent rise in compensation is less steep than in the SILVER policy. Accordingly, CONV operates stricter than CAPSS for the GOLD policy and vice versa for the SILVER policy. This is a second reason why CONV underfulfills the compliance target with the SILVER policy but overfulfills it with the GOLD policy.

The results discussed above have been generated with an average node pair distance of 500 km. The simulations have been repeated for average node pair distances of 100 km and 700 km. The lower distance of 100 km leads to higher steady-state link availabilities, while

the higher distance of 700 km leads to lower ones. The corresponding results confirm the fundamental conclusions we have drawn so far.

5.4.2.2 Behavior for Different Contract Periods

Another important factor that leads to differences between CONV and CAPSS is the contract period. CONV does not consider the contract period. However, for a connection with otherwise fixed parameters, a longer contract period statistically leads to a higher total amount of compensation because more billing cycles are aggregated. CAPSS takes this into account with the result that connections with longer contract periods are provisioned on more reliable routes.

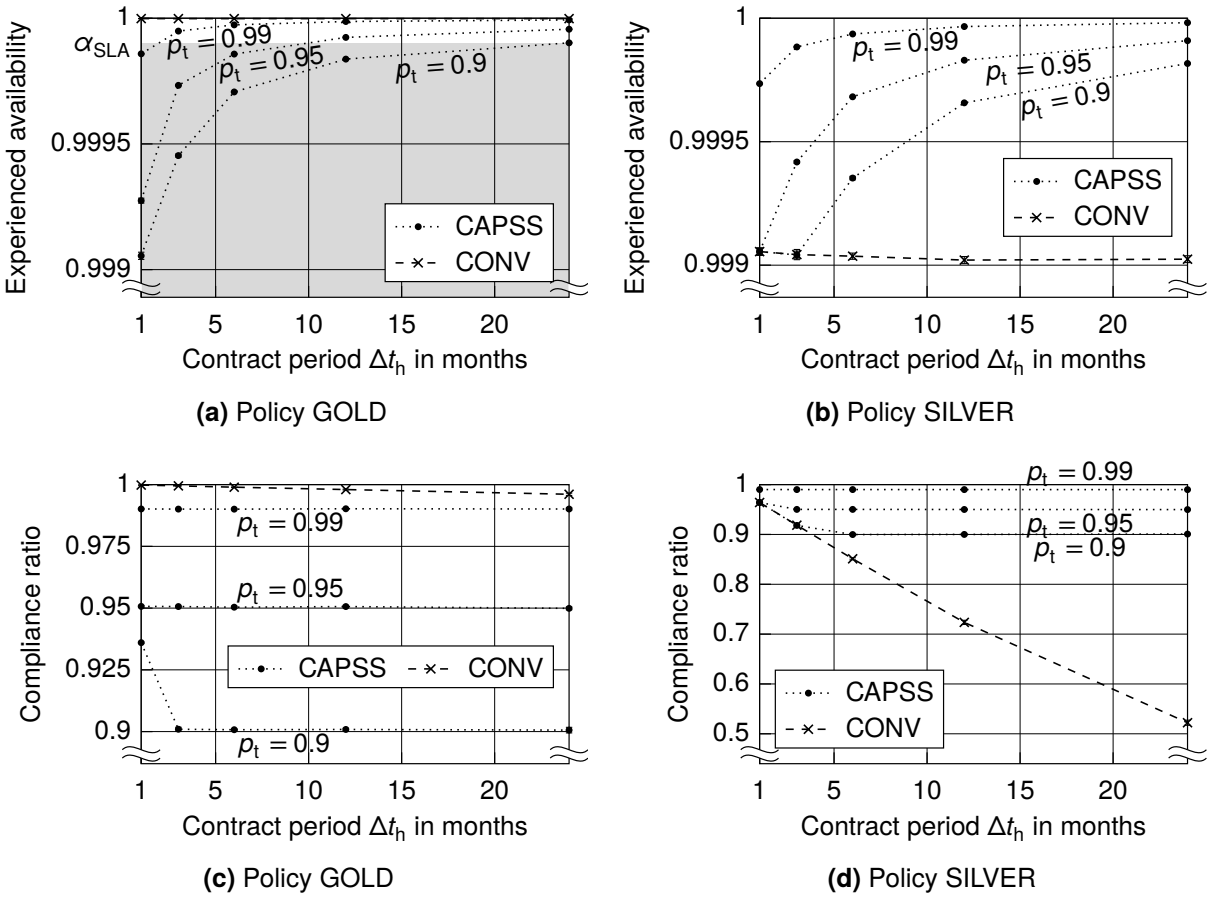


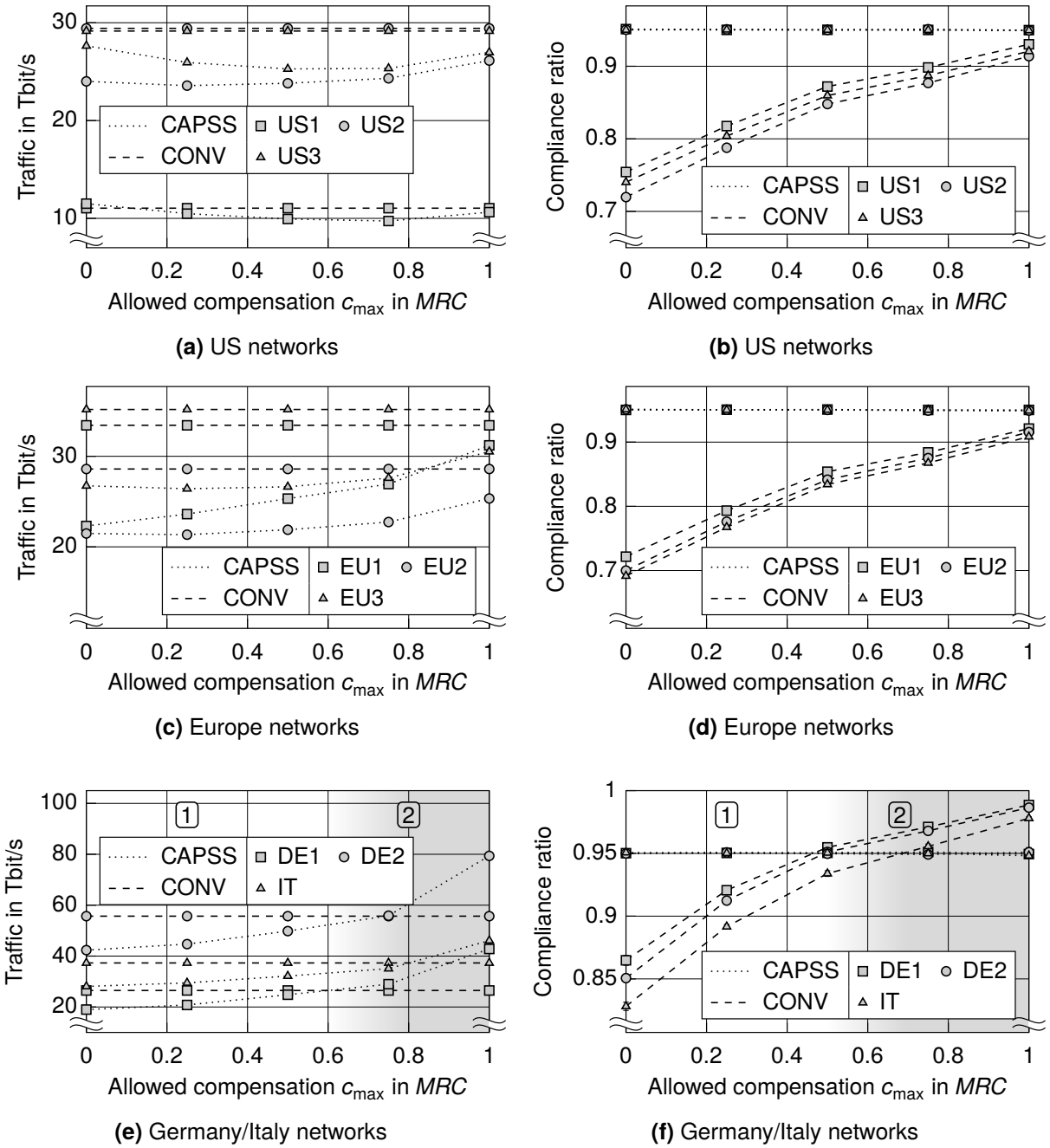
Figure 5.14 Comparison of CAPSS and CONV in the synthetic topology with respect to the contract period

Figure 5.14 shows simulation results for a varying contract period, Δt_h. The allowed compensation is set to $c_{max} = 0.25 \cdot MRC$. In Figures 5.14a and 5.14b, it can be seen that the experienced availability increases with the contract period for CAPSS. As mentioned above, this indicates that CAPSS selects more reliable routes. Figures 5.14c and 5.14d show that CAPSS provides the desired compliance ratios (some exceptions exist for the same reason as discussed in Section 5.4.2.1). For CONV, the compliance ratio decreases with an increasing contract period. Especially for the SILVER policy, the compliance ratio falls substantially, close to only 0.5 for a two-year contract. That means if CONV is used to provision two-year connections with the SILVER policy, almost half of them will aggregate more than $0.25 \cdot MRC$

in compensation over their contract period. Only for very short contract periods of 1 and 3 months, CONV achieves a compliance ratio of 0.9. To achieve a compliance ratio of 0.95, the contract period must not exceed one month.

5.4.3 Results in Realistic Topologies

In this section, we compare CONV and CAPSS in the realistic scenarios introduced in Section 5.1.4.1. Figures 5.15 and 5.16 show the corresponding results. Each figure shows simulation results for both CAPSS and CONV in a group of networks, namely the US networks, the Europe networks, and the Germany and Italy networks.



(a) US networks

(b) US networks

(c) Europe networks

(d) Europe networks

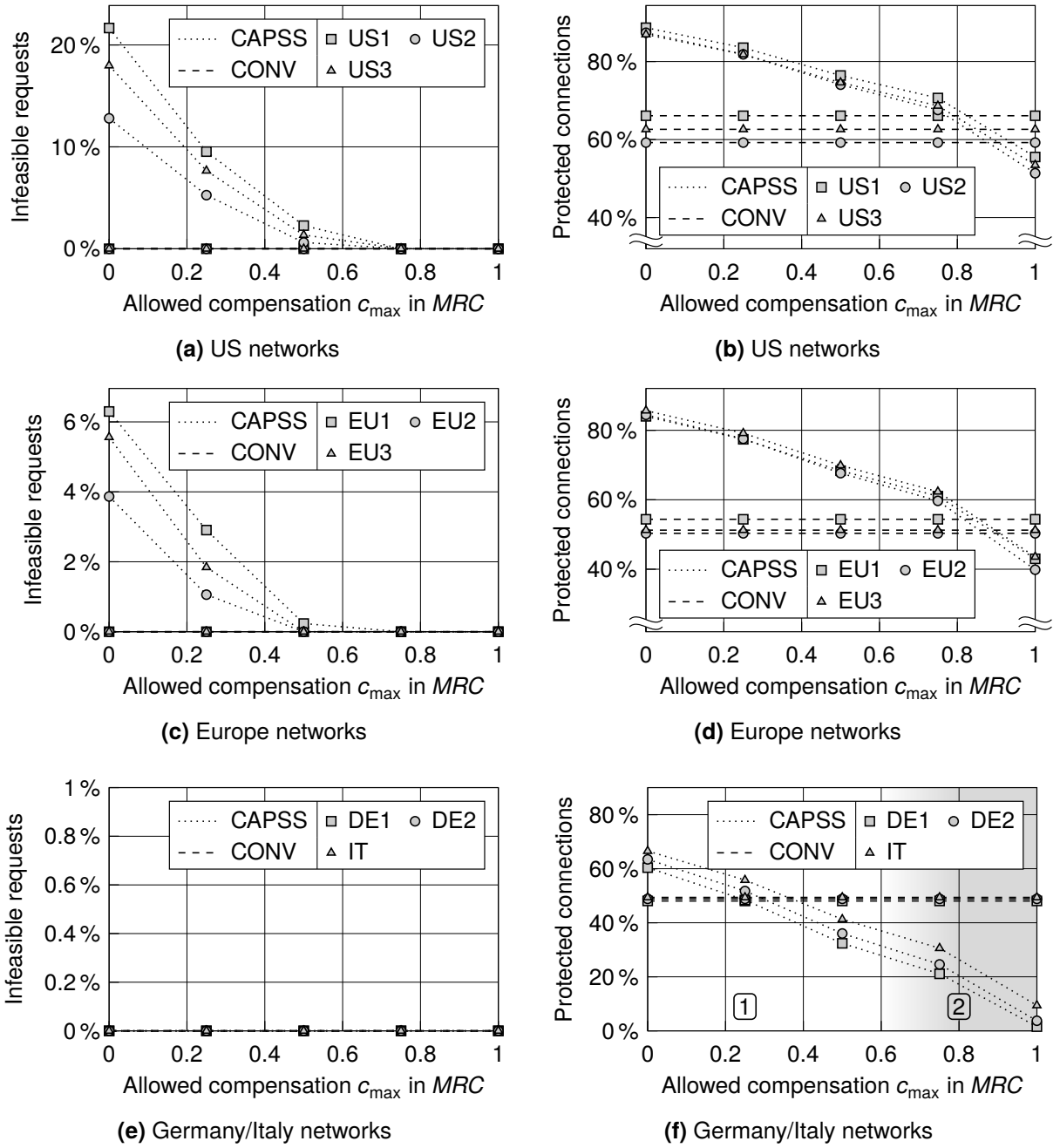
(e) Germany/Italy networks

(f) Germany/Italy networks

Figure 5.15 Comparison of CAPSS and CONV in the realistic topologies

Considering the achieved compliance ratios in the right column of Figure 5.15, it can be seen that CONV consistently underfulfills the compliance target in the US and Europe topologies. Only for the smaller topologies of Germany and Italy, and if the allowed compensation is high (Figure 5.15f, area 2), CONV also overfulfills the compliance targets.

Like in the synthetic topology, compliance over- and underfulfillment is tightly coupled with capacity increase and decrease. For the topologies of Germany and Italy and high allowed compensation (Figure 5.15e, area 2), CAPSS increases the network capacity compared with CONV. In the other cases, the capacity decreases since CAPSS provisions connections with more or a similar amount of protection resources on average. This can be seen in the right columns of Figure 5.16. For the Germany and Italy topologies, CAPSS can reduce the share of protected connections substantially compared with CONV (Figure 5.16f, area 2).



(a) US networks

(b) US networks

(c) Europe networks

(d) Europe networks

(e) Germany/Italy networks

(f) Germany/Italy networks

Figure 5.16 Comparison of CAPSS and CONV in the realistic topologies

Another significant difference between CONV and CAPSS is observable in the left column of Figure 5.16, which shows the share of infeasible requests. Connection requests are considered infeasible if there is no route that provides the required figure of merit independent of the link capacities. That means the operator cannot offer connections with this or an even stricter SLA between the respective nodes. CAPSS, often operating more strictly than CONV, finds infeasible node pairs when CONV can still provision connections for all node pairs. This is visible in Figure 5.16c and even more so in Figure 5.16a. The lower the allowed compensation, $c_{\max}$, the more infeasible node pairs CAPSS finds. Like before, the Germany and Italy topologies show a different behavior (Figure 5.16e). Here, no infeasible node pairs exist because the topologies have higher link availabilities due to their smaller spatial extent.[5] Of course, infeasible node pairs can be avoided, e.g., by augmenting link availabilities or by deploying even more protection resources. However, such measures are additional cost factors.

5.4.4 Summary

In Section 5.4, we compared CAPSS with CONV. The two mechanisms have fundamentally different goals: CAPSS tries to control the incurred compensation and shares compliance surplus, while CONV "only" tries to fulfill SLA availabilities.

The presented simulation results show that the behavior of CAPSS differs significantly from CONV in many respects. While CAPSS provides consistent compliance ratios that match the operator-defined compliance target, CONV has no means of controlling compliance under- or overfulfillment. In particular, for the GOLD policy, CONV overfulfills the compliance target, while for the SILVER policy, it underfulfills it substantially.

In the realistic network scenarios with mixed traffic, CAPSS often has to deploy more reliable routes than CONV in order to fulfill the compliance target. As a result, if the compensation policy or the operator-defined parameters are strict, CAPSS finds infeasible requests when CONV can still provision every request. This is an important effect the network operator has to consider carefully from different perspectives. First, if the operator simply replaces CONV with CAPSS, some node pairs can no longer be connected. Second, and much more importantly, the fact that infeasible requests exist for which CONV still finds routes shows that CONV exposes the operator to an unquantifiable risk of SLA violation.

As far as the resulting network capacity is concerned, CONV is inferior to CAPSS for the GOLD policy but superior for the SILVER policy. Neither of the two mechanisms surpasses the other in each and every scenario. Operators might be tempted to opt for CAPSS in those scenarios in which capacity increases can be expected. However, the choice should be based primarily on whether the operator wants to achieve a specific compliance probability with respect to the SLA compensation or not. That means the choice between a conventional provisioning mechanism and a compensation-aware one like CAPSS is a fundamental choice and should be made independently of the effects on the network capacity.

The results of this study provide an understanding of the differences between CONV and CAPSS. They also show that a direct comparison with respect to a single metric like the network capacity is not reasonable because both mechanisms have fundamentally different goals. For a meaningful comparison, the experienced availability and the compliance ratio are vital metrics as well.

[5] The presence of infeasible requests has an additional effect in our studies. If the share of infeasible requests exceeds a certain level, the network capacity may grow. More details can be found in Section 5.5.2.

Since we consider the choice between a conventional availability-based and a compensation-aware mechanism as a fundamental decision, we use CAP, the compensation-aware provisioning mechanism *without* surplus sharing (Section 4.1.1), as reference mechanism in the remaining sections. That means we assume the network operator is already using compensation-aware provisioning. We then focus on the effects CAPSS creates by adding surplus sharing to CAP.

5.5 Compensation-Aware Provisioning with and without Surplus Sharing

Section 5.4 compared CAPSS with a conventional provisioning mechanism, which is based on the steady-state availability. We argued that the decision to incorporate the compensation directly into the provisioning process is a fundamental decision for the network operator. The findings in Section 5.4 help guide this decision-making process.

In this section, we assume the network operator already employs CAP as a compensation-aware provisioning strategy. We focus on the additional benefits surplus sharing provides. Consequently, we compare CAPSS and CAP in the following studies.

We first have a detailed look into the behavior in different network load situations (Section 5.5.1). Subsequently, we study the impact of the spatial network extent in Section 5.5.2. Since the length of a link is related to its reliability, this study can also be interpreted with respect to the network reliability. Section 5.5.3 concentrates on the achievable capacity increase and extends the evaluation scenario to a large number of topologies and parameters. Section 5.5.4 confirms our previous findings in the realistic networks and provides some additional insights. Lastly, Section 5.5.5 summarizes the results.

5.5.1 Behavior in Different Load Situations

In Section 5.3, we identified two effects CAPSS has on the network, namely reduced protection requirements and increased path diversity. Both effects increase the network capacity when the load offered to the network is high. In this section, we study the behavior of CAPSS across a wide range of offered loads to identify network situations in which CAPSS is advantageous. In particular, we also consider low offered loads.

5.5.1.1 Simulation Setup and Evaluation Parameters

We use the synthetic BASE topology depicted in Figure 5.4 again. This time, we set the average straight-line distance between any two node pairs to 200 km by scaling all link lengths appropriately. We assume that each connection requests a data rate of one abstract traffic unit, i.e., one traffic unit equals one connection. The link capacities are dimensioned based on the betweenness centrality, as described in Section 5.1.4.4, with a minimum and maximum link capacity of $\kappa_{\min} = 40$ and $\kappa_{\max} = 80$ connections. We set the MTTR of a link to 9 hours and the CC value to 628 km [Ver+05].

Connection requests are generated randomly with exponentially distributed IATs. Each connection has a contract period of $\Delta t_{\mathrm{h}} = 1\,\mathrm{year}$. The SLA stipulates the GOLD compensation policy. Further, we assume that the operator targets $p_{\mathrm{t}} = 0.95$ for an allowed compensation of $c_{\max} = 0.25 \cdot MRC$. Both provisioning mechanisms can deploy path protec-

tion if necessary but not segment protection. Let l denote the load offered to the network, and let $\overline{IAT}$ denote the mean IAT. The offered load is the ratio of the contract period and the mean IAT, i.e.,

$$l = \frac{\Delta t_{\mathrm{h}}}{\overline{IAT}} . \tag{5.17}$$

To study the behavior of CAPSS in different load situations, we run several simulations, each with a different mean IAT. In order to provide CIs, each simulation consists of 10 batches.

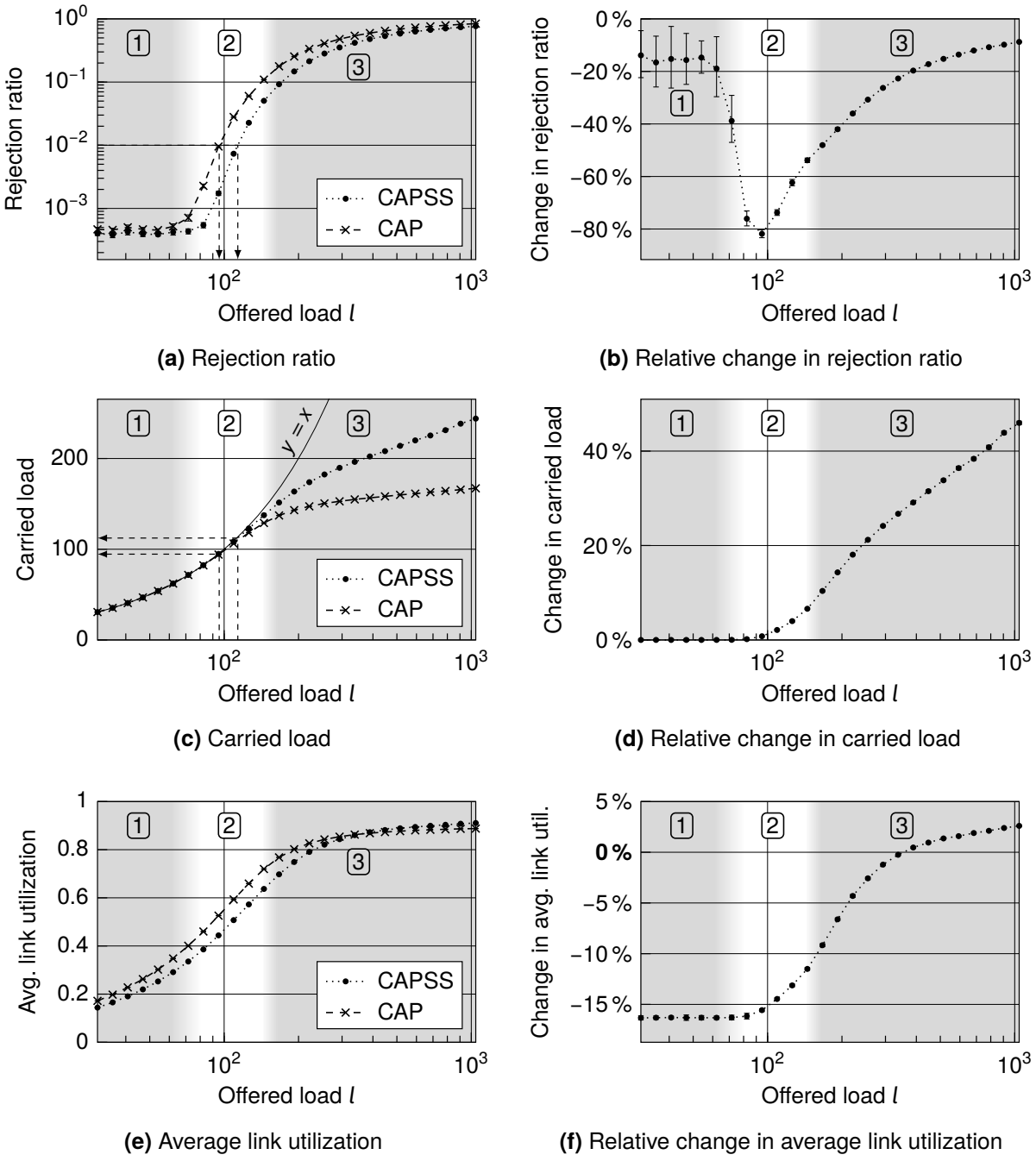
5.5.1.2 Results

Figure 5.17 shows the results of the simulation, namely the request rejection ratio, the average link utilization, and the carried load. The left column shows values for CAPSS and CAP. The right column shows the relative difference between the two mechanisms. All figures show CIs with a 95 % confidence level. However, for most simulations, the intervals are too small to be visible. The CIs are computed as described in Section 5.1.3.6. In particular, for the columns on the right, we employ the MOVER-R method. In each figure, the shading with smooth transitions indicates three ranges of offered load characterized by different magnitudes of request rejection ratios.

Figure 5.17a shows the request rejection ratio. For low offered loads in area 1, the rejection ratio stagnates at around $4 \cdot 10^{-4}$. That means request rejection cannot be completely avoided in this scenario. Remember that CAPSS only provisions a connection on a working route. If a link failure is present in the network and makes a potential route infeasible, CAPSS has to resort to an alternative route. However, some node pairs cannot be connected on alternative routes because their compliance probability would be insufficient. Consequently, connection requests for those node pairs have to be rejected, even though the network is only lightly loaded. This is the reason for the residual rejection ratio at low offered loads. In fact, with less strict parameters, e.g., $c_{\max} = 1 \cdot MRC$, CAPSS avoids rejections completely at low loads. If, additionally, the SILVER policy is used, CAP manages to accommodate all requests as well. That means the residual rejection ratio is not a fundamental limitation of our provisioning mechanisms but is a result of the choice of parameters in this study.

As can be seen in area 1 of Figure 5.17b, CAPSS lowers this residual rejection ratio by around 15 %. While this improvement is considerable in its relative value, it is negligible in absolute terms because a rejection ratio on the order of 10^{-4} is barely noticeable anyway. The advantage of surplus sharing becomes apparent in area 2, around an offered load of 100. Here, the rejection ratios are on the order of 0.01 to 0.1, i.e., between 1 % and 10 %. According to [Zho+16; Mus+19], this range is relevant for network operation. We denote this coarse range of rejection ratios as *medium-load*. Accordingly, we consider rejection ratios much smaller than 0.01 as *low-load* and ratios much higher than 0.1 as *high-load*. CAPSS reduces the rejection ratio by up to 80 % in this range. For even higher offered loads in area 3, CAPSS still provides significant reductions. However, rejection ratios above 10 % do not represent a desirable operating range for most operators.

Figures 5.17c and 5.17d show the behavior in terms of the carried load. Both mechanisms can handle offered loads of up to 100 without problems. This observation is in line with the low rejection ratios at low offered loads. As soon as the offered load exceeds 100, both mechanisms start having problems accommodating all connections. However, in area 3, CAPSS can accommodate many more connections than CAP. In Figure 5.17d, it can be seen that the advantage of CAPSS grows with the growing offered load. However, remember that



(a) Rejection ratio

(b) Relative change in rejection ratio

(c) Carried load

(d) Relative change in carried load

(e) Average link utilization

(f) Relative change in average link utilization

Figure 5.17 Comparison of CAPSS and CAP in different load situations – The shading indicates different magnitudes of request rejection ratios.

for offered loads above 100, we quickly leave the reasonable operating range due to high rejection ratios. The load increase finds a maximum of around 65 % for an offered load of 10^4 (not visible in Figure 5.17d).

Figures 5.17e and 5.17f show the average link utilization and the corresponding relative change CAPSS achieves. It can be seen that CAPSS reduces the link utilization up to an offered load of around 380 in area 3. Only for higher offered loads, the utilization is higher than with CAP. For low offered loads, reductions of more than 15 % are achieved. This shows the main advantage of CAPSS in low-load situations like area 1: It reduces the link utilization because it relaxes protection requirements. Since network operators often upgrade

the link capacity when a certain utilization is reached [Has+13], CAPSS helps postpone such upgrades. Considering the packet-oriented network layers, low link utilization prevents excessive queuing delays [Cho+07a].

All of these improvements come at the price of increased compensation. While the average compensation per connection under CAP stays below $0.001 \cdot MRC$, it reaches a maximum of $0.05 \cdot MRC$ and a mean value of $0.04 \cdot MRC$ with CAPSS. Of course, this is a significant increase. Nevertheless, the operator's compliance target of $p_t = 0.95$ for an allowed compensation of $c_{max} = 0.25 \cdot MRC$ is fulfilled.

Apart from the present study, most studies in this thesis consider the network capacity as the main performance indicator. The network capacity is the carried load at a request rejection ratio of 0.01. The network capacities for CAPSS and CAP can be derived from Figures 5.17a and 5.17c in this study. The dashed arrows in Figure 5.17a show the offered load that leads to a rejection ratio of 0.01. For CAPSS, the offered load is 112, while for CAP, it is 95. Both values are linearly interpolated because none of the simulated offered loads result in an exact rejection ratio of 0.01. The dashed arrows in Figure 5.17c lead to the *carried* loads that correspond to the identified *offered* loads. For CAPSS, the corresponding carried load is 111, while for CAP, it is 94 (again, those values are linearly interpolated). Consequently, CAPSS increases the network capacity by around $100\,\% \cdot (111/94 - 1) \approx 18\,\%$ at a rejection ratio of 0.01 in this study.

Other studies in this thesis use an iterative simulation procedure to calibrate the rejection ratio by adjusting the mean IAT of connection requests. As explained in Section 5.1.4.5, this procedure cannot guarantee to match a rejection ratio of 0.01 exactly. We target an accuracy of 0.5 %, i.e., the final rejection ratios range from 0.00995 to 0.01005. From the results of this study, we can see that such small deviations in the rejection ratio do not change the carried load significantly. In Figure 5.17a, a deviation in the rejection ratio of 0.5 % around a base ratio of 0.01 leads to a deviation in the *offered load* of less than 0.05 %, both for CAPSS and CAP. Also, for a rejection ratio of 0.01, the *carried load* in Figure 5.17c is almost the same as the *offered load*. Therefore, also the carried load only deviates by around 0.05 %. Consequently, when comparing the network capacities under CAPSS and CAP, differences around 0.05 % might be caused by inaccuracies in the operating points. However, larger differences suggest fundamental differences between the two mechanisms. As shown above, the increase in carried load of 18 % is orders of magnitude higher than 0.05 %, and hence, the small uncertainty in the accuracy of the rejection ratio is irrelevant.

5.5.1.3 Summary

This study evaluated CAPSS and CAP over a broad range of offered loads. The scenario is certainly too small to claim the general validity of the exact numerical values. Nevertheless, we have identified the fundamental advantages of CAPSS in different load situations. First, CAPSS reduces the link utilization by more than 10 % in low- and medium-load situations. Since network operators often upgrade the link capacity when a certain utilization is reached [Has+13], CAPSS helps postpone such upgrades. Additionally, low link utilization prevents excessive queuing delays in packet-oriented network layers [Cho+07a]. Second, CAPSS reduces the rejection ratio and increases the carried load substantially. The network capacity increase in the presented scenario amounts to 18 % at a rejection ratio of 0.01. The price for these improvements is higher compensation. However, we emphasize that the compliance probability matches the operator's target. Hence, the incurred compensation is accepted by the operator and is supposed to match its business models.

The study provides an understanding of CAPSS's behavior in different load situations. The most significant effect is the capacity increase in medium network loads. Therefore, the following studies focus on the achievable increase in network capacity at a rejection ratio of 0.01.

5.5.2 Behavior in Different Network Extents

As a continuation of Section 5.5.1, we now study the significance of the spatial network extent on the achievable increase in network capacity. We consider a single topology first and scale it to different sizes by uniformly scaling its links' lengths. In the second step, we evaluate different instances of this topology to improve the confidence in the results.

We model a link's MTTF as a function of its length, namely

$$MTTF_e = \frac{CC \cdot 360\,\text{days} \cdot 24\,\text{h/day}}{\ell_e}. \tag{5.18}$$

Therefore, a longer link has a shorter MTTF. Since we assign the same MTTR to all links, a longer link also has a lower steady-state availability. Consequently, the study can also be interpreted as an evaluation of different network availability levels.

The study is comparable to the illustrative three-node example in Section 5.3. There, we varied the CC value, which affects the MTTF similarly to a variation of the link length.

5.5.2.1 Simulation Setup and Evaluation Parameters

The simulation setup equals the one in Section 5.5.1 almost entirely. The only difference is that the study in Section 5.5.1 considered the effects of CAPSS for different network loads. In this study, we adjust the offered load prior to each simulation such that the resulting rejection ratio is 0.01. The procedure is described in Section 5.1.4.5. In this way, we are able to estimate the achievable capacity increase for a realistic operating point.

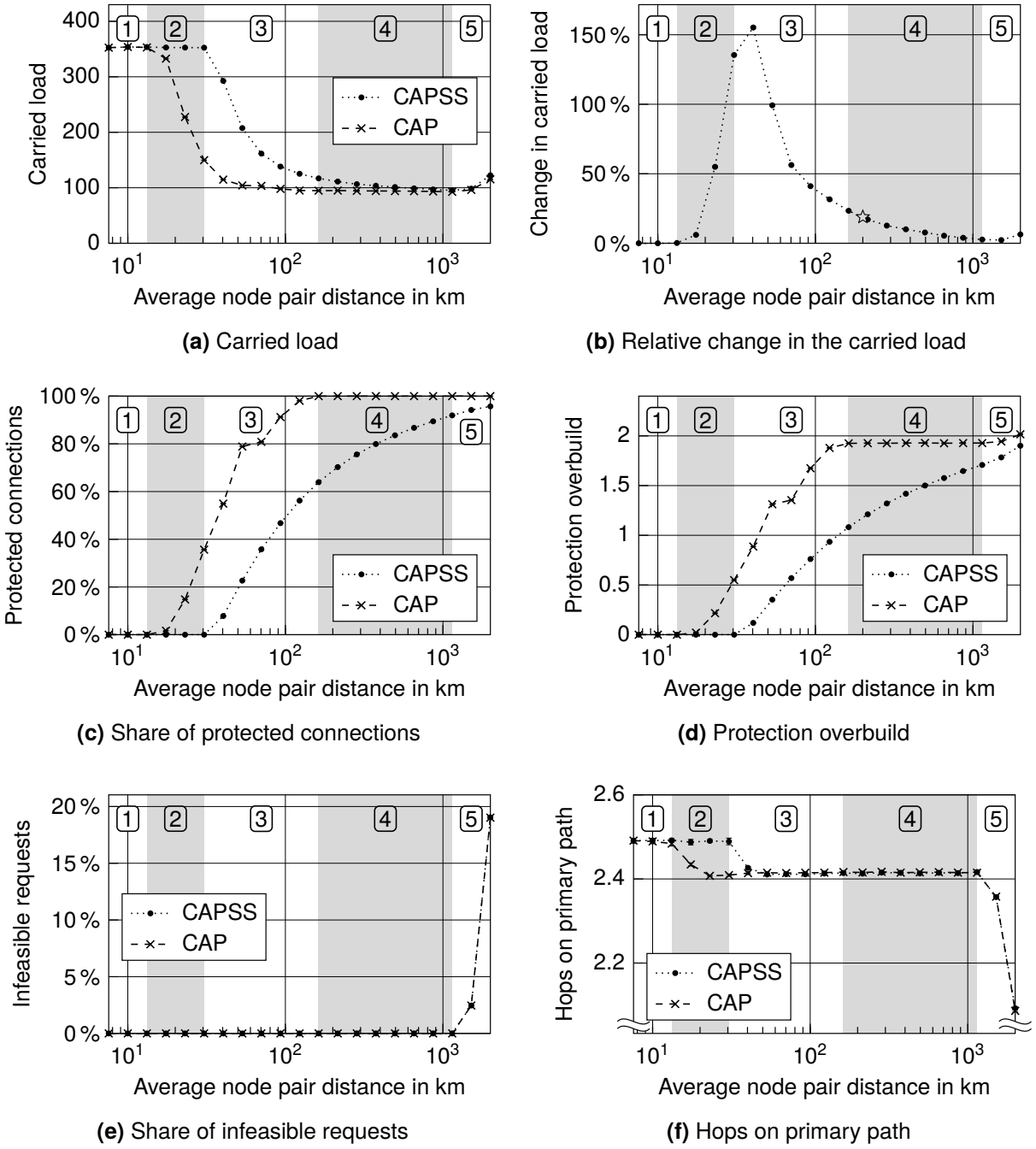
The parameter we vary is the average straight-line distance between any two node pairs, i.e., the spatial extent of the network topology. We vary this average distance between 7.6 km and 2000 km. In this way, we cover metro- and core-sized networks. As in Section 5.5.1, we set the general link MTTR to 9 hours and $CC = 628$ km [Ver+05].

Since the MTTF of a link is a function of its length and CC, a variation in the average node pair distance can also be considered a variation in the network's availability level. For example, assume that one unprotected connection exists between each node pair. A connection is routed along the most reliable path [Zha+03b], and there are no limiting link capacities. Then, the average steady-state availability over all connections is 0.999986 for the minimum average node pair distance of 7.6 km. For the maximum node pair distance of 2000 km, it is 0.996.

5.5.2.2 Results for a Single Topology

Figure 5.18 shows various statistics measured during the simulations. In particular, Figure 5.18a shows the carried load, while Figure 5.18b shows the *relative change* in carried load when comparing CAPSS with CAP. The behavior of the carried load can be explained with the help of the remaining four figures. They show the share of protected connections, the resource overbuild required for protection, the share of infeasible requests, and the number of hops on the primary path. Based on [Li+02], we define the overbuild as the number of

links in the backup path divided by the number of links in the primary path. The overbuild of an unprotected route is zero. The x-axis shows the average node pair distance in all figures. A growing distance results in a decreasing network availability. The 95 % CIs are too small to be visible in most of the data points.



(a) Carried load

(b) Relative change in the carried load

(c) Share of protected connections

(d) Protection overbuild

(e) Share of infeasible requests

(f) Hops on primary path

Figure 5.18 Comparison of CAPSS and CAP in different network extents – The shading indicates areas with different shares of protected connections or infeasible requests.

The figures are subdivided into five areas. In area 1, the network availability is so high that even CAP does not need to deploy protection (see Figure 5.18c). In area 2, CAP deploys protection for some connections, while CAPSS still does not require any protection at all. In area 3, both mechanisms deploy protection. However, only in area 4, CAP has to protect *all*

connections. Area 5 is characterized by a non-zero infeasibility ratio (see Figure 5.18e), i.e., at least one node pair can no longer be connected because the SLA requirement is too strict.

CAP Consider now the carried load depicted in Figure 5.18a only for CAP. We discuss the figure starting in area 4 and moving to the left toward area 1 because the observable effects accumulate in this direction. The load is constant throughout area 4 because all connections have to be protected, and also, the resource overbuild is constant. As can be seen in Figures 5.18c and 5.18d, the share of connections that need protection and the overbuild drop substantially in area 3. Since a reduction in overbuild frees resources, more connections can be provisioned, and the carried load increases. The strongest load increase can be observed in area 2. Here, two effects are superimposed. First, the overbuild further decreases to 0, i.e., all connections are provisioned without protection. Second, the primary path length increases, as can be seen in Figure 5.18f. That means connections are increasingly provisioned on alternative, longer routes, too. In that way, unused capacity on less utilized links is incorporated. In area 1, the carried load stays constant.

In area 5, the carried load increases as well. The reason is that some node pairs become infeasible to connect and, hence, capacity can be used for other connections. The first node pairs to become infeasible are the most distant ones. For a *single* long-distance connection, *multiple* short-distance connections along the way can be provisioned. This leads to a drop in the average length of the primary paths, as can be seen in Figure 5.18f. Furthermore, the carried load increases.

CAPSS For CAPSS, the fundamental behavior is the same. However, due to surplus sharing, the carried load increases slightly in area 4 and considerably in area 3. In area 2, the carried load has already reached its maximum because no protection is needed, and longer routes are used as well.

Except for area 1, in which both mechanisms behave equally, CAPSS increases the carried load compared with CAP. That also means CAPSS increases the network capacity. The relative change in the carried load is visualized in Figure 5.18b. With a value of 2.3 %, the increase is minimal in area 5, i.e., for very large and, hence, unreliable topologies in which only little compliance surplus arises. The maximum of 155 % is found in area 3, at an average node pair distance of 40 km. Here, CAPSS uses longer routes and less protection than CAP. It becomes clear that the effectiveness of CAPSS depends very much on the network's extent or, equivalently, its reliability parameters.

As mentioned before, the capacity increase is made possible by the relaxation of protection requirements and the diversification of routes. While the diversification plays an important role only in very small, highly reliable topologies (mainly area 2), the relaxation of protection causes capacity increases for a broad range of topology extents or reliability levels, respectively (areas 3 and 4).

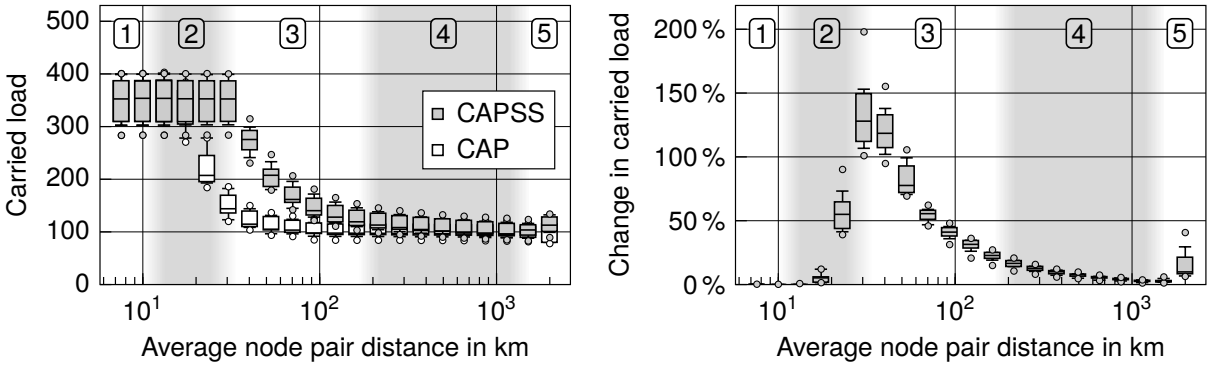
In Section 5.5.1, we evaluated a very similar scenario for an average node pair distance of 200 km. We found that the increase in carried load at a rejection ratio of 0.01 amounts to 18 %. As can be seen in Figure 5.18b at the star mark, the load increase at 200 km in this study is around 19 % after linear interpolation. This confirms the result in Section 5.5.1.

5.5.2.3 Results for Multiple Topologies

In this section, we extend the previous results to more network topologies in order to confirm the general validity of the results. As explained in Section 5.1.4.2, we have synthetically generated ten topology instances per topology parametrization. In the previous section, we

employed the BASE topology, i.e., the first instance with an average node degree of 2.8, and a diameter of 5 hops. In this section, we add results for the remaining nine instances. They are still biconnected, have a degree of 2.8, and a diameter of 5 hops, but they have different links and node locations.

Figure 5.19 shows box plots containing all ten topology instances. Figure 5.19a shows the carried load for CAPSS and CAP. Figure 5.19b shows the relative increase in carried load, i.e., the increase in network capacity.



(a) Carried load **(b)** Relative change in the carried load

Figure 5.19 Comparison of CAPSS and CAP in different network extents and ten synthetic topology instances – The shading indicates areas with different shares of protected connections or infeasible requests.

The areas 1 to 5 identified in the results of the single topology apply here too. However, since the box plots overlay the results of various topologies, the borders of the different areas blur.

The results confirm the fundamental findings we have made so far. For very small distances in area 1, no capacity increase is possible in any of the topologies. At the transition from area 2 to area 3, at around 30 km distance, the capacity increase reaches the highest values with a first and third quartile of 113 % and 149 %. This demonstrates the effectiveness of CAPSS in small topologies once more. Toward area 5, the capacity gains drop again as expected. The first and third quartiles at 283 km distance are 12 % and 14 %. At 1145 km, they are 2.5 % and 3.3 %. For even higher average node pair distances, the capacity increase becomes stronger again, but only because some node pairs are infeasible at such high distances.

5.5.2.4 Summary

In this section, we evaluated CAPSS's behavior in differently scaled topologies. We fixed the MTTR at 9 hours and the CC value at 628 km. However, since the MTTF of a link depends on its length, the variation of the topology size can also be interpreted as a variation of the links' MTTFs or the CC value. A doubling of the average node pair distance corresponds to a halving of the CC value. In that way, a large range of network scenarios is covered by this study.

The results reveal the two main effects of surplus sharing, namely lowered protection overbuild and increased path diversity. For large and medium-sized topologies down to an average node pair distance of around 70 km, only overbuild reduction is observable. The resulting capacity gains reach up to 50 % in medium-sized topologies. In large topologies,

the gains are much lower. In small topologies with an average node pair distance down to 10 km, also the path diversity is increased. The capacity increase is maximized at around 30 km with a median of 132 %. That means the network capacity at a rejection ratio of 0.01 is more than doubled. In even smaller topologies, neither of the two effects comes into play, and no capacity gains are present.

5.5.3 Capacity Increase for Other Important Parameters

The previous sections focused on the behavior of CAPSS with respect to specific parameters, namely the network load and the network extent. In this section, we evaluate the impact of a broader range of parameters following the procedure in Section 5.5.2.3. For each parameter configuration, we evaluate ten randomly generated topology instances in order to achieve statistical confidence. While the previous studies explained the causes underlying the observed effects in much detail and with the help of various recorded statistics, the following study mainly concentrates on the achievable capacity gains. The underlying causes are the ones identified in the previous studies.

5.5.3.1 Simulation Setup and Evaluation Parameters

We group the parameters into *topology parameters*, *operator-defined parameters*, and *contractual parameters*. For the topology parameters, we evaluate different average node degrees and diameters. Operator-defined parameters are parameters for provisioning which are defined by the network operator. Here, we consider the compliance target, the protection mechanism, and the allowed compensation. Lastly, we consider contractual parameters, namely the compensation policy and the contract period.

Table 5.3 Simulated parameter values – The highlighted values are the *default values*.

Parameter	Values	Unit/base quantity
Average node degree	2.4, **2.8**, 3.2, 3.6, 4, 4.4	–
Diameter	4, **5**, 6, 7	Hops
Compliance target	0.9, **0.95**, 0.99	–
Protection	**Path**, segment	–
Allowed compensation	0, **0.25**, 0.5, 0.75, 1	*MRC*
Compensation policy	**GOLD**, SILVER	–
Contract period	1, 3, 6, **12**, 24	Months

Table 5.3 shows the simulated parameter values. The highlighted values are the *default values*. That means while the values of one parameter are varied, the remaining parameters' values are kept constant at their default values. This reduces the influence of other parameters and allows pinpointing specific behavior.

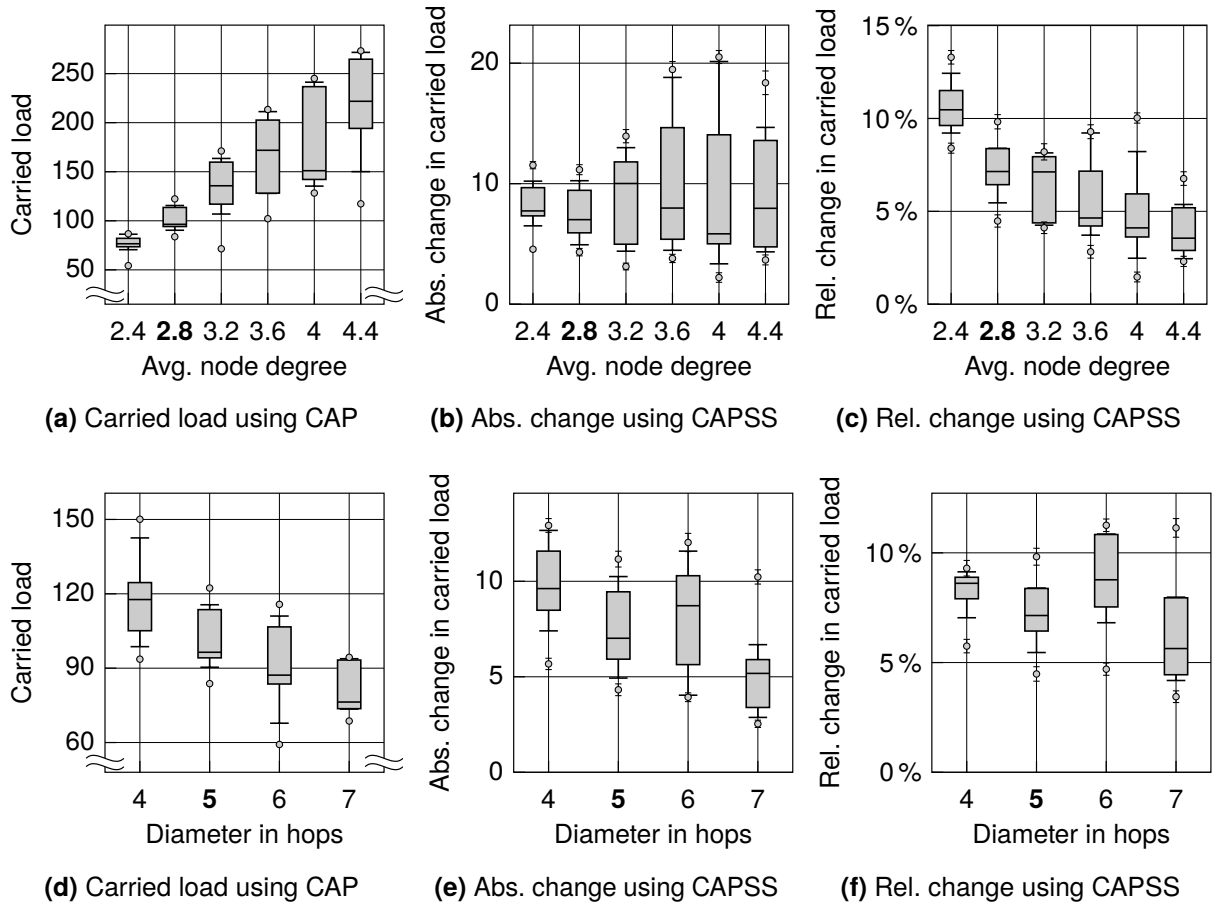
5.5.3.2 Results

Figure 5.20 describes the carried load for the different topology parameters. Figure 5.21 concentrates on the operator-defined parameters, and Figure 5.22 concentrates on the con-

tractual parameters. Like in the previous section, a box plot contains the simulation results of ten randomly generated topology instances. In Figures 5.21 and 5.22, every box plot contains the same ten instances of the default topology with an average node degree of 2.8 and a diameter of 5 hops. In contrast, Figure 5.20 concentrates on the topology parameters, i.e., the average node degree and the diameter. Here, the x-axis value determines the topology parameters. Consequently, every box plot contains a different set of topologies. The average node pair distance is set to 500 km in all simulations. This resembles medium-sized nationwide transport networks (see Table 5.1).

In each of the following figures, the left column shows the carried load for CAP as a reference. The middle column shows the *absolute* change in carried load when CAPSS is compared with CAP. That means the sum of the left and the middle column corresponds to the carried load using CAPSS. Similarly, the right column shows the *relative* change in carried load.

In each plot, the default value, as shown in Table 5.3, is highlighted on the x-axis. Box plots for the default values are equal across different plots, e.g., the box plot in Figure 5.20a at a degree of 2.8 is the same as the box plot in Figure 5.20d at a diameter of 5 hops.



(a) Carried load using CAP **(b)** Abs. change using CAPSS **(c)** Rel. change using CAPSS

(d) Carried load using CAP **(e)** Abs. change using CAPSS **(f)** Rel. change using CAPSS

Figure 5.20 Carried load using CAP and capacity increase using CAPSS for different topology parameters

Topology parameters Figure 5.20a shows the carried load for CAP, i.e., without surplus sharing. The carried load increases with the average node degree because a higher degree means the network contains more links, and more links lead to a higher total link capacity.

Consequently, also more connections can be routed. This effect is mostly independent of the provisioning mechanism in use. However, for the interpretation of Figures 5.20b and 5.20c, it is important to know that the reference level increases. Figure 5.20b shows the additional load the networks carry with CAPSS. While the spread of values increases with the average degree, no real trend is visible in terms of the median. In contrast, Figure 5.20c shows a downward trend for the relative change. That is only logical because the reference (Figure 5.20a) is increasing. Overall, CAPSS is more effective in lower-degree networks. A possible explanation is that a lower average degree often also means that the path diversity in the network is lower. Thus, provisioning on alternative paths and finding backup paths is more difficult than in high-degree networks. However, this also means that CAPSS can better realize its potential in low-degree networks.

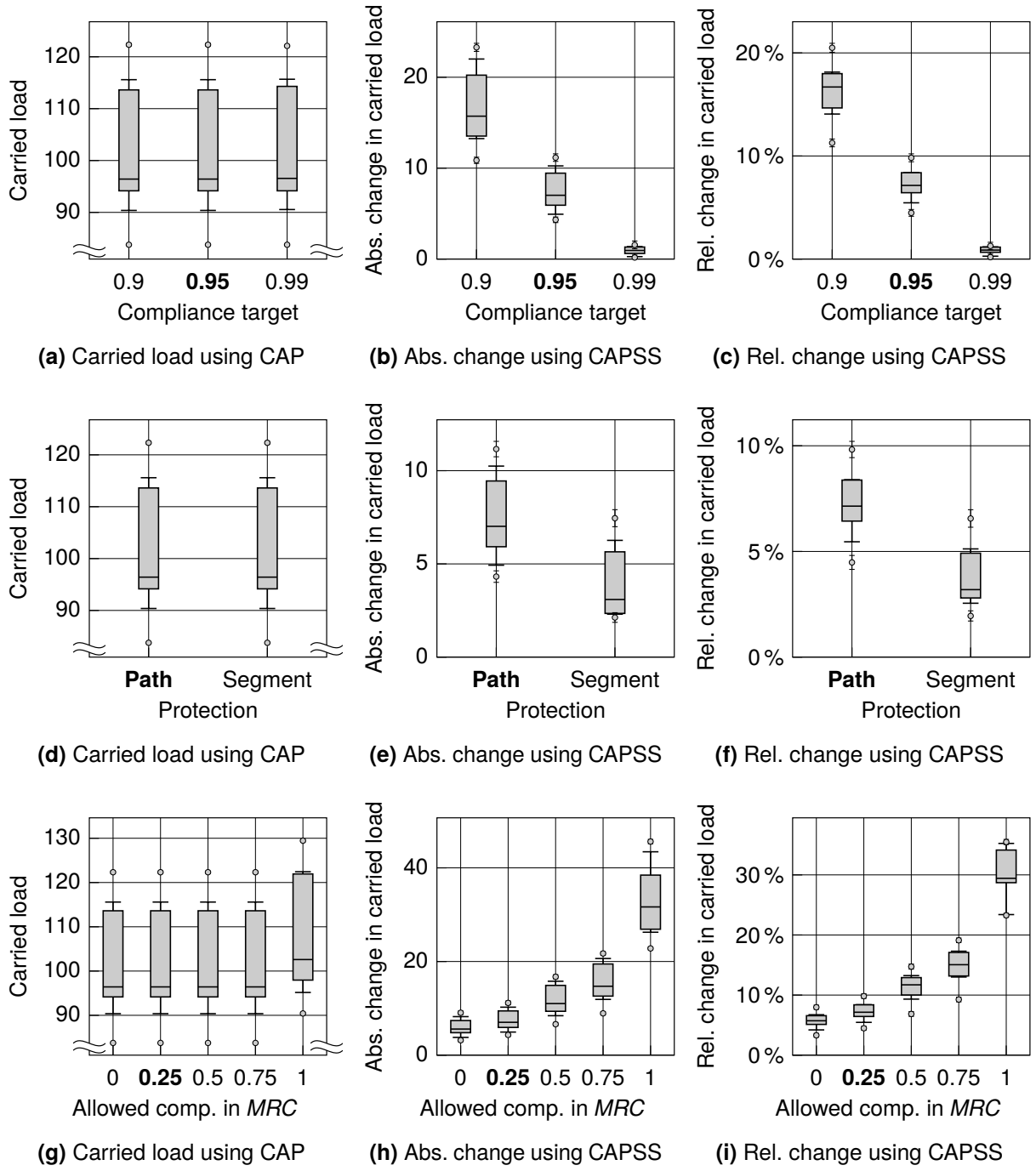
Figures 5.20d to 5.20f show the behavior when the network diameter varies. There is no significant trend visible in the absolute and relative changes.

Operator-defined parameters Next, consider the results for the operator-defined parameters in Figure 5.21. As can be seen in the first column (Figures 5.21a, 5.21d, and 5.21g), the carried load under CAP is mostly independent of the evaluated parameters. One exception is visible in Figure 5.21g for $c_{\max} = 1 \cdot MRC$. While CAP has to protect *all* connections for $c_{\max} < 1 \cdot MRC$, it can route several connections without protection if $c_{\max} = 1 \cdot MRC$ (this information is not visible in the provided figures). This, in turn, frees resources for additional connections. Nevertheless, we conclude that the reference, i.e., the carried load under CAP, does not change significantly with the varied parameters. On the other hand, significant trends are visible for CAPSS, both in the absolute and relative changes.

When the compliance target grows, the increase in network capacity diminishes from a median of 17 % down to a median of 1.0 % (Figure 5.21c). The reason is that a higher compliance target leads to less compliance surplus and requires more reliable routes in the first place. As a result, CAPSS's potential to increase the network capacity is lower for high compliance targets.

Figures 5.21e and 5.21f show that CAPSS increases the network capacity more when the operator employs only path protection instead of segment protection. This behavior can be explained with the help of additional statistics recorded during the simulation but not shown here. The reason for the behavior is as follows. With path protection only, fewer connections are relaxed than with segment protection because more compliance surplus is required to relax a connection. However, if a connection can only be path-protected or completely unprotected, relaxing such a connection means saving a *full* backup path. With segment protection, relaxing a connection can mean saving a full backup path as well, but most of the time, it only means saving a *part* of the backup path. Overall, CAPSS saves less backup resources under segment protection than under path protection in this scenario. Put differently, the amount of backup resources saved per unit of invested compliance surplus is higher under path protection. An additional reason is that the amount of generated compliance surplus is lower with segment protection than with path protection because segment protection often overfulfills the compliance probability less than path protection. This has been indicated in Section 3.3.3 for the steady-state availability but can be transferred to the compliance probability.

Lastly, Figures 5.21h and 5.21i show the capacity increase over the allowed compensation. The higher the allowed compensation, the higher the capacity increase. The reason is that connection requests can be relaxed more easily, i.e., less surplus is required when the allowed compensation is higher. Consequently, the share of relaxed and, hence, unprotected



(a) Carried load using CAP **(b)** Abs. change using CAPSS **(c)** Rel. change using CAPSS

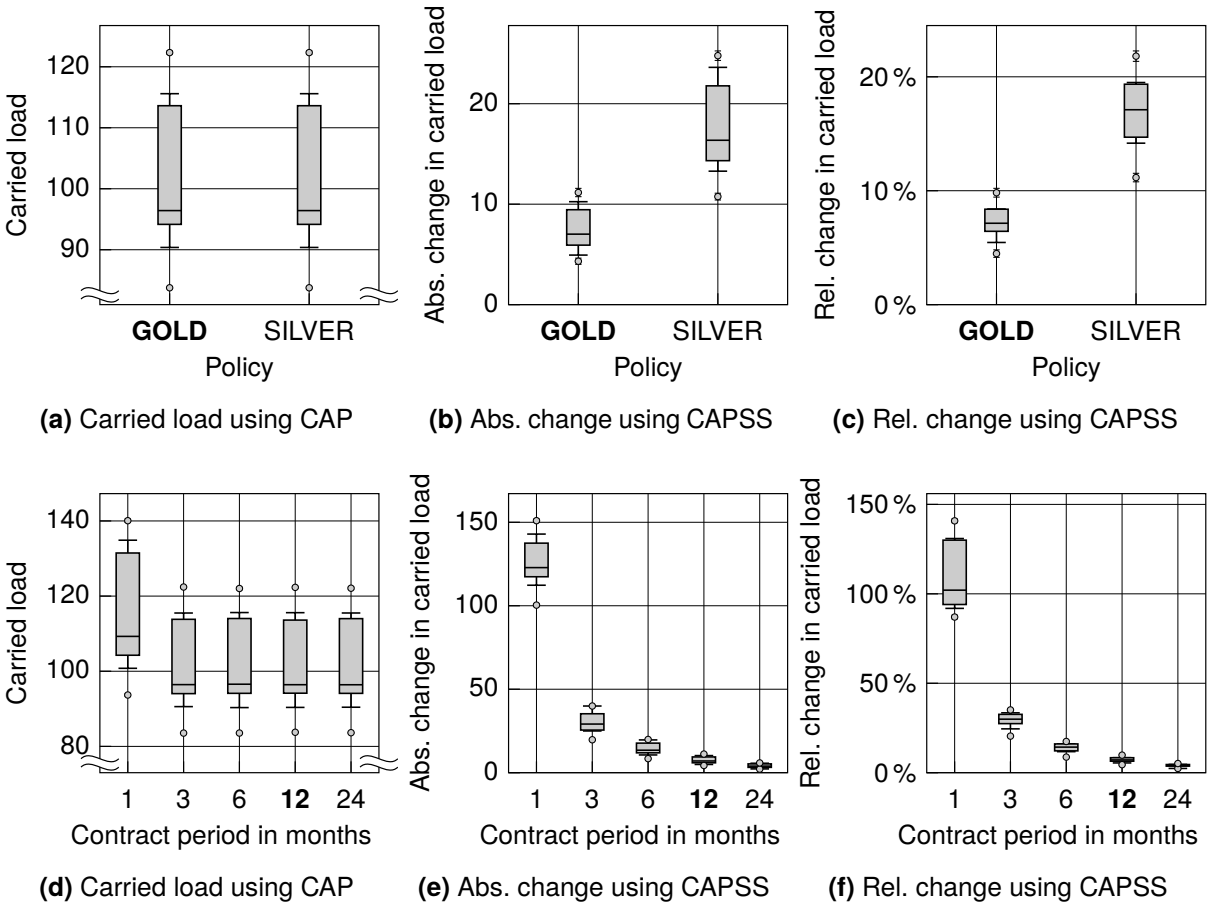
(d) Carried load using CAP **(e)** Abs. change using CAPSS **(f)** Rel. change using CAPSS

(g) Carried load using CAP **(h)** Abs. change using CAPSS **(i)** Rel. change using CAPSS

Figure 5.21 Carried load using CAP and capacity increase using CAPSS for different operator-defined parameters

connections grows. This frees link capacity for additional connections.

In the depicted range of allowed compensation from 0 to $1 \cdot MRC$, the capacity increase is exponential. However, this behavior cannot be easily extrapolated to higher values because more and more connections will be unprotected under CAP already. If they are unprotected under CAP, CAPSS cannot relax them any further, and no more capacity gains are possible. The same restriction exists for an extrapolation of the compliance target to lower values in Figure 5.21c.



(a) Carried load using CAP **(b)** Abs. change using CAPSS **(c)** Rel. change using CAPSS

(d) Carried load using CAP **(e)** Abs. change using CAPSS **(f)** Rel. change using CAPSS

Figure 5.22 Carried load using CAP and capacity increase using CAPSS for different contractual parameters

Contractual parameters Finally, consider Figure 5.22 for the behavior with respect to the contractual parameters. Like before, the carried load under CAP (Figures 5.22a and 5.22d) is constant except for a one-month contract period in Figure 5.22d. With a one-month contract, many connections can be routed without protection by CAP, whereas with longer contract periods, the majority or all connections require path protection and, hence, more link capacity.

Figures 5.22b and 5.22c show the change in carried load for the two compensation policies GOLD and SILVER. For the SILVER policy, CAPSS increases the network capacity much more than for the GOLD policy because, with the SILVER policy, relaxation requires less compliance surplus.

Similarly, Figures 5.22e and 5.22f show that the capacity increase shrinks when the contract period grows. In all simulations, the allowed compensation over the whole contract period is set to $0.25 \cdot MRC$. The longer the contract period, the more difficult it is to adhere to this compensation limit. Therefore, the longer the contract period, the more reliable the routes must be, and hence, relaxations require more and more surplus. Consequently, fewer connections can be relaxed and, hence, less capacity gains are possible. While the median increase for a one-month contract period is 104 %, the median for 24 months is 4.0 %. This wide range is, in part, caused by the fact that the allowed compensation is not scaled with the contract period. We conducted additional simulations in which the allowed compensation was scaled with the contract period such that $c_{\max} = 0.25 \cdot MRC \cdot \Delta t_{\mathrm{h}}/12\,\mathrm{months}$. The

resulting median capacity increase ranges from 81 % for a one-month contract down to 6 % for 24 months. That means the fundamental behavior is the same, but the range is not as wide as with a fixed allowed compensation.

5.5.3.3 Summary

This study evaluated the behavior of CAPSS in a broad range of parameters. In terms of topology parameters, there is a slight trend toward higher capacity increases for lower-degree networks. For the topology diameter, no trend is observable. The evaluation shows that the capacity gain mainly depends on the operator-defined parameters and the contractual parameters. Generally speaking, the stricter the scenario, the lower the achieved capacity gains. Here, stricter means a lower allowed compensation, a higher compliance target, and selection of the GOLD policy instead of the SILVER policy. Segment protection results in lower gains than path protection (3 % vs. 7 %). The level of allowed compensation has a very strong impact on the capacity increase. If a compensation of $1 \cdot MRC$ is allowed, capacity increases of more than 30 % are achievable. Furthermore, shorter contract periods result in much higher capacity gains. For one-month contracts, increases of more than 100 % are possible. In the default configuration, the capacity gains range from 5 % to 10 %.

5.5.4 Behavior in Realistic Networks

After intensive comparisons between CAPSS and CAP in synthetic networks in the previous sections, this section evaluates surplus sharing in realistic networks. The focus is on the capacity increase. However, we also investigate infeasible requests and compare the actual amount of compensation incurred under both mechanisms.

5.5.4.1 Simulation Setup and Evaluation Parameters

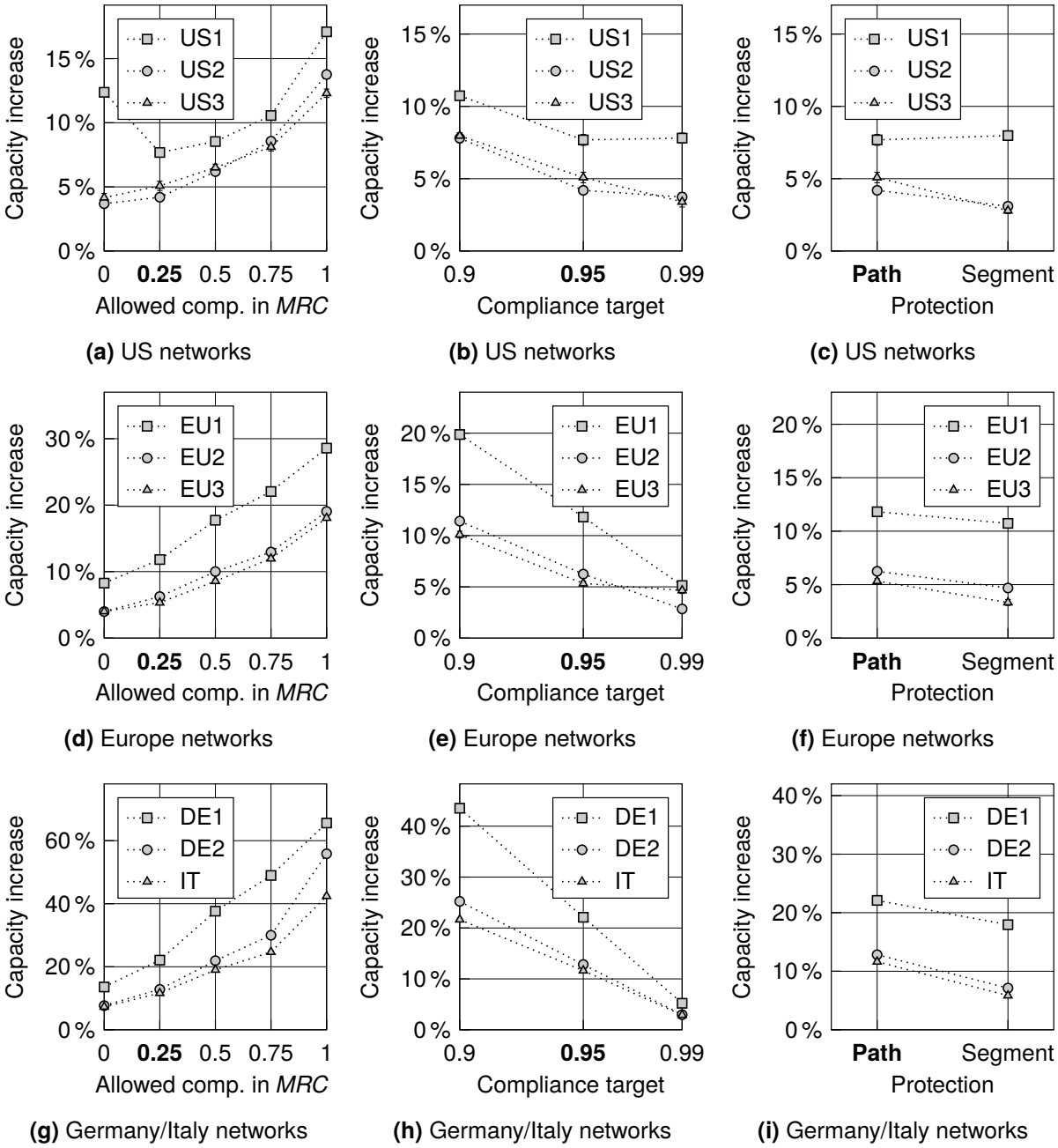
The simulation setup is similar to the one in Section 5.5.3. However, we do not vary any topology parameters because we want to retain the characteristics of the individual networks. Another difference is that we generate connection requests with different contractual parameter values in one simulation run. More precisely, a connection request either uses the GOLD or the SILVER policy and has a contract period of 3, 6, 12, or 24 months. The requested data rates are 1, 5, 10, and 50 Gbit/s. Such a mixed scenario is more realistic than the separate simulations we conducted before. The parameters that remain for variation are the operator-defined ones, i.e., the compliance target, the protection mechanism, and the allowed compensation.

5.5.4.2 Results

We show results for the relative capacity increase in Figure 5.23, the share of infeasible requests in Figure 5.24, and the amount of compensation in Figure 5.25.

The results concerning the capacity increase are in line with the previous results. Therefore, we will only discuss exceptions here. Since the networks have very different topology properties, we do not show the results as box plots but as separate data points in Figures 5.23 and 5.24.

In Figure 5.23a, the capacity increase is monotonically increasing with the allowed compensation, except for the US1 network. Here, the capacity increase of around 12 % for



(a) US networks **(b)** US networks **(c)** US networks

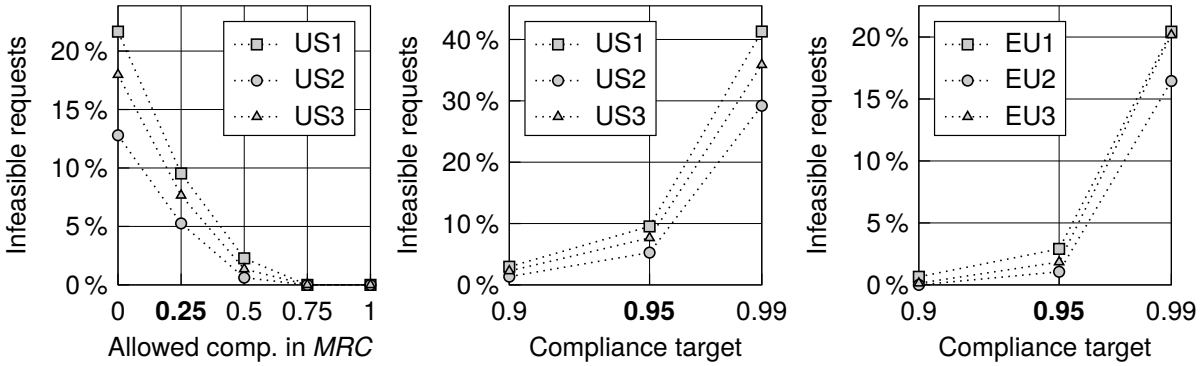
(d) Europe networks **(e)** Europe networks **(f)** Europe networks

(g) Germany/Italy networks **(h)** Germany/Italy networks **(i)** Germany/Italy networks

Figure 5.23 Capacity increase using CAPSS for different operator-defined parameters

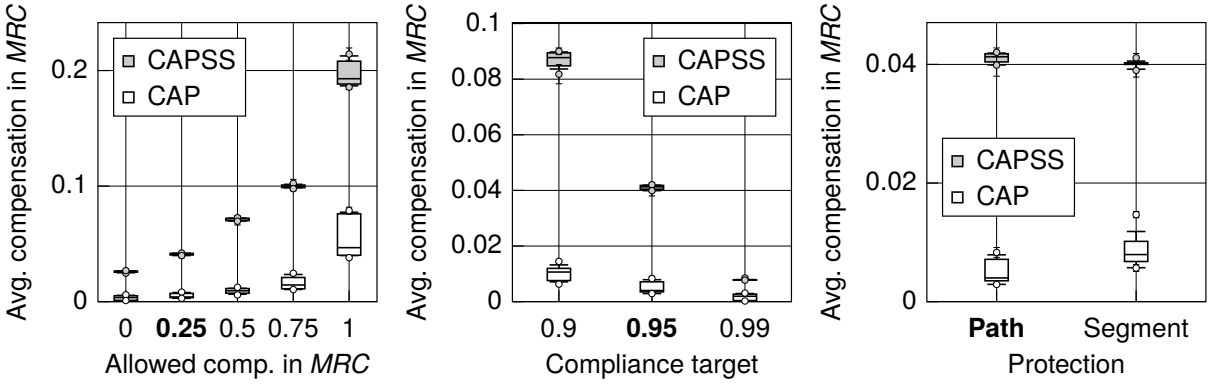
$c_{\max} = 0$ is exceptionally high. The reason is that for $c_{\max} = 0$, more than 20 % of all connection requests are infeasible, as can be seen in Figure 5.24a. As discussed in Section 5.5.2, this may lead to a capacity increase. The same effect, albeit not as strong, can be observed in Figure 5.23b for US1 at a compliance target of $p_t = 0.99$. For US2 in Figure 5.23b and EU3 in Figure 5.23e, the capacity increase almost settles for $p_t = 0.99$ due to the infeasible requests. The corresponding shares of infeasible requests are shown in Figures 5.24b and 5.24c.

Section 5.5.3 showed that the capacity gains for segment protection are smaller than those for path protection. This finding also holds for the realistic networks except for the US1 network. In Figure 5.23c, it can be seen that the capacity increase is almost the same for both types of protection mechanisms. The reason is that the US1 network is rather sparse and has the lowest average node degree among all considered realistic networks. Therefore,



(a) US networks **(b)** US networks **(c)** Europe networks

Figure 5.24 Share of infeasible requests for CAP and CAPSS in specific parametrizations



(a) Over allowed compensation **(b)** Over compliance target **(c)** Over protection

Figure 5.25 Average compensation per connection over all realistic networks

protection of route segments is often impossible, and segment protection effectively equals path protection.

The maximum capacity increase in the US networks is 17 %, and the mean over all depicted parameter configurations is 8 %. For the Europe networks, the maximum and mean gains are 29 % and 11 %. Lastly, for the Germany and Italy networks, the maximum and mean gains are 66 % and 23 %. Generally, the smaller the spatial extent of the networks, the higher the capacity gains are. Equivalently, more reliable networks allow higher capacity gains. This is in line with the findings in Section 5.5.2.

Figure 5.25 shows the average compensation per connection in *MRC*. Each box plot consists of all nine networks. Keep in mind that connections with different contract periods were present in the simulations. It can be seen that CAPSS increases the amount of compensation compared with CAP. This is expected because CAPSS removes compliance overfulfillment and increases the carried traffic. The price is higher compensation. Considering the depicted median values, the relative increase is between 294 % for $p_\mathrm{t} = 0.99$ in Figure 5.25b, and 852 % for the default configuration ($c_\mathrm{max} = 0.25 \cdot MRC$, $p_\mathrm{t} = 0.95$, and path protection). Even though these percentages are high, they are the result of a compliance ratio that matches the compliance target defined and accepted by the operator.

5.5.5 Summary

In Section 5.5, we compared CAPSS with CAP to investigate the effects of compliance surplus sharing. We first identified the behavior in different network loads. It turned out that the main advantage in medium- to high-load scenarios is a reduction in the request rejection ratio and an increase in carried load. In addition to that, CAPSS also reduces the average link utilization for medium loads and, in particular, for low loads. Since this link utilization goes hand in hand with the increase in carried load, this is a positive result because reduced link utilization helps postpone network upgrades and prevents excessive queuing delay.

Considering the network capacity, measured at a rejection ratio of 0.01, the achievable gains mainly depend on the network extent as well as the strictness of the compensation policy and operator-defined parameters. The stricter the parameter values or the larger the network, the lower the capacity gains. Considering the default parameters, the capacity increase in very large networks with average node pair distances $\gg 1000$ km is only a few percent. However, with a decreasing network extent, the capacity increase grows substantially. The average ranges from 5 % to 10 %, but in small networks (around 30 km node pair distance), CAPSS more than doubles the network capacity. For the mixed traffic scenario in the realistic networks, the average capacity increase in the US and Europe networks is 8 % and 11 %, respectively. In the Germany and Italy networks, even a 23 % increase is achieved by employing surplus sharing.

Our simulations also show that surplus sharing is not possible in very small or highly reliable networks because connection requests do not need relaxation in these cases.

Lastly, it is important to note that the reliability of a link is typically modeled as a function of its length. Therefore, statements with respect to the network extent can, in part, be transferred to statements with respect to the network reliability and vice versa. As a consequence, CAPSS can also increase the network capacity substantially in large networks *if* they are highly reliable despite their large extent. Our simulations considered a variety of parametrizations that network operators can map their specific network parameters to.

5.6 Evaluation Summary and Recommendations

In this chapter, we evaluated the performance and accuracy of CAPSS in a broad range of network scenarios. Concerning the accuracy, we showed in Section 5.2 that the approximations in the model and the violation of assumptions indeed lead to small approximation errors. The maximum error is around a probability of 10^{-2} in the evaluated scenarios. However, the majority of observed errors are much smaller. Overall, we consider the errors negligible.

CAPSS combines two ideas, namely compensation-aware provisioning to better incorporate economic aspects into the provisioning process, and compliance surplus sharing to increase the network capacity and prevent SLA overfulfillment. To highlight the differences between compensation-aware provisioning and conventional availability-aware provisioning, we compared CAPSS and CONV in Section 5.4. The two mechanisms do not share the same goal and, therefore, behave very differently. Compared with CONV, CAPSS lowers the compliance ratio and increases the network capacity for connections using the GOLD policy. The opposite is true for the SILVER policy due to its steep slope in the compensation policy function. In mixed traffic scenarios with realistic networks, CAPSS increases the network capacity only in the small networks of Germany and Italy. The larger networks for Europe and the US lose capacity. Furthermore, CAPSS finds infeasible connection requests

when CONV can still provision all requests on routes with sufficient steady-state availability. This can be considered a limitation of CAPSS. On the other hand, it can be considered an indicator that conventional provisioning unconsciously takes too much risk in routing such connections.

On the plus side, in typical network scenarios, CAPSS always provides a compliance ratio that matches the operator-defined compliance target. This enables greater alignment of connection provisioning and the operator's business models and represents a major goal of CAPSS. In contrast, CONV has no means to control the compliance ratio. We recommend that operators select the provisioning mechanism mainly based on whether they want to achieve a certain compliance target and not only based on potential capacity gains.

In Section 5.5, we compared CAPSS and CAP to investigate the benefits of compliance surplus sharing, given that the network operator already employs a compensation-aware provisioning strategy. Surplus sharing allows for the relaxation of connection requests. This has two major effects: Some connections can be provisioned with less protection resources, and some connections can be provisioned on alternative routes. Both effects eventually increase the network capacity and reduce the link utilization. The latter is the main benefit in low-load situations. Network capacity increases are realized in medium- and high-load situations. We focused on practically relevant medium-load situations characterized by a request rejection ratio of 0.01.

Surplus sharing is not advantageous in extremely reliable networks. The reason is that in such networks, CAP alone can provision connections without protection and on diverse routes. Surplus sharing is not required, and as a matter of fact, it is not even possible because there are no connection requests that absorb surplus. This is a clear limitation of CAPSS.

The capacity gains in the default scenario, i.e., path protection, GOLD policy, $p_\mathrm{t} = 0.95$, $c_\mathrm{max} = 0.25 \cdot MRC$, and a contract period of $\Delta t_\mathrm{h} = 12$ months, range from 5 % to 10 % for an average node pair distance of 500 km. Average distances below 500 km, e.g., in the Germany and Italy networks, lead to higher gains. Also, the selection of the SILVER policy and less strict operator-defined parameters increase the capacity gains to around 50 %. In contrast, stricter parameters and larger networks decrease the gains.

For the mixed traffic scenario in the realistic networks, the average capacity increase in the US and Europe networks is 8 % and 11 %, respectively. In the Germany and Italy networks, even a 23 % increase is achieved by employing surplus sharing.

Overall, network operators of typical nationwide transport networks can expect considerable capacity gains by employing CAPSS instead of CAP. However, a precondition is that they adopt the new operator-focused perspective on SLAs and connection provisioning, which targets the overall SLA compensation instead of the individual SLA fulfillment.

6

Conclusions and Outlook

Network operators generate profits from the sale of network services. In order to be economically successful, they have to utilize the network's capacity optimally. Hence, services must be accommodated in a resource-efficient way. However, the traffic demand in today's networks is ever-increasing, and operators must upgrade their networks regularly to provide enough capacity. Technologies like wavelength-division multiplexing (WDM), erbium-doped fiber amplifiers (EDFAs), and coherent transmission have helped augment capacity. However, recent advances like probabilistic constellation shaping (PCS) are approaching the physical limits of the deployed infrastructure. To sustain the increase in capacity, more intelligent and resource-efficient connection provisioning is crucial.

A second important aspect of transport networks is reliability. Service level agreements (SLAs) typically define a guaranteed monthly availability. It is the operator's responsibility to provide this availability. Otherwise, the operator has to compensate the customer according to the compensation policy in the SLA. Availability-aware provisioning mechanisms take the SLA specifications into account. Often, this means that the operator has to deploy recovery mechanisms and redundant network resources. However, this contradicts the goal of resource efficiency, and a balance must be found.

This thesis proposes the connection provisioning mechanism CAPSS, which operates in this field of tension. CAPSS jointly considers aspects of reliability and resource efficiency. Furthermore, it brings technical and economic aspects closer together because it incorporates the SLA compensation directly into the provisioning process.

In Chapter 3, we discussed similar approaches from the literature. First, we considered conventional provisioning mechanisms which select routes based on the steady-state availability. We explained that such mechanisms ignore important stochastic properties of network connections, particularly the probability of SLA violations. Furthermore, they do not incorporate the SLA compensation as an economic parameter. Lastly, they usually overfulfill the SLA requirement and, thus, waste network resources. This counteracts the goal of resource efficiency. We introduced a variety of solutions to those shortcomings from the literature, namely mathematical models for the SLA violation probability and the SLA

compensation and sophisticated provisioning mechanisms. Most of the provisioning mechanisms target resource efficiency while fulfilling the SLA availability. Common schemes are sharing of protection resources and adapted reprovisioning of connections. Other mechanisms go one step further and incorporate economic aspects to maximize the operator's profit. However, none of the approaches considers economic aspects, stochastic properties about connection downtimes, and resource efficiency jointly like CAPSS.

CAPSS comprises two main features. First, it provisions connections based on the SLA compensation the operator has to pay its customer for availability violations. More specifically, it calculates the compliance probability of network routes during connection provisioning. Subsequently, it selects a route such that the compliance probability over all connections in the network matches an operator-defined compliance target. This approach fundamentally differs from conventional SLA-aware provisioning, which only considers a route's steady-state availability. The second feature is that CAPSS exploits SLA overfulfillment to increase the network capacity. SLA overfulfillment is exploited by sharing compliance surplus among connections.

The details of CAPSS have been introduced in Chapter 4. The core of CAPSS is the mathematical model for the probability distribution of a connection's SLA compensation. Contrary to other works in the literature, the model takes into account that the contract for a network connection has a finite contract period that consists of multiple billing cycles. Each billing cycle possibly incurs compensation. CAPSS can be subdivided into the compensation-aware base mechanism CAP and the surplus sharing functionality. CAP provisions connections such that the compliance probability fulfills the operator's compliance target. The compliance probability is calculated with the mathematical model for the compensation. CAP alone achieves two design goals: It considers the connection failure and repair process stochastically, and it incorporates the economic aspect of SLA compensation into the provisioning process. In addition to that, the surplus sharing feature in CAPSS concentrates on resource efficiency. It exploits the overfulfillment of compliance probability by sharing it among connections and relaxing their compliance requirement. As a result, the protection overbuild is reduced, and the path diversity grows.

CAPSS provisions connections such that the amount of compensation is statistically limited to an operator-defined level. This does not necessarily mean that the SLA availability guaranteed to the customer is fulfilled. Consequently, CAPSS fundamentally changes the perspective on connection provisioning from customer-focused to operator-focused.

In Chapter 5, we evaluated the accuracy and the performance of CAPSS in simulation studies. For the performance evaluation, we concentrated on the achievable increase in network capacity since this is one of the main goals of CAPSS. We compared CAPSS with a conventional provisioning mechanism that only considers the SLA availability but not the compensation. The results provide guidelines for operators that consider switching from a conventional mechanism to CAPSS. Furthermore, we investigated the effects of the surplus sharing feature. We considered realistic networks as well as synthetically generated networks with controlled properties.

The results show that the accuracy of CAPSS is generally very good. In specific parametrizations, small and expected approximation errors occur. However, they are negligible in most cases. CAPSS successfully eliminates compliance overfulfillment if connections exist that can absorb compliance surplus. This is a major advantage over the conventional mechanism, which consistently overfulfills its figure of merit, the SLA availability. In contrast, the conventional mechanism underfulfills or overfulfills the operator's compliance target, depending on the parametrization, in particular the compensation policy. This is expected because the

conventional mechanism has no notion of compensation or compliance. Regarding the network capacity, neither of the two mechanisms surpasses the other in each and every scenario. However, CAPSS tends to be advantageous in small networks or, equivalently, networks with high availability. We recommend that network operates choose between CAPSS and a conventional mechanism primarily based on whether compensation awareness is important. The behavior with respect to the network capacity has to be evaluated separately.

In contrast, the comparison between CAPSS and CAP clearly shows that surplus sharing significantly increases the network capacity across a wide range of network scenarios. Only in very reliable networks, surplus sharing does not provide any further benefits over CAP because CAP already maximizes the path diversity and does not need to deploy any protection. Typical capacity increases in the synthetic networks range from 5 % to 10 %. However, much stronger gains are also possible, e.g., in small networks with high availability or if the compensation policy and the operator-defined target parameters are not too strict. The reason is that much surplus for sharing is generated in these cases. For the mixed traffic scenario, the average capacity increase in the US and Europe networks is 8 % and 11 %, respectively. In the Germany and Italy networks, even a 23 % increase is achieved by employing surplus sharing.

Overall, we conclude that CAPSS is a powerful provisioning mechanism with the two major benefits of incorporating the SLA compensation into the provisioning process and eliminating overfulfillment by surplus sharing. While the underlying mathematical model can be considered rather complex, CAPSS is relatively easy to implement in practical network operation compared with other approaches from the literature. It does not involve reprovisioning and, thus, does not render the network unstable. It does not require online information about other connections, which keeps signaling requirements low. Furthermore, the sharing mechanism is realized on a purely mathematical basis. No actual hardware needs to be shared, and no connections must be preempted.

CAPSS has been designed with several assumptions in mind. Most notably, up- and downtimes are assumed to follow exponential distributions, and failures are assumed to occur independently of each other. Of course, such assumptions are idealized. Therefore, additional research is necessary to evaluate the robustness of CAPSS in scenarios that do not adhere to those assumptions. Moreover, modifications of CAPSS are conceivable to align it with such scenarios. For example, the probability distribution of the cumulated downtime per billing cycle for non-exponential downtime distributions can be approximated with the *Panjer recursion* [Pan81]. Furthermore, the supported protection modes could be extended, e.g., to allow more than one backup path.

CAPSS's goal is to limit the overall compensation, not the fulfillment of the SLA availability as such. Hence, SLA violation is tolerated. Therefore, another interesting direction for future research is the effect of CAPSS on the operator's reputation and potential customer churn. Estimating the resulting economic losses and integrating them into an overall model and into the provisioning strategy is a challenging but necessary task to unify technical and economic aspects during connection provisioning further.

Lastly, the concept of SLA compensation and resource-efficient service provisioning is also present in other domains, e.g., in cloud computing and 5G radio access network (RAN) slicing. The concept of CAPSS and, in particular, the model for the distribution of compensation can be adapted to those use cases.

References

[Ahm+21] Tanjila Ahmed, Abhijit Mitra, Sabidur Rahman, Massimo Tornatore, Andrew Lord, and Biswanath Mukherjee. "C+L-band upgrade strategies to sustain traffic growth in optical backbone networks." In: *Journal of Optical Communications and Networking* 13.7 (2021), pp. 193–203. DOI: 10.1364/JOCN.427097.

[Ale09] Carol Alexander. *Market Risk Analysis: Value-at-Risk Models*. Vol. 4. Market Risk Analysis. Wiley, 2009. ISBN: 978-0-470-99788-8.

[Bax81] Laurence A. Baxter. "Availability measures for a two-state system." In: *Journal of Applied Probability* 18.1 (1981), pp. 227–235. DOI: 10.2307/3213182.

[Bir17] Alessandro Birolini. *Reliability Engineering: Theory and Practice*. 8th ed. Springer, 2017. DOI: 10.1007/978-3-662-54209-5.

[Bra08] Ulrik Brandes. "On variants of shortest-path betweenness centrality and their generic computation." In: *Social Networks* 30.2 (2008), pp. 136–145. DOI: 10.1016/j.socnet.2007.11.001.

[BRV20] Sima Barzegar, Marc Ruiz, and Luis Velasco. "Soft-Failure Localization and Time-Dependent Degradation Detection for Network Diagnosis." In: *International Conference on Transparent Optical Networks (ICTON)*. 2020. DOI: 10.1109/ICTON51198.2020.9203029.

[Cav+07] Cicek Cavdar, Lei Song, Massimo Tornatore, and Biswanath Mukherjee. "Holding-Time-Aware and Availability-Guaranteed Connection Provisioning in Optical WDM Mesh Networks." In: *International Symposium on High Capacity Optical Networks and Enabling Technologies*. 2007. DOI: 10.1109/HONET.2007.4600278.

[Cen11] CenturyLink. *CenturyLink Domestic Optical Wavelength Service; Service Level Agreement*. Aug. 2011. URL: https://assets.lumen.com/is/content/Lumen/qcc-optical-wavelength-service-sla. Accessed: June 26, 2023.

[CGR14] Piotr Chołda, Piotr Guzik, and Krzysztof Rusek. "Risk-awareness in Resilient Networks Design: Value-at-Risk is Enough." In: *International Telecommunications Network Strategy and Planning Symposium*. 2014. DOI: 10.1109/NETWKS.2014.6958530.

[Che+15] Xiaoliang Chen, Massimo Tornatore, Shilin Zhu, Fan Ji, Wenshuang Zhou, Cen Chen, Daoyun Hu, Liu Jiang, and Zuqing Zhu. "Flexible Availability-Aware Differentiated Protection in Software-Defined Elastic Optical Networks." In: *Journal of Lightwave Technology* 33.18 (2015), pp. 3872–3882. DOI: 10.1109/JLT.2015.2456152.

[Cho+07a] Baek-Young Choi, Sue Moon, Zhi-Li Zhang, Konstantina Papagiannaki, and Christophe Diot. "Analysis of point-to-point packet delay in an operational network." In: *Computer Networks* 51.13 (2007), pp. 3812–3827. DOI: 10.1016/j.comnet.2007.04.004.

[Cho+07b] Piotr Chołda, Anders Mykkeltveit, Bjarne E. Helvik, Otto J. Wittner, and Andrzej Jajszczyk. "A Survey of Resilience Differentiation Frameworks in Communication Networks." In: *IEEE Communications Surveys & Tutorials* 9.4 (2007), pp. 32–55. DOI: 10.1109/COMST.2007.4444749.

[Cis19a] Cisco. *Cisco ASR 9000 Series Aggregation Services Router Overview and Reference Guide*. 2019.

[Cis19b] Cisco. *Cisco Visual Networking Index: Forecast and Trends 2017–2022*. 2019.

[Cis20] Cisco. *Cisco Annual Internet Report (2018–2023)*. 2020.

[CJH16] Marcel Caria, Admela Jukan, and Marco Hoffmann. "SDN Partitioning: A Centralized Control Plane for Distributed Routing Protocols." In: *IEEE Transactions on Network and Service Management* 13.3 (2016), pp. 381–393. DOI: 10.1109/TNSM.2016.2585759.

[CK04] Marek Capiński and Peter Ekkehard Kopp. *Measure, Integral and Probability*. 2nd ed. Springer, 2004. DOI: 10.1007/978-1-4471-0645-6.

[CL21] Christos G. Cassandras and Stéphane Lafortune. *Introduction to Discrete Event Systems*. 3rd ed. Springer, 2021. DOI: 10.1007/978-3-030-72274-6.

[Cle+05] Roberto Clemente, Maurizio Bartoli, Maria Chiara Bossi, Giancarlo D'Orazio, and Giuseppe Cosmo. "Risk Management in Availability SLA." In: *International Workshop on Design of Reliable Communication Networks (DRCN)*. 2005. DOI: 10.1109/DRCN.2005.1563900.

[CRG16] Piotr Chołda, Krzysztof Rusek, and Piotr Guzik. "Upper Bound for Failure Risk in Networks." In: *Electronic Notes in Discrete Mathematics* 51 (2016), pp. 31–38. DOI: 10.1016/j.endm.2016.01.005.

[Cug+13] Filippo Cugini, Francesco Paolucci, Gianluca Meloni, Gianluca Berrettini, Marco Secondini, Francesco Fresi, Nicola Sambo, Luca Potì, and Piero Castoldi. "Push-Pull Defragmentation Without Traffic Disruption in Flexible Grid Optical Networks." In: *Journal of Lightwave Technology* 31.1 (2013), pp. 125–133. DOI: 10.1109/JLT.2012.2225600.

[CW19] Junho Cho and Peter J. Winzer. "Probabilistic Constellation Shaping for Optical Fiber Communications." In: *Journal of Lightwave Technology* 37.6 (2019), pp. 1590–1607. DOI: 10.1109/JLT.2019.2898855.

[Das12] Ananya Das. "Maximizing Profit Using SLA-Aware Provisioning." In: *IEEE Network Operations and Management Symposium (NOMS)*. 2012. DOI: 10.1109/NOMS.2012.6211923.

[DGM94] D. Anthony Dunn, Wayne D. Grover, and Mike H. MacGregor. "Comparison of k-Shortest Paths and Maximum Flow Routing for Network Facility Restoration." In: *IEEE Journal on Selected Areas in Communications* 12.1 (1994), pp. 88–99. DOI: 10.1109/49.265708.

[DTM15] Ferhat Dikbiyik, Massimo Tornatore, and Biswanath Mukherjee. “Exploiting Excess Capacity, Part II: Differentiated Services Under Traffic Growth.” In: *IEEE/ACM Transactions on Networking* 23.5 (2015), pp. 1599–1609. DOI: 10.1109/TNET.2014.2335252.

[EK21] Tobias Enderle and Andreas Kirstädter. “Reducing Network Operators’ Expenses by Adjusting the MTTR.” In: *International Telecommunication Networks and Applications Conference (ITNAC)*. 2021. DOI: 10.1109/ITNAC53136.2021.9652138.

[Ela+19] Salah Eddine Elayoubi, Sana Ben Jemaa, Zwi Altman, and Ana Galindo-Serrano. “5G RAN Slicing for Verticals: Enablers and Challenges.” In: *IEEE Communications Magazine* 57.1 (2019), pp. 28–34. DOI: 10.1109/MCOM.2018.1701319.

[End21] Tobias Enderle. “On the Impact of Billing Cycles on Compensations in Network SLAs.” In: *International Conference on Optical Network Design and Modeling (ONDM)*. 2021. DOI: 10.23919/ONDM51796.2021.9492378.

[End22] Tobias Enderle. “Protected Connection Provisioning with Low Availability Overfulfillment in Meshed Core Networks.” In: *ITG-Symposium on Photonic Networks*. 2022.

[EPP12] António Eira, João Pedro, and João Pires. “Optimized Design of Shared Restoration in Flexible-Grid Transparent Optical Networks.” In: *National Fiber Optic Engineers Conference (NFOEC)*. 2012. DOI: 10.1364/NFOEC.2012.JTh2A.37.

[FB06] Adrian Farrel and Igor Bryskin. *GMPLS: Architecture and Applications*. Morgan Kaufmann, 2006. DOI: 10.1016/B978-0-12-088422-3.X5000-X.

[FH16] Eirik L. Følstad and Bjarne E. Helvik. “The cost for meeting SLA dependability requirements; implications for customers and providers.” In: *Reliability Engineering & System Safety* 145 (2016), pp. 136–146. DOI: 10.1016/j.ress.2015.09.011.

[FR91] Thomas M. J. Fruchterman and Edward M. Reingold. “Graph Drawing by Force-directed Placement.” In: *Journal of Software: Practice and Experience* 21.11 (1991), pp. 1129–1164. DOI: 10.1002/spe.4380211102.

[FS17] Alvaro Fernandez and Norvald Stol. “Economic, Dissatisfaction, and Reputation Risks of Hardware and Software Failures in PONs.” In: *IEEE/ACM Transactions on Networking* 25.2 (2017), pp. 1119–1132. DOI: 10.1109/TNET.2016.2619062.

[G.694.1] ITU-T. *Spectral grids for WDM applications: DWDM frequency grid*. G.694.1. Oct. 2020.

[G.808.1] ITU-T. *Generic protection switching – Linear trail and subnetwork protection*. G.808.1. May 2014.

[G.808.3] ITU-T. *Generic protection switching – Shared mesh protection*. G.808.3. Oct. 2012.

[G.841] ITU-T. *Types and characteristics of SDH network protection architectures*. G.841. Oct. 1998.

[G.873.1] ITU-T. *Optical transport network: Linear protection*. G.873.1. Oct. 2017.

[G.873.3] ITU-T. *Optical transport network – Shared mesh protection*. G.873.3. Sept. 2017.

[GH11] Andrés J. González and Bjarne E. Helvik. "Analysis of Failures Characteristics in the UNINETT IP Backbone Network." In: *IEEE International Conference on Advanced Information Networking and Applications*. 2011. DOI: 10.1109/WAINA.2011.55.

[GH12a] Andrés J. González and Bjarne E. Helvik. "A Study of the Interval Availability and Its Impact on SLAs Risk." In: *Advances in Computer Science, Engineering & Applications*. Ed. by David C. Wyld, Jan Zizka, and Dhinaharan Nagamalai. Springer, 2012, pp. 879–890. DOI: 10.1007/978-3-642-30157-5_87.

[GH12b] Andrés J. González and Bjarne E. Helvik. "Guaranteeing SLA Availability in Telecommunications Networks." In: *International Telecommunications Network Strategy and Planning Symposium*. 2012. DOI: 10.1109/NETWKS.2012.6381678.

[GH12c] Andrés J. González and Bjarne E. Helvik. "System Management to Comply with SLA Availability Guarantees in Cloud Computing." In: *IEEE International Conference on Cloud Computing Technology and Science*. 2012. DOI: 10.1109/CloudCom.2012.6427508.

[GH14] Matthias Gunkel and Martin Horneffer. "Multi-layer Resilience in Deutsche Telekom's IP Core and Aggregation Network." In: *ITG-Symposium on Photonic Networks*. 2014.

[Glo11] Global Crossing. *Service Terms and SLA for Wavelength Service*. Sept. 2011. URL: https://assets.lumen.com/is/content/Lumen/gc-wavelength-service-sep-2011. Accessed: June 27, 2023.

[Gon+10] Andrés J. González, Bjarne E. Helvik, Jon K. Hellan, and Pirkko Kuusela. "Analysis of Dependencies Between Failures in the UNINETT IP Backbone Network." In: *IEEE Pacific Rim International Symposium on Dependable Computing*. 2010. DOI: 10.1109/PRDC.2010.12.

[Gro04] Wayne D. Grover. *Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking*. Prentice Hall, 2004. ISBN: 0-13-494576-X.

[Gun+12] Matthias Gunkel, Achim Autenrieth, Markus Neugirg, and Jörg-Peter Elbers. "Advanced Multilayer Resilience Scheme with Optical Restoration for IP-over-DWDM Core Networks." In: *International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT)*. 2012. DOI: 10.1109/ICUMT.2012.6459749.

[Has+13] Avinatan Hassidim, Danny Raz, Michal Segalov, and Ariel Shaqed. "Network Utilization: the Flow View." In: *IEEE International Conference on Computer Communications (INFOCOM)*. 2013. DOI: 10.1109/INFCOM.2013.6566937.

[Hua+04] Yurong (Grace) Huang, Jonathan P. Heritage, Biswanath Mukherjee, and Wushao Wen. "Availability-Guaranteed Service Provisioning with Shared-Path Protection in Optical WDM Networks." In: *Optical Fiber Communication Conference (OFC)*. 2004.

[Hua23] Huawei. *iMaster NCE-T*. 2023. URL: https://e.huawei.com/en/products/network-analysis/imaster-nce-t. Accessed: January 26, 2023.

[KGV83] Scott Kirkpatrick, C. Daniel Gelatt, Jr., and Mario P. Vecchi. “Optimization by Simulated Annealing.” In: *Science* 220.4598 (1983), pp. 671–680. DOI: 10.1126/science.220.4598.671.

[KMO08a] Burak Kantarci, Hussein T. Mouftah, and Sema Oktuğ. “Arranging Shareability Dynamically for the Availability-Constrained Design of Optical Transport Networks.” In: *IEEE Symposium on Computers and Communications*. 2008. DOI: 10.1109/ISCC.2008.4625603.

[KMO08b] Burak Kantarci, Hussein T. Mouftah, and Sema Oktuğ. “Availability Analysis and Connection Provisioning in Overlapping Shared Segment Protection for Optical Networks.” In: *International Symposium on Computer and Information Sciences*. 2008. DOI: 10.1109/ISCIS.2008.4717963.

[KMO08c] Burak Kantarci, Hussein T. Mouftah, and Sema Oktuğ. “Connection Provisioning with Feasible Shareability Determination for Availability-Aware Design of Optical Networks.” In: *International Conference on Transparent Optical Networks (ICTON)*. 2008. DOI: 10.1109/ICTON.2008.4598645.

[KMO08d] Burak Kantarci, Hussein T. Mouftah, and Sema Oktuğ. “Differentiated Availability-Aware Connection Provisioning in Optical Transport Networks.” In: *IEEE Global Telecommunications Conference (GLOBECOM)*. 2008. DOI: 10.1109/GLOCOM.2008.ECP.520.

[KMO09] Burak Kantarci, Hussein T. Mouftah, and Sema F. Oktuğ. “Adaptive Schemes for Differentiated Availability-Aware Connection Provisioning in Optical Transport Networks.” In: *Journal of Lightwave Technology* 27.20 (2009), pp. 4595–4602. DOI: 10.1109/JLT.2009.2025246.

[Lew95] Elmer E. Lewis. *Introduction to Reliability Engineering*. 2nd ed. Wiley, 1995. ISBN: 978-0-471-01833-9.

[Li+02] Guangzhi Li, Dongmei Wang, Charles Kalmanek, and Robert Doverspike. “Efficient Distributed Path Selection for Shared Restoration Connections.” In: *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. 2002. DOI: 10.1109/INFCOM.2002.1019255.

[LM12] Rafael B. R. Lourenço and Darli A. A. Mello. “On the Exponential Assumption for the Time-to-Repair in Optical Network Availability Analysis.” In: *International Conference on Transparent Optical Networks (ICTON)*. 2012. DOI: 10.1109/ICTON.2012.6253743.

[Luc+09] Diego Lucerna, Massimo Tornatore, Biswanath Mukherjee, and Achille Pattavina. “Availability Target Redefinition for Dynamic Connections in WDM Networks with Shared Path Protection.” In: *International Workshop on Design of Reliable Communication Networks (DRCN)*. 2009. DOI: 10.1109/DRCN.2009.5340002.

[Luc+12] Diego Lucerna, Massimo Tornatore, Biswanath Mukherjee, and Achille Pattavina. “Trading availability among shared-protected dynamic connections in WDM networks.” In: *Computer Networks* 56.13 (2012), pp. 3150–3162. DOI: 10.1016/j.comnet.2012.04.021.

[Lum23] Lumen Technologies. *Lumen Service Level Agreement*. Apr. 2023. URL: https://assets.lumen.com/is/content/Lumen/lumen-service-level-agreementpdf. Accessed: June 26, 2023.

[Mar+08] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen-Nee Chuah, Yashar Ganjali, and Christophe Diot. "Characterization of Failures in an Operational IP Backbone Network." In: *IEEE/ACM Transactions on Networking* 16.4 (2008), pp. 749–762. DOI: 10.1109/TNET.2007.902727.

[Mel+05] Darli A. A. Mello, Jefferson U. Pelegrini, Rafael P. Ribeiro, Dominic A. Schupke, and Helio Waldman. "Dynamic Provisioning of Shared-Backup Path Protected Connections with Guaranteed Availability Requirements." In: *International Conference on Broadband Networks*. 2005. DOI: 10.1109/ICBN.2005.1589761.

[Mel+06] Darli A. A. Mello, Gustavo S. Quitério, Helio Waldman, and Dominic A. Schupke. "Specification of SLA Survivability Requirements for Optical Path Protected Connections." In: *National Fiber Optic Engineers Conference (NFOEC)*. 2006. DOI: 10.1109/OFC.2006.215857.

[Mic+20] Dimitrios Michail, Joris Kinable, Barak Naveh, and John V. Sichi. "JGraphT—A Java Library for Graph Data Structures and Algorithms." In: *ACM Transactions on Mathematical Software* 46.2 (2020). DOI: 10.1145/3381449.

[ML13] Darli A. A. Mello and Rafael B. Lourenço. "New Analytical Method to Compute Availability Non-Compliance Risks for Lambda Services." In: *National Fiber Optic Engineers Conference (NFOEC)*. 2013. DOI: 10.1364/NFOEC.2013.NW4I.4.

[MMN19] Loretta Mastroeni, Alessandro Mazzoccoli, and Maurizio Naldi. "Service Level Agreement Violations in Cloud Storage: Insurance and Compensation Sustainability." In: *Future Internet* 11.7 (2019). DOI: 10.3390/fi11070142.

[MN11a] Loretta Mastroeni and Maurizio Naldi. "Compensation Policies and Risk in Service Level Agreements: A Value-at-Risk Approach under the ON-OFF Service Model." In: *Economics of Converged, Internet-Based Networks*. Ed. by Johanne Cohen, Patrick Maillé, and Burkhard Stiller. Springer, 2011, pp. 2–13. DOI: 10.1007/978-3-642-24547-3_2.

[MN11b] Loretta Mastroeni and Maurizio Naldi. "Violation of Service Availability Targets in Service Level Agreements." In: *Federated Conference on Computer Science and Information Systems (FedCSIS)*. 2011.

[MR07] Deepankar Medhi and Karthikeyan Ramasamy. *Network Routing: Algorithms, Protocols, and Architectures*. Morgan Kaufmann, 2007. ISBN: 978-0-12-088588-6.

[MS08] Clara Meusburger and Dominic A. Schupke. "Method to Estimate the Break-Even Point Between SLA Penalty Expenses and Protection Costs." In: *Optical Fiber Communication Conference (OFC)*. 2008. DOI: 10.1109/ofc.2008.4528516.

[Muk+20] Biswanath Mukherjee, Ioannis Tomkos, Massimo Tornatore, Peter Winzer, and Yongli Zhao, eds. *Springer Handbook of Optical Networks*. Springer, 2020. DOI: 10.1007/978-3-030-16250-4.

[Muk06] Biswanath Mukherjee. *Optical WDM Networks*. Springer, 2006. DOI: 10.1007/0-387-29188-1.

[Mus+19] Francesco Musumeci, Omran Ayoub, Monica Magoni, and Massimo Tornatore. “Latency-Aware CU Placement/Handover in Dynamic WDM Access-Aggregation Networks.” In: *Journal of Optical Communications and Networking* 11.4 (2019), B71–B82. DOI: 10.1364/JOCN.11.000B71.

[MWQ11] Darli A. A. Mello, Helio Waldman, and Gustavo S. Quitério. “Interval availability estimation for protected connections in optical networks.” In: *Computer Networks* 55.1 (2011), pp. 193–204. DOI: 10.1016/j.comnet.2010.07.018.

[Naf+11] Alireza Nafarieh, Shyamala C. Sivakumar, William Phillips, and William Robertson. “Memory-aware SLA-based Mechanism for Shared-Mesh WDM Networks.” In: *International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT)*. 2011.

[Nal17] Maurizio Naldi. “Evaluation of Customer’s Losses and Value-at-Risk under Cloud Outages.” In: *International Conference on Telecommunications and Signal Processing (TSP)*. 2017. DOI: 10.1109/TSP.2017.8075927.

[Nok19] Nokia. *NSP Network Services Platform: System Architecture Guide*. Release 19.9, 3HE-15139-AAAC-TQZZA, Issue 1. 2019.

[NRR15] Alireza Nafarieh, Muhammad Raza, and William Robertson. “A comprehensive analysis of QoS-based routing mechanisms over shared mesh protected optical infrastructures.” In: *Journal of Ambient Intelligence and Humanized Computing* 6.4 (2015), pp. 463–472. DOI: 10.1007/s12652-015-0291-x.

[Ora12] Orange. *Service Level Agreement for International Ethernet Link Service*. Mar. 2012. URL: https://www.orange-business.com/sites/default/files/sla.international_ethernet_link.gbl_.03-12.pdf. Accessed: June 27, 2023.

[Orl+10] Sebastian Orlowski, Roland Wessäly, Michal Pióro, and Artur Tomaszewski. “SNDlib 1.0—Survivable Network Design Library.” In: *Networks* 55.3 (2010), pp. 276–286. DOI: 10.1002/net.20371.

[Pán+06] Zsolt Pándi, Marco Tacca, Andrea Fumagalli, and Lena Wosinska. “Dynamic Provisioning of Availability-Constrained Optical Circuits in the Presence of Optical Node Failures.” In: *Journal of Lightwave Technology* 24.9 (2006), pp. 3268–3279. DOI: 10.1109/JLT.2006.879505.

[Pan81] Harry H. Panjer. “Recursive evaluation of a family of compound distributions.” In: *ASTIN Bulletin* 12.1 (1981), pp. 22–26. DOI: 10.1017/S0515036100006796.

[Poi17] Yvan Pointurier. “Design of Low-Margin Optical Networks.” In: *Journal of Optical Communications and Networking* 9.1 (2017), A9–A17. DOI: 10.1364/JOCN.9.0000A9.

[Raz+19] Muhammad Rehan Raza, Carlos Natalino, Peter Öhlen, Lena Wosinska, and Paolo Monti. “Reinforcement Learning for Slicing in a 5G Flexible RAN.” In: *Journal of Lightwave Technology* 37.20 (2019), pp. 5161–5169. DOI: 10.1109/JLT.2019.2924345.

[REB21] Ronald Romero Reyes, Shkurte Esati, and Thomas Bauschert. “Traffic Protection in Multilayer Core Networks by Optimum Thinning of MPLS Tunnel Capacities.” In: *International Conference on Optical Network Design and Modeling (ONDM)*. 2021. DOI: 10.23919/ONDM51796.2021.9492463.

[RFC4090] Alia Atlas, George Swallow, and Ping Pan. *Fast Reroute Extensions to RSVP-TE for LSP Tunnels*. RFC 4090. May 2005.

[RFC4872] Jonathan Lang, Dimitri Papadimitriou, and Yakov Rekhter. *RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery*. RFC 4872. May 2007.

[RFC4873] Adrian Farrel, Igor Bryskin, Dimitri Papadimitriou, and Lou Berger. *GMPLS Segment Recovery*. RFC 4873. May 2007.

[RGC16] Krzysztof Rusek, Piotr Guzik, and Piotr Chołda. "Effective Risk Assessment in Resilient Communication Networks." In: *Journal of Network and Systems Management* 24 (2016), pp. 491–515. DOI: 10.1007/s10922-016-9370-3.

[Sev+11] Spencer Sevilla, Ming Xia, Charles U. Martel, and Biswanath Mukherjee. "Time-Differentiated Resilience in Telecom Mesh Networks." In: *IEEE International Conference on Communications (ICC)*. 2011. DOI: 10.1109/icc.2011.5963300.

[SG86] Edmundo de Souza e Silva and H. Richard Gail. "Calculating Cumulative Operational Time Distributions of Repairable Computer Systems." In: *IEEE Transactions on Computers* C-35.4 (1986), pp. 322–332. DOI: 10.1109/TC.1986.1676765.

[Sim08] Jane M. Simmons. *Optical Network Design and Planning*. Springer, 2008. DOI: 10.1007/978-0-387-76476-4.

[SS10] Jörg Sommer and Joachim Scharf. "IKR Simulation Library." In: *Modeling and Tools for Network Simulation*. Ed. by Klaus Wehrle, Mesut Güneş, and James Gross. Springer, 2010, pp. 61–68. DOI: 10.1007/978-3-642-12331-3_4.

[SW07] Andrew P. Snow and Gary R. Weckman. "What Are The Chances An Availability SLA Will Be Violated?" In: *International Conference on Networking (ICN)*. 2007. DOI: 10.1109/ICN.2007.106.

[SZM05] Lei Song, Jing Zhang, and Biswanath Mukherjee. "Dynamic Provisioning with Reliability Guarantee and Resource Optimization for Differentiated Services in WDM Mesh Networks." In: *Optical Fiber Communication Conference (OFC)*. 2005. DOI: 10.1109/OFC.2005.192746.

[SZM07] Lei Song, Jing Zhang, and Biswanath Mukherjee. "Dynamic Provisioning with Availability Guarantee for Differentiated Services in Survivable Mesh Networks." In: *IEEE Journal on Selected Areas in Communications* 25.4 (2007), pp. 35–43. DOI: 10.1109/TWC.2007.024505.

[Tak57] Lajos Takács. "On certain sojourn time problems in the theory of stochastic processes." In: *Acta Mathematica Academiae Scientiarum Hungarica* 8 (1957), pp. 169–191. DOI: 10.1007/BF02025241.

[TEL23] TELUS. *TELUS Wavelength Service Terms*. 2023. URL: https://www.telus.com/en/bc/business/support/article/telus-wavelength-service. Accessed: June 27, 2023.

[Tip14] David Tipper. "Resilient network design: challenges and future directions." In: *Telecommunication Systems* 56 (2014), pp. 5–16. DOI: 10.1007/s11235-013-9815-x.

[Tka10] Robert W. Tkach. “Scaling Optical Communications for the Next Decade and Beyond.” In: *Bell Labs Technical Journal* 14.4 (2010), pp. 3–9. DOI: 10.1002/bltj.20400.

[Tor+08] Massimo Tornatore, Diego Lucerna, Lei Song, Biswanath Mukherjee, and Achille Pattavina. “Dynamic SLA Redefinition for Shared-Path-Protected Connections with Known Duration.” In: *Optical Fiber Communication Conference (OFC)*. 2008. DOI: 10.1109/OFC.2008.4528258.

[Tor+12] Massimo Tornatore, Diego Lucerna, Biswanath Mukherjee, and Achille Pattavina. “Multilayer Protection with Availability Guarantees in Optical WDM Networks.” In: *Journal of Network and Systems Management* 20 (2012), pp. 34–55. DOI: 10.1007/s10922-011-9210-4.

[TW11] Andrew S. Tanenbaum and David J. Wetherall. *Computer Networks*. 5th ed. Prentice Hall, 2011. ISBN: 978-0-13-212695-3.

[Uch14] Masato Uchida. “Statistical characteristics of serious network failures in Japan.” In: *Reliability Engineering & System Safety* 131 (2014), pp. 126–134. DOI: 10.1016/j.ress.2014.07.001.

[Ver+05] Sofie Verbrugge, Didier Colle, Piet Demeester, Ralf Huelsermann, and Monika Jaeger. “General Availability Model for Multilayer Transport Networks.” In: *International Workshop on Design of Reliable Communication Networks (DRCN)*. 2005. DOI: 10.1109/DRCN.2005.1563848.

[Ver20] Verizon. *Wavelength Services Solution +; Part IV: Service Level Agreement*. 2020. URL: https://www.verizon.com/business/service_guide/reg/cp_wss_plus_sla.pdf. Accessed: June 26, 2023.

[Von+15] Ann Von Lehmen, Robert Doverspike, George Clapp, Douglas M. Freimuth, Joel Gannett, Aleksandar Kolarov, Haim Kobrinski, Christian Makaya, Emmanuil Mavrogiorgis, Jorge Pastor, Michael Rauch, K. K. Ramakrishnan, Ron Skoog, Brian Wilson, and Sheryl L. Woodward. “CORONET: Testbeds, Demonstration, and Lessons Learned [Invited].” In: *Journal of Optical Communications and Networking* 7.3 (2015), A447–A458. DOI: 10.1364/JOCN.7.00A447.

[VPD04] Jean-Philippe Vasseur, Mario Pickavet, and Piet Demeester. *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*. Morgan Kaufmann, 2004. DOI: 10.1016/B978-0-12-715051-2.X5012-1.

[Wan+23] Peng Wang, Yilei Ma, Siqi Xu, Yi-Xin Wang, Yu Zhang, Xiangyang Lou, Ming Li, Baolin Wu, Guimin Gao, Ping Yin, and Nianjun Liu. “MOVER-R and Penalized MOVER-R Confidence Intervals for the Ratio of Two Quantities.” In: *The American Statistician* (2023), pp. 1–9. DOI: 10.1080/00031305.2023.2173294.

[Wei+08] Xuetao Wei, Lei Guo, Xingwei Wang, Qingyang Song, and Lemin Li. “Availability guarantee in survivable WDM mesh networks: A time perspective.” In: *Information Sciences* 178.11 (2008), pp. 2406–2415. DOI: 10.1016/j.ins.2008.01.013.

[Win19] Windstream. *Complete Data MPLS; Service Level Agreement*. Oct. 2019. URL: https://www.windstreamenterprise.com/wp-content/uploads/2022/07/MPLS_SLA.pdf. Accessed: June 26, 2023.

[WN17] Peter J. Winzer and David T. Neilson. “From Scaling Disparities to Integrated Parallelism: A Decathlon for a Decade.” In: *Journal of Lightwave Technology* 35.5 (2017), pp. 1099–1115. DOI: 10.1109/JLT.2017.2662082.

[XCW09] Ming Xia, Joonho Choi, and Ting Wang. “Risk Assessment in SLA-Based WDM Backbone Networks.” In: *Optical Fiber Communication Conference (OFC)*. 2009. DOI: 10.1364/OFC.2009.OThQ6.

[Xia+08] Ming Xia, Marwan Batayneh, Lei Song, Charles U. Martel, and Biswanath Mukherjee. “SLA-Aware Provisioning for Revenue Maximization in Telecom Mesh Networks.” In: *IEEE Global Telecommunications Conference (GLOBECOM)*. 2008. DOI: 10.1109/GLOCOM.2008.ECP.266.

[Xia+09a] Ming Xia, Charles U. Martel, Massimo Tornatore, and Biswanath Mukherjee. “Service Cluster: A New Framework for SLA-Oriented Provisioning in WDM Mesh Networks.” In: *IEEE International Conference on Communications (ICC)*. 2009. DOI: 10.1109/ICC.2009.5199453.

[Xia+09b] Ming Xia, Massimo Tornatore, Charles U. Martel, and Biswanath Mukherjee. “Service-Centric Provisioning in WDM Backbone Networks for the Future Internet.” In: *Journal of Lightwave Technology* 27.12 (2009), pp. 1856–1865. DOI: 10.1109/JLT.2009.2021486.

[Xia+10a] Ming Xia, Charles U. Martel, Lei Shi, Massimo Tornatore, and Biswanath Mukherjee. “A Novel SLA for Time-Differentiated Resilience with Efficient Resource Sharing in WDM Networks.” In: *IEEE International Conference on Communications (ICC)*. 2010. DOI: 10.1109/ICC.2010.5502474.

[Xia+10b] Ming Xia, Massimo Tornatore, Charles Martel, and Biswanath Mukherjee. “Risk-Aware Routing for Optical Transport Networks.” In: *IEEE International Conference on Computer Communications (INFOCOM)*. 2010. DOI: 10.1109/INFCOM.2010.5462168.

[Xia+11a] Ming Xia, Massimo Tornatore, Charles U. Martel, and Biswanath Mukherjee. “Risk-Aware Provisioning for Optical WDM Mesh Networks.” In: *IEEE/ACM Transactions on Networking* 19.3 (2011), pp. 921–931. DOI: 10.1109/TNET.2010.2095037.

[Xia+11b] Ming Xia, Massimo Tornatore, Spencer Sevilla, Lei Shi, Charles U. Martel, and Biswanath Mukherjee. “A Novel SLA Framework for Time-Differentiated Resilience in Optical Mesh Networks.” In: *Journal of Optical Communications and Networking* 3.4 (2011), pp. 312–322. DOI: 10.1364/JOCN.3.000312.

[Xia08] XiPeng Xiao. *Technical, Commercial and Regulatory Challenges of QoS: An Internet Service Model Perspective*. Morgan Kaufmann, 2008. DOI: 10.1016/B978-0-12-373693-2.X0001-8.

[YR04] Wang Yao and Byrav Ramamurthy. “Survivable Traffic Grooming with Differentiated End-to-End Availability Guarantees in WDM Mesh Networks.” In: *IEEE Workshop on Local and Metropolitan Area Networks (LANMAN)*. 2004. DOI: 10.1109/LANMAN.2004.1338407.

[ZG05] Ling Zhou and Wayne D. Grover. “A Theory for Setting the ‘Safety Margin’ on Availability Guarantees in an SLA.” In: *International Workshop on Design of Reliable Communication Networks (DRCN)*. 2005. DOI: 10.1109/DRCN.2005.1563899.

[Zha+03a] Jing Zhang, Keyao Zhu, Biswanath Mukherjee, and Hui Zang. “Service Provisioning to Provide Per-Connection-Based Availability Guarantee in WDM Mesh Networks.” In: *Optical Fiber Communication Conference (OFC)*. 2003. DOI: 10.1109/OFC.2003.316143.

[Zha+03b] Jing Zhang, Keyao Zhu, Hui Zang, and Biswanath Mukherjee. “A New Provisioning Framework to Provide Availability-Guaranteed Service in WDM Mesh Networks.” In: *IEEE International Conference on Communications (ICC)*. 2003. DOI: 10.1109/ICC.2003.1204638.

[Zha+07] Jing Zhang, Keyao Zhu, Hui Zang, Norman S. Matloff, and Biswanath Mukherjee. “Availability-Aware Provisioning Strategies for Differentiated Protection Services in Wavelength-Convertible WDM Mesh Networks.” In: *IEEE/ACM Transactions on Networking* 15.5 (2007), pp. 1177–1190. DOI: 10.1109/TNET.2007.896232.

[Zho+16] Zhizhen Zhong, Nan Hua, Massimo Tornatore, Yao Li, Haijiao Liu, Chen Ma, Yanhe Li, Xiaoping Zheng, and Biswanath Mukherjee. “Energy Efficiency and Blocking Reduction for Tidal Traffic via Stateful Grooming in IP-Over-Optical Networks.” In: *Journal of Optical Communications and Networking* 8.3 (2016), pp. 175–189. DOI: 10.1364/JOCN.8.000175.

[ZM04] Jing Zhang and Biswanath Mukherjee. “A Review of Fault Management in WDM Mesh Networks: Basic Concepts and Research Challenges.” In: *IEEE Network* 18.2 (2004), pp. 41–48. DOI: 10.1109/MNET.2004.1276610.

[ZS15] Juzi Zhao and Suresh Subramaniam. “QoT- and SLA-aware Survivable Resource Allocation in Translucent Optical Networks.” In: *IEEE Global Communications Conference (GLOBECOM)*. 2015. DOI: 10.1109/GLOCOM.2015.7416967.

A

Appendix

A.1 Properties of the Cumulated Downtime per Billing Cycle

The cumulative distribution function (CDF) of the cumulated downtime per billing cycle is given by

$$F_{X_j}(x) = a(\Delta t_{\mathrm{s},j}) \cdot F_B(x;\lambda,\mu) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - F_B\big((t_{\mathrm{c},j} - x)^-;\mu,\lambda\big)\Big) \tag{A.1}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot F_B(x;\lambda,\mu) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - \lim_{y \nearrow (t_{\mathrm{c},j} - x)} F_B(y;\mu,\lambda)\Big). \tag{A.2}$$

The random variable (RV) X_j is a mixed RV with a point mass at $x = 0$ and a point mass at $x = t_{\mathrm{c},j}$. Therefore, we consider the value of $F_{X_j}(x)$ at the four points $x = 0^-$, 0, $t_{\mathrm{c},j}^-$, and $t_{\mathrm{c},j}$ in more detail.

For $x = 0^-$ we have

$$F_{X_j}(0^-) = \lim_{x \nearrow 0} F_{X_j}(x) \tag{A.3}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot \underbrace{\lim_{x \nearrow 0} F_B(x;\lambda,\mu)}_{=0} + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \underbrace{\Big(1 - \lim_{x \nearrow 0} \underbrace{\lim_{y \nearrow (t_{\mathrm{c},j} - x)} F_B(y;\mu,\lambda)}_{=1 \text{ for } x<0}\Big)}_{=1} \tag{A.4}$$

$$= 0. \tag{A.5}$$

In the double limit, the inner limit is 1 for all $x < 0$. The outer limit approaches 0 from below, i.e., $x < 0$. Therefore, the double limit resolves to 1.

For $x = 0$ we obtain

$$F_{X_j}(0) = a(\Delta t_{\mathrm{s},j}) \cdot F_B(0;\lambda,\mu) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - \underbrace{\lim_{y \nearrow t_{\mathrm{c},j}} F_B(y;\mu,\lambda)}_{=1}\Big) \tag{A.6}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot F_B(0;\lambda,\mu) \tag{A.7}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot e^{-\lambda t_{\mathrm{c},j}}. \tag{A.8}$$

For $x = t_{\mathrm{c},j}^-$ we get

$$F_{X_j}(t_{\mathrm{c},j}^-) = \lim_{x \nearrow t_{\mathrm{c},j}} F_{X_j}(x) \tag{A.9}$$

$$= a(\Delta t_{\mathrm{s},j}) \cdot \underbrace{\lim_{x \nearrow t_{\mathrm{c},j}} F_B(x;\lambda,\mu)}_{=1} + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - \underbrace{\lim_{x \nearrow t_{\mathrm{c},j}} \lim_{y \nearrow (t_{\mathrm{c},j} - x)} F_B(y;\mu,\lambda)}_{=F_B(0;\mu,\lambda)}\Big) \tag{A.10}$$

$$= a(\Delta t_{\mathrm{s},j}) + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \big(1 - e^{-\mu t_{\mathrm{c},j}}\big) \tag{A.11}$$

$$= 1 - \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot e^{-\mu t_{\mathrm{c},j}}. \tag{A.12}$$

In the double limit, the outer limit approaches $t_{\mathrm{c},j}$ from below, hence, $x < t_{\mathrm{c},j}$. In the inner limit, if $x < t_{\mathrm{c},j}$, then $t_{\mathrm{c},j} - x > 0$. Consequently, y approaches a point that moves toward zero from above, and, therefore, the double limit equals $F_B(0;\mu,\lambda)$.

Finally, for $x = t_{\mathrm{c},j}$ we have

$$F_{X_j}(t_{\mathrm{c},j}) = a(\Delta t_{\mathrm{s},j}) \cdot \underbrace{F_B(t_{\mathrm{c},j};\lambda,\mu)}_{=1} + \big(1 - a(\Delta t_{\mathrm{s},j})\big) \cdot \Big(1 - \underbrace{\lim_{y \nearrow 0} F_B(y;\mu,\lambda)}_{=0}\Big) \tag{A.13}$$

$$= 1. \tag{A.14}$$

A.2 Computing the Compliance Probability

This section explains how the compliance probability $\mathbf{P}(C_{\mathrm{AC}} \leq c_{\max})$ in (4.88) in Section 4.4.2.3 is computed. Let us reiterate the relevant parameters. The allowed compensation is $c_{\max} = 1 \cdot MRC$. Connection AC traverses two links in series. The links are assumed to be equal. Each link has a steady-state availability of $a_1 = 0.99995$ and a mean time to repair (MTTR) of $MTTR_1 = 1\,\mathrm{h}$. The contract period, Δt_{h}, is 12 months. The compensation policy is given in (4.86). It contains two steps, the first at an unavailability of $u_1 = 0.0001$, and the second at $u_2 = 0.001$. In the following, we sometimes omit the variable MRC for brevity, i.e., $c_{\max} = 1$ means $c_{\max} = 1 \cdot MRC$, for example.

The connection AC is a series system consisting of two links. In the first step, we reduce this system to an equivalent single-component system using the procedure described in Section 4.3. The failure rate of one link is

$$\lambda_1 = \frac{1 - a_1}{MTTR_1 \cdot a_1} \tag{A.15}$$

$$= 5 \cdot 10^{-5}\,\mathrm{h}^{-1}. \tag{A.16}$$

The combined failure rate of both links is the sum of the individual failure rates, i.e.,

$$\lambda = \lambda_1 + \lambda_1 \tag{A.17}$$

$$= 1 \cdot 10^{-4}\,\mathrm{h}^{-1}. \tag{A.18}$$

The combined steady-state availability is

$$a = a_1 \cdot a_1 \tag{A.19}$$
$$= 0.9999000025 \tag{A.20}$$

and the combined repair rate is

$$\mu = \lambda \cdot \frac{a}{1-a} \tag{A.21}$$
$$= 0.999975\,\mathrm{h}^{-1}. \tag{A.22}$$

The parameters λ and μ specify the behavior of connection AC. The values will be used in the following.

According to (4.84) and (4.83), the compliance probability we seek is given by

$$\mathbf{P}(C_{\mathrm{AC}} \leq 1) = \sum_{c \in \mathcal{C}_{12} : c \leq 1} \mathbf{P}(C_{1,12} = c) \tag{A.23}$$

where $C_{1,12}$ is the total compensation over all billing cycles in the contract period, i.e., cycle 1 through cycle 12. In this example, a billing cycle coincides with a calendar month, i.e., $t_{\mathrm{c},j} = 720\,\mathrm{h}$ for $1 \leq j \leq 12$. Therefore, we omit the index j and use $t_{\mathrm{c}} = 720\,\mathrm{h}$.

The probability mass function (PMF) of $C_{1,12}$ is computed iteratively from the PMFs of $C_{1,11}$, $C_{1,10}$, ..., $C_{1,1}$, and the PMFs of C_{12}, C_{11}, ..., C_2.

Let us consider the compensation of the first billing cycle, namely C_1. Since it is the first billing cycle, C_1 coincides with $C_{1,1}$, and $\Delta t_{\mathrm{s},1} = 0$. The compensation policy defines three levels of compensation, namely 0, $0.5 \cdot MRC$, and $1 \cdot MRC$. Hence, the support of C_1 is $\mathcal{C}_1 = \{0, 0.5, 1\}$, and the PMF of C_1 contains the probabilities $\mathbf{P}(C_1 = 0)$, $\mathbf{P}(C_1 = 0.5)$, and $\mathbf{P}(C_1 = 1)$. With (4.74) and $x_1 = u_1 \cdot t_{\mathrm{c}} = 0.072\,\mathrm{h}$, we have

$$\mathbf{P}(C_1 = 0) = F_{X_1}(x_1; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) \tag{A.24}$$
$$= a(\Delta t_{\mathrm{s},1}) \cdot F_B(x_1; t_{\mathrm{c}}, \lambda, \mu) + \big(1 - a(\Delta t_{\mathrm{s},1})\big) \cdot \Big(1 - F_B\big((t_{\mathrm{c}} - x_1)^-; t_{\mathrm{c}}, \mu, \lambda\big)\Big). \tag{A.25}$$

Similarly, with (4.69) and $x_2 = u_2 \cdot t_{\mathrm{c}} = 0.72\,\mathrm{h}$, we have

$$\mathbf{P}(C_1 = 0.5) = F_{X_1}(x_2; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) - F_{X_1}(x_1; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) \tag{A.26}$$

with

$$F_{X_1}(x_2; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) = a(\Delta t_{\mathrm{s},1}) \cdot F_B(x_2; t_{\mathrm{c}}, \lambda, \mu) + \big(1 - a(\Delta t_{\mathrm{s},1})\big) \cdot \Big(1 - F_B\big((t_{\mathrm{c}} - x_2)^-; t_{\mathrm{c}}, \mu, \lambda\big)\Big). \tag{A.27}$$

Lastly, we have

$$\mathbf{P}(C_1 = 1) = F_{X_1}(t_{\mathrm{c}}; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) - F_{X_1}(x_2; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) \tag{A.28}$$

with

$$F_{X_1}(t_{\mathrm{c}}; t_{\mathrm{c}}, \lambda, \mu, \Delta t_{\mathrm{s},1}) = a(\Delta t_{\mathrm{s},1}) \cdot F_B(t_{\mathrm{c}}; t_{\mathrm{c}}, \lambda, \mu) + \big(1 - a(\Delta t_{\mathrm{s},1})\big) \cdot \Big(1 - F_B\big(0^-; t_{\mathrm{c}}, \mu, \lambda\big)\Big). \tag{A.29}$$

Table A.1 Probability values of F_B

u	x $= t_c \cdot u$	$F_B(x; t_c, \lambda, \mu)$ $= \mathbf{P}(X_j \leq x \mid \xi_j = \mathbb{W})$	$F_B\big((t_c - x)^-; t_c, \mu, \lambda\big)$ $= \mathbf{P}(X_j > x \mid \xi_j = \mathbb{F})$
0.0001	0.072 h	0.9351942	0.9351925
0.001	0.72 h	0.9653443	0.5113977
1	720 h	1	0

As can be seen, the PMF of C_1 depends on the value of $a(\Delta t_{s,1})$, and the values of $F_B(x; t_c, \lambda, \mu)$ and $F_B\big((t_c - x)^-; t_c, \mu, \lambda\big)$ for $x \in \{x_1, x_2, t_c\}$. The values of F_B are listed in Table A.1.

Since all billing cycles have the same duration, t_c, the PMFs for the other billing cycles, i.e., for $C_2, C_3, \ldots, C_{12}$, are computed from the same values for F_B. The only value that changes with the billing cycle is the point availability at the cycle's beginning, $a(\Delta t_{s,j})$. For the first billing cycle, we have $a(\Delta t_{s,1}) = a(0) = 1$. The point availability converges quickly, and therefore, for all other billing cycles, the point availability equals the steady-state availability, i.e., $a(\Delta t_{s,j}) = a = 0.9999000025$ for $j > 1$. Columns three to five in Table A.2 show the resulting PMF values. As can be seen in the table, the PMFs $\mathbf{P}(C_2 = c)$ to $\mathbf{P}(C_{12} = c)$ all share the same values because they all share the same point availability. Only the PMF of C_1 is different (first row).

Table A.2 PMF values for different billing cycles – The variable *MRC* is omitted for brevity. Several values that are relevant in the text are highlighted.

Cycle i, j	$\Delta t_{s,j}$	$\mathbf{P}(C_j = c)$			$\mathbf{P}(C_{1,i} = c)$			$\mathbf{P}(C_{1,i} \leq 1)$
		$c = 0$	$c = 0.5$	$c = 1$	$c = 0$	$c = 0.5$	$c = 1$	
1	0 h	0.93519	0.03015	0.03466	**0.93519**	**0.03015**	**0.03466**	1
2	720 h	**0.93511**	**0.03019**	**0.0347**	0.87451	0.05643	**0.06577**	0.9967
3	1440 h	0.93511	0.03019	0.0347	0.81776	0.07917	0.09356	0.99048
4	2160 h	0.93511	0.03019	0.0347	0.76469	0.09872	0.11825	0.98166
5	2880 h	0.93511	0.03019	0.0347	0.71507	0.1154	0.1401	0.97056
6	3600 h	0.93511	0.03019	0.0347	0.66867	0.1295	0.1593	0.95746
7	4320 h	0.93511	0.03019	0.0347	0.62527	0.14128	0.17608	0.94263
8	5040 h	0.93511	0.03019	0.0347	0.5847	0.15099	0.19062	0.9263
9	5760 h	0.93511	0.03019	0.0347	0.54676	0.15884	0.2031	0.90869
10	6480 h	0.93511	0.03019	0.0347	0.51127	0.16504	0.21369	0.89
11	7200 h	0.93511	0.03019	0.0347	0.4781	0.16976	0.22255	0.87041
12	7920 h	0.93511	0.03019	0.0347	0.44707	0.17318	0.22982	**0.85007**

With the PMFs for the individual billing cycles, we can iteratively compute the PMFs for aggregated billing cycles, i.e., $\mathbf{P}(C_{1,2} = c)$ to $\mathbf{P}(C_{1,12} = c)$. The aggregated compensation of the first two cycles is $C_{1,2} = C_{1,1} + C_2$. As mentioned before, $C_{1,1}$ trivially equals C_1. Hence, $C_{1,1}$ can also be replaced with C_1 in the following. We consider how $\mathbf{P}(C_{1,2} = 1)$ is computed. $C_{1,2}$ equals 1 if $C_{1,1} = 1$ and $C_2 = 0$, or $C_{1,1} = 0$ and $C_2 = 1$, or $C_{1,1} = 0.5$ and $C_2 = 0.5$. In

terms of the probability, this is approximated in (4.80) as

$$\mathbf{P}(C_{1,2} = 1) \approx \sum_{(y,z)\in\mathcal{C}_1\times\mathcal{C}_1 : y+z=1} \mathbf{P}(C_{1,1} = y) \cdot \mathbf{P}(C_2 = z). \tag{A.30}$$

Expanding the summation, we obtain

$$\begin{aligned}\mathbf{P}(C_{1,2} = 1) \approx\ & \mathbf{P}(C_{1,1} = 1) \cdot \mathbf{P}(C_2 = 0) + \mathbf{P}(C_{1,1} = 0) \cdot \mathbf{P}(C_2 = 1) + \\ & \mathbf{P}(C_{1,1} = 0.5) \cdot \mathbf{P}(C_2 = 0.5).\end{aligned} \tag{A.31}$$

With the corresponding values highlighted in Table A.2, we finally get

$$\mathbf{P}(C_{1,2} = 1) \approx 0.03466 \cdot 0.93511 + 0.93519 \cdot 0.0347 + 0.03015 \cdot 0.03019 \tag{A.32}$$

$$\approx 0.06577. \tag{A.33}$$

The value of $\mathbf{P}(C_{1,2} = 1)$ is also highlighted in the table. All other PMF values are computed accordingly. The result is shown in columns six to eight in Table A.2. As can be seen, the probability that the aggregated compensation is zero, i.e., $\mathbf{P}(C_{1,i} = 0)$, falls when the number of aggregated billing cycles, i, grows. In contrast, the probabilities for higher compensation, i.e., $\mathbf{P}(C_{1,i} = 0.5)$ and $\mathbf{P}(C_{1,i} = 1)$, grow.

In the billing cycles two and above, the aggregated compensation can be larger than one monthly recurring charge (MRC). Consequently, the PMFs also have non-zero values $\mathbf{P}(C_{1,2} = 1.5)$, $\mathbf{P}(C_{1,2} = 2)$, $\mathbf{P}(C_{1,3} = 2.5)$, etc. However, since we are only interested in the compliance probability for $c_{\max} = 1 \cdot MRC$, the table omits those higher values. As a matter of fact, those values are not even computed in our algorithms because they are not needed.

The last column shows the compliance probability after billing cycle i for $c_{\max} = 1$. It is the sum of columns six to eight, i.e., $\mathbf{P}(C_{1,i} = 0) + \mathbf{P}(C_{1,i} = 0.5) + \mathbf{P}(C_{1,i} = 1)$. The bottom right value of 0.85007 is the final compliance probability of connection AC for the 12-month contract, $\mathbf{P}(C_{\mathrm{AC}} \leq 1)$. The compliance probability for the first billing cycle, $\mathbf{P}(C_{1,1} \leq 1)$, is 1 since the maximum compensation the compensation policy stipulates is 1. Only when several billing cycles are aggregated, the compensation can exceed 1, and hence, the compliance probability for $i > 1$ is less than 1. Since every additional billing cycle potentially adds compensation, the compliance probability falls when the number of aggregated billing cycles grows.

A.3 Confidence Intervals for Derived Quantities

Most statistics recorded in our studies are point estimates. In order to assess the uncertainty of those point estimates, the simulation environment automatically computes confidence intervals (CIs) as well. As mentioned in Section 5.1.3.6, the methodology is based on Student's t-distribution and explained in [CL21, Sec. 10.7.2]. However, we often compare the results of two different simulations as differences or ratios, e.g., the increase in carried load using compensation-aware provisioning with surplus sharing (CAPSS) and a reference mechanism. In order to provide CIs for those derived quantities as well, we employ the MOVER-D method for differences and the MOVER-R method for ratios [Wan+23]. In both cases, we consider the two point estimates θ_1 and θ_2. The estimate θ_1 corresponds to the recorded mean of a certain statistic in simulation 1, and θ_2 corresponds to the recorded mean of the same statistic in simulation 2. Both estimates have symmetric CIs $[\theta_1 - i_1, \theta_1 + i_1]$ $(i_1 \geq 0)$ and $[\theta_2 - i_2, \theta_2 + i_2]$ $(i_2 \geq 0)$, respectively.

A.3.1 MOVER-D for Differences

The quantities we are interested in are the lower and upper confidence limit (CL) l_d and u_d for the difference $d = \theta_1 - \theta_2$. Since the CIs of θ_1 and θ_2 are symmetric, and we assume that there is no correlation between the two means, the equations in [Wan+23] simplify to

$$l_d = d - \sqrt{i_1^2 + i_2^2} \tag{A.34}$$

and

$$u_d = d + \sqrt{i_1^2 + i_2^2}. \tag{A.35}$$

As can be seen, the CI of d is again symmetric.

A.3.2 MOVER-R for Ratios

This time, we consider the relative difference

$$r = \frac{\theta_1 - \theta_2}{\theta_2} \tag{A.36}$$

which boils down to

$$r = \frac{\theta_1}{\theta_2} - 1 \tag{A.37}$$

$$= f - 1 \tag{A.38}$$

with $f \equiv \theta_1/\theta_2$. We are interested in the CI of r, namely $[l_r, u_r]$. With (A.38) and Appendix A.3.1, this is equivalent to the CI of f shifted by 1, i.e., $[l_r, u_r] = [l_f - 1, u_f - 1]$.

Since f is a ratio, we can employ the MOVER-R method to compute the CLs l_f and u_f. The method covers a variety of value ranges. However, in our scenarios, it is sufficient to consider the cases in which $\theta_1 > i_1$ and $\theta_2 > i_2$. Furthermore, we assume that θ_1 and θ_2 are uncorrelated. With these preconditions, the equations in [Wan+23] simplify to

$$l_f = \frac{-b - \sqrt{\delta}}{2a} \tag{A.39}$$

and

$$u_f = \frac{-b + \sqrt{\delta}}{2a} \tag{A.40}$$

where

$$a = \theta_2^2 - i_2^2 \tag{A.41}$$

$$b = -2\theta_1\theta_2 \tag{A.42}$$

$$\delta = b^2 - 4ac \tag{A.43}$$

$$c = \theta_1^2 - i_1^2. \tag{A.44}$$

Consequently, the CLs for r are

$$l_r = \frac{-b - \sqrt{\delta}}{2a} - 1 \tag{A.45}$$

and

$$u_r = \frac{-b + \sqrt{\delta}}{2a} - 1. \tag{A.46}$$

The CI is typically asymmetric.

A.4 Synthetic Network Topologies

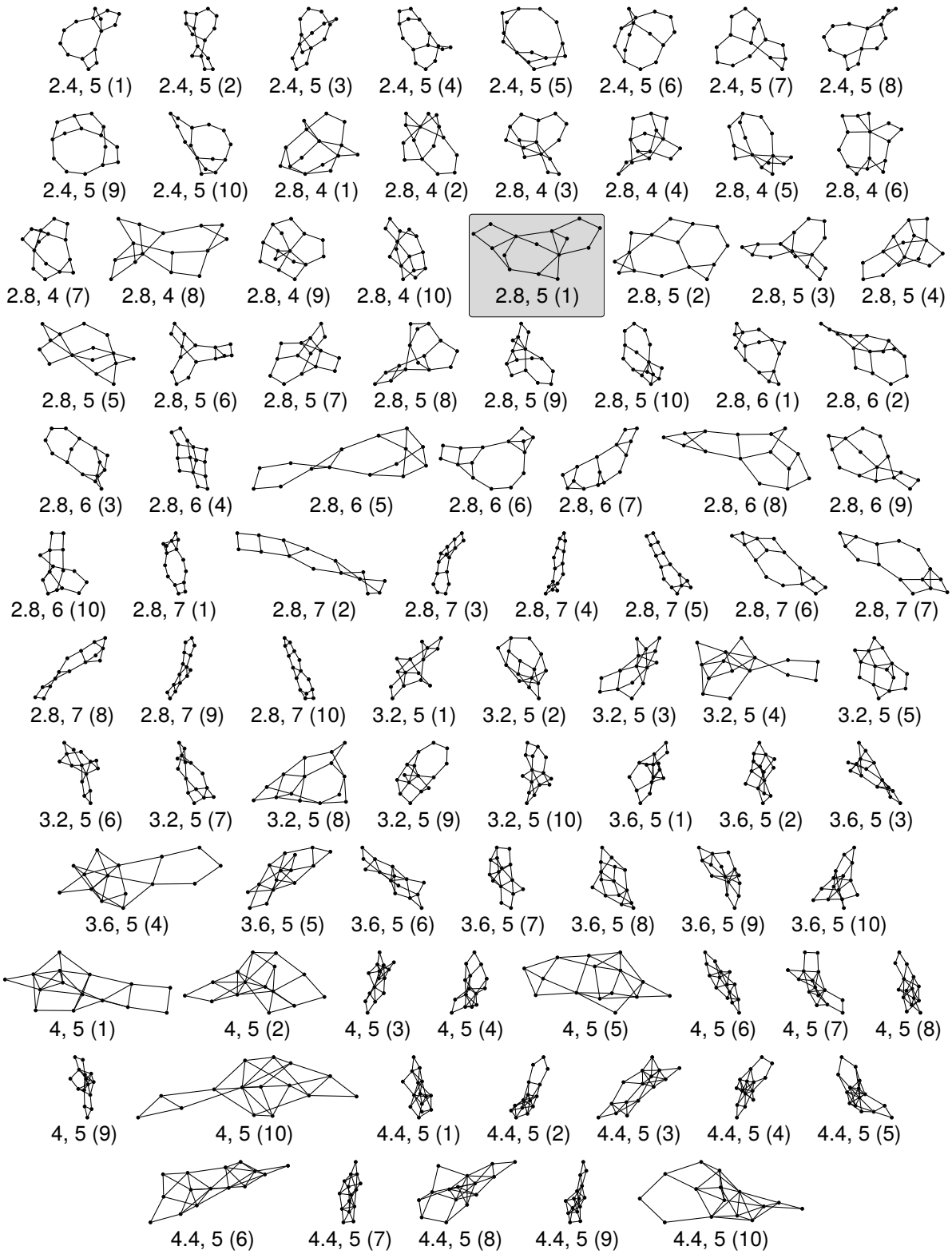


Figure A.1 Synthetic network topologies – The labels are of the form ‹average node degree›, ‹diameter in hops› (‹instance›). All topologies are scaled to the same height in this visualization. The BASE topology is highlighted.