Modeling and Performance Analysis of Networks with Flow Transformations

Vom Fachbereich Informatik
der Technischen Universität Kaiserslautern
zur Verleihung des akademischen Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
genehmigte

Dissertation

von

Hao Wang

Datum der wissenschaftlichen Aussprache: 24.07.2015

Dekan: Prof. Dr. Klaus Schneider
Prüfungskommission:
Vorsitz: Prof. Dr. Christoph Garth
Erster Berichterstatter: Prof. Dr. Jens Schmitt
Zweiter Berichterstatter: Prof. Dr. Florin Ciucu

D 386

Abstract

Data flows in modern networks have exhibited increasingly complex prospect along with the evolving network infrastructures and emerging diverse applications. Performance evaluation must adequately consider this trend. The tractable queueing network as a traditional theoretical tool for performance evaluation can not satisfy the new requirement anymore. One reason is that the important assumption of queueing theory, i.e. Poisson arrivals, does not further hold accurate for many networks, within which the flows are more bursty, self-similar, or long-range dependent. Many alternative methodologies to the classical queueing theory appeared. Network calculus, over two decades after Cruz's pioneer work in 1991, has established itself as one promising theoretical tool for assessing this kind of networks. It can deal with problems that are fundamentally hard for queueing theory based on the fact that it works with (probabilistic) bounds rather than striving for exact solutions.

The complexity of network flows attributes itself not only to the upgrading infrastructure and enlarging number of users but also to the operations onto the flows caused by various algorithms, protocols, services, and even network topologies. The flows might be altered due to the operations along the path from source to destination, e.g., in a lossy network, being transcoded, randomly routed, or somehow processed for certain purpose like improving energy efficiency. This is called flow transformation. It is a great challenge for the existing queueing methodologies, including network calculus. One reason is, the basic assumption that the system is lossless when defining the performance metric delay, does not hold anymore. The other is, usually these flow transformations are random. This thesis addresses an extension of the network calculus to deal with the random flow transformations and provides the performance evaluation.

The thesis mainly comprises of three inter-connected parts. The first is to develop the so-called stochastic data scaling elements to model the flow

transformation and derive the performance bounds. The main technique is, by using the instrumental equivalent systems theory, which enables transforming the service to the part that only the transformed fraction of the flow may receive, we can reorder the series of the service and scaling elements and thus guarantee a convolution-form service to the flow of interest. Accordingly, the delay bounds scale in $\mathcal{O}(n)$, where n is the number of nodes the flow traverses. The second is to deepen the understanding of the models with a deconstruction perspective. We investigate a very important case of flow transformation - demultiplexing, and conclude that under the FIFO scheduling assumption the stochastic scaling effect allows a conversion from demultiplexing of the flow to the virtual multiplexing of the subflows. The third is to apply the models to two very important scenarios, unreliable links with retransmissions and flows of variable length packets, in order to validate the models and widen the scope of their applications.

Zusammenfassung

Datenflüsse in modernen Netzwerken werden aufgrund der sich stets weiter entwickelnden Infrastrukturen und der Vielzahl an neuen Anwendungen immer komplexer. Dadurch wird auch die Leistungsbewertung solcher Netze immer schwieriger und muss sich anpassen. Das klassische Modell der Leistungsbewertung , das Warteschlangennetzwerk, kann diesen Anforderungen nicht mehr genügen. Ein Grund dafür ist, dass die essentielle Annahme der Poisson-verteilten Ankünfte für viele Netzwerke nicht mehr genau genug ist, da Verkehr oftmals stoßweise auftritt und Eigenschaften wie Selbstähnlichkeit oder Langzeitkorrelationen besitzt. Aus diesem Grund haben sich mehrere Alternativen zu der klassischen Warteschlangentheorie entwickelt. Das Netzwerkkalkül ist eines davon und hat sich seit Cruz's Pionierarbeit von 1991 als vielversprechendes Werkzeug zur Leistungsbewertung solcher Netzwerke etabliert. Durch das Verwenden von probabilistischen Schranken kann das Netzwerkkalkül mit Netzwerken umgehen, bei denen die klassische Warteschlangetheorie, die nach exakten Lösungen sucht, fundamentale Probleme hat.

Die Komplexität der Datenflüsse kann aber nicht nur auf die Infrastruktur und die Anwendungen zurückgeführt werden. Auch die Art der Operationen auf dem Netzwerkverkehr durch verschiedene Algorithmen, Protokolle, Dienste und Topologien tragen einen erheblichen Teil dazu bei, da sie Flüsse auf ihrem Weg durch das Netzwerk verändern. So können Daten verloren, zufällig umgeleitet oder für bestimmte Anwendungen aufbereitet werden. Diesen Vorgang nennt man Flusstransformation und er stellt eine große Herausforderung für existierende Analysemethodiken (auch für das Netzwerkkalkül) dar. Ein Grund dafür ist, die Grundannahme (verlustfreies System) für die Definition der Leistungsmetrik, Ver- zögerung, ist nicht mehr gehalten. Der andere ist, in der Regel sind diese Flusstransformationen zufällig. Diese Dissertation befasst sich mit einer Erweiterung des Netzwerkkalküls, um genau diese zufälligen Flusstransformationen zu bewältigen, und analy-

siert die Verzögerung sowie weitere Leistungsmetriken.

Die Arbeit besteht im Wesentlichen aus drei miteinander verbundenen Teilen. Im ersten Teil wird das Netzwerkkalkül um sogenannte stochastische Datenskalierungselemente erweitert, um die zufälligen Flusstransformationen zu modellieren und Leistungsschranken abzuleiten. Die schlüsseltechnique ist die Umwandlung von quivalenten Systemen. Damit sind die Dienste zu den Teil, den die transformierte Flüssenteilen bekommen, auch transformiert. Dementsprechend kann die Reihenfolge der Dienste und Skalierungselemente geändert werden ohne dass das Netzwerk seine Faltungsform (d.h. die Ende-zu-Ende-Modellierbarkeit von Diensten der Flüsse durch die $(min, +)$-Faltung) verliert. Diese ist wichtig, damit die Schranken für Verzögerungen in $\mathcal{O}(n)$ skalieren, wobei n die Anzahl der Knoten ist, die der Fluss durchquert. Im zweiten Teil dieser Arbeit werden die Modelle mit einer Dekonstruktionsperspektive vertieft. Dazu wird der wichtige Transformationsfall Demultiplexing untersucht, wobei wir zu dem Ergebnis kommen, dass unter der Annahme einer FIFO-Strategie, der stochastische Skalierungseffekt eine Umwandlung vom Demultiplexing eines Flusses zum virtuellen Multiplexing von Unterflüssen erlaubt. Im dritten Teil der Arbeit werden die Modelle auf zwei relevante Szenarien angewandt: unzuverlässige Verbindungen mit erneuten Übertragungen und Flüsse mit variabler Paketlänge. Durch diese Szenarien werden die entwickelten Modelle validiert und ihr erweitertes Anwendungsgebiet demonstriert.

Acknowledgements

On the completion of my dissertation, I would like to express my deepest gratitude to all those whose kindness and advice have made this work possible.

First and foremost, I would like to give the special thanks to my supervisor, Prof. Dr. Jens B. Schmitt, for his continuous encouragement, great patience, excellent guidance, kind personality, and intelligent research skills. His help covers every aspect beyond the research. It is impossible to finish this dissertation without these helps. To be supervised by and work with him is a great thing. I feel fortunate and happy to be his student, not only for the master and PhD study but also for my whole life. I would like to give my equivalently special thanks to another person who will have influence on my whole career, my second supervisor, Prof. Dr. Florin Ciucu. His outstanding research skills and elaborate research attitude influence me a lot. I also gratefully thank his special encouragement and strict guidance. Without these, this dissertation would not be finished. The contribution of my supervisors to this dissertation are also very straightforward, as the major results of it are the product of our collaboration.

I would like to extend my gratitude to the chair of my defense committee, Prof. Dr. Christoph Garth and the remaining members of my defense committee, whose thoughtful insights and suggestions invaluably improve this dissertation. I feel honored to have them on my committee.

My sincere thanks also go to my dear past and present colleagues of DISCO lab: Adam Bachorek, Michael Beck, Daniel Berger, Steffen Bondorf, Babara Erlewein, Nicos Gollan, Ivan and Petra Martinovic, Wint Yi Poe, Matthias Schäfer, Steffen Reithermann, and Matthias Wilhelm. The atmosphere of doing research at DISCO is fantastic. I will forever remember our friendship and their help. I would like to express my special acknowledgment to Steffen Reithermann. Without his tremendous help I may not continue my research career in Germany.

I am also deeply grateful to TU Kaiserslautern, DFG, and AMSYS for their financial support to this work.

Finally, I owe my loving thanks to my mom, dad, wife, and son. Their unconditional support and care continuously encourage me to finish this dissertation. I dedicate this dissertation to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over thousands of years, people never stopped the steps in pursuit of higher performance when transmitting information or goods. Although inventing new form or raising the capacity of communication medias can improve the performance, people often face the same situation - resource sharing, whether the beacon fires were sharing the tower along the Great Wall, or the data are sharing a server in today's Internet. What people do is, queueing. The ways of sharing and queueing are always crucial consideration for the performance when people design, establish, and run a communication system. One indispensable mathematical framework to analyze such resource sharing systems with queues is the *queueing theory*. After Erlang's foundation work [59, 60] in telephone networks, queueing theory has been developed over a century. It built up the basic models of analyzing many performance metrics and provided insights into many practical applications, among which a very important one is for data networks [15]. Based on Jackson's network [77], Kleinrock conducted the first studies of queueing theory onto data networks in the 1960s [89] and established the theoretical foundation of the Internet. Along that, the theory has also been generalized from the simple single-queue model with Poisson arrivals/exponential service to a much more complex network of queues model with various service time distributions, scheduling, topologies and routing. The main property of this model framework is the product form solution of the sojourn time/joint queue length distribution.

Despite the remarkable evolution, the product form queueing network models have been consistently restricted to the fundamental assumptions like Poisson traffic model or Markovian routing. For example modeling Internet as a queueing system, the traffics sourced from different layers can be

characterized by a high variability of models, more bursty, not only Poisson [117]. To accurately predict the performance under these traffic assumptions, some novel approximate theories were conceived, mostly over the last two decades. One of them is the *network calculus*. Cruz's has pioneered this research through conceiving a deterministic network calculus in his seminal work [50, 51]. Since then, network calculus has become a promising theory for performance analysis of queueing systems and been receiving more and more significant research attention. However, network calculus has its own limitations, in particular when the traffic flows in the networks are altered at some intermediate nodes along its transmission path related to the network topologies, routing, the fluctuating system circumstances, the interactions among traffic flows, or even certain predefined control scheme. We name this traffic flow altering behavior a *flow transformation*. And we want to understand the performance of the system, from the viewpoint basically on the original flows only knowing the information about the altered ones or vice versa, towards a more versatile analytical framework for the whole network with large number of flows. These represent the basic motivation of this thesis.

In the rest of this chapter, we first provide a brief overview of the queueing theories, especially for data networks and of the emphasis on the network calculus context. Then we provide the overview of the network calculus by comparison and show the challenge when analyzing the networks with flow transformations. Finally we present the main contribution and the structure of this thesis.

1.1 Network Performance Modeling and Analysis

1.1.1 Related Queueing Theories

Among the classical queueing theories, the class of the tractable queueing network models has been extensively applied to efficiently represent the resource sharing systems and accurately evaluate their performance. A queueing network model is a collection of services with queues that provide service to a collection of customers, whereby under certain assumptions, the stationary joint queue length distribution can be expressed by the product of the individual queue length distributions. This product form expression has played a fundamental role in constructing the queueing network models. It yields various interesting properties include the insensitivity [11], arrival theorem

[93, 134], exact aggregation [83], and allows to develop computationally efficient algorithms for analyzing performance measures such as the convolution algorithm [30] and the mean value analysis [119]. Insensitivity states that the stationary queue length distribution depends on the service time distributions only through the average, which enables us to only estimate the average of the resource parameters. Exact aggregation supports us to do hierarchical system analysis because we can aggregate parallel services into one single service such that they behave the same in terms of certain set of performance indices. Arrival theorem states that the stationary queue length distribution seen at arrival times is the same as at arbitrary times. This facilitates the computational transform from complex equations into convenient ones and leads to the mean value analysis. The two algorithms have also promoted applying the product form networks (especially in the 1970-80s), which not only could be mathematically expressed in some convenient and possibly implicit form, but more importantly could be numerically tractable.

There are three landmarks in the evolution of the product form networks - Jackson's network [77], Gordon and Newell (GN) networks [70], and Baskett, Chandy, Muntz and Palacios (BCMP) [11] networks. Jackson's theorem for the networks of queues states that under Kleinrock's independence assumption [89], i.e., the interarrival time of packets and their sizes are independent, and some further assumptions, the stationary distribution of the overall queue states in the network can be expressed as the production of the single queues. This is the origin of the term *product form queueing networks*. Knowing the queue distribution, from the Little's law [104] we also know the delay distribution. Besides these fundamental results, we classify further results that Jackson networks cover open exponential networks, i.e., with Poisson arrivals and exponential service times, first-in-first-out (FIFO) scheduling, and arbitrary Markovian routing. The GN extension covers closed networks, whereas the much more substantial BCMP extension covers a mixture of open and closed network scenarios, multiple classes of customers, and various service time distributions. Further extensions cover state-dependent routing [84, 140, 22], finite capacity queues and blocking [4, 10], batch arrivals and service [73], or positive and negative arrivals [65].

Jackson network theories promote the adoption of the packets of bits instead of circuit as the switching objects when designing a communication networks. The information are thus carried by the bits and transported over the links within such data communication networks, where the links are usually imperfect, i.e., they can delay, lose, or modify the information they transport. All these theories also lay the theoretical foundation of various mechanisms to share the imperfect bit carrier resources and capacities of the data commu-

nication networks, and further provide to the network applications and users a certain Quality of Service (QoS) they expect.

However, the Poisson assumption of the traffics restricts the application of the queueing network theory. That became especially problematic with the appearance of the high speed data communication networks, e.g., Internet, and triggered an intense debate on the relevance of the queueing theory when applying it to those networks. In the 1990s many publications studied the network traffics and convincingly concluded that the properties of many network traffics fundamentally differ from the Poisson traffic ([97]). The traffics show more burstiness, self-similarity, or long-range dependency. In order to avoid misleading results many development of new arrival models and analytical tools for queueing analysis were introduced. Markov modulated process models have been proposed for voice and video sources ([56, 72, 106]). Fractional Brownian motion (fBm) [113] or stable Lévy process [112] have been proposed for the Internet traffics. The effective bandwidth theory ([76, 66, 71, 86]) has been proposed in order to provide a measure of resources usage, adequately represent their trade-off, and account for their very generally varying statistical characteristics and the QoS requirements. These models used a lot of mathematical techniques such as large-derivation [57], or extremal properties of Gaussian processes [108]. These significant extension of the tractable queueing systems has however come at the price of loosing mathematical exactness. For instance, the results of the effective bandwidth are mostly large buffer or many sources asymptotic, which may be misleading for finite regimes ([21, 38, 136, 1]), moreover, the results are mostly limited to single node scenarios.

1.1.2 Network Calculus

Network calculus (NC) is a new theory for queueing analysis, which can circumvent the inherent difficulties of non-Poisson arrivals, and provide a well-founded design of future networks , protocols, and mechanisms. It was conceived by Cruz in the early 1990s [50, 51] in a deterministic framework, and soon after by Chang [32] in a probabilistic framework. Subsequently, a significant number of researchers have contributed to both the deterministic and stochastic formulations of the network calculus (see the books of Chang [36], Le Boudec and Thiran [95], and Jiang and Liu [82]).

Instead of using average and asymptotic values, like in product form network model used performance indices, the key innovation of the network calculus stands in the modeling of arrivals and services in terms of bounds and the relaxation, i.e., bounds with probability. They are known as *ar-*

rival curve and *service curve*. The consequences of these representations are accordingly the worst-case and stochastic performance bounds. We denote these two branches as *deterministic network calculus* (DNC) and *stochastic network calculus* (SNC). They provide comprehensive insights and many interesting results for the analysis of queueing network systems. A major result of network calculus is the model of the service curve (process) [115, 116, 52, 54, 122, 2, 35, 94, 3] and the expression of the multi-node network as a single-node case [95, 42]. Through using $(min, +)$ algebra this elegant reduction is easy to obtain. Given the arrival and departure processes $A(t)$ and $D(t)$, the service curve process $S(s,t)$ satisfies

$$D(t) \geq A \otimes S(t) \ ,$$

where $\otimes$ denotes the $(min, +)$ convolution defined for bivariate functions $f(s,t), g(s,t)$ as $f \otimes g(t) = \inf_{0 \leq s \leq t} \{f(0,s) + g(s,t)\}$. The n-nodes network service curve consequently has a single-node representation

$$S_{net} = S_1 \otimes S_2 \otimes \cdots \otimes S_n \ ,$$

where $S_i(s,t)$ is the i-th node along the traversing path of the flow. Hence the networks with their service curves expressed by this $(min, +)$ convolution form is referred to as *convolution-form networks*. This guarantees several further results when analyzing the performance, like pay burst only once (PBOO) [95, 126], pay multiplexing only once (PMOO) [131, 130], and yields the exact scaling of queueing measures [26]. When some arrival burst comes to a sequence of service nodes, the former nodes usually regulate the burst such that the latter nodes will not experience it again. Node by node performance analysis redundantly accounts for the burst at each node, while using S_{net} avoids that and can pay burst only once. This is our main motivation to assure the convolution form expression of networks in this thesis.

The arrival and service curves have shown great modeling power. The (deterministic and stochastic) arrival curve model can capture a broad class of traffics, such as Markovian arrival processes [36], deterministically regulated [98], fractional Brownian motion [120, 121], and the heavy-tailed and self-similar [99]. The service curve models different capacity supply patterns [95, 36, 82] and also various scheduling schemes (static priority (SP), earliest deadline first (EDF), generalized processor sharing (GPS), first-in-first-out (FIFO) [61, 98, 44, 101, 46]), even a generalized formulation [102].

The high modeling power of the network calculus has been transposed into several important applications for network engineering problems: traditionally in the Internet's Quality of Service proposals IntServ and DiffServ,

and more recently in diverse environments such as wireless sensor networks [91, 127, 132, 133, 68, 17], switched Ethernets [137], network coding [158], power grid [153], software defined networks [9], Systems-on-Chip (SoC) [31], or even to speed-up simulations [88].

In order to support the queueing network analysis, a number of software tools exist that focus on certain aspects and implement related functionalities. The DISCO network calculators [129, 69, 18, 14] developed in Java are well-covered tools for both DNC and SNC. Most of the others are developed for DNC. The CyNC toolbox [124, 123] is for the MATLAB/SimuLink environment, the RTC toolbox [54] in Matlab is for the real-time system analysis, the COINC toolbox [144] in C++, and the related commercial SymTA/S toolbox [74] provide the similar functionality.

In this thesis our main attention is the stochastic network calculus, whereas we will not completely ignore its deterministic counterpart, because the results of the stochastic network calculus can carry over from the deterministic calculus [28] using statistical characterizations of arrivals and services and sample path arguments. An inherent concern of the stochastic network calculus lies in the statistical dependency. Consequently, the stochastic network calculus can be applied to network scenarios with complete statistical independence of arrivals and service [36, 61], no independence assumptions of arrivals and service across multiple network nodes [43], or even a mixture of the two scenarios [41].

Recently, some other effort and attention are put to for example, improve the accuracy of the analytical results [39, 48, 47], widen the scope of the applications [5, 45, 105, 160], and improve the theory flexibility [13, 12]. This thesis also relates to these aspects.

1.2 Flow Transformations

An underlying assumption of the convolution-form expression of network services is that the flows are transported unaltered over the networks, from the perspective of the traversing nodes, the service should represent a lossless system. However this can not always hold. How can we evaluate the performance of the system on the transformed flows? In this thesis we study this problem in the context of (stochastic) network calculus and try to provide a general and insightful model. But actually, this challenging problem was firstly studied in queueing theory. As an important case of flow transformation, data loss, has been modeled by the seminal work of Erlang [25]. This work models the behaviour of a single telephone link. Later Kelly general-

ized this study to the loss network [85]. In a loss network, a data arriving at one site might be lost (rejected), if there is no buffer on this site and all the servers on it are busy. Although this can model many source sharing systems like telephone networks, ATM networks, broadband telecommunication networks and many other networks, the cause of loss seems to be not well suitable to many today's cases, even simply for a call center, nowadays we may more possibly wait online listening to some music instead of being rejected directly. Furthermore, assume that we set buffers in the loss network, the overflow of buffer is, however, only one possible reason causing data loss. There are many other reasons that cause data loss, like unreliable link or certain control scheme. Although there have also been some other works on modeling the lossy channel, like Markov model ([67, 58, 152]), data loss is just one case of flow transformation. Many modern networked and distributed systems transform the flows for specific purposes or inherently in different ways, e.g., a wireless sensor network which transforms the transported data while delivering it to a sink node for energy-efficiency purposes, dynamic routing, load balancing, P2P content distribution systems, media transcoding, network coding, or distributed real-time systems. We should design a model to generally express all these flow transformation behaviours. We can do it both in product-form network based on queueing theory and in convolution-form network based network calculus. In this thesis we focus on the latter. Now the further question is: can the networks with flow transformations still be expressed in convolution-form (for the purpose of PBOO)?

A novel extension of network calculus to deal with flow transformations is introduced in purely deterministic settings [64] - the so-called data scaling elements. It is simply defined as a wide-sense increasing function that scales some amount of data to some other amount. Then by controlling the movement of these elements in the network, the exact scaling properties of the analytical performance bounds derived from the convolution-form representation are preserved. Although widening the scope of the network calculus, this approach has limitations. An obvious one is that the deterministic modeling can very loosely capture the behavior of networks with stochastic settings. One must accordingly resort to stochastic modeling techniques.

Other related attempts to deal with flow transformations have been proposed. Chang introduced an element called router, which has a single data input, a control input and an output [36]. The control input determines which packets appear at the output. Cruz introduced an element called clipper [55], which regulate the traffic according to required output specifications. Maxiaguine *et. al.* introduced the so-called workload curves [109] into the real-time calculus to translate an event stream into specific requirements for a

certain resource. Jiang [154] attempted to use the entropy function to extract information from the flows within the so-called information-driven networks. There are also other interesting related works, packet curve for bounding packetization [23], control function in software defined networks [9], lossy server models [6, 81]. All these efforts that may be, or not, aware of their modeling capacity for the flow transformation, have even bigger limitations than the deterministic data scaling element - losing generality, ignoring the convolution-form representation, or being only deterministic (we discuss the details in Section 2.4).

1.3 Thesis Contribution

In spite of the previous fruitful attempts in the direction of modeling flow transformations, the stochastic network calculus is still a great lack of the wide-applicable model. This thesis contributes by introducing two forms of the *stochastic scaling element*, in the framework of the stochastic network calculus, to model the flow transformations in great generality. The new scaling elements are carefully defined to achieve (1) convolution-form network representations, and (2) a flexible means of capturing actual random transformation processes inside a network. As a consequence, the former allows us to derive competitive performance bounds, in particular, by preserving the elegant scaling properties brought by the convolution-form expression and the latter opens up the modeling scope widely.

First, we capture the stochastic behaviour of the flow transformation in two forms: one is to define bounds (the stochastic scaling curves) such that the sample paths (the scaling functions, which transform the data) are bounded in probability; the other is to straightforward define the scaling operation as a stochastic process that has effects on each data in the flow. Moreover, the scaling elements own several useful algebraic properties, of which the most important is that both forms facilitate the reordering of a series of service and scaling elements, and then enable the system to transform from a multi-node viewpoint into the convolution-form, i.e., the single-node viewpoint. Accordingly, we pay burst only once and thus the performance bounds generally scale in the order of the number of nodes.

Second, we review the definitions with a deconstruction viewpoint through investigating an important case of the flow transformation, the demultiplexing. We find out that under appropriate assumptions, e.g., FIFO scheduling, we can move the scaling effect for a served flow to the correspondent arrival flow and thus dismiss the scaling elements between the adjacent servers.

Third, we validate our theories by applying them to two hard, but very important scenarios. One is to model the impairment of unreliable links with stochastic scaling element and further describe the retransmission-based loss recovery schemes as a fixed-point problem then solve it. The other is to model the transformation of flows that consist of variable length packets, which usually applies to most of the networks.

1.4 Thesis Organization

We organize the rest of the thesis as follows.

In Chapter 2, we recall the background of the network calculus, with most of the emphasis on those contents, which can support the coming chapters. We structure this chapter according to the basic research branches of network calculus, i.e., the deterministic network calculus and the stochastic network calculus. As a motivation, we specifically introduce the state-of-the-art of the flow transformation modeling techniques.

In Chapter 3, we extend the data scaling element in the direction of stochastic sample path bounding by defining the stochastic scaling curves. We use the sample path argument such that the results of the deterministic network calculus can relatively easily carry over to these stochastic settings. An important theoretical result enables the reordering of a sequence of servers and scalers and accordingly the expression of the network in convolution-form. The reordering consists two directions, one is towards the ingress, the other is towards the egress. Then we derive end-to-end performance bounds using both analytical results and compare them with other analyses for the cases like deterministic or ideal.

In Chapter 4, we introduce a different model of the stochastic data scaling element. This definition is closely assembled with the flow description together. We focus more on the scaled flows. By doing that, the utilization of the statistical scaling characteristics is more flexible and effective. Different from the previous model based on the sample path bounding curves, this model assumes to know more information about the scaling process itself and thus derives more accurate performance bounds. The theories also guarantee the commutation of the servers and the scalers, such that we still preserve the convolution-form expression of the networks. To validate the new model, we compare the analytical results with the node-by-node analysis.

In Chapter 5, we review the previous two models and provide a deconstruction of both definitions. We illustrate the viewpoint by analyzing the important case of the flow transformation - demultiplexing. We show that

the scaling effect that is used to model the demultiplexing operation, e.g., dynamic load balancing, can be moved to the arrival side as multiplexing. The meaning is, under FIFO scheduling assumption, we can do the demultiplexing a little bit earlier, i.e., before being served. We then use the leftover service for one of the later demultiplexed subflow and iteratively apply this translation for a larger network scenario. We compare the results with those from the previous chapter in the numerical examples.

In Chapter 6, we apply the stochastic data scaling viewpoint and the stochastic scaling curves definition from respectively Chapter 4 and 3 to describe the loss process in the unreliable links and further the whole link model with retransmission-based loss recovery. Since retransmitted lost data can be lost again, we face with a difficult fixed-point problem. To solve this, we first calculate the arrival curves of the retransmitted flows with sample path argument, and then we apply the probabilities of each violation of the sample path bounds to derive the stochastic performance bounds. To ease the exposition we assume that the times of retransmission is limited. The numerical results indicate that the retransmission number dominates the delay bounds, which can be used as a reference in practice, i.e., to limit the times of retransmissions.

In Chapter 7, we extend the stochastic data scaling element in variable length data granularities, e.g., for the flows with variable length packets. To preserve the convolution-form representation, we need to note the concatenated scaling elements, because for this case the scaled flows are very hard to express. At the same time the commutation of the servers and the scalers as well as the derivation of the end-to-end delay bounds are also influenced by the varying packet sizes. The most difficult is that we can not simply ignore the inherent dependency of the service times brought by the unnormalized packet sizes. In this chapter, we address these problems and avoid the dependency of the service times by limiting the packet sizes. As a comparison, we adapt the results of Chapter 4 in the way we normalize the service capacity by the packet sizes. The numerical results show the clear superiority of the former one and also validate the bounds by simulation results.

In Chapter 8, we conclude this thesis and discuss the future work.

Chapter 2

Background on Network Calculus

In this chapter, we introduce the fundamental results of the network calculus that this thesis is based on and develops. We first describe the network models considered in the network calculus and this thesis. Next, we review the necessary concepts and main results of the deterministic network calculus respectively stochastic network calculus , which model the network elements and provide insights of the system behavior, especially on the performance. Then we discuss how to develop the network calculus to model the flow transformation and some previous related work.

2.1 Network Model

The networks we study in this thesis are communication networks modeled by a set of service nodes with queues connected by communication links (see Figure 2.1). The service capacity of a node decides the rate at which it processes the data. The information data are operated in packet switch mode and routed from source to destination node after being served. In our model, a "packet" stands for different data granularities like bit, word, frame, cell, packet, or even infinite small. We do not limit the data granularity to a discrete value, because at times the continuous data model is computationally simpler and provides better abstraction, while at other times the discrete data model does better. Accordingly we use both discrete and continuous time models. And we assume the models start at time/data zero. In our model,

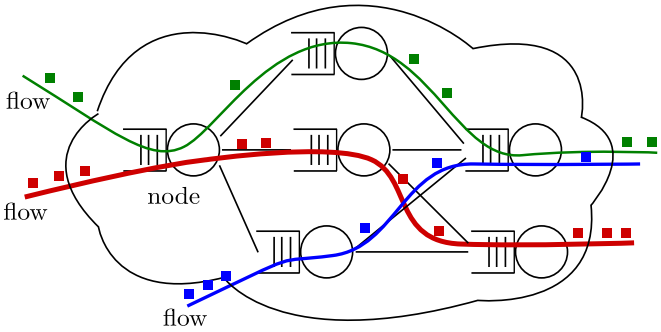


Figure 2.1: General network model.

we call the aggregation or fluid of arriving respectively departing data as *data flow* or *data traffic*. We try to keep the network model simple but with wide applicability, e.g., for different network layers.

Since the service capacity and the buffer for queueing are either expensive or limited resources, we study the performance of a given communication network to see whether these resources are efficiently provisioned and utilized under certain constraints like avoiding loss or limiting end-to-end delay. With evaluating their joint performance we can quantifiablly compare different network design alternatives and dimension the networks. We model the ways of operating (switching, multiplexing and scheduling, demultiplexing) the data and allocating the resources. We quantify the delays of the data traversing from the input to the output ends of the network and the buffers required at each node. Doing these we can discover the tradeoff among these parameters with respect to the changing nature of data flows. However, in this thesis we do not assume that the network inherently puts any constraints on the delay and buffer, i.e., there is no predefined deadline and the buffer size is infinite large. To summarize, we study the behavior mainly of the following performance measures: delay, buffer, service capacity and the data flow characteristics.

Another important issue is when analyzing networks we may meet different kinds of topologies, which makes the analysis more complex. We simplify our modeling by basically observing the data flow of interest. Therefore, when traversing a larger network, it will face with a tandem of nodes, like shown in Figure 2.1. Without restricting the topology, a tree or a feed-forward topology is a simultaneously sweeping of any branching tandem nodes, while a cycle means there exist some same nodes in the tandem nodes. We depict

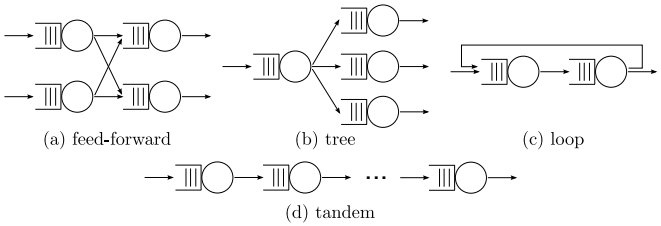


Figure 2.2: Network model simplifications.

these in Figure 2.2.

In order to evaluate the performance of either the discrete or the continuous data/time network models, it is convenient to describe the data flows by means of cumulating the amount of data units arriving at a node until time $t, t \geq 0$. We denote it as an *arrival process* $A(t)$, which is a left continuous, nondecreasing stochastic process. By convention, $A(0) = 0$ and the doubly-indexed form is $A(s,t) := A(t) - A(s)$ for $0 \leq s \leq t$, so, $A(t) = A(0,t)$. We also denote the instantaneous arrival data units in time slot t as $a(t) := A(t-1,t)$. For a continuous time model, $a(t)$ represents the instantaneous arrival rate at time t, accordingly, $A(s,t) = \int_s^t a(\tau)d\tau$. We apply similar definitions for the *departure process* $D(t)$. A causality of $A(t)$ and $D(t)$ holds for any time t, $A(t) \geq D(t)$, unless the flow is transformed before departure. On the other hand, we describe the service provided by a node in the similar way. We denote the cumulative service amount in time interval $[s,t]$ as $S(s,t)$. A node has an infinite buffer and a scheduler.

Now, with $A(t)$ and $D(t)$ we can already represent the intuitive definitions of the following performance measures: *backlog* and *delay*. We first define for any time $t \geq 0$ the backlog $B(t)$ as the amount of data which are waiting in the buffer together with those being served; the delay $W(t)$ as the duration the data experienced therein. Then the idea for representation is to view the node as a black box. We assume that the node is a lossless system. The backlog process of a node at time t is represented as the amount of data which arrived at the node until time t but still not yet departed.

$$B(t) = A(t) - D(t) \; . \tag{2.1}$$

If we further assume that the node schedules the data units locally FIFO, the delay process at time t of the departing flow can be represented as the delay experienced by a departing data at time t or the delay virtually experienced

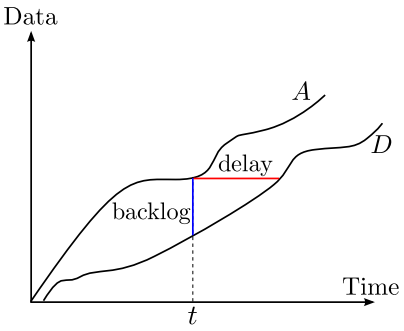


Figure 2.3: Backlog and delay at time t.

by a virtual data at time t, although it does not exist at that time.

$$W(t) = \inf\{d : A(t-d) \leq D(t)\} \,. \tag{2.2}$$

Alert reader may find that the perspective of the backlog process is intuitively distinguished from the delay process. The former is a perspective from the node, while the latter is mainly from the flow. In spite of that, as the departures are the joint result of interacting the arrivals and the service, these two processes are actually decided by both the service and the arrival flow. With this consideration, it is flexible to choose alternative representation of the delay process.

$$W(t) = \inf\{d : A(t) \leq D(t+d)\} \,. \tag{2.3}$$

Eq. (2.2) and Eq. (2.3) are not exactly the same. Eq. (2.2) is from the viewpoint of the departure flow, whereas Eq. (2.3) is from the arrival. For a lossless system, there is no difference for use. But it reveals some important messages, especially for analyzing the networks with flow transformations: we must check on which side we want to observe the delays, because the flows may be altered, then we should keep the losslessness for this flow throughout its end-to-end path. In this thesis, we mainly use Eqs. (2.1) respectively (2.3) as the definition of the backlog respectively delay process. They are depicted in Figure 2.3.

Network calculus builds on the queueing network model and develops its own system theoretic description of network elements. It characterizes the traffic and the service with bounds and relates them to the backlog and the delay bounds, and then, reveal the intrinsic dependencies among these system

parameters. In the following, we summarize the basic concepts and the main results on the basis of two subclasses: deterministic network calculus and stochastic network calculus.

2.2 Deterministic Network Calculus

Network calculus was conceived by Cruz in the early 1990s ([50, 51]), later evolved into *deterministic* network calculus. The motivation is to model the networks non-probabilistically such that it avoids the intractable exact analysis of system parameters ([15]), since they are correlated.

2.2.1 Deterministic Arrival Curve

The original idea is on the traffic, i.e., without identifying the traffic as a known stochastic process it is possibly "*unknown*", and only satisfies certain regularity constraints on the burstiness (see [50]). Defining these constraints, we point out that it is convenient to directly observe each sample-path function of the traffic and describe its burstiness instead of using the stochastic form, whereby deterministic network calculus uses the instrumental definition of the set $\mathcal{F}$ of real-valued, non-negative, and wide-sense increasing functions passing through the origin.

$$\mathcal{F} = \left\{ f : \mathbb{R}^+ \to \mathbb{R}^+, \forall t \geq s : f(t) \geq f(s), f(0) = 0 \right\} .$$

In particular, the arrival and departure flows are represented as the arrival function $F(t)$ and the departure function $F'(t)$ in $\mathcal{F}$. Note, we often use functions F, F' instead of A, D to represent the arrival and departure flows in deterministic network calculus. Now, the bounding function for the burstiness of $F(t)$ is defined as *arrival curve* (also known as *envelope*).

Definition 1. (Arrival Curve) Given a flow with input function $F(t)$, a function $\alpha(t) \in \mathcal{F}$ is an arrival curve for $F(t)$ iff

$$\forall t, s \geq 0, s \leq t : F(t) - F(s) \leq \alpha(t - s) . \tag{2.4}$$

Note, because the departure flow from one node is often the arrival flow of the next node, we call the envelope of the departure flow an arrival curve. An arrival curve builds a bound on the burstiness of any time interval, and for the same length of time intervals the bounds are the same, i.e., both $F(t) - F(s)$ and $F(t + \tau) - F(s + \tau)$ are bounded by $\alpha(t - s)$. Arrival curves on one

hand describe the traffic flows, on the other hand, define new network elements which can regulate the traffic according to the constraints. When flows $F_i(t), i = 1, \ldots, N$ are multiplexing at the same node, each has arrival curve $\alpha_i(t)$, the aggregated flow $\sum_{i=1}^{N} F_i(t)$ has the arrival curve $\sum_{i=1}^{N} \alpha_i(t)$.

An important example of arrival curve is the so-called (σ, ρ)-envelope. It is defined for $\sigma, \rho \geq 0$ by the affine function

$$\alpha(t) = \rho t + \sigma \ ,$$

where ρ upper bounds the long term average rate of the flow and σ bounds any instantaneous burstiness such that it lifts up the bound to allow all the potential burstiness in any time interval. Note that, we may also use $\gamma_{r,b}$ to denote the affine function when discussing deterministic network calculus in the following sections, where $\gamma_{r,b} = rt + b$ if $t > 0$, otherwise 0. A more intuitive interpretation is the buffered *token-bucket*. We extend this function to a more general N-level form as we view it as N token-buckets

$$\alpha(t) = \min_{1 \leq i \leq N} \{\rho_i t + \sigma_i\} \ .$$

This concave function provides time varying constraints, especially for bounding the long term decreasing arrival rates in increasing time intervals. This is applied in some models like IntServ architecture of Internet [24], or Deterministically Bounding INterval-length Dependent (D-BIND) model [90].

2.2.2 Deterministic Service Curve

For the purpose of the performance evaluation, especially to maintain the black box viewpoint, deterministic network calculus also represents the *unknown* service provided by a node with bounds - *service curves*.

Definition 2. (Service Curve) If the service provided by a system S for a given input function $F(t)$ results in an output function $F'(t)$, we say that S offers a service curve $\beta(t)$ if

$$F'(t) \geq \inf_{0 \leq s \leq t} \{F(s) + \beta(t - s)\} \ . \tag{2.5}$$

Note, the service curve is said to be *exact* if Eq. (2.5) holds with equality. We view the server as a black box and only know its inputs and the outputs together with an abstract description to relate them. Sometimes we do not know what really happened in the server, sometimes we do know some information. A not very "black" example is the constant-rate server $\beta(t) = rt$ with service

rate $r > 0$. In fact, instead of showing the service details, which is dependent of the arrivals, it bounds the departures per time unit with r packets. This node is called *work conserving*, since it is not idle if there are packets in the buffer. Using Lindley's equation [103]

$$B(t+1) = [B(t) + a(t+1) - r]^+ \ ,$$

where $[x]^+ = max(0, x)$, we obtain for all $t \geq 0$

$$F'(t) = \inf_{0 \leq s \leq t} \{F(s) + r(t-s)\} \ .$$

The constant-rate service has an exact service curve.

Another simple example is the impulse function $\delta_T(t)$ for $t \geq 0, T \geq 0$, which is defined as $\delta_T(t) = 0$, if $0 \leq t \leq T$, otherwise $\delta_T(t) = \infty$. We have

$$F'(t) \geq \inf_{0 \leq s \leq t} \{F(s) + \delta_T(t-s)\} = \begin{cases} F(t-T) & \text{if } t \geq T \ , \\ 0 & \text{otherwise} \ . \end{cases}$$

From the definition of $W(t)$ in Eq. (2.2) we get $W(t) \leq T$. That means this server delays any arrivals at most by T.

A more general example is the so-called rate-latency service curve $\beta_{R,T}(t)$ $= R[t-T]^+$ for some latency $T \geq 0$. We interpret this service curve as that the node is separated into tow concatenated nodes providing service curve Rt and δ_T.

By defining the service curve, network calculus relates the output of a network system to the input and the service provided by the system. This idea already existed in the traditional system theory, which was successfully used to design electronic circuits. Given a simple circuit with input signal $x(t) \in \mathbb{R}$ and impulse response $h(t)$ for all $t \geq 0$, the corresponding output signal $y(t) \in \mathbb{R}$ is

$$y(t) = \int_0^t x(s)h(t-s)ds := x \otimes h(t) \ ,$$

where the operator $\otimes$ is referred as *convolution*. Review the service curve definition Eq. (2.5), the main difference is that the operations are changed: addition becomes infimum (denoted by $\wedge$), multiplication becomes addition. For those functions defined in $\mathcal{F}$ this change of operations actually means the change of the algebraic structures - from $(\mathbb{R} \cup \{+\infty\}, +, \times)$ algebra to $(\mathbb{R} \cup \{+\infty\}, \wedge, +)$ algebra. And we re-define the convolution as the so-called *min-plus* convolution on the new dioid, i.e., the min-plus algebra.

Definition 3. (Min-plus Convolution) The min-plus convolution of two functions $f, g \in \mathcal{F}$ is defined for all $t \geq 0$ to be

$$(f \otimes g)(t) = \inf_{0 \leq s \leq t} \{f(t-s) + g(s)\} \ .$$

With this definition we can represent the Eqs. (2.4) and (2.5) in the definitions of the arrival curve and the service curve.

$$F(t) \leq F \otimes \alpha(t) \ , \tag{2.6}$$
$$F'(t) \geq F \otimes \beta(t) \ . \tag{2.7}$$

Next we introduce a counterpart of the min-plus convolution, which is also used to simplify the representation of some equations in the following chapters.

Definition 4. (Min-plus Deconvolution) The min-plus deconvolution of two functions $f, g \in \mathcal{F}$ is defined for all $t \geq 0$ to be

$$(f \oslash g)(t) = \sup_{u \geq 0} \{f(t+u) - g(u)\} \ .$$

We also use the convolution and deconvolution defined on the *max-plus* algebra to represent equations in this thesis. They are defined on the dioid $(\mathbb{R} \cup \{+\infty\}, \vee, +)$, where $\vee$ denotes maximum.

Definition 5. (Max-plus Convolution and Deconvolution)

$$(f \bar{\otimes} g)(t) = \sup_{0 \leq s \leq t} \{f(t-s) + g(s)\} \ , \ (\vee, + \text{ convolution}) \ ,$$

$$(f \bar{\oslash} g)(t) = \inf_{u \geq 0} \{f(t+u) - g(u)\} \ , \ (\vee, + \text{ deconvolution}) \ .$$

Deterministic network calculus can also describe the upper bound on the service, the so-called maximum service curve, denoted by $\gamma(t)$, if $F'(t) \leq F \otimes \gamma(t)$ for all $t \geq 0$. This maximum service curve is not that often used in this thesis as the (minimum) service curve. Hence, when we say service curve, that means minimum service curve in the rest of this thesis. We may also omit the brackets around $f \otimes g$ and the other operations.

Furthermore, a class of network nodes offer *strict* service curve. We define busy period as the time interval in which the system always has positive backlog.

Definition 6. (Strict Service Curve) A network node offers a strict service

curve $\beta(t)$ to a flow, if during any busy period $u, u \geq 0$ the output is at least equal to $\beta(u)$.

Note, a strict service curve is also a service curve, however, the converse is not true. We will later introduce some properties that only hold with strict service curve. The strict service curve is the minimum guaranteed service amount during any busy period u. This is similar to the bounds on the arrivals - we observe and describe the process itself. Consider a node that offers a variable service capacity to a flow. We denote the cumulative capacities using function $C(t) = \sum_{i=0}^{t} c(i)$ for the time from 0 to t. These capacities reflect the amount of data which can be (virtually) served by the node if the according arrival data exist. If a fixed function $\beta(t)$ satisfies $C(t) - C(s) \geq \beta(t-s)$ for all $0 \leq s \leq t$ that fall into a continuously busy period, then $\beta(t)$ is a strict service curve. Thus $\beta(t)$ is also a service curve. Because in a busy period the departures are equal to the cumulative service capacities, this condition can also be equivalently expressed by $F'(t) - F'(s) \geq \beta(t-s)$. The concept of variable capacity strict service curve potentially provides a convenient way to construct service curve.

2.2.3 Worst-case Analysis

The outstanding fundamental results of the deterministic network calculus are that it models the network services and offers worst-case analysis by deriving the performance bounds. Arrival curve and the service curve facilitate the derivation.

Deterministic Single Node Performance Bounds

The next theorem derives the dealy bound, backlog bound, and the bound, i.e., arrival curve, of the output flow.

Theorem 1. *(Performance Bounds) Consider a system S that offers a service curve $\beta(t)$ for any time $t \geq 0$. Assume a flow $F(t)$ traversing the system has an arrival curve $\alpha(t)$. The output is $F'(t)$. Then we obtain the following performance bounds for all $t \geq 0$*

$$\begin{aligned}
&\textit{backlog: } B(t) \leq (\alpha \oslash \beta)(0) =: v(\alpha, \beta) \,, \\
&\textit{delay: } W(t) \leq \inf\{t \geq 0 : (\alpha \oslash \beta)(-t) \leq 0\} =: h(\alpha, \beta) \,, \\
&\textit{output (arrival curve } \alpha' \textit{ for } F'\textit{): } \alpha' = \alpha \oslash \beta \,.
\end{aligned}$$

Note, the backlog and the delay bounds are graphically interpreted as the maximal vertical and the horizontal distances between arrival curve and

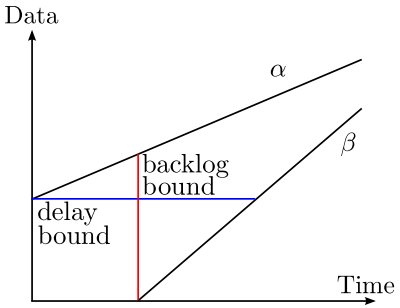


Figure 2.4: Backlog bound and delay bound.

service curve, v and h. See Figure 2.4 for the illustration. These bounds are as *tight* as $\alpha(t)$ and $\beta(t)$ are ([95]), if $\alpha(t)$ is sub-additive, $\alpha(0) = 0$ and $\beta(0) = 0$. That means, there exists such a causal system such that $B(t) = v(\alpha, \beta)$ and $W(t) = h(\alpha, \beta)$ for some $t \geq 0$ respectively. We next sketch the proof of the backlog bound, because we use the similar argument in many parts of this thesis. We first denote the vertical distance of two functions $f(t)$ and $g(t)$ at some index t as $v(f(t), g(t))$. Then $v(f, g) := sup_{t \geq 0} v(f(t), g(t))$. Using the definition of backlog in Eq. (2.1) we have

$$\begin{aligned} B(t) &= F(t) - F'(t) \\ &\leq F(t) - \inf_{0 \leq s \leq t} \{F(s) + \beta(t-s)\} \\ &= \sup_{0 \leq s \leq t} \{F(t) - F(s) - \beta(t-s)\} \\ &\leq \sup_{0 \leq s \leq t} \{\alpha(t-s) - \beta(t-s)\} \\ &= \sup_{0 \leq s \leq t} \{v(\alpha(t-s), \beta(t-s))\} \\ &\leq v(\alpha, \beta) \ . \end{aligned}$$

We can see that the backlog bound originates from the bounds on the service and its arrivals. The proof for the delay bounds and the output bounds can be referred to [95]. Note, this theorem requires the stability condition for a system, i.e., the condition under which the buffer can never be infinity. To guarantee that we should ensure $v(\alpha, \beta) < \infty$, which implies the long term rate of arrival curve should not exceed the long term rate of service curve (e.g., $\alpha = \gamma_{r,b}, \beta = \beta_{R,T}$, then $r \leq R$), which are usually interpreted as the long term arrival rate and service rate.

Deterministic Convolution-Form Representation of Networks

Now let us introduce another fundamental result of the network calculus - convolution-form representation of networks, within an underlying min-plus algebra, whereby the service curve provided to a single flow by the whole network can be expressed by convolving the service curve of each node. Before the introduction we review some useful properties of the min-plus convolution.

Theorem 2. (Properties of $\otimes$)

- *(Closure)* $(f \otimes g) \in \mathcal{F}$.
- *(Associativity)* $(f \otimes g) \otimes h = f \otimes (g \otimes h)$.
- *(Commutativity)* $f \otimes g = g \otimes h$.
- *(Distributivity with respect to* $\wedge$*)* $(f \wedge g) \otimes h = (f \wedge h) \otimes (g \wedge h)$.
- *(Addition of a constant) For any* $K \in \mathbb{R}^+$, $(f+K) \otimes g = (f \otimes g) + K$.
- *(Isotonicity) Let* $f, g, f', g' \in \mathcal{F}$. *If* $f \leq f'$ *and* $g \leq g'$ *then* $f \otimes g \leq f' \otimes g'$.

The proofs of these properties can be found in [95]. These properties instrumentally build up the concatenation theorem.

Theorem 3. *(Concatenation of Nodes) Consider a flow* $F(t)$ *traversing a set of network nodes* $S_i, i = 1, \ldots, N, N \geq 1$ *in sequence. Each node offers a service curve* $\beta_i(t)$ *to the flow. Then the concatenation of these nodes offers a service curve* $\beta_{net}(t) = \beta_1 \otimes \beta_2 \otimes \cdots \otimes \beta_N(t)$ *to the flow such that the output function* $F'(t)$ *of the network satisfies*

$$F'(t) \geq F \otimes \beta_{net}(t) \ .$$

Proof. Denote the input respectively output functions of node i by $F_i(t)$ respectively $F'_i(t)$, clearly $F'_{i-1}(t) = F_i(t)$ and $F'(t) = F'_N(t)$. We can iteratively derive the following

$$\begin{aligned} F'(t) &\geq F_N \otimes \beta_N(t) \\ &\geq (F_{N-1} \otimes \beta_{N-1}) \otimes \beta_N(t) \\ &\cdots \\ &\geq (\cdots ((F_1 \otimes \beta_1) \otimes \beta_2) \otimes \cdots)(t) \end{aligned}$$

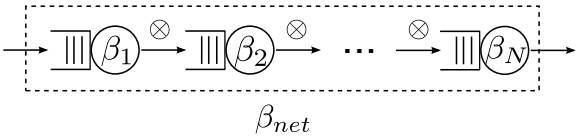


Figure 2.5: Network service curve.

$$\begin{aligned} &= F_1 \otimes (\beta_1 \otimes \beta_2 \otimes \cdots \otimes \beta_N)(t) \\ &= F \otimes \beta_{net}(t) \,. \end{aligned}$$

From the second line to the fourth line we recursively used the closure and isotonicity properties. In the fifth line we recursively used the associativity. □

$\beta_{net}(t)$ is referred as the network service curve. We depict this in Figure 2.5. Not only used for concatenating nodes, we can also use this result to decompose a service curve. For example, a rate-latency service curve $\beta_{R,t}(t)$ can be decomposed as $\delta_T \otimes \lambda_R(t)$, where $\lambda_R(t)$ is the peak rate function defined as Rt if $t > 0$, 0 otherwise. Moreover interestingly, applying the commutativity property of the min-plus convolution to $\beta_{net}(t)$ we find that the order of the nodes has no effect. That implies only changing the location of a bottleneck node will not solve the problem. Consider the concatenation of N constant-rate services with rate $R_1, \ldots, R_N$. $\beta_{net}(t) = min\{R_1, \ldots, R_N\}t$, wherever the minimum rate service locates.

Now we introduce the main contribution of the concatenation theorem. In order to derive a delay bound of a network with multiple nodes, we have two options. One is to compute the delay bounds on each node and sum them up, the other is to apply the network service curve. The former is called *node-by-node* analysis. The latter allows us to *pay bursts only once (PBOO)*. Let us compare them for a simple example. Consider a flow traversing a simple network with two nodes. We denote the service curve of the nodes as $\beta_i(t), i = 1, 2$ and the arrival curve of the input respectively the output at node i as $\alpha_i(t)$ and $\alpha_i'(t)$, again $\alpha_1' = \alpha_2$. Assume rate-latency service curves $\beta_i(t) = \beta_{R_i,T_i}$ and token-bucket arrival curve $\alpha_1(t) = \gamma_{r_1,b_1}$. The network satisfies the stability condition $r_1 < R_i$, such that the backlogs of both nodes will never be infinity. From the output bound in Theorem 1 we know $\alpha_2(t) = \alpha_1 \oslash \beta_1(t)$. Then we get the delay bounds at each node

$$W_1(t) \quad \leq \quad h(\alpha_1, \beta_1) = T_1 + \frac{b_1}{R_1} := W_1$$

$$W_2(t) \quad \leq \quad h(\alpha_1 \oslash \beta_1, \beta_2) = T_2 + \frac{b_1 + r_1 T_1}{R_2} := W_2 \ .$$

We get the end-to-end delay bound as

$$W(t) \leq W_1 + W_2 = T_1 + T_2 + \frac{b_1}{R_1} + \frac{b_1}{R_2} + \frac{r_1 T_1}{R_2} \ .$$

Applying network service curve we have

$$\begin{aligned} W(t) \quad &\leq \quad h(\alpha_1, \beta_1 \otimes \beta_2) \\ &= \quad T_1 + T_2 + \frac{b_1}{min\{R_1, R_2\}} := W \ . \end{aligned}$$

It is easy to see that $W_1 + W_2 > W$. The former considers the delays caused by the arrival burstiness on both nodes, which is, however, redundant. We only need to consider it once on the bottleneck node ($min\{R_1, R_2\}$). $\frac{r_1 T_1}{R_2}$ part is also redundant, because the burstiness gathered by the delay element in time T_1 at the first node is potentially decided by the burstiness from the arrival. If consider a more general n nodes network with the same service curve $\beta_i(t) = \beta_{R,T}$, we have the node-by-node delay bound $nT + \frac{(n^2-n)r_1 T + 2nb_1}{2R}$, while using network service curve the delay bounds is $nT + \frac{b_1}{R}$. The delay bounds scale in the order $\mathcal{O}(n^2)$ v.s. $\mathcal{O}(n)$. This example shows the PBOO gain generally for FIFO scheduling. For non-FIFO scheduling, we need to construct different service curve, more details can be found in [128, 126]. In this thesis we assume locally FIFO scheduling. In the next section we discuss more scheduling schemes when we face the multiplexing of flows, but still based on that each flow is served locally FIFO.

Scheduling Modeling with DNC

A further strength of the network calculus is that service curve can abstract a variety of scheduling algorithms. Without loss of generality , we consider two flows $F_1(t)$ and $F_2(t)$ multiplexing at one node. It is possible to derive the performance bounds for the aggregated flows, which can be represented by $F(t) = F_1(t) + F_2(t)$. The challenge is to derive the performance bounds for the individual flow, which is sharing the service with the other flow under different scheduling rules. The service curve model can identify the amount of service left over for one flow by the other. Then for a flow traversing a network, we can still represent the network service curve for it as the convolution-form of the *leftover* service curve at each node. There

exists a number of scheduling algorithms. Next, we show some general examples, as they are used in this thesis. A more thorough list can be found in [95, 102, 63].

First, consider two flows $F_1(t), F_2(t)$ multiplexing at one node without assuming any order in which the flows are served - blind multiplexing. If the node offers a strict service curve $\beta(t)$ and the arrival curves are $\alpha_1(t)$ and $\alpha_2(t)$, then the service curve seen by $F_1(t)$ is

$$\beta_1(t) = [\beta(t) - \alpha_2(t)]^+ \ . \tag{2.8}$$

We see that this is symmetric to both flows. We also should note that this can not be derived from the general service curve ([95]). This model assumes knowing no information of order. When the scheduler employs Static Priority (SP) [95, 125, 98], we can then improve the leftover service curve. Now suppose that n flows $F_1(t), F_2(t), \ldots, F_n(t)$ with decreasing priorities. Flow i sees a wide-sense increasing leftover service curve by subtracting the arrival curves of the flows with higher priority

$$\beta_i(t) = \left[\beta(t) - \sum_{j=1}^{i} \alpha_j(t)\right]^+ \ . \tag{2.9}$$

This equation implies that flow 1 sees the whole service curve. In fact, for a discrete data model, if we know the maximal packet length in the other flows with lower priority $l_{max,i}$, we should also subtract this if we assume to protect the service of a packet from the preemptive scheduling.

The leftover service curve can model another general case: First-In-First-Out (FIFO) scheduling. If the nodes offers a service curve $\beta(t)$, then for parameter $x \geq 0$ we have the leftover service curve

$$\beta_1(t) = [\beta(t) - \alpha(t-x)]^+ 1_{\{t>x\}} \ .$$

When $x = 0$, this is the blind multiplexing case. Otherwise we can choose different x. Although it is shown that not every x makes it fulfill the service curve definition ([95]), we can still optimize the performance bounds by carefully changing x. The proof can also be found in [95].

2.3 Stochastic Network Calculus

Deterministic network calculus assumes that the stochastic traffic and the service capacity are "unknown", thus it bounds them and provides the worst-case performance of networks. This is especially useful to those critical systems. However, in many other real world applications, on one hand, we can not always assume that the bounds exist or to use a regulator to bound the traffic according to some specification. For example, consider a Poisson arrival flow, without regulation there exists no deterministic arrival curve. On the other hand, although the deterministic bounds exist, the results the worst-case analysis produces are too conservative, since we do actually "know" that the things are not always running in a nearly collapse situation. The random fluctuation between some normal values and the critical value does not violate the deterministic bound but will bring us more possibilities.

A simple example is the statistical multiplexing. Consider that N Bernoulli flows are multiplexing, each generates one data unit per time unit with probability $p, 0 \leq p \leq 1$. The deterministic arrival curve for each flow is $\alpha_i(t) = 1 \cdot t, i = 1, \ldots, N$, thus for the aggregate is $\alpha(t) = Nt$. This bound is greedy only when all these N flows generate data simultaneously and for all the time. However, the probability of this event is zero. From the Central Limit Theorem (CLT), we know that each flow generates data at rate p asymptotically, so that for N flows at rate Np. For large times t, the deterministic bound is too conservative. We can loosen the deterministic bounds through a violation probability, such that we improve the resource utilization only with a small loss of quality of service (QoS) accuracy if the system can tolerate it. Usually a small violation probability, e.g., in the order of 10^{-6}, can already requite a large statistical gain over the deterministic bounds.

Although there are queueing theory and some other stochastic modeling techniques, we still want to provide a theoretic framework to provision the system by analyzing the bounds of its characteristics. Stochastic network calculus has been explored to achieve this objective. In recent years, many efforts towards a stochastic network calculus have been made (see e.g., [33, 156, 53, 138, 28, 8, 42, 61, 79, 96]). Many different definitions of stochastic extensions of arrival and service curves have been proposed and discussed. In particular, to provide a stochastic service curve definition that still allows for a favorable concatenation has shown to be a hard problem for some time. In this section, we simply provide the necessary definitions and basic results as they pertain to the work in this thesis, without delving into the deep discussions on alternative definitions. Our defintions are classified in two groups. One is to use the non-random bounding functions, which is mainly based on [28] and

can be seen as the direct generalizations of the deterministic network calculus counterparts. The other is to directly observe the stochastic traffic and service processes and describe the bounds either as a bounding random process or some characterization relating to time interval [36, 86, 61]. In this section, we denote the traffics and services as random processes. Next, we introduce the definitions in turn.

2.3.1 Stochastic Bounds of Arrivals

Directly extended from the deterministic arrival curve, we have the following definition.

Definition 7. (Stochastic Arrival Curve) A nonnegative wide-sense increasing function $\alpha(t)$ is defined as the stochastic arrival curve for the arrival process $A(t)$, if it satisfies for all $t \geq 0$ that

$$Pr\left(\sup_{0\leq s\leq t}\{A(s,t)-\alpha(t-s)\}\geq 0\right)\leq \varepsilon \;\Leftrightarrow\; Pr\left(A(t)\geq A\otimes\alpha(t)\right)\leq\varepsilon\,. \tag{2.10}$$

Note, ε might also be a violation function of some hidden parameters from the arrival curve, or inversely, α is a function of ε. Both expressions can be found in the literatures [155, 138, 19]. Note further, this definition provides a *sample path bound* as for example discussed in [28], where it is also called sample path effective envelope. It is a stronger extension from the so-called *effective envelope* defined in [20, 29], where we move $s \leq t$ outsides of the probability such that together with for all $t \geq 0$

$$Pr\left(A(s,t)\geq\alpha(t-s)\right)\leq\varepsilon\,.$$

The sample path envelope reversely observes the entire history until time t of the arrivals and relates them to the violation probability, while the effective envelope observe the arrivals in all the fixed intervals $[s,t]$. We can understand the difference if we for example consider the following expressions: "at most one sample path violates all the time" v.s. "at most one sample path violates in any time interval". The former constrains the same sample path all the time; although one, the latter can constrain different one sample path at different time points. The advantage of the sample path envelope is that it simplify the derivation of the performance bounds for a single node case and for the multiple nodes case if carefully choosing the violation functions, since it resemble to the corresponding deterministic bounds. However, the simplicity hurts a bit the agility and produces a looser bound. This can be seen from

the single way transform of both envelopes. We usually construct the sample path envelope from the effective envelope using Boole's inequality, i.e.,

$$Pr\Big(\sup_{0\leq s\leq t}\{A(s,t)-\alpha(t-s)\}\geq 0\Big)\leq\sum_{0\leq s\leq t}Pr\left(A(s,t)-\alpha(t-s)\geq 0\right)\ .$$

That means, we can use the effective envelope as the sample path envelope, instead, with redundant sum of the violation probabilities. An alternative to do that is to add a slack factor $\delta > 0$ to the rate of the effective envelope $\alpha_{eff}(t)$ such that the sample path envelope $\alpha_{sp}(t)$ satisfies $\alpha_{sp}(t) = \alpha_{eff}(t) + \delta t$. This adjustment helps us to find a better violation probability according to the requirement through inserting δ into the violation function of the effective envelope. See [40, 82] for details. The essence behind this operation is the inherent dependence between the envelope and the violation probability. A trivial example illustrates that very well. If the envelope is ∞ instead of some reasonable values, the violation probability is clearly 0. Therefore, changing envelop will have effect on the violation probability and vice versa, intuitively, smaller envelope, bigger violation probability. In the literatures the usual treatment to establish the relationship is as stated above, to mathematically insert parameters into either the envelope or the violation function such that this parameter is used by the counterpart. Next we show an important class of the arrival curves as an example of the stochastic arrival curve.

The generalized Stochastically Bounded Burstiness (gSBB) model ([157, 53, 7]) belongs to the stochastic arrival curve. It defines $\alpha(t)$ as an affine function with rate r and instantaneous burst bound $\sigma \geq 0$, i.e., $\alpha(t) = \rho t + \sigma$. Then for all $t \geq 0$

$$Pr\left(\sup_{0\leq s\leq t}\{A(t)-A(s)-\rho(t-s)-\sigma\}\geq 0\right)\leq\varepsilon(\sigma)\ .$$

Here we explicitly denote the violation probability as a function of σ as it is given in the envelope expression. When σ increases, $\varepsilon(\sigma)$ decreases. This arrival curve is a sample path envelope. It extends the Stochastically Bounded Burstiness (SBB) model [138] and can be constructed from it. The violation function for SBB and gSBB should be n-fold integrable, i.e., for any order n

$$\underbrace{\int_{\sigma}^{\infty}\cdots\int_{\sigma}^{\infty}}_{n\text{ times}}\varepsilon(u)(du)^n$$

is bounded for any $\sigma > 0$, such that when the traffic traverses a network of nodes we can still get an envelope. Then the SBB model provides the effective envelope as for all $0 \leq s \leq t$

$$Pr\left(A(s,t) \geq \rho t + \sigma\right) \leq \varepsilon(\sigma) \ .$$

In both the gSBB and SBB stochastic arrival curves we map the violation probability onto the instantaneous burstiness bound. Choosing different forms of violation function we can model traffics with different characteristics. A very important class is the Exponentially Bounded Burstiness (EBB) [156] with $\varepsilon(\sigma) = Me^{-\theta\sigma}$, where M and θ are closely related to the traffic process. The exponential decay rate θ determines the shape of the violation function. The rate ρ also depends on θ.

Since $\varepsilon(\sigma)$ is in general form, the gSBB and SBB can model many kinds of traffics [107]. EBB can model short range dependent (SRD) traffics, e.g., Markov-Modulated Processes (MMP), while the sum of exponential functions $\varepsilon(\sigma) = \sum_i M_i e^{-\theta_i \sigma}$ or a Weibullian Bounded Burstiness (WBB) $\varepsilon(\sigma) = Me^{-\theta\sigma^{2(1-H)}}$ can model some long range dependent (LRD) or self-similar traffics.

Besides the bounding functions, one can also indirectly or directly observe the traffic using its statistical characteristics and find a bound. Indirectly we can bound the arrival process with a *stochastically larger* process if it is more convenient to get the statistical characteristics of this bounding process. The stochastic ordering for the arrival process $A(t)$ and bounding process $E(t)$ is defined as $A(s,t) \leq_{st} E(t-s)$, if for all $0 \leq s \leq t$ and all real value x

$$Pr(A(s,t) > x) \leq Pr(E(t-s) > x) \ .$$

Based on that, Kurose [92] provides a stochastic arrival curve by bounding the arrivals in each time intervals $t_k, k = 1, 2, \ldots$ with a family of random variables (*r.v.*) $E(t_k)$, such that $A(s, s+t_k) \leq_{st} E(t_k)$ for all $s \geq 0$. Directly, we can use the arrival process itself as the bounding process, or more precisely and strictly, find a bounding process *almost surely larger* than the arrivals, i.e., $A(s,t) \leq E(s,t)\ a.s.$ [40]. A lot of effort on capturing the traffics using known stochastic processes permit us to simply observe the arrival process instead of constructing its counterparts according to the orderings.

An accurate manner to represent the arrival process is to use its moment generating function (MGF) and describe the bound of it. If we define and denote the MGF of a random variable X as $M_X(\theta) = E\left[e^{\theta X}\right]$ for any $\theta \geq 0$, we have the following definition.

Definition 8. (MGF Bound of Arrivals) The arrival process $A(s,t)$ has MGF bound, if its MGF exists and satisfies $M_{A(s,t)}(\theta) \leq e^{\theta\alpha(t-s,\theta)}$ for all $0 \leq s \leq t$, where α is a function of time interval and θ.

We see that this is a stationary bound. The use of MGF in network calculus originates from the *effective bandwidth* [86], which is defined for an arrival process $A(t)$ as

$$\alpha^{eb}(\theta,t) = \sup_{s\geq 0}\left\{\frac{1}{\theta t}\log E\left[e^{\theta A(s,t+s)}\right]\right\} .$$

Later Chang [34, 36] explicitly generalizes it to extend Cruz's (σ,ρ) calculus. The arrivals are $(\sigma(\theta),\rho(\theta))$-upper constrained for some $\theta > 0$, if

$$\frac{1}{\theta}\log E\left[e^{\theta A(s,t)}\right] \leq \rho(\theta)(t-s)+\sigma(\theta) , \tag{2.11}$$

for all $0 \leq s \leq t$. We can see that the new term θ is generated from the MGF's parameter. A direct construction of the MGF bound comes from the effective bandwidth, i.e.,

$$M_{A(s,t)}(\theta) \leq e^{\theta\alpha^{eb}(\theta,t-s)\cdot(t-s)} .$$

A further characterization, the θ-minimum envelope rate (θ-MER) denoted by $a^*(\theta)$, is defined to lower bound all the $\rho(\theta)$ with the same θ,

$$a^*(\theta) = \limsup_{t\to\infty}\alpha^{eb}(\theta,t) . \tag{2.12}$$

We say $A(t)$ has θ-envelope rate $\rho(\theta)$ if $\rho(\theta) \geq a^*(\theta)$. If $\sigma(\theta) < \infty$, a $(\sigma(\theta),\rho(\theta))$-upper constrained $A(t)$ always has a θ-envelope rate $\rho(\theta)$, which can be easily proved by constructing Eq. (2.12) from Eq. (2.11). It is also shown in [36] that θ-MER is increasing in θ: from the mean rate to the peak rate, as θ increases from 0 to ∞. A later work of Fidler [61] more systematically extends this MGF bound in the framework of network calculus, particular in solving the performance bounds of networks with multiple tandem nodes. The MGFs are usually known if the traffic models are given, e.g., regulated, Poisson, Markov on-off, periodic even fractional Brownian motion (fBm) for the self-similar or long-range dependent (LRD) traffics.

Now we expose the relationship between the MGF bound and the stochastic arrival curve. We can connect the effective bandwidth with the effective envelope, since they are respectively used to construct the MGF bound and the sample path stochastic arrival curve. This is explicitly studied by Li *et.*

al.[98]. Given effective bandwidth $\alpha^{eb}(\theta, t)$, an effective envelope with violation probability ε is given by

$$\alpha(t) = \inf_{\theta>0} \left\{ t\alpha^{eb}(\theta, t) - \frac{\log \varepsilon}{\theta} \right\} .$$

Conversely, for each $\varepsilon \in (0, 1)$ and given $\alpha(t)$

$$\alpha^{eb}(\theta, t) \leq \frac{1}{\theta t} \log \left(\int_0^1 e^{\theta\alpha(t)} d\varepsilon \right) .$$

This facilitates the construction of the traffic bounds in both directions. But it is worth noting that we loosen the bounds by cycling the construction. We do not show the proof for this result, instead, we show the construction for an important example, the EBB, and the conversion, as it uses the same idea as the general SBB. But before that, let us recall a instrumentally used mathematical result, the Chernoff bound, which says, for a random variable X and $\theta > 0$, we have

$$Pr(X \geq x) \leq e^{-\theta x} E\left[e^{\theta X}\right]$$

for all x. A direct application is that, for random variable $A(s, t)$, we have $Pr(A(s, t) \geq x) \leq e^{-\theta x} E\left[e^{\theta A(s,t)}\right]$. It is a form of stochastic arrival curve, which relates the violation probability to the MGF of $A(t)$. This bridges the gap between both bounds.

Now, consider that an arrival process $A(t)$ has MGF bound $E\left[e^{A(s,t)}\right] \leq e^{\theta r(t-s)}$. We use the Chernoff bound and construct the form of EBB with a parameter σ like

$$\begin{aligned} Pr(A(s, t) \geq r(t - s) + \sigma) &\leq e^{-\theta r(t-s)} e^{-\theta\sigma} E\left[e^{\theta A(s,t)}\right] \\ &\leq e^{-\theta\sigma} . \end{aligned}$$

Then we can use the union bound to derive the stochastic arrival curve that is defined for the sample paths.

Conversely, given $Pr(A(s, t) \geq r(t - s) + \sigma) \leq e^{-\theta\sigma}$ and note that $A(t) \geq 0$ we have

$$\begin{aligned} E\left[e^{\theta' A(s,t)}\right] &= \int_0^\infty Pr\left(e^{\theta' A(s,t)} > x\right) dx \\ &\leq 1 + \int_1^\infty e^{-\theta\sigma} d\left(e^{\theta'(r(t-s)+\sigma)}\right) \end{aligned}$$

Bounding Process of $\mathbf{A(s,t)}$	**MGF Bound**
Bernoulli	$e^{\theta\frac{1}{\theta}\log(1-p+pe^{\theta})(t-s)}$
Poisson	$e^{\theta\frac{1}{\theta}\lambda(e^{\theta}-1)(t-s)}$
MMP	$e^{\theta\frac{1}{\theta}\log(sp(\phi(\theta)\lambda))(t-s)}$
Marked Point Process $((T_i)_{i\geq 1},(X_i)_{i\geq 1})$	$X_i \sim exp(\mu)$ *i.i.d.* If $T_i = i$, then $e^{\theta\frac{1}{\theta}\log(\frac{\mu}{\mu-\theta})(t-s)}$ If $T_{i+1}-T_i \sim exp(\frac{1}{\lambda})$, then $e^{\theta\frac{\lambda}{\mu-\theta}(t-s)}$

Table 2.1: MGF Bounds of Arrivals.

$$= \quad 1+\frac{\theta'}{\theta-\theta'}e^{\theta r(t-s)}\ ,\ \theta'<\theta\ .$$

In the second line we let $x = e^{\theta'(r(t-s)+\sigma)}$.

A critical prerequisite for MGF bound is that MGF must exist. This restricts the conversion to EBB and at the same time limits the application to a more general class of traffics. Another critical point of the MGF bound is that in order to enable a lower computational complexity, the MGF bound should better be an exponential of a linear function of t. An counterexample is the MGF of the fBm, which does not fall into the EBB class. Next, we list the MGF (bounds) of some very useful traffic models, since most of them are used in the rest of this thesis, see Table 2.1. See the concrete definition of MMP in Section 4.1.1.

The applications of the two kinds of definitions, the stochastic arrival curve and MGF bound, on bounding the traffic multiplexing exhibit different perspectives. Using the former, as discussed at the beginning of this section, the gain is the smaller bound than simply summarizing the respective arrival bounds. Consider SBB firstly. Let A_1 and A_2 be two multiplexing flows, each has effective envelop $\rho_i t+\sigma, i=1,2$ with violation probability ε_i, then the aggregate $A(s,t) = A_1(s,t)+A_2(s,t)$ has an effective envelope $(\rho_1+\rho_2)t+\sigma$ with violation probability $\varepsilon_1(p\sigma)+\varepsilon_2((1-p)\sigma)$, where $p\in(0,1)$. We get this by observing that $\{A(s,t)\geq(\rho_1+\rho_2)(t-s)+\sigma\}$ $\subseteq\{A_1(s,t)\geq\rho_1(t-s)+p\sigma\}\cup\{A_2(s,t)\geq\rho_2(t-s)+(1-p)\sigma\}$. This derivation requires no independence of flows. But the drawback of this calculation is the possible unbounded cumulation of the violation probabilities when the number of flows is large. The MGF bound of the aggregate flow is derived differently. It is easy to see that $M_{A(s,t)}(\theta) = M_{A_1(s,t)}(\theta)M_{A_2(s,t)}(\theta)$, if we assume the independence of the flows. The bound is the sum of respective bounds. The drawback is the independence assumption. But on the

	Tail Bound	MGF Bound
computation	complex	relatively simple
dependency assumption	no	ind. or depnd. (Hölder)
prob. convergence	slow	fast

Table 2.2: Tail bound v.s. MGF bound.

other hand, we can solve it by using Hölder inequality (see [36]), i.e., for two *r.v.* X, Y and any $p > 1, q > 1$ with $\frac{1}{p} + \frac{1}{q} = 1$ we have

$$E|XY| \leq (E|X|^p)^{\frac{1}{p}} + (E|Y|^q)^{\frac{1}{q}} . \tag{2.13}$$

The benefit is also obvious, the calculation is greatly simplified and the potential unbounded cumulation can be scaled down to a bounded value because we move the sum to the exponent of the exponential function. We summarize these statements in Table 2.2.

2.3.2 Stochastic Service Curve and Dynamic Server

The networks consists of heterogeneous elements with diverse capacities. The services these elements provide to the traffics are stochastic processes. Besides the deterministic service curve we can also adopt the same ways to stochastically bound the service like what we do for the arrivals, i.e., as bounding function with violation probability as well as bounding process with its MGF bound.

Definition 9. (Stochastic Service Curve) If the service S provided by a system for a given arrival $A(t)$ results in a departure $D(t)$, we say that S offers a stochastic service curve β if $\forall t \geq 0$

$$Pr\left(D(t) \geq A \otimes \beta(t)\right) \geq 1 - \varepsilon . \tag{2.14}$$

We can also denote it as β^{ε} to differ with the deterministic one. This definition follows again [28], where it is called *effective service curve*. In fact, it is a modification of the prior definition introduced by Cruz [53], where the service curve is stochastically delivered to the traffic with deficit profile $\varepsilon(\sigma)$, such that

$$Pr\left(D(t) \leq A \otimes S(t) - \sigma\right) \leq \varepsilon(\sigma) . \tag{2.15}$$

The difference between Eq. (2.15) and (2.14) is the form of service curve functions. They can be transformed to each other through defining $\beta(t) =$

$S(t) - \sigma$ or more carefully, $\beta(t) = [S(t) - \sigma]^+$, and vice versa. Hence, although Eq. (2.14) exhibits a single violation probability, it is implied as a function of some σ, if we construct $\beta(t) = \beta(t) + \sigma - \sigma := S(t) - \sigma$. Note, these definitions are sample path wise definition as the convolution is used. When generalizing these sample path stochastic service curves, we may meet a trap. Assuming stationarity and ergodicity, $Pr\left(\exists t \geq 0 : D(t) \leq A \otimes \beta(t)\right)$ is either 0 or 1 [28, 48].

Another important class of service bound is the bounding process. It is a stochastic interpretation of a former concept introduced by Chang [34], the dynamic F-server, where the deterministic service curve $F(t)$ is extended to the time varying setting as a bi-variate function $F(s,t)$ in $\mathcal{F}$ (from $\beta(t)$ to $\beta(s,t)$ using our terminology). To do that, the convolution is extended to the bi-variate functions. For bi-variate functions f, g we define $f \otimes g(s,t) := \inf_{s \leq u \leq t} \{f(s,u) + g(u,t)\}$ for $0 \leq s \leq t$. Then we write $f \otimes g(t) = f \otimes g(0,t)$. Now we interpret F into a random process $S(s,t)$ and have

Definition 10. (Dynamic Server) A server guarantees a double indexed dynamic service process $S(s,t)$ for an arrival process $A(t)$ if the departure process $D(t)$ satisfies that

$$D(t) \geq A \otimes S(t)$$

for all $t \geq 0$. If the inequality holds with the equality, we say the dynamic service is exact.

Note, since this definition extends the lower bounding function of service, it is actually still a characterization of the service (lower bound) instead of the service process itself. But in order to avoid the trivial cases, e.g. $0(s,t)$, we should usually understand it as the service process itself by considering that the service process is almost surely lower bounded by itself. For this case, we call a server that guarantees a dynamic service process $S(s,t)$ as a dynamic server $S(s,t)$. An example of dynamic server is the work conserving server with time varying service capacity $S(s,t)$. Consider current time t and the beginning of the last busy period τ, $\tau \leq t$. On one hand, for the cases t falls into the last busy period respectively not in, we know $D(t) = A(\tau) + S(\tau,t)$ or $D(t) = A(t)$. Then we have $D(t) \geq \inf_{0 \leq s \leq t} \{A(s) + S(s,t)\}$. Hence, a time varying work conserving server is a dynamic server. Further on the other hand, a work conserving server also assures that $D(t) \leq D(s) + S(s,t)$, $\forall s \in [0,t]$, which implies $D(t) \leq A(s) + S(s,t)$, $\forall s \in [0,t]$. We can see that it is an exact dynamic server.

Similar to the MGF bounds of the arrival processes, we define the MGF (Laplace transform) bound of a dynamic server.

Definition 11. (MGF Bound of Dynamic Server) The dynamic server $S(s,t)$ has MGF bound, if its MGF exists and satisfies $M_{S(s,t)}(-\theta) \leq e^{-\theta\beta(t-s,\theta)}$ for all $0 \leq s \leq t$, where β is a function of time interval and θ.

Clearly, this is again a stationary bound like its arrival counterpart. And because we need a lower bound, we use negative parameter for the MGF. A direct example of this MGF bound is to mimic the (σ,ρ) bound, i.e., $M_{S(s,t)}(-\theta) \leq e^{-\theta(\rho(t-s)-\sigma)}$, which recovers the constant rate bound $C\cdot(t-s)$ if $\rho = C$ and $\sigma = 0$, respectively, the rate-latency wise bound $R\cdot(t-s-T)$ if $\rho = R$ and $\sigma = RT$. However, in this thesis, we will lay our focus mainly onto the modeling of the flow transformation instead of the service bound. So for the ease of exposition, unless specified we assume to know the constant rate bound for the MGF of a service process, in particular of the time varying work conserving servers, i.e., $M_{S(s,t)}(-\theta) \leq e^{-\theta C(t-s)}$. This assumption is also reasonable in real world, because in many cases it is natural to build up a service with a relative constant capacity and the fluctuation is generated by the (cross) traffic. Now consider a slightly generalized version of a constant rate server - a memoryless on-off server. We first denote $S(s,t) = \sum_{i=s+1}^{t} S_i$, where S_i's are *i.i.d.* random variables and S_i equals either to R in ON state or 0 in OFF state. The MGF of $S(s,t)$ follows as

$$\begin{aligned} M_{S(s,t)}(-\theta) &= \left(M_{S_1}(-\theta)\right)^{(t-s)} \\ &= e^{-\theta\left(-\frac{1}{\theta}\log M_{S_i}(-\theta)\right)(t-s)}. \end{aligned}$$

We can see that $\rho = -\frac{1}{\theta}\log M_{S_i}(-\theta)$ for $\theta > 0$. $M_{S_i}(-\theta)$ is known if know about the service process, e.g., Bernoulli with ON probability p at each time unit, $M_{S_i}(-\theta) = 1 - p + pe^{-\theta R}$, where R is the rate in ON state. Alert reader will find that the bounding rate ρ is in fact a function of θ. Given the description of service process, we can construct the ρ function. Nevertheless, we may not simply assume to know ρ's value, e.g. $\rho = 4.0Mb/s$, the service information is implied in θ. If we do that, θ will be determined, which will increase the computational complexity, or possibly loosen the results of the performance analysis.

2.3.3 Stochastic Analysis

Stochastic Single Node Performance Bounds

Based on the stochastic arrival and service curve definitions, which are sample path bounds, we get a direct stochastic extension of the performance bounds from Theorem 1 as following (see also [28]).

Theorem 4. *(Stochastic Performance Bounds Given Sample Path Bounds of Arrivals and Services) Consider a flow $A(t)$ with an arrival curve $\alpha^{\varepsilon_\alpha}$ traversing a system S that offers a stochastic service curve $\beta^{\varepsilon_\beta}$. Then we obtain the following stochastic performance bounds for all $t \geq 0$ and $0 \leq s \leq t$*

$$\begin{aligned}
&\textit{backlog: } Pr(b(t) \leq v(\alpha^{\varepsilon_\alpha}, \beta^{\varepsilon_\beta})) \geq 1 - \varepsilon_\alpha - \varepsilon_\beta \ , \\
&\textit{delay: } \forall t : Pr(d(t) \leq h(\alpha^{\varepsilon_\alpha}, \beta^{\varepsilon_\beta})) \geq 1 - \varepsilon_\alpha - \varepsilon_\beta \ , \\
&\textit{output: } Pr\left(D(s,t) \leq \alpha^{\varepsilon_\alpha} \oslash \beta^{\varepsilon_\beta}(t-s)\right) \geq 1 - \varepsilon_\alpha - \varepsilon_\beta \ .
\end{aligned}$$

The proof directly follows the same arguments as for Theorem 1 together with using Boole's inequality when jointly consider the sample path violations of the arrival curve and service curve.

It should be noted that under the stochastic service curve definition being used here the concatenation of nodes is problematic without further assumptions. In particular, the violation probabilities for concatenated service curves are time-dependent and can therefore be made equal to one, which makes the guarantees of the concatenated service curve void. Several resorts have been proposed in the literature, the most obvious being the introduction of time-scale bounds which avoids the degeneration of the service curve guarantee for large time durations. We refer the reader to the very good discussion about these issues in [28]. In this thesis, we stay with the straightforward definition of a stochastic service curve, which suffices for our purposes.

For the case of assuming statistical independence between arrivals and services, the MGF bounds, as discussed in Section 2.3.1, will be a better option to derive the performance bounds. Next, we exhibit the derivation of the stochastic backlog, delay and output bounds. We point out that the basic argument here will be used throughout this thesis.

From Eq. (2.1) in the definition of the backlog process we have

$$\begin{aligned}
B(t) &= A(t) - D(t) \\
&\leq A(t) - A \otimes S(t) \\
&= A(t) - \inf_{0 \leq s \leq t} \{A(0,s) + S(s,t)\} \\
&= \sup_{0 \leq s \leq t} \{A(s,t) - S(s,t)\} \ .
\end{aligned}$$

Thus we have the following probability inequality when we measure the event

that backlog $B(t)$ over an upper bound b.

$$\begin{aligned} Pr(B(t) \geq b) &\leq Pr\left(\sup_{0\leq s\leq t}\{A(s,t) - S(s,t)\} \geq b\right) \\ &\leq \sum_{0\leq s\leq t} Pr\left(A(s,t) - S(s,t) \geq b\right) \\ &\leq \sum_{0\leq s\leq t} e^{-\theta b} E\left[e^{\theta(A(s,t)-S(s,t))}\right]. \end{aligned}$$

In the second line we used Boole's inequality (also known as the union bound), through which we collect all the probabilities that for changing $s \in [0,t]$ the event $A(s,t) - S(s,t) \geq b$ is true. This convenient treatment leads to although loose bounds for some cases, gives us however much freedom to seek the probability bound for a wide range combinations of arrivals and services in terms of this sample path event. A tighter bound is obtained in [47] by representing this sample path event as bounded stopping time and constructing martingales. But the result only applies to the single node case and assumes to know more information about the arrival and service processes. In this thesis we will use the Boole's inequality in order to make our results suit more cases especially the multiple nodes analysis. Since we want to build up the models of flow transformation and provide performance bounds, we concern more applicability than the tightness of the results.

As following steps, one can either convole the distribution bounds of $A(s,t)$ and $S(s,t)$, or, like in the third line shown, more elegantly use the Chernoff bound for $\theta > 0$. Then like many literatures, we assume the statistical independence between A and S, or, use Hölder's inequality. We do the former and for all $\theta > 0$ get

$$\begin{aligned} Pr(B(t) \geq b) &\leq \sum_{0\leq s\leq t} e^{-\theta b} E\left[e^{\theta A(s,t)}\right] E\left[e^{-\theta S(s,t)}\right] \\ &\leq e^{-\theta b} \sum_{0\leq s\leq t} e^{-\theta(C-r(\theta))(t-s)} \\ &\leq e^{-\theta b} \lim_{t\to\infty} \sum_{0\leq s\leq t} e^{-\theta(C-r(\theta))(t-s)} \\ &= e^{-\theta b} \frac{1}{\theta(C-r(\theta))}, \end{aligned}$$

if we assume that $M_{A(s,t)}(\theta) \leq e^{\theta r(\theta)(t-s)}$ and $M_{S(s,t)}(-\theta) \leq e^{-\theta C(t-s)}$ in

the second line and let t go to infinity. For the non-triviality of the bounding violation probability, we need it smaller than 1, or, at least the geometric series part should be bounded. Otherwise ∞ can also be a backlog bound ($Pr(B(t) > \infty)$ satisfies the inequality). That means the system is instable. The stability condition is then $C > r(\theta)$.

Now we derive the stochastic delay bound. We argue by analogy. Because we define delay $W(t)$ at any time t as the infimum of all τ's such that $A(t-\tau) \leq D(t)$, we have $Pr(W(t) \geq d) = Pr(A(t-d) > D(t))$. Then we get

$$\begin{aligned} & Pr(W(t) \geq d) \\ \leq \; & Pr\left(A(t-d) > \inf_{0\leq s\leq t}\{A(0,s) + S(s,t)\}\right) \\ = \; & Pr\left(\sup_{0\leq s\leq t-d}\{A(s,t-d) - S(s,t)\} > 0\right) \\ \leq \; & \sum_{0\leq s\leq t-d} Pr(A(s,t-d) - S(s,t) > 0) \\ \leq \; & e^{-\theta C d}\frac{1}{\theta(C-r(\theta))}\,, \end{aligned}$$

if we assume to have the same MGF bounds as for calculating backlog bound. Similarly, we calculate the MGF bound for the departure process $D(s,t)$. The sample path argument is

$$\begin{aligned} D(s,t) \; & \leq \; D(t) - \inf_{0\leq u\leq s}\{A(0,u) + S(u,s)\} \\ & \leq \; \sup_{0\leq u\leq s}\{A(u,t) - S(u,s)\}\;. \end{aligned}$$

Then we get

$$E\left[e^{\theta D(s,t)}\right] \leq e^{\theta r(\theta)(t-s)}\frac{1}{\theta(C-r(\theta))}\,.$$

This represents the $\sigma(\theta), \rho(\theta)$ constraints.

We conclude the above derivation of the performance bounds as following theorem.

Theorem 5. *(Stochastic Performance Bounds Given MGF Bounds of Arrivals and Services) Consider a node that offers a dynamic server $S(s,t)$ for any time $0 \leq s \leq t$. Assume the flow $A(s,t)$ traversing the node has MGF bound $M_{A(s,t)}(\theta) \leq e^{\theta r(\theta)(t-s)}$ for any $\theta > 0$. We also assume the MGF (Laplace) bound on the service as $M_{S(s,t)}(-\theta) \leq e^{-\theta C(t-s)}$ for any $\theta > 0$. Then we*

obtain the following stochastic performance bounds for all $t \geq 0$

$$\begin{aligned}
&\textit{backlog: } Pr\left(B(t) \geq b\right) \leq e^{-\theta_1 b} K_1 \ , \\
&\textit{delay: } Pr\left(W(t) \geq d\right) \leq e^{-\theta_2 C d} K_2 \ , \\
&\textit{output: } M_{D(s,t)}(\theta_3) \leq e^{\theta_3 r(\theta_3)(t-s)} K_3 \ , \\
&\textit{where } K_i = \frac{1}{\theta_i (C - r(\theta_i))} \ , \ i = 1, 2, 3 \ .
\end{aligned}$$

It is worthy to note that when we use the same θ instead of $\theta_1, \theta_2, \theta_3$, we reveal that with the same violation probability $d = \frac{b}{C}$. This is an approximative counterpart to the average analysis. At the same time, the factor (K_3) also plays an important role for bridging the MGF bounds of arrivals and departures. We also note that for many cases we exactly know the MGF functions of the arrival and service processes, we thus directly use them as the MGF bounds. When assuming the MGF bound of the arrivals we ignore σ part for simplicity of the exposure. It is reasonable to assume so for the most traffic cases, e.g. for Poisson arrivals $A(s,t)$, $M_{A(s,t)}(\theta) = e^{\lambda(t-s)(e^{\theta}-1)}$. But after being served, the departures MGF bound has that σ part (transform K_3), which should be applied when necessary.

Stochastic Convolution-Form Representation of Networks

In Section 2.2.3 we have shown that the convolution of the service curves of each node represents the service curve of the tandem network of them (Theorem 3). This result can not simply carry over to the stochastic setting by the same argument stated in the proof of Theorem 3. The difficulty is discussed in [28]. Consider a two nodes network with stochastic service curves β_1, β_2 with violation probabilities $\varepsilon_1, \varepsilon_2$ as defined as Eq. (2.14), such that

$$\begin{aligned}
Pr\left(D_1(s) \geq \inf_{0 \leq u \leq s} \{A_1(u) + \beta_1(u-s)\}\right) &\geq 1 - \varepsilon_1 \\
Pr\left(D_2(t) \geq \inf_{0 \leq \tau \leq t} \{A_2(\tau) + \beta_2(t-\tau)\}\right) &\geq 1 - \varepsilon_2 \ .
\end{aligned}$$

Assume at $\overline{\tau}$ we get the infimum of the second equation. As for a given t, $A_2(t) = D_1(t)$ is random, therefore $\overline{\tau}$ is a random variable. We can not simply use the first inequality by replacing s with $\overline{\tau}$. To solve this problem, we can construct a sample path bound of $D_1(s)$ which implies the event that $D_1(\overline{\tau})$ is the same bounded. But by invoking Boole's inequality the violation probability for this construction has the form of $t\varepsilon$. For a big t, the result

is trivial. To avoid this, many attempts are done. One can bound the time scale by T such that the violation probability is $T\varepsilon$ [98], or use the stochastic adaptive service guarantee [28]. They are too difficult to deal with. One can also construct the sample path service curve through losing a bit rate in the service and gain a non-trivially bounded sum of violation probabilities. It is defined in [43] as

$$Pr\left(\sup_{0\leq s\leq t}\left\{A\otimes\left[\beta-\rho(t-s)-\sigma\right]^{+}(s)-D(s)\right\}>0\right)$$
$$\leq\sum_{s=0}^{t}Pr\left(D(s)<A\otimes\left[\beta-\rho(t-s)-\sigma\right]^{+}(s)\right)\ .$$

Letting $t\to\infty$ a violation probability is obtained as $\varepsilon^{\rho}(\sigma)=\sum_{s=0}^{\infty}\varepsilon(\sigma+\rho s)$ whose integral from 0 to ∞ should be ensured to be bounded. Then denote $\beta^{-\rho}(t)=\beta(t)-\rho t$ together with $\varepsilon_i^{\rho}(\sigma_i)$, we get the service curve for the n nodes tandem network

$$\beta_{net}(t)=\left[\beta_1\otimes\beta_2^{-\rho}\otimes\cdots\otimes\beta_n^{-(n-1)\rho}-\sigma\right]^{+}(t)\text{ with} \tag{2.16}$$
$$\varepsilon_{net}(\sigma)=\inf_{\sigma_1+\cdots+\sigma_n=\sigma}\left\{\varepsilon_1^{\rho}(\sigma_1)+\cdots+\varepsilon_{n-1}^{\rho}(\sigma_{n-1})+\varepsilon_n(\sigma_n)\right\} \tag{2.17}$$

It is shown in [27] that using this stochastic network service curve the derived delay bound scales as $\Theta(n\log n)$.

The problem is easy to solve in case we use the bivariate representation of the dynamic servers. First, we do not need any sample path argument when iteratively convolving the dynamic servers. Second, in Section 2.3.3 we see that the derivation of the performance bounds involves the MGF bound, whose exponential form transforms the addition to a multiplication that converges faster for the values smaller than 1.

Theorem 6. *(Dynamic Server of Concatenated Nodes) Consider a flow $A(t)$ traversing a set of network nodes with dynamic servers $S_i(s,t), i=1,\ldots,N$, $N\geq 1$ in sequence. The concatenation of these nodes offers a dynamic server $S_{net}(s,t)=S_1\otimes S_2\otimes\cdots\otimes S_n(s,t)$ to the flow such that the departure process $D(t)$ satisfies*

$$D(t)\geq A\otimes S_{net}(t)\ .$$

Proof. The proof follows the same argument as shown in Theorem 3 by noting that $D_i(t)=A_{i+1}(t)$.

$$D(t)=D_n(t)\quad\geq\quad A_n\otimes S_n(t)$$

$$
\begin{aligned}
&= D_{n-1} \otimes S_n(t) \\
&\geq (A_{n-1} \otimes S_{n-1}) \otimes S_n(t) \\
&\geq \cdots \\
&\geq (\cdots((A \otimes S_1) \otimes S_2) \otimes \cdots \otimes S_n)(t) \\
&= A \otimes (S_1 \otimes S_2 \otimes \cdots \otimes S_n)(t) .
\end{aligned}
$$

□

We recursively used the isotonicity and associativity of the min-plus convolution. This theorem enables the derivation of performance bounds for a multiple nodes network as that for a single node case. Note, the commutativity of the min-plus convolution does not hold for the bivariate functions, the change of order will lead to different dynamic server of the network. In this thesis, we mainly use the dynamic server rather than the stochastic service curve to represent the network service bound.

Scheduling Modeling with SNC

We provide the ways to model different scheduling schemes among multiplexing flows with SNC in this section. A basic assumption is still that the flow is locally FIFO. And the key idea is as stated in Section 2.2.3, to characterize the leftover service, either in form of stochastic curve, or constructing its bivariate representation.

Liebeherr *et. al.* [102] introduced the so-called δ-scheduler to model the static priority (SP), blind multiplexing (BMUX), FIFO and earliest deadline first (EDF) scheduling algorithms in the sense of stochastic leftover service curve. Consider N multiplexing flows. A δ-scheduler is defined as a work-conserving locally FIFO scheduling algorithm if there exist constants $\{\delta_{ij}\}$ such that an arrival at time t from flow i has precedence precisely over those arrivals from flow j that occur after $t + \delta_{ij}$. Then for a constant server with capacity C, the stochastic leftover service is given. And if we only consider $N = 2$ here for example, it is

$$\beta_1(t) = [Ct - \alpha_2(t - x + \delta_{12}(x))]^+ 1_{\{t>x\}}$$

with violation probability $\varepsilon_{s1}(\sigma) = \varepsilon_{\alpha 2}(\sigma)$. Here, α_2 is the sample path stochastic arrival curve defined as Eq. (2.10). The indicator function $1_{cond} = 1$ if $cond = true$, $1_{cond} = 0$ otherwise. As it is shown in [102] that when analyzing the end-to-end delay bounds, the number of network nodes dominates the scheduling algorithms (the schedulings differ slightly from each

other when number of nodes not too small, e.g., 5), we simply lay our focus on the scheduling algorithms we need in this thesis, i.e., BMUX, SP, FIFO. To avoid the difficulty we deal with these by constructing the leftover dynamic server instead of using this leftover service curve. Fidler [61] extended the blind multiplexing (see Eq. (2.8) in Section 2.2.3) to the dynamic server case. The flow of interest ($A_1(t)$) at a work conserving server $S(s,t)$ receives a "leftover" dynamic server after S being used by cross flow ($A_2(t)$)

$$S_1(s,t) = [S(s,t) - A_2(s,t)]^+ \ . \tag{2.18}$$

We now extend the leftover service curve for FIFO scheduler shown in Section 2.2.3 Eq. (2.2.3) to its dynamic server representation. The corresponding proof can also recover the BMUX case. The leftover dynamic server for a work conserving FIFO scheduler is defined as

$$S_1(s,t) = [S(s,t) - A_2(s,t-x)]^+ 1_{\{t-x>s\}} \ . \tag{2.19}$$

The proof follows relating the departure to the arrival at the start time of the last busy period of the server. First fix t. We define $\tau, 0 \leq \tau \leq t$ as the start time of the last busy period of the node before t. Now denote $A(t) = A_1(t) + A_2(t)$ and $D(t) = D_1(t) + D_2(t)$, then $A(\tau) = D(\tau), A_1(\tau) = D_1(\tau)$. Now we consider two cases.

(1) If at time t the node is not busy, then we have

$$D_1(t) = A_1(t) = A_1(t) + S_1(t,t) \ . \tag{2.20}$$

(2) Otherwise, at time t the node is busy, we use $D(t) = D(\tau) + S(\tau,t)$ and have

$$D_1(t) = A_1(\tau) + S(\tau,t) - (D_2(t) - A_2(\tau)) \ . \tag{2.21}$$

Because $D_2(t) \leq A_2(t)$ and we know $D_1(t) \geq D_1(\tau) = A_1(\tau)$, we get

$$D_1(t) \geq A_1(\tau) + [S(\tau,t) - A_2(\tau,t)]^+ \ .$$

Together with Eq. (2.20) and let $x = 0$, the leftover dynamic server given in Eq. (2.18) is proved. In fact, assuming FIFO we know more information than BMUX. That means, for Eq. (2.21) we can find a tighter bound for $D_2(t)$ instead the trivial one $A_2(t)$. There should exist some time point before t, at which the arrivals A_2 have already reached or overtaken the amount of $D_2(t)$. We should find out that time point and the corresponding arrival amount. Define

$$u := \sup\{v : A(v) \leq D(t)\} \ . \tag{2.22}$$

Thus $u \leq t$ and

$$A(u) \leq D(t) \text{ and } A_r(u) > D(t) \ , \tag{2.23}$$

where $A_r(u) = \inf_{v>u}\{A(v)\}$. Clearly, for a continuous time and data model we can get $A(u) = D(t)$. The consideration here can also cover the discrete data model, i.e., there is possibly a left-continuous break point of A at time t. Now we prove Eq. (2.23) is also true for the individual flow. Because the node is busy at $u < t$. From $A_1(u) + A_2(u) \leq D_1(t) + D_2(t)$ we claim that

$$A_1(u) \leq D_1(t) \ . \tag{2.24}$$

By contradiction, if it is not true, i.e., $A_1(u) > D_1(t)$, we have $A_2(u) < D_2(t)$. That means, some data from flow 2 arrived after u and departed by t (locally FIFO), while some data from flow 1 arrived by u have not yet departed. That contradicts the FIFO assumption. It is analogous to claim that

$$A_{2r}(u) \geq D_2(t) \ . \tag{2.25}$$

Otherwise, $A_{2r}(u) < D_2(t)$. Then, from the definition of $A_r(u)$ and u together with the fact that $u < t$, for any left continuous sample path of A_2, there exists some $v \in (u, t]$ such that $A_2(v) < D_2(t)$. For such $v \in (u, t]$ we know $A(v) \geq A_r(u) > D(t)$ from Eq. (2.23). It follows that $A_1(v) + A_2(v) > D_1(t) + D_2(t) > D_1(t) + A_2(v)$ and thus $A_1(v) > D_1(t)$. This, again, contradicts the FIFO assumption.

Recall that $D(t) = A(\tau) + S(\tau, t)$, we have $D(t) \geq A(\tau)$, thus $\tau \leq u$. Then we can find some $t - x$ that falls into (u, t), such that $A_2(t - x) \geq A_{2r}(u)$. From Eq. (2.25) we get $A_2(t - x) \geq D_2(t)$ and use it as a tighter bound into Eq. (2.21) and derive

$$D_1(t) \geq A_1(\tau) + S(\tau, t) - A_2(\tau, t - x) \ . \tag{2.26}$$

Here $t - x > u$ thus $t - x > \tau$. And we know $D_1(t) \geq A_1(\tau)$, which shows that

$$D_1(t) \geq A_1(\tau) + [S(\tau, t) - A_2(\tau, t - x)]^+ \ .$$

When $t - x$ falls into $[0, u]$, i.e., $u \geq t - x$, it is not FIFO and we have a trivial dynamic server 0, such that

$$D_1(t) \geq A_1(u) = A_1(u) + [S(u, t) - A_2(u, t - x)]^+ 1_{\{t-x>u\}} \ .$$

Therefore, we conclude all cases and get the leftover dynamic server as shown in Eq. (2.19).

2.4 Express Flow Transformations in Network Calculus

A major limitation of the scope of convolution-form networks (see Theorem 3,6 and Eq. (2.16)) is caused by an underlying assumption that flows are transported unaltered, for instance lossless, over the network. Concretely, it is challenging whether many networks, in which data flows are being transformed on the way to their destinations, can be expressed in convolution-form. Ordinary examples are networks with lossy links, dynamic routing or load balancing, and more sophisticated ones are wireless sensor networks with their typical in-network processing, P2P content distribution systems, software defined networks with their agile flow action definitions, media streaming applications with some transcoding happening inside the network (e.g., to accommodate heterogeneous multicast receivers), or even network coding scenarios and distributed real-time systems with heterogeneous resources. Without the convolution-form representation, such networks can be still in principle analyzed with network calculus by conducting an additive node-by-node analysis. But on one hand, the resulting network queueing measures would scale, as evidenced earlier, poorly; on the other hand, this can only apply with using deterministic network calculus.

Many attempts have been done to deal with flow transformation but rare can yet achieve the desirable convolution-form representation. One outstanding work has been proposed in purely deterministic settings [64]. By introducing the so-called *data scaling element* (also called as *scaler*) to model actual transformation processes, and by controlling the movement of these elements in the network, the authors showed that the exact scaling properties from the deterministic network calculus are preserved. This idea is the origin of this thesis.

Next, we provide the necessary definitions and results for introducing scaling elements into network calculus models as presented in [64].

Definition 12. (Scaling Function) A scaling function $S \in \mathcal{F}$ assigns an amount of scaled data $S(a)$ to an amount of data a. See illustration in Figure 2.6.

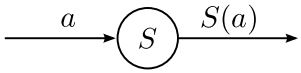


Figure 2.6: Data scaling element.

We follow the denotation of the scaling function used in [64]. Although it is also used to denote service process, we will not be confused when we read the following chapters, because in case we use scaling function S, we usually correspondingly need the service curve β instead of service process. When we talk about service process denoted by S, we will also need the stochastic representation of the scaling element by using either stochastic scaling curves $\overline{S}, \underline{S}$ or scaling process $\mathbf{X}$ and denote the scaling element with $\mathcal{S}$.

As can be seen from the definition of the scaling function, it is a very general concept for taking into account data transformations in a network calculus model. Note, however, that it does not model any queueing effects – scaling is assumed to be done infinitely fast. Queueing related effects are still modeled in the service curve element of the respective component.

Corollary 1. *(Inverse Scaling Functions) Given a bijective scaling function $S \in \mathcal{F}$ it follows that its inverse scaling function S^{-1} is a scaling function, too.*

Inverse scaling functions play a role in transforming systems into alternative systems that can be analysed more efficiently. More details are to follow.

Definition 13. (Scaling Curves) Given a scaling function S, two functions $\underline{S}, \overline{S} \in \mathcal{F}$ are minimum and maximum scaling curves of S if for all $0 \leq a \leq b$

$$S(b) - S(a) \geq \underline{S}(b-a) \,,$$
$$S(b) - S(a) \leq \overline{S}(b-a) \,.$$

Here we slightly change the expression from [64], which does not differ from this definition but may set up a trap when extending them to the stochastic setting.

Corollary 2. *(Sub- and Super-Additive Closure) Consider a scaling function S with minimum and maximum scaling curve $\underline{S}$ and $\overline{S}$. The super-additive closure of $\underline{S}$ is a minimum scaling curve of S and the sub-additive closure of $\overline{S}$ is a maximum scaling curve of S.*

A function f is sub-additive respectively super-additive if $f(x+y) \leq f(x)+f(y)$ respectively $f(x+y) \geq f(x)+f(y)$ for all x, y. The sub-additive closure of f is defined as $\inf_{n\geq 1}\left\{f^{(n)}\right\}$ where $f^{(n)}$ is the n-fold min-plus self-convolution of f with $f^{(1)} = f, f^{(2)} = f \otimes f, f^{(3)} = f \otimes f \otimes f$ and so on. The super-additive closure of f is defined as $\sup_{n\geq 1}\left\{f^{(n)}\right\}$ where $f^{(n)}$ is the n-fold max-plus self-convolution of f. As a consequence of this corollary, we consider only sub-additive minimum scaling curve and super-additive maximum scaling curve.

Figure 2.7: Alternative systems.

Corollary 3. *(Inverse Scaling Curves) Consider a bijective scaling function S and let $\underline{S}$ and $\overline{S}$ be the respective minimum and maximum scaling curves. If $\underline{S}$ and $\overline{S}$ are bijective , a valid maximum scaling curve of the inverse scaling function $\overline{S^{-1}}$ is $\underline{S}^{-1}$ and a valid minimum scaling curve of the inverse scaling function $\underline{S^{-1}}$ is $\overline{S}^{-1}$.*

Theorem 7. *(Scaled Servers – Alternative Systems) Consider the two systems in Fig. 2.7 and let $F(t)$ be the input function. System (a) consists of a server with minimum service curve β and maximum service curve γ whose output is scaled with scaling function S and system (b) consists of a scaling function S whose output is input to a server with minimum and maximum service curve β_S and γ_S, respectively. Given system (a) the lower and upper bounds of the output function of system (b), that are $S(F) \otimes \beta_S$ and $S(F) \otimes \gamma_S$ are also valid lower and upper bounds for the output function of system (a) if*

$$\beta_S = \underline{S}(\beta) \ ,$$

$$\gamma_S = \overline{S}(\gamma) \ ,$$

where $\overline{S}$ and $\underline{S}$ are the respective scaling curves of S. Given system (b), the lower and upper bounds for the output function of system (a), that are $S\,(F \otimes \beta)$ and $S\,(F \otimes \gamma)$ respectively, hold also for system (b) if S is bijective and

$$\beta = \underline{S^{-1}}\,(\beta_S) \ ,$$

$$\gamma = \overline{S^{-1}}\,(\gamma_S) \ ,$$

where $\overline{S^{-1}}$ and $\underline{S^{-1}}$ are the respective scaling curves of S^{-1}.

This means in effect that performance bounds for system (b) under this assumption are also valid bounds for system (a) and vice versa, because these bounds are derived based on the bounds of the output function. We can thus effectively move a scaling function, e.g., in front of a service curve element as long as we transform the respective service curve using the minimum scaling curve of the scaling element. In [64], it is also shown that bounds computed

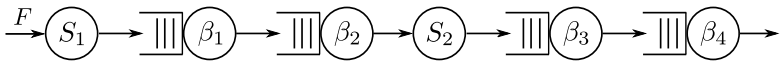


Figure 2.8: A network model with servers and scaling elements.

in the alternative system, i.e., after shifting the scaling elements, remain tight. Now the utility of Corollary 3 is meaningful, since we can observe that it enables us to compute scaled versions of the service curves when scaling elements are shifted over service curve elements in the direction of the data stream (behind the service curve).

The following corollary states the effect scaling has on arrival constraints of a traffic flow.

Corollary 4. *(Arrival Constraints under Scaling) Let F be an input function with arrival curve α that is fed into a scaling function S with maximum scaling curve $\overline{S}$. An arrival curve for the scaled output from the scaling element is given by*

$$\alpha_S = \overline{S}(\alpha) \ .$$

If S is bijective and S^{-1} has a maximum scaling curve $\overline{S^{-1}}$, then given an arrival curve for the scaled output process α_S can be given as

$$\alpha = \overline{S^{-1}}\,(\alpha_S) \ .$$

The particular advantage of above results now becomes clear. When analyzing the performance bounds of a network with scaling element, we can shift the scaling elements to the ingress or the egress. And along the path of shifting, we scaled the service curves this scaling element meets. Consider the following tandem network which is modeled as a sequence of a scaling function S_1, two nodes with service curves β_1, β_2, a further scaling function S_2, and another two nodes β_3, β_4. See Figure 2.8. We can alternatively do the following operations:

(1) Move S_1 and S_2 to the egress. First, we convolve β_1 and β_2, respectively β_3 and β_4 using Theorem 3. Let $\beta_{12}(t) = \beta_1 \otimes \beta_2(t)$ and $\beta_{34}(t) = \beta_3 \otimes \beta_4(t)$. Second, we shift S_2 to the egress and transform the service curve $\beta_{34}(t)$ to the scaled version $\underline{S_2^{-1}}(\beta_{34}(t))$. Third, we convolve β_{12} and $\underline{S_2^{-1}}(\beta_{34}(t))$, then shift S_1 to the egress. Doing that the service curve of the network becomes

$$\beta(t) = \underline{S_1^{-1}}(\beta_{12} \otimes \underline{S_2^{-1}}(\beta_{34})(t)) \ .$$

We can treat the network as a single node server and derive the performance bounds using Theorem 1. Since in [64] the authors did not discuss the concatenation of multiple scaling functions in detail, we ignore the case first move S_1 to the egress and concatenate S_1 and S_2 and move. We will discuss this in the next two chapters. Another case to avoid is moving S_1 to the back of β_{12} and $\underline{S_2^{-1}}(\beta_{34})$ one by one, because it is generally worse than firstly convolving the service curves and then move when we assume $\underline{S_1^{-1}}$ to be super-additive.

(2) Move S_1 and S_2 to the ingress. It is analogous to (1) that we first convolve the service curves. Second, we move S_2 to the front of β_{12} and get the service curve of the network

$$\beta(t) = \underline{S_2}(\beta_{12}) \otimes \beta_{34}(t) \ .$$

Third, we use Corollary 4 iteratively to derive the arrival curve of the scaled arrivals and compute the performance bounds, again, using Theorem 1. Interestingly, if two successive scaling functions are mutually inverse functions, when they meet, they cancel out each other directly. For example, when a pair of symmetric scaling functions - encoder S_E and decoder S_D concatenate each other, we have $S_D(S_E(a)) = a$.

Before this generalized modeling of flow transformation in network calculus, there are other works that attempt to model flow transformation but rather for specialized examples, like data loss, routing, or event stream translation. Chang makes the first attempt by introducing an element called *router* to model data routing [36] respectively splitting, a case of flow transformation, which has a single data input $A(t)$, a control input P and an output $D(t)$ such that $D(t) = P(A(t))$ for all t. The control input determines which packets are selected to appear at the output. He derives the effect a router has on arrival constraints using the (σ, ρ) format like shown in Section 2.2.1 (δ, γ-upper constraints on router), but convolution-form expressions of end-to-end service are not investigated. In comparison, a router P is a scaling function with maximum scaling curve $\overline{P}(a) = \gamma a + \delta$, and the routed (σ, ρ)-upper constrained arrivals has arrival curve $\gamma\rho t + \gamma\sigma + \delta$, the same as Corollary 4. Interestingly, Chang extends the deterministic router to the MGF bound representation and also derives the MGF bound of its output (Lemma 7.5.1 in [36]), which is an extension of router in the stochastic settings. However, the application of this result in [36] is specialized, closely related to effective bandwidth theory, and still does not consider the convolution-form of the end-to-end service. But we will generalize this result in Section 4.1.1 for providing the stochastic end-to-end performance analysis.

Another important attempt is to model a special case of traffic regulating, i.e., the so-called *traffic clipping*, using *clipper*, which is introduced by Cruz [55]. The definition is a kind of "black-box" description. The authors only observe the input and output and the goal is to find out how to drop packets such that the output satisfies the specified arrival curve after clipping. Given $f \in \mathcal{F}$, a f-clipper is defined as a network element which satisfies two conditions: (1) the departure is less than or equal to the arrival at all time t; (2) f is the arrival curve of the accumulated departures. In [36] and [95] there are also some extended results. In particular, Chang ([36, 37]) extends the f function to a bivariate function $F(s,t)$ and provides the construction of the departure $D(t)$ such that how to drop the packet at time t is thus decided. Let $D(t) = (\min(A, F))^*(0,t)$, where F^* means closure of F defined as $\lim_{n\to\infty}(\min_{n\geq 0}\{F^{(n)}\})$, $F^{(0)}(s,t) = 0$ if $s = t$ and ∞ otherwise, we have

$$D(t) = \min\left\{D(t-1) + a(t), \min_{0\leq s<t}\{D(s) + F(s,t)\}\right\} .$$

Then the loss $L(t) = a(t) - d(t)$ at time t is

$$L(t) = \max\left\{0, D(t-1) + a(t) - \min_{0\leq s<t}\{D(s) + F(s,t)\}\right\} .$$

Dropping packets according to this equation we will get a departure with the given bound F. An example to construct such a clipper given $F(s,t) = \rho(t-s) + \sigma$ is to use a work conserving server with constant capacity ρ and a finite buffer σ. All the results about clipper orient themselves towards constructing the clipper such that the traffic is regulated according to the specified bounds. They are not general and provide no further results on the multiple nodes analysis, hence no expression of the network service when the traffic is clipped along the path to the destination in the network.

Maxiaguine *et. al.* introduce the so-called workload curves [109, 110] into the real-time calculus, a recent customization of network calculus for hard real-time systems [139]. Workload curves are a special scaling element that translates an event stream into specific requirements for a certain resource, e.g., CPU time or link bandwidth, thus bridging between different subsystems with heterogeneous resources. While [143] focuses on how to actually compute such workload curves (e.g., based on finite-state machine representations of the actual processing components) and it basically just applies a variant of Chang's router element, it is interesting to note that this application of scaling opened up an active research avenue for distributed real-time systems. Convolution-form expressions of end-to-end service are not yet dealt

with in that work.

An interesting and somewhat related work can be found in [154]. In that work, a calculus for the so-called information-driven networks is proposed. The authors firstly use the Shannon entropy function to get information from a given flow and then define the arrival curves and service curves relating with this information flow instead of the data flow. Similarly, the calculus also deals with flow processing inside the network, but represents a different direction of research as it centers around the notion of information, whereas we are willing to concern with the transport of data (albeit possibly transformed inside the network).

Another two works are introduced in [23] and [9]. Both of them can be viewed as the application of the scaling function and scaling curves. The former defines a function $\mathcal{P}$ to count the number of packets in a flow $A(t)$ by time t and accordingly defines the upper and lower bound of $\mathcal{P}$, called packet curves. It is again, a specialized definition for packetization, and the application is not clear either. The latter also uses the same idea as the scaling element, where it is called flow control functions, to model the control done by the controller to the flows in the Software Defined Networks (SDN). The only contribution is that it shows the modeling power of the scaling element. The calculation of the backlog is repeating the existing results in network calculus. The key question to be answered is how to know the bound of the flow control functions according to the SDN flow table, which is not answered. Again, both works do not consider the convolution-form expression of the networks with flow transformation components along the flow path.

In fact, some existing works in network calculus have already touched the flow transformation but not been aware of and further generalized it, because they only study the individual application examples of the flow transformation and limit their scope within the corresponding application domain. Data loss, for example, is studied. Most work combine this loss behavior with the server model. One is given in [6], where the server model twists with deadline assignment and considers only certain constant loss fraction $1 - \alpha$. The use of α to construct the service curve of the server with loss is unapparent, concretely, the construction of the so-called loss operation L can not lead to a clear relation with α (one special case to show the weakness can exist for a discrete data model and some usual α values as $\alpha \in [0, 1]$). And the meaning of $L(F \otimes \beta)(t)$ used there is also ambiguous. However, this work considers the concatenation of multiple servers with loss. Another work which considers loss in network calculus is [81]. The authors explicitly express the loss operation as an impairment process $I(s, t)$. But in [81], this loss means the loss of service capacity not the loss of data flow, thus not a case of flow

transformation.

So far, we discussed many potential work on the flow transformation in network calculus. Some have shown how to perform an end-to-end analysis in the presence of flow transformations inside the network based on a convolution-form expression of the end-to-end service. In that sense, the scaling element goes beyond the others and also generalizes previous scaling elements to some degree. The scaling element potentially widens the scope of convolution-form networks to model more application scenarios, for example the wireless sensor networks with in-network processing [133]. Nevertheless, the scaling element approach still largely leaves open the limitation in scope of convolution-form networks. The reason lies in the deterministic modeling which can very loosely capture the behavior of networks with stochastic settings. For instance, modeling scaling elements with deterministic bounds can be extremely impractical to capture loss processes in wireless networks, since extreme situations must be accounted for (e.g., all data is lost), which would further lead to trivial results (e.g., zero end-to-end delays). Other impractical situations include scenarios with random routing or load-balancing and so on. In a word, many flow transformation behaviors happen inherently very random. Only considering the extreme situation will be trivial and does not match the nature. It thus becomes clear that in order to accurately capture the inherent stochastic behavior of flow transformations in networks, we must accordingly resort to stochastic modeling techniques. A good reference for doing that is simply the methods used for modeling the random arrival process - bounding function and MGF bound. But the referring is not straightforward. The difference is when we define a process for scaling element, the index set for such a *scaling process* may transfer from time domain to data domain. Besides, since the flow transformation violates the losslessness assumption for defining virtual delay, in other words, the flow we observe at different nodes within the network is always changing, the main challenge still lies in finding the stochastic convolution-form representation of the network service for this changing flow at some observing point, and in reverse, this may also decide the choice of the form to characterize the stochastic scaling element. The modeling power of different choices of the scaling forms will perform differently in diverse applications.

Chapter 3

Stochastic Data Scaling Element - Bounding Functions

In this chapter, we develop the *stochastic* data scaling in the direction of stochastic sample path bounding, which generally expresses the flow transformation with certain random nature and directly adapts the deterministic analysis to provide the stochastic end-to-end performance bounds. We first define the stochastic scaling element and the bounding functions. Then we present the commutation theorem, which enables the convolution-form expression of the networks with the presence of the flow transformations. To validate the results, we apply them to the most important application scenarios of the flow transformation - dynamic demultiplexing and load balancing.

The main contribution is that, after finding a way to support the stochastic sample path descriptions of the scaling element, we keep the alternative system expression (Theorem 7) in the stochastic setting which still allows us to change the order of scalers and servers, and thus get the stochastic convolution-form network service curve. We also show the modeling power of this form of stochastic scaling element and reveal the challenges we might face when analyze concrete applications. The results presented in this chapter are from the joint work with J. Schmitt and I. Martinovic (see [151]).

3.1 Stochastic Data Scaling

We provide fundamental results on stochastic data scaling in this section. We will introduce stochastic versions of minimum and maximum scaling curves. For some results, the deterministic arguments from Section 2.4 can be easily adapted, for others some care needs to be taken, as in the deterministic setting, it is convenient and not too restrictive to assume bijectivity of the scaling functions. First, we need to define however, the notion of a stochastic scaling process of which scaling functions are its realizations.

Definition 14. (Stochastic Scaling Element - Scaling Sample Paths) A stochastic scaling element $\mathcal{S}$ is described by a stochastic process whose ensemble (all possible sample paths or realizations of $\mathcal{S}$) is a set of scaling functions $\{S(a,\omega) : \omega \in I_{\mathcal{S}}\}$, where $I_{\mathcal{S}}$ is some index set. We depict it in Figure 3.1.

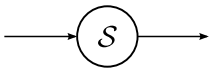


Figure 3.1: Stochastic scaling element.

Next, we provide the definitions for stochastically relaxed versions of maximum and minimum scaling curves.

Definition 15. (Stochastic Scaling Curves) Let $\mathcal{S}$ be a stochastic scaling element, two functions $\underline{S}_{\underline{\varepsilon}}$, $\overline{S}^{\overline{\varepsilon}} \in \mathcal{F}$ are called stochastic minimum and maximum scaling curves of $\mathcal{S}$, respectively, if for all $b \geq 0$

$$Pr\left(\inf_{0\leq a\leq b}\left\{\mathcal{S}(b)-\mathcal{S}(a)-\underline{S}_{\underline{\varepsilon}}(b-a)\right\}\leq 0\right)\leq \underline{\varepsilon}\,,$$
$$Pr\left(\sup_{0\leq a\leq b}\left\{\mathcal{S}(b)-\mathcal{S}(a)-\overline{S}^{\overline{\varepsilon}}(b-a)\right\}\geq 0\right)\leq \overline{\varepsilon}\,.$$

Here, $\underline{\varepsilon}$ and $\overline{\varepsilon}$ denote the violation probabilities for stochastic minimum and maximum scaling curves, respectively.

Note that the stochastic scaling curve properties are defined over sample paths (scaling functions) as realized by the respective scaling process. In the context of stochastic arrival curves this has also been coined as sample-path effective envelope (see Section 2.3.1). We fix the ending point as b such that the space is limited, which facilitates deriving a bounded probability.

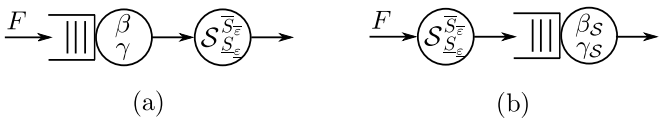


Figure 3.2: Alternative systems under stochastic setting.

In the deterministic case, Corollary 3 provides a way to calculate scaling curves for the inverse scaling function based on knowledge of the scaling curves for the original scaling function. We can directly transfer these results under the following definitions.

Definition 16. (Bijectivity of Stochastic Scaling Elements) A stochastic scaling element $\mathcal{S}$ is said to be bijective if $\forall \omega \in I_{\mathcal{S}}$ the respective scaling function $S(a, \omega)$ is bijective.

Definition 17. (Inverse Stochastic Scaling Element) Given a bijective stochastic scaling element $\mathcal{S}$ with ensemble $\{S(a, \omega) : \omega \in I_{\mathcal{S}}\}$ we define its inverse stochastic scaling element $\mathcal{S}^{-1}$ by the ensemble $\left\{S^{-1}(a, \omega) : \omega \in I_{\mathcal{S}}\right\}$. Here, it is assumed that $\mathcal{S}$ and $\mathcal{S}^{-1}$ are realized together, based on the same random experiment, such that it is justified to state that $\mathcal{S}^{-1} \circ \mathcal{S} = \mathcal{S} \circ \mathcal{S}^{-1} =$ id.

Now we turn to the presumably most important result, the stochastic generalization of the alternative systems theorem.

Theorem 8. *(Stochastically Scaled Servers – Alternative Systems) Consider the two systems in Figure 3.2 and let F be the input function. System (a) consists of a server with deterministic minimum service curve β and maximum service curve γ whose output is scaled by a stochastic scaling element $\mathcal{S}$, for which we have stochastic minimum and maximum scaling curves $\underline{S}_{\underline{\varepsilon}}$ and $\overline{S}^{\overline{\varepsilon}}$ with violation probability $\underline{\varepsilon}$ and $\overline{\varepsilon}$, respectively. System (b) consists of a stochastic scaling element $\mathcal{S}$, which has stochastic minimum and maximum scaling curves $\underline{S}_{\underline{\varepsilon}}$ and $\overline{S}^{\overline{\varepsilon}}$ with violation probability $\underline{\varepsilon}$ and $\overline{\varepsilon}$, and whose output is input to a server with deterministic minimum and maximum service curves $\beta_{\mathcal{S}}$ and $\gamma_{\mathcal{S}}$, respectively.*

(1) Given system (a) the lower and upper bounds of the output function of system (b), that are $\mathcal{S}(F) \otimes \beta_{\mathcal{S}}$ and $\mathcal{S}(F) \otimes \gamma_{\mathcal{S}}$, respectively, are also valid stochastic *lower and upper bounds for the output function of system (a), i.e.,* $\forall t \geq 0$:

$$Pr(\mathcal{S}(F'(t)) \geq (\mathcal{S}(F) \otimes \beta_{\mathcal{S}})(t)) \quad \geq \quad 1 - \underline{\varepsilon}\,,$$

$$Pr(\mathcal{S}(F'(t)) \leq (\mathcal{S}(F) \otimes \gamma_{\mathcal{S}})(t)) \quad \geq \quad 1 - \overline{\varepsilon}\,,$$

if system (b) consists of the same stochastic scaling element as in system (a) together with service curves

$$\begin{aligned} \beta_{\mathcal{S}} &= \underline{S_{\varepsilon}}(\beta)\,, \\ \gamma_{\mathcal{S}} &= \overline{S^{\overline{\varepsilon}}}(\gamma)\,. \end{aligned}$$

(2) Assume bijectivity of $\mathcal{S}$. Given system (b) the lower and upper bounds of the output function of system (a), that are $\mathcal{S}(F \otimes \beta)$ and $\mathcal{S}(F \otimes \gamma)$, respectively, are also valid stochastic *lower and upper bounds for system (b), i.e., $\forall t \geq 0$:*

$$\begin{aligned} Pr((\mathcal{S}(F(t)))' \geq (\mathcal{S}(F \otimes \beta))(t)) \quad &\geq \quad 1 - \underline{\varepsilon}\,, \\ Pr((\mathcal{S}(F(t)))' \leq (\mathcal{S}(F \otimes \gamma))(t)) \quad &\geq \quad 1 - \overline{\varepsilon}\,, \end{aligned}$$

if system (a) consists of the same stochastic scaling element as in system (b) together with service curves

$$\begin{aligned} \beta &= \underline{S^{-1}}_{\underline{\varepsilon}}(\beta_{\mathcal{S}})\,, \\ \gamma &= \overline{S^{-1}}^{\overline{\varepsilon}}(\gamma_{\mathcal{S}})\,. \end{aligned}$$

Here $\underline{S^{-1}}_{\underline{\varepsilon}}$ and $\overline{S^{-1}}^{\overline{\varepsilon}}$ are the respective stochastic scaling curves of inverse stochastic scaling $\mathcal{S}^{-1}$.

Proof. Some preliminary remarks: Note that a stochastic scaling curve $\underline{S_{\varepsilon}}$ only bounds a subset of all possible scaling functions from the stochastic scaling element $\mathcal{S}$. Let us denote that subset by

$$\mathcal{S}_{\underline{S_{\varepsilon}}applies} = \left\{ S(a,\omega) : \omega \in I_{\mathcal{S}}, \forall x \geq 0, \underline{S_{\varepsilon}} \leq S(x+a) - S(x) \right\} \subseteq \mathcal{S}\,.$$

Note that $Pr(\mathcal{S}_{\underline{S_{\varepsilon}}applies}) \geq 1 - \underline{\varepsilon}$, as well as that $\underline{S_{\varepsilon}}$ is a deterministic minimum scaling curve for each scaling function $S(a,\omega) \in \mathcal{S}_{\underline{S_{\varepsilon}}applies}$.

We now start proving the first part of the theorem (going from system (a) to system (b)). We begin with the stochastic lower bound on the output function of the composite system.

For system (a), we know from the minimum service curve property that

$$F' \geq F \otimes \beta\,.$$

Assuming that the stochastic scaling element realizes a scaling function $S(a,\omega)$ $\in \mathcal{S}_{\underline{S_{\underline{\varepsilon}}} applies}$, we can deterministically conclude that

$$\begin{aligned} S(F'(t)) &\geq S(F \otimes \beta))(t) \\ &= S\left(\inf_{0\leq s\leq t}\{F(t-s)+\beta(s)\}\right) \\ &= \inf_{0\leq s\leq t}\{S\left(F(t-s)+\beta(s)\right)\} \\ &= \inf_{0\leq s\leq t}\{S(F(t-s))+S\left(F(t-s)+\beta(s)\right)-S(F(t-s))\} \\ &\geq \inf_{0\leq s\leq t}\left\{S(F(t-s))+\underline{S_{\underline{\varepsilon}}}(\beta(s))\right\} \\ &= \left(S(F)\otimes \underline{S_{\underline{\varepsilon}}}(\beta)\right)(t)\ . \end{aligned}$$

Now consider system (b). Its output function can be bounded as follows:

$$\left(S(F(t))\right)' \geq \left(S(F)\otimes\beta_{\mathcal{S}}\right)(t)\ .$$

If we let $\beta_{\mathcal{S}} = \underline{S_{\underline{\varepsilon}}}(\beta)$ in system (b), we get the same bound on the output function based on the assumption that $S(a,\omega) \in \mathcal{S}_{\underline{S_{\underline{\varepsilon}}} applies}$, which holds with probability $1-\underline{\varepsilon}$. Hence, we obtain $\forall t \geq 0$:

$$Pr(\mathcal{S}\left(F'(t)\right) \geq \left(\mathcal{S}(F)\otimes\beta_{\mathcal{S}}\right)(t)) \quad \geq \quad 1-\underline{\varepsilon}\ .$$

Establishing the connection between the upper bound on the output functions of system (a) and (b) follows as an immediate variation.

Now for the second part of the theorem: going from system (b) to system (a). Again, we start with the lower bound on the output function of the composite system. Similar to above, we introduce the subset of scaling functions for which their inverse adheres to $\underline{S^{-1}_{\underline{\varepsilon}}}$ as

$$\mathcal{S}_{\underline{S^{-1}_{\underline{\varepsilon}}} applies}=\left\{S(a,\omega)\colon \omega\in I_{\mathcal{S}}, \forall x\geq 0, \underline{S^{-1}_{\underline{\varepsilon}}}\leq S^{-1}(x+a)-S^{-1}(x)\right\}\subseteq\mathcal{S}\ .$$

For system (b), we know from the deterministic minimum service curve property that

$$\left(S\left(F\right)\right)' \geq S(F)\otimes\beta_{\mathcal{S}}\ .$$

Assuming that the stochastic scaling element realizes a scaling function $S(a,\omega)$ $\in$ $\mathcal{S}_{\underline{S^{-1}_{\underline{\varepsilon}}} applies}$, we can deterministically conclude that (since each inverse

scaling function is wide-sense increasing)

$$\begin{aligned} S^{-1}\left((S\left(F(t)\right))'\right) &\geq S^{-1}\left(\left(S(F)\otimes\beta_{\mathcal{S}}\right)(t)\right) \\ &= S^{-1}\left(\inf_{0\leq s\leq t}\left\{S(F(s))+\beta_{\mathcal{S}}(t-s)\right\}\right) \\ &= \inf_{0\leq s\leq t}\left\{S^{-1}\left(S(F(s))+\beta_{\mathcal{S}}(t-s)\right)\right\} \\ &= \inf_{0\leq s\leq t}\left\{S^{-1}\left(S(F(s))\right)\right. \\ &\quad \left.+S^{-1}\left(S(F(s))+\beta_{\mathcal{S}}(t-s)\right)-S^{-1}\left(S(F(s))\right)\right\} \\ &\geq \inf_{0\leq s\leq t}\left\{S^{-1}\left(S(F(s))\right)+\underline{S^{-1}}_{\overline{\varepsilon}}(\beta_{\mathcal{S}}(t-s))\right\} \\ &= \left(F\otimes\underline{S^{-1}}_{\overline{\varepsilon}}(\beta_{\mathcal{S}})\right)(t)\,. \end{aligned}$$

Since $S\circ S^{-1}=\mathrm{id}$, we can conclude from the above inequality that

$$\left(\mathcal{S}\left(F(t)\right)\right)' \geq \mathcal{S}\left(\left(F\otimes\underline{S^{-1}}_{\overline{\varepsilon}}(\beta_{\mathcal{S}})\right)(t)\right)\,.$$

Now consider system (a). Its output function can be bounded as follows:

$$S(F'(t))\geq S\left(F\otimes\beta\right)(t)\,.$$

If we let $\beta=\underline{S^{-1}}_{\overline{\varepsilon}}(\beta_{\mathcal{S}})$ in system (a), we get the same bound on the output function based on the assumption that $S(a,\omega)\in\mathcal{S}_{\underline{S^{-1}}_{\overline{\varepsilon}}applies}$, which holds with probability $1-\underline{\varepsilon}$. Hence, we obtain $\forall t\geq 0$:

$$Pr((\mathcal{S}\left(F(t)\right))'\geq(\mathcal{S}(F\otimes\beta))(t)) \quad\geq\quad 1-\underline{\varepsilon}\,.$$

Establishing the connection between the upper bound on the output functions of system (a) and (b) follows as an immediate variation. □

As in the deterministic case this alternative system theorem allows to move scaling elements over service curve elements in the stochastic setting. These enable to perform an efficient end-to-end analysis using the concatenation theorem as much as possible.

In the following corollary, we state the effect stochastic scaling elements have on the arrival constraints of an input function.

Corollary 5. *(Arrival Constraints under Stochastic Scaling) Let $A(t)$ be an arrival process with stochastic arrival curve $\alpha^{\varepsilon_\alpha}$ that is fed into a stochastic scaling element $\mathcal{S}$ with stochastic maximum scaling curve $\overline{S}^{\overline{\varepsilon}}$. A stochastic*

arrival curve for the scaled output from the scaling element is given by

$$\alpha_S = \overline{S}^{\overline{\varepsilon}}(\alpha^{\varepsilon_\alpha}) \,,$$

and it applies $\forall t \geq 0$ *that*

$$Pr\left(\mathcal{S}(A(t)) \leq (\mathcal{S}(A) \otimes \alpha_S)(t)\right) \geq 1 - \overline{\varepsilon} - \varepsilon_\alpha \,.$$

If $\mathcal{S}^{-1}$ *has a maximum scaling curve* $\overline{S^{-1}}^{\overline{\varepsilon}}$*, and given a stochastic arrival curve for the scaled output process* $\alpha_S^{\varepsilon_{\alpha_S}}$*, a stochastic arrival curve for the input can be given as*

$$\alpha = \overline{S^{-1}}^{\overline{\varepsilon}}\left(\alpha_S^{\varepsilon_{\alpha_S}}\right) \,,$$

and it applies $\forall t \geq 0$ *that*

$$Pr\left(A(t) \leq A \otimes \alpha(t)\right) \geq 1 - \overline{\varepsilon} - \varepsilon_{\alpha_S} \,.$$

Proof. First we choose a sample path function $F(t)$ of the arrival process $A(t)$ which does not violate the stochastic arrival curve $\alpha^{\varepsilon_\alpha}$ and fix t. Let $0 \leq s \leq t$.

From the stochastic maximum scaling curve property we have

$$\begin{aligned} & Pr\left(\sup_{0\leq s\leq t}\left\{\mathcal{S}(F(t)) - \mathcal{S}(F(s)) - \overline{S}^{\overline{\varepsilon}}(\alpha(t-s))\right\} \geq 0\right) \\ \leq \;& Pr\left(\sup_{0\leq s\leq t}\left\{\mathcal{S}(F(t)) - \mathcal{S}(F(s)) - \overline{S}^{\overline{\varepsilon}}(F(t)-F(s))\right\} \geq 0\right) \\ \leq \;& \overline{\varepsilon} \,. \end{aligned}$$

Here we used that $\alpha^{\varepsilon_\alpha}$ is an arrival curve of F and that $\overline{S}^{\overline{\varepsilon}}$ is wide-sense increasing. Using sample path argument and Boole's inequality establishes $\alpha_S = \overline{S}^{\overline{\varepsilon}}(\alpha^{\varepsilon_\alpha})$ as a stochastic arrival curve of the scaled process $\mathcal{S}(A(t))$ from the scaling element with probability $\geq 1 - \overline{\varepsilon} - \varepsilon_\alpha$.

Now, for the second part of the corollary, we first construct

$$F(t) - F(s) = \mathcal{S}^{-1}(\mathcal{S}(F(t))) - \mathcal{S}^{-1}(\mathcal{S}(F(s))) \,. \tag{3.1}$$

Then use Eq. (3.1) into the definition of the stochastic maximum scaling curve of $\mathcal{S}^{-1}$. The rest follows directly the argument in the first part. □

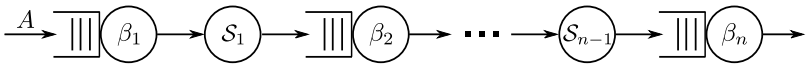


Figure 3.3: A n nodes tandem network traversed by a flow of interest.

3.2 End-to-end Performance Bounds

In this section, we now provide some insights in the scaling of the delay bounds when an increasing number of nodes is traversed by a flow and thereby also demonstrate how to apply our results in more general settings. We depict the n nodes scenario in Figure 3.3. We assume there is a stochastic scaling element between every two servers. For n nodes, we have $n-1$ scalers, $\mathcal{S}_i, i = 1, 2, \ldots, n-1$. For the case there is no flow transformation happening between two servers, we either consider there were a "transparent" scaling element with scaling function $S(a) = a$ for any $a \geq 0$, or regard the adjacent servers as one single server by convolving their service curves before we move them. The stochastic scaling elements are given according to Definition 14 and 15, i.e., for each $\mathcal{S}_i$, there are minimum and maximum scaling curves $\underline{S_i}^{\underline{\varepsilon_i}}$ and $\overline{S_i}^{\overline{\varepsilon_i}} \in \mathcal{F}$, such that for all $b \geq 0$

$$Pr\left(\inf_{0\leq a\leq b}\left\{\mathcal{S}_i(b) - \mathcal{S}_i(a) - \underline{S_i}_{\underline{\varepsilon_i}}(b-a)\right\} \leq 0\right) \leq \underline{\varepsilon_i}\,,$$

$$Pr\left(\sup_{0\leq a\leq b}\left\{\mathcal{S}_i(b) - \mathcal{S}_i(a) - \overline{S_i}^{\overline{\varepsilon_i}}(b-a)\right\} \geq 0\right) \leq \overline{\varepsilon_i}\,.$$

We point out, there are several alternatives for the end-to-end analysis. Next we compare them for the n nodes network. Since the meaning of the end-to-end backlog bound is quite unclear, i.e., we can merely derive the backlog bounds node by node, we ignore the backlog bounds and only compare the methods for analyzing delay bounds.

First, we provide an "idealistic" analysis for reference purposes. When demultiplexing a flow the number of generated subflow is usually dynamic, we however assume an homogeneous situation where this number is equally determined at every demultiplexer, denoted by k. For a hierarchical feed forward network this is a full k-nary tree. So for an idealistic analysis we assume that we exactly and deterministically know the scaling functions to be $S_i(F) = \frac{1}{k}F$, and thus their scaling curves $\underline{S_i}(a) = \overline{S_i}(a) = \frac{1}{k}a$ (for a not full k-nary tree, we can simply use different $k_i, k_i \leq k$). We can invoke (trivially) Theorem 7 (in both directions to move the scaler, to the front or the

back) and Corollary 4 to obtain a delay bound as (further applying Theorem 1 and 3)

$$
\begin{aligned}
&d^{ideal,ingr} \\
&= h\left(\overline{S_{n-1}}(\cdots\overline{S_1}(\alpha)\cdots), \underline{S_{n-1}}(\cdots\underline{S_2}(\underline{S_1}(\beta_1)\otimes\beta_2)\cdots\otimes\beta_{n-1})\otimes\beta_n\right) \\
&= h\left(\left(\frac{1}{k}\right)^{n-1}\alpha, \left(\frac{1}{k}\right)^{n-1}\beta_1\otimes\left(\frac{1}{k}\right)^{n-2}\beta_2\otimes\cdots\otimes\frac{1}{k}\beta_{n-1}\otimes\beta_n\right) \text{ or}
\end{aligned}
$$

$$
\begin{aligned}
d^{ideal,egr} &= h\left(\alpha, \beta_1\otimes\underline{S_1^{-1}}(\beta_2\otimes\cdots\underline{S_{n-2}^{-1}}(\beta_{n-1}\otimes\underline{S_{n-1}^{-1}}(\beta_n))\cdots)\right) \\
&= h\left(\alpha, \beta_1\otimes k\beta_2\otimes\cdots\otimes k^{n-1}\beta_n\right) .
\end{aligned}
$$

Clearly, none of the following analysis alternatives can go below these values as we now make more realistic assumptions on the knowledge we have about the scaling process. Both $d^{ideal,ingr}$ and $d^{ideal,egr}$ serve as the target value for the following alternatives. Consider an example that the arrival curve is a token-bucket $\gamma_{r,b}(t) = rt+b$ and the service curve is the constant rate service $\beta_i(t) = R_i t$. Using the results in Section 2.2.3 we get $d^{ideal,ingr} = \frac{\left(\frac{1}{k}\right)^{n-1}b}{R^{ingr}}$, where $R^{ingr} = \min\left\{\left(\frac{1}{k}\right)^{n-1}R_1, \left(\frac{1}{k}\right)^{n-2}R_2, \cdots, \left(\frac{1}{k}\right)^{0}R_n\right\}$, respectively, $d^{ideal,egr} = \frac{b}{R^{egr}}$, where $R^{egr} = \min\left\{R_1, kR_2, \cdots, k^{n-1}R_n\right\}$. Clearly the bounds are equal.

Next we show the deterministic delay bound, which can be regarded as another reference bound. We ignore the stochastic information known on the scaling process and put the deterministic bounds on the scaling as $\underline{S_i} = 0$ and $\overline{S_i} = id$. So we can directly ignore the scaling elements and get

$$d^{det} = h(\alpha, \beta_1\otimes\beta_2\otimes\cdots\otimes\beta_n) .$$

This only applies for the case that the flow is clipped. Otherwise when the scaled flow is enlarged, we have no trivial maximum scaling curve and still need to use the scaling process information to construct it. Certainly, the deterministic bound is conservative, because it essentially assumes that all of the traffic go to a single egress point of the network. Note that the deterministic scaling becomes excessively pessimistic if a large number of nodes are involved.

Now, we use all our "weapons" and provide an end-to-end analysis under stochastic scaling bounds. Here, using Th. 8 and shifting scaling elements

conveniently we are able to leverage again from the concatenation theorem. We have already seen that, we have two options, moving the scalers to the ingress or the egress of the overall system. But in further details, we have more options when moving the scalers. See Figure 3.3 again, if we move for example $\mathcal{S}_2$ to the ingress, we will face the concatenation of $\mathcal{S}_1$ and $\mathcal{S}_2$. We can treat it in two ways: the first we can put the effect of the scalers to the next neighbor element one by one ($\underline{S_2}(\underline{S_1}(\beta_1))$); the second we can regard this concatenation of scalers as one single scaler ($\mathcal{S}_1 \circ \mathcal{S}_2$ with stochastic scaling curves $\overline{S_1 \circ S_2}^{\overline{\varepsilon_{12}}}$ and $\underline{S_1 \circ S_2}_{\underline{\varepsilon_{12}}}$, where $\overline{\varepsilon_{12}}$ and $\underline{\varepsilon_{12}}$ are the violation probabilities to be given). So in the next steps we give the delay bounds for the cases of both moving directions and further for each case the results of two treatments of the concatenated scalers.

1. Move Stochastic Scaling to ingress
   a) nested scalings

   If move the scaler from $\mathcal{S}_1$ to $\mathcal{S}_{n-1}$, we have

$$d^{stoch,e2e} = h\Bigg(\overline{S_{n-1}}^{\overline{\varepsilon_{n-1}}}(\cdots \overline{S_1}^{\overline{\varepsilon_1}}(\alpha)\cdots), \underline{S_{n-1}}_{\underline{\varepsilon_{n-1}}}(\cdots \underline{S_2}_{\underline{\varepsilon_2}}(\underline{S_1}_{\underline{\varepsilon_1}}(\beta_1)\otimes\beta_2)\cdots\otimes\beta_{n-1})\otimes\beta_n\Bigg). \quad (3.2)$$

   If move the scaler from $\mathcal{S}_{n-1}$ to $\mathcal{S}_1$, we have

$$\begin{aligned} d^{stoch,e2e} = h\Bigg(&\overline{S_{n-1}}^{\overline{\varepsilon_{n-1}}}\left(\overline{S_{n-2}}^{\overline{\varepsilon_{n-2}}}\left(\cdots\left(\overline{S_1}^{\overline{\varepsilon_1}}(\alpha)\right)\cdots\right)\right), \\ &\underline{S_{n-1}}_{\underline{\varepsilon_{n-1}}}\left(\underline{S_{n-2}}_{\underline{\varepsilon_{n-2}}}\left(\cdots\left(\underline{S_1}_{\underline{\varepsilon_1}}(\beta_1)\right)\cdots\right)\right) \\ &\otimes\underline{S_{n-1}}_{\underline{\varepsilon_{n-1}}}\left(\underline{S_{n-2}}_{\underline{\varepsilon_{n-2}}}\left(\cdots\left(\underline{S_2}_{\underline{\varepsilon_2}}(\beta_2)\right)\cdots\right)\right) \\ &\otimes\cdots\otimes\underline{S_{n-1}}_{\underline{\varepsilon_{n-1}}}(\beta_{n-1})\otimes\beta_n\Bigg). \end{aligned}$$

   with probability

$$if\ independent \geq \prod_{i=1}^{n-1}(1-\overline{\varepsilon_i}-\underline{\varepsilon_i});$$

$$with\ Boole's\ inequality \quad \geq \quad 1 - \sum_{i=1}^{n-1} \left(\overline{\varepsilon_i} + \underline{\varepsilon_i} \right) .$$

Clearly, the former is better. The proof easily follows that for example $\underline{S}(\beta_1 \otimes \beta_2) \geq \underline{S}(\beta_1) \otimes \underline{S}(\beta_2)$ for a super-additive minimum scaling curve $\underline{S}$. The reason to assume a super-additive $\underline{S}$ is already given in Corollary 2.

b) concatenated scaling

If move the scaler from $\mathcal{S}_1$ to $\mathcal{S}_{n-1}$, the service curve part is not different from Eq. (3.2). But for the arrival curve part we can use the concatenation form $\overline{S_1 \circ S_2 \circ \cdots \circ S_{n-1}}^{\overline{\varepsilon}_{12\cdots n-1}}$ as the stochastic maximum scaling curve.

The probability satisfies

$$\begin{aligned} if\ independent \quad &\geq \quad (1 - \overline{\varepsilon_{12\cdots n-1}} - (1 - \prod_{i=1}^{n-1} (1 - \underline{\varepsilon_i})) ; \\ with\ Boole's\ inequality \quad &\geq \quad 1 - \overline{\varepsilon_{12\cdots n-1}} - \sum_{i=1}^{n-1} \underline{\varepsilon_i} . \end{aligned}$$

If move the scaler from $\mathcal{S}_{n-1}$ to $\mathcal{S}_1$, we have

$$\begin{aligned} d^{stoch,e2e} = h \Bigg(& \overline{S_1 \circ S_2 \circ \cdots \circ S_{n-1}}^{\overline{\varepsilon}_{12\cdots n-1}}(\alpha), \beta_n \otimes \underline{S_{n-1}}_{\underline{\varepsilon}_{n-1}}(\beta_{n-1}) \\ & \otimes \underline{S_{n-2} \circ S_{n-1}}_{\underline{\varepsilon}_{n-2,n-1}}(\beta_{n-2}) \otimes \cdots \\ & \otimes \underline{S_1 \circ S_2 \circ \cdots \circ S_{n-1}}_{\underline{\varepsilon}_{12\cdots n-1}}(\beta_1) \Bigg) . \end{aligned}$$

Because of the inherent dependency among these concatenated scalers, the probability satisfies only using Boole's inequality that

$$prob. \geq 1 - \overline{\varepsilon_{12\cdots n-1}} - \underline{\varepsilon_{n-1}} - \underline{\varepsilon_{n-2,n-1}} - \cdots - \underline{\varepsilon_{12\cdots n-1}} .$$

2. Move Stochastic Scaling to egress
   a) nested scalings

If move the scaler from $\mathcal{S}_{n-1}$ to $\mathcal{S}_1$,

$$d^{stoch,e2e} = h\left(\alpha, \beta_1 \otimes \underline{S_1^{-1}}_{\overline{\varepsilon_1}}\left(\beta_2 \otimes \cdots \underline{S_{n-1}^{-1}}_{\overline{\varepsilon_{n-1}}}(\beta_n)\cdots\right)\right) .$$

If move the scaler from $\mathcal{S}_1$ to $\mathcal{S}_{n-1}$,

$$\begin{aligned} d^{stoch,e2e} &= h\Bigg(\alpha, \beta_1 \otimes \underline{S_1^{-1}}_{\overline{\varepsilon_1}}(\beta_2) \otimes \underline{S_1^{-1}}_{\overline{\varepsilon_1}}\left(\underline{S_2^{-1}}_{\overline{\varepsilon_2}}(\beta_3)\right) \otimes \cdots \\ &\quad \otimes \underline{S_1^{-1}}_{\overline{\varepsilon_1}}\left(\underline{S_2^{-1}}_{\overline{\varepsilon_2}}\left(\cdots \underline{S_{n-1}^{-1}}_{\overline{\varepsilon_{n-1}}}(\beta_n)\cdots\right)\right)\Bigg) . \end{aligned}$$

The probability satisfies

$$\begin{aligned} if\ independent &\geq \prod_{i=1}^{n-1}(1-\overline{\varepsilon_i}) ; \\ with\ Boole's\ inequality &\geq 1 - \sum_{i=1}^{n-1}\overline{\varepsilon_i} . \end{aligned}$$

b) concatenated scaling

Only moving scaler from $\mathcal{S}_1$ to $\mathcal{S}_{n-1}$ applies.

$$\begin{aligned} d^{stoch,e2e} \leq\ & h(\alpha, \beta_1 \otimes \underline{S_1^{-1}}_{\overline{\varepsilon_1}}(\beta_2) \otimes \underline{(S_1 \circ S_2)^{-1}}_{\varepsilon_{12}}(\beta_3) \\ & \otimes \cdots \otimes \underline{(S_1 \circ S_2 \circ \cdots \circ S_{n-1})^{-1}}_{\varepsilon_{12\cdots n-1}}(\beta_n)) . \end{aligned}$$

with probability

$$\begin{aligned} if\ independent &\geq (1-\overline{\varepsilon}_1)(1-\overline{\varepsilon_{12}})\cdots(1-\overline{\varepsilon_{12\cdots n-1}}) ; \\ with\ Boole's\ inequality &\geq 1 - \sum \overline{\varepsilon_{12\cdots n-1}} . \end{aligned}$$

We point out that these bounds may not apply simultaneously for some cases and thus are not necessarily equal. The reason is that for different application scenarios, moving scalers to the ingress and the egress possibly have different meanings. For some scenarios moving the scalers to the ingress is physically meaningful whereas for the others to the egress is. We will see this explained in Chapter 4 and 5. However, since moving to a not so meaningful direction can actually be seen as a virtual equivalent system, we can still cal-

culate both and choose the one with better result. Another point to note is that the concatenated scaling is essentially not different from the nested scaling, but only a specialization of it. We provide this result because we possibly get a better bound when we treat the adjacent scalers as a single conjuncted one instead of applying the scaler one by one. We ignore the details here, which are discussed in [151].

As a contrast we show a straightforward node-by-node (nbn) delay bound, which fundamentally differs from the delay bounds obtained by using Theorem 8 and commuting the scalers and servers. We know the stochastic arrival curve of the flow scaled by $\mathcal{S}_i$ and arrived at β_{i+1} is $\alpha_{i+1} = \overline{S_i}^{\overline{\varepsilon_i}}(\alpha_i^{\varepsilon_{\alpha_i}} \oslash \beta_i)$. Recursively using the sample path bound and Boole's inequality we get

$$\begin{aligned} d^{stoch,nbn} &= h(\alpha,\beta_1) + h(\overline{S_1}^{\overline{\varepsilon_1}}(\alpha \oslash \beta_1),\beta_2) + \cdots + \\ & h(\overline{S_{n-1}}^{\overline{\varepsilon_{n-1}}}(\overline{S_{n-2}}^{\overline{\varepsilon_{n-2}}}(\cdots(\overline{S_1}^{\overline{\varepsilon_1}}(\alpha \oslash \beta_1) \oslash \beta_2)\cdots) \oslash \beta_{n-1}),\beta_n)\,. \end{aligned}$$

3.3 Modeling Dynamic Demultiplexing

Queueing networks are typically subject to demultiplexing operations, whereby network nodes split flows into multiple subflows. This concept captures many relevant network aspects such as packet loss, multi-path routing, load balancing, or flow control. In this section, we introduce a new demultiplexing element model with the benefit of the stochastic scaling element which can accommodate for dynamic demultiplexing decisions. The idea is to capture the frequent uncertainty about demultiplexing decisions within bounds of scaling functions for each of the outputs of the demultiplexer. We do not use the service curve or dynamic server to model it as the decision delay can usually be ignored comparing with the other delays. In order to illustrate the versatility of the new demultiplexing element, we later provide several sample application scenarios.

3.3.1 The Demultiplexer

The demultiplexer is illustrated in Figure 3.4. Note, we use the sample path expressions for the ease of exposition when exhibiting the modeling. The demultiplexer is an element with one input and multiple outputs. The actual distribution of the input flow over the output flows is determined according to a vector of scaling functions $\vec{S}$, i.e., we have for each output $i = 1, \ldots, n$

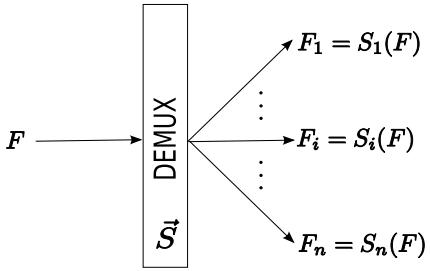


Figure 3.4: The demultiplexer.

that its output function $F_i = S_i(F)$ and furthermore

$$F = \sum_{i=1}^{n} F_i = \sum_{i=1}^{n} S_i(F) \; .$$

Hence, mathematically the demultiplexer provides the following mapping

$$\Sigma \; : \; \mathcal{F} \rightarrow \mathcal{F}^n \text{ with } F \rightarrow \Sigma(F) = \vec{S}(F) \; .$$

If the demultiplexing is not static, we will usually not know $\vec{S}$ explicitly, but have to work with bounds on it, the scaling curves. In many applications as discussed in the subsequent subsections, it is also very restrictive to assume that we can bound the demultiplexer deterministically. While there are always deterministic scaling curves, they may become the trivial alternatives of $\overline{S}_i = F$ and $\underline{S}_i = 0$, which certainly results in a very pessimistic performance analysis. Hence, what is required are stochastic bounds on a demultiplexer's behavior. The required fundamental stochastic generalization of scaling in network calculus was provided in Section 3.1.

3.3.2 Application 1: Load Balancing

A straightforward use of the demultiplexer is the case of load balancing over a number of servers as illustrated in Figure 3.5. Very often load balancing either makes randomized decisions, i.e., scheduling the next work unit on a server from the pool based on some random distribution, or demultiplexing is based on some state which we can only describe stochastically. The uncertainty about load balancing decisions may be due to data dependent switching decisions, information hiding by a network provider, or simply due to the im-

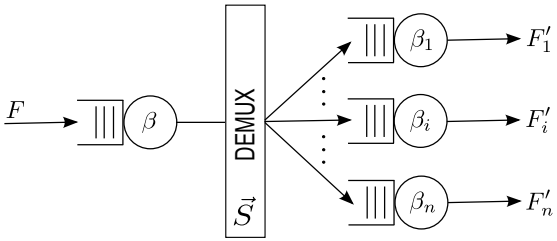


Figure 3.5: A load balancing model.

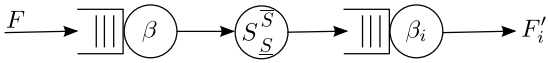


Figure 3.6: Load balancing from the perspective of subflow i.

practicality of obtaining all necessary switching information. In any case, the only sensible option to characterize the scaling of each of the outputs and thus the demultiplexing is via stochastic bounds. Examples for load-balancing systems that fall in this category are abundant (see, e.g., [87, 159, 141, 135] for recent systems from the networking and parallel computing domain).

In Figure 3.6, we also provide a view on the load-balancing model from the perspective of subflow i. This shows, provided that we can set up similar results to deterministic data scaling for the stochastic setting, that an end-to-end network calculus analysis is again possible for each subflow. Note that for each subflow the whole input flow needs to be taken into account as it drives the scaling element of the subflows ($\rightarrow S_i(F)$). As a concrete example, let us consider a simple weighted random load balancing. Assume $n = 3$, the distributions of the load are according to Bernoulli processes. Let the weights be $0.6, 0.3, 0.1$ for $\beta_1, \beta_2, \beta_3$. We can use stochastic scaling element to model each process and construct the stochastic scaling curves for these scaling elements, in order to capture the randomness. So between β and β_1 there is stochastic scaling element with Bernoulli scaling process $B(0.6)$. We will see how to construct the stochastic scaling curves and so thus analyze the performance for this application in Section 3.4.

3.3.3 Application 2: Lossy Links

A less obvious application of the demultiplexer is in networks with lossy links, a simple tandem scenario is shown in Figure 3.7. Here, the ground

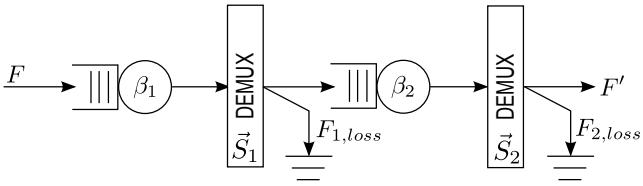


Figure 3.7: A tandem network with lossy links.

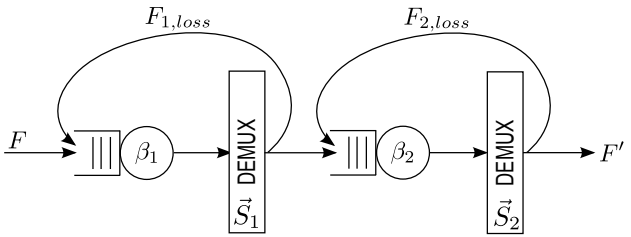


Figure 3.8: A tandem network with lossy links and retransmissions.

symbol means that data units are lost. Even more obviously here than in the load-balancing case, a deterministic bound on the scaling behaviour represents a mismatch and will only result in trivial scaling curves for the outputs. In particular, a deterministic bound would mean that all data units are lost, thus not providing any non-trivial insight in the system's performance. A stochastic generalization of scaling curves is inevitable for a system consisting of lossy links. The stochastic scaling models introduced in this chapter and Chapter 4 can deal with this. The key point is to find the suitable scaling processes to model the loss process, e.g., a Gilber-Elliot model as the scaling process. Given the parameters - good/bad state error rate and state transition probabilities, we can construct the stochastic scaling curves. This is also shown in Section 3.4.

Under the assumption that each link uses retransmissions to cater for data loss we can modify the model according to Figure 3.8. We will address this model in Chapter 6.

3.4 Application: Delay Bounds under Uncertain Load Balancing

The point of this section is to illustrate how we can apply the new demultiplexing element together with the results from the previous section in the case of a load-balancing system. In particular, we assume that we have an outsider view on the network and do not know how the switching decisions are made at each of the demultiplexing points. So, we try to find stochastic delay bounds under uncertainty about the load balancing decisions made inside the network.

For ease of exposition, the example calculations are kept as simple as possible without loss of generality, wherever possible. First, we compare in a simple scenario different analysis alternatives for determining delay bounds under uncertain load-balancing based on: deterministic scaling, stochastic scaling with a node-by-node analysis, and stochastic scaling with an end-to-end analysis. As we shall see, care needs to be taken on how to actually use the theoretical results derived in the previous sections. In fact, we demonstrate that achieving the best possible stochastic delay bound with a given violation probability requires us to solve a non-trivial optimization problem. For larger scenarios, solving the respective optimzation problem is out of scope for this chapter, yet we provide fallback options to achieve approximate results and illustrate the increasing benefit of using a stochastic instead of a deterministic analysis when the number of nodes grows.

3.4.1 Scenario and Preliminaries

The network scenario we assume is depicted in Figure 3.9. We have aggregate flows that enter a network at a certain ingress point and while traversing the load balancing network are demultiplexed inside the network, thus resulting in many potential egress flows. Consider for example the dynamic load balancing of cloud computing [114]. The flows can be hierarchically balanced to the lower level nodes (virtual machines or hosts). And the ingress node can be either centralized or distributed. Fix an ingress node, or in other words, from the perspective of a single aggregate flow, the network looks like a tree as shown also in Figure 3.9.

The demultiplexing or, more concretely, the load-balancing decisions are assumed not to be under the control of the flow, or more concretely, the corresponding user. These decisions are either done by the provider of the network or they are just random processes, for example depending on the contents of the data units. Formally, the demultiplexing decisions can be modeled as

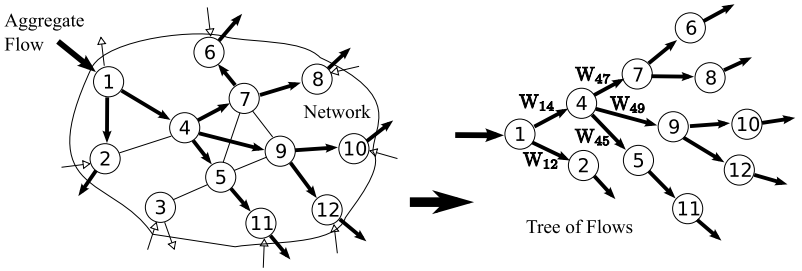


Figure 3.9: Network scenario of load-balancing.

the ratios W_{ij} of the incoming traffic at node i that is forwarded to node j, where the W_{ij} are random variables with support $[0, 1]$ and it applies that $\sum_j W_{ij} \leq 1$. The user, however, still wants to be able to compute delay bounds for all of its subflows (corresponding to different egress points from the network) even under uncertainty about the W_{ij}, i.e., the load-balancing decisions. Different options to model the uncertainty may be assumed. Here, the main question we should answer is: on which data granularity should the load-balancing decision be made? To get a tentative answer is not hard. We can not define this random ratio for the cumulative traffic we observe. Because on one hand, the random ratios for two overlapped amount of accumulated data will be dependent; on the other hand, if we used scaling elements to model the load-balancing decisions, we need them to be non-decreasing and obviously, this is not satisfied. Let us, for example, assume $W_{ij,k} \sim U(0, 1), k \geq 0$, where U means uniform distribution. To clarify, when we use W_{ij}, we need a further index k to differ different random variables effecting on different data aggregations. For example, we use $W_{ij,1}$ for data aggregation $a + b$, while $W_{ij,2}$ for a, $a, b > 0$. Then we possibly get $W_{ij,1} \cdot (a + b) < W_{ij,2} \cdot (a)$, which is not non-decreasing any more. Another possible solution is to fix a random ratio for all the time, which however shows its limitation by the fact that the scaling process can not be an ergodic one. Hence, it is reasonable to model this dynamic decision as random variables effecting onto each separated data amount.

For ease of exposition, but actually mostly without loss of generality, we further constrain our discussion on fully occupied binary trees for the flow under analysis as depicted in Figure 3.10. This has the nice side effect that each of the subflows faces the same situation with respect to the delay analysis. Hence, it does not matter which one we pick and we can actually focus on a scenario as depicted in Figure 3.11. We use the stochastic scaling el-

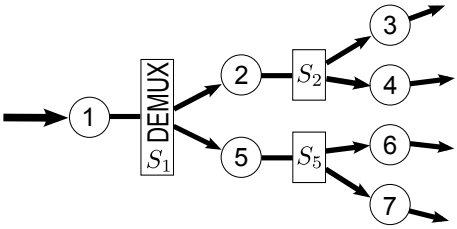


Figure 3.10: Full binary tree.

Figure 3.11: Subflow from full binary tree.

ement between servers to model the dynamic load-balancing. Before step further, we see that, for the fully occupied binary tree selecting all $W_{ij} = 0.5$ would be optimal with respect to minimizing the worst-case delay over all subflows. While this selection is not under our control (the W_{ij} are assumed to be random variables), it will still serve as an idealistic reference to assess our alternative analysis methods.

From now on we define the actual scaling elements from the concrete scenario in Figure 3.11 and construct the maximum and minimum scaling curves, such that we can derive the end-to-end delay bounds. Before that, we claim to use a simplification of denotation. Instead of $W_{ij,k}$ we simply use W_k for a general scaler (not necessarily between node i and j). And in order to match the index name with the data sequence, we replace the index k with i. As an immediate interpretation of above discuss, we define $S(a)$ as the sum of a sequence of decision random variables W_i's on the ordered sequence of the separations of a, formally, if we define the length of the i-th separation of a as $a_i, i = 1, 2, \ldots$ and $\sum_{i \geq 1} a_i = a$, then

$$S(a) = \sum_{i \geq 1} a_i W_i \ . \tag{3.3}$$

For the construction of the scaling curves, we use a simpler definition for the ease of exposition. We assume the data model to be discrete model and all the a_i's be normalized to 1. That means, we scale each bit, or each packet, or each equally-sized data unit. Hence, $S(a) = \sum_{i=1}^{a} W_i$. To construct the stochastic maximum and minimum scaling curves, we imitate the construction of the

stochastic sample path arrival curve provided in Section 2.3.1. The main difference here is the index set is changed from time domain to the space (data) domain. So we can not simply use the traffic model for the scaling element, which is a kind of control sequence. But there we utilized the MGF functions (bounds) for an arrival process, while here we do the same. Instead of unreasonably using diverse concrete process models for a control process, we classify W_i's into two sets, *i.i.d.* and non-*i.i.d.* , and concretely use the MMP to represent those non-*i.i.d.* scenarios without loss of generality. We show the derivation of the former case, while the latter one follows the similar steps in [36] and will also be discussed in the next chapter.

Lemma 1. *(MGF and Laplace Bounds of a Scaling Process) Consider scaling process $\mathcal{S}(a) = \sum_{i=1}^{a} W_i$ and W_i's are i.i.d. , its MGF $M_{\mathcal{S}(a)}(\theta)$ and Laplace function $M_{\mathcal{S}(a)}(-\theta)$ satisfies for $\theta > 0$ that*

$$\begin{aligned} M_{\mathcal{S}(a)}(\theta) &\leq e^{\theta \rho_S(\theta) a}\,, \\ M_{\mathcal{S}(a)}(-\theta) &\leq e^{-\theta \rho_S(-\theta) a}\,, \end{aligned}$$

where $\rho_S(\theta) = \frac{1}{\theta} \log M_{W_1}(\theta)$ and $\rho_S(-\theta) = -\frac{1}{\theta} \log M_{W_1}(-\theta)$.

For the MMP case, $\rho_S(\theta) = \frac{1}{\theta} \log sp(\phi(\theta)\mathbf{r})$, where sp is the spectral radius of the matrix $\phi(\theta)\mathbf{r}$, $\phi(\theta)$ is the diagonal matrix using the MGFs of processes to be modulated in each state, and $\mathbf{r}$ is the transition matrix of the Markov process. See details in [36]. Although easy, since the argument is quite often used in this thesis we show the proof.

Proof. $E\left[e^{\theta \sum_{i=1}^{a} W_i}\right] = \left(E\left[e^{\theta W_1}\right]\right)^a = e^{a \log M_{X_1}(\theta)} = e^{\theta \frac{1}{\theta} \log M_{X_1}(\theta) \cdot a}$. The other part follows directly. □

Now we construct the stochastic sample path scaling curves.

Lemma 2. *(Construction of the Stochastic Maximum and Minimum Scaling Curves) If a scaling element $\mathcal{S}(a)$ has the MGF and Laplace bounds, i.e., $M_{\mathcal{S}(a)}(\theta) \leq e^{\theta \rho_S(\theta) a}$ and $M_{\mathcal{S}(a)}(-\theta) \leq e^{-\theta \rho_S(-\theta) a}$, for $\overline{\delta}, \overline{\sigma}, \underline{\delta}, \underline{\sigma} > 0$, $\overline{S}^{\overline{\varepsilon}}(a)$ respectively $\underline{S}^{\underline{\varepsilon}}(a)$ is the stochastic maximum respectively minimum scaling curve, if*

$$\begin{aligned} \overline{S}^{\overline{\varepsilon}}(a) &= (\rho_S(\theta) + \overline{\delta})a + \overline{\sigma}\,, \\ \underline{S}^{\underline{\varepsilon}}(a) &= (\rho_S(-\theta) - \underline{\delta})a - \underline{\sigma}\,, \end{aligned}$$

with violation probabilities

$$\begin{aligned}\overline{\varepsilon} &= \frac{e^{-\theta\overline{\sigma}}}{\theta\overline{\delta}}\,,\\ \underline{\varepsilon} &= \frac{e^{-\theta\underline{\sigma}}}{\theta\underline{\delta}}\,.\end{aligned}$$

Proof. We first prove the maximum curve. For all $b \geq 0$ we have

$$\begin{aligned}& Pr\left(\sup_{0\leq a\leq b}\left\{\mathcal{S}(a,b) - \overline{S}^{\overline{\varepsilon}}(b-a)\right\} \geq 0\right)\\ \leq\ & \sum_{0\leq a<b} Pr\left(\mathcal{S}(a,b) - \overline{S}^{\overline{\varepsilon}}(b-a) \geq 0\right)\\ \leq\ & \sum_{0\leq a<b} e^{-\theta\overline{S}^{\overline{\varepsilon}}(b-a)} E\left[e^{\theta\mathcal{S}(a,b)}\right]\\ \leq\ & \sum_{0\leq a<b} e^{-\theta(\overline{\delta}(b-a)+\overline{\sigma})}\\ \leq\ & \frac{e^{-\theta\overline{\sigma}}}{\theta\overline{\delta}}\,.\end{aligned}$$

In the second line we used Union bound. In the third we used Chernoff bound. We get the last line if we let b go to infinity. Letting the last term equal to $\overline{\varepsilon}$ completes the proof of the maximum curve. The proof for the minimum curve follows directly when we consider

$$Pr\left(\mathcal{S}(a,b) - \underline{S}^{\underline{\varepsilon}}(b-a) \leq 0\right) = Pr\left(-\mathcal{S}(a,b) \geq -\underline{S}^{\underline{\varepsilon}}(b-a)\right)\,.$$

□

3.4.2 Comparison of Alternative Analyses

We now compare different alternatives we have for the end-to-end delay analysis of the uncertain load-balancing scenario. We analyze the network like shown in Figure 3.11 but with n nodes and $n-1$ scalers between every two nodes. For the illustrative purpose we limit n to 10, i.e., $n = 2, 3, \ldots, 10$. We assume the violation probability of the end-to-end delay bound as $\varepsilon = 0.1$. We assume the arrivals of this network have token-bucket arrival curve $\gamma_{r,b}$ and the service curve at each node is rate-latency $\beta_i = \beta_{R_i,T_i}, i = 2, 3, \ldots, 10$. In this comparison we assume the scaling has *i.i.d.* W_i's and

for the ease of exposition to be Bernoulli process with parameter p. We numerically further assume $r = 4(kp/s)$, $b = 40(kp)$, $[R_1, R_2, \ldots, R_{10}] = [9, 8.5, 8, 7.5, 7, 6.5, 6, 5.5, 5, 4.5](kp/s), T_i = 0.01(s), i = 1, 2, \ldots, 10$, and $p = 0.7$, where kp/s means kilo packets per second. Next we list the formular results and plot them with the numerical settings.

Stochastic Delay Bounds

In Section 3.2 we summarized the alternative analytical results of the stochastic delay bounds. Now we concretize them under given assumptions. We also use the results of Lemma 1 and 2 and denote $\overline{\rho} = \rho_S(\theta) + \overline{\delta}$ and $\underline{\rho} = \rho_S(-\theta) - \underline{\delta}$, where $\rho_S(\theta) = \frac{1}{\theta}\log(1 - p + pe^{\theta})$ for Bernoulli scaling with parameter p.

a) *Ideal:*

$$delay^{ideal} = (2^n - 1)T + \frac{b}{\min_{1\le i\le n}\{2^{i-1}R_i\}}$$

with the stability condition $r < \min_{1\le i\le n}\{2^{i-1}R_i\}$.

b) *Deterministic:*

$$delay^{det} = nT + \frac{b}{\min_{1\le i\le n}\{R_i\}}$$

with the stability condition $r < \min_{1\le i\le n}\{R_i\}$.

c) *Stochastic node-by-node:*

$$delay^{stoch.nbn} = nT + \frac{b}{R_1} + \sum_{i=2}^{n}\frac{(n-1)\overline{\rho}^{n-1}rT + \overline{\rho}^{n-1}b + \frac{1-\overline{\rho}^{n-1}}{1-\overline{\rho}}\overline{\sigma}}{R_n}$$

with the stability conditions $\overline{\rho}^{i-1}r < R_i, i = 1, 2, \ldots, n$ and violation probability $\sum_{1\le i\le n-1}\overline{\varepsilon}$.

d) *Stochastic end-to-end (ingress vs. egress):*

$$\begin{aligned} delay^{ingr} &= nT + \frac{\underline{\sigma}}{\underline{\rho}R_1} + \frac{\underline{\sigma}}{\min\{\underline{\rho}^2R_1, \underline{\rho}R_2\}} + \cdots \\ &+ \frac{\underline{\sigma}}{\min\{\underline{\rho}^{n-1}R_1, \underline{\rho}^{n-2}R_2, \cdots, \underline{\rho}R_{n-1}\}} + \frac{\overline{\rho}^{n-1}b + \frac{1-\overline{\rho}^{n-1}}{1-\overline{\rho}}\overline{\sigma}}{\min_{1\le i\le n}\{\underline{\rho}^{n-i}R_i\}} \end{aligned}$$

with the stability condition $\overline{\rho}^{n-1}r < \min\{\underline{\rho}^{n-1}R_1, \cdots, \underline{\rho}R_{n-1}, R_n\}$

and violation probability $\sum_{1\le i\le n-1}\overline{\epsilon}_i + \underline{\epsilon}_i$.

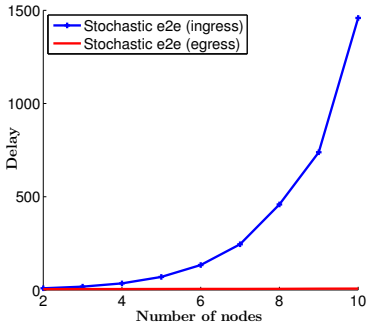


Figure 3.12: Comparison of moving the scalers to the ingress and egress.

$$delay^{egr} = nT + \frac{\overline{\sigma}}{R_n} + \frac{\overline{\sigma}}{\min\left\{R_{n-1}, \frac{R_n}{\overline{\rho}}\right\}} + \cdots + \frac{\overline{\sigma}}{\min\left\{R_2, \frac{R_3}{\overline{\rho}}, \frac{R_n}{\overline{\rho}^{n-2}}\right\}} + \frac{b}{\min_{1\le i\le n}\left\{\frac{R_i}{\overline{\rho}^{i-1}}\right\}}$$

with stability condition $r < \min\left\{R_1, \frac{R_2}{\overline{\rho}}, \cdots, \frac{R_n}{\overline{\rho}^{n-1}}\right\}$

and violation probability $\sum_{1\le i\le n-1} \overline{\epsilon}_i$.

We do not provide the delay bounds calculated using the concatenated scaling treatment, because the way we scale leads to more complexity of calculation and gains no great benefit (see [151]). When moving the scalers to the ingress as well as to the egress, we move the nearer one first, which overtakes the other order as discussed in Section 3.2. We compare the results in Figure 3.12, 3.13, and 3.14.

From Figure 3.12, we can see that moving the scalers to the ingress and to the egress results in dramatically different stochastic delay bounds. Only for a small number (3) of nodes both results are still comparable, after that, the delay bounds by moving scalers to the ingress are very pessimistic. The reason behind that is when we calculate the delay bounds, we simultaneously use the stochastic maximum and minimum scaling curves, which was concluded to be loose in [64]. As moving the scalers to the ingress is inferior to moving to the egress, we focus on the latter here.

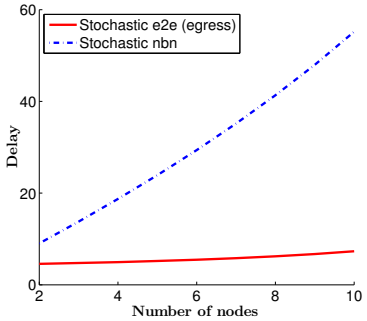


Figure 3.13: Comparison of node-by-node and end-to-end analyses.

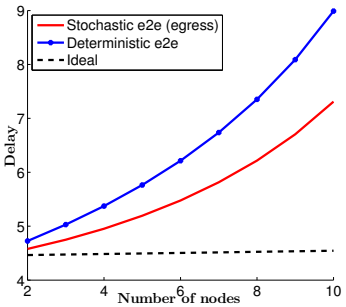


Figure 3.14: Comparison of ideal, deterministic, and stochastic scalings.

From Figure 3.13, we can see that a node-by-node analysis can only provide a worse bound. This is discussed in Section 2.2.3. But interestingly these node-by-node delay bounds are even better than the end-to-end delay bounds moving the scalers to the ingress.

In Figure 3.14 we compare the end-to-end delay bounds moving the scalers to the egress with other two alternatives: ideal and deterministic scalings. The results from Ideal and deterministic scaling models serve as reference values. We know the "ideal" is only a case to unbiased balance the traffic according the branches to the next nodes. We still need to adjust the parameter in ideal case when the capacities of the next nodes are biased. The deterministic result seems not bad and even better than the node-by-node analysis. But the problem of it lies in that without considering the scaling effect, stability condition

may not be satisfied at the bottleneck node, while in fact the traffic is already scaled before that node and the stability validation is passed. Therefore, the deterministic method can only serve as a reference for most cases instead of a practical calculation.

Remarks on the Choice of the Parameters

While in the previous subsections we showed how to work out different stochastic delay bounds, we suppressed, for the sake of ease of exposition, the discussion of composing a harmonic parameter choice. In fact, the calculation of the delay bounds values depends on the way we relate the violation probability ε of the delay bound to the violation probabilities $\overline{\varepsilon}_i, \underline{\varepsilon}_i$ of all the stochastic scaling curves, the way we relate $\overline{\delta}$ to $\overline{\sigma}$ respectively $\underline{\delta}$ to $\underline{\sigma}$, and even the felicitous choice of the free parameter θ.

We use the Boole's inequality to derive the violation probability and thus get the form like $\varepsilon = \sum_i \overline{\varepsilon}_i$. The first challenge is the assignment of the value among these "sub" violation probability values. We can systematically set up a series of weights for each and find an optimal combination. For the ease of exposition we let them equal. Although this has already resulted in the feasible values for our illustrative purpose, we should know that a careful choice of weights may lead to better results. The second challenge is the balance between σ and ρ, e.g., $\overline{\sigma}$ and $\overline{\rho}$. A smaller $\overline{\sigma}$ relates to a larger $\overline{\rho}$. Alert reader may find that $\overline{\rho}$ is a function of $\overline{\sigma}$. Under our assumption, $\overline{\rho} = \frac{e^{-\theta\overline{\sigma}}}{\theta\overline{\varepsilon}}$. Consider a simple case whereby a traffic with arrival curve $\alpha = \gamma_{r,b}$ traverses firstly a scaler then a server with service curve $\beta = \beta_{R,T}$. Fix the violation probability of the stochastic maximum scaling curve $\overline{\varepsilon}$, we get an optimal delay bound when $\overline{\sigma} = \frac{1}{\theta}\log\frac{b}{\overline{\varepsilon}}$. But it is difficult to apply this result directly to a more complicated formula like $delay^{egr}$ shown in the previous subsection. What we do here is simply numerically searching the better result instead of analytically. Careful choice of θ can also improve the final result potentially. An advanced way to get better results is to use different θ at each independent occasion when we apply the Chernoff bound. In this comparison that means we can use different θ for each stochastic maximum respectively minimum scaling curve. All these factors together compose a complex optimization problem. However, we must point out that, for the tightness of the stochastic delay bounds, the dominant issue is possibly not the parameter optimization, but the use of Boole's inequality. So here we leave the optimization open and refer the reader to a good discuss of replacing Boole's inequality with Martingale inequality [47, 118]. The complete adoption of Martingale is out of the scope of this thesis, but we still attempt to apply it, at least, for specific

scaling model (see Section 6.2.1).

Chapter 4

Stochastic Data Scaling Element - Process

In this chapter we aim to provide a parallel scaling element model for the flow transformation networks. While this new model could also be seen as a stochastic extension of the deterministic scaling element, in that we still strive for convolution-form expressions, we emphasize that we had to depart significantly from the deterministic one by defining the scaling at the level of the arrival processes, instead of purely aggregating scaling sample paths and characterizing the stochastic scaling curves as seen in Chapter 3. If we view the model using the scaling curves as a black-box modeling, the new one is a white-box modeling. It is feasible to assume some known knowledge on the scaling behavior, since we do not want our data control sequence completely out of control. Towards this goal, this chapter contributes by introducing a new defined stochastic scaling element, in the framework of the stochastic network calculus, to model flow transformations in great generality. The new scaling element is carefully defined to achieve (1) convolution-form network representations, and (2) a straightforward and flexible means of capturing actual transformation processes inside a network. The former allows to preserve the exact scaling properties in network calculus and the latter opens up the modeling scope widely. More technically speaking, unlike solely observing the scaling itself, the new stochastic scaling element also involves the flow to be transformed defines all in the process expression. This expression facilitates deriving several useful algebraic properties, of which the most important is that it can still be commuted with a dynamic server element. In this way, the end-to-end performance bounds can be computed by first reordering a se-

ries of dynamic server and scaling elements, and then applying concatenation properties for each type. We have seen this property in Chapter 3, whereas the new expression supports a finer treatment when modeling the real applications or deriving the performance bounds. The results presented in this chapter are from the joint work with F. Ciucu and J. Schmitt [49].

Closely following the structure of the two previous chapters we structure this chapter as follows. First, in Section 4.1 we introduce the new stochastic scaling elements with some of their properties. Then we provide the commutation lemma for transforming the system into an analytically equivalent system and use these elements in Section 4.3 to derive end-to-end delays in a network with flow transformations. These results are numerically illustrated in Section 4.4. Differently, in order to show the great modeling generality of the new stochastic scaling element, we will not provide any application scenario like the load balancing shown in Chapter 3 here. We leave the applications as show cases and to inspire the thoughts in the next chapters. Another point to note is, we use X to denote the scaling element in this chapter, accordingly S for the service. Unless specified, we keep this denotation pattern in the rest chapters: if X for the scaling *r.v.* , then S for the dynamic service, otherwise S is for the scaling function.

4.1 Stochastic Data Scaling Element

In this section we first newly define the stochastic data scaling element, state its basic properties, and give an example of a Markov-modulated scaling process. Then we show the commutativity property of the scaling and dynamic server elements, which is instrumental for expressing networks with flow transformations in convolution-form.

The time model is discrete. Again, an arrival process $A(t)$ is modeled with non-decreasing and non-negative random processes, taking integer values, and defined on some joint probability space. We model the transformation with the following stochastic scaling element definition.

Definition 18. (Stochastic Scaling Element - Scaled Arrivals) A (stochastic) scaling element consists of an arrival process $A(t)$, a scaling process $\mathbf{X} = (X_i)_{i\geq 1}$ taking non-negative integer values, and a scaled process $A^{\mathbf{X}}(t)$ defined for all $t \geq 0$ as

$$A^{\mathbf{X}}(t) = \sum_{i=1}^{A(t)} X_i \, . \tag{4.1}$$

For an illustration see Figure 4.1.

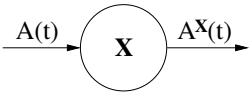


Figure 4.1: A scaling element with arrival process $A(t)$, scaling process $\mathbf{X} = (X_i)_{i\geq 1}$, and scaled process $A^{\mathbf{X}}(t)$.

Note, we use a different denotation for the scaling element here, i.e. $\mathbf{X}$ instead of $\mathcal{S}$. When $X_i \in \{0, 1\}$ for all $i \geq 1$ we say that the scaling process $\mathbf{X}$ is a loss process, which is useful for modeling losses at a link. In turn, if $X_i > 1$ for some i, then the scaling process $\mathbf{X}$ is useful for modeling retransmissions of previous losses or redundant transmissions. For notation convenience, when $\mathbf{X}$ is *i.i.d.*, we generically refer to X_i by X. Also, two scaling processes $\mathbf{X} = (X_i)_{i\geq 1}$ and $\mathbf{Y} = (Y_i)_{i\geq 1}$ are *i.i.d.* if $\{X_{i\geq 1}, Y_{i\geq 1}\}$ are (mutually) *i.i.d.*. And for two r.v.'s X and Y, we denote equality in distribution by $X =_d Y$.

For an arrival process $A(t)$ and a scaling process $\mathbf{X} = (X_i)_{i\geq 1}$, it is helpful to define the corresponding scaled process in *bivariate* form as:

$$A^{\mathbf{X}}(s, t) := A^{\mathbf{X}}(t) - A^{\mathbf{X}}(s) = \sum_{i=A(s)+1}^{A(t)} X_i \, . \tag{4.2}$$

Note that in general $A^{\mathbf{X}}(s, t) \neq (A(s, t))^{\mathbf{X}}$, but they are equal in distribution under appropriate stationarity assumption on $\mathbf{X}$ and independence between $A(t)$ and $\mathbf{X}$.

We remark that the scaled process $A^{\mathbf{X}}(t)$ from Eq. (4.1) is defined by space scaling at the granularity of data units, i.e., packets. A coarser scaling may be defined by scaling at the granularity of time units. Concretely, for a scaling process $\mathbf{Y} = (Y_s)_{s\geq 1}$, the probabilistic scaled process of $A(t)$ is the process $A^{\mathbf{Y}}(t)$ defined as

$$A^{\mathbf{Y}}(t) = \sum_{s=1}^{t} a(s) Y_s \, , \tag{4.3}$$

for all $t \geq 1$, where $a(s) = A(s) - A(s-1)$ is the instantaneous arrival process of $A(t)$. This is actually a variant of Eq. (3.3) at the level of arrival process.

The space and time scaling models are tightly related in the following

way.

Lemma 3. *(Scaling Granularities) See the following two scaling models*

$$(a)\ \ A^{\mathbf{X}}(t) = \sum_{i=1}^{A(t)} X_i \qquad (b)\ \ A^{\mathbf{Y}}(t) = \sum_{s=1}^{t} a(s) Y_s \ .$$

Assume $A(0) = 0$. *We have, (1) given (a), one can define for all* $s \geq 1$ *the process* $Y_s = \frac{\sum_{i=A(s-1)+1}^{A(s)} X_i}{a(s)}$ *such that* $A^{\mathbf{Y}}(t) = A^{\mathbf{X}}(t)$*; (2) conversely given (b), one can define for all* $i \geq 1$ *the process* $X_i = Y_{min\{s:i\leq A(s)\}}$ *such that* $A^{\mathbf{X}}(t) = A^{\mathbf{Y}}(t)$.

Proof. We prove (1) and (2) respectively. First (1),

$$\begin{aligned}
A^{\mathbf{Y}}(t) &= \sum_{s=1}^{t} a(s) Y_s \\
&= \sum_{s=1}^{t} a(s) \frac{\sum_{i=A(s-1)+1}^{A(s)} X_i}{a(s)} \\
&= \sum_{s=1}^{t} \sum_{i=A(s-1)+1}^{A(s)} X_i \\
&= \sum_{i=A(0)+1}^{A(1)} X_i + \sum_{i=A(1)+1}^{A(2)} X_i + \cdots + \sum_{i=A(t-1)+1}^{A(t)} X_i \\
&= \sum_{i=A(0)+1}^{A(t)} X_i \\
&= \sum_{i=1}^{A(t)} X_i \\
&= A^{\mathbf{X}}(t) \ .
\end{aligned}$$

Then (2),

$$A^{\mathbf{X}}(t) = \sum_{i=1}^{A(t)} X_i$$

$$
\begin{aligned}
&= \sum_{i=1}^{A(t)} Y_{min\{s:i\le A(s)\}} \\
&= \sum_{i=1}^{A(1)} Y_{min\{s:i\le A(s)\}} + \sum_{i=A(1)+1}^{A(2)} Y_{min\{s:i\le A(s)\}} + \cdots \\
&\qquad + \sum_{i=A(t-1)+1}^{A(t)} Y_{min\{s:i\le A(s)\}} \\
&= \sum_{i=1}^{A(1)} Y_1 + \sum_{i=A(1)+1}^{A(2)} Y_2 + \cdots + \sum_{i=A(t-1)+1}^{A(t)} Y_t \\
&= a(1)Y_1 + a(2)Y_2 + \cdots + a(t)Y_t \\
&= \sum_{s=1}^{t} a(s)Y_s \\
&= A^{\mathbf{Y}}(t) \, .
\end{aligned}
$$

For part (1), in the fourth line, we expand the outer sum. In the sixth line we use the assumption that $A(0) = 0$. For part (2), in the third line we seperate the sum into segments, in each segment we can decide $min\{s : i \le A(s)\}$. In the fifth line we use the definition of $a(t)$. □

The rest of this chapter considers the space scaling model only.

The next lemma states some basic properties of the scaling elements which are useful in analyzing networks with flow transformations.

Lemma 4. *(Properties of Scaling Elements) For an arrival process $A(t)$ and an independent and stationary scaling process $\mathbf{X} = (X_i)_{i\ge 1}$ the following basic properties hold.*

1. *(Scaling Additivity) If $\mathbf{Y} = (Y_i)_{i\ge 1}$ is a scaling process then*

$$A^{\mathbf{X}+\mathbf{Y}} = A^{\mathbf{X}} + A^{\mathbf{Y}} \, ,$$

   as illustrated in Figure 4.2.

2. *(Arrival Additivity) If $B(t)$ is an arrival process and X_i's are i.i.d. then*

$$(A + B)^{\mathbf{X}} =_d A^{\mathbf{X}} + B^{\mathbf{Y}} \, ,$$

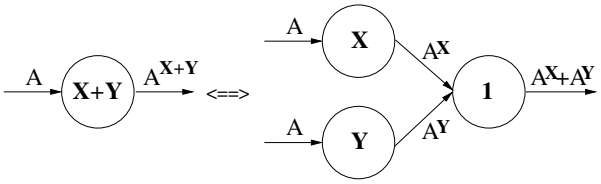


Figure 4.2: Scaling with a sum. The identity scaling element $\mathbf{1} = (1, 1, \dots)$ plays the role of a multiplexer.

as illustrated in Figure 4.3, where $\mathbf{Y} = (Y_i)_{i \geq 1}$ *is a scaling process equal in distribution with* $\mathbf{X}$ *and also independent of* $\mathbf{X}$.

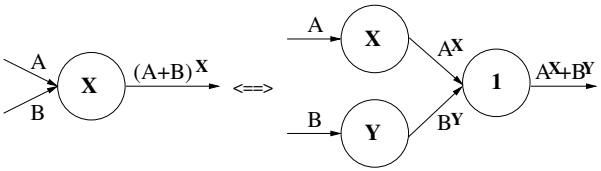


Figure 4.3: Scaling of a sum; $\mathbf{X}$ and $\mathbf{Y}$ are *i.i.d.* .

3. *(Stationarity) If both* $A(t)$ *and* $\mathbf{X}$ *are stationary and independent then the scaled process* $A^{\mathbf{X}}(t)$ *is stationary, i.e., for all* $s, t \geq 0$

$$A^{\mathbf{X}}(s, s+t) =_d A^{\mathbf{X}}(t) \, . \tag{4.4}$$

However, if $B(s)$ *is an additional arrival process, not necessarily independent of* $A(t)$*, and* $\mathbf{X}$ *is independent of* $(A(t), B(s))$*, then we have the following bound for* $s, t \geq 0$ *and* $x \geq 0$

$$Pr\left(A^{\mathbf{X}}(t) - B^{\mathbf{X}}(s) \geq x\right) \leq Pr\left((A(t) - B(s))^{\mathbf{X}} \geq x\right) \, . \tag{4.5}$$

4. *(Concatenation and Commutativity) If* $\mathbf{X}$ *is a loss process,* $\mathbf{Y} = (Y_i)_{i \geq 1}$ *is a scaling process independent of* $\mathbf{X}$*, and* X_i*'s,* Y_i*'s are i.i.d. then*

$$\left(A^{\mathbf{X}}\right)^{\mathbf{Y}} =_d A^{\mathbf{XY}} \, ,$$

where $\mathbf{XY}$ *is the scalar product, as illustrated in Figure 4.4.*

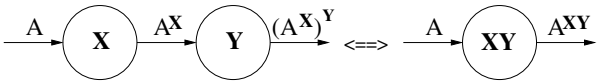


Figure 4.4: Concatenation; $\mathbf{X}$ is a loss process and $\mathbf{X}$ and $\mathbf{Y}$ are *i.i.d.* .

If $\mathbf{Y}$ *is also a loss process then we have the commutativity property*

$$\left(A^{\mathbf{X}}\right)^{\mathbf{Y}} =_d \left(A^{\mathbf{Y}}\right)^{\mathbf{X}} .$$

5. *(Absorbing Zero and Identity Elements) If* $\mathbf{0} = (0, 0, \dots)$ *and* $\mathbf{1} = (1, 1, \dots)$ *then*
$$A^{\mathbf{0}} = 0, \;\; A^{\mathbf{1}} = A.$$

We point out that the properties of concatenation and commutativity require the strong condition that the processes are *i.i.d.*; as it will be evident from the proof, this condition cannot be relaxed to the stationarity of the processes.

Proof. The two additivity properties and the absorbing zero and identity elements' properties follow directly from the definition of the scaled process.

(1)

$$A^{\mathbf{X}+\mathbf{Y}} = \sum_{i=1}^{A(t)} (\mathbf{X}+\mathbf{Y})_i = \sum_{i=1}^{A(t)} (X_i + Y_i) = \sum_{i=1}^{A(t)} X_i + \sum_{i=1}^{A(t)} Y_i = A^{\mathbf{X}} + A^{\mathbf{Y}}.$$

The key point here is to define the "sum" random process.

(2)

We check the probabilities of both sides.

$$\begin{aligned}
&Pr\left(\sum_{i=1}^{A(t)+B(t)} X_i \le x\right) \\
=&\sum_B \sum_A Pr\left(\sum_{i=1}^{A+B} X_i \le x\right) Pr(B(t) = B \mid A(t) = A) Pr(A(t) = A) \\
=&\sum_B \sum_A Pr\left(\sum_{i=1}^{A} X_i + \sum_{i=A+1}^{A+B} X_i \le x\right) Pr(A(t) = A, B(t) = B)
\end{aligned}$$

$$
\begin{aligned}
=&\sum_B \sum_A \left(\sum_{x_A} Pr\left(x_A + \sum_{i=A+1}^{A+B} X_i \le x \mid \sum_{i=1}^{A} X_i = x_A \right) \cdot \right. \\
&\left. Pr\left(\sum_{i=1}^{A} X_i = x_A \right) \right) \cdot Pr(A(t) = A, B(t) = B) \\
=&\sum_B \sum_A \left(\sum_{x_A} Pr\left(x_A + \sum_{i=A+1}^{A+B} X_i \le x \right) Pr\left(\sum_{i=1}^{A} X_i = x_A \right) \right) \\
&\cdot Pr(A(t) = A, B(t) = B) \\
=&\sum_B \sum_A \left(\sum_{x_A} Pr\left(x_A + \sum_{i=1}^{B} Y_i \le x \right) Pr\left(\sum_{i=1}^{A} X_i = x_A \right) \right) \\
&\cdot Pr(A(t) = A, B(t) = B) \\
=&\sum_B \sum_A Pr\left(\sum_{i=1}^{A} X_i + \sum_{i=1}^{B} Y_i \le x \right) Pr(A(t) = A, B(t) = B) \\
=&Pr\left(A^{\mathbf{X}}(t) + B^{\mathbf{Y}}(t) \le x \right) .
\end{aligned}
$$

Hence, $(A+B)^{\mathbf{X}} =_d A^{\mathbf{X}} + B^{\mathbf{Y}}$. Note, because $\sum_{i=1}^{A} X_i$ may be dependent of $\sum_{i=1}^{B} X_i$, if we replace $\sum_{i=A+1}^{A+B} X_i$ with $\sum_{i=1}^{B} X_i$, we cannot add/remove the condition part of the probability. That's why we introduce Y_i's.

(3) (a) The proof is similar to the proof of arrival additivity.

$$
\begin{aligned}
&Pr\left(A^{\mathbf{X}}(s, s+t) \le x \right) \\
&= Pr\left(\sum_{i=A(s)+1}^{A(s+t)} X_i \le x \right) \\
&= \sum_{A_s} \sum_{A_{st}} Pr\left(\sum_{i=A_s+1}^{A_{st}} X_i \le x \right) Pr(A(s) = A_s, A(s+t) = A_{st}) \\
&= \sum_{A_s} \sum_{A_{st}} Pr\left(\sum_{i=1}^{A_{st}-A_s} X_i \le x \right) Pr(A(s) = A_s, A(s+t) = A_{st}) \\
&= \sum_{A_s} \sum_{A_{st}} Pr\left(\sum_{i=1}^{A_{st}-A_s} X_i \le x \right) Pr(A(0) = A_s, A(t) = A_{st})
\end{aligned}
$$

$$
\begin{aligned}
&= Pr\left(\sum_{i=1}^{A(t)-A(0)} X_i \le x\right) \\
&= Pr\left(\sum_{i=1}^{A(t)} X_i \le x\right) .
\end{aligned}
$$

In the third line we use the independence of $\mathbf{X}$ and $A(t)$. In the fourth and fifth lines we respectively use the stationarity of $\mathbf{X}$ and $A(t)$. In the last line we use the assumption that $A(0) = 0$.

(b)

$$
\begin{aligned}
&Pr\left(A^{\mathbf{X}}(t) - B^{\mathbf{X}}(s) \ge x\right) \\
&= Pr\left(\sum_{i=1}^{A(t)} X_i - \sum_{i=1}^{B(s)} X_i \ge x\right) \\
&= Pr\left(\sum_{i=B(s)+1}^{A(t)} X_i \ge x, A(t) > B(s)\right) + \\
&\qquad\qquad Pr\left(\sum_{i=B(s)+1}^{A(t)} X_i \ge x, A(t) \le B(s)\right) \\
&= \sum_B \sum_A Pr\left(\sum_{i=B+1}^{A} X_i \ge x, A > B\right) Pr(B(s) = B, A(t) = A) \\
&= \sum_B \sum_A Pr\left(\sum_{i=1}^{A-B} X_i \ge x, A > B\right) Pr(B(s) = B, A(t) = A) \\
&= Pr\left(\sum_{i=1}^{A(t)-B(s)} X_i \ge x, A(t) > B(s)\right) \\
&\le Pr\left(\sum_{i=1}^{A(t)-B(s)} X_i \ge x\right) .
\end{aligned}
$$

In the third line we use the total probability. In the fourth and fifth line we respectively use conditioning on $B(s)$ and $A(t)$. In the fourth line we use the independence of $\mathbf{X}$ and arrivals. In the fifth line, the stationarity of $\mathbf{X}$ is used. The equality in Eq. (4.5) appears in the case of $x = 0$. Using the same

argument, one may prove a generalized statement that for $x \geq 0$

$$Pr\left((A^{\mathbf{X}}(t) - B^{\mathbf{X}}(s) - C)^{\mathbf{Y}} \geq x\right) \leq Pr\left(\left((A(t) - B(s))^{\mathbf{X}} - C\right)^{\mathbf{Y}} \geq x\right), \tag{4.6}$$

where C and $\mathbf{Y}$ are additional *r.v.* / scaling process, stationary and independent of the rest. Critical for the proof is the positivity of $\mathbf{X}$ and $\mathbf{Y}$.

(4) (a)

$$\begin{aligned}
& Pr\left(\left(A^{\mathbf{X}}\right)^{\mathbf{Y}}(t) \leq z\right) \\
= \;& Pr\left(\sum_{i=1}^{\sum_{j=1}^{A(t)} X_j} Y_i \leq z\right) \\
= \;& \sum_A Pr\left(\sum_{i=1}^{\sum_{j=1}^{A} X_j} Y_i \leq z \mid A(t) = A\right) Pr(A(t) = A) \\
= \;& \sum_A Pr\left(\sum_{i=1}^{X_1+X_2+\cdots+X_A} Y_i \leq z\right) Pr(A(t) = A) \\
= \;& \sum_A \sum_{x_1} Pr\left(\sum_{i=1}^{x_1+X_2+\cdots+X_A} Y_i \leq z \mid X_1 = x_1\right) \\
& \qquad \cdot Pr(A(t) = A \mid X_1 = x_1) Pr(X_1 = x_1) \\
= \;& \sum_A \sum_{x_1} Pr\left(\sum_{i=1}^{x_1+X_2+\cdots+X_A} Y_i \leq z\right) Pr(X_1 = x_1) Pr(A(t) = A) \\
= \;& \sum_A \sum_{x_1} \sum_{x_2} \cdots \sum_{x_A} P\left(\sum_{i=1}^{x_1+x_2+\cdots+x_A} Y_i \leq z\right) \\
& \qquad Pr(X_1 = x_1) \cdots Pr(X_A = x_A) Pr(A(t) = A) \, .
\end{aligned}$$

In the third line we condition the probability on $A(t)$. In the fourth line we use the independence of scaling and arrival process. The fifth line uses conditioning on X_1. In the sixth line we use the independence of X_i's and Y_i's. Now it is sufficient to prove $\sum_{i=1}^{x_1+x_2+\cdots+x_A} Y_i =_d \sum_{i=1}^{A} x_i Y_i$. Because $\mathbf{X}$ is a loss process, $x_i \in 0, 1$. Let $x_{j_1} = x_{j_2} = \cdots = x_{j_n} = 1$, other x_i's be 0, where $1 \leq j_1 < j_2 \cdots < j_n \leq A$ and $1 \leq n \leq A$. We have

$$Y_1 + Y_2 + \cdots + Y_{x_1+x_2+\cdots+x_A}$$

$$
\begin{aligned}
&= Y_1 + Y_2 + \cdots + Y_{x_{j_1}+x_{j_2}+\cdots+x_{j_n}} \\
&= Y_{j_1} + Y_{j_2} + \cdots + Y_{j_n} \\
&= x_{j_1} Y_{j_1} + x_{j_2} Y_{j_2} + \cdots + x_{j_n} Y_{j_n} \\
&= x_1 Y_1 + x_2 Y_2 + \cdots + x_A Y_A \ .
\end{aligned}
$$

We get the third line because Y_i's are *i.i.d.* . We get the fifth line because other x_i's are 0. Continuing with the former proof, we have

$$
\begin{aligned}
& Pr\left(\left(A^{\mathbf{X}}\right)^{\mathbf{Y}}(t) \le z\right) \\
&= \sum_{A} \sum_{x_1} \sum_{x_2} \cdots \sum_{x_A} P\left(\sum_{i=1}^{A} x_i Y_i \le z\right) Pr(X_1 = x_1) \cdots \\
&\qquad\qquad Pr(X_A = x_A) \cdot Pr(A(t) = A) \\
&= Pr\left(\sum_{i=1}^{A(t)} X_i Y_i \le z\right) \\
&= Pr\left(A^{\mathbf{XY}}(t) \le z\right) \ .
\end{aligned}
$$

Finally, we have $\left(A^{\mathbf{X}}\right)^{\mathbf{Y}} =_d A^{\mathbf{XY}}$.

(b)

If $\mathbf{Y}$ is also a loss process, we have $\left(A^{\mathbf{Y}}\right)^{\mathbf{X}} =_d A^{\mathbf{YX}}$. Because $A^{\mathbf{XY}} = \sum_{i=1}^{A(t)} X_i Y_i = \sum_{i=1}^{A(t)} Y_i X_i = A^{\mathbf{YX}}$, we get $\left(A^{\mathbf{X}}\right)^{\mathbf{Y}} =_d \left(A^{\mathbf{Y}}\right)^{\mathbf{X}}$.

(5)

$$
\begin{aligned}
A^{\mathbf{0}}(t) &= \sum_{i=1}^{A(t)} \mathbf{0}_i = \sum_{i=1}^{A(t)} 0 = 0 \\
A^{\mathbf{1}}(t) &= \sum_{i=1}^{A(t)} \mathbf{1}_i = \sum_{i=1}^{A(t)} 1 = A(t) \ .
\end{aligned}
$$

Note, these are just two cases of "constant process". Using constant K to define $\mathbf{K} = (K, K, \cdots)$, $A^{\mathbf{K}}(t) = K \cdot A(t)$ will be a generalization. □

4.1.1 Example: Markov-Modulated Scaling Processes (MMSP)

We define a scaling process $\mathbf{X} = (X_i)_{i \ge 1}$ as being modulated by a discrete and homogeneous Markov process $S(i)$ with states $1, 2, \ldots, M$ and transition probabilities $\lambda_{i,j}$ for all $1 \le i, j \le M$, as in Figure 4.5. Let also the

i.i.d. random processes $L_i(n)_{n\geq 1}$ for all $1 \leq i \leq M$. The scaling process is

$$X_i = L_{S(i)}(i) \ ,$$

i.e., the scaling follows the distribution of $L_{S(i)}(1)$ while in state $S(i)$. In the special case when the Markov chain has a single state, i.e., $M = 1$, or even two states with $\lambda_{1,2} + \lambda_{2,1} = 1$, the entire scaling process $\mathbf{X} = (X_i)_{i\geq 1}$ is *i.i.d.*.

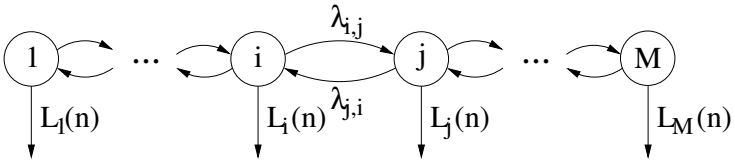


Figure 4.5: A Markov chain $S(i)$ with M states and transition probabilities $\lambda_{i,j}$, modulating the scaling process $\mathbf{X}$ as $X_i = L_{S(i)}(i)$.

A loss process can be fitted from the classical two-state Gilbert-Elliott loss model [67, 58] or the finite-state Markov channel for modeling Rayleigh fading [152]. Further, by inversing these models and adding a time offset, one may consequently determine a retransmission scaling process.

In order to carry out a calculus with the dynamic scaling expression it is useful to compute the MGF bounds of the scaled processes in terms of the MGFs of the arrival processes.

Lemma 5. *(Moment Generating Function of a Scaled Process) Let an arrival process $A(t)$ and an MMSP $\mathbf{X} = (X_i)_{i\geq 1}$ be defined as above. Then we have the MGFs for some $\theta > 0$:*

1. *(General case) If the matrix $\lambda = (\lambda_{i,j})_{i,j}$ is irreducible and aperiodic then*

$$M_{A^{\mathbf{X}}(t)}(\theta) \leq M_{A(t)}\left(\log sp\left(\phi(\theta)\lambda\right)\right) \ , \tag{4.7}$$

   where $\phi(\theta) := diag\left(M_{L_1(1)}(\theta), \ldots, M_{L_M(1)}(\theta)\right)$ and $sp\left(\phi(\theta)\lambda\right)$ denotes the spectral radius of $\phi(\theta)\lambda$.

2. *(I.i.d. case) If X_i's are i.i.d. then*

$$M_{A^{\mathbf{X}}(t)}(\theta) = M_{A(t)}\left(\log M_X(\theta)\right) \ . \tag{4.8}$$

The proof follows using conditioning.

Proof. 1. (General case)

$$\begin{aligned}
M_{A^{\mathbf{X}}(t)}(\theta) = E\left[e^{\theta A^{\mathbf{X}}(t)}\right] &= E\left[e^{\theta \sum_{k=1}^{A(t)} X_k}\right] \\
&= E\left[E\left[e^{\theta \sum_{k=1}^{A(t)} X_k} \mid A(t)\right]\right] \\
&= \sum_A E\left[E\left[e^{\theta \sum_{k=1}^{A} X_k} \mid A(t) = A\right]\right] Pr(A(t) = A) \\
&= \sum_A E\left[E\left[e^{\theta \sum_{k=1}^{A} X_k}\right]\right] Pr(A(t) = A) \\
&= \sum_A E\left[E\left[e^{\theta \sum_{k=1}^{A} L_{S(k)}(k)}\right]\right] Pr(A(t) = A)\ .
\end{aligned}$$

In the second and the third line we use conditional expectation. We get the fourth line because $A(t)$ and $\mathbf{X}$ are independent. Now we condition on the starting state of Markov process $S(1)$

$$\begin{aligned}
& E\left[e^{\theta \sum_{k=1}^{A} L_{S(k)}(k)} \mid S(1) = i\right] \\
= \ & E\left[e^{\theta L_{S(1)}(1)} \cdot e^{\theta \sum_{k=2}^{A} L_{S(k)}(k)} \mid S(1) = i\right] \\
= \ & E\left[e^{\theta L_{S(1)}(1)} \mid S(1) = i\right] E\left[e^{\theta \sum_{k=2}^{A} L_{S(k)}(k)} \mid S(1) = i\right] \\
= \ & \phi_i(\theta) \sum_{j=1}^{M} E\left[e^{\theta \sum_{k=2}^{A} L_{S(k)}(k)} \mid S(2) = j, S(1) = i\right] \cdot \\
& \qquad\qquad\qquad\qquad\qquad\qquad Pr(S(2) = j \mid S(1) = i) \\
= \ & \phi_i(\theta) \sum_{j=1}^{M} E\left[e^{\theta \sum_{k=2}^{A} L_{S(k)}(k)} \mid S(2) = j\right] \lambda_{i,j}\ .
\end{aligned}$$

Note, in the third line, because $\left\{L_{S(k)}(k), k > 1\right\}$ doesn't contain $L_{S(1)}(1)$, they are independent. Now we first prove $E[f(X_2)g(X_3) \mid X_2 = j] = E[f(X_1)g(X_2) \mid X_1 = j]$, where $X_1, X_2, X_3, \cdots$ are stationary. Let $h(X) = f(j)g(X)$, we get

$$\begin{aligned}
& E[f(X_2)g(X_3) \mid X_2 = j] \\
= \ & E[h(X_3) \mid X_2 = j] \\
= \ & \sum_{x_3} h(x_3) Pr\{X_3 = x_3 \mid X_2 = j\}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{x_3} h(x_3) Pr\{X_2 = x_3 \mid X_1 = j\} \\
&= E[h(X_2) \mid X_1 = j] \\
&= E[f(j)g(X_2) \mid X_1 = j] \\
&= E[f(X_1)g(X_2) \mid X_1 = j] \,.
\end{aligned}
$$

In the third line we use the stationarity of X sequence. For the general case in our following equation, the proof will be an immediate extention. Note, because $S(i)$'s are stationary, $(L_{S(k)}(k))$'s are stationary, i.e., X_i's are stationary. That means $\sum_{i=A}^{A+B} L_{S(k)}(k)$ has the same distribution with $\sum_{i=1}^{B} L_{S(k)}(k)$, where $A, B > 0$. So we have following equations

$$
\begin{aligned}
& \phi_i(\theta) \sum_{j=1}^{M} E\left[e^{\theta \sum_{k=2}^{A} L_{S(k)}(k)} \mid S(2) = j\right] \lambda_{i,j} \\
= \; & \phi_i(\theta) \sum_{j=1}^{M} E\left[e^{\theta \sum_{k=1}^{A-1} L_{S(k)}(k)} \mid S(1) = j\right] \lambda_{i,j} \,.
\end{aligned}
$$

Define

$$
\psi(\theta, A) = \left(E\left[e^{\theta \sum_{k=1}^{A} X_k} \mid S(1) = 1\right], \cdots, E\left[e^{\theta \sum_{k=1}^{A} X_k} \mid S(1) = M\right]\right).
$$

Then $\psi(\theta, A) = \phi(\theta)\lambda\psi(\theta, A-1)^T$ with initial condition $\psi(\theta, 1)^T = \phi(\theta)\mathbf{1}^T$, because for $A = 0$, $E\left[e^{\theta \sum_{k=1}^{0} X_k} \mid S(1) = j\right] = E[1 \mid S(1) = j] = 1$ and $\lambda = I$.

Next step, we have

$$
\begin{aligned}
\psi(\theta, A)^T &= \phi(\theta)\lambda\psi(\theta, A-1)^T \\
&= \phi(\theta)\lambda\left(\phi(\theta)\lambda\psi(\theta, A-2)^T\right) \\
&= (\phi(\theta)\lambda)^2\psi(\theta, A-2)^T \\
&\cdots \\
&= (\phi(\theta)\lambda)^{A-1}\phi(\theta)\mathbf{1}^T \,.
\end{aligned}
$$

Let $\pi = (\pi_1, \pi_2, \cdots, \pi_M)$ be the vector of probabilities of $S(1)$ being at state $i = \{1, 2, \cdots, M\}$. Then we use total probability and get

$$
E\left[e^{\theta \sum_{k=1}^{A} X_k}\right] = \pi\psi(\theta, A)^T = \pi(\phi(\theta)\lambda)^{A-1}\phi(\theta)\mathbf{1}^T.
$$

So far we get

$$\begin{aligned} M_{A^{\mathbf{x}}(t)}(\theta) &= \sum_{A} E\left[e^{\theta\sum_{k=1}^{A} X_k}\right] Pr(A(t)=A) \\ &= \sum_{A} \left(\pi(\phi(\theta)\lambda)^{A-1}\phi(\theta)\mathbf{1}^T\right) Pr(A(t)=A)\ . \end{aligned}$$

And because λ is primitive (λ is irreducible and aperiodic, see e.g., Corollary 5.6.13 of [75]), we have

$$\lim_{A\to\infty}\left[\phi(\theta)\lambda/\text{sp}(\phi(\theta)\lambda)\right]^A \le C, \text{where } C \text{ is a constant matrix.}$$

Then we have the following limitation

$$\begin{aligned} & \lim_{A\to\infty}\frac{1}{\theta A}\log E\left[e^{\theta\sum_{k=1}^{A} X_k}\right] \\ = & \lim_{A\to\infty}\frac{1}{\theta A}\log\left(\pi(\phi(\theta)\lambda)^{A-1}\phi(\theta)\mathbf{1}^T\right) \\ = & \lim_{A\to\infty}\frac{1}{\theta A}\log\left(\pi\left[\text{sp}(\phi(\theta)\lambda)\right]^{A-1}\left[\frac{\phi(\theta)\lambda}{\text{sp}(\phi(\theta)\lambda)}\right]^{A-1}\phi(\theta)\mathbf{1}^T\right) \\ \le & \lim_{A\to\infty}\frac{1}{\theta A}\log\left[\text{sp}(\phi(\theta)\lambda)\right]^{A-1} + \lim_{A\to\infty}\frac{1}{\theta A}\log\left[\pi C(\theta)\phi(\theta)\mathbf{1}^T\right] \\ = & \frac{1}{\theta}\log\text{sp}(\phi(\theta)\lambda) + 0 \\ = & \frac{1}{\theta}\log\text{sp}(\phi(\theta)\lambda)\ . \end{aligned}$$

Because we assume $A(t)$ to be stationary, and we know X_k's are stationary, from lemma 1 (stationarity of scaled process), we know that $\sum_{k=1}^{A(t)} X_k$ are stationary. Thus the θ-MER of $\sum_{k=1}^{A(t)} X_k$ can be presented as below

$$\limsup_{A\to\infty}\frac{1}{\theta A}\log E\left[e^{\theta\sum_{k=1}^{A} X_k}\right] = \frac{1}{\theta}\log\text{sp}(\phi(\theta)\lambda)\ .$$

Note, here A is like the index "t" in the original definition of θ-MER (p.241 of [36]). Hence, we get

$$E\left[e^{\theta\sum_{k=1}^{A(t)} X_k}\right] \le E\left[e^{\theta\left(\left(\frac{1}{\theta}\right)\log\text{sp}(\phi(\theta)\lambda)\right)A(t)}\right]\ .$$

That is

$$M_{A^{\mathbf{X}}(t)}(\theta) \leq M_{A(t)}(\log \text{sp}(\phi(\theta)\lambda)) \ .$$

2. (*I.i.d.* case)

$$\begin{aligned}
M_{A^{\mathbf{X}}(t)}(\theta) &= E\left[e^{\theta \sum_{i=1}^{A(t)} X_i}\right] \\
&= \sum_A E\left[e^{\theta \sum_{i=1}^{A} X_i} \mid A(t) = A\right] Pr(A(t) = A) \\
&= \sum_A E\left[e^{\theta \sum_{i=1}^{A} X_i}\right] Pr(A(t) = A) \\
&= \sum_A E\left[e^{\theta X_1} e^{\theta X_2} \cdots e^{\theta X_A}\right] Pr(A(t) = A) \\
&= \sum_A \left(E\left[e^{\theta X_1}\right]\right)^A Pr(A(t) = A) \\
&= \sum_A E\left[\left(E\left[e^{\theta X}\right]\right)^A\right] Pr(A(t) = A) \\
&= E\left[\left(E\left[e^{\theta X}\right]\right)^{A(t)}\right] \\
&= M_{A(t)}(\log M_{\mathbf{X}}(\theta)) \ .
\end{aligned}$$

From the first line to the second line we use the conditional expectation. In the third line, we use the independence of $A(t)$ and $\mathbf{X}$. We get the fifth line because X_i's are *i.i.d.* . □

The above lemma can be applied recursively to derive the MGF bound of a scaled process through a series of scaling elements. For instance, if $\mathbf{X}$ and $\mathbf{Y}$ are *i.i.d.* then

$$\begin{aligned}
M_{(A^{\mathbf{X}})^{\mathbf{Y}}(t)}(\theta) &= M_{A^{\mathbf{X}}(t)}\left(\log M_Y(\theta)\right) \\
&= M_{A(t)}\left(\log M_X\left(\log M_Y(\theta)\right)\right) \ .
\end{aligned}$$

If $\mathbf{X}$ is additionally a loss process, then the last line further equals to $M_{A(t)}$ $(\log M_{XY}(\theta))$, by applying the concatenation property from Lemma 4.

4.2 Commutation

Here we show how to commute a series of a dynamic server element and a scaling element. As mentioned earlier, this property is instrumental for

expressing networks with flow transformations in convolution-form. Before that we briefly depict the dynamic server element in Figure 4.6.

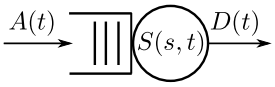


Figure 4.6: A dynamic server element with arrival process $A(t)$, service process $S(s,t)$, and departure process $D(t)$.

Lemma 6. *(Commuting Dynamic Server and Scaling Elements) Consider a system with an arrival process $A(t)$ which goes through a dynamic server $S(s,t)$ and then a scaling element $\mathbf{X} = (X_i)_{i\geq 1}$. In another system, $A(t)$ goes first through $\mathbf{X}$ and then through the exact dynamic server $T(s,t) := \sum_{i=A(s)+1}^{A(s)+S(s,t)} X_i$, as shown in Figure 4.7. If $A(t)$, $\mathbf{X}$, and $S(s,t)$ are inde-*

Figure 4.7: Commuting dynamic server and scaling elements

pendent, then for all $t \geq 0$

$$E(t) \leq D^{\mathbf{X}}(t)\,, \tag{4.9}$$

where $D^{\mathbf{X}}(t)$ and $E(t)$ are the departure processes in the two systems. Moreover, if A, S and $\mathbf{X}$ are independent, and $\mathbf{X}$ is stationary, then for any $\theta > 0$ we have $M_{T(s,t)}(-\theta) = M_{S(s,t)}(\log M_X(-\theta))$.

Proof. From the definition of the scaling and service elements we have immediately for some $t \geq 0$

$$\begin{aligned} E(t) &= \inf_{0\leq s\leq t} \left\{A^{\mathbf{X}}(s) + T(s,t)\right\} \\ &= \inf_{0\leq s\leq t} \left\{\sum_{i=1}^{A(s)} X_i + \sum_{i=A(s)+1}^{A(s)+S(s,t)} X_i\right\} \\ &= \inf_{0\leq s\leq t} \sum_{i=1}^{A(s)+S(s,t)} X_i \leq \sum_{i=1}^{D(t)} X_i = D^{\mathbf{X}}(t)\,. \end{aligned}$$

The rest of proof follows by successive conditioning. □

The lemma ensures that backlog/delay processes in the transformed system are bigger in distribution than in the original system. We also point out that although the expression of $T(s,t)$ depends on $A(s)$, the expression of $T(s,t)$'s Laplace transform is sufficient to elegantly carry out end-to-end computations. The next section presents a detailed end-to-end delay analysis of a network with flow transformations by means of Lemma 6.

4.3 End-to-End Delay Bounds

In this section we compute end-to-end delays in a flow transformation network consisting of a series of alternate scaling and service elements. In particular we demonstrate that by using Lemma 6, which allows the transformation of this network in a convolution-form network by repeatedly commuting scaling and service elements, the end-to-end delays scale linearly in the number of service elements. In contrast, we also show that by applying the alternative node-by-node and additive analysis with the new defined scaling element, end-to-end delays scale quadratically.

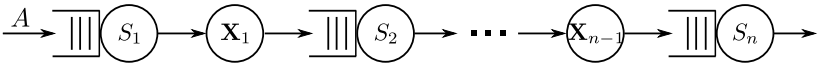


Figure 4.8: A flow transformation network consisting of sequence of alternating service and scaling elements.

We consider the flow transformation network scenario from Figure 4.8. A stationary arrival process $A(t)$ crosses a series of alternate service and scaling elements denoted by S_i and $\mathbf{X}_i$, respectively. We assume that all the service and scaling processes are stationary and (mutually) independent. This network scenario can be seen as a flow's view, or a part of it, in a network with loss, random routing, transcoding, or more generally, certain agile data operation.

4.3.1 Transformation in Convolution-Form

The next theorem provides a bound on the end-to-end delay and the corresponding order of growth for the network shown in Figure 4.8.

Theorem 9. *(End-to-End Delays in a flow transformation network) Consider the network scenario from Figure 4.8 where a stationary arrival process* $A(t)$

crosses a series of alternate stationary and (mutually) independent service and scaling elements denoted by $S_1, S_2, \ldots, S_n$ and i.i.d. $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_{n-1}$, respectively. Assume that the MGF bounds on the arrivals and services are $M_{A(s,t)}(\theta) \leq e^{\theta r(\theta)(t-s)}$ and $M_{S_k(t)}(-\theta) \leq e^{-\theta C_k t}$, for $k = 1, \ldots, n$, and some $\theta > 0$. Under a stability condition, to be explicitly given in the proof, we have the following end-to-end steady-state delay bounds for all $d \geq 0$

$$Pr\Big(W > d\Big) \leq K^n b^d \,, \tag{4.10}$$

where the constants K and b are to be given in the proof as well. Moreover, the ε-quantiles scale as $\mathcal{O}(n)$, for some $\varepsilon > 0$.

Proof. Fix $t, d \geq 0$ and denote for convenience for all $k, s \geq 0$

$$A^{(k)}(s) := \left(\cdots \left(A^{\mathbf{X}_1}\right)^{\mathbf{X}_2} \cdots \right)^{\mathbf{X}_k} (s) \tag{4.11}$$

the iterative scaling of A by $\mathbf{X}_1,\ \mathbf{X}_2,\ \ldots,\ \mathbf{X}_k$. Also, for $k \geq 0$ we introduce the scaled processes $U_k(s, u_k)$ defined as

$$U_0(s, u_0) = A(s) \,,$$

for $u_0 = s$, and then recursively

$$U_k(s, u_k) = \Big(U_{k-1}(s, u_{k-1}) + S_k(u_{k-1}, u_k)\Big)^{\mathbf{X}_k} \tag{4.12}$$

for $k \geq 1$ and $u_{k-1} \leq u_k$.

We are going to prove the claim from the theorem by induction. For $k \geq 1$ we let the following two statements $(\mathcal{S}_1)$ and $(\mathcal{S}_2)$ for the induction process:

$$(\mathcal{S}_1) \quad Pr\Big(W_k(t) > d\Big) \leq \sum_{0 \leq s \leq t-d} \sum_{s \leq u_1 \leq \cdots \leq u_{k-1} \leq t} Pr\left(A^{(k-1)}(t-d) > U_{k-1}\left(s, u_{k-1}\right) + S_k\left(u_{k-1}, t\right)\right) \,,$$

and for fixed s and u_k

$$(\mathcal{S}_2) \quad \Big(A^{(k-1)}(s) + T_{k-1} \otimes S_k(s, u_k)\Big)^{\mathbf{X}_k} = \inf_{s \leq u_1 \leq \cdots \leq u_k} U_k\left(s, u_k\right) \,,$$

where T_k is defined recursively as $T_0(0) = 0$, $T_0(s) = \infty$ for all $s > 0$, and

$$T_k(s,t) := \sum_{i=A^{(k-1)}(s)+1}^{A^{(k-1)}(s)+T_{k-1}\otimes S_k(s,t)} X_{k,i} \ . \tag{4.13}$$

For the initial step of the induction, i.e., $k = 1$, we have

$$\begin{aligned} Pr\Big(W_1(t) > d\Big) &= Pr\left(A(t-d) > D(t)\right) \\ &\leq Pr\left(A(t-d) > A \otimes S_1(t)\right) \\ &\leq \sum_{0\leq s\leq t-d} Pr\left(A(t-d) > U_0\left(s,s\right) + S_1(s,t)\right) \ , \end{aligned} \tag{4.14}$$

which verifies the first statement $(\mathcal{S}_1)$. In the first line $D(t)$ is the output process from the service element S_1 and we used the equivalence $W_1(t) > 0 \Leftrightarrow A(t-d) > D(t)$, in the second line we used the definition of the dynamic server, and in the third line we expanded the $(min, +)$ convolution and applied the union bound.

In turn, for the second statement $(\mathcal{S}_2)$, we have

$$\begin{aligned} \left(A(s) + T_0 \otimes S_1(s,u_1)\right)^{\mathbf{X}_1} &= \left(A(s) + S_1\left(s,u_1\right)\right)^{\mathbf{X}_1} \\ &= \inf_{s\leq u_1} U_1\left(s,u_1\right) \ , \end{aligned}$$

which verifies $(\mathcal{S}_2)$. In the first line we used that $T_0(0) = 0,\ T_0(s) = \infty$ for $s > 0$, and then we used the definition of $U_k(s,u_k)$ from Eq. (4.12).

For the inductive step we assume that $(\mathcal{S}_1)$ and $(\mathcal{S}_2)$ hold for some $k \geq 1$ and we will prove them for $k+1$.

First, we observe that after iteratively commuting for k times the service and service elements from Figure 4.8, we obtain the system from Figure 4.9; note that, according to Lemma 6, the output before the k^{th} service element in the transformed system is smaller than in the original system.

Figure 4.9: Transformation of the system from Figure 4.8 after iteratively applying Lemma 6 for k times.

Using the argument from Eq. (4.14) we can write for the end-to-end delay

until the $k+1^{th}$ scaling element

$$
\begin{aligned}
& Pr\left(W_{k+1}(t) > d\right) \\
\leq & \sum_{0 \leq s \leq t-d} Pr\Big(A^{(k)}(t-d) > A^{(k)}(s) + T_k \otimes S_{k+1}(s,t)\Big) \\
\leq & \sum_{0 \leq s \leq t-d} \sum_{s \leq u_k \leq t} Pr\Big(A^{(k)}(t-d) > \\
& \qquad A^{(k)}(s) + T_k(s,u_k) + S_{k+1}(u_k,t)\Big) \\
\leq & \sum_{0 \leq s \leq t-d} \sum_{s \leq u_k \leq t} Pr\Big(A^{(k)}(t-d) > \\
& \qquad \Big(A^{(k-1)}(s) + T_{k-1} \otimes S_k(s,u_k)\Big)^{\mathbf{X}_k} + S_{k+1}(u_k,t)\Big) \\
\leq & \sum_{0 \leq s \leq t-d} \sum_{s \leq u_1 \leq \cdots \leq u_k \leq t} Pr\Bigg(A^{(k)}(t-d) > \\
& \qquad\qquad U_k(s,u_k) + S_{k+1}(u_k,t)\Bigg)\,,
\end{aligned}
$$

which proves statement $(\mathcal{S}_1)$ for $k+1$. In the third line we expanded the convolution and applied the union bound, and finally we used the induction hypothesis for $(\mathcal{S}_2)$ together with the union bound.

Lastly, for the induction argument, we need to prove the statement $(\mathcal{S}_2)$ for $k+1$. We have

$$
\begin{aligned}
& \Big(A^{(k)}(s) + T_k \otimes S_{k+1}(s,u_{k+1})\Big)^{\mathbf{X}_{k+1}} \\
= & \inf_{s \leq u_k \leq u_{k+1}} \Big(A^{(k)}(s) + T_k(s,u_k) + S_{k+1}(u_k,u_{k+1})\Big)^{\mathbf{X}_{k+1}} \\
= & \inf_{s \leq u_k \leq u_{k+1}} \Bigg(\Big(A^{(k-1)}(s) + T_{k-1} \otimes S_k(s,u_k)\Big)^{\mathbf{X}_k} \\
& \qquad\qquad + S_{k+1}(u_k,u_{k+1})\Bigg)^{\mathbf{X}_{k+1}} \\
= & \inf_{s \leq u_k \leq u_{k+1}} \left(U_k(s,u_k) + S_{k+1}(u_k,u_{k+1})\right)^{\mathbf{X}_{k+1}} \\
= & \inf_{s \leq u_k \leq u_{k+1}} U_{k+1}(s,u_{k+1})\,,
\end{aligned}
$$

which proves the claim. In the next to last line we used the induction hypothesis and then the definition of $U_k(s, u_k)$ from Eq. (4.12). The induction argument is thus complete.

Next we compute the end-to-end delay bound on $W_n(t)$ by using the statement $(\mathcal{S}_1)$ for $k = n$. We have

$$\begin{aligned}
Pr\Big(W_n(t) > d\Big) &\leq \sum_{0\leq s\leq t-d} \sum_{s\leq u_1\leq\cdots\leq u_{n-1}\leq t} Pr\Bigg(A^{(n-1)}(t-d) \\
&> \Bigg(\dots \Big(\left(A(s) + S_1(s,u_1)\right)^{\mathbf{X}_1} + S_2(u_1,u_2)\Big)^{\mathbf{X}_2} \\
&\qquad + \cdots + S_{n-1}(u_{n-2},u_{n-1})\Bigg)^{\mathbf{X}_{n-1}} + S_n(u_{n-1},t)\Bigg) \\
&\leq \sum_{0\leq s\leq t-d} \sum_{s\leq u_1\leq\cdots\leq u_{n-1}\leq t} \\
&\quad Pr\Bigg(\Bigg(\dots \Big(\left(A(t-d-s) - S_1(s,u_1)\right)^{\mathbf{X}_1} - S_2(u_1,u_2)\Big)^{\mathbf{X}_2} \\
&\qquad - \cdots - S_{n-1}(u_{n-2},u_{n-1})\Bigg)^{\mathbf{X}_{n-1}} > S_n(u_{n-1},t)\Bigg) ,
\end{aligned}$$

after repeatedly applying the stationary bounds in Eqs. (4.5) and (4.6) from Lemma 4. Next, using the Chernoff bound for some $\theta > 0$, we obtain

$$\begin{aligned}
Pr\Big(W_n(t) > d\Big) \leq \sum_{0\leq s\leq t-d} \sum_{s\leq u_1\leq\cdots\leq u_{n-1}\leq t} & e^{a_{n-1}r(a_{n-1})(t-d-s)} \\
& e^{-a_{n-1}C_1(u_1-s)} e^{-a_{n-2}C_2(u_2-u_1)} \cdots e^{-a_0 C_n(t-u_{n-1})} .
\end{aligned}$$

Here we recursively used Lemma 5 and obtained the bounds

$$E\left[e^{\theta A^{(k)}(t)}\right] \leq e^{a_k r(a_k) t} , \tag{4.15}$$

for $k \geq 0$ (refer to Eq. (4.11) for the definition of $A^{(k)}(t)$). Note that $a_0 = \theta$ and a_k's for $k \geq 1$ do not depend on $A(t)$ but instead are formed iteratively as

$$a_k = \log M_{X_1}\left(\log M_{X_2}\left(\cdots\left(\log M_{X_k}(\theta)\right)\cdots\right)\right) \tag{4.16}$$

in the case when $X_{1,i}$'s are *i.i.d.*. More generally, when $\mathbf{X}_k$'s are MMSPs,

the recursion is given by

$$a_k = \log \mathrm{sp}\left(\phi_{X_1}\left(\log \mathrm{sp}\left(\phi_{X_2}\left(\cdots \log \mathrm{sp}\left(\phi_{X_k}(\theta)\lambda_k\right)\cdots\right)\lambda_2\right)\right)\lambda_1\right) \quad (4.17)$$

by Lemma 5, where the diagonal matrices $\phi_{\mathbf{X}_{k+1}}(a_k)$ are defined as $\phi(\theta)$ from Lemma 5, but now relative to each $\mathbf{X}_{k+1}$ for $k \geq 0$.

Denoting

$$b = \sup_{k=0,\ldots,n-1} e^{-a_k C_{n-k}} \quad (4.18)$$

we can bound the sums as

$$\begin{aligned} &Pr\Big(W_n(t) > d\Big) \\ &\quad \leq \sum_{0\leq s\leq t-d} \sum_{s\leq u_1\leq\cdots\leq u_{n-1}\leq t} e^{a_{n-1}r(a_{n-1})(t-d-s)} b^{t-s} \\ &\quad = b^d \sum_{0\leq s\leq t-d} \binom{t-s+n-1}{n-1} e^{(\log b + a_{n-1}r(a_{n-1}))(t-d-s)} \,. \end{aligned}$$

where $\binom{t-s+n-1}{n-1}$ means "pick $t-s+1$ things into $n-1$ slots". Let $q = e^{\log b + ar(a)}$ and impose the following stability condition

$$a_{n-1}r(a_{n-1}) + \log b < 0 \,, \quad (4.19)$$

i.e., $q \in (0,1)$. Let $p = 1-q$, we can further reformulate the bound as

$$\begin{aligned} & Pr(W_n(t) > d) \\ \leq\ & b^d \frac{1}{p^n} \sum_{0\leq s<t-d} \binom{t-s+n-1}{n-1} p^n q^{t-d-s} \\ =\ & b^d \frac{1}{p^n}\frac{1}{q^d} \sum_{0\leq s<t-d} \binom{t-s+n-1}{n-1} p^n q^{t-s} \qquad \text{let } t\to\infty \\ \leq\ & b^d \frac{1}{p^n}\frac{1}{q^d} \sum_{t-s>d}^{\infty} \binom{t-s+n-1}{n-1} p^n q^{t-s} \,. \end{aligned}$$

The sum part of the last line has the form of negative binomial distribution. We can interprete it into the sum of geometric r.v.'s. That is $X = \sum_{i=1}^{n} X_i$. X represents the number of trials needed to obtain n successes. X_i is the number of trials needed to obtain i-th success. X_i's are independent. So $M_X(\theta') = (M_{X_i}(\theta'))^n = \left(\frac{pe^{\theta'}}{1-qe^{\theta'}}\right)^n$. And now the sum can be transformed,

through using Chernoff bound, as below

$$\begin{aligned} & Pr(W_n(t) > d) \\ \leq\ & b^d \frac{1}{p^n}\frac{1}{q^d} Pr(X \geq n+d) \\ \text{chernoff bound}\ \leq\ & b^d \frac{1}{p^n}\frac{1}{q^d} \inf_{\theta' \in [0,\infty)} \left\{ e^{-\theta'(n+d)} M_X(\theta') \right\} . \end{aligned}$$

In order to get the infimum we differentiate θ'. The point exists when

$$e^{\theta'} = \frac{d}{q(n+d)} \tag{4.20}$$

and note $\theta' \geq 0$, it implies $d \geq \frac{nq}{1-q}$. Therefore we get the delay bound

$$\begin{aligned} & Pr(W_n(t) > d) \\ \leq\ & b^d \frac{1}{p^n}\frac{1}{q^d} \left(\frac{q(n+d)}{d} \right)^{n+d} \left(\frac{p\frac{d}{q(n+d)}}{1 - q\frac{d}{q(n+d)}} \right)^n \\ =\ & b^d \left(\frac{n+d}{n} \right)^n \left(\frac{n+d}{d} \right)^d \\ =\ & b^d \left(\frac{(1+d/n)^{1+d/n}}{(d/n)^{d/n}} \right)^n \qquad \to K^n b^d . \end{aligned}$$

We note that using the stability condition from Eq. (4.19) and taking $t \to \infty$ proves the result from Eq. (4.10). Finally, the order of growth of the ε-quantiles for some $0 < \varepsilon < 1$ follow directly from Eq. (4.10) by observing that K is bounded in n. The proof is now complete. □

4.3.2 Alternative Node-by-Node Analysis

Here we show that end-to-end delays obtained by a straightforward node-by-node analysis fundamentally differ from those obtained in Theorem 9 in that the order of growth is quadratic in number of nodes, as opposed to linear.

Consider the notations from Figure 4.10, where A_k represents the output from the k^{th} service element, for $k = 1, \ldots, n$. Also, denote by convention $A_0 = A$ and let $\mathbf{X}_0 = \mathbf{1}$.

Assume as in Theorem 9 that $M_{A(s,t)}(\theta) \leq e^{\theta r(\theta)(t-s)}$ and $M_{S_k(t)}(-\theta) \leq e^{-\theta C_k t}$, for $k = 1, \ldots, n$, and some $\theta > 0$. Assume also the stability con-

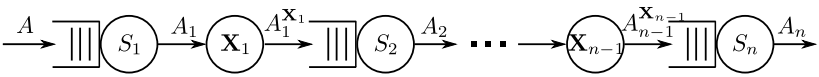


Figure 4.10: A flow transformation network consisting of sequence of alternating service and scaling elements.

dition $a_{k-i+1}C_i > a_k r(a_k)$ for all $1 \leq i \leq k$, where a_k's are defined as in Eq. (4.16) or (4.17).

We will first prove by induction the statement

$$(\mathcal{S}) \quad E\left[e^{\theta A_k^{\mathbf{X}_k}(t)}\right] \leq M_k e^{a_k r(a_k) t}\,, \tag{4.21}$$

where $M_0 = 1$ and for $k \geq 1$

$$M_k = \prod_{i=1}^{k} \frac{1}{a_{k-i+1}C_i - a_k r(a_k)}\,. \tag{4.22}$$

The statement is immediately true for $k = 0$ from the initial assumptions. Let us now assume that the statement $(\mathcal{S})$ holds for k and prove it for $k+1$. We can write for the MGF of the output $A_{k+1}(s,t)$ from the $(k+1)^{th}$ service element (see [61])

$$\begin{aligned} E\left[e^{\theta A_{k+1}(s,t)}\right] &\leq E\left[e^{\theta\left(A_k^{\mathbf{X}_k}(t) - A_{k+1}(s)\right)}\right] \\ &\leq E\left[e^{\theta\left(A_k^{\mathbf{X}_k}(t) - A_k^{\mathbf{X}_k} \otimes S_{k+1}(s)\right)}\right] \\ &\leq \sum_{0 \leq u < s} M_k e^{a_k r(a_k)(t-u)} e^{-\theta C_{k+1}(s-u)} \\ &\leq \frac{M_k}{\theta C_{k+1} - a_k r(a_k)} e^{a_k r(a_k)(t-s)}\,. \end{aligned} \tag{4.23}$$

In the first line we used that the output at time t is dominated by the corresponding input. In the second line we used the definition of the dynamic server. In the third line we used the union bound and the induction hypothesis, and finally we estimated the sum by an integral.

Next, for the MGF of $A_{k+1}^{\mathbf{x}_{k+1}}(s,t)$, Lemma 5 yields

$$E\left[e^{\theta A_{k+1}^{\mathbf{x}_{k+1}}(s,t)}\right] = E\left[e^{a_1 A_{k+1}(s,t)}\right]$$

Since the bound on the MGF of $A_{k+1}(s,t)$ from Eq. (4.23) is derived for θ, we need to replace all the occurrences of θ by a_1 (note that all the a_k's depend on θ, according to their definitions from either Eq. (4.16) or (4.17)). Consequently, a_k is to be replaced by a_{k+1} for all $k \geq 0$, which yields the MGF bound

$$\begin{aligned} E\left[e^{\theta A_{k+1}^{\mathbf{x}_{k+1}}(s,t)}\right] &\leq \frac{1}{a_{k+1}C_1 - a_{k+1}r(a_{k+1})} \\ \cdots \frac{1}{a_2 C_k - a_{k+1}r(a_{k+1})} &\frac{e^{a_{k+1}r(a_{k+1})(t-s)}}{a_1 C_{k+1} - a_{k+1}r(a_{k+1})} \\ &= M_{k+1}e^{a_{k+1}r(a_{k+1})(t-s)} \ , \end{aligned}$$

which proves that the statement $(\mathcal{S})$ holds for $k+1$, and thus the induction proof is complete.

In the following, having MGF bounds for the arrivals at the k^{th} service element, and the dynamic servers $S_k(s,t)$, we can derive corresponding per-node delay bounds $W_k(t)$. Using the same arguments as in Eq. (4.14) we obtain for all $k \geq 1$ and $d \geq 0$

$$Pr\Big(W_k > d\Big) \leq L_k e^{-\theta C_k d} \ , \tag{4.24}$$

where the prefactors L_k's are defined as

$$L_k = \prod_{i=1}^{k} \frac{1}{a_{k-i}C_i - a_{k-1}r(a_{k-1})} \ ,$$

Note that replacing all the occurrences of θ from L_k with a_1 yields the M_k's defined earlier in Eq. (4.22).

Let us next make the convenient choices

$$b_k = \inf_{i=1\ldots k} a_{k-i}C_i$$

such that $L_k \leq \left(\frac{1}{b_k - a_{k-1} r(a_{k-1})}\right)^k$, and

$$b = \sup_{k=1,\ldots,n} \frac{1}{b_k - a_{k-1} r(a_{k-1})} .$$

We can thus bound the bound on W_k by

$$Pr\Big(W_k > d\Big) \leq b^k e^{-\theta C_k d} ,$$

for all $k \geq 1$.

Finally, a bound on the end-to-end delay $W = \sum_k W_k$ can be formulated as the optimization problem

$$Pr\Big(W > d\Big) \leq \inf_{d_1 + \cdots + d_n = d} \left\{ b e^{-\theta C_1 d_1} + \cdots + b^n e^{-\theta C_n d_n} \right\} .$$

Letting $C = \sup_{k=1,\ldots,n} C_k$ we find the infimum (see [42])

$$Pr\Big(W > d\Big) \leq n b^{\frac{n+1}{2}} e^{-\frac{\theta}{n} C d} . \tag{4.25}$$

From here one may easily determine that the quantiles of the end-to-end delay grow as $\mathcal{O}\left(n^2\right)$, which proves the claim from the beginning of this section.

4.4 Numerical Evaluation

In this section we numerically compare the end-to-end delay bounds from Theorem 1 with those obtained by the alternative node-by-node analysis presented in Subsection 4.3.2, and illustrate the corresponding order of growths. Then we draw some insights on how the burstiness in arrival vs. scaling processes affect the results from Theorem 9.

For Theorem 1 we directly use the delay bound from Eq. (4.10). In turn, for the node-by-node analysis, we use the end-to-end delay bound

$$P\Big(W > d\Big) \leq \inf_{\sum_k d_k = d} \left\{ \sum_k L_k e^{-\theta C_k d_k} \right\} \tag{4.26}$$

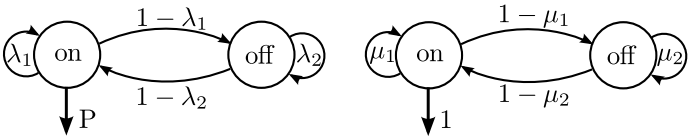
for which the infimum can be computed exactly using a convex optimization result from [42] (see also Eq. (4.24) for the values of L_k). We point out that we do not use the expression from Eq. (4.25) which was subject to several

Arrival Process	Scaling Process
Poisson	Bernoulli
MMOO	Bernoulli
Poisson	MMOO
MMOO	MMOO

Table 4.1: Arrival and scaling processes.

convenient bounding choices; these were made in order to derive a result which can concisely express the $\mathcal{O}(n^2)$ growth.

We consider the flow transformation network scenario from Figure 4.8 with two examples of arrival processes $A(t)$: Poisson with rate λ and Markov-Modulated On-Off (MMOO). The MMOO process is represented in Figure 4.11.(a) in terms of the transition probabilities λ_1 and λ_2, and also the peak-rate P, i.e., the process transmits at rate P while in state 'on' and is idle while in state 'off'. When $\lambda_1 + \lambda_2 = 1$ then $A(t)$ has independent increments and is thus a sum of Bernoulli random variables $B(\lambda_1)$. We consider the scaling (loss) processes from Figure 4.8 as MMOO processes as represented in Figure 4.11.(b) (note that for the loss process the rate while in the 'on' state is 1). See Table 4.1.



(a) Arrival process $A(t)$ (b) Scaling processes $\mathbf{X}_i$'s

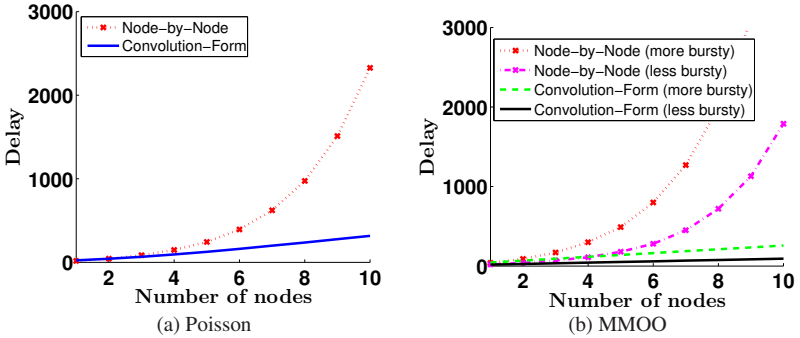
Figure 4.11: Representation of Markov-Modulated On-Off (MMOO) processes.

For the MMOO process $A(t)$ from Figure 4.11.(a) we have the following bound on its MGF [36]

$$E\left[e^{\theta A(t)}\right] \leq e^{\theta r(\theta) t},$$

where $r(\theta) = \frac{1}{\theta} \log \frac{\lambda_1 e^{\theta P} + \lambda_2 + \sqrt{(\lambda_1 e^{\theta P} + \lambda_2)^2 - 4(\lambda_1 + \lambda_2 - 1)e^{\theta P}}}{2}$. Similar bounds apply for the MGF's of the cumulative processes of the $\mathbf{X}_i$'s, with different parameters.

In the case when $X_{i,1}$'s are *i.i.d.* Bernoulli $B(p)$ for some $0 < p < 1$



(a) Poisson (b) MMOO

Figure 4.12: Scaling of end-to-end delay bounds with Theorem 9 and the method of node-by-node analysis (arrival process with rate 1 (Poisson in (a) and two MMOO's in (b) with $P = 2$ (less bursty) and $P = 3$ (more bursty) and $\lambda_1 = \frac{1}{P}$), Bernoulli (0.75) scaling processes, violation probability $\varepsilon = 10^{-3}$)

(i.e., $\mu_1 = p$ in Figure 4.11.(b)) and $A(t)$ is Poisson then we have explicit expressions for the values of a_k and $r(a_k)$'s which appear in Eq. (4.15), and in the final expressions for end-to-end delays (both with Theorem 9 and the node-by-node analysis). These are

$$a_k = \log\left(1 + p^k\left(e^\theta - 1\right)\right), \quad r(a_k) = \frac{\lambda p^k\left(e^\theta - 1\right)}{\log\left(1 + p^k\left(e^\theta - 1\right)\right)}\,.$$

In turn, if $A(t)$ is a sum of Bernoulli random variables $B(\lambda_1)$, i.e., $\lambda_1 + \lambda_2 = 1$, then the expressions of $r(a_k)$'s become

$$r(a_k) = \frac{\log\left(1 + \lambda_1\left(\left(1 + p^k\left(e^\theta - 1\right)\right)^P - 1\right)\right)}{\log\left(1 + p^k\left(e^\theta - 1\right)\right)}\,.$$

We use the following numerical settings. The average rates of the arrivals are normalized to one packet per one time unit. The utilization at the first node is .75, i.e., $C_1 = 1.33$. The capacities at the rest of the nodes are set as $C_k = \frac{a_{n-1}}{a_{n-k}} C_1$, such that there is no loss in accuracy in the end-to-end delay from Theorem 1, as a result of the bounding from Eq. (4.18).

Figures 4.12.(a,b) illustrate the order of growths of the end-to-end delay bounds obtained with Theorem 1 and the node-by-node analysis for $n = 1, \ldots, 10$ service elements in Figure 4.8, by plotting the corresponding ε-

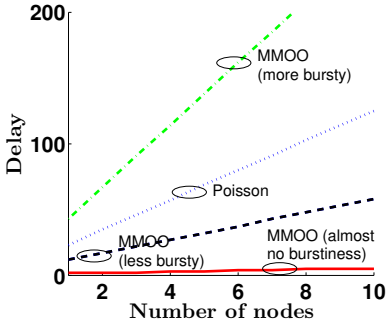


Figure 4.13: Arrivals' burstiness dominates scalings' burstiness (three MMOO scalings ($\mu_1 = .67, .75, .99$, average=.75) for each arrivals with average 1 (three MMOO's with $\lambda_1 = .7, .4, .01$, $P = 2$), $\varepsilon = 10^{-3}$)

quantiles (in time units) with $\varepsilon = 10^{-3}$. We consider Bernoulli scaling processes (with $\mu_1 = 1 - \mu_2 = .75$ in Figure 4.11.(b)) and different arrival processes (Poisson in (a) and two MMOO's with different levels of burstiness in (b); the parameters are displayed in the caption). For all the arrival processes the figure clearly illustrates the $\mathcal{O}(n)$ vs. $\mathcal{O}(n^2)$ order of growths of the end-to-end delays. Moreover, as illustrated by (b), the $\mathcal{O}(n)$ results are much less sensitive to the arrivals' burstiness than the $\mathcal{O}(n^2)$ results, indicating large pre-constants in the latter.

Figure 4.13 illustrates the impact of burstiness in the scaling processes over the burstiness in the arrival processes for $n = 1, \ldots, 10$ service elements in Figure 4.8. We consider four arrival processes (one Poisson and three MMOO's, each with three levels of burstiness, by adjusting the transition probability λ_1 from Figure 4.11.(a)). For each arrival process we consider three scaling processes as MMOO's, each with three levels of burstiness, by adjusting μ_1 from Figure 4.11.(b), while keeping the same average rate of .75 (for the values of all the parameters see the caption).

Interestingly, the figure indicates that the burstiness in the arrivals completely dominates the burstiness in the scaling processes (the plots with different scalings' burstiness are visually indistinguishable, for all four arrival cases). This phenomenon can be justified by the independence assumption between arrival and scaling processes, i.e., unless some forms of correlations exist between the two (e.g., always drop when bursty traffic occurs), the scalings' burstiness has only a negligible impact on the arrivals' burstiness. The figure also illustrates that when there is almost no burstiness in the arrivals

(i.e., $\lambda_1 \to 0$ which means that the arrival process looks roughly like a periodic source of 2 (the peak rate P) packets every two time units) the delays converge to roughly zero time units. As this would actually be the case for such periodic arrivals, the figure provides evidence that the delay bounds from Theorem 9 are reasonably accurate.

Since flow transformations are manifold and frequent, we believe that the stochastic scaling elements open up the modeling scope of the stochastic network calculus widely. To that end, we have introduced a versatile stochastic scaling element and have shown how networks with flow transformations could still be expressed in convolution-form. Consequently, as demonstrated analytically as well as by numerical examples, the fundamental scaling properties of the network calculus are retained. These stochastic scaling element models lay the theoretical foundation for a rich set of new applications of network calculus, ranging from lossy networks over dynamic routing up to network coding scenarios. While some of these may require further thought, e.g., how to deal with non-Markov modulated scaling processes, many opportunities lie ahead.

In the previous chapter we also provided a model on the flow transformations, which is based on the sample path functions and the stochastic curvewise bounds. Together with the model introduced in this chapter, we seemingly have two options when we model the flow transformation. That is not an accurate opinion. Actually, the model with bounding functions is more general than the latter one. If we model the stochastic scaling element with the form $A^{\mathbf{x}}(t)$, we can also construct the stochastic scaling curves such that we can use the methods of the former model. However, there may be also other ways, known or unknown, to model the flow transformations and we may not know so many details (like the way to transform or the data granularity to be transformed), but possibly still know the bounding curves, then we simply use the former model. But that does not mean that the former one is better. On the contrary, many times we do know many details of the scaling processes, or assume to know priorly and later do some fitting work. In this case, the latter model will gain more accuracy and flexibility when doing scaler-server-commuting as well as analysing performance.

Chapter 5

Deconstruction of Stochastic Data Scaling Element

In Chapter 3 and 4 we introduced two alternative data scaling elements to model the flow transformation. The new models allow us to express the network with flow transformations in convolution-form and derive the performance bounds. On an abstract level, the demultiplexing of flows inside the network is a very obvious and yet highly important case of the flow transformation. While we have in Chapter 3 addressed it using the sample path modeling of the scaling element, in this chapter, we come up with a new way to solve this analytically hard problem by using the stochastic scaling element for arrivals introduced in Chapter 4, and conversely by dint of analyzing the demultiplexing, provide an insightful *deconstruction* of the scaling element. This helps us to deeper understand the previous scaling models and advance the performance bounds derivation. At the same time, recall that, with the classical queueing networks theory, the demultiplexing analysis is typically possible for the class of Poisson arrivals and Bernoulli demultiplexing processes. We can now extend these classes in a greater generality, e.g., by considering Markov modulated demultiplexing processes and *general classes* of the arrival processes (i.e., of flows) subject to demultiplexing. This is the power of the scaling element (see Section 4.1). The results of this chapter are from the joint work with F. Ciucu and J. Schmitt [145].

By the abstract demultiplexing operation we mean here the separation of a flow into multiple subflows, one of which is subject to the analysis (e.g., with respect to delay). Although in Section 3.3.1 we attempted to model the demultiplexing with the sample paths of the scaling element and then stochas-

tically bound these sample paths accordingly, the indirect approach still restricts itself on the ambiguity of the description. On the other hand, many concrete real-world utilities of the abstract demultiplexing like losing part of a data flow in, e.g., a wireless transmission, or distributing a data flow to a set of servers for load balancing, or simply a randomized multi-path routing require a dynamic expression of the scaling element. In the previous chapter we did not really touch this problem although we have already provided the required expression, because on one hand, the main focus of the previous chapter is to preserve the convolution-form expression of the network even after using the new scaling element, on the other hand, in that scope the meaning of the model can not really overtake the sample path description in Chapter 3. Now, the approach coming up in this chapter provides a new viewpoint to model the demultiplexing, which is based on finding an equivalent formulation for the demultiplexing as a leftover service curve computation problem. And this explains the meaning of all the scaling elements (deterministic and two stochastic ones) used for demultiplexing. Interestingly, with respect to the achievable delay bounds the new method and the one in Section 4.3.1 perform quite differently. Although the new method can not completely dominates the other (see Section 5.4), it still shows a clear advantage in scenarios where the subflow of interest is rather small and only rarely outperformed by the other.

5.1 A Novel Model for Flow Demultiplexing

In this section, we introduce a *novel model* for the operation of demultiplexing a flow into two flows.

Let us consider that a flow carried between source and destinations, is demultiplexed into two subflows, for example, a part of the original flow is routed to another destination out of the source node. We denote these subflows as two arrival processes $A^{(1)}(t), A^{(2)}(t)$ satisfying

$$A(t) = A^{(1)}(t) + A^{(2)}(t) \,,$$

for all $t \geq 0$. If we describe the splitting operation on the data level as an indicator function $\mathbf{1}_{\{\text{“this data goes to destination (1)”}\}}$, which equals to 1 if term true, 0 otherwise, we have that $A^{(1)}(t) = \sum_{i=1}^{A(t)} \mathbf{1}_{\{\text{“data } i \text{ goes to destination (1)”}\}}$. Denoting this indicator function for data i as X_i, we utilize the *scaling element* $\mathbf{X} = (X_i)_{i\geq 1}$ and denote all the data to destination (1) as the scaled arrivals

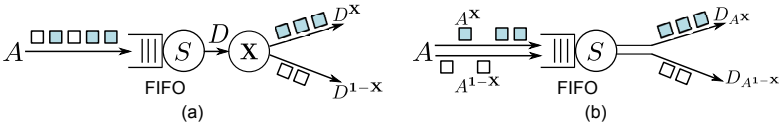


Figure 5.1: Two equivalent systems for the demultiplexing operation. In (a), the output process $D(t)$ is demultiplexed according to a scaling process $\mathbf{X}$. In (b), the input process $A(t)$ is virtually demultiplexed according to the *same* $\mathbf{X}$. In both (a) and (b), the node S runs FIFO scheduling.

$A^{\mathbf{X}}(t)$,

$$A^{(1)}(t) = \sum_{i=1}^{A(t)} X_i = A^{\mathbf{X}}(t) \ , \ \forall t \geq 0 \ .$$

Clearly, with $\mathbf{1} = (1, 1, \dots)$ we get $A^{(2)}(t) = A^{\mathbf{1}-\mathbf{X}}(t)$. Thus the demultiplexing is represented as

$$A(t) = A^{\mathbf{X}}(t) + A^{\mathbf{1}-\mathbf{X}}(t) \ .$$

Note, as we assume that the demultiplexing operation happens instantaneously, the scaling element has no queue. For brevity we focus on the particular case of two subflows demultiplexing. In general, the demultiplexing operation into n subflows can be modeled with multiple scaling processes $\mathbf{X}^1, \dots, \mathbf{X}^n$, such that $\sum_{j=1}^{n} \mathbf{X}^j = 1$.

Let us now consider a work-conserving network node, denoted by S, with arrival and departure processes $A(t)$ and $D(t)$, respectively. We assume that a demultiplexing process $\mathbf{X}$, which is independent from the data flows, splits $D(t)$ into two sub-processes: $D^{\mathbf{X}}(t)$ and $D^{\mathbf{1}-\mathbf{X}}(t)$ (see Figure 5.1.(a)). Based on the demultiplexing process $\mathbf{X}$, we can virtually split the arrival process $A(t)$ into two sub-processes, $A^{\mathbf{X}}(t)$ and $A^{\mathbf{1}-\mathbf{X}}(t)$, and we denote the corresponding (virtual) output processes by $D_{A^{\mathbf{X}}}(t)$ and $D_{A^{\mathbf{1}-\mathbf{X}}}(t)$ (see Figure 5.1.(b)). The next lemma establishes the equivalence between the two systems from Figures 5.1.(a,b), and it is instrumental for our proposed approach to analyze queueing systems with general demultiplexing processes.

Lemma 7. *(Equivalent Systems for the Demultiplexing Operation) Consider the systems (a) and (b) depicted in Figure 5.1 and described above. If the node is locally FIFO then the two systems are equivalent in the sense that for*

all $t \geq 0$

$$\begin{aligned} D^{\mathbf{X}}(t) &= D_{A^{\mathbf{X}}}(t) \text{ and} \\ D^{\mathbf{1}-\mathbf{X}}(t) &= D_{A^{\mathbf{1}-\mathbf{X}}}(t) \ . \end{aligned} \tag{5.1}$$

The proof is straightforward and relies on the assumption of *locally* FIFO scheduling, i.e., $A(t)$'s packets are served in the order of their arrivals.

Proof. Let $t \geq 0$. The departure process $D(t)$ from system (a) can be represented as a sequence $d_1 d_2 \cdots d_{D(t)}$, where $d_i = i$, i.e., d_i's stand for the sequence numbers of $D(t)$'s packets. Then, because the scaling operation does not alter the order of the packets' arrivals, with some abuse of notation, the $\mathbf{X}$-scaled departure processes $D^{\mathbf{X}}(t)$ and $D^{\mathbf{1}-\mathbf{X}}(t)$ can be represented as the subsequences

$$\begin{aligned} D^{\mathbf{X}}(t) &= d_{x_1} d_{x_2} \cdots d_{x_i} \cdots d_{x_m} \text{ and} \\ D^{\mathbf{1}-\mathbf{X}}(t) &= d_{y_1} d_{y_2} \cdots d_{y_j} \cdots d_{y_n} \ , \end{aligned}$$

where $m, n \in [0, D(t)], m + n = D(t), x_i, y_j \in [0, D(t)], x_i < x_{i+1}, y_j < y_{j+1}$, and $x_i \neq y_j$, $\forall i \in [0, m]$ and $j \in [0, n]$. That means, those packets in $D(t)$ are mutually demultiplexed into $D^{\mathbf{X}}(t)$ and $D^{\mathbf{1}-\mathbf{X}}(t)$. In turn, because the node S runs FIFO scheduling, the arrival process $A(t)$ must be represented (again, with abuse of notation) as the following sequence

$$A(t) = d_1 d_2 \cdots d_{D(t)} d_{D(t)+1} \cdots d_{A(t)} \ ,$$

and furthermore after using the same scaling process $\mathbf{X}$ onto this sequence of arrivals we will get the virtual sub-processes $A^{\mathbf{X}}(t)$ and $A^{\mathbf{1}-\mathbf{X}}(t)$. So the starting part must stay the same as for the $\mathbf{X}$-scaled departures, and the remainder arrival packets are $\mathbf{X}$-scaled in the similar ways (with abuse of notation). Thus, we have

$$\begin{aligned} A^{\mathbf{X}}(t) &:= d_{x_1} d_{x_2} \cdots d_{x_i} \cdots d_{x_m} d_{x_{m+1}} \cdots d_{x_{m+a}} \cdots d_{x_{m+u}} \text{ and} \\ A^{\mathbf{1}-\mathbf{X}}(t) &:= d_{y_1} d_{y_2} \cdots d_{y_j} \cdots d_{y_n} d_{y_{n+1}} \cdots d_{y_{n+b}} \cdots d_{y_{n+v}} \ , \end{aligned}$$

where $u, v \in [0, A(t) - D(t)]$, $u + v = A(t) - D(t)$, $x_{m+a}, y_{n+b} \in [0, A(t)]$, $x_{m+a} < x_{m+a+1}$, $y_{n+b} < y_{n+b+1}$, and $x_{m+a} \neq y_{n+b}$, $\forall a \in [0, u]$ and $b \in [0, v]$. So far, we only changed the notation and the departed sequences up to time t are still those two sequences $d_{x_1} d_{x_2} \cdots d_{x_i} \cdots d_{x_m}$ and $d_{y_1} d_{y_2} \cdots d_{y_j} \cdots d_{y_n}$. We can find a matched sequence of $d_{x_1} \cdots d_{x_i} \cdots d_{x_m}$

in $A^{\mathbf{X}}(t)$, which proves that

$$D_{A^{\mathbf{X}}}(t) = d_{x_1} d_{x_2} \cdots d_{x_i} \cdots d_{x_m} \ ,$$

and thus $D^{\mathbf{X}}(t) = D_{A^{\mathbf{X}}}(t)$. Similarly, one can show that $D^{\mathbf{1}-\mathbf{X}}(t) = D_{A^{\mathbf{1}-\mathbf{X}}}(t)$, which completes the proof. □

Using the equivalence of systems (a) and (b) from Figure 5.1, we will next compute statistical end-to-end delay bounds in queueing systems subject to the demultiplexing operation. The basic idea is to use the representation from system (b) in order to construct the *leftover dynamic servers* for the demultiplexed processes. Using these service processes, the desired performance bounds follow by applying conventional techniques from stochastic network calculus. This provides us a new viewpoint to renew our understanding about the effect of the scaling element models. It seems like a *deconstruction* of the scaling element, because we dismiss it at the place where its modeling objects, the flow transformation operations, really happen. We move these effects to another time phase of the flow, for example in this case, to the time phase before being served.

5.2 Single Node Deconstruction: Main Idea

In this and the following sections, we compute end-to-end delay bounds in a network with flow demultiplexing. For illustrative purposes, we focus on the single and two nodes cases, and later comment on the generalization to an arbitrary number of nodes. For numerical comparisons, we also reproduce an existing parallel result from Chapter 4.

Before we continue, let us recall the leftover dynamic server. Adjusted using Lemma 7 for the demultiplexing, the leftover dynamic server for $A^{\mathbf{X}}$, obtained in the particular case of FIFO scheduling (see Eq. (2.19) or [54, 95]) , is

$$S_{LO}(s,t) = \left[S(s,t) - A^{\mathbf{1}-\mathbf{X}}(s,t-x)\right]^+ 1_{\{t-x>s\}} \ ,$$

where $x \geq 0$ can be regarded as an optimization parameter (the meaning can be found in Eq. (2.26)). Referring to Figure 5.1.(a), we are interested in the virtual delay of the (sub-)flow of interest $D^{\mathbf{X}}$. The corresponding arrival process is $A^{\mathbf{X}}$ (see the interpretation from system (b)), and the leftover service process is $S_{LO}(s,t)$ shown above. Then, a probabilistic delay bound can be obtained as below

$$Pr(W(t) \geq d)$$

$$
\begin{aligned}
&= Pr\left(A^{\mathbf{X}}(t-d) \geq D_{A^{\mathbf{X}}}(t)\right) \\
&= Pr\left(A^{\mathbf{X}}(t-d) \geq D^{\mathbf{X}}(t)\right) \\
&\leq Pr\Bigg(\sup_{0\leq s<t-d} \Big\{ A^{\mathbf{X}}(t-d) - A^{\mathbf{X}}(s) - \\
&\qquad \left[S(s,t) - A^{\mathbf{1}-\mathbf{X}}(s,t-x)\right]^{+} 1_{\{t-x>s\}} \Big\} \geq 0 \Bigg) ,
\end{aligned}
$$

where in the third line we used Lemma 7, in the fourth line we used the definition of a service process and moved the infimum to the left side of the inequality. The derivation can be continued depending on two cases for x:

(i) $x > d$, i.e., $t - x < t - d$:

$$
\begin{aligned}
&= Pr\Bigg(\max\Bigg(\sup_{0\leq s<t-x} \Big\{ A^{\mathbf{X}}(s,t-d) - \\
&\qquad\qquad \left[S(s,t) - A^{\mathbf{1}-\mathbf{X}}(s,t-x)\right]^{+} \Big\}, \\
&\qquad \sup_{t-x\leq s<t-d} \left\{ A^{\mathbf{X}}(s,t-d) \right\} \Bigg) \geq 0 \Bigg) \\
&= Pr\Bigg(\max\Bigg(\sup_{0\leq s<t-x} \Big\{ A^{\mathbf{X}}(s,t-d) - \\
&\qquad\qquad \left[S(s,t) - A^{\mathbf{1}-\mathbf{X}}(s,t-x)\right]^{+} \Big\}, \\
&\qquad A^{\mathbf{X}}(t-x,t-d) \Bigg) \geq 0 \Bigg) \\
&= 1 ,
\end{aligned}
$$

(ii) $0 \leq x \leq d$, i.e., $t - x \geq t - d$:

$$
\begin{aligned}
&= Pr\Bigg(\sup_{0\leq s<t-d} \Big\{ A^{\mathbf{X}}(s,t-d) - \\
&\qquad\qquad \left[S(s,t) - A^{\mathbf{1}-\mathbf{X}}(s,t-x)\right]^{+} \Big\} \geq 0 \Bigg) \\
&\leq Pr\Bigg(\sup_{0\leq s<t-d} \Big\{ A(s,t-d) - \\
&\qquad\qquad S(s,t) + A^{\mathbf{1}-\mathbf{X}}(t-d,t-x) \Big\} \geq 0 \Bigg) .
\end{aligned}
$$

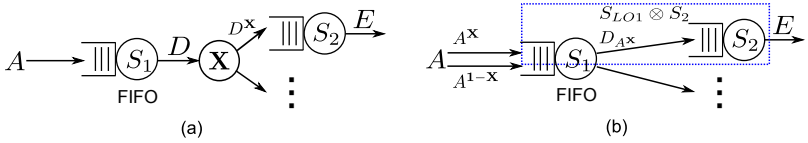


Figure 5.2: A network with two nodes, S_1 and S_2, and a demultiplexer element in between.

The optimal value is for $x = d$, and the last probability becomes

$$= \quad Pr\left(\sup_{0\leq s<t-d}\{A(s,t-d) - S(s,t)\} \geq 0\right).$$

Interestingly, the same bound can be obtained for the delay of the aggregate departure flow D. In other words, the per-flow delay bound is equal to the aggregate delay bound; the equality between the per-flow and aggregate delays is known to hold, on average, in the case of Poisson arrivals and Bernoulli demultiplexing, as a consequence of the PASTA property [111].

5.3 Two Nodes Deconstruction and Delay Bounds

In this section, we derive statistical end-to-end delay bounds in a network consisting of two service nodes and one scaling element implementing a demultiplexing operation in between (see Figure 5.2). We shall use the idea presented above, of first computing the leftover service curve for one demultiplexed flow of interest. Consider the network scenario of interest from Figure 5.2(a), of which the first node and the demultiplexing element can be equivalently transformed as detailed in Lemma 7. The result is shown in Figure 5.2(b). The next theorem provides the end-to-end delay bounds under different assumptions regarding the increments of the arrival process $A(t)$.

Theorem 10. *(Statistical End-to-End Delay Bounds in a Two Nodes Network with Demultiplexing) Consider the network scenario from Figure 5.2(a). A stationary arrival process $A(t)$ crosses two FIFO nodes offering the stationary service processes $S_1(s,t)$ and $S_2(s,t)$. Assume the MGF bounds of arrivals and services: $M_{A(s,t)}(\theta) \leq e^{\theta r(\theta)(t-s)}$ and $M_{S_k(s,t)}(-\theta) \leq e^{-\theta C_k(t-s)}$, where $k = 1, 2$ and for some $\theta > 0$. The demultiplexing process $\mathbf{X}$ is such that X_i's are either 0 or 1, and denote $\delta(\theta) = \frac{1}{\theta}\log M_{\mathbf{X}}(\theta)$ and $\bar{\delta}(\theta) = \frac{1}{\theta}\log M_{\mathbf{1-X}}(\theta)$. All the processes are assumed to be statistically*

independent, and we focus on the end-to-end subflow with departure process $E(t)$.

(1) If $A(t)$ *has statistically independent increments then under some stability conditions, to be explicitly given in the proof, we have the following delay bounds for all* $d \geq 0$

$$Pr(W(t) \geq d) \leq K_1 e^{-\theta C_2(d-x)} + K_2 e^{-\theta C_1 d} e^{\theta \bar{\delta}(\theta) r(\theta \bar{\delta}(\theta))(d-x)} + K_3 e^{-\theta C_1 d} e^{\theta(C_1 - C_2)(d-x)} ,$$

where K_1, K_2 *and* K_3 *are constants to be given in the proof.* x *is an optimization parameter introduced by FIFO leftover service.*

(2) If the increments of $A(t)$ *are not necessarily independent then under some stability conditions, to be explicitly given in the proof, we have the following delay bounds for all* $d \geq 0$

$$Pr(W(t) \geq d) \leq K e^{-\theta C_2 d} ,$$

where the constant K *will be given in the proof.*

Proof. As discussed in Section 5.1, we can equivalently transform the system (a) from Figure 5.2 into the system (b) from the same figure. Then we have for the end-to-end subflow of interest with input $A^{\mathbf{X}}(t)$ and output $E(t)$:

$$\begin{aligned}
&Pr(W(t) \geq d) \\
&= Pr\left(A^{\mathbf{X}}(t-d) \geq E(t)\right) \\
&\leq Pr\left(\sup_{0 \leq s < t-d} \left\{A^{\mathbf{X}}(t-d) - A^{\mathbf{X}}(s) - S_{LO} \otimes S_2(s,t)\right\} \geq 0\right) \\
&\leq Pr\left(\sup_{0 \leq s < t-d} \sup_{s \leq u \leq t} \left\{A^{\mathbf{X}}(t-d) - A^{\mathbf{X}}(s) - S_{LO}(s,u) - S_2(u,t)\right\} \geq 0\right) \\
&= Pr\Bigg(\sup_{0 \leq s < t-d} \sup_{s \leq u \leq t} \Big\{A^{\mathbf{X}}(t-d) - A^{\mathbf{X}}(s) - \big[S_1(s,u) - \\
&\qquad\qquad A^{\mathbf{1}-\mathbf{X}}(s,u-x)\big]^+ 1_{\{u-x>s\}} - S_2(u,t)\Big\} \geq 0\Bigg) . \qquad (5.2)
\end{aligned}$$

In the second line we used the definition of the virtual delay process for the arrival process $A^{\mathbf{X}}(t)$ and the departure process $E(t)$, and also the equivalence $W(t) \geq d \Leftrightarrow A^{\mathbf{X}}(t-d) \geq E(t)$. In the third line we used the definition and the convolution property of service processes and then expanded the convolution and the expression of $S_{LO}(s,u)$ in the rest lines.

The rest of the proof follows the derivations from Section 5.2, depending on the value of the parameter x:

(i) $x > d$, or, $u - x < t - d$. Let us consider only a part of the domain for the supremum as below

$$\begin{aligned}
&\sup_{t-x\le s<t-d}\sup_{s\le u\le t}\big\{A^{\mathbf{X}}(t-d)-A^{\mathbf{X}}(s)-\\
&\quad\big[S_1(s,u)-A^{\mathbf{1-X}}(s,u-x)\big]^+1_{\{u-x>s\}}-S_2(u,t)\big\}\\
=&\sup_{t-x\le s<t-d}\sup_{s\le u\le t}\big\{A^{\mathbf{X}}(t-d)-A^{\mathbf{X}}(s)-S_2(u,t)\big\}\\
\ge&0\,,
\end{aligned}$$

and thus the optimal value of x does not fall in this interval.

(ii) $0 \le x \le d$, and we continue Eq. (5.2) as follows:

$$\begin{aligned}
=&Pr\bigg(\sup_{0\le s<t-d}\bigg\{max\bigg(\sup_{s\le u\le s+x}\Big\{A^{\mathbf{X}}(s,t-d)-\\
&\big[S_1(s,u)-A^{\mathbf{1-X}}(s,u-x)\big]^+1_{\{u-x>s\}}-S_2(u,t)\Big\},\\
&\qquad\sup_{s+x<u\le t}\Big\{A^{\mathbf{X}}(s,t-d)-\big[S_1(s,u)-\\
&\qquad A^{\mathbf{1-X}}(s,u-x)\big]^+1_{\{u-x>s\}}-S_2(u,t)\Big\}\bigg)\bigg\}\ge 0\bigg)\\
\le&\sum_{0\le s<t-d}Pr\left(A^{\mathbf{X}}(s,t-d)-S_2(s+x,t)\ge 0\right)+\\
&\sum_{0\le s<t-d}\sum_{s+x<u\le t}Pr\Big(A^{\mathbf{X}}(s,t-d)-\\
&\qquad\big[S_1(s,u)-A^{\mathbf{1-X}}(s,u-x)\big]^+-S_2(u,t)\ge 0\Big)\\
\le&\sum_{0\le s<t-d}Pr\left(A^{\mathbf{X}}(s,t-d)-S_2(s+x,t)\ge 0\right)+\\
&\sum_{0\le s<t-d}\sum_{s+x<u\le t}Pr\Big(A^{\mathbf{X}}(s,t-d)+A^{\mathbf{1-X}}(s,u-x)-\\
&\qquad S_1(s,u)-S_2(u,t)\ge 0\Big)\,. \qquad (5.3)
\end{aligned}$$

The first line is just a separation of the interval of u in order to eliminate the indicator function "1". Besides that we applied the union bound in the second line. In the third line, we eliminated "$[\cdot]^+$". Next we use the properties of the

arrival process $A(t)$ in order to prove (1) and (2), respectively.

(1) $A(t)$ *has independent increments.* Since the two intervals $[s, t-d]$ and $[s, u-x]$ are overlapping, we can merge the overlapping parts of $A^{\mathbf{X}}(s, t-d)$ and $A^{\mathbf{1}-\mathbf{X}}(s, u-x)$. Yet we do not know which value, $t-d$ or $u-x$, is larger. So from Eq. (5.3) we separate the interval of u as below

$$
\begin{aligned}
&\leq \sum_{0\leq s<t-d} Pr\left(A^{\mathbf{X}}(s,t-d) - S_2(s+x,t) \geq 0\right) + \\
&\quad \sum_{0\leq s<t-d} \sum_{s+x<u\leq t-d+x} Pr\Big(A^{\mathbf{X}}(u-x,t-d) + \\
&\qquad\qquad A(s,u-x) - S_1(s,u) - S_2(u,t) \geq 0\Big) + \\
&\quad \sum_{0\leq s<t-d} \sum_{t-d+x<u\leq t} Pr\Big(A^{\mathbf{1}-\mathbf{X}}(t-d,u-x) + \\
&\qquad\qquad A(s,t-d) - S_1(s,u) - S_2(u,t) \geq 0\Big) \\
&\leq \sum_{0\leq s<t-d} e^{-\theta C_2(t-s-x)} e^{\theta\delta(\theta) r(\theta\delta(\theta))(t-d-s)} + \sum_{0\leq s<t-d} \sum_{s+x<u\leq t-d+x} \\
&\qquad e^{-\theta C_1(u-s)} e^{-\theta C_2(t-u)} e^{\theta r(\theta)(u-x-s)} e^{\theta\delta(\theta) r(\theta\delta(\theta))(t-d-u+x)} \\
&\quad + \sum_{0\leq s<t-d} \sum_{t-d+x<u\leq t} e^{-\theta C_1(u-s)} e^{-\theta C_2(t-u)} e^{\theta r(\theta)(t-d-s)} \,. \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad e^{\theta\bar{\delta}(\theta) r\left(\theta\bar{\delta}(\theta)\right)(u-x-t+d)} \\
&\leq \frac{e^{-\theta C_2(d-x)}}{e^{\theta(C_2-\delta(\theta)r(\theta\delta(\theta)))}-1} + \frac{\frac{1}{e^{\theta(C_1-r(\theta))}-1} - \frac{1}{e^{\theta(C_2-\delta(\theta)r(\theta\delta(\theta)))}-1}}{1-e^{\theta(C_1-C_2-r(\theta)+\delta(\theta)r(\theta\delta(\theta)))}} e^{-\theta C_1 d} \,. \\
&\quad e^{\theta(C_1-C_2)(d-x)} + \frac{e^{-\theta C_1 d}\left(e^{\theta\bar{\delta}(\theta) r\left(\theta\bar{\delta}(\theta)\right)(d-x)} - e^{\theta(C_1-C_2)(d-x)}\right)}{\left(1-e^{\theta\left(C_1-C_2-\bar{\delta}(\theta)r\left(\theta\bar{\delta}(\theta)\right)\right)}\right)(\theta C_1 - \theta r(\theta))} \,. \quad (5.4)
\end{aligned}
$$

In the second line we used the Chernoff bound for some $\theta > 0$ together with Lemma 5 for *i.i.d.* $X_i's$ (Eq. 4.8). Because A has independent increments and is independent of $\mathbf{X}$, it follows that $A^{\mathbf{X}}(u-x, t-d)$ is independent of $A(s, u-x)$ and $A^{\mathbf{1}-\mathbf{X}}(t-d, u-x)$ is independent of $A(s, t-d)$. In the case when $\mathbf{X}$ is MMSP, we have that $\delta(\theta) = \frac{1}{\theta}\log \mathrm{sp}\left(\phi_{\mathbf{X}}(\theta)\lambda_{\mathbf{X}}\right)$ and $\bar{\delta}(\theta) = \log \mathrm{sp}\left(\phi_{\mathbf{1}-\mathbf{X}}(\theta)\lambda_{\mathbf{1}-\mathbf{X}}\right)$, according to Eq. (4.7). In the third line, imposing the following two stability conditions

$$\theta(C_1 - r(\theta)) > 0 \text{ and } \theta(C_2 - \delta(\theta) r(\theta\delta(\theta)) > 0$$

for the first term by letting $t \to \infty$, we get an infinite geometric series; for the second term we compute the sum of geometric series over u and get the exact sum of infinite geometric series by letting $t \to \infty$; the computation for the third term is similar. By letting

$$\begin{aligned} K_1 &= \frac{1}{e^{\theta(C_2-\delta(\theta)r(\theta\delta(\theta)))}-1} \\ K_2 &= \frac{1}{\left(e^{\theta(C_1-r(\theta))}-1\right)\left(1-e^{\theta\left(C_1-C_2-\bar{\delta}(\theta)r\left(\theta\bar{\delta}(\theta)\right)\right)}\right)} \\ K_3 &= K_1 \cdot K_2 \cdot \left(1-e^{\theta\left(C_1-\delta(\theta)r(\theta\delta(\theta))-\bar{\delta}(\theta)r\left(\theta\bar{\delta}(\theta)\right)\right)}\right) \end{aligned}$$

in Eq. (5.4), we get

$$\begin{aligned} Pr(W(t) \geq d) \leq\ & K_1 e^{-\theta C_2(d-x)} + \\ & K_2 e^{-\theta C_1 d} e^{\theta\bar{\delta}(\theta)r(\theta\bar{\delta}(\theta))(d-x)} + K_3 e^{-\theta C_1 d} e^{\theta(C_1-C_2)(d-x)}, \end{aligned}$$

and the proof of (1) is complete.

(2) The increments of $A(t)$ *are not necessarily independent*. We can continue Eq. (5.3) as follows

$$\begin{aligned} \leq\ & \sum_{0\leq s<t-d} e^{-\theta C_2(t-s-x)} e^{\theta\delta(\theta)r(\theta\delta(\theta))(t-d-s)} + \\ & \sum_{0\leq s<t-d} \sum_{s+x<u\leq t} e^{-\theta C_1(u-s)} e^{-\theta C_2(t-u)} \cdot E\left[e^{\theta\left(A^{\mathbf{X}}(s,t-d)+A^{\mathbf{1}-\mathbf{X}}(s,u-x)\right)}\right], \end{aligned}$$

after using the Chernoff bound. To compute the bound of the expectation we use Hölder's inequality because of the possible dependence between $A^{\mathbf{X}}$ and $A^{\mathbf{1}-\mathbf{X}}$. Let g, $h \geq 1$ such that $\frac{1}{g}+\frac{1}{h}=1$. For $t \to \infty$, we continue the above inequality with

$$\begin{aligned} \leq\ & \frac{e^{-\theta C_2(d-x)}}{\theta(C_2-\delta(\theta)r(\theta\delta(\theta)))} + \\ & \frac{e^{-\theta C_1 d} e^{\theta(C_1-C_2)(d-x)}}{\theta(C_1-C_2-\bar{\delta}(h\theta)r(h\theta\bar{\delta}(h\theta))) \cdot \theta(C_2-\delta(g\theta)r(g\theta\delta(g\theta)))}. \end{aligned}$$

Here, without losing tightness, we used an upper bound on the infinite geometric series for simplicity. Let us now assume the stability conditions

$$\theta\left(C_2-\delta(\theta)r\left(\theta\delta(\theta)\right)\right) > 0$$

$$\theta\left(C_2 - \delta(g\theta)r\left(g\theta\delta(g\theta)\right)\right) > 0$$
$$\theta\left(C_1 - C_2 - \bar{\delta}(h\theta)r\left(h\theta\bar{\delta}(h\theta)\right)\right) > 0\,.$$

To continue the previous derivation, we use the next convex optimization result

$$\inf_{x>0}\left\{\alpha e^{-\beta x} + e^{\gamma x}\right\} = \left(\frac{\alpha\beta}{\gamma}\right)^{\frac{\gamma}{\beta+\gamma}} \frac{\beta+\gamma}{\beta}\,.$$

We finally obtain that

$$Pr(W(t) \geq d) \leq Ke^{-\theta C_2 d}\,,$$

where

$$\begin{aligned} K &= \frac{C_1}{C_2}\left(\frac{C_1}{(C_1-C_2)\theta(C_2-\delta(\theta)r(\theta\delta(\theta))}\right)^{1-\frac{C_2}{C_1}} \\ &\quad \cdot\left(\theta(C_1 - C_2 - \bar{\delta}(h\theta)r(h\theta\bar{\delta}(h\theta))) \cdot \theta(C_2 - \delta(g\theta)r(g\theta\delta(g\theta))\right)^{-\frac{C_2}{C_1}}\,. \end{aligned}$$

The proof is now complete. □

5.4 Delay Bounds Comparison

We compare the analytical results of Theorem 10 with Theorem 9 quantitatively. Before we can do so, we need to provide some results regarding MGF bounds, because for both methods, we assume to have the MGF bounds for the arrivals and the service, and also need to use the MGF bounds of the scaling processes and the scaled processes to compute the delay bounds.

5.4.1 MGF Bounds of the Arrivals and the Scalings

In this section, we present necessary prerequisites regarding MGF bounds for three examples of arrival processes $A(t)$: Bernoulli(-modulated) arrivals with rate R together with Bernoulli parameter p_0, Poisson arrivals with rate λ and MMOO arrivals with peak rate P. We also present results for the MGF bounds of scaled processes with two examples of scaling processes $\mathbf{X}$: Bernoulli scaling and MMOO scaling.

For Bernoulli arrivals with rate R, we consider arrivals with constant rate R passing through a Bernoulli scaling process with Bernoulli parameter p_0. We know that the MGF of a Bernoulli r.v. X_B with parameter $p_B \in [0,1]$

is $M_{X_B}(\theta) = 1 - p_B + p_B e^{\theta}$. So Lemma 5 yields the MGF bound for the *i.i.d.* Bernoulli arrivals with rate R and probability $p_0 \in [0,1]$ as

$$\begin{aligned} M_{A(s,t)}(\theta) &= M_{R(t-s)}\left(\log\left(1-p_0+p_0e^{\theta}\right)\right) \\ &= E\left[e^{\log\left(1-p_0+p_0e^{\theta}\right)R(t-s)}\right] \\ &= \left(1-p_0+p_0e^{\theta}\right)^{R(t-s)}. \end{aligned}$$

We rewrite this as

$$M_{A(s,t)}(\theta) = e^{\theta\cdot\frac{1}{\theta}\log\left(1-p_0+p_0e^{\theta}\right)R\cdot(t-s)} = e^{\theta\cdot r(\theta)\cdot(t-s)},$$

where we denote $\frac{1}{\theta}\log\left(1-p_0+p_0e^{\theta}\right)R$ as $r(\theta)$ according to the form of MGF bound for the arrivals in Theorem 10. Next, we derive $\delta(\theta)$ for the scaling process $\mathbf{X}$ as a Bernoulli process with parameter $p \in [0,1]$ and $\bar{\delta}(\theta)$ for its complementary scaling process $\mathbf{1}-\mathbf{X}$. Again, knowing the MGF of the *i.i.d.* Bernoulli process $\mathbf{X}$ we have

$$\begin{aligned} \delta(\theta) &= \frac{1}{\theta}\log M_X(\theta) = \frac{1}{\theta}\log\left(1-p+pe^{\theta}\right), \\ \bar{\delta}(\theta) &= \frac{1}{\theta}\log M_{1-X}(\theta) = \frac{1}{\theta}\log\left(p+(1-p)e^{\theta}\right). \end{aligned}$$

Now we have $r(\theta), \delta(\theta)$ as well as $\bar{\delta}(\theta)$. As all other parameters in Theorem 10 are already given, we can compute the delay bounds for the two nodes network. On the other hand, noting that a_1 is $\theta\delta(\theta)$, we can use Theorem 9 to compute alternative delay bounds.

In the following, we consider the following scenario: Poisson process as arrivals and Markov-Modulated On-Off (MMOO) process as scaling. The MMOO scaling processes are presented in Figure 5.3. μ_1 and μ_2 are tran-

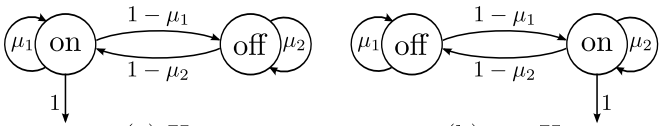


Figure 5.3: Complementary MMOO scaling processes.

sition probabilities. The processes let data pass along this subflow while in state 'on' (thus the rate is 1) and block while in state 'off'. If the scaling

processes are MMOO, we derive $\delta(\theta)$ and $\bar{\delta}(\theta)$ from Theorem 10 as

$$\begin{aligned}\delta(\theta) &= \frac{1}{\theta}\log \text{sp}\left(\phi_{\mathbf{X}}(\theta)\lambda_{\mathbf{X}}\right)\\ &= \frac{1}{\theta}\log\frac{\mu_2+\mu_1 e^{\theta}+\sqrt{\left(\mu_2+\mu_1 e^{\theta}\right)^2-4(\mu_1+\mu_2-1)e^{\theta}}}{2}\,,\\ \bar{\delta}(\theta) &= \frac{1}{\theta}\log \text{sp}\left(\phi_{\mathbf{1}-\mathbf{X}}(\theta)\lambda_{\mathbf{1}-\mathbf{X}}\right)\\ &= \frac{1}{\theta}\log\frac{\mu_1+\mu_2 e^{\theta}+\sqrt{\left(\mu_1+\mu_2 e^{\theta}\right)^2-4(\mu_2+\mu_1-1)e^{\theta}}}{2}\,,\end{aligned}$$

where μ_1 and μ_2 have the following relation for Figure 5.3(a), given the average 'on' probability p for scaling process $\mathbf{X}$,

$$p = \frac{1-\mu_2}{1-\mu_1+1-\mu_2}\,.$$

Then, of course, the scaling process $\mathbf{1}-\mathbf{X}$ has the average 'on' probability $1-p$. Hence, knowing p, given μ_1 we can compute μ_2, and vice versa. Further knowing the MGF bound of the Poisson arrival process, which is

$$M_{A(t)}(\theta) = e^{\theta r(\theta)t},\ \text{where } r(\theta) = \frac{1}{\theta}\lambda(e^{\theta}-1)\,,$$

we can again use Theorem 10 to compute the delay bounds for the two nodes network.

Similarly, considering an MMOO process as arrival process, we use P as the peak rate instead of 1 and $\lambda_1,\ \lambda_2$ as transition probabilities instead of $\mu_1,\ \mu_2$. We compute the MGF bound of the arrivals as

$$E\left[e^{\theta A(t)}\right] \leq e^{\theta r(\theta)t}\,,$$

where $r(\theta) = \frac{1}{\theta}\log\frac{\lambda_1 e^{\theta P}+\lambda_2+\sqrt{(\lambda_1 e^{\theta P}+\lambda_2)^2-4(\lambda_1+\lambda_2-1)e^{\theta P}}}{2}$. For other combinations of different arrival and scaling cases, we have similar computations.

5.4.2 Delay Bounds: Numerical Examples

Next, we numerically compare the statistical delay bounds using both methods in the above mentioned three examples of arrivals: Bernoulli with probability p_0 and rate R, Poisson with rate λ, and MMOO with peak rate P and

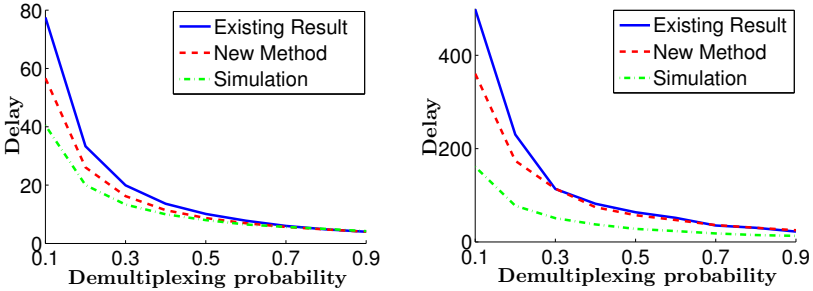
transition probabilities $\lambda_1,\ \lambda_2$. We consider the network senario with two nodes shown in Figure 5.2(a). For the scaling process we consider for two examples: Bernoulli process with probability p and MMOO process with average probability p and transition probabilities $\mu_1,\ \mu_2$. Here, we only show the results for MMOO arrivals with MMOO scaling, so we have five combinations and for all the combinations we compare the statistical delay bounds of the two nodes network using the dual analytical methods presented in Theorem 9 and 10. Moreover, for the five combinations we compare the delay bounds in different utilizations for the first node - low and high utilization (30% and 80%). As a reference, we also provide the results of discrete-event simulations. We use OMNeT++ [142] version 4.2 to simulate the queueing network. To compute the empirical 10^{-3}-quantiles, we observe 10^6 packets and use the P^2 algorithm [78] for calculating the quantiles without storing so many observations. Random number generating during the simulations is done using the class cLCG32.

In the following, we give the numerical settings. First, the service rate of the first node C_1 is normalized to one packet per time unit. Correspondingly, we set the utilization of the first node as either 0.3 or 0.8, i.e., the average rates of arrivals are set as 0.3 and 0.8, respectively. We let $R = 2$, thus in the case of Bernoulli arrivals p_0 equals to 0.15 and 0.4, respectively. For the case of MMOO arrivals, we also set $P = 2$ and $\lambda_1 = 0.75$. With the average passing probability p of the scaling process $\mathbf{X}$ the capacity at the second node C_2 is set to $C_2 = p \cdot C_1$. These settings guarantee the stability conditions. We vary p from 0.1 to 0.9 in steps of 0.1. In the case of MMOO scaling, we set μ_1 as 0.75. We plot the ε-quantiles (in time units) of the end-to-end delay bounds with $\varepsilon = 10^{-3}$. In all figures, we can perceive, that the delays decrease for higher values of the pass probabilities p. This behavior is due to setting $C_2 = p \cdot C_1$ for a constant C_1.

Figure 5.4 shows the statistical delay bounds obtained with Theorem 10 (new method) and Theorem 9 (existing result) as well as the simulation results in the case of Bernoulli arrivals and Bernoulli scaling. For demultiplexing probabilities $p \leq 0.7$ the figures clearly illustrate the order of the results, with the new leftover service computation method as a superior option especially for smaller pass probabilities p.

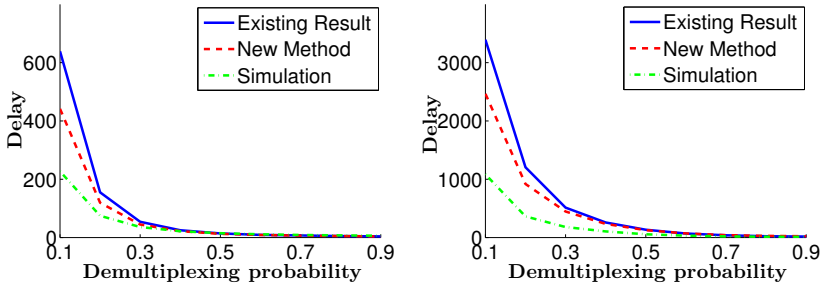
For $p > 0.7$, the results of Theorem 10 (with independent increments) and Theorem 9 are nearly the same. Going to the higher utilization of 0.8 in Figure 5.4.b of course results in higher delays, but also seems to loosen the bounds somewhat as the gap to the simulation results grows.

Figure 5.5 shows the statistical delay bounds in the case of Bernoulli arrivals and MMOO scaling. Basically the same observations as in the previous



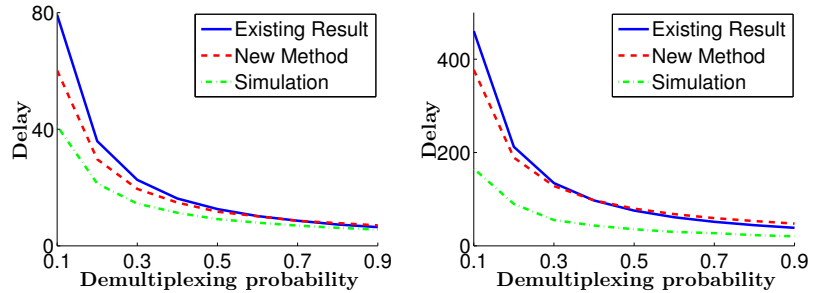
(a) Utilization of S_1=0.3 (b) Utilization of S_1=0.8

Figure 5.4: Bernoulli arrivals, Bernoulli scaling.



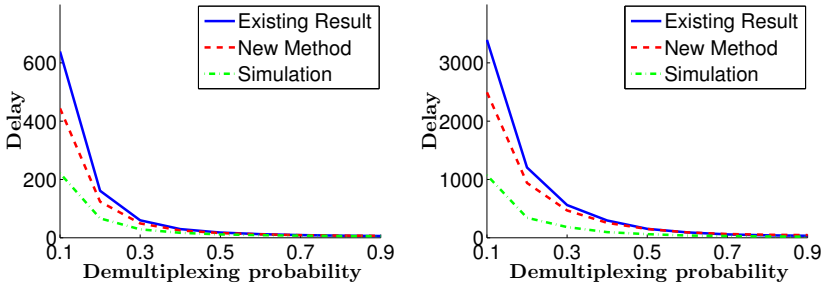
(a) Utilization of S_1=0.3 (b) Utilization of S_1=0.8

Figure 5.5: Bernoulli arrivals, MMOO scaling.



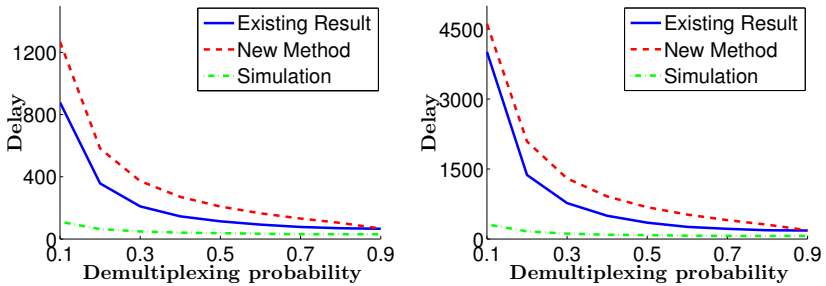
(a) Utilization of S_1=0.3 (b) Utilization of S_1=0.8

Figure 5.6: Poisson arrivals, Bernoulli scaling.



(a) Utilization of S_1=0.3 (b) Utilization of S_1=0.8

Figure 5.7: Poisson arrivals, MMOO scaling.



(a) Utilization of S_1=0.3 (b) Utilization of S_1=0.8

Figure 5.8: MMOO arrivals, MMOO scaling.

paragraph, for Bernoulli arrivals and Bernoulli scaling, can be made. It is interesting to note, however, how the MMOO scaling leads to much higher delay bounds compared to the Bernoulli scaling case.

Next, Figure 5.6-5.7 display the same as Figure 5.4-5.5, but now for Poisson arrivals. For most of it, the conclusions for the Poisson arrivals are the same as for the Bernoulli arrivals, yet one interesting observation is that for Poisson arrivals and Bernoulli scaling for the high utilization case, the leftover service compution results in slightly, but clearly visible worse delay bounds than the method from Theorem 9. This is not the case for the MMOO scaling which indicates that things are not so simple here ...

Figure 5.8 shows the statistical delay bounds in the case of MMOO arrivals and MMOO scaling. Note that in this scenario we do not have independent increments any more and and thus have to resort to the respective case in Theorem 10. We can see that the method by commuting service curve and scaling element now clearly dominates the leftover service method. As a general remark, we can observe that the bounds are quite pessimistic for low values of the pass probability p when compared to the simulation results. It is simply harder to cope with correlations in arrivals and scaling processes.

To summarize, the leftover amount of service should be generally more than that scaled part by using Lemma 6 in particular when the utilization less than 1, and hence, enables the derivation of tighter bounds. But the effort on extra processing the dependence causes some loss of the tightness when using Theorem 10, e.g., using Hölder inequality may cause this problem. However, the meaning of the leftover service model is clearer than the exact dynamic server model. It exactly reflect the system behaviour. The ambiguous treatment (Lemma 6 and Theorem 9), though lose some tightness, still has its own advantage. The obvious one is that it can apply to the data enlarging case of flow transformation. If we use leftover model, we have to introduce a virtual flow adhereing to the original one, which is not that flexible like the exact dynamic server model in Lemma 6.

5.5 N Nodes Deconstruction

In the previous section we compared the results obtained by the dual methods from Theorem 10 and Theorem 9 for some numerical examples. We can see that, on one hand, Theorem 10 provides the opportunity to utilize some additional information on arrivals to show an advantage, especially in scenarios where the subflow of interest is small. On the other hand, a disadvantage of using the leftover service curve method is that the delay bound computation

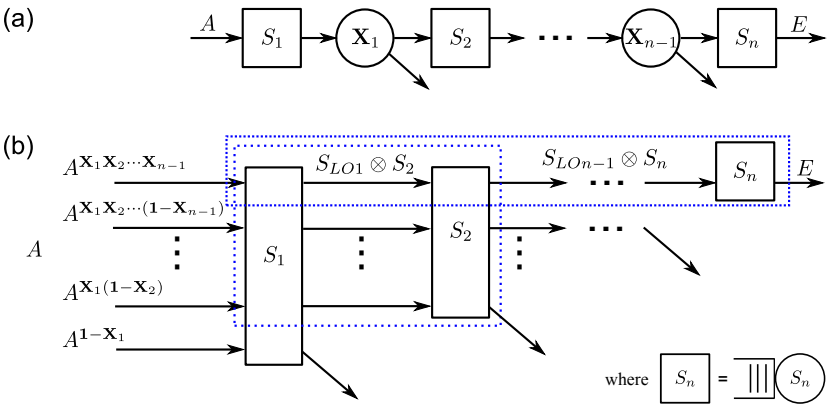


Figure 5.9: A flow demultiplexing network consisting of services and scaling elements.

is not easily extensible to the n nodes case. This is because we would have to introduce different "x" for the leftover service curve element at each node, which makes the analytical solution complicated. In contrast, Theorem 9 is more easily applicable to the n nodes case.

Albeit the complexity of the calculation, what we compare are in fact the methods behind these two theorems, one is commuting the scaling and the service elements, the other changes to the leftover dynamic server viewpoint (Lemma 6 and 7). It is obvious to see that the latter is the actual understanding of the scaling when we apply it for dealing with the demultiplexing. A demultiplexing operation is transformed into an inherently equivalent flow multiplexing, which means the use of the scaling element for the demultiplexer is trivial and should be dismissed. We view this phenomena as the deconstruction of the scaling element (it is intuitive by comparing Figure 5.1 and 4.7). This deconstruction also supports the n nodes modeling and analysis. In the following, we sketch the iterative computation steps to be taken, but keep the concrete computation as future work.

Consider a network scenario from Figure 5.9(a) which is the general extension of Figure 5.2(a). We can firstly apply all the scaling effects to A and after iteratively computing the leftover service curves we then obtain the transformed system in Figure 5.9(b). The iteration can be stated as follows

$$S_{LO1}(s,t) = \left[S_1(s,t) - A^{\mathbf{1}-\mathbf{X}}(s,t-x_1)\right]^+ 1_{\{t-x_1>s\}}$$

$$S_{LOn-1}(s,t) = \Big[(S_{LOn-2} \otimes S_{n-1})(s,t) - A^{\mathbf{X}_1\mathbf{X}_2\cdots(\mathbf{1}-\mathbf{X}_{n-1})}(s,t-x_{n-1})\Big]^+ 1_{\{t-x_{n-1}>s\}} .$$

As a consequence we face with a single node model

$$\xrightarrow{A^{\mathbf{X}_1\mathbf{X}_2\cdots(\mathbf{1}-\mathbf{X}_{n-1})}} S_{LOn-1} \otimes S_n \xrightarrow{E}$$

Besides demultiplexing, the deconstruction idea also applies conversely for the case where the scaling element models the flow enlarging or aggregating. The difference from the demultiplexing case shown in Figure 5.9 is that the flow we observe changes from the destination end to the source end. Consider the n nodes with $n-1$ scalers case, at each scaler the flow is enlarged. If we denote the departure at node i as D_i, then we face with the single node network for the performance analysis

$$\xrightarrow{A} S_{LOn} \xrightarrow{E}$$

where S_{LOn}=$S_1 \otimes \left(\left[S_2 \otimes \left(\cdots \left(\left[S_n - D_{n-1}^{\mathbf{X}_{n-1}-\mathbf{1}} \right]^+ \right) \cdots \right) - D_1^{\mathbf{X}_1-\mathbf{1}} \right]^+ \right)$ for $n \geq 2$. Note, because we can not guarantee a FIFO scheduling for the original flow and the enlarged part, we use the leftover dynamic server defined for the blind multiplexing (see Eq. (2.18) in Section 2.3.3). Since the calculation of the above cases involve many new parameters, introduced by the FIFO leftover dynamic servers and Hölder inequality, we do not present the calculation in this thesis. We may also need to find a smarter method to represent the problem. We leave it as a good clue for future work.

Chapter 6

Scaling Element for Unreliable Links with Retransmissions

A very tough challenge for network calculus is found in applying it in those network scenarios, where links (or servers) may be unreliable and some packets are lost. This actually shatters network calculus in one of its foundations which is the assumption of a lossless system operation. Yet, the key of our approach in dealing with such losses lies in the stochastic data scaling element, and the simple realization that loss is just a specific flow transformation. To make things even more compounded, unreliable links usually employ retransmission-based loss recovery schemes, which produces a *feedback* cycle between in- and output – the main difficult issue for the analysis in this chapter.

There is a set of previous research of network calculus that deal with the lossy systems, but only a little really tried to solve the retransmission problem (see the summary in Section 2.4) and the methods therein used are very inflexible. In this chapter we improve the modeling flexibility using the stochastic (sample path) scaling element and provide a two-step procedure to derive the stochastic performance bounds under the influence of loss and retransmissions-based compensation. We specifically make the following contributions:

- We set up a stochastic network calculus model for an unreliable link employing retransmissions for loss recovery.

- For a specific loss process, the binary symmetric channel, we determine a tight stochastic scaling curve based on a Martingale argument.
- We solve the feedback cycle problem under the realistic assumption of a limited number of retransmissions.

These results are from the joint work with J. Schmitt and F. Ciucu [150, 146].

Next we first present the model for an unreliable link with retransmission-based loss recovery, using our stochastic data scaling model (Section 6.1). Then we analyze it and provide the end-to-end performance bounds by using an advanced construction of scaling curve and accounting for the feedback loop for sample paths (Section 6.2). After that we numerically validate the analytical results (Section 6.3).

6.1 A Model of an Unreliable Link with Retransmissions

In this section, we propose a model for an unreliable link that employs a retransmission scheme to recover from data loss due to channel impairments. The model builds upon the novel concept of stochastic scaling as introduced in the previous chapters and the existing results of network calculus.

Data loss is a frequent event when transmitting data over an unreliable link. In order to compensate for data loss, many protocols and methods employ retransmission schemes. The data loss process can be captured in a network calculus model by a scaling process $\mathcal{S}$ and its stochastic scaling curve $\overline{S}^{\overline{\varepsilon}}$ as elaborated in Section 3.1. Before retransmitting a lost packet, we assume that the sender must wait to be certain that the packet has really been lost. Thus retransmitted packets will experience a certain delay. To cover typical retransmission schemes we model two mechanisms to detect such a loss event: a local countdown timer at the sender if no positive acknowledgment is received, and, optionally, an explicit negative acknowledgment from the receiver. Thus the delay experienced before a retransmission is performed is upper bounded by the countdown timer, but may be lower if a negative acknowledgment is received before timer expiration. We further assume the countdown timer to be set to a fixed value. Clearly, loss detection may not be perfect resulting in duplicate data packets. We assume that duplicate packets can be identified by the receiver (e.g., through using sequence number). In any case, due to the countdown timer or a negative acknowledgement received at the sender side, we can abstractly model the retransmission as a data flow being fed back to the sender.

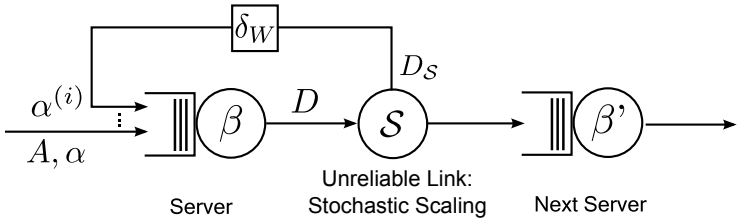


Figure 6.1: Network calculus model of an unreliable link with retransmissions.

Concretely, modeling the retransmissions of the lost data units consists of 1) a feedback loop of the lost data units to the entrance of the server, 2) a feedback delay each retransmitted packet potentially experiences, and 3) a deterministic strict service curve β for characterizing the service capacity available to the aggregate of original and retransmitted units. We point out that the deterministic nature of the service curve precludes accounting for the possibly time-varying nature of the unreliable channel's capacity; while such more realistic scenarios can be captured using a stochastic service curve (see, e.g., [62]), we consider the simplified deterministic model for ease of exposition. Or in other words, we can firstly lay our focus onto the sample path modeling and then apply the Boole's inequality argument like shown in Section 3.4. Moreover, to model the feedback delay, we know that the waiting time of either the countdown timer or the potential negative acknowledgement is bounded by a maximum feedback delay denoted as W. Thus, the delay process has a service curve δ_W (see Section 2.2.2), with $W \geq 0$ and $\delta_W(t) = 0$ if $t \leq W$, otherwise $\delta_W(t) = \infty$. Note that retransmitted data units may have to be retransmitted again due to new loss, resulting in further retransmitted flows. We make the assumption, which holds true in most practical implementations, that the number of retransmissions is limited to some fixed value. This model of an unreliable link with retransmission-based loss recovery is depicted in Figure 6.1.

We distinguish between different retransmission flows, those consisting of data units being retransmitted once, twice, and so on. Correspondingly, we represent all flows in the system with their arrival curves $\alpha^{(0)}, \alpha^{(1)}, \cdots, \alpha^{(N)}$, where $\alpha^{(0)} = \alpha$ and N is the limit of the number of retransmissions for any data unit. Denote by $\mathcal{S}_i$ the scaling process of flow $i-1$ and by $\overline{S_i^{\varepsilon_i}}$ the corresponding scaling envelope, where $i \geq 1$. Note that the $\mathcal{S}_i$'s form a partition of the overall scaling process $\mathcal{S}$ and $\sum_{i=1}^{N} \mathcal{S}_i = \mathcal{S}$. Here, $\overline{\varepsilon_i}$ is the violation probability of $\mathcal{S}_i$. We also make the assumption that retransmission flow i,

consisting of the data units retransmitted for i times, has lower priority than retransmission flow $i+1$, for all $i = 1, \ldots, N-1$. One instance of this policy is exemplified by a simple stop-and-wait protocol. More generally, any ARQ protocol that sends data units with lower sequence numbers first satisfies this assumption, for example, TCP does, too.

6.2 End-to-End Performance Bounds

In this section we first model the lossy channel, and derive the arrival curves of the original and retransmitted flows, then we provide the stochastic performance bounds.

6.2.1 Modeling a Binary Symmetric Channel (BSC)

To exemplify how the loss process at an unreliable link can be captured with stochastic scaling, we consider a binary symmetric channel (BSC) model; more complicated channel models can also be considered (see Section 3.4.1 and 4.1.1, or [49] for loss processes defined as Markov arrival processes). For the ease of presentation, and also to deal with inherent technical complications due to modeling retransmissions, we mainly focus on the BSC model.

We model the BSC with the scaling process

$$\mathcal{S}(b) = X_1 + X_2 + \cdots + X_b \,, \tag{6.1}$$

where $X_i's \in \{0, 1\}$ are *i.i.d.* Bernoulli random variables with parameter p, i.e., the crossover probability of the BSC. The next theorem gives the stochastic scaling curve for the BSC, which will be used later. We point out that the stochastic scaling curve specifically for BSC is a more advanced construction than the one we showed with Lemma 2 in Section 3.4.1.

Theorem 11. *(Stochastic Scaling Curve for BSC) Consider the scaling process $\mathcal{S}(b)$ for BSC from Eq. (6.1). Then the function*

$$\overline{S}^{\overline{\varepsilon}}(b) = pb + 1 - \overline{\varepsilon} \,. \tag{6.2}$$

is a stochastic scaling curve for BSC, in the sense of Definition 15.

Proof. Let us first construct the process

$$T(c) := \mathcal{S}(c) - \overline{S}^{\overline{\varepsilon}}(c) + 1 \,, \; \forall c \,.$$

Construct also the filtration $\mathcal{F}_c = \sigma(X_1, X_2, \ldots, X_c)$ capturing the partial histories of the process X_c. Then we have for some fixed c:

$$\begin{aligned} E\left[T(c+1) \mid \mathcal{F}_c\right] &= E\left[T(c) + X_{c+1} - p \mid \mathcal{F}_c\right] \\ &= T(c) + E\left[X_{c+1} - p\right] \\ &= T(c)\,, \end{aligned}$$

and thus $T(c)$ is a martingale. In the second line we used that $T(c)$ is $\mathcal{F}_c$-measurable, and that X_{c+1} is independent of $\mathcal{F}_c$. The last line follows since X_{c+1} is a Bernoulli r.v. with parameter p.

Since $T(c)$ is a martingale we can write for all $b \geq 0$:

$$\begin{aligned} & Pr\left(\sup_{0\leq a\leq b}\left\{\mathcal{S}(b) - \mathcal{S}(a) - \overline{S}^{\overline{\varepsilon}}(b-a)\right\} \geq 0\right) \\ &= Pr\left(\sup_{0\leq c\leq b} T(c) \geq 1\right) \\ &\leq E\left[T(b)\right] \\ &= \overline{\varepsilon}\,. \end{aligned}$$

In the second line we used the stationarity of the Bernoulli process. In the third line we used Doob's maximal inequality applied to the martingale $T(c)$, i.e., for a martingale $T(c)$ and each $x > 0$ we have $Pr\left(\sup_{0\leq c\leq b} T(c) \geq x\right) \leq \frac{E[T(b)]}{x}$ (see Billingsley [16], p. 466). □

This result will be used to compute the arrival curves of the retransmission flows.

6.2.2 Arrival Curves for Retransmission Flows

General Problem Statement

Under the general assumptions of Section 6.1, we can use existing network calculus results on priority multiplexing (see Eq. (2.9)) to obtain the following formulations for the service and arrival curves of each retransmission flow. Under the assumption that the node offers a strict service curve β, using the output bound of Theorem 1 in Section 2.3.3 and Corollary 5, each flow i is offered a service curve $\beta^{(i)}$ defined as follows

$$\beta^{(0)} = [\beta - \sum_{k=1}^{N} \alpha^{(k)}]^{+}\,, \quad \alpha^{(0)} = \alpha$$

$$\beta^{(1)} = [\beta - \sum_{k=2}^{N} \alpha^{(k)}]^+ , \quad \alpha^{(1)} = \overline{S_1^{\varepsilon_1}}(\alpha^{(0)} \oslash \beta^{(0)}) \oslash \delta_W$$
$$\beta^{(2)} = [\beta - \sum_{k=3}^{N} \alpha^{(k)}]^+ , \quad \alpha^{(2)} = \overline{S_2^{\varepsilon_2}}(\alpha^{(1)} \oslash \beta^{(1)}) \oslash \delta_W$$
$$\cdots \qquad\qquad \cdots$$
$$\beta^{(N)} = \beta , \qquad \alpha^{(N)} = \overline{S_N^{\varepsilon_N}}(\alpha^{(N-1)} \oslash \beta^{(N-1)}) \oslash \delta_W .$$

We point out that, with a small loss of generality, we ignored packetization effects in the expressions of $\beta^{(i)}$'s (see Section 2.2.3). We can further simplify and remove the dependency of the $\alpha^{(i)}$'s and the $\beta^{(i-1)}$'s as follows

$$\begin{aligned}
\alpha^{(0)} &= \alpha \\
\alpha^{(1)} &= \overline{S_1^{\varepsilon_1}} \left(\alpha^{(0)} \oslash [\beta - \alpha^{(1)} - \alpha^{(2)} - \cdots - \alpha^{(N)}]^+ \right) \oslash \delta_W \\
\alpha^{(2)} &= \overline{S_2^{\varepsilon_2}} \left(\alpha^{(1)} \oslash [\beta - \alpha^{(2)} - \alpha^{(3)} - \cdots - \alpha^{(N)}]^+ \right) \oslash \delta_W \\
&\cdots \\
\alpha^{(N)} &= \overline{S_N^{\varepsilon_N}} \left(\alpha^{(N-1)} \oslash [\beta - \alpha^{(N)}]^+ \right) \oslash \delta_W .
\end{aligned}$$

From these equations, it is apparent that the different arrival flows are dependent on each other and thus a probabilistic interpretation of the curves would need to take into account the corresponding correlations. However, since, in the first step, we argue purely deterministically this becomes no technical problem. The deterministic argument is under the assumption that we are on a sample path of the system for which the scaling curves are not violated. Only in the second step, in Section 6.2.3, when we evaluate the probability of this event, we reason stochastically, yet then the correlations between the arrival flows pose no technical problem any more.

Our goal next is to find explicit formulations of $\alpha^{(1)}$, $\alpha^{(2)}$, $\cdots$, $\alpha^{(N)}$ using this recursive set of equations (in the $\alpha^{(i)}$'s only, as the $\beta^{(i)}$'s were just removed). We use a fixed-point approach to resolve the self-dependency issue of the $\alpha^{(i)}$'s; see also Figure 6.2 for a graphical representation of the recursion system. Thereby we interpret our equation system as a mapping

$$T(\vec{\alpha}_n) = \vec{\alpha}_{n+1} ,$$

where $\vec{\alpha}_n = (\alpha_n^{(1)}, \alpha_n^{(2)}, \cdots, \alpha_n^{(N)})$. To find a fixed-point for system T we are going to rely on further assumptions on the shape of the functions α_i's, as elaborated next.

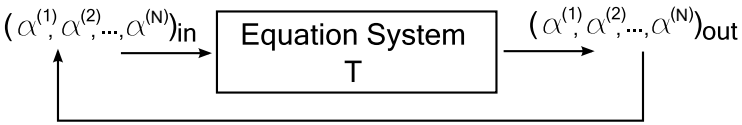


Figure 6.2: Self-dependent equation system.

Fixed-Point Calculation

Next we tackle the fixed-point problem just described by making further assumptions to instantiate a form of the problem that can actually be solved. The assumptions are simplifying but at the same time they are realistic and some of them are actually without loss of generality.

Further Assumptions

We assume an affine arrival curve (a token bucket) for the original input flow $\alpha = \gamma_{r,b}$ (i.e., $\alpha(t) = rt + b$), and a rate-latency function as the strict service curve $\beta = \beta_{R,T}$ (i.e., $\beta(t) = R[t-T]^+$). The feedback is assumed to be delayed, so we use a positive W for the service curve δ_W of the feedback delay process. With respect to the scaling curves, we set $\forall i = 1, \ldots, N$: $\overline{S_i^{\overline{\varepsilon_i}}}(b) = \overline{S^{\overline{\varepsilon}}}(b) = Cb + B$. This means all the scaling processes $\mathcal{S}_i$'s of the retransmission flows can be bounded by the same stochastic scaling curve; this holds true for any overall scaling process that is *i.i.d.* (e.g., the BSC). For ease of exposition, we also assume homogeneity of the scaling violation probabilities $\overline{\varepsilon_i}$'s. Note further that under the BSC, C and B can be calculated using Eq. (6.2), and should satisfy $B \geq 0$ and $0 \leq C < 1$. The latter condition is necessary for the convergence of T, since otherwise the scaling would not act as a contractor but as an expander such that α_n would diverge to infinity. We point out that the analysis of more complex arrival, service, and scaling curves would follow a similar line of argument, and is left for future work.

For illustrative purposes, we demonstrate the derivation of the arrival curves of retransmission flows for two cases: a single retransmission case and the general case of N retransmissions. We focus especially on the first case, as the second one can be treated as a generalization by using the same method.

(1) N = 1 ↔ one retransmission flow

We know that

$$\alpha^{(0)} = \alpha \,,$$

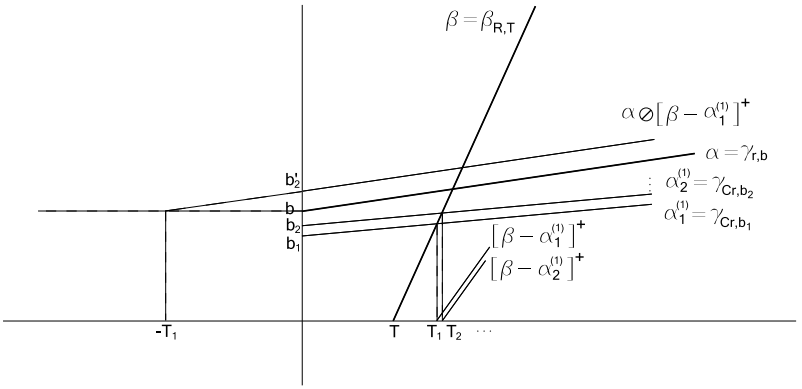


Figure 6.3: Illustration of the calculation for one retransmission flow.

$$\alpha^{(1)} = \overline{S^{\varepsilon}} \left(\alpha^{(0)} \oslash \beta^{(0)} \right) \oslash \delta_W = \overline{S^{\varepsilon}} \left(\alpha \oslash \left[\beta - \alpha^{(1)} \right]^+ \right) \oslash \delta_W \ .$$

It is to be checked whether the mapping for $\alpha^{(1)}$ is convergent and what is the fixed point of the mapping $\alpha_\infty^{(1)}$. Put differently, if we set the initial input of $\alpha^{(1)}$ as $\alpha_1^{(1)} = \gamma_{Cr,b_1}$, where b_1 is the variable of burst needed to solve the fix-point equations, the task is to check whether there is a convergent limit b_∞ for b_1 and what is its value. Note that the rate of $\alpha^{(1)}$ is Cr because any other rate of $\alpha^{(1)}$ will be limited to the rate of α (i.e., r) after the deconvolution $\alpha \oslash [\beta - \alpha^{(1)}]^+$, and also after the invocation of the scaling curve $\overline{S^{\varepsilon}}(b) = Cb + B$.

Next, we calculate the formulation $\alpha^{(1)} = \overline{S^{\varepsilon}} \left(\alpha \oslash [\beta - \alpha^{(1)}]^+ \right) \oslash \delta_W$ step by step until we achieve enough information to assess its convergence and are able to calculate the fixed-point value b_∞ of $b_1, b_2, \ldots$ This process is depicted in Figure 6.3 and explained in the following steps:

1. Calculate curve $\left[\beta - \alpha_1^{(1)} \right]^+ = \beta_{R-Cr,T_1}$, where T_1 is the latency to be determined. From $CrT_1 + b_1 = R(T_1 - T)$ we have

$$T_1 = \frac{RT + b_1}{R - Cr} \ .$$

2. Calculate $\alpha \oslash \left[\beta - \alpha_1^{(1)} \right]^+$. Draw a line at point $(-T_1, b)$ with rate equal to the rate of the arrival curve α, which is r. Now calculate the burst of the

above curve - b_2' as

$$b_2' = rT_1 + b \, .$$

3. At last, calculate $\alpha_2^{(1)} = \overline{\overline{S}^{\varepsilon}} \left(\alpha \oslash \left[\beta - \alpha_1^{(1)} \right]^+ \right) \oslash \delta_W$ as $\alpha_2^{(1)} = \gamma_{Cr,b_2}$ with

$$b_2 = Cb_2' + B + CrW = C(rT_1 + b) + B + CrW \, .$$

Repeating step 1, 2 and 3 for $\alpha_2^{(1)}$ we obtain

$$\begin{aligned} T_2 &= \frac{RT + b_2}{R - Cr} \, , \\ b_3 &= C(rT_2 + b) + B + CrW \, . \end{aligned}$$

Then we repeat this calculation again to obtain $b_4 = C(rT_3 + b) + B + CrW$ and so on. That means the convergence of $\alpha^{(1)}$ depends on the sequence of $T_1, T_2, T_3, \ldots$ We can write this sequence for integer $j > 1$ as follows:

$$\begin{aligned} T_1 &= \frac{RT + b_1}{R - Cr} \, , \cdots \\ T_j &= \frac{Cr}{R - Cr} T_{j-1} + \frac{RT + Cb + B + CrW}{R - Cr} \, . \end{aligned}$$

At the same time, we have

$$b_j = (R - Cr)T_j - RT \, .$$

For the deconvolution $\alpha \oslash \left[\beta - \alpha^{(1)} \right]^+$ to exist, the following condition must hold

$$\lim_{t \to \infty} \frac{\left[\beta - \alpha^{(1)} \right]^+ (t)}{t} \geq \lim_{t \to \infty} \frac{\alpha(t)}{t} \implies R - Cr \geq r \implies \frac{Cr}{R - Cr} \leq C < 1 \, .$$

This is in fact a stability condition for the system. In particular, $R > r + Cr$ means that the long-term capacity of the server can satisfy the long-term needs of the original and the retransmitted flow (recall also our previous assumption that $C < 1$ to guarantee stability). Applying the stability condition to

$$T_j = \frac{Cr}{R - Cr} T_{j-1} + \frac{RT + Cb + B + CrW}{R - Cr} \, ,$$

we obtain that the sequence of T_j is convergent, i.e., there is a fixed point T_∞

with

$$T_\infty \quad = \quad \frac{RT + Cb + B + CrW}{R - 2Cr} \, .$$

Finally, we can calculate the arrival curve of the retransmission flow as

$$\begin{aligned} \alpha^{(1)} &= \gamma_{Cr,b_\infty} \, , \text{ where} \\ b_\infty &= (R - Cr)T_\infty - RT \, . \end{aligned}$$

(2) General N ↔ N retransmission flows

Because the calculation process for $N \geq 2$ is very similar to $N = 1$ (the only difference is that now we face an equation system due to $\alpha^{(1)}, \ldots, \alpha^{(N)}$), we ignore the computational details and provide the results directly. First, we assume the following condition

$$R > \left(1 + C + C^2 + \cdots + C^N\right) r = \frac{1 - C^{N+1}}{1 - C} r \tag{6.3}$$

for the system's stability. This inequality can be used as a tool to adjust the server capacity, the input flow rate or maximum number of retransmissions. If we denote the convergent latency of $\left[\beta - \alpha^{(j)} - \alpha^{(j+1)} - \cdots - \alpha^{(N)}\right]^+$ with $T_{j,\infty}$, for all $1 \leq j \leq N$, then the fixed-point problem reduces to solving the following equation system

$$A \times (T_{1,\infty}, T_{2,\infty}, \ldots, T_{j,\infty}, \ldots, T_{N,\infty})^t = \phi \, , \text{where}$$

$$A = \begin{pmatrix} R - 2r\sum_{i=1}^{N} C^i & -r\sum_{i=2}^{N} C^i & \ldots & -r\sum_{i=N}^{N} C^i \\ -r\sum_{i=2}^{N} C^i & R - 2r\sum_{i=2}^{N} C^i & \ldots & -r\sum_{i=N}^{N} C^i \\ \vdots & \vdots & \ddots & \vdots \\ -r\sum_{i=N}^{N} C^i & -r\sum_{i=N}^{N} C^i & \ldots & R - 2r\sum_{i=N}^{N} C^i \end{pmatrix} ,$$

$$\phi = \begin{pmatrix} RT + b\sum_{i=1}^{N} C^i + B \cdot \sum_{p=0}^{N-1}\sum_{q=0}^{p} C^q + rW \cdot \sum_{i=1}^{N} iC^i \\ RT + b\sum_{i=2}^{N} C^i + B \cdot \sum_{p=1}^{N-1}\sum_{q=0}^{p} C^q + rW \cdot \sum_{i=2}^{N} iC^i \\ \vdots \\ RT + b\sum_{i=N}^{N} C^i + B \cdot \sum_{p=N-1}^{N-1}\sum_{q=0}^{p} C^q + rW \cdot \sum_{i=N}^{N} iC^i \end{pmatrix} .$$

We next use Cramer's Rule to derive

$$T_{1,\infty} = \frac{det(A_1)}{det(A)} \, , \, T_{2,\infty} = \frac{det(A_2)}{det(A)} \, , \, \cdots \, , \, T_{N,\infty} = \frac{det(A_N)}{det(A)} \, ,$$

where $A_{i,1\leq i\leq N}$ is the matrix A with the ith column of A replaced by ϕ. If all roots are positive, then a fixed point exists. In this case, the arrival curves of all N retransmission flows are given as follows

$$\begin{aligned} \alpha^{(j)} &= \gamma_{C^j r, b_{j,\infty}}, \text{ where} \\ b_{j,\infty} &= C^j r(T_{j,\infty} + T_{j-1,\infty} + \cdots + T_{1,\infty}) \\ &\quad + C^j b + (C^{j-1} + \cdots + C + 1)B + jC^j rW \, . \end{aligned}$$

6.2.3 Performance Bounds

In the previous subsection, we have shown a fixed-point approach to derive the arrival curves for each retransmission flow $\alpha^{(i)}$ and, consequently, also the service curves $\beta^{(i)}$ as seen by each of these flows. As discussed above, the arguments were given under a deterministic interpretation of arrival, service, and scaling curves. However, for the derivation of probabilistic performance bounds (e.g., the delay of a data unit through the unreliable link), we now need to take into account the stochastic nature of the unreliable link and therefore of the underlying scaling process.

Firstly, let us assume that there exists a sample-path over which the scaling functions of all the flows, original and retransmissions, do not violate their sample-path stochastic scaling curves $\overline{S}_i^{\overline{\varepsilon}}$'s.If this assumption applies, according to the calculation described in the previous section, we can firstly derive the arrival curve for the aggregate arrivals as $\sum_{i=0}^{N} \alpha^{(i)}$ (see Section 2.2.1). And the service curve for the aggregate arrivals is β. Now we obtain the delay

bound

$$\forall t : d(t) \leq h\left(\sum_{i=0}^{N} \alpha^{(i)}, \beta\right) .$$

Note that this delay bound excludes the direct delay contributions of the feedback for retransmitted packets, though it takes the feedback loops burstiness increase effect into account (see also Section 6.3). These delay contributions can simply be added according to the number of necessary retransmissions and the maximum feedback delay, but are omitted in the following due to their rather uninteresting nature.

Secondly, we need to calculate the violation probability of the above sample-path assumption. As discussed above, each flow i is subject to a scaling process $\mathcal{S}_i$, and all $\mathcal{S}_i$'s form a partition of the overall scaling process $\mathcal{S}$. Given an *i.i.d.* overall scaling process $\mathcal{S}$ (e.g., the BSC), all $\mathcal{S}_i$'s are *i.i.d.* as well and mutually independent. A probabilitic delay bound can be computed by calculating the probability of the sample-path event that the stochastic scaling curves are not violated as follows

$$\begin{aligned} Pr\left(d(t) \leq h\left(\sum_{i=0}^{N} \alpha^{(i)}, \beta\right)\right) &\geq Pr\left(\bigcap_{i=1}^{N} \left\{\overline{S}_i^{\overline{\varepsilon}} \text{not violated}\right\}\right) \\ &= \prod_{i=1}^{N} Pr\left(\overline{S}_i^{\overline{\varepsilon}} \text{not violated}\right) \\ &\geq (1-\overline{\varepsilon})^N , \end{aligned} \tag{6.4}$$

by invoking the statistical independence in the second line. Note that, if the $\mathcal{S}_i$'s were not *i.i.d.* , we could still use the union bound to compute the violation probability (of course, resulting in a more conservative bound).

Similar reasoning can be applied to compute the probabilistic backlog bound for all of the flows:

$$Pr\left(b(t) \leq v\left(\alpha^{(0)} + \alpha^{(1)} + \cdots + \alpha^{(i)}, \beta\right)\right) \quad \geq \quad (1-\overline{\varepsilon})^N ,$$

with statistical independence assumptions. Note, given a violation probability ε of the stochastic delay bound, we can either use $(1-\overline{\varepsilon})^N = 1-\varepsilon$ assuming the homogeneous link quality among the flows (original and retransmitted), or numerically optimize the bound value subject to that $\prod_{i=1}^{N}(1-\overline{\varepsilon}_i) = 1-\varepsilon$ (independent case), and more generally, $1 - \sum_{i=1}^{N} \overline{\varepsilon}_i = 1 - \varepsilon$ (discussed in Section 3.4.2).

6.3 Numerical Evaluation

In order to illustrate the application of the model, let us go through a numerical example in this section. Before the calculation, we state some necessary assumptions.

Assumptions: *Consider the scaling process $\mathcal{S}$ to be a BSC with loss probability p varying from 0.1 to 0.9 (from normal state to channel collapse). The arrival curve of the input flow is $\alpha = \gamma_{r,b} = \gamma_{0.1,3}$. The service curve is $\beta = \beta_{R,T} = \beta_{1,3}$. The service curve of the feedback delay is $\delta_W = \delta_8$. The violation probability of the sample-path stochastic scaling curve is $\overline{\varepsilon} = 0.001$. The value of W will later be varied from 0 to 40, in order to illustrate the impact of the feedback delay.*

Target of Calculations: *We compute probabilistic delay bounds with different loss probabilities and for different number of retransmissions $N = 1, 2, 3$, as well as with varying feedback delays.*

First, in order to illustrate how to derive the arrival curves for all the retransmission flows, let us pick $N = 2$ as an example. From Section 6.2.1 we know that a sample-path stochastic scaling curve for a BSC with parameter p can be calculated as in Eq. (6.2). Let $C = p$ and $B = 1 - \overline{\varepsilon}$. We can now use the results from Section 6.2.2 to derive the formulas of arrival curves $\alpha^{(1)}$ and $\alpha^{(2)}$ for the two retransmission flows. As a result, we derive the arrival curves $\alpha^{(1)}$, $\alpha^{(2)}$ and the leftover service curves $\beta^{(1)}$, $\beta^{(2)}$. For example, for $p = 0.1$, these are depicted in Figure 6.4 together with horizontal lines representing the delay bounds for each retransmission flow.

Comparing the rates of α and $\alpha^{(1)}$, we observe that the rate of $\alpha^{(1)}$ (i.e., Cr) is much smaller than the rate of α (i.e., r); this is because $\alpha^{(1)}$ is the retransmission flow caused by data loss and the probability of data loss is not too high, which means that the effect of the retransmission flow is rather weak (i.e., $C << 1$). Consequently, the retransmission flow for $\alpha^{(1)}$, which is $\alpha^{(2)}$, is even weaker. Next we observe from Figure 6.4 that $b > b_1 > b_2$. This is intuitive, since b as the original flow's parameter is expected to be greater than b_1 and b_2.

With the computed values of $\alpha^{(1)}$ and $\alpha^{(2)}$ we can next calculate the arrival curve for the aggregate flows as $\sum_{i=0}^{2} \alpha^{(i)} = \gamma_{0.111,5.59}$. And using Eq. (6.4) we can now compute the probabilistic delay bound with two retransmissions (recall that $N = 2$)

$$Pr\left(d(t) \leq h\left(\sum_{i=0}^{2} \alpha^{(i)}, \beta\right) = 8.5902\right) \geq (1 - \overline{\varepsilon})^2 = 0.9980\,.$$

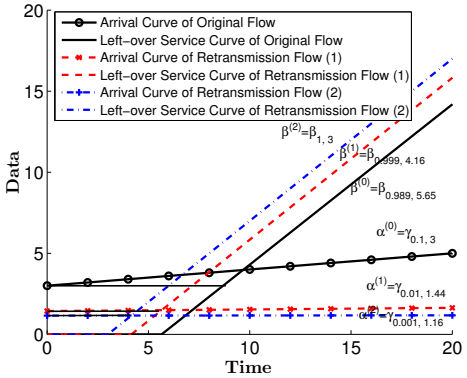


Figure 6.4: Arrival, service curves, and delay bounds.

Next, we show the delay bounds for $N = 1, 2, 3$ and $p = 0.1, 0.2, \cdots, 0.9$ in Figure 6.5. In the figure, for each N, we plot a curve for p from 0.1 to 0.9. The delay bounds are expectedly increasing in N and p. Clearly, the more data is lost and the higher reliability the communication requires (higher N), the higher is the delay. Most interestingly, the figure shows for $N = 3$ a steep rise for increasing loss probabilities, whereas for $N = 1, 2$ the bounds are relatively insensitive to the loss probability. Hence, we encounter an interesting phase transition phenomenon for the impact of the number of retransmissions on the performance bounds, i.e., there exists a threshold value for N above which the performance bounds blow up. As discussed in Section 6.2.3, the feedback delay is excluded from the total delay bound, such that this blow-up does not directly and trivially relate to the maximum feedback delay but is due to queueing effects only. From the stability condition Eq. (6.3), we can clearly derive a maximal N such that the node is still in stable state. A potential usage of this knowledge could be to dynamically adapt the maximum number of retransmissions per data unit to control the trade-off between delay and reliability according to the current utilization and loss characteristics of a lossy link.

In Figure 6.6, we show the impact of the maximum feedback delay on the delay bounds for $N = 1, 2, 3$ and $W = 0, 1, \cdots, 40$. The delay bounds are increasing in the maximum feedback delay for all the three cases. Although we have excluded the direct contribution of the feedback delay, it still increases the possibility to cumulate burstiness in the retransmitted flow at the server. Clearly, this cumulated burstiness eventually increases the delay

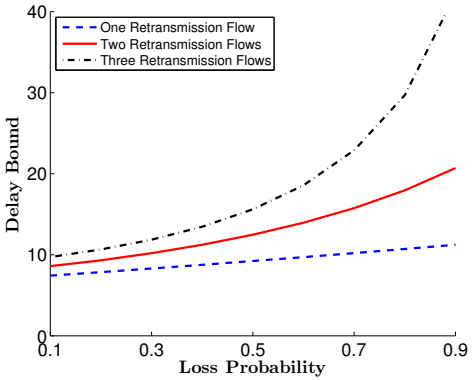


Figure 6.5: Delay bounds with retransmission attempts i ($i = 1, 2, 3$).

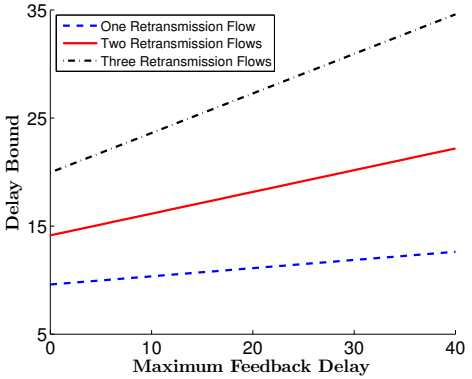


Figure 6.6: Delay bounds with changing maximum feedback delay (loss probability $p = 0.7$).

bound. The higher the maximum retransmission number, the more often a retransmitted packet may experience the feedback delay, and thus, the burstier the aggregate flow becomes. This is illustrated in the Figure 6.6 by the increasing slopes for $N = 1, 2, 3$.

In this chapter, we made a step forward on the way to model unreliable networks using the stochastic network calculus. Based on the stochastic data scaling elements we showed how to model and analyze an unreliable link that employs a retransmission-based loss recovery. Solving this model involved a fixed-point analysis yielding probabilistic performance bounds. In the numerical example, we illustrated how to apply the theoretical results and demonstrated the model's capabilities to provide interesting insights into the system behavior. In particular, we showed that even at small utilizations, a relatively small number of retransmission attempts already lends itself to a delay bound's blow-up. This provides incentives for protocols to dynamically adapt the maximum number of retransmissions. Moreover, our model can also reveal the quantitative impact of the feedback delay on the delay bound of the aggregate flow. An interesting and desirable future work along the way in this chapter is to investigate whether our technique to analyze single unreliable links and the concatenation principle from network calculus can be combined to analyze unreliable networks.

Chapter 7

Scaling Element for Variable Length Packet Transmissions

The scaling element models we introduced to this end have a limitation: they scale the flow only at the granularity of identical length data units (bits or packets). This is quite restrictive as many networks use variable-length packets and information, and events like sending and receiving cannot be observed at the bit-level. In this paper, we therefore propose a new scaling element which respects flows as a sequence of (variable-length) packets rather than just bits. The critical challenge in defining such a scaling element at the packet-level is that it should preserve the convolution-form expression of multi-node networks. To ease the exposure we again focus on the abstract but widely applicable flow transformation operation: the demultiplexing of packets, i.e. to thin a flow of packets by selecting only some of them, e.g. due to network operations such as routing, load balancing, transcoding, or simply loss of packets.

We are, of course not the first to treat the case of variable-length packets (though, to the best of our knowledge we are the first to take this into account under flow transformations). In particular, the packetizer element [95, 36] has been introduced in network calculus to deal with flows of variable-length packets. [100, 80] have extended it to the stochastic settings. [100] models heavy-tailed arrivals with packet distributions. [80] reveals the inherent dependence brought by the packet process to the arrivals and services. We

also use the packetizer but now in combination with a scaling element in order to model flow transformations at the packet level. In Chapter 5 we showed a novel model to understand the demultiplexing for first-in-first-out (FIFO) servers and to compute tighter end-to-end delay bounds. Yet, for the n-node network to preserve the convolution-form the computation of the performance measures turned out to be hard. Therefore, in this chapter, we commute the packet-level scaling element and the dynamic server (as proposed for the bit level in Chapter 4), in order to provide a tractable end-to-end delay bound computation. We generalize the results from Chapter 4 by introducing the so-called packet scaling element which scales on each variable length packet. The results of this chapter are from the joint works with J. Schmitt [149, 147, 148].

7.1 Modeling the Demultiplexing of Variable Length Packet Flows

In real-world the bits perhaps belong to different (variable length) packets, and transformations happen on the whole packet instead of each bit. For the demultiplexing example, we may simply know the routing probability of a complete packet to one destination. To model this (packet-level) demultiplexing operation, we need to extend the scope of the scaling element. Yet, before we do that, we first integrate variable length packets into the network calculus framework. We denote the packet lengths as a sequence of positive integer random variables $l_1, l_2, \ldots$. A packet process $L(n), n \geq 1$ is a cumulation of these *r.v.* 's, $L(n) = l_1 + l_2 + \cdots + l_n$, and $l_n = L(n) - L(n-1)$ with $L(0) = 0$. A packet flow is modeled using the definition of packetizer ([36], [95]).

Definition 19. (Packetizer) Given a packet process $L(n)$ and an arrival process $A(t)$, an L-packetizer is a network element expressed by a function $P^L(\cdot)$ satisfying for all $A(t), t \geq 0$

$$P^L\left(A(t)\right) = L(N_t)\,,$$

where

$$N_t = \max\left\{m : L(m) \leq A(t)\right\}. \tag{7.1}$$

We say that a flow $A(t)$ is L-packetized if $A(t) = P^L\left(A(t)\right)$ for any $t \geq 0$. So a packet flow is an L-packetized arrival process. Note, the function P^L is not restricted to a real network element with a queue, it can also be used

to parse a bit flow (e.g., with marks) into packets and not change its timing. In the rest of this chapter, we will use both meanings.

Now we consider the demultiplexing of a packet flow. We extend the definition of the scaling element using the "L-regulated" Eq. (3.3).

Definition 20. (Packet Scaling Element) A packet scaling element consists of an L-packetized arrival process $A(t) = \sum_{i=1}^{N_t} l_i$, a packet scaling process $\mathbf{X}$ taking non-negative integer values and a scaled packetized flow defined for all $t \geq 0$ as

$$A^{\mathbf{X}}(t) = \sum_{i=1}^{N_t} l_i X_i \, .$$

We can use the packet scaling element to model the transformation of the packet flow, specifically, the demultiplexing case. Accordingly, $l_i X_i$ means $l_i \cdot \mathbf{1}_{\{\text{"packet } i \text{ goes to destination (1)"}\}}$, i.e., demultiplexing operates on each packet and X_i equals either 0 or 1.

A packet flow is usually processed or served by a queueing system before or after being demultiplexed. To analyze the delay of a packet through this system we distinguish two models. One is, after being served by each node the output is always packets, i.e., the bits are packetized by a packetizer P^L. The other is, there is no packetizer after service, yet we observe from the bit flow the last bit of each packet according to a packet process L. In Chapter 3, 4 we derive the end-to-end delay bound for the bit flow under flow transformation. The second case can be a critical challenge for that approach (L-modulated scaling process and sampling due to L). In this chapter, we focus on the first case and assume that the packetization is not changed along the path. Therefore we need to define a *packetized server* as a bit server followed by a packetizer P^L, and denote the dynamic server of it as $S^L(s,t)$. Given the dynamic server of the bit server S and the packet process L and assuming that a maximum packet size l_{max} exists, i.e. $l_{max} < \infty$, we obtain a possible S^L,

$$S^L(s,t) = [S(s,t) - l_{max}]_+ \, . \tag{7.2}$$

The proof follows using a busy time analysis.

We illustratively summarize the network elements in Figure 7.1. Since we observe a packetized flow, it is then apparent to pursue the delay of the packets. Now we slightly adjust the virtual "bit" delay definition (see Eq. (2.3)) to the packet delay.

Definition 21. (Packet Delay) A process $W(t)$ is called packet delay (pro-

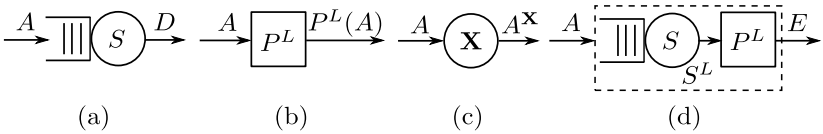


Figure 7.1: Network elements: (a) dynamic server, (b) packetizer, (c) packet scaling element, (d) packetized server.

cess), if for all $t \geq 0$

$$W(t) = \inf \left\{ d \geq 0 : P^L\left(A(t)\right) \leq P^L\left(D(t+d)\right) \right\}.$$

Here we assume the service is FIFO. The packet delay is a virtual delay that would be experienced by a packet which arrives at time t. Next, we provide the derivation of the end-to-end packet delay bounds.

7.2 Delay Bounds of a Network with Flow Demultiplexing

In this section, we compute the end-to-end packet delay for networks with multiple demultiplexers. According the results stated in previous chapters, there are two ways to compute the end-to-end delay: (1) commute service and scaling elements (see also Section 3.1 and 4.2), (2) get the leftover service for the flow of interest if the server uses FIFO scheduling (see also Section 5.2). In this chapter, we use the first, i.e., we repeatedly move all the packet scaling elements in front of the packetized servers and obtain the convolution-form of the network. Then we calculate the end-to-end delay bounds. Here, we have two choices: one is to "normalize" the packet flow as well as the bitwise service with packet size, so that the observation is directly on each packet irrespective of its size ($\rightarrow$ Section 7.2.1); the other is to use the definition of packet delay (Definition 21) and derive the delay bound directly through observing the original bit flow with packetizers ($\rightarrow$ Section 7.2.2). For the packet flow we assume that the packet lengths l_i's are *i.i.d.* with $l_{max} < \infty$. In fact, this assumption can be justified in many real-world applications with heterogeneous, large-scale, and high degree of multiplexing environment.

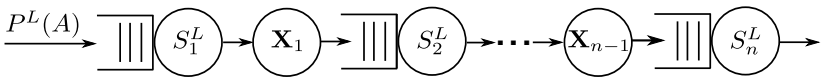


Figure 7.2: A network model consisting of packetized arrivals, services and packet scaling elements.

7.2.1 Observing the Packet Flow

Consider Figure 7.2, we lift our observation of the flow directly from the bit level to the packet level. This means we view each packet as a single data unit ignoring its size. Then we re-express the service this packet receives. After doing so we can derive the end-to-end packet delay bound directly using the calculation from Chapter 4.

Consider the arrivals consist of packets whose arrival times are defined as the arrival time of the last bit, we can model these time jumps with a counting process and together with a packet size distribution, model the arrival process as a compound process - $A(t) = \sum_{i=1}^{N(t)} l_i$, where $\{N(t), t \geq 0\}$ is the counting process, i.e., the number of arriving packets in time t, and l_i is the i-th packet size. This seems to be a slightly different description of a packetized flow, because here we do not assume a packetizer element in the network. Yet, the packetized process resulting from packetizer is also covered by this definition. Consequently, we obtain an arrival process of packets - $\{N(t), t \geq 0\}$. We call this approach "normalization" of the bit flow by the packet sizes.

Such a sequence of packets will be served by a service element described by the bitwise service capacity together with a packetizer. How much service capacity does a packet receive? To answer this question is not very hard. For example, assume that a packet with length l will be served by a server with constant service capacity $C\ bits/s$, so the service rate for this packet is $C/l\ packets/s$. This is the "normalization" on the service side. The constant capacity server is transformed into a variable capacity server. We write it as $S^{norm}(s,t) = \sum_{i=s}^{t} c(i)$ for all $0 \leq s \leq t$. Here all the $c(i)$'s are the time varying capacities of serving a packet at time i.

In Chapter 4, we derive the end-to-end delay bounds for a flow with identically sized data units. The derivation is based on the MGF bounds of the arrivals and the services and expresses a network with flow transformations in a convolution-form. Therein, the servers are assumed to have constant MGF bounds. However, to use the same derivation is quite challenging, because now on one hand, the servers have variable capacities and to know their MGF bounds is hard; on the other hand, they are "normalized" by the same

packet process and hence dependent of each other.

To obtain the MGF of the dynamic server we can firstly express the inter-service time. Then we use the (inverse) Laplace transform of the convolution of inter-arrival times and packet size distributions to compute the $p.d.f.$ of the inter-service time. Thirdly, we use renewal theory to obtain the $p.d.f.$ of the counting process of the service. At last, the MGF follows by its definition. About the dependency, Hölder inequality might be a solution, but many parameters are introduced (see Eq. (2.13)). This approach lays its focus on the accuracy of the expressiveness, yet, loses the analytical tractability. In this section, we just want to provide a method to calculate the end-to-end delay for variable-length packet flows that follows closely the approach in Chapter 4 and then compare it against the more sophisticated method using the packet scaling element. Assume that the bit-wise capacity $S(t)$ is offered work-conserving with variant rates and let $S(t) \geq Ct$ for any $t \geq 0$ such that MGF bound $M_{S(t)}(-\theta) \leq e^{-\theta Ct}$ for $\theta > 0$, then we can vaguely write $c(i) \geq C/l_x$, where l_x means either some packet length or ∞. We assume the packet size has a limit, i.e., $l_x \leq l_{max}$. Clearly, $c(i) \geq C/l_{max}$. We obtain a lower bound of this normalized dynamic server $S^{norm}(s,t) \geq \frac{C}{l_{max}}(t-s)$. Now, we represent the dynamic server as a server with the normalized capacity C/l_{max}, which, at the same time, solves the above problems. Consider the network in Figure 7.2 again. We assume the compound process as the arrivals instead of using packetizer. We also assume $M_{S_j(t)}(-\theta) \leq e^{-\theta C_j t}, j \geq 1$ at each bit server. The end-to-end delay has the following stochastic bound (refer to Eq. (4.10))

$$Pr(W > d) \leq K^n b^d \, .$$

We point out, the only difference between this result and Theorem 9 in Chapter 4 is that the MGF bound of each service is

$$M_{S_i^{norm}(s,t)}(-\theta) \leq e^{-\theta \frac{C_i}{l_{max}}(t-s)} \, . \tag{7.3}$$

We also point out, when we do the "normalization" to the service, whether there exists a real packetizer component or not does not change the packet delay analysis, because only after the last bit of a packet is served by the bit-wise server, the service of this packet is considered to be finished, this is just as if there was a packetizer virtually.

7.2.2 Observing the Original Bit Flow with Packetizers

In the previous subsection, we provided an approach to calculate end-to-end delay bounds for variable-lengths packet flows under flow demultiplexing

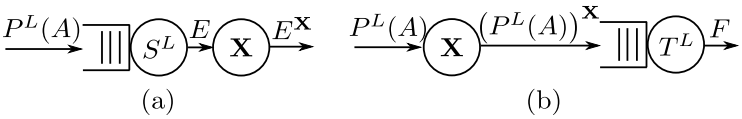


Figure 7.3: Commutation of packetized service and packet scaling.

which observes a flow on the packet-level rather than the bit level. Now we directly observe the flow on the bit level. We know that, when deriving the end-to-end delay bound we should avoid summing up the delay bounds node-by-node, but rather use the "pay burst only once" principle. To do so, we express the network in convolution-form through moving the scaling elements between two servers to the front. The challenge now is that the scaling element is not at the bit level anymore. We provide the following lemma to commute the service and scaling element at packet-level, which is instrumental to the derivation of end-to-end delay bounds.

Lemma 8. *(Commutation of Packet Scaling Element and Dynamic Server) Consider system (a) and (b) with packetized arrivals $A(t) = P^L(A(t))$ in Figure 7.3. We define $T^L(s,t) := \sum_{i=M_s+1}^{N_t} l_i X_i$ as the exact dynamic server in (b), where $A(s) = \sum_{i=1}^{M_s} l_i$, $A(s) + S^L(s,t) = \sum_{i=1}^{N_t} l_i$. If A, S, $\mathbf{X}$, and L are independent, then for all $t \geq 0$, the departures satisfy*

$$F(t) \leq E^{\mathbf{X}}(t) \, .$$

Proof. Because T^L is an exact dynamic server, we have

$$\begin{aligned} F(t) &= \inf_{0 \leq s \leq t} \left\{ A^{\mathbf{X}}(s) + T^L(s,t) \right\} \\ &= \inf_{0 \leq s \leq t} \left\{ \sum_{i=1}^{M_s} l_i X_i + \sum_{i=M_s+1}^{N_t} l_i X_i \right\} \\ &= \inf_{0 \leq s \leq t} \sum_{i=1}^{N_t} l_i X_i, \text{where } A(s) + S^L(s,t) = \sum_{i=1}^{N_t} l_i \\ &= \left(\inf_{0 \leq s \leq t} \sum_{i=1}^{N_t} l_i \right)^{\mathbf{X}} \end{aligned}$$

$$
\begin{aligned}
&= \left(\inf_{0 \le s \le t} \left\{ A(s) + S^L(s,t) \right\} \right)^{\mathbf{X}} \\
&\le E^{\mathbf{X}}(t) \, .
\end{aligned}
$$

In the fourth line we can use a proof by contradiction. □

Through this lemma, we see that there are less departures for the transformed system than in the original system, which ensures that the delays are larger. The expression of T^L looks complicated, but the meaning is clear. $\sum_{i=M_s+1}^{N_t} l_i$ are the packets served from time s to t. And because after passing through a scaling element $\mathbf{X}$, only a scaled part of these packets is sent to the next server, the service they received should also be a scaled part of the total service. After recursively using this lemma we get an expression for the network in terms of a scaled arrival process served by a dynamic server in convolution-form. Recall that, the arrivals have the form

$$
\left(\cdots \left(A^{\mathbf{X}_1} \right)^{\mathbf{X}_2 \cdots} \right)^{\mathbf{X}_k} (t) \, ,
$$

if there are k packet scaling elements. We still denote it as $A^{(k)}(t)$. The alert reader may note that, for the bit flow, the concatenation of scaling elements can be naturally formulated as $\left(A^{\mathbf{X}_1} \right)^{\mathbf{X}_2} (t) = \sum_{i=1}^{\sum_{j=1}^{A(t)} X_{1,j}} X_{2,i}$, whereas for the packet flow, this is not true any more. We point out that they are just different in appearance but the same in essence - after each round of scaling we choose a part of the packets (bits) from the input flow. Therefore, we provide the delicate expression of $A^{(k)}(t)$, which will be used in the rest of this section. Assume an L-packetized flow $A(t) = l_1 + l_2 + \cdots + l_{N_t}$, where N_t is given in Eq. (7.1). We first denote the packets respectively the number of packets in the arrivals until time t after each round of scaling as $l_{k,i}$ respectively $m_t^{(k)}$. Clearly for $k > 0$

$$
\begin{aligned}
m_t^{(0)} &= N_t \, , \\
m_t^{(1)} &= \sum_{i=1}^{m_t^{(0)}} 1_{\{X_{1,i}>0\}} \, , \\
&\cdots \\
m_t^{(k)} &= \sum_{i=1}^{m_t^{(k-1)}} 1_{\{X_{k,i}>0\}} \, . \qquad (7.4)
\end{aligned}
$$

Further we denote $A^{(k)}(t)$ as

$$\begin{aligned} A^{\mathbf{X}_1}(t) &= l_1 X_{1,1} + \cdots + l_{N_t} X_{1,N_t} \\ &= l_{1,1} + \cdots + l_{1,m_t^{(1)}} , \\ \left(A^{\mathbf{X}_1}\right)^{\mathbf{X}_2}(t) &= l_{1,1} X_{2,1} + \cdots + l_{1,m_t^{(1)}} X_{2,m_t^{(1)}} \\ &= l_{2,1} + \cdots + l_{2,m_t^{(2)}} , \\ &\cdots \\ A^{(k)}(t) &= l_{k,1} + \cdots + l_{k,m_t^{(k)}} \\ &= \sum_{i=1}^{m_t^{(k-1)}} l_{k-1,i} X_{k,i} . \end{aligned} \tag{7.5}$$

The recursion is just to do sampling in each round. We point out that this expression is more important than the concrete values of each l to further derive the delay bounds.

Next, we provide two useful lemmas for deriving the end-to-end delay bounds.

Lemma 9. *(Stationarity Bound) Assume that the packets l_i's of a packet process L are i.i.d., the X_i's of a packet scaling element $\mathbf{X}$ are also i.i.d., A and B are two L-packetized arrival processes, then for all $s, t, x > 0$,*

$$Pr\left(A^{\mathbf{X}}(t) - B^{\mathbf{X}}(s) \geq x\right) \leq Pr\left(\left(A(t) - B(s)\right)^{\mathbf{X}} \geq x\right) .$$

Proof. On the one hand we have

$$\begin{aligned} & Pr\left(A^{\mathbf{X}}(t) - B^{\mathbf{X}}(s) \geq x\right) \\ = \;& Pr\left(\sum_{i=1}^{N_t} l_i X_i - \sum_{i=1}^{N_s} l_i X_i \geq x, A(t) > B(s)\right) \\ \leq \;& Pr\left(\sum_{i=N_s+1}^{N_t} l_i X_i \geq x\right) . \end{aligned}$$

On the other hand we know

$$Pr\left(\left(A(t) - B(s)\right)^{\mathbf{X}} \geq x\right) = Pr\left(\sum_{i=N_s+1}^{N_t} l_i X_{i-N_s} \geq x\right) .$$

X_i's and l_i's are *i.i.d.* , which completes the proof. □

Lemma 10. *(Recursive MGF Bound of Scaled Process) Assume that A is an L-packetized process, S_i^L is the packetized server, l_i's are i.i.d. with maximal length $l_{max} < \infty$, and $\mathbf{X}_i$'s are MMOO loss processes, independent of A and S_i^L, if we denote $V_{n-1}(\theta_n)$ as*

$$E\left[e^{\theta\left(\cdots\left(A(t-s)-S_1^L(s,u_1)\right)^{\mathbf{X}_1}-\cdots-S_{n-1}^L(u_{n-2},u_{n-1})\right)^{\mathbf{X}_{n-1}}}\right] ,$$

then for all $0 \le s \le u_1 \le \cdots \le u_{n-1} \le t$, and $n > 1$,

$$V_{n-1}(\theta_n) \le e^{-\theta_{n-1}S_{n-1}^L(u_{n-2},u_{n-1})}V_{n-2}(\theta_{n-1}) ,$$

where $\theta_i > 0$, $1 \le i \le n$ is given in the proof.

Proof. Let $\theta_n = \theta$, which can be a given value. To simplify the notation, let us assume

$$\left(\cdots\left(A(t-s)-S_1^L(s,u_1)\right)^{\mathbf{X}_1}-\cdots-S_{n-2}^L(u_{n-3},u_{n-2})\right)^{\mathbf{X}_{n-2}} = \sum_{i=1}^{P} l_i ,$$

where P is a $r.v.$ and

$$S_{n-1}^L(u_{n-2},u_{n-1}) = \sum_{i=1}^{Q} l_i ,$$

where Q is a $r.v.$. Because l_i's are *i.i.d.* and $\mathbf{X}_{n-1}$ is a MMOO loss process, we can see that $\left(\sum_{i=1}^{P} l_i - \sum_{i=1}^{Q} l_i\right)^{\mathbf{X}_{n-1}}$ is also an MMOO process. It has θ-envelope $R_{n-1}(\theta)$ (see Eq. (2.12), also given in [36, 49]), e.g., for an *i.i.d.* scaling process,

$$R_{n-1}(\theta) = \frac{1}{\theta}\log M_{\mathbf{X}_{n-1}}(\log M_l(\theta)) .$$

Thus we obtain

$$V_{n-1}(\theta_n) = E\left[e^{\theta\left(\sum_{i=1}^{P} l_i-\sum_{i=1}^{Q} l_i\right)^{\mathbf{X}_{n-1}}}\right] \le E\left[e^{\theta R_{n-1}(\theta)(P-Q)}\right] . \quad (7.6)$$

To get a bound on the right term above, we first note that, because l_i's are

i.i.d. ,

$$M_{\sum_{i=Q+1}^{P} l_i}(\theta_{n-1}) = E\left[e^{\log M_l(\theta_{n-1})(P-Q)}\right] . \tag{7.7}$$

We can derive a bound on this by using l_{max} and noting that S_{n-1}^L is independent of other S_i^L's and $\mathbf{X}_i$'s. So we can easily obtain the following MGF bound

$$M_{\sum_{Q+1}^{P} l_i}(\theta_{n-1}) \leq e^{-\theta_{n-1} S_{n-1}^L(u_{n-2}, u_{n-1})} V_{n-2}(\theta_{n-1}) . \tag{7.8}$$

Combining Eq. (7.7) and (7.8) we have

$$E\left[e^{\log M_l(\theta_{n-1})(P-Q)}\right] \leq e^{-\theta_{n-1} S_{n-1}^L(u_{n-2}, u_{n-1})} V_{n-2}(\theta_{n-1}) .$$

Using this bound in Eq. (7.6) and letting

$$\theta_n R_{n-1}(\theta_n) = \log M_l(\theta_{n-1}) ,$$

completes the proof. This equation implies every $\theta_i, 1 \leq i \leq n$ if given $\theta_n = \theta$. □

We now derive the end-to-end delay bound and show that it grows in $\mathcal{O}(n)$ where n is the number of nodes.

Theorem 12. *(End-to-end Delay Bounds in a Packet Flow Transformation Network) Consider the network scenario from Figure 7.2 where an L-packetized arrival process $A(t) = P^L(A(t))$ traverses a series of stationary and (mutually) independent bit level service elements followed by an L-packetizer and scaling elements denoted by $S_1^L, S_2^L, \ldots, S_n^L$ and i.i.d. loss processes $X_1, X_2, \ldots, X_{n-1}$, respectively. Assume the packet lengths of L - l_i's are i.i.d. . Assume the MGF bounds $M_{A(s,t)}(\theta) \leq e^{\theta r_A(\theta)(t-s)}$ and $M_{S_k(t)}(-\theta) \leq e^{-\theta C_k t}$, for $k = 1, 2, \ldots, n$, and some $\theta > 0$. We also assume that the maximum packet length $l_{max} < \infty$. Under a stability condition, to be explicitly given in the proof, for $\theta_i > 0, i = 1, 2, \ldots, n$, we have the following end-to-end steady state delay bounds for all $d \geq 0$*

$$Pr(W > d) \leq e^{\left(\sum_{i=1}^{n} \theta_i + \theta_1\right) l_{max}} K^n b^d , \tag{7.9}$$

where the constants K and b are also given in the proof. Moreover, the ε-quantiles scale as $\mathcal{O}(n)$, for $\varepsilon > 0$.

Proof. First we use Lemma 8 to transform the system view. To do so, we iteratively commute the packetized server and the packet scaling element k

times. See Figure 7.4. Since the output of the transformed system is smaller than or equal to the original system, the delay bound of the transformed one must be larger than or equal to the delay bound of the original one, hence, we compute the delay bound of this transformed system.

Figure 7.4: Apply Lemma 8 for k times.

Next, fix $t, d \geq 0$. For $k, s \geq 0$ we define $U_0(s, u_0) = A(s)$, for $u_0 = s$, and then recursively

$$U_k(s, u_k) = \big(U_{k-1}(s, u_{k-1}) + S_k^L(u_{k-1}, u_k)\big)^{\mathbf{x}_k}$$

for $k \geq 1$ and $u_{k-1} \leq u_k$. We prove the theorem at the first steps by induction. For $k \geq 1$ we assume the following two statements $(\mathcal{S}_1)$ and $(\mathcal{S}_2)$ for the induction:

$$(\mathcal{S}_1) \qquad Pr(W_k(t) > d) \leq \sum_{0 \leq s \leq t} \sum_{s \leq u_1 \leq \cdots \leq u_{k-1} \leq t+d} Pr\left(A^{(k-1)}(t) > U_{k-1}(s, u_{k-1}) + S_k^L(u_{k-1}, t+d)\right) ,$$

and for fixed s and u_k,

$$(\mathcal{S}_2) \qquad \left(A^{(k-1)}(s) + T_{k-1}^L \otimes S_k^L(s, u_k)\right)^{\mathbf{x}_k} = \inf_{s \leq u_1 \leq \cdots \leq u_k} U_k(s, u_k) ,$$

where T_k^L is defined recursively as $T_0^L(0) = 0$, $T_0^L(s) = \infty$ for all $s > 0$, and for N_s the number of packets in $A(s)$

$$T_k^L(s, u_k) := \sum_{i=m_s^{(k-1)}}^{N_{u_k}} l_{k-1,i} X_{k,i} \text{ , where } \sum_{i=1}^{m_s^{(k-1)}} l_{k-1,i} = A^{(k-1)}(s) ,$$
$$\sum_{i=1}^{N_{u_k}} l_{k-1,i} = A^{(k-1)}(s) + T_{k-1}^L \otimes S_k^L(s, u_k) . \qquad (7.10)$$

First we prove the initial step of the induction, i.e., $k = 1$. For statement $(\mathcal{S}_1)$, we have

$$Pr(W_1(t) > d) \quad = \quad Pr(A(t) > D(t+d))$$

$$
\begin{aligned}
&\leq \quad Pr\left(A(t) > A \otimes S_1^L(t+d)\right) \\
&\leq \quad \sum_{0\leq s\leq t} Pr\left(A(t) > A(s) + S_1^L(s,t+d)\right) \\
&= \quad \sum_{0\leq s\leq t} Pr\left(A^{(0)}(t) > U_0(s,u_0) + S_1^L(s,t+d)\right) .
\end{aligned}
$$

In the first line we used the definition of packet delay. In the second line we used the definition of dynamic server. And in the third line we expanded the convolution and used Boole's inequality. In turn for statement $(\mathcal{S}_2)$, we have

$$
\begin{aligned}
&\left(A^{(0)}(s) + T_0^L \otimes S_1^L(s,u_1)\right)^{\mathbf{X}_1} \\
&= \quad \left(A(s) + \inf_{s\leq x\leq u_1}\left\{T_0^L(s,x) + S_1^L(x,u_1)\right\}\right)^{\mathbf{X}_1} \\
&= \quad \left(A(s) + S_1^L(s,u_1)\right)^{\mathbf{X}_1} \\
&= \quad \inf_{s\leq u_1}\left(A(s) + S_1^L(s,u_1)\right)^{\mathbf{X}_1} \\
&= \quad \inf_{s\leq u_1}\left(U_0(s,u_0) + S_1^L(u_0,u_1)\right)^{\mathbf{X}_1} \\
&= \quad \inf_{s\leq u_1} U_1(s,u_1) .
\end{aligned}
$$

In the third line we used that $T_0^L(0) = 0, T_0^L(s) = \infty$. In the fourth line we rewrote the third line using $\inf$, because s and u_1 are actually fixed. In the fifth line we used the definition of U_0. In the last line we used the recursive definition of U_k.

For the induction we next assume that $(\mathcal{S}_1)$ and $(\mathcal{S}_2)$ hold for $k \geq 1$. Then we prove them for $k+1$. Using the argument from the initial step of the induction we can write the end-to-end delay until the $k+1^{th}$ node

$$
\begin{aligned}
&Pr(W_{k+1}(t) > d) \\
&\leq Pr\left(A^{(k)}(t) \geq \inf_{0\leq s\leq t+d}\left\{A^{(k)}(s) + T_k^L \otimes S_{k+1}^L(s,t+d)\right\}\right) \\
&\leq \sum_{0\leq s\leq t}\sum_{s\leq u_k\leq t+d} Pr\left(A^{(k)}(t) \geq A^{(k)}(s) + T_k^L(s,u_k) + S_{k+1}^L(u_k,t+d)\right) \\
&= \sum_{0\leq s\leq t}\sum_{s\leq u_k\leq t+d} Pr\left(A^{(k)}(t) \geq \sum_{i=1}^{m_s^{(k-1)}} l_{k-1,i}X_{k,i} + \right.
\end{aligned}
$$

$$
\begin{aligned}
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \sum_{i=m_s^{(k-1)}}^{N_{u_k}} l_{k-1,i}X_{k,i} + S_{k+1}^L(u_k, t+d)\Big)\\
&= \sum_{0\le s\le t}\ \sum_{s\le u_k\le t+d} Pr\Big(A^{(k)}(t) \ge \sum_{i=1}^{N_{u_k}} l_{k-1,i}X_{k,i} + S_{k+1}^L(u_k, t+d)\Big)\\
&= \sum_{0\le s\le t}\ \sum_{s\le u_k\le t+d} Pr\Big(A^{(k)}(t) \ge \big(A^{(k-1)}(s) +\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad T_{k-1}^L \otimes S_k^L(s,u_k)\big)^{\mathbf{X}_k} + S_{k+1}^L(u_k, t+d)\Big)\\
&= \sum_{0\le s\le t}\ \sum_{s\le u_k\le t+d} Pr\Big(A^{(k)}(t) \ge \inf_{s\le u_1\le\cdots\le u_k} U_k(s,u_k) + S_{k+1}^L(u_k, t+d)\Big)\\
&\le \sum_{0\le s\le t}\ \sum_{s\le u_1\le\cdots\le u_k\le t+d} Pr\big(A^{(k)}(t) \ge U_k(s,u_k) + S_{k+1}^L(u_k, t+d)\big)\ .
\end{aligned}
$$

In the third line we expanded the convolution and used Boole's inequality. In the fourth line we used Eq. (7.4), (7.5), and (7.10). In the sixth line we used Eq. (7.10) again. Next we used the inductive hypothesis for $(\mathcal{S}_2)$ and Boole's inequality in the last two lines, which completes the induction for $(\mathcal{S}_1)$.

To prove $(\mathcal{S}_2)$ for $k+1$ we have

$$
\begin{aligned}
&\Big(A^{(k)}(s) + T_k^L \otimes S_{k+1}^L(s, u_{k+1})\Big)^{\mathbf{X}_{k+1}}\\
&= \left(A^{(k)}(s) + \inf_{s\le u_k\le u_{k+1}} \big\{T_k^L(s,u_k) + S_{k+1}^L(u_k, u_{k+1})\big\}\right)^{\mathbf{X}_{k+1}}\\
&= \inf_{s\le u_k\le u_{k+1}} \Big(A^{(k)}(s) + T_k^L(s,u_k) + S_{k+1}^L(u_k, u_{k+1})\Big)^{\mathbf{X}_{k+1}}\\
&= \inf_{s\le u_k\le u_{k+1}} \left(\sum_{i=1}^{m_s^{(k-1)}} l_{k-1,i}X_{k,i} + \sum_{i=m_s^{(k-1)}}^{N_{u_k}} l_{k-1,i}X_{k,i} + S_{k+1}^L(u_k, u_{k+1})\right)^{\mathbf{X}_{k+1}}\\
&= \inf_{s\le u_k\le u_{k+1}} \left(\sum_{i=1}^{N_{u_k}} l_{k-1,i}X_{k,i} + S_{k+1}^L(u_k, u_{k+1})\right)^{\mathbf{X}_{k+1}}\\
&= \inf_{s\le u_k\le u_{k+1}} \left(\Big(A^{(k-1)}(s) + T_{k-1}^L \otimes S_k^L(s,u_k)\Big)^{\mathbf{X}_k} + S_{k+1}^L(u_k, u_{k+1})\right)^{\mathbf{X}_{k+1}}\\
&= \inf_{s\le u_k\le u_{k+1}} \left(\inf_{s\le u_1\le\cdots\le u_k} U_k(s,u_k) + S_{k+1}^L(u_k, u_{k+1})\right)^{\mathbf{X}_{k+1}}
\end{aligned}
$$

$$= \inf_{s \le u_1 \le \cdots \le u_{k+1}} U_{k+1}(s, u_{k+1}) \; .$$

In the sixth line we used Eq. (7.10). In the seventh line we used the induction hypothesis. In the last line we used the definition of U_k.

Next, we use the statement $(\mathcal{S}_1)$ to compute the end-to-end delay bound on $W_n(t)$ for $k = n$. We have

$$\begin{aligned}
& Pr(W_n(t) > d) \\
\le & \sum_{0 \le s \le t} \sum_{s \le u_1 \le \cdots \le u_{n-1} \le t+d} Pr\Bigg(A^{(n-1)}(t) > \Bigg(\cdots \\
& \left((A(s) + S_1^L(s, u_1))^{\mathbf{X}_1} + S_2^L(u_1, u_2) \right)^{\mathbf{X}_2} + \cdots \\
& + S_{n-1}^L(u_{n-2}, u_{n-1}) \Bigg)^{\mathbf{X}_{n-1}} + S_n^L(u_{n-1}, t+d) \Bigg) \\
\le & \sum_{0 \le s \le t} \sum_{s \le u_1 \le \cdots \le u_{n-1} \le t+d} Pr\Bigg(\Bigg(\cdots \left((A(t-s) - \right. \\
& \left. S_1^L(s, u_1))^{\mathbf{X}_1} - S_2^L(u_1, u_2) \right)^{\mathbf{X}_2} - \cdots - \\
& S_{n-1}^L(u_{n-2}, u_{n-1}) \Bigg)^{\mathbf{X}_{n-1}} > S_n^L(u_{n-1}, t+d) \Bigg) \\
\le & \sum_{0 \le s \le t} \sum_{s \le u_1 \le \cdots \le u_{n-1} \le t+d} e^{-\theta_n S_n^L(u_{n-1}, t+d)} \; \cdot \\
& E\left[e^{\theta_n \left(\cdots \left((A(t-s) - S_1^L(s,u_1))^{\mathbf{X}_1} - S_2^L(u_1,u_2) \right)^{\mathbf{X}_2} - \cdots - S_{n-1}^L(u_{n-2},u_{n-1}) \right)^{\mathbf{X}_{n-1}}} \right] \\
\le & \sum_{0 \le s \le t} \sum_{s \le u_1 \le \cdots \le u_{n-1} \le t+d} e^{-\theta_n S_n^L(u_{n-1}, t+d)} \\
& e^{-\theta_{n-1} S_{n-1}^L(u_{n-2}, u_{n-1})} \cdots e^{-\theta_1 S_1^L(s, u_1)} e^{\theta_1 r_A(\theta_1)(s,t)} \; .
\end{aligned}$$

In the second line we expanded the recursion in the statement $(\mathcal{S}_1)$. In the third line we repeatedly applied the stationarity bound from Lemma 9. In the fourth line we used Chernoff's bound for some $\theta_n > 0$. In the fifth line we recursively applied Lemma 10. To do so, we let $\theta_i R_{i-1}(\theta_i) = \log M_l(\theta_{i-1})$, which is already stated in Lemma 10, $R_{i-1}(\theta_i) = \frac{1}{\theta_i} \log M_{\mathbf{X}_{i-1}}(\log M_l(\theta_i))$. Here, all S_i^L's are packetized dynamic servers in the form of Eq. (7.2). Note

that, if we let

$$b = \sup\left\{e^{-\theta_n C_n}, e^{-\theta_{n-1} C_{n-1}}, \ldots, e^{-\theta_1 C_1}\right\} , \tag{7.11}$$

we have

$$\begin{aligned} & Pr(W_n(t) > d) \\ \leq & \sum_{0 \leq s \leq t} e^{d \cdot \log b} e^{\sum_{i=1}^{n} \theta_i l_{max}} e^{(\log b + \theta_1 r_A(\theta_1))(t-s)} \cdot \sum_{s \leq u_1 \leq \cdots \leq u_{n-1} \leq t+d} 1 \\ \leq & \; b^d e^{\sum_{i=1}^{n} \theta_i l_{max}} K^n . \end{aligned}$$

Here we let $K = \frac{\left(1+\frac{d}{n}\right)^{1+\frac{d}{n}}}{\left(\frac{d}{n}\right)^{\frac{d}{n}}}$ and used $\log b + \theta_1 r_A(\theta_1) < 0$ as the stability condition. Taking $t \to \infty$ proves the result. We used the same argument as in Section 4.3.1 for the last step of computation. Finally, the order of growth of the ε-quantiles for $0 < \varepsilon < 1$ follows directly as $\mathcal{O}(n)$. □

7.3 Numerical Evaluation

To evaluate the analytical results, we use the following numerical example settings. First, we let the packet sizes be discrete uniformly distributed *i.i.d. r.v.*'s, $l \sim U[a, b]$. Thus, we know $M_l(\theta) = \frac{e^{a\theta} - e^{(b+1)\theta}}{(b-a+1)(1-e^{\theta})}$. Let $a = 1, b = 16$ for illustration. Clearly, $l_{max} = 16$. Next, we use the Bernoulli process as the scaling process - $\mathbf{X} \sim B(p)$, where p represents the data through probability, so that we know $R(\theta) = \frac{1}{\theta}\log(1 - p + pM_l(\theta))$. Further we assume that all servers are work-conserving with constant bit rate C_i. Next, we first compare the delay bounds from Section 7.2.1 with those from Section 7.2.2 ($\rightarrow$ Theorem 12) and also validate them against simulation results. Then we evaluate our main result from Theorem 12 changing the scaling parameters.

For the first comparison we assume that the arrivals are a compound process instead of being packetized by a packetizer before being served. Note, our results in Theorem 12 also imply this case, since the MGF bound of the arrival process that the theorem requires can be given directly. Without loss of applicability in real-world, we assume $A(t)$ is a compound Poisson process, so that $r_A(\theta) = \frac{1}{\theta}\lambda(M_l(\theta) - 1)$. The average rate of the Poisson process $N(t)$ is normalized to one data unit (bit) per one time unit, i.e., $\lambda = 1$. The number of the scaling elements varies from 1 to 9, which means maximal 10 servers. We assume the utilization of the first server is 0.8, so

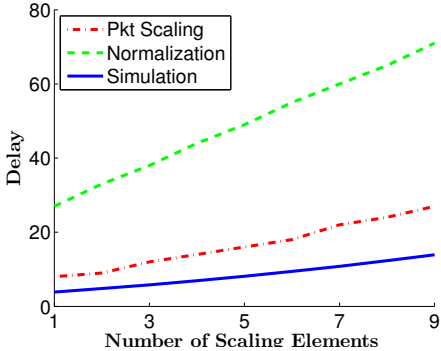


Figure 7.5: Delay bounds with Theorem 12 (concretely consider packet scaling), "normalized" flow, and simulations.

$C_1 = 1.25$. To choose $C_2, \ldots, C_{10}$, we refer to Eq. (7.11). Avoiding that some server becomes the bottleneck, we can let all the terms in Eq. (7.11) be equal, i.e., $\theta_i C_i = \theta_{i-1} C_{i-1}, 2 \leq i \leq n$, where θ_i's are implied in Lemma 10. This is actually a criterion to assign the service capacities along the path a flow traverses. It must not be so strict, or in other words, the service capacities in practice may already be set before we know the other network settings. So here, for simplicity, we just statically set the capacities as $C_2 \ldots C_{10} = [1.15, 1.05, 0.95, 0.85, 0.80, 0.75, 0.70, 0.65, 0.60]$. The quantile ε is set to 10^{-3}. We use Omnet++ to do the simulations. We measure 10^6 packet delays at the destination node and use the empirical quantile from these for the simulation results. This will increase the result accuracy so that we ignore the confidence intervals.

Figure 7.5 shows the bounds on the 10^{-3}-quantiles of the delay. The plot shows the $\mathcal{O}(n)$ order of growth. We observe that the results from Theorem 12 are much closer to the simulation results than the results from analyzing the normalized flow. The mathematical reason is that, although with both methods we used the maximum packet size l_{max}, in Theorem 12 we used the form of $[C_i \cdot t - l_{max}]_+$, while for the normalization we used the form of $C_i / l_{max} \cdot t$. Obviously, the loss in precision caused by the division is higher than for subtraction. The gap to the simulation results implies that the tightness still can be improved. Yet, as this is the first attempt to model the variable length packet flow transformation, we focused on the expression of such a network scenario and provided the first insights calculate delay bounds in this setting. The key to improve on the tightness will be to make smarter

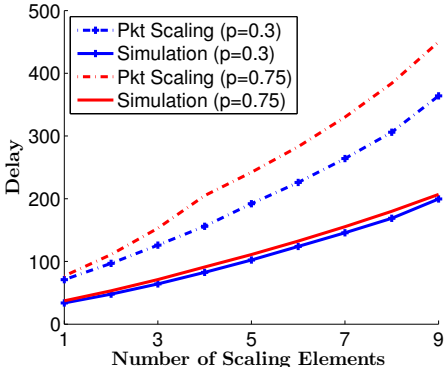


Figure 7.6: Delay bounds considering packet scaling and with simulations.

usage of the packet length distribution, than just resorting to l_{max}. On the other hand, as we can also see in [95, 36], it can circumvent several technical difficulties, otherwise we would have to consider the inherent correlations among arrivals, services and packet scaling elements, which is, however, as we discussed in previous sections or in [100, 80], very difficult even in the single node case without flow transformations. Furthermore, the usage of Boole's inequality could be improved by the construction of a martingale as in [118]. Yet, again this is, so far only possible for the single node case. So, we leave this for future work.

For the second comparison we slightly change the arrival description. Frequently we only know the statistical properties of the bit flow and that the bits are packetized. The result from Theorem 12 can also deal with this. So we use a bit flow followed by a packetizer as the arrival for the server. Assume that the original arrival flow of bits is a Poisson process $Poi(\lambda)$. Then we know $r_A(\theta)(s,t) \leq \frac{\lambda(e^{\theta}-1)}{\theta}(t-s) + l_{max}$. The other numerical settings we use the same as before.

Figure 7.6 shows the bounds on the 10^{-3}-quantiles of the delay under varying scaling parameters. We can see that Theorem 12 increases with the through probability p. That means if more of the flow is kept during the transformation, the higher the burstiness at the next server node will become. Interestingly, the gap between those curves from the theorem is larger than that of the simulation results. The reason is that we use l_{max}/C as the extra latency for each packet after being served by the packetized server, while actually most packets have a much smaller latency increase. This treatment en-

larges the sensitivity of the results, because the more the flow passes through, the more tightness we lose.

In this chapter, we extended network calculus to model networks with variable length packet flow transformations. The main contribution is the definition of a scaling element that works on the packet level (rather than the bit level). This facilitates a commutation of the service element with the scaling element on the packet level, and thus preserves the convolution-form expression of this kind of networks. Based on this we derived the end-to-end delay bounds. We also discussed another method, which is a direct extension of a previous model by normalizing the bit flow and the bit-wise service with the packet sizes, as if the flow was treated as a flow with identical data units and the service rate was in $packets/s$. We evaluated both methods and validated them against simulations. We found that the method based on the new packet scaling element is much closer to the simulation results than the other one. However, we also point out that improving the tightness is still a challenge for future work. We hope to achieve this by finding a more precise expression for the dynamic server of the packetized service.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this thesis, we have developed the stochastic network calculus for modeling and analyzing the networks with flow transformations. The main contribution is that even the flow is transformed, in other words, the assumption that the system is lossless for defining the virtual delay does not hold anymore, we can still preserve the fundamental result of the network calculus - convolution-form expression of the network and provide competitive end-to-end performance bounds.

We first theoretically introduced two versatile stochastic data scaling elements: one is to presumably know less information on the data scaling behavior and thus capture the statistical properties with the stochastic scaling curves, the other is to assume knowing more about the scaling behavior and capture the statistical properties through its MGF bound. The stochastic models facilitate an agile and meticulous analysis for the networks with flow transformation in general, and particularly for the most important dynamic demultiplexing case. Through using the equivalent systems or commuting the servers and scalers, we can express the network service in $(min, +)$ convolution-form. As a consequence, the fundamental scaling properties of the end-to-end delay analysis are retained. We can see that, these models lay the theoretical foundation for a rich set of new applications of network calculus, e.g., lossy networks, dynamic routing, load balancing, transcoding, SDN,

and so on.

We further stated a deconstruction viewpoint of the scaling behavior by using the demultiplexing example. Under the assumption that the scheduling is FIFO, the demultiplexing is actually multiplexing and the service one subflow receives is the leftover service. Through this deconstruction, we on one hand explained many aspects whereby the two scaling models showed ambiguity (the meaning of moving scalers or the commutation with servers) and advanced the performance analysis; on the other hand, potentially widened the scope of applying the stochastic scaling elements to the situations where the flows are simply divided or aggregated without any systematic reasons such as dynamic routing, coding, or certain control requirement.

Furthermore, we exhibited the modeling power of the stochastic scaling element by applying it to describe and analyze the unreliable links with retransmission-based loss recovery scheme. We provided a two-phase derivation of the stochastic performance bounds, i.e., firstly observe the sample-path, then consider the violation probability. The most difficult challenge lies in the recursive feedback loops, where the lost data can be lost again. We solve this fixed-point problem under practically applicable assumptions.

As another application, or, an extension, we studied the stochastic scaling element in different granularities. Since in more and more network analysis scenarios we concern cross-layer problems or face a diverse network circumstances, the scaling elements in the data and time granularities can be defined quite differently according to their flow specifications. To that end, we provided the stochastic scaling element definitions for the identical and variable length data units, and used them to model the packets demultiplexing, then derived the end-to-end delay bounds for the whole data units. We also provided the scaling element defined for a discrete time slot, which is however easy to define but rather hard to calculate.

In order to validate all these theoretical and analytical results we provided many helpful numerical examples and interesting discuss.

8.2 Future Work

Recently, there is a growing interest of analyzing the performance of a system across layers, in hybrid or self-defined circumstances. Many operations on the data flow within such heterogeneous networked systems may lead to the flow transformations and thus can be modeled and analyzed with the results that we contributed in this dissertation. Although we established a modeling framework for this challenging problem, several tasks in the list of the future

work are still need to be carefully treated, so that we have more flexibility and accuracy when applying this modeling framework to more and more sophisticated real-world applications.

The first task is to find a smarter way to model the scaling on the data arrived at each discrete time slot. The difficulty lies in that the arrivals and departures are different time sequences for the same amount of data. As the scaling is now time-sensitive, the sub-sequence in the departed flow which are scaled can not be easily identified in the arrivals, thus it is hard to do the commutation or use the deconstruction viewpoint introduced in this thesis to enable further performance derivation. In Chapter 4 we provided a transform between the scalings in time and space domains. However, the statistical properties of the space domain scaling can not be directly carried over from the time domain scaling; moreover, the arrivals are dependent of the time domain scaling, because the *timing* really connects the scaling and the instantaneous arrivals. If we desire a "clean" modeling and analysis, we need to find another way to identify the transformed part of the flow on the arrival side and describe the service it requires subject to the better stochastic performance bounds.

Another future work is to keep the convolution-form expression when modeling the networks with flow transformations under the other scheduling schemes besides FIFO. The difficulty is similar as the previous one. The identification of the transformed flow of interest is a rather hard problem.

The stochastic scaling element can model the in-network processing. But the modeling capacity still needs to be strengthened. At least considering different topologies is necessary. We can model the feed-forward, or even feedback loop for the flow of interest, whereas only a more systematic treatment of the whole network and a more elaborate analysis can indeed meet the real-world requirement. Several basic however very important and challenging problems should be solved, for example, the multiplexing of the scaling process and loosening the assumptions on the system and traffic characterizations.

Index

Bibliography

[1] J. Abate, G. L. Choudhury, and W. Whitt. Waiting-time tail probabilities in queues with long-tail service-time distributions. *Queueing Systems*, 16(3-4):311–338, September 1994.

[2] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan. Performance bounds for guaranteed and adaptive services. Technical report, IBM Research, RC 20649, 1996.

[3] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan. Performance bounds for flow control protocols. *IEEE/ACM Transactions on Networking*, 7(3):310–323, June 1999.

[4] I. F. Akyildiz. Exact product form solution for queueing networks with blocking. *IEEE Transactions on Computers*, 36(1):122–125, 1987.

[5] H. Al-Zubaidy, J. Liebeherr, and A. Burchard. A (min,x) network calculus for multi-hop fading channels. In *Proceedings of IEEE INFOCOM*, pages 1833–1841, April 2013.

[6] S. Ayyorgun and R. L. Cruz. A service-curve model with loss and a multiplexing problem. In *Proceedings of ICDCS*, pages 756–765, 2004.

[7] S. Ayyorgun and W. Feng. A probabilistic definition of burstiness characterization. Technical Report LA-UR 03-3668, Los Alamos National Laboratory, May 2003.

[8] S. Ayyorgun and W.-C. Feng. A systematic approach for providing end-to-end probabilistic qos guarantees. In *Proceedings of the 13th IEEE International Conference on Computer Communications and Networks (ICCCN), Chicago, IL*, October 2004.

[9] S. Azodolmolky, R. Nejabati, M. Pazouki, P. Wieder, R. Yahyapour, and D. Simeonidou. An analytical model for software defined networking: A network calculus-based approach. In *Proceedings of IEEE GLOBECOM*, pages 1397–1402, December 2013.

[10] S. Balsamo and V. De Nitto. A survey of product-form queueing networks with blocking and their equivalences. *Annals of Operations Research*, 48:31–61, 1994.

[11] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Open, closed and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, April 1975.

[12] M. Beck, S. Henningsen, S. Birnbach, and J. Schmitt. Towards a statistical network calculus - dealing with uncertainty in arrivals. In *Proceedings of IEEE INFOCOM*, pages 2382–2390, April 2014.

[13] M. Beck and J. Schmitt. On the calculation of sample-path backlog bounds in queueing systems over finite time horizons. In *Proceedings of ICST VALUETOOLS*, pages 148–157, October 2012.

[14] M. Beck and J. Schmitt. The DISCO stochastic network calculator version 1.0: When waiting comes to an end. In *Proceedings of ICST VALUETOOLS*, pages 282–285. ICST, 2013.

[15] D. Bertsekas and R. Gallager. *Data Networks*. PRENTICE HALL, 1992.

[16] P. Billingsley. *Probability and Measure (3^{rd} Edition)*. Wiley, 1995.

[17] S. Bondorf and J. Schmitt. Statistical response time bounds in randomly deployed wireless sensor networks. In *Proceedings of IEEE LCN*, pages 340–343, October 2010.

[18] S. Bondorf and J. Schmitt. The DiscoDNC V2: A comprehensive tool for deterministic network calculus. In *Proceedings of ICST VALUETOOLS*, pages 44–49. ICST, 2014.

[19] R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn. Effective envelopes: Statistical bounds on multiplexed traffic in packet networks. In *Proceedings of IEEE INFOCOM*, March 2000. Also:Statistical Multiplexing Gain of Link Scheduling Algorithms in QoS Networks -Short Version, University of Virginia, Department of Computer Science, CS-99-23, July 1999.

[20] R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn. Statistical service assurances for traffic scheduling algorithms. *IEEE Journal on Selected Areas in Communications*, 18(12):2651–2664, December 2000.

[21] D. D. Botvich and N. G. Duffield. Large deviations, economies of scale, and the shape of the loss curve in large multiplexers. *Queueing Systems*, 20:293–320, 1995.

[22] R. J. Boucherie and N. M. Van Dijk. Product forms for queueing networks with state-dependent multiple job transitions. *Advances in Applied Probability*, 23(1):152–187, March 1991.

[23] A. Bouillard, N. Farhi, and B. Gaujal. Packetization and packet curves in network calculus. In *Proceedings of ICST VALUETOOLS*, pages 136–137, October 2012.

[24] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. RFC 1633, June 1994.

[25] E. Brockmeyer, H.L. Hallstrom, and A. Jensen. The life and works of A. K. Erlang. *Transactions of the Danish Academy of Technical Sciences*, (2), 1948.

[26] A. Burchard, J. Liebeherr, and F. Ciucu. On theta(h log h) scaling of network delays. In *Proc. IEEE INFOCOM*, March 2007.

[27] A. Burchard, J. Liebeherr, and F. Ciucu. On $\Theta\left(H \log H\right)$ scaling of network delays. In *Proceedings of IEEE INFOCOM*, May 2007.

[28] A. Burchard, J. Liebeherr, and S. D. Patek. A min-plus calculus for end-to-end statistical service guarantees. *IEEE Transactions on Information Theory*, 52(9):4105–4114, September 2006.

[29] A. Burchard, J. Liebeherr, and S. D. Patek. A min-plus calculus for end-to-end statistical service guarantees. *IEEE Transactions on Information Theory*, 52(9):4105 – 4114, September 2006.

[30] J. P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, 16(9):527–531, September 1973.

[31] S. Chakraborty, S. Künzli, L. Thiele, A. Herkersdorf, and P. Sagmeister. Performance evaluation of network processor architectures:

Combining simulation with analytical estimation. *Computer Networks*, 42(5):641–665, April 2003.

[32] C.-S. Chang. Stability, queue length and delay, Part II: Stochastic queueing networks. In *Proceedings of the 31st IEEE Conference on Decision and Control*, pages 1005–1010, December 1992.

[33] C.-S. Chang. Stability, queue length and delay of deterministic and stochastic queueing networks. *IEEE Transactions on Automatic Control*, 39(5):913–931, May 1994.

[34] C.-S. Chang. Stability, queue length, and delay of deterministic and stochastic queueing networks. *IEEE Transactions on Automatic Control*, 39(5):913–931, May 1994.

[35] C.-S. Chang. On deterministic traffic regulation and service guarantees: A systematic approach by filtering. *IEEE Transactions on Information Theory*, 44(3):1097–1110, May 1998.

[36] C.-S. Chang. *Performance Guarantees in Communication Networks*. Springer-Verlag, 2000.

[37] C.-S. Chang and R. L. Cruz. A time varying filtering theory for constrained traffic regulation and dynamic service guarantees. In *Proceedings of IEEE INFOCOM 1999*, pages 63–70, New York, 1999.

[38] G. Choudhury, D. Lucantoni, and W. Whitt. Squeezing the most out of ATM. *IEEE Transactions on Communications*, 44(2):203–217, February 1996.

[39] F. Ciucu. Network calculus delay bounds in queueing networks with exact solutions. In *Proc. Proceedings International Teletraffic Congress*, 2007.

[40] F. Ciucu. *Scaling Properties in the Stochastic Network Calculus*. PhD thesis. University of Virginia, 2007.

[41] F. Ciucu. End-to-end delay analysis for networks with partial assumptions of statistical independence. In *Proceedings of ICST VALUETOOLS*, 2009.

[42] F. Ciucu, A. Burchard, and J. Liebeherr. A network service curve approach for the stochastic analysis of networks. In *ACM SIGMETRICS*, volume 33, pages 279–290, 2005.

[43] F. Ciucu, A. Burchard, and J. Liebeherr. Scaling properties of statistical end-to-end bounds in the network calculus. *IEEE Transactions on Information Theory*, 52(6):2300–2312, June 2006.

[44] F. Ciucu and O. Hohlfeld. Scaling of buffer and capacity requirements fo voice traffic in packet networks. In *Proceedings of the International Teletraffic Congress (ITC)*, 2009.

[45] F. Ciucu, O. Hohlfeld, and P. Hui. Non-asymptotic throughput and delay distributions in multi-hop wireless networks. In *The 48th Annual Allerton Conference on Communication, Control, and Computing*, 2010.

[46] F. Ciucu and J. Liebeherr. A case for decomposition of FIFO networks. In *Proceedings of IEEE INFOCOM*, pages 1071–1079, April 2009.

[47] F. Ciucu, F. Poloczek, and J. Schmitt. Sharp per-flow delay bounds for bursty arrivals: The case of FIFO, SP, and EDF scheduling. In *Proceedings of IEEE INFOCOM*, pages 1896–1904, April 2014.

[48] F. Ciucu and J. Schmitt. Perspectives on network calculus: No free lunch, but still good value. *SIGCOMM Computer Communication Review*, 42(4):311–322, August 2012.

[49] F. Ciucu, J. Schmitt, and H. Wang. On expressing networks with flow transformation in convolution-form. In *Proceedings of IEEE INFOCOM*, pages 1979–1987, April 2011.

[50] R. L. Cruz. A calculus for network delay, Part I: Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[51] R. L. Cruz. A calculus for network delay, Part II: Network analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991.

[52] R. L. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE Journal on Selected Areas in Communications*, 13(6):1048–1056, August 1995.

[53] R. L. Cruz. Quality of service management in integrated services networks. In *Proceedings of the 1st Semi-Annual Research Review, Center for Wireless Communications*, 1996.

[54] R. L. Cruz. SCED+: Efficient management of quality of service guarantees. In *Proc. IEEE INFOCOM*, volume 2, pages 625–634, March 1998.

[55] R. L. Cruz and M. Taneja. An analysis of traffic clipping. In *Conference of Information Sciences and Systems*, March 1998.

[56] J. N. Daigle and J. D. Langford. Models for analysis of packet voice communications systems. *IEEE Journal on Selected Areas in Communications*, 4(6):847–855, September 1986.

[57] N. G. Duffield and N. O'Connell. Large deviations and overflow probabilities for the general single-server queue, with applications. *Mathematical Proceedings of the Cambridge Philosophical Society*, 118(2):363–374, September 1995.

[58] E. O. Elliott. Estimates of error rates for codes on bursty-noise channels. *Bell System Technical Journal*, 42(9):1977–1997, September 1963.

[59] A. K. Erlang. The theory of probabilities and telephone conversations. *Nyt Tidsskrift for Matematik B*, 20:33–39, 1909 (in Danish).

[60] A. K. Erlang. Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *Elektrotkeknikeren*, 13, 1917 (in Danish).

[61] M. Fidler. An end-to-end probabilistic network calculus with moment generating functions. In *Proceedings of IEEE IWQoS*, pages 261–270, June 2006.

[62] M. Fidler. A network calculus approach to probabilistic quality of service analysis of fading channels. In *Proceedings of IEEE Globecom*, Nov 2006.

[63] M. Fidler. A survey of deterministic and stochastic service curve models in the network calculus. *IEEE Communications Surveys and Tutorials*, 12(1), 2010.

[64] M. Fidler and J. Schmitt. On the way to a distributed systems calculus: An end-to-end network calculus with data scaling. In *Proceedings of ACM SIGMETRICS/Performance*, pages 287–298, 2006.

[65] E. Gelenbe. Product form networks with negative and positive customers. *Journal of Applied Probability*, 28(3):656–663, 1991.

[66] R. J. Gibbens and P. J. Hunt. Effective bandwidths for the multi-type UAS channel. *Queueing Systems*, 9(1-2):17–28, September 1991.

[67] E. N. Gilbert. Capacity of a bursty-noise channel. *Bell System Technical Journal*, 39(5):1253–1265, September 1960.

[68] N. Gollan and J. Schmitt. Energy-efficent tdma design under real-time constraints in wireless sensor networks. In *Proceedings of MASCOTS*, pages 80–87. Springer, LNCS 3560, October 2007.

[69] N. Gollan, F. Zdarsky, I. Martinovic, and J. Schmitt. The disco network calculator. In *Proceedings of GI/ITG MMB*, pages 1–3, March 2008.

[70] W. J. Gordon and G. F. Newell. Closed queueing systems with exponential servers. *Operations Research*, 15(2):254–265, August 1967.

[71] R. Guérin, H. Ahmadi, and M. Naghshineh. Equivalent capacity and its application to bandwidth allocation in high-speed networks. *IEEE Journal on Selected Areas in Communications*, 9(7):968–981, September 1991.

[72] H. Heffes and D. M. Lucantoni. A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. *IEEE Journal on Selected Areas in Communications*, 4(6):856–867, September 1986.

[73] W. Henderson and P. G. Taylor. Product form in networks of queues with batch arrivals and batch services. *Queueing Syst. Theory Appl.*, 6(1):71–87, 1990.

[74] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst. System level performance analysis - the SymTA/S approach. In *Proceedings of Computers and Digital Techniques*, pages 148–168, March 2005.

[75] R. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[76] J. Y. Hui. Resource allocation for broadband networks. *IEEE Journal of Selected Areas in Communications*, 6(9):1598–1608, December 1988.

[77] J. R. Jackson. Networks of waiting lines. *Operations Research*, (5):518–521, August 1957.

[78] R. Jain and I. Chlamtac. The P-Square algorithm for dynamic calculation of percentiles and histograms without storing observations. *Communications of the ACM*, pages 1076–1085, October 1985.

[79] Y. Jiang. A basic stochastic network calculus. *ACM Computer Communication Review*, 36(4):123–134, October 2006.

[80] Y. Jiang. Stochastic service curve and delay bound analysis: A single node case. In *Proceedings of the 25th International Teletraffic Congress (ITC 25)*, September 2013.

[81] Y. Jiang and P. J. Emstad. Analysis of stochastic service guarantees in communication networks: A server model. In *Proc. IFIP IWQoS*, pages 233–245, June 2005.

[82] Y. Jiang and Y. Liu. *Stochastic Network Calculus*. Springer-Verlag, 2008.

[83] U. Herzog K. M. Chandy and L. Woo. Parametric analysis of queuing networks. *IBM Journal of Research and Development*, 19(1):36–42, January 1979.

[84] F. P. Kelly. *Reversibility and stochastic networks*. John Wiley & Sons Ltd., 1979.

[85] F. P. Kelly. Loss networks. *The Annals of Applied Probability*, 1(3):319–378, August 1991.

[86] F. P. Kelly. Notes on effective bandwidths. In F. P. Kelly, S. Zachary, and I. Ziedins, editors, *Stochastic Networks: Theory and Applications*, number 4 in Royal Statistical Society Lecture Notes, pages 141–168. Oxford University Press, 1996.

[87] I. Keslassy, C.-S. Chang, N. McKeown, and D.-S. Lee. Optimal load-balancing. In *Proc. IEEE INFOCOM*, March 2005.

[88] H. Kim and J. C. Hou. Network calculus based simulation: theorems, implementation, and evaluation. In *Proceedings of IEEE INFOCOM*, March 2004.

[89] L. Kleinrock. *Communication Nets : Stochastic Message Flow and Delay*. McGraw-Hill, Inc., 1964.

[90] E. W. Knightly and H. Zhang. D-BIND: an accurate traffic model for providing QoS guarantees to VBR traffic. *IEEE/ACM Transactions on Networking*, 5(2):219–231, April 1997.

[91] A. Koubaa, M. Alves, and E. Tovar. Modeling and worst-case dimensioning of cluster-tree wireless sensor networks. In *Proceedings of the 27th IEEE International Real-Time Systems Symposium*, pages 412–421, December 2006.

[92] J. Kurose. On computing per-session performance bounds in high-speed multi-hop computer networks. In *Proceedings of ACM Sigmetrics*, pages 128–139, June 1992.

[93] S. S. Lavenberg and M. Reiser. Stationary state probabilities at arrival instants for closed queueing networks with multiple types of customers. *Journal of Applied Probability*, 17(4):1048–1061, December 1980.

[94] J.-Y. Le Boudec. Application of network calculus to guaranteed service networks. *IEEE Transactions on Information Theory*, 44(3):1087–1096, May 1998.

[95] J.-Y. Le Boudec and P. Thiran. *Network Calculus A Theory of Deterministic Queuing Systems for the Internet*. Number 2050 in Lecture Notes in Computer Science. Springer-Verlag, 2001.

[96] J.-Y. Le Boudec and M. Vojnović. Elements of probabilistic network calculus for packet scale rate guarantee nodes. In *Proc. of 15th Int'l Symp. of Mathematical Theory of Networks and Systems*.

[97] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of Ethernet traffic. *IEEE/ACM Transactions on Networking*, 2(1):1–15, February 1994.

[98] C. Li, A. Burchard, and J. Liebeherr. A network calculus with effective bandwidth. *IEEE/ACM Transactions on Networking*, 15(6):1063–6692, December 2007.

[99] J. Liebeherr, A. Burchard, and F. Ciucu. Non-asymptotic delay bounds for networks with heavy-tailed traffic. In *Proceedings of IEEE INFOCOM*, pages 1–9, March 2010.

[100] J. Liebeherr, A. Burchard, and F. Ciucu. Delay bounds in communication networks with heavy-tailed and self-similar traffic. *IEEE Transactions on Information Theory*, 58(2):1010–1024, February 2012.

[101] J. Liebeherr, Y. Ghiassi-Farrokhfal, and A. Burchard. Does link scheduling matter on long paths? In *Proceedings of IEEE INFOCOM*, pages 199–208, Genova, June 2010.

[102] J. Liebeherr, Y. Ghiassi-Farrokhfal, and A. Burchard. On the impact of link scheduling on end-to-end delays in large networks. *IEEE Journal on Selected Areas in Communications*, 29(5):1009–1020, May 2011.

[103] D. V. Lindley. The theory of queues with a single server. volume 48, pages 277–289, April 1952.

[104] J. D. C. Little. A Proof for the Queuing Formula: $L = \lambda W$. *Operations Research*, 9(3):383–387, June 1961.

[105] R. Lübben, M. Fidler, and J. Liebeherr. Stochastic bandwidth estimation in networks with random service. *IEEE/ACM Transactions on Networking*, 22(2):484–497, April 2014.

[106] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. D. Robbins. Performance models of statistical multiplexing in packet video communications. *IEEE Transactions on Communications*, 36(7):834 – 844, July 1988.

[107] S. Mao and S. S. Panwar. A survey of envelope processes and their applications in quality of service provisioning. *IEEE Communications Surveys & Tutorials*, 8(3):2–20, 3rd Quarter 2006.

[108] L. Massoulié and A. Simonian. Large buffer asymptotics for the queue with fractional Brownian input. *Journal of Applied Probabability*, 36(3):894–906, September 1999.

[109] A. Maxiaguine, S. Künzli, and L. Thiele. Workload characterization model for tasks with variable execution demand. In *Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, volume 2, pages 1040–1045, February 2004.

[110] A. Maxiaguine, S. Künzli, and L. Thiele. Workload characterization model for tasks with variable execution demand. In *Design Automation and Test in Europe (DATE)*, pages 1040–1045, Paris, France, Feb 2004. IEEE Computer Society.

[111] R. R. Mazumdar. *Performance Modeling, Loss Networks, and Statistical Multiplexing*. Synthesis Lectures on Communication Networks. Morgan & Claypool Publishers, 2009.

[112] T. Mikosch, S. Resnick, H. Rootzén, and A. Stegeman. Is network traffic approximated by stable Lévy motion or fractional Brownian motion? *Annals of Applied Probability*, 12(1):23–68, February 2002.

[113] I. Norros. On the use of fractional Brownian motion in the theory of connectionless networks. *IEEE Journal on Selected Areas in Communications*, 13(6):953–962, August 1995.

[114] K. A. Nuaimi, N. Mohamed, and J. Al-Jaroodi. A survey of load balancing in cloud computing: Challenges and algorithms. In *Proceedings of Network Cloud Computing and Applications (NCCA)*, pages 137–142. IEEE, dec 2012.

[115] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control - the single node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[116] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.

[117] V. Paxson and S. Floyd. Wide area traffic: The failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(5):226–244, June 1995.

[118] F. Poloczek and F. Ciucu. Scheduling analysis with martingales. *Performance Evaluation*, 79:56–72, September 2014.

[119] Martin Reiser and Stephen S. Lavenberg. Mean-value analysis of closed multichain queuing networks. *Journal of the ACM*, 27(2):313–322, April 1980.

[120] A. Rizk and M. Fidler. Sample path bounds for long memory FBM traffic. In *Proceedings of IEEE INFOCOM*, pages 1–5, March 2010.

[121] A. Rizk and M. Fidler. Statistical end-to-end performance bounds for networks under long memory FBM cross traffic. In *Proceedings of IEEE IWQoS*, pages 1–9, June 2010.

[122] H. Sariowan, R. L. Cruz, and G. C. Polyzos. Scheduling for quality of service guarantees via service curves. In *Proceedings of IEEE ICCCN*, pages 512–520, September 1995.

[123] H. Schioler, J. Jessen, J. D. Nielsen, and K. G. Larsen. Network calculus for real time analysis of embedded systems with cyclic task dependencies. In *Proceedings of CATA*, pages 326–332, March 2005.

[124] H. Schioler, H. P. Schwefel, and M. B. Hansen. CyNC: A MATLAB/SimuLink toolbox for network calculus. In *Proceedings of ICST VALUETOOLS*, pages 60:1–60:10. ICST, 2007.

[125] J. Schmitt. On average and worst case behaviour in non-preemptive priority queueing. In *Proceedings of SPECTS*, 2003.

[126] J. Schmitt, N. Gollan, S. Bondorf, and I. Martinovic. Pay bursts only once holds for (some) non-FIFO systems. In *Proceedings of IEEE INFOCOM*, April 2011.

[127] J. Schmitt and U. Roedig. Sensor Network Calculus - A Framework for Worst Case Analysis. In *Proceedings of IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS'05), Marina del Rey, USA*, pages 141–154. Springer, LNCS 3560, June 2005. ISBN 3-540-26422-1.

[128] J. Schmitt, H. Wang, and I. Martinovic. A self-adversarial approach to delay analysis under arbitrary scheduling. In *Proceedings of the 4th International Symposium On Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2010)*. Springer Verlag, October 2010.

[129] J. Schmitt and F. Zdarsky. The DISCO Network Calculator - a toolbox for worst case analysis. In *Proceedings of ICST VALUETOOLS*, November 2006.

[130] J. Schmitt, F. Zdarsky, and M. Fidler. Delay bounds under arbitrary aggregate multiplexing: When network calculus leaves you in the lurch... In *Proceedings of IEEE INFOCOM*, April 2008.

[131] J. Schmitt, F. Zdarsky, and I. Martinovic. Improving performance bounds in feed-forward networks by paying multiplexing only once. In *Proceedings of GI/ITG MMB*, March 2008.

[132] J. Schmitt, F. Zdarsky, and U. Roedig. Sensor Network Calculus with Multiple Sinks. In *Proceedings of IFIP Networking 2006, Workshop on Performance Control in Wireless Sensor Networks, Coimbra, Portugal*, pages 6–13. Springer LNCS, May 2006. ISBN 972-95988-5-1.

[133] J. Schmitt, F. Zdarsky, and L. Thiele. A Comprehensive Worst-Case Calculus for Wireless Sensor Networks with In-Network Processing. In *IEEE Real-Time Systems Symposium (RTSS'07)*, 2007.

[134] K. C. Sevcik and I. Mitrani. The distribution of queuing network states at input and output instants. *Journal of the ACM*, 28(2):358–371, April 1981.

[135] F. B. Shepherd and P. J. Winzer. Selective randomized load balancing and mesh networks with changing demands. *J. Opt. Netw.*, 5:320–339, 2006.

[136] N. B. Shroff and M. Schwartz. Improved loss calculations at an ATM multiplexer. *IEEE/ACM Transactions on Networking*, 6(4):411–421, August 1998.

[137] T. Skeie, S. Johannessen, and O. Holmeide. Timeliness of real-time IP communication in switched industrial ethernet networks. *IEEE Transactions on Industrial Informatics*, 2(1):25–39, February 2006.

[138] D. Starobinski and M. Sidi. Stochastically bounded burstiness for communication networks. *IEEE Transactions on Information Theory*, 46(1):206–212, January 2000.

[139] L. Thiele, S. Chakraborty, and M. Naedele. Real-time calculus for scheduling hard real-time systems. In *IEEE International Symposium on Circuits and Systems*, volume 4, pages 101–104, 2000.

[140] D. Towsley. Queuing network models with state-dependent routing. *Journal of the ACM*, 27(2):323–337, April 1980.

[141] L. G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11(2):350–361, 1982.

[142] András Varga. OMNeT++. http://www.omnetpp.org/.

[143] E. Wandeler, A. Maxiaguine, and L. Thiele. Quantitative Characterization of Event Streams in Analysis of Hard Real-Time Applications. *Real-Time Systems*, 29(2):205–225, March 2005.

[144] E. Wandeler and L. Thiele. Real-Time Calculus (RTC) toolbox. 2006.

[145] H. Wang, F. Ciucu, and J. Schmitt. A leftover service curve approach to analyze demultiplexing in queueing networks. In *Proceedings of ICST VALUETOOLS*, pages 168–177, October 2012.

[146] H. Wang and J. Schmitt. On the way to a wireless network calculus - the single node case with retransmissions. Technical Report 375/10, University of Kaiserslautern, Germany, January 2010.

[147] H. Wang and J. Schmitt. Delay bounds calculus for variable length packet transmissions under flow transformations. Technical Report 390/14, University of Kaiserslautern, Germany, November 2014.

[148] H. Wang and J. Schmitt. A delay calculus for streaming media subject to video transcoding. *IEEE COMSOC MMTC E-Letter*, 9(2), March 2014.

[149] H. Wang and J. Schmitt. End-to-end delay bounds for variable length packet transmissions under flow transformations. In *Proceedings of ICST VALUETOOLS*, December 2014.

[150] H. Wang, J. Schmitt, and F. Ciucu. Performance modelling and analysis of unreliable links with retransmissions using network calculus. In *Proceedings of the 25th International Teletraffic Congress (ITC 25)*, September 2013.

[151] H. Wang, J. Schmitt, and I. Martinovic. Dynamic demultiplexing in network calculus – theory and application. *Performance Evaluation, Elsevier*, 68(2):201–219, Feb 2011.

[152] H. S. Wang and N. Moayeri. Finite-state Markov channel – a useful model for radio communication channels. 44(1):163–171, February 1995.

[153] K. Wang, F. Ciucu, C. Lin, and S. H. Low. A stochastic power network calculus for integrating renewable energy sources into the power grid. *IEEE Journal on Selected Areas in Communications*, 30(6):1037–1048, May 2012.

[154] K. Wu, Y. Jiang, and G. Hu. A calculus for information-driven networks. In *IEEE International Workshop on Quality of Service(IWQoS)*, pages 1–9, Jul 2009.

[155] O. Yaron and M. Sidi. Generalized processor sharing networks with exponentially bounded burstiness arrivals. *Journal of High Speed Networks*, 3.

[156] O. Yaron and M. Sidi. Performance and stability of communication networks via robust exponential bounds. *IEEE/ACM Transactions on Networking*, 1(3):372–385, June 1993.

[157] Q. Yin, Y. Jiang, S. Jiang, and P. Y. Kong. Analysis on generalized stochastically bounded bursty traffic for communication networks. In *Proceedings of IEEE Local Computer Networks (LCN)*, pages 141–149, November 2002.

[158] Y. Yuan, K. Wu, W. Jia, and Y. Jiang. Performance of acyclic stochastic networks with network coding. *IEEE Transactions on Parallel and Distributed Systems*, 22(7):1238–1245, July 2011.

[159] R. Zhang-Shen and N. McKeown. Designing a predictable internet backbone network. In *Proc. HotNets III*, November 2004.

[160] K. Zheng, F. Liu, L. Lei, C. Lin, and Y. Jiang. Stochastic performance analysis of a wireless finite-state markov channel. *IEEE/ACM Transactions on Wireless Communications*, 12(2):782–793, January 2013.

Curriculum Vitae

Personal Information

Name:	Hao Wang
Place of Birth:	Anshan, Liaoning, China
Nationality:	China

Education

08.2009 - 07.2015	Ph.D., Distributed Computer Systems Lab (DISCO) of the department of computer science, TU Kaiserslautern, Germany
10.2005 - 05.2009	M.Sc., computer science, TU Kaiserslautern, Germany. WS06/07 break for learning German
09.1998 - 07.2002	B.Eng., computer science, Northeasten University, China

Professional Experience

from 08.2009	Scientific staff member, then from 04.2012 research assistant, at DISCO Lab, TU Kaiserslautern, Germany
05.2011 - 07.2011	Internship at Telekom Innovation Laboratories (T-Labs), Berlin
06.2010 - 07.2010	Internship at Telekom Innovation Laboratories (T-Labs), Berlin
07.2002 - 07.2004	Software engineer, Neusoft Co., Ltd., Shenyang, China