On the Way to a Distributed Systems Calculus: An End-to-End Network Calculus with Data Scaling

Markus Fidler*
Centre for Quantifiable Quality of Service
NTNU Trondheim, Norway
fidler@ieee.org

Jens B. Schmitt
Distributed Computer Systems Lab
University of Kaiserslautern, Germany
jschmitt@informatik.uni-kl.de

ABSTRACT

Network calculus is a min-plus system theory which facilitates the efficient derivation of performance bounds for networks of queues. It has successfully been applied to provide end-to-end quality of service guarantees for integrated and differentiated services networks. Yet, a true end-to-end analysis including the various components of end systems as well as taking into account mid-boxes like firewalls, proxies, or media gateways has not been accomplished so far. The particular challenge posed by such systems are transformation processes, like data processing, compression, encoding, and decoding, which may alter data arrivals drastically. The heterogeneity, which is reflected in the granularity of operation, for example multimedia applications process video frames which, however, are represented by packets in the network, complicates the analysis further. To this end this paper evolves a concise network calculus with scaling functions, which allow modelling a wide variety of transformation processes. Combined with the concept of packetizer this theory enables a true end-to-end analysis of distributed systems.

Categories and Subject Descriptors

C.2.4 [**Computer System Organization**]: Computer Communication Networks—*Distributed Systems*; C.4 [**Computer System Organization**]: Performance of Systems

General Terms

Performance

Keywords

Network calculus, scaling functions, packetizers

*This work was supported in part by an Emmy Noether grant of the German Research Foundation (DFG) and by the Centre for Quantifiable Quality of Service in Communication Systems (Q2S). The Q2S Centre of Excellence is appointed by the Research Council of Norway and funded by the Research Council, NTNU, and UNINETT.

SIGMetrics/Performance'06, June 26–30, 2006, Saint Malo, France.

1. INTRODUCTION

A central aspect to networked applications is the performance of underlying communication systems. Relevant characteristics comprise capacity that has to be provisioned, buffer sizes that are required to achieve certain small loss rates, and end-to-end delays that need to be considered by real-time applications, including telephony, video broadcasting, and online gaming. Queuing theory has been widely applied to analyze such performance measures and with the advent of network architectures for quality of service it has been complemented by network calculus [4, 12] which is a deterministic framework for worst-case analysis of backlogs and delays.

Network calculus can be seen as a system theory under min-plus algebra [2] and in almost the same manner as classical system theory it features intuitive and powerful convolution formulae for analysis of single servers as well as for concatenation of tandem servers. This concatenation property is of the utmost importance since it establishes a general framework for efficient analysis of multiple node scenarios, where performance bounds exhibit an outstanding linear scaling in the number of traversed nodes [5, 12]. In contrast end-to-end performance bounds that are derived iteratively on a node-by-node basis scale quadratically.

However, network calculus and queuing theory in general apply only to forwarding of data but not to any kind of processing where data are scaled. Whenever such data scaling applies, systems need to be divided into subsystems which are at best analyzable separately. Data scaling is frequent, but often neglected since its effects are considered to be minor, for example in case of protocol overhead. Yet, there exists a variety of systems for which this assumption does not hold and which consequently cannot be analyzed holistically so far. Examples comprise encoding and decoding, mid-boxes of different types such as media-gateways, distributed multimedia systems, up to wireless sensor networks where data are aggregated inside the network.

Fig. 1 displays a video delivery and transcoding scenario where video frames are grabbed from a camera, encoded and packetized by a server and broadcast across a network. The client receives the sequence of packets, decodes it, and displays the video frames on a screen. In addition a transcoder is situated in the network which may adapt the video codec with respect to the capabilities of heterogeneous clients.

A detailed model of the system in Fig. 1 comprises numerous elements, including the camera, the bus between camera and video server, the server running the video encoder, network routers, the transcoder, and so on. Network cal-

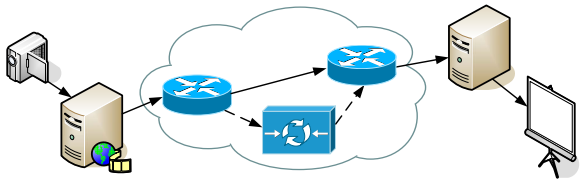

Figure 1: Video Delivery and Transcoding

culus models network elements, such as routers, using the notion of service curve which characterizes the input-output relation of a server. Service curves for a variety of schedulers have been derived. Among these are first-in first-out aggregate scheduling, rate-based schedulers, such as processor sharing, priority scheduling, as well as earliest deadline first scheduling. The general concept is, however, of much broader applicability and various other resources, for example processing units, such as the video encoder and decoder in Fig. 1, can be abstracted using service curves.

When applying known network calculus for analysis the system in Fig. 1 falls apart into a subsystem that is the part of the network upstream of the transcoder and a subsystem that is the part of the network downstream of the transcoder. In addition encoder, transcoder, and decoder have to be analyzed individually. This separation is very unfortunate since it results in weak end-to-end performance bounds, yet, it has to be applied whenever data are scaled.

Apart from data scaling the example poses a second challenge, that is the heterogeneous granularity of data units that are processed. The coders operate on video frames or slices as opposed to the network which forwards packets.

The remainder of this paper is organized as follows: Sect. 2 provides a brief background on known network calculus. In Sect. 3 we introduce the concepts of scaling functions and curves and evolve a corresponding network calculus framework, which allows for a true end-to-end analysis of systems with data scaling. Sect. 4 shows a sample scaling function for video coding. Examples for end-to-end concatenation of servers in the presence of scaling functions, which are fundamental for an efficient end-to-end analysis, are provided in Sect. 5, where we also come back to the system in Fig. 1.

2. NETWORK CALCULUS BASICS

Network calculus is a min-plus system theory for deterministic queuing systems which builds on the calculus for network delay in [6, 7] and on the work on generalized processor sharing in [14, 15]. The important concept of minimum service curve was introduced in [8, 16, 3, 11, 1] and the concept of maximum service curve in [9]. The service curve based approach facilitates the efficient analysis of tandem queues, where a linear scaling of performance bounds in the number of traversed queues is achieved as elaborated in [5] and referred to as pay bursts only once phenomenon in [12]. A detailed overview on min-plus algebra can be found in [2] and on network calculus in [4, 12]. The tightness of performance bounds is in particular shown in [12].

A basic set of functions that are extensively used by network calculus is the set of real-valued, non-negative, and wide-sense increasing functions that pass through the origin.

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

Time and data are assumed to be continuous, that is $t \in \mathbb{R}^+ = [0, \infty)$. Note that it is always possible to map a continuous model to a discrete model with $n \in \mathbb{N}_0 = \{0, 1, 2, \dots\}$ by choosing an interval Δ and sampling at $t = n\Delta$.

Let us now recall the following operations of min-plus respectively max-plus convolution and de-convolution.

Definition 2.1 (Convolution and De-Convolution) Min-plus convolution and de-convolution of two functions f and g are defined to be

$$h(t) = (f \otimes g)(t) = \inf_{s \in [0,t]} \{f(t-s) + g(s)\}$$
$$h(t) = (f \oslash g)(t) = \sup_{s \in [0,\infty)} \{f(t+s) - g(s)\}.$$

Accordingly max-plus convolution and de-convolution are

$$h(t) = (f \overline{\otimes} g)(t) = \sup_{s \in [0,t]} \{f(t-s) + g(s)\}$$
$$h(t) = (f \overline{\oslash} g)(t) = \inf_{s \in [0,\infty)} \{f(t+s) - g(s)\}.$$

Th. 2.1 recapitulates some properties of min-plus convolution which are of major importance in the sequel. The list is by far not exhaustive.

Theorem 2.1 (Properties of $\otimes$) *Let $f, g, h \in \mathcal{F}$. The following properties hold:*

1. **Closure of $\otimes$:** $(f \otimes g) \in \mathcal{F}$,
2. **Commutativity of $\otimes$:** $f \otimes g = g \otimes f$,
3. **Associativity of $\otimes$:** $(f \otimes g) \otimes h = f \otimes (g \otimes h)$,

Note that de-convolution is not closed in $\mathcal{F}$ and not commutative.

Network calculus applies the general model of arrival functions to describe traffic flows.

Definition 2.2 (Arrival Function) An arrival function is a cumulative function $F(t) \in \mathcal{F}$ that is defined to be the amount of data of a flow seen in the interval [0,t].

Based on the description of traffic flows using arrival functions performance measures can be defined.

Definition 2.3 (Backlog and Delay) Assume a server has input arrival function $F(t)$ and output arrival function $F'(t)$. The backlog at the server for time t is defined to be

$$b(t) = F(t) - F'(t).$$

Assuming first-in first-out data delivery the delay for time t is defined as

$$d(t) = \inf\{\tau \geq 0 : F(t) \leq F'(t+\tau)].$$

Arrival curves and service curves are used to specify upper bounds on the amount of traffic generated by flows and lower and upper bounds on the service offered by servers, respectively.

Definition 2.4 (Arrival Curve) Consider an arrival function $F(t)$. Any function $\alpha(t) \in \mathcal{F}$ is said to be an arrival curve of $F(t)$ if for all $t \geq 0$ it holds that

$$\alpha(t) \geq (F \oslash F)(t).$$

Generally it is meaningful to limit the set of arrival curves to sub-additive functions since otherwise the arrival curve $\alpha(t)$ itself allows deriving a tighter upper bound that is the sub-additive closure of $\alpha(t)$. We will come back to sub-additivity and related issues in Sect. 3.1.

Definition 2.5 (Service Curves) Consider a server with input and output arrival function $F(t)$ and $F'(t)$, respectively. Any two functions $\beta(t) \in \mathcal{F}$ and $\gamma(t) \in \mathcal{F}$ are said to be a minimum respectively maximum service curve of the server if for all $t \geq 0$ it holds that

$$F'(t) \geq (F \otimes \beta)(t)$$
$$F'(t) \leq (F \otimes \gamma)(t).$$

Throughout this work we generally assume that all types of servers, including forwarding but for example also processing units, are characterized by service curves.

Based on the concepts of arrival and service curve three fundamental performance bounds can be derived at first for single servers, which comprise output arrival curves, backlog bounds, and delay bounds.

Theorem 2.2 (Performance Bounds) *Consider a server which offers a minimum service curve $\beta(t)$. Let the input to the server be upper constrained by an arrival curve $\alpha(t)$. An output arrival curve of the server is*

$$\alpha'(t) = (\alpha \oslash \beta)(t).$$

The backlog at the server is upper bounded by

$$b \leq (\alpha \oslash \beta)(0).$$

Assuming first-in first-out order the delay is upper bounded according to

$$d \leq \inf\{t \geq 0 : (\alpha \oslash \beta)(-t) \leq 0\}.$$

Graphically, the backlog bound is the maximum vertical deviation of arrival and service curve and the delay bound the maximum horizontal deviation. If the server also offers a maximum service curve $\gamma(t)$ the output arrival curve can be refined to be $\alpha'(t) = ((\alpha \otimes \gamma) \oslash \beta)(t)$.

Theorem 2.3 (Tightness of Bounds) *The backlog and the delay bound in Th. 2.2 are tight, that is there exists a sample path such that the bounds hold with equality, if $\alpha, \beta \in \mathcal{F}$ are tight, which implies that α is sub-additive. If in addition α is left-continuous and $\alpha \overline{\oslash} \alpha$ is not bounded from above then the output bound is also tight.*

Theorem 2.4 (Concatenation) *Consider two servers with minimum service curves $\beta_1(t)$ and $\beta_2(t)$ and maximum service curves $\gamma_1(t)$ and $\gamma_2(t)$ in sequence. There exists an equivalent single server system with minimum and maximum service curve*

$$\beta(t) = (\beta_1 \otimes \beta_2)(t)$$
$$\gamma(t) = (\gamma_1 \otimes \gamma_2)(t).$$

The proof of Th. 2.4 follows from the associativity of min-plus convolution, that is $((F \otimes \beta_1) \otimes \beta_2)(t) = (F \otimes (\beta_1 \otimes \beta_2))(t)$. The same applies for maximum service curves.

The concatenation theorem is of particular importance since it allows deriving end-to-end delay and backlog bounds which scale linearly in the number of traversed servers. In contrast deriving delay or backlog bounds for each server and summing the individual bounds up results in quadratic scaling in the number of servers. Let us provide a simple example to support the importance of this issue.

Example 2.1 (Scaling of End-to-End Delay Bounds) Consider a series of n identical first-in first-out rate-latency servers with service curves $\beta_i(t) = R[t-T]^+$ for $i \in [1, n]$, where $[.]^+$ is equal to its argument if it is positive and zero otherwise. Let the input to the first server be leaky bucket constrained with arrival curve $\alpha_0(t) = rt + b$. An arrival curve of the output of the first server is $\alpha_1(t) = rt + b + rT$ which is the input arrival curve of the second server and so on. For the i-th server an output arrival curve is $\alpha_i(t) = rt + b + irT$. A delay bound for the first server is $d_1 \leq T + b/R$. For the second server we find $d_2 \leq T + (b + rT)/R$ and for the i-th server $d_i \leq T + (b + (i-1)rT)/R$. The sum over all n yields a bound for the end-to-end delay

$$d \leq \sum_{i=1}^{n} d_i \leq nT + n\frac{b}{R} + \frac{n}{2}(n-1)\frac{rT}{R}$$

which scales in $\mathcal{O}(n^2)$. In detail, there are two effects which relate to the burstiness of the flow: The term nb/R is in $\mathcal{O}(n)$ and reflects the cumulated time it takes to transmit b units of data by each of the servers. The second term $(n/2)(n-1)rT/R$ in $\mathcal{O}(n^2)$ is due to an increase in burstiness seen for the output of each subsequent server. Unfortunately the delay bound is not tight, which can be easily seen because the example considers both the worst-case output and the worst-case delay for each node simultaneously which, however, are mutually exclusive.

In contrast the end-to-end service curve obtained from concatenation of the n servers becomes $\beta = R[t-nT]^+$ and an end-to-end delay bound follows immediately as

$$d \leq nT + \frac{b}{R}$$

which scales in $\mathcal{O}(n)$. Note that the term which is due to the burstiness of the flow b/R does not depend on n nor is it subject to any kind of burstiness increase. Thus, it scales in $\mathcal{O}(1)$. This property of end-to-end concatenated service curves is also referred to as pay bursts only once phenomenon. It is fundamental for the efficient analysis of multiple node scenarios.

Definition 2.6 (Packetizer) Let $L(n) \in \mathcal{F}$ with $n \in \mathbb{N}_0$ be a discrete sequence of cumulative packet lengths, such that the length of the n-th packet is $L(n) - L(n-1)$. An L-packetizer is a system which given $L(n)$ and an input arrival function $F(t)$ enforces the output

$$P^L(F(t)) = \sup_{n \in [0,\infty)} \{L(n) 1_{L(n) \leq F(t)}\}$$

where $1_{[.]}$ is one if its argument is true and zero otherwise. The maximum packet length

$$l_{\max} = \sup_{n \in [0,\infty)} \{L(n+1) - L(n)\}$$

is assumed to be finite. The output arrival function $F'(t) = P^L(F(t))$ of the L-packetizer is said to be L-packetized, that is $F'(t) = P^L(F'(t))$.

Theorem 2.5 (Impact of Packetizer) *Consider the server in Def. 2.5. Assume the input to the server is L-packetized and let the output of the server be input to another L-packetizer, where $l_{\max}$ is the maximum packet size. The combined system offers a minimum respectively maximum service curve*

$$\beta'(t) = [\beta(t) - l_{\max}]^+$$
$$\gamma'(t) = \gamma(t).$$

Th. 2.5 essentially says that given an L-packetized arrival function $F(t)$ is input to the system described above, then $(F \otimes \beta')(t) \leq P^L((F \otimes \beta)(t))$ for all $t \geq 0$.

3. A CALCULUS WITH DATA SCALING

Whenever data are not just forwarded as they are but processed and altered, the amount of data may be subject to change, thus limiting the applicability of known network calculus as introduced in Sect. 2. We introduce and define the concept of scaling function to describe corresponding effects and extend the framework of network calculus accordingly.

3.1 Scaling Functions and Curves

This section provides a thorough definition of scaling functions and corresponding upper and lower bounding functions, called scaling curves. The definitions reflect the fundamental principles of network calculus and are key to an intuitive framework for analysis of distributed systems.

Definition 3.1 (Scaling Function) A scaling function $S \in \mathcal{F}$ assigns an amount of scaled data $S(a)$ to an amount of data a.

The definition of scaling functions matches the notion of a router respectively de-multiplexer in [4], where corresponding input-output relations are derived for several traffic models. The following investigations extend this concept to a broader framework.

Corollary 3.1 (Inverse Scaling Function) *Given a bijective scaling function $S \in \mathcal{F}$ it follows for its inverse S^{-1} that $S^{-1} \in \mathcal{F}$ is a scaling function, too.*

Proof. If S is bijective its inverse S^{-1} exists. Since $S \in \mathcal{F}$ implies $S : \mathbb{R}^+ \to \mathbb{R}^+$ the range of S which is the domain of S^{-1} is $\mathbb{R}^+$, that is S^{-1} is defined for $\mathbb{R}^+$ and it holds that $S^{-1}(S(a)) = S(S^{-1}(a)) = a$. Since $S(0) = 0$ it follows that $S^{-1}(0) = 0$. Further on, since $S(b) \geq S(a)$ for all $b \geq a$ it also follows that $S^{-1}(b) \geq S^{-1}(a)$ for all $b \geq a$, such that $S^{-1} \in \mathcal{F}$ is a scaling function. □

Continuous scaling functions are closely related to fluid traffic models, where irregularities that are introduced for example by the granularity of frames or packets can be easily incorporated using packetizers. Consider a scaling function $S(a)$ which is known only for a set of discrete amounts of data $a = L(n)$ where $L(n) \in \mathcal{F}$ and $n \in \mathbb{N}_0$. An example are video coders which operate on the granularity of frames or slices. Using the concept of packetizer from Def. 2.6 any arrival function $F(t) \in \mathcal{F}$ can be L-packetized and $P^L(F(t))$ can be used as input to a scaling function S which is defined for $L(n)$ with $n \in \mathbb{N}_0$ only. Moreover with the addition of the L-packetizer the domain of S can be extended to $\mathbb{R}^+$ and S can be defined where it previously was undefined such that $S \in \mathcal{F}$. Note that the output $S(P^L(F(t)))$ of the tandem of L-packetizer and scaling function S is not altered by this extension. Further on, if we investigate the term $S(P^L(F'(t)))$, where $F'(t)$ is the output of a server with service curve $\beta(t)$ and input arrival function $F(t)$ the extended scaling function $S : \mathbb{R}^+ \to \mathbb{R}^+$ allows applying Th. 2.5. It follows that $S(P^L((F \otimes \beta)(t))) \geq S((F \otimes \beta')(t))$, where $\beta'(t) = [\beta(t) - l_{\max}]^+$, since S is wide-sense increasing and $P^L((F \otimes \beta)(t)) \geq (F \otimes \beta')(t)$ for all $t \geq 0$. We come back to packetizers and scaling functions in Sect. 3.4.

In the sequel we frequently use inverse functions where we implicitly assume their existence. Note that the inverse of a function $f \in \mathcal{F}$ cannot always be assumed to exist, however, the pseudo-inverse $f^{-1}_{\inf}(a) = \inf\{b : f(b) \geq a\}$ as defined in [12] generally does. For the pseudo-inverse it is known that $f^{-1}_{\inf}(f(a)) \leq a$ and a worst-case deviation term δ can be defined, such that $f^{-1}_{\inf}(f(a)) \geq a - \delta$. Owing to these upper and lower bounds a similar theory as the one presented in the following can be derived using pseudo-inverse functions where, however, notational complexity increases and the deviation term δ may result in pessimistic bounds.

An example of an implementation of a bijective scaling function is a pair of encoder and decoder. Let $F(t)$ be the arrival function of a raw data stream. The effects of encoding can be described using a scaling function S_E which assigns a correspondingly scaled data stream with arrival function $F_S(t) = S_E(F(t))$ to the stream of raw data. In the same way there exists a scaling function S_D which reflects the properties of the decoder. Under the assumption of symmetry of encoder and decoder the function S_D is the inverse function of S_E denoted by $S_D = S_E^{-1}$ and we find that $S_D(F_S(t)) = F(t)$. Note that the generality of the definition of scaling functions allows modelling both a coding which removes redundancy, such as source coding, as well as a coding which adds redundancy, such as channel coding.

Definition 3.2 (Scaling Curves) Consider a scaling function S. Any two functions $\underline{S} \in \mathcal{F}$ and $\overline{S} \in \mathcal{F}$ are said to be a minimum respectively maximum scaling curve of S if for all $b \geq 0$ it holds that

$$\underline{S}(b) \leq \inf_{a \in [0,\infty)} \{S(b+a) - S(a)\} = (S \mathbin{\overline{\oslash}} S)(b)$$
$$\overline{S}(b) \geq \sup_{a \in [0,\infty)} \{S(b+a) - S(a)\} = (S \oslash S)(b).$$

If S is bijective it fulfills the condition of Cor. 3.1 and its inverse S^{-1} is a scaling function, too. Def. 3.2 allows deriving minimum and maximum scaling curves for S^{-1}, that is $\underline{S^{-1}}$ and $\overline{S^{-1}}$, in the same way. The following corollary provides an interesting property of scaling curves for inverse scaling functions.

Corollary 3.2 (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 $\overline{S}^{-1}$.*

PROOF. From Def. 3.2 it is known that $\overline{S^{-1}}$ is a maximum scaling curve of S^{-1} if for all $b \geq a \geq 0$ it holds that

$$\overline{S^{-1}}(b-a) \geq S^{-1}(b) - S^{-1}(a). \tag{1}$$

Since $S \in \mathcal{F}$ is wide-sense increasing and S is assumed to be bijective, that is it has range $\mathbb{R}^+$, (1) is equivalent to

$$\begin{aligned}\overline{S^{-1}}(S(b) - S(a)) \geq & S^{-1}(S(b)) - S^{-1}(S(a)) \\ = & b - a\end{aligned} \tag{2}$$

for all $b \geq a \geq 0$. Thus any function $\overline{S^{-1}}$ which fulfills (2) is a valid maximum scaling curve of S^{-1}.

Since $\underline{S} \in \mathcal{F}$ is wide-sense increasing it follows that its inverse $\underline{S}^{-1}$ is also wide-sense increasing. Thus we have for all $b \geq a \geq 0$ that

$$\underline{S}^{-1}(\underline{S}(b-a)) \leq \underline{S}^{-1}(S(b) - S(a)).$$

Interchanging the sides and using that $\underline{S}^{-1}(\underline{S}(b-a)) = b-a$ from the assumption of bijectivity yields

$$\underline{S}^{-1}(S(b) - S(a)) \geq b - a$$

which proves that $\underline{S}^{-1}$ is a valid maximum scaling curve of S^{-1} since it fulfills (2).

The proof for the minimum scaling curve of the inverse scaling function follows in the same way. □

Corollary 3.3 (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 said to be sub-additive respectively super-additive if $f(a+b) \leq f(a) + f(b)$ respectively $f(a+b) \geq f(a) + f(b)$. The sub-additive closure of f is defined to be $\inf_{n\geq 1}\{f^{(n)}\}$ 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 $\sup_{n\geq 1}\{f^{(n)}\}$ where $f^{(n)}$ is the n-fold max-plus self-convolution of f with $f^{(1)} = f$, $f^{(2)} = f \mathbin{\overline{\otimes}} f$, $f^{(3)} = f \mathbin{\overline{\otimes}} f \mathbin{\overline{\otimes}} f$, and so on. The sub-additive closure of $\overline{S}$ is the largest sub-additive function that is point-wise smaller than or equal to $\overline{S}$ and the super-additive closure of $\underline{S}$ is the smallest super-additive function that is point-wise greater than or equal to $\underline{S}$. If a function is sub-additive then it equals its sub-additive closure and if it is super-additive it equals its super-additive closure. If $\overline{S}$ is not sub-additive then there exists a tighter upper bound of S than $\overline{S}$ namely the sub-additive closure of $\overline{S}$. In the same way if $\underline{S}$ is not super-additive then its super-additive closure is a better lower bound of S than $\underline{S}$. Consequently we consider only sub-additive respectively super-additive maximum and minimum scaling curves.

PROOF. Def. 3.2 yields for all $b \geq 0$ and all $a \geq 0$ that

$$\overline{S}(b) \geq S(b+a) - S(a).$$

It also follows for all $c \geq b \geq 0$ and all $a \geq 0$ that

$$\overline{S}(c-b) \geq S(c+a) - S(b+a).$$

Addition of the two inequalities yields

$$\overline{S}(c-b) + \overline{S}(b) \geq S(c+a) - S(a)$$

for all $c \geq b \geq 0$ and all $a \geq 0$. Consequently it also holds for all $c \geq 0$ that

$$\inf_{b\in[0,c]}\{\overline{S}(c-b) + \overline{S}(b)\} \geq \sup_{a\in[0,\infty)}\{S(c+a) - S(a)\}$$

which is immediately equivalent to $(\overline{S} \otimes \overline{S})(c) \geq (S \oslash S)(c)$, such that given a maximum scaling curve $\overline{S}(c) \geq (S \oslash S)(c)$ it follows that $(\overline{S} \otimes \overline{S})(c)$ is a valid maximum scaling curve, too. Repeating this step infinitely and taking the infimum over all maximum scaling curves yields the sub-additive closure.

The super-additive closure of $\underline{S}$ follows analogously. □

3.2 Scaled Server

For end-to-end concatenation of systems with scaling functions the following scaling of servers is of major importance.

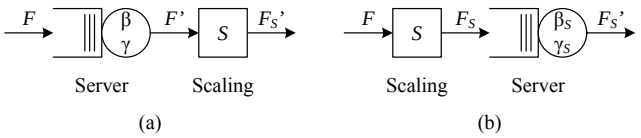

Figure 2: Scaling of servers

Theorem 3.1 (Scaled Server) *Consider the two systems in Fig. 2 and let $F(t)$ be the input arrival function. System (a) consists of a server with minimum and maximum service curve $\beta(t)$ and $\gamma(t)$ respectively 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 $\beta_S(t)$ and $\gamma_S(t)$ respectively. Given system (a), the lower and upper bounds of the output arrival function of system (b), that are $(S(F) \otimes \beta_S)(t)$ and $(S(F) \otimes \gamma_S)(t)$ respectively, are also valid lower and upper bounds of the output arrival function of system (a) if*

$$\begin{aligned}\beta_S(t) &= \underline{S}(\beta(t)) \\ \gamma_S(t) &= \overline{S}(\gamma(t))\end{aligned}$$

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

$$\begin{aligned}\beta(t) &= \underline{S^{-1}}(\beta_S(t)) \\ \gamma(t) &= \overline{S^{-1}}(\gamma_S(t))\end{aligned}$$

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

PROOF. From Def. 2.5 it is known that the output process of the server in system (a) is lower bounded by

$$F'(t) \geq (F \otimes \beta)(t).$$

Since S is wide-sense increasing it holds that

$$S(F'(t)) \geq S((F \otimes \beta)(t))$$

as well as

$$S\left(\inf_{s\in[0,t]}\{F(s) + \beta(t-s)\}\right) = \inf_{s\in[0,t]}\{S(F(s) + \beta(t-s))\}.$$

With $S(F'(t)) = F'_S(t)$ and $S(F(t)) = F_S(t)$ we have

$$\begin{aligned} F'_S(t) &\geq \inf_{s\in[0,t]}\{S(F(s)+\beta(t-s))\} \\ &= \inf_{s\in[0,t]}\{F_S(s)+S(F(s)+\beta(t-s))-S(F(s))\} \\ &\geq \inf_{s\in[0,t]}\{F_S(s)+\underline{S}(\beta(t-s))\} \\ &= (F_S \otimes \underline{S}(\beta))(t) \end{aligned}$$

which proves that the lower bound of the output arrival function obtained from system (b), that is $(S(F) \otimes \beta_S)(t)$, is a lower bound of the output arrival function of system (a), if $\beta_S(t) = \underline{S}(\beta(t))$.

For system (b) it holds by Def. 2.5 that

$$F'_S(t) \geq (F_S \otimes \beta_S)(t).$$

We use the same reasoning as above now for S^{-1}, which is also wide-sense increasing, and with $S^{-1}(F'_S(t)) = F'(t)$ and $S^{-1}(F_S(t)) = F(t)$ we find that

$$\begin{aligned} F'(t)) &\geq \inf_{s\in[0,t]}\{S^{-1}(F_S(s)+\beta_S(t-s))\} \\ &= \inf_{s\in[0,t]}\{F(s)+S^{-1}(F_S(s)+\beta_S(t-s))-S^{-1}(F_S(s))\} \\ &\geq \inf_{s\in[0,t]}\{F(s)+\underline{S^{-1}}(\beta_S(t-s))\} \\ &= (F \otimes \underline{S^{-1}}(\beta_S))(t) \end{aligned}$$

Since S is bijective we obtain in a final step that

$$F'_S(t) \geq S((F \otimes \underline{S^{-1}}(\beta_S))(t))$$

which proves that the lower bound of the output arrival function obtained from system (a), that is $S((F \otimes \beta)(t))$, is a lower bound of the output arrival function of system (b), if $\beta(t) = \underline{S^{-1}}(\beta_S(t))$.

The proof for the maximum service curve follows as an immediate variation. □

The particular advantage of such scaled servers becomes immediately obvious when considering end-to-end performance analysis of heterogenous systems as in the end-to-end concatenation examples in Sect. 5.

3.3 Performance Bounds

Def. 2.3 defines performance measures which, however, are not generally applicable in the context of scaling functions. While they are still meaningful if a single server is analyzed, we require adapted definitions for end-to-end backlog and delay which account for scaling functions to be able to address such systems as shown in Fig 2.

Definition 3.3 (Backlog and Delay) Consider a system with input arrival function $F(t)$ and output arrival function $F'_S(t)$, where the system implements a bijective scaling function S. The backlog of the system is defined as

$$b(t) = F(t) - S^{-1}(F'_S(t)).$$

Assuming first-in first-out data delivery the delay of the system is defined as

$$d(t) = \inf\{\tau \geq 0 : F(t) \leq S^{-1}(F'_S(t+\tau))\}.$$

In the presence of scaling functions there exists no single definition of backlog. By convention we use Def. 3.3 which states backlog as it can be seen from the data source. Note that due to scaling the actual amount of backlogged data within the system can be smaller but also larger. For example for the system in Fig 2 (b) the end-to-end backlog is $b(t) = F(t) - S^{-1}(F'_S(t))$, whereas the server sees a backlog of $b_S(t) = F_S(t) - F'_S(t)$ according to Def. 2.3.

The alert reader will notice that transforming the system in Fig. 2 (b) into the system in Fig. 2 (a) simplifies the derivation of end-to-end backlog and delay bounds according to Def. 3.3 significantly. If S is bijective the quantity $S^{-1}(F'_S(t)) = F'(t)$ is the output of the server in Fig. 2 (a). Consequently backlog and delay bounds according to Def. 3.3 can be derived by applying Th. 2.2 to the server in Fig. 2 (a).

Corollary 3.4 (Bounds for Scaling Functions) *Let $F(t)$ be an arrival function which is input to a scaling function S with maximum scaling curve $\overline{S}$ and let $\alpha(t)$ be an arrival curve of $F(t)$. An arrival curve of the scaled output arrival function $F_S(t)$ is*

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

Since $\alpha, \overline{S} \in \mathcal{F}$ it follows that $\overline{S}(\alpha(t)) \in \mathcal{F}$. If in addition α and $\overline{S}$ are sub-additive then $\overline{S}(\alpha(t))$ is sub-additive, too.

If S is bijective and S^{-1} has maximum scaling curve $\overline{S^{-1}}$ the same holds for $\overline{S^{-1}}$. Given an arrival curve of the scaled output process $\alpha_S(t)$ an arrival curve for the input is

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

Backlog and delay of scaling functions are zero.

Proof. From Def. 3.2 and with Def. 2.4 we have for all $t \geq 0$ and $s \geq 0$ that

$$\begin{aligned} F_S(s+t) - F_S(s) &= S(F(s+t)) - S(F(s)) \\ &\leq \overline{S}(F(s+t) - F(s)) \\ &\leq \overline{S}(\alpha(t)) \end{aligned}$$

which proves that $\overline{S}(\alpha(t))$ is an arrival curve of $F_S(t)$. The proof that $\overline{S^{-1}}(\alpha_S(t))$ is an arrival curve of $F(t)$ follows by substitution of S by S^{-1} and F_S, α_S by F, α and vice versa.

The closure $\overline{S}(\alpha(t)) \in \mathcal{F}$ is obvious. If $\overline{S}$ is sub-additive we have that $\overline{S}(\alpha(t)+\alpha(s)) \leq \overline{S}(\alpha(t)) + \overline{S}(\alpha(s))$. For sub-additive $\alpha(t)$ and since $\overline{S}$ is wide-sense increasing this implies that $\overline{S}(\alpha(t+s)) \leq \overline{S}(\alpha(t)) + \overline{S}(\alpha(s))$. The same follows for the inverse maximum scaling curve $\overline{S^{-1}}$.

With Def. 3.3 backlog and delay follow to be zero. □

Corollary 3.5 (Crosswise Validity of Bounds) *Consider the systems in Fig. 2. Given system (a), output arrival curves, end-to-end backlog bounds, and end-to-end delay bounds derived for the alternative system (b) from Th. 3.1 are valid performance bounds for system (a). Conversely given system (b), respective bounds derived with Th. 3.1 for system (a) are valid performance bounds for system (b).*

Proof. The crosswise validity of end-to-end performance bounds follows immediately from Th. 3.1. Recall that Th. 3.1 allows deriving a lower respectively upper bound for the output arrival function $F'_S(t)$ for system (a) respectively system (b) using either of the two systems and the input arrival function $F(t)$. An alternative, trivial upper bound for the output arrival function follows from the causality of the systems as $F'_S(t) \leq S(F(t))$. Output arrival curves, backlog bounds, and delay bounds are, however, defined only in

terms of input and output arrival functions for which Th. 3.1 provides the above mentioned bounds. □

For illustrative purposes we exemplarily show the derivation of end-to-end backlog and delay bounds for system (b) using the alternative system (a) from Th. 3.1. The output arrival function of system (a) $F'_S(t)$ is lower bounded according to

$$F'_S(t) \geq S((F \otimes \beta)(t)).$$

With Th. 3.1 the same lower bound holds for the output arrival function of system (b) if $\beta(t) = \underline{S^{-1}}(\beta_S(t))$. With Def. 3.3

$$\begin{aligned} b(t) =& F(t) - S^{-1}(F'_S(t)) \\ \leq& F(t) - S^{-1}(S((F \otimes \beta)(t))) \end{aligned}$$

is a backlog bound derived for system (a) which under the condition of Th. 3.1 holds also for system (b). In Def. 3.3 S is assumed to be bijective. With this property we obtain

$$b(t) \leq F(t) - (F \otimes \beta)(t).$$

The rest is an application of a well-known network calculus result. With $\alpha(s) \geq F(t) - F(t-s)$ it follows that

$$\begin{aligned} b(t) \leq& F(t) - \inf_{s \in [0,t]} \{F(t-s) + \beta(s)\} \\ =& \sup_{s \in [0,t]} \{F(t) - F(t-s) - \beta(s)\} \\ \leq& \sup_{s \in [0,t]} \{\alpha(s) - \beta(s)\} \\ \leq& (\alpha \oslash \beta)(0). \end{aligned}$$

Regarding the delay it follows with Def. 3.3 that

$$\begin{aligned} d(t) =& \inf\{\tau \geq 0 : F(t) \leq S^{-1}(F'_S(t+\tau))\} \\ \leq& \inf\{\tau \geq 0 : F(t) \leq S^{-1}(S((F \otimes \beta)(t+\tau)))\} \end{aligned}$$

is a delay bound derived for system (a) which under the condition of Th. 3.1 holds also for system (b). Since S is assumed to be bijective we have

$$d(t) \leq \inf\{\tau \geq 0 : F(t) \leq (F \otimes \beta)(t+\tau)\}.$$

With $\alpha(s-\tau) \geq F(t) - F(t+\tau-s)$ and using a known network calculus result we conclude that

$$\begin{aligned} d(t) \leq& \inf\{\tau \geq 0 : F(t) \leq \inf_{s \in [0,t+\tau]} \{F(t+\tau-s) + \beta(s)\}\} \\ =& \inf\{\tau \geq 0 : \sup_{s \in [0,t+\tau]} \{F(t) - F(t+\tau-s) - \beta(s)\} \leq 0\} \\ \leq& \inf\{\tau \geq 0 : \sup_{s \in [0,t+\tau]} \{\alpha(s-\tau) - \beta(s)\} \leq 0\} \\ \leq& \inf\{\tau \geq 0 : (\alpha \oslash \beta)(-\tau) \leq 0\}. \end{aligned}$$

The performance bounds in Th. 2.2 have been shown to be tight in [12], that is they are attained in case of greedy sources and lazy servers. Concerning the tightness of performance bounds derived for systems with scaling functions we make the following important conclusions.

Corollary 3.6 (Tightness of Bounds from Cor. 3.4) *The output arrival curve $\alpha_S(t)$ in Cor. 3.4 is tight, that is it is attained for a certain sample path, if the input arrival curve $\alpha(t)$ and the maximum scaling curve $\overline{S}$ are sub-additive and simultaneously tight. Conversely, if $\alpha_S(t)$ and $\overline{S^{-1}}$ are sub-additive and simultaneously tight then $\alpha(t)$ is tight. The backlog and delay bounds of scaling functions are zero and thus trivially tight.*

Proof. Consider a greedy source with arrival function $F(t) = \alpha(t)$ and let $F(t)$ be input to a scaling function S which is greedy, that is $S(F(t)) = \overline{S}(F(t))$. Then the output arrival function of the scaling function is $F_S(t) = S(F(t)) = \overline{S}(F(t)) = \overline{S}(\alpha(t))$ and the output arrival curve $\alpha_S(t) = \overline{S}(\alpha(t))$ is attained. The converse follows as an immediate variation. □

Corollary 3.7 (Tightness of Bounds from Cor. 3.5) *Consider the system in Fig. 2 (a). Let the input arrival function $F(t)$ be upper bounded by an arrival curve $\alpha(t)$ and the scaling function S by a maximum scaling curve $\overline{S}$. Assume that the conditions of Th. 2.3, Def. 3.3 and Cor. 3.6 are fulfilled. The following output arrival curve, backlog bound, and delay bound of the system are tight, that is there exists a sample path such that the bounds hold with equality:*

$$\begin{aligned} \alpha'_S(t) &= \overline{S}((\alpha \oslash \beta)(t)) \\ b &\leq (\alpha \oslash \beta)(0) \\ d &\leq \inf\{t \geq 0 : (\alpha \oslash \beta)(-t) \leq 0\}. \end{aligned}$$

Now consider the system in Fig. 2 (b). Further on, assume that the conditions of Th. 2.3 are fulfilled for $\alpha_S(t) = \overline{S}(\alpha(t))$ and $\beta_S(t)$. The following output arrival curve, backlog bound, and delay bound of the system, which follow with Cor. 3.2 and Th. 3.1, are tight:

$$\begin{aligned} \alpha'_S(t) &= (\overline{S}(\alpha) \oslash \beta_S)(t) \\ b &\leq (\alpha \oslash \overline{S}^{-1}(\beta_S))(0) \\ d &\leq \inf\{t \geq 0 : (\alpha \oslash \overline{S}^{-1}(\beta_S))(-t) \leq 0\}. \end{aligned}$$

Proof. We start with the system in Fig. 2 (a). The output arrival curve of the server $\alpha'(t) = (\alpha \oslash \beta)(t)$ is tight according to Th. 2.3. Since $\alpha'(t)$ is tight it follows from Cor. 3.6 that $\overline{S}(\alpha'(t))$ is tight, too. From Th. 2.3 it is also known that b and d are tight backlog and delay bounds for the server in system (a). Since backlog and delay of scaling functions are zero, see Cor. 3.6, these bounds apply immediately as tight end-to-end bounds for the system.

Regarding the system in Fig. 2 (b) the output of the scaling function has an arrival curve $\alpha_S(t) = \overline{S}(\alpha(t))$ which under the conditions of Cor. 3.6 is tight. If $\alpha_S(t)$ and $\beta_S(t)$ fulfill the additional conditions of Th. 2.3 then the output arrival curve $\alpha'_S(t) = (\alpha_S \oslash \beta_S)(t)$ is tight, too. It is known from Cor. 3.4 that $\alpha_S \in \mathcal{F}$ is sub-additive if $\alpha, \overline{S} \in \mathcal{F}$ are sub-additive.

The backlog and delay bound are derived using the scaled server with service curve $\underline{S^{-1}}(\beta_S(t))$ as established by Th. 3.1. With Cor. 3.2 we choose $\underline{S^{-1}}$ to be $\overline{S}^{-1}$. Since $\overline{S} \in \mathcal{F}$ is wide-sense increasing it follows that its inverse $\overline{S}^{-1}$ is also wide-sense increasing, which is required in the sequel.

We start with the delay bound. As before assume the output of the scaling function $\alpha_S(t) = \overline{S}(\alpha(t))$ is tight and sub-additive. With Th. 2.2 and Th. 2.3 it follows that $\inf\{t \geq 0 : (\overline{S}(\alpha) \oslash \beta_S)(-t) \leq 0\}$ is a tight delay bound for the server and with Cor. 3.6 also for the system. Thus, it is sufficient to show that the same delay bound can be found applying the alternative system in Fig. 2 (a) from Th. 3.1. The delay bound in Th. 2.2 can be restated to be the smallest $d \geq 0$ such that

$$\begin{aligned} & \overline{S}(\alpha(t)) && \leq && \beta_S(t+d) && \forall t \geq 0 \\ \Leftrightarrow\ & \overline{S}^{-1}(\overline{S}(\alpha(t))) && \leq && \overline{S}^{-1}(\beta_S(t+d)) && \forall t \geq 0 \\ \Leftrightarrow\ & \alpha(t) && \leq && \overline{S}^{-1}(\beta_S(t+d)) && \forall t \geq 0 \end{aligned}$$

which shows the equivalence and thus proves the tightness. Note that Cor. 3.2, which has been used before, assumes that $\overline{S}$ is bijective.

The proof of the backlog bound uses a sample path argument as in [12]. Let the input to the system be $F(t) = \alpha(t)$ and let the maximum scaling curve of the scaling function be attained, that is $S(\alpha(t)) = \overline{S}(\alpha(t))$, such that its output is $F_S(t) = \alpha_S(t) = \overline{S}(\alpha(t))$. If the server is lazy, that is its service curve $\beta_S(t)$ is attained, the output of the server becomes

$$\begin{aligned} & F'_S(t) &=& \min\{\alpha_S(t), \beta_S(t)\} \\ \Leftrightarrow\ & \overline{S}^{-1}(F'_S(t)) &=& \overline{S}^{-1}(\min\{\alpha_S(t), \beta_S(t)\}) \\ \Leftrightarrow\ & \overline{S}^{-1}(F'_S(t)) &=& \min\{\alpha(t), \overline{S}^{-1}(\beta_S(t))\} \end{aligned}$$

Since $\overline{S}^{-1} = S^{-1}$ for the investigated sample path, we find that $\overline{S}^{-1}(F'_S(t)) = S^{-1}(F'_S(t))$. With $F(t) = \alpha(t)$ and $S^{-1}(F'_S(t)) = \overline{S}^{-1}(\beta_S(t))$ if $\alpha(t) \geq \overline{S}^{-1}(\beta_S(t))$ the backlog becomes $F(t) - S^{-1}(F'_S(t)) = \alpha(t) - \overline{S}^{-1}(\beta_S(t))$ which proves that the bound is attained. □

Cor. 3.7 shows how the alternative systems in Fig. 2 can be used efficiently to derive tight performance bounds. Note, however, that alternative systems have to be selected carefully. An example of an unfavorable choice is transforming the system in Fig. 2 (a) into the one in Fig. 2 (b). In this case the output bound becomes $(\overline{S}(\alpha) \oslash \underline{S}(\beta))(t)$. The problem of this choice becomes immediately obvious since S can generally not attain $\overline{S}$ and $\underline{S}$ simultaneously. Consequently, the bound is conservative and will usually not be tight. Tab. 1 summarizes recommendations for proper use of scaled servers such that tight performance bounds can be derived as established by Cor. 3.7.

Table 1: Use of systems according to Cor. 3.7

bounds	original system	
Def. 3.3	Fig. 2 (a)	Fig. 2 (b)
output	use (a)	use (b)
backlog	use (a)	use (a)
delay	use (a)	(a) or (b)

3.4 Packetizer

This section investigates packetizers in the presence of scaling functions and develops the notion of sub- and super-packetizer to incorporate packetization at different levels.

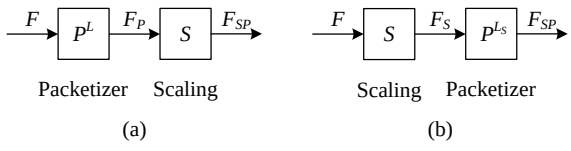

Figure 3: Scaling of packetizers

Corollary 3.8 (Scaled Packetizer) *Consider the two systems in Fig. 3. System (a) consists of an L-packetizer P^L with maximum packet size $l_{\max}$ whose output is scaled with scaling function S and system (b) consists of a scaling function S whose output is input to a L_S-packetizer P^{L_S} with maximum packet size $l_{S,\max}$.*

Given system (a), system (b) is equivalent, if $L_S = S(L)$. For the maximum packet size it follows that $l_{S,\max} \leq \overline{S}(l_{\max})$, where $\overline{S}$ is the corresponding maximum scaling curve of S.

Given system (b), system (a) is equivalent, if S is bijective and $L = S^{-1}(L_S)$. For the maximum packet size it follows that $l_{\max} \leq \overline{S^{-1}}(l_{S,\max})$, where $\overline{S^{-1}}$ is the corresponding maximum scaling curve of S^{-1}.

PROOF. Let $F(t)$ be the input arrival function. Since S is wide-sense increasing we have

$$\begin{aligned} S(P^L(F(t))) =& S\Big(\sup_{n\in[0,\infty)} \big\{L(n)1_{L(n)\leq F(t)}\big\}\Big) \\ =& \sup_{n\in[0,\infty)} \big\{S(L(n))1_{L(n)\leq F(t)}\big\} \end{aligned}$$

for the system displayed in Fig. 3 (a). For the system in Fig. 3 (b) we find

$$P^{L_S}(S(F(t))) = \sup_{n\in[0,\infty)} \big\{L_S(n)1_{L_S(n)\leq S(F(t))}\big\}.$$

Letting $L_S = S(L)$ respectively $L = S^{-1}(L_S)$ and since S is wide-sense increasing it follows that

$$\begin{aligned} P^{L_S}(S(F(t))) =& \sup_{n\in[0,\infty)} \big\{S(L(n))1_{S(L(n))\leq S(F(t))}\big\} \\ =& \sup_{n\in[0,\infty)} \big\{S(L(n))1_{L(n)\leq F(t)}\big\} \end{aligned}$$

which establishes the equivalence of the two systems expressed by $S(P^L(F(t))) = P^{L_S}(S(F(t)))$.

Next we investigate the maximum sized packets. Since

$$\begin{aligned} l_{\max} =& \sup_{n\in[0,\infty)} \{L(n+1) - L(n)\} \\ l_{S,\max} =& \sup_{n\in[0,\infty)} \{S(L(n+1)) - S(L(n))\} \end{aligned}$$

with $L_S = S(L)$. It follows that $l_{S,\max} \leq \overline{S}(l_{\max})$. We also have

$$\begin{aligned} l_{S,\max} =& \sup_{n\in[0,\infty)} \{L_S(n+1) - L_S(n)\} \\ l_{\max} =& \sup_{n\in[0,\infty)} \{S^{-1}(L_S(n+1)) - S^{-1}(L_S(n))\} \end{aligned}$$

with $L = S^{-1}(L_S)$. Consequently $l_{\max} \leq \overline{S^{-1}}(l_{S,\max})$. □

Definition 3.4 (Sub- and Super-Packetizer) Let $L_1(n)$, $L_2(m) \in \mathcal{F}$ with $n, m \in \mathbb{N}_0$ be discrete sequences of cumulative packet lengths. $L_2(m)$ is a sub-packetized sequence of $L_1(n)$ and $L_1(n)$ is a super-packetized sequence of $L_2(m)$ if for all $n \geq 0$ there exists an $m \geq 0$ such that $L_1(n) = L_2(m)$, which implies $m \geq n$. The corresponding packetizers follow according to Def. 2.6.

Lemma 3.1 (Sub- and Super-Packetizer in Series) *Let $L_1(n)$ and $L_2(m)$ be a super- and a sub-packetized sequence according to Def. 3.4 respectively and let P^{L_1} and P^{L_2} characterize the corresponding packetizers. Consider the series of the two packetizers and let $F(t)$ be the input arrival function. It follows for all $t \geq 0$ that*

$$\begin{aligned} P^{L_1}(P^{L_2}(F(t))) =& P^{L_1}(F(t)) \\ P^{L_2}(P^{L_1}(F(t))) =& P^{L_1}(F(t)). \end{aligned}$$

PROOF. From Def. 3.4 we have for all $t \geq 0$ that

$$\sup_{n\in[0,\infty)}\{L_1(n)1_{L_1(n)\leq F(t)}\} \leq \sup_{m\in[0,\infty)}\{L_2(m)1_{L_2(m)\leq F(t)}\}$$

and it follows for all $n \geq 0$ and $t \geq 0$ that

$$\begin{aligned} &L_1(n) \leq F(t) \\ \Leftrightarrow &L_1(n) \leq \sup_{m\in[0,\infty)}\{L_2(m)1_{L_2(m)\leq F(t)}\} = P^{L_2}(F(t)). \end{aligned}$$

Consequently we have

$$\begin{aligned} P^{L_1}(P^{L_2}(F(t))) &= \sup_{n\in[0,\infty)}\left\{L_1(n)1_{L_1(n)\leq P^{L_2}(F(t))}\right\} \\ &= \sup_{n\in[0,\infty)}\left\{L_1(n)1_{L_1(n)\leq F(t)}\right\} \\ &= P^{L_1}(F(t)) \end{aligned}$$

which proves the first part. Also we immediately have from Def. 3.4 that

$$\begin{aligned} P^{L_2}(P^{L_1}(F(t))) &= \sup_{m\in[0,\infty)}\{L_2(m)1_{L_2(m)\leq P^{L_1}(F(t))}\} \\ &= P^{L_1}(F(t)) \end{aligned}$$

since there generally exists an $m \geq 0$ such that $L_2(m) = P^{L_1}(F(t))$ is attained. □

Corollary 3.9 (Impact of Sub-Packetizer) *Consider a server with minimum service curve $\beta(t)$ and maximum service curve $\gamma(t)$ and L_1-packetized input. Let the output of the server be input to an L_2-packetizer and let $l_{2,\max}$ be the corresponding maximum packet size. If L_2 is a sub-packetized sequence of L_1 the combined system offers a minimum respectively maximum service curve*

$$\begin{aligned} \beta'(t) &= [\beta(t) - l_{2,\max}]^+ \\ \gamma'(t) &= \gamma(t). \end{aligned}$$

The proof is a variation of the proof of Th. 2.5 in [4, 12].

PROOF. First note that generally

$$F(t) \geq P^{L_i}(F(t)) \geq F(t) - l_{i,\max}. \tag{3}$$

Let $F(t)$ and $F'(t)$ be the input and output arrival functions of the server respectively, where $F'(t) \geq (F \otimes \beta)(t)$. Since $P^{L_i} \in \mathcal{F}$ is wide-sense increasing we have

$$P^{L_2}(F'(t)) \geq P^{L_2}((F\otimes\beta)(t)) = \inf_{s\in[0,t]}\{P^{L_2}(F(s)+\beta(t-s))\}.$$

With (3) it follows for all $t \geq 0$ and $s \in [0, t]$ that

$$P^{L_2}(F(s) + \beta(t-s)) \geq F(s) + \beta(t-s) - l_{2,\max}.$$

Since $F(t)$ is L_1-packetized we have $F(t) = P^{L_1}(F(t))$. Further on, since L_2 is a sub-packetized sequence of L_1 we have from Lem. 3.1 for all $t \geq 0$ and $s \in [0, t]$ that

$$\begin{aligned} P^{L_2}(F(s) + \beta(t-s)) &= P^{L_2}(P^{L_1}(F(s)) + \beta(t-s)) \\ &\geq P^{L_2}(P^{L_1}(F(s))) \\ &= P^{L_1}(F(s)) \\ &= F(s) \end{aligned}$$

which proves the $[.]^+$ condition of the minimum service curve. Concerning the maximum service curve we immediately have from (3) that for all $t \geq 0$

$$P^{L_2}((F \otimes \gamma)(t)) \leq (F \otimes \gamma)(t)$$

which completes the proof. □

4. VIDEO CODING EXAMPLE

For illustrative purposes let us consider a video encoder with scaling function S and maximum scaling curve $\overline{S}$ whose output is transmitted across a network which is represented by a rate-latency service curve $\beta_s(t) = R[t-T]^+$. The system is for example shown in Fig. 2 (b). Assume that the input to the scaling function is a constant bit rate stream of raw video data with arrival curve $\alpha(t) = rt$. The effects which are for example due to the granularity of raw video frames can be easily incorporated using the concept of packetizer in Def. 2.6. Related aspects are, however, excluded here for ease of presentation. An output arrival curve of the scaling function is $\overline{S}(\alpha(t))$ such that a delay bound for the system becomes $\inf\{t \geq 0 : (\overline{S}(\alpha) \oslash \beta_s)(-t) \leq 0\}$.

Alternatively we can investigate the system in Fig. 2 (a) where the server offers a service curve $\overline{S}^{-1}(\beta_s(t))$ according to Th. 3.1 and Cor. 3.2. The corresponding delay bound for the system is $\inf\{t \geq 0 : (\alpha \oslash (\overline{S}^{-1}(\beta_s))(-t) \leq 0\}$ which according to Cor. 3.5 is a valid delay bound for the original system, too.

We use a conservative approximation of the maximum scaling curve applying affine functions with parameters p_i, q_i which correspond to a series of n leaky buckets given by

$$\overline{S}(a) = \min_{i\in[1,n]}\{p_i + q_i a\} \geq (S \oslash S)(a) \tag{4}$$

where a is the cumulated raw video data. The inverse maximum scaling curve follows as

$$\overline{S}^{-1}(b) = \max_{i\in[1,n]}\{[b-p_i]^+/q_i\} \leq (S^{-1} \mathbin{\overline{\oslash}} S^{-1})(b) \tag{5}$$

where b is the cumulated encoded video data. Given (4) the inequality in (5) holds due to Cor. 3.2. An example of such a multiple affine approximation of a scaling curve and its inverse is illustrated in Fig. 4.

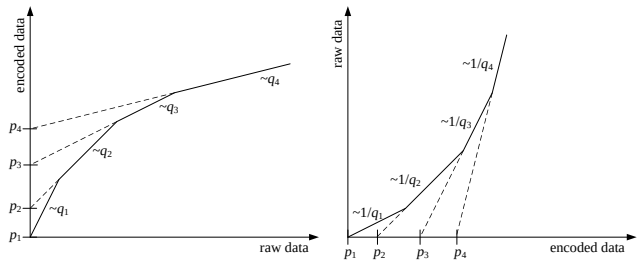

Figure 4: Affine maximum scaling curve and inverse

Fig. 5 shows the accordingly scaled arrival curve $\overline{S}(\alpha(t))$ with $\alpha(t) = rt$ and the service curve $\beta_s(t) = R[t-T]^+$ as well as the arrival curve $\alpha(t) = rt$ and the scaled service curve $\overline{S}^{-1}(\beta_s(t))$ with $\beta_s(t) = R[t-T]^+$ of the alternative system.

The corresponding delay bound d is identical for both systems, whereas the backlog bound is not, which is due to the fact that b_S represents the backlog obtained at the server in terms of scaled data, whereas b is the backlog as seen from the point of view of the data source according to Def. 3.3 in terms of raw data. Note that the scaling curve in Fig. 4 exhibits a compression gain for large amounts of raw data, but not for small amounts. As a consequence Fig. 5 shows that $b_S > b$.

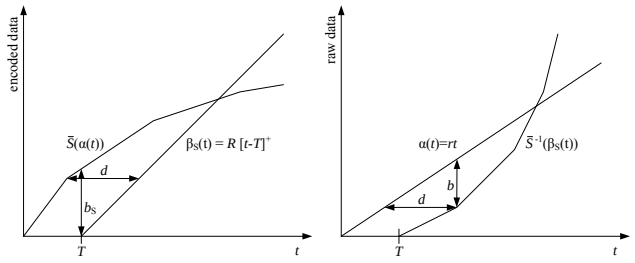

Figure 5: Backlog and delay for alternative systems

Let us exemplify the approximation in (4) for the first 2000 frames of the news video sequence from [10]. The raw sequence is in QCIF YUV format with 176×144 pixel resolution, 8 bit depths, and luminance chrominance sub-sampling of 4:2:0, resulting in a raw video frame size of 38.016 kB. The frame rate is 25 frames/s. The sequence is MPEG-4 encoded using Intracoded (I)-frames, Predicted (P)-frames, and Bi-directionally predicted (B)-frames, and a fixed Group of Pictures (GOP) which is IBBPBBPBBP. The largest encoded frame is 12.55 kB. The encoded sequence is shown in Fig. 6.

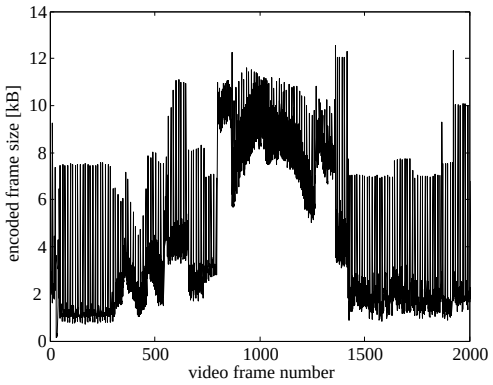

Figure 6: News video sequence

Fig. 7 shows the corresponding scaling function S, whose domain has been extended to $\mathbb{R}^+$ using linear interpolation, the tightest maximum scaling curve, that is $S \oslash S$, and a multiple affine maximum scaling curve according to (4). The corresponding parameters of a quick estimation are given in Tab. 2. For an elaborate overview on the characterization of video traffic using deterministic arrival curves and multiple leaky bucket descriptors see for example [13, 17].

Table 2: Affine upper bounds on $S \oslash S$

i	p_i	q_i
1	0 kB	0.331
2	2.55 kB	0.263
3	138.55 kB	0.221
4	4230.55 kB	0.079

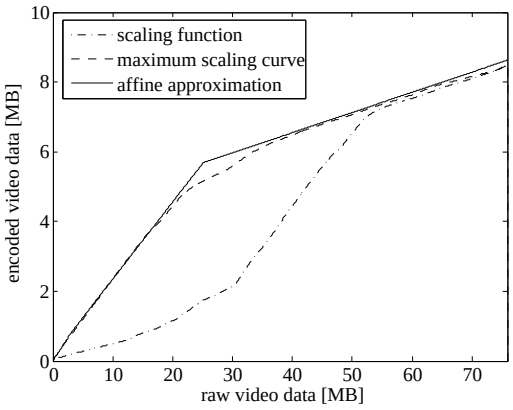

Figure 7: News video sequence scaling curves

5. END-TO-END CONCATENATION

This section demonstrates how the theory from Sect. 3 can be efficiently used to derive single server representations of tandem servers in the presence of scaling functions and heterogeneous packetizers. The concatenation property of service curves is of significant importance when deriving end-to-end performance bounds. As established by Ex. 2.1 end-to-end delay bounds scale linearly in the number of concatenated servers, compared to quadratic scaling, if bounds are derived iteratively for each server and summed up afterwards. We start with an investigation of symmetric scaling functions before analyzing the general case. Finally we take effects which are due to packetization into account and add sub- and super-packetizers to the scenario.

5.1 Symmetric Scaling Functions

Consider the system in Fig. 8 which consists of an encoder, a two-node network, and a symmetric decoder in series. Each of the system elements is abstracted by a service curve where $\beta_E(t)$ and $\beta_D(t)$ correspond to the processing units of encoder and decoder respectively and $\beta_1(t)$ and $\beta_2(t)$ are the service curves of the two network nodes. For ease of presentation effects due to packetization are not yet considered.

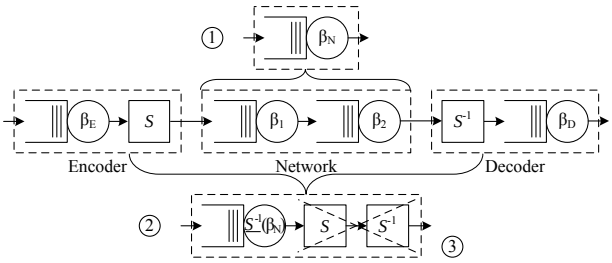

Figure 8: Symmetric scaling functions

Owing to the renowned network calculus concatenation theorem the network can be represented by a single server system with service curve $\beta_N(t) = (\beta_1 \otimes \beta_2)(t)$ as shown in Fig. 8 step (1). It is worthwhile to repeat this step and lump encoder, network, and decoder into one single server system,

however, since encoder and decoder alter the amount of data that is transmitted, this is not possible.

With Th. 3.1 we can substitute the subsystem consisting of the scaling function S in the encoder and the network by an alternative system where data is first input to the network which is described by an accordingly scaled service curve $\underline{S^{-1}}(\beta_N(t))$ before the scaling with S applies, as shown in step (2). In the alternative system the output of the scaling function S is immediately input to the scaling function S^{-1} and the two scaling functions cancel out each other, see step (3). Thus, the end-to-end service curve can be derived to be

$$\beta(t) = (\beta_E \otimes \underline{S^{-1}}(\beta_N) \otimes \beta_D)(t).$$

Note that the same result is obtained if the scaling function S^{-1} is shifted from the egress to the ingress of the network. The concatenated service curve immediately allows deriving end-to-end backlog and delay bounds according to Def. 3.3 by applying Th. 2.2.

Let us now come back to the motivating example in Fig. 1. The system in Fig. 8 corresponds to the example, if the transcoder is not used, which can also be expressed by setting the scaling function of the transcoder to $S_T(a) = a$. Differing from Fig. 8 we denote the scaling function of the encoder S_E and the one of the decoder S_D. Under the assumption that the raw video stream is recovered at the decoder we have $S_D(S_E(a)) = a$. In the presence of the transcoder $S_D(S_T(S_E(a))) = a$ can be assumed. Thus, the scaling functions S_E, S_T, and S_D in series cancel out each other in the same way as S_E and S_D did before. The scaling function of the transcoder S_T is located within the network. It can be easily shifted to the egress of the network using Th. 3.1 before Fig. 8 step (1) is carried out.

5.2 General Scaling Functions

Consider the system in Fig. 9 which consists of a scaling function S_1, a two-node network, a scaling function S_2, and another two-node network in sequence. Each of the queuing stations is represented by a service curve where the service curve of the network in the left becomes $\beta_{N_1}(t) = (\beta_{11} \otimes \beta_{12})(t)$ and the service curve of the one in the right $\beta_{N_2}(t) = (\beta_{21} \otimes \beta_{22})(t)$, as shown in Fig. 9 step (1) and step (2).

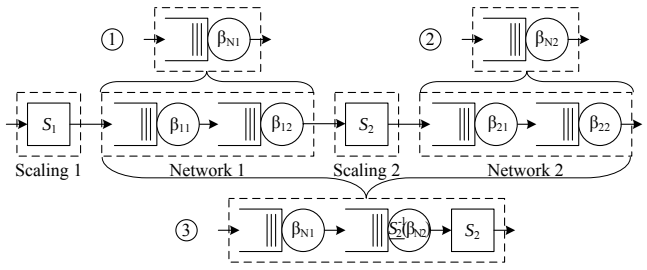

Figure 9: General scaling functions

With Th. 3.1 and following the recommendations in Tab. 1 we have the option of deriving an alternative system where all scaling functions apply at the egress of the system. In a first step we shift the scaling function S_2 from the ingress of network two to its egress resulting in a correspondingly scaled service curve $\underline{S_2^{-1}}(\beta_{N_2}(t))$, see step (3). After concatenation of the network service curves of network one and two we obtain $(\beta_{N_1} \otimes \underline{S_2^{-1}}(\beta_{N_2}))(t)$ and after shifting the scaling function S_1 to the egress of the system the end-to-end service curve of the system becomes

$$\beta(t) = \underline{S_1^{-1}}((\beta_{N_1} \otimes \underline{S_2^{-1}}(\beta_{N_2}))(t)).$$

Note that it is also possible to shift the scaling function S_1 to the egress of the system before the service curves of network one and network two are concatenated. Doing so is, however, unfavorable since $\underline{S_1^{-1}}$ is assumed to be super-additive according to Cor. 3.3. That is, it is generally better to apply $\underline{S_1^{-1}}$ to a sum rather than to the summands. Another alternative is to shift S_2 to the egress of the system and leave S_1 at the ingress or to shift S_2 to the ingress of the system. However, a careful selection of alternative systems has to be made regarding the tightness of bounds as established by Cor. 3.7 and summarized in Tab. 1, where it usually is advantageous to move scaling functions to the egress of the system when deriving end-to-end backlog and delay bounds.

5.3 Sub- and Super-Packetizer

Let us exemplify the end-to-end concatenation of systems with sub- and super-packetizers on the basis of the system displayed in Fig. 10. The input to the system is L_1-packetized, scaled by S, and afterwards $L_{2,S}$-packetized, before it is transmitted across a network with service curve β_N. The network may already consist of a number of concatenated nodes including further $L_{2,S}$-packetizers. The output of the network is again $L_{2,S}$-packetized, then $L_{1,S}$-packetized and finally scaled by S^{-1}. Assume that $L_2 = S^{-1}(L_{2,S})$ is a sub-packetized sequence of L_1 and let $l_{1,\max}$ respectively $l_{2,\max}$ be the corresponding maximum packet sizes. It follows that $L_{2,S}$ is a sub-packetized sequence of $L_{1,S} = S(L_1)$.

As an example consider a sequence of video frames L_1 where each raw video frame is encoded using the scaling function S and individually packetized into a series of datagrams $L_{2,S}$ which are transmitted across a packet network. At the egress of the network the packetizer $L_{1,S}$ groups packets that belong to the same encoded video frame and the complete encoded frame is decoded using the scaling function S^{-1} to recover the raw video frame.

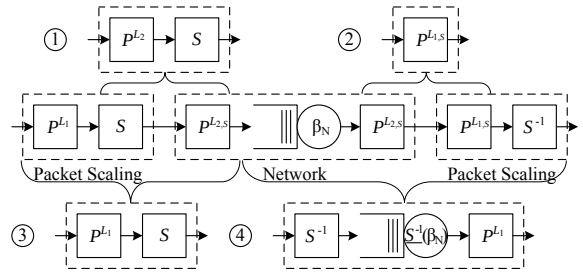

Figure 10: Sub- and Super-Packetizer

With Cor. 3.8 we can replace the series of S and $P^{L_{2,S}}$ by the tandem of P^{L_2} and S, as shown in Fig. 10 step (1). Since $L_{2,S}$ and L_2 are sub-packetized sequences of $L_{1,S}$ and L_1 respectively, we can apply Lem. 3.1 to replace the series of $P^{L_{2,S}}$ and $P^{L_{1,S}}$ by $P^{L_{1,S}}$, see step (2), and the series of P^{L_1} and P^{L_2} by P^{L_1}, as in step (3). In step (4) the scaling function S^{-1} is shifted to the ingress of the network using Cor. 3.8 and Th. 3.1, where the packetizer $P^{L_{1,S}}$ becomes P^{L_1} and the service curve of the network is derived

as $\underline{S^{-1}}(\beta_N)$. Finally the scaling functions S and S^{-1} cancel each other out. Since the input to the server $\underline{S^{-1}}(\beta_N)$ is L_1-packetized we can apply Th. 2.5 and collapse the server and the subsequent packetizer. If further on the input to the system is assumed to be L_1-packetized the L_1-packetizer at the ingress of the system is redundant and the end-to-end service curve of the system becomes

$$\beta(t) = [\underline{S^{-1}}(\beta_N(t)) - l_{1,\max}]^+.$$

Fig. 8 and 10 each concern relevant aspects of the example in Fig. 1. Their combination, that is adding servers with service curve β_E and β_D at the ingress respectively egress of the system in Fig. 10, provides a detailed model for the end-to-end video delivery system in Fig. 1 including packetization effects which are due to the granularity of video frames and network datagrams.

6. CONCLUSION

We extended the framework of network calculus from networks with pure forwarding to distributed systems with various kinds of data processing. To this end we introduced the concept of scaling functions and developed a concise network calculus with data scaling. In particular we showed the existence of alternative systems which make the relocation of scaling functions along the data path possible. These systems lay the foundation for end-to-end concatenation of servers with data scaling and potentiate the derivation of tight performance bounds.

Scaling functions constitute a fundamental concept for modelling system elements beyond forwarding. The general definition yields a universal framework for analysis of distributed systems. The developed methodology for data scaling widens the applicability of network calculus significantly and finally incorporates many basic aspects of communications, such as source and channel coding.

7. REFERENCES

[1] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan. Performance bounds for flow control protocols. *IEEE/ACM Trans. Networking*, 7(3):310–323, June 1999.

[2] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat. *Synchronization and Linearity: An Algebra for Discrete Event Systems*. Probability and Mathematical Statistics. John Wiley & Sons Ltd., West Sussex, Great Britain, 1992.

[3] C.-S. Chang. On deterministic traffic regulation and service guarantees: A systematic approach by filtering. *IEEE Trans. Inform. Theory*, 44(3):1097–1110, May 1998.

[4] C.-S. Chang. *Performance Guarantees in Communication Networks*. Telecommunication Networks and Computer Systems. Springer-Verlag, London, Great Britain, 2000.

[5] F. Ciucu, A. Burchard, and J. Liebeherr. A network service curve approach for the stochastic analysis of networks. In *Proc. ACM SIGMETRICS*, pages 279–290, June 2005.

[6] R. L. Cruz. A calculus for network delay, Part I: Network elements in isolation. *IEEE Trans. Inform. Theory*, 37(1):114–131, January 1991.

[7] R. L. Cruz. A calculus for network delay, Part II: Network analysis. *IEEE Trans. Inform. Theory*, 37(1):132–141, January 1991.

[8] R. L. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE J. Select. Areas Commun.*, 13(6):1048–1056, August 1995.

[9] R. L. Cruz. SCED+: Efficient management of quality of service guarantees. In *Proc. IEEE INFOCOM*, volume 2, pages 625–634, March 1998.

[10] F. H. P. Fitzek and M. Reisslein. MPEG–4 and H.263 video traces for network performance evaluation. *IEEE Network*, 15(6):40–54, November/December 2001.

[11] J.-Y. Le Boudec. Application of network calculus to guaranteed service networks. *IEEE Trans. Inform. Theory*, 44(3):1087–1096, May 1998.

[12] 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, Berlin, Germany, 2001.

[13] J. Liebeherr and D. E. Wrege. An efficient solution to traffic characterization of VBR video in quality-of-service networks. *ACM/Springer Multimedia Systems Journal*, 6(4):271–284, July 1998.

[14] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Trans. Networking*, 1(3):344–357, June 1993.

[15] 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 Trans. Networking*, 2(2):137–150, April 1994.

[16] H. Sariowan, R. L. Cruz, and G. C. Polyzos. Scheduling for quality of service guarantees via service curves. In *Proc. IEEE ICCCN*, pages 512–520, September 1995.

[17] D. E. Wrege and J. Liebeherr. Video traffic characterization for multimedia networks with a deterministic service. In *Proc. IEEE INFOCOM*, pages 537–544, March 1996.