An Introduction to SLA Calculus for the Analytical Validation of SLAs

Peter Buchholz, Sebastian Vastag
Informatik IV, TU Dortmund, Germany
{peter.buchholz,sebastian.vastag}@cs.tu-dortmund.de

Abstract

Quantitative properties of modern software systems are often defined as a part of a service level agreement (SLA) that fixes the maximal load to be submitted to a system and guarantees bounds for the response time. The evaluation of software architectures in order to validate SLAs is a challenging task since the systems tend to be complex, highly dynamic and to some extent unpredictable. Thus, there is a need for fast and abstract techniques to evaluate the performance of modern software architectures based on the information available in the SLAs.

The paper presents an efficient approach to compute bounds on the response time of composed systems based on available bounds for the load and the response times of components. The technique can be used by a user of a software architecture to validate SLAs of composed services based on SLAs of the components. It can also be used by a provider of a software architecture to validate whether additional users can be accepted or to compute required service capacities to fulfill an SLA. Finally, the approach can be a base for software brokers to determine an optimal selection of services to fulfill some SLAs including cost measures.

Keywords: Service Level Agreements, Performance Analysis, Analytical Techniques, Quantitative Validation, Capacity Planning

1 Introduction

Modern software architectures are highly distributed, component-based and service-oriented. The term *Service Oriented Architecture* (SOA) is commonly used to describe the fact that user and provider agree on specific services that are implemented by the SOA. The environment in which such a software system runs is dynamically changing and for a user to a large extent unknown. Nevertheless, there are still functional and non-functional requirements which have to be fulfilled by the system. Usually these requirements are formalized in *Service Level Agreements* (SLAs) which are contracts between user and provider that include a detailed specification of the functionality and the non-functional properties as well as a specification of acceptable user behavior. We consider the non-functional part of an SLA and in particular measures which are subsumed under the term *Quality of Service* (QoS). Thus, if we speak of an SLA in the sequel of this paper we mean the quantitative part that describes the required QoS.

From a QoS perspective the provider has to assure that her or his promises given in the SLA are observed by the user which means that the available system has enough capacity or that it is possible to add additional capacity from third parties to meet the user requirements.

This is a classical planning problem which should be solved before a system is implemented or before a service is offered to a user. On the other hand, a user may compose her or his service from several sub-services that are realized by different providers. The QoS of the combined service then has to be computed from the QoS of the constituting sub-services. This is important especially if the user becomes a provider who offers the combined service to other users. In general there is a need for some rigorous specification of QoS parameters which can be applied for capacity planning and analysis.

The analysis of QoS is a common problem of performance analysis and capacity planning. For performance analysis of computer, communication and software systems established approaches like measurements, queuing networks or simulation [1, 11, 18, 22, 27] are available. Measurements are not adequate for capacity planning because one would like to have results on a system or a situation that has not been realized yet. Model based approaches are more appropriate. Ideally models should be built from the available information which are for SOAs the SLAs at the user side. At the provider side some additional information about the underlying system and its performance are available but usually this information is incomplete because often a provider uses third party components for which also only SLAs are given. Consequently, a performance analysis approach for SOAs should be based on the QoS specification in SLAs. Usually an abstract approach is sufficient because the whole environment is dynamically changing and rough estimates or better bounds for the results are the best one can expect. It is more important to have an efficient and fast method to react quickly to parameter changes.

The mentioned requirements show that simulation is usually not the right choice to analyze SOAs, more appropriate are queuing networks (QNs) which are much more abstract and can be analyzed very efficiently as long as one assumes product form as done here. Indeed several approaches based on (extended) QNs are available to analyze SOAs and validate SLAs [15, 19, 21, 23, 29, 36]. However, the mentioned approaches often do not compute the right results and are based on assumptions and parameters that are not available in SLAs. For a QN the input parameters are the arrival and service rates. In SLAs service rates are not or only partially available, whereas usually some bound on the arrival process and the QoS of the system, often in terms of the response time, are provided. Response time is commonly a result of QN analysis and not a parameter. Since QNs analysis is based on simple analytical formulas, it is possible to reorganize the equations such that service rates are computed from arrival rates and response times. This is still not really what is needed for the analysis based on SLAs because QN analysis relies on mean values of the parameters and computes mean values of results. SLAs are based on upper bounds. The user ensures that the load provided to the system does not exceed the upper bound and the provider guarantees that the system will meet the response time bound for every load that is conform to the input bound. An average value is not a good bound since at least the short and long term behavior have to be separated. From the user perspective this means that at least the load that can be given instantaneously to the system and the maximal load over some predefined time interval have to be fixed. For the provider the maximal response time for a predefined number of service invocations and a short time deviation for a few service calls have to be defined. These quantities cannot be adequately considered in QNs even if some approximate methods exist to analyze deviations from the average behavior [9, 22]. In [25] a fluid model is used to analyze SLAs which allows one to compute bounds on the probability of violating an SLA requirement but the approach requires some information about the probabilistic behavior of the system or the workload which is usually not given in the SLA.

Analysis of systems based on bounds for the arrival and service process is a common approach for analyzing certain computer and communication networks. Based on work in the early nineties [10, 12, 13], the so called *Network Calculus* became popular in network analysis [5]. Very roughly, *Network Calculus* computes bounds on the response time and the buffer filling in QN-like models using bounds on the arrival and service process. For an easier computation of the results, a fluid approach is used. Usually deterministic bounds on result measures are computed from deterministic bounds on the arrival and service process. More recently also some extensions to derive probabilistic bounds have been published [16] but yield partially to fairly complex results which are only useful under strict assumption on the involved processes. The ideas of *Network Calculus* have as well been applied in other areas where performance guarantees are required. For the analysis of real time systems, the *Real Time Calculus* [28] and for sensor networks the *Sensor Network Calculus* [26] have been developed as extensions of *Network Calculus*.

The basis of *Network Calculus*, namely the derivation of bounds on result measures from bounds of the input parameters, describes exactly the situation of SLA analysis. However, in contrast to the problems in communication networks or real time systems, SLAs have no information about the processing capacity. Instead bounds on the response time are part of the specification and may be used to compute results like joint response times of composed services or necessary service requirements. Even if the problems of SLA and network or real time analysis are not identical, it is quite surprising that the bounding approaches have not been applied in the area. Ideas of bounding performance results have, to the best of our knowledge, not been used for capacity planning in SLA based systems. Only [14] presents an application of *Network Calculus* for the analysis of component based software but the paper only applies the available formulas for the analysis of workflows and does not further develop the approach to meet the needs of SLA based analysis.

This report is based on [33] and preliminary results in [30, 32]. Here we extend the approach in various details. In particular, a new technique to compute bounds on the departure process from bounds on the response time and the arrival process is developed and presented for the first time. A more application oriented but less detailed version of this report can be found in [8]. The main idea of the proposed approach, which is denoted as *SLA Calculus*, is to start with the quantitative specification available in SLAs, formalize this quantitative information in order to define bounds on arrival processes and response times, introduce computations to perform basic operations, like concatenation, thinning or maximization, of the processes, and derive results on composed systems. The major advantage of the approach is that it is solely based on the information which is available in the SLAs and generates an abstract model from this information. From the user perspective bounds on the results of composed services can be derived, different realizations of a SOA can be compared and it can be determined whether the restrictions on the arrival processes of sub-services are met in a composed service. A provider can compute with the approach bounds on the processing capacity from the SLAs of accepted services or he or she can determine whether another service can be added without violating SLAs. The main advantages of *SLA Calculus* are the limited need of information about the SOA, the information in SLAs is sufficient, and the often efficient computation of the results, which is based on analytical formulas only. The basic equations used in the approach are already implemented in some freely available tools [4, 35]. *SLA Calculus* requires some additional operations which are not available in other tools. Therefore a new software tool with a graphical interface is currently under development. A prototype version of the tool, named *SLA Tool* is available [3, 2]. In contrast to

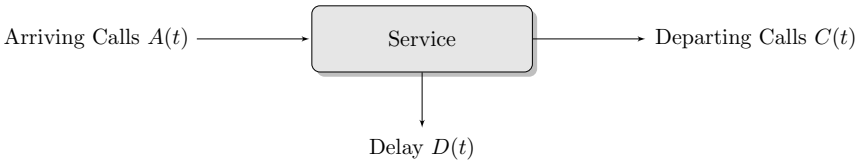

Figure 1: Model of a service.

other tools using the max/+ approach for system analysis, *SLA Tool* allows a completely graphical system specification which hides the mathematical details from a user. In this way the approach is comparable to the specification of product form queuing networks.

The remainder of the paper is organized as follows. In the next section, a formal approach to define bounds on arrivals and response times is introduced and it is shown how basic operations on these quantities are realized. In Section 3, the perspective of a user is presented. It is shown how SLAs for composed services are derived from SLAs of sub-services. Afterwards system analysis from the providers' perspective is introduced. In this case, bounds on the necessary capacity are computed and this capacity can be compared with the available capacity in order to decide whether to accept an additional customer or not. The paper ends with the presentation of an application example and some concluding remarks.

2 Specification and Analysis of SLAs

We first consider the specification and analysis of QoS parameters of a single service. Figure 1 shows the basic model view. Service calls are submitted, are processed by the service and afterwards leave. Additionally, the response time which the call spends in the service is measured. The response time will be denoted as the *delay* to be consistent with the literature on *Network Calculus* and to distinguish it from the response time in a queueing network which is different as explained below. Service calls arrive at discrete points $t_1, t_2, \ldots$ in time and each call brings some load into the system which can be interpreted as the size of the call. Size is measured and specified application specific. In a database, it describes the complexity of a query, for a compute server it specifies the number of computations and in a computer network it is measured as packet size. For other applications similar measures have to be defined to quantify calls. If all calls are identical, then they all have the same size which is the simplest case. Let the ith call arrive at time t_i, then its size is denoted by $a(t_i)$. The call is processed and leaves afterwards the service. The time between arrival and departure of a call is the delay. We describe this behavior by three processes. Arrivals are described by $A(t)$ where for a given sequence of arrivals $a(t_1), a(t_2), \ldots$, $A(0) = 0$ and $A(t) = \sum_{t_i \leq t} a(t_i)$. At time t_i $A(t)$ jumps from $A(t_i^-) = \sum_{t_j < t_i} a(t_j)$ to $A(t_i) = A(t_i^-) + a(t_i)$. $A(t)$ is non-decreasing and $A(t) = 0$ for $t < 0$. $A(t)$ includes the accumulated arrivals until time t. We denote $A(t)$ as an arrival process. Similarly the process $C(t)$ contains the accumulated departures (i.e., processed arrivals) until time t. Since departures occur after arrivals $A(t) \geq C(t)$. $C(t)$ is also non-decreasing with $C(t) = 0$ for $t < 0$. $C(t)$ is the departure process of the service. Let $b(t) = A(t) - C(t)$ be the backlog of load in the system at time t. The virtual delay equals $d(t) = \inf\{\tau \geq 0 : A(t) \leq C(t + \tau)\}$. Figure 2 shows an example for the processes $A(t)$, $C(t)$, $b(t)$ and $d(t)$. We use t for the time and x for the accumulated load in the system.

Observe that the representation in Figure 2 corresponds to the classical description of the

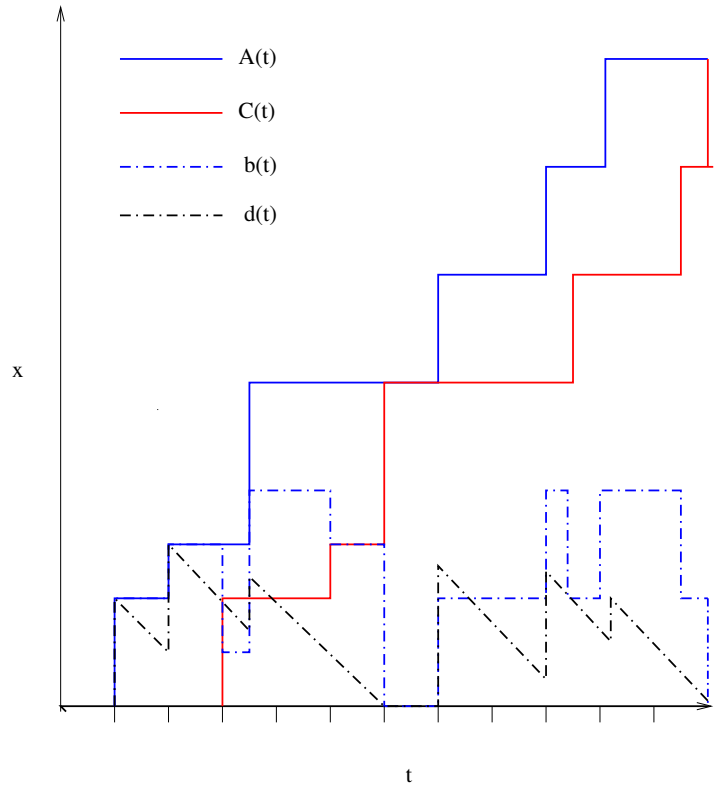

Figure 2: An example for the course of the different processes.

load in queues [17] which is often used to describe sample paths of stochastic systems.

The next process we consider is the delay process $D(t)$. The delay process measures the accumulated delay in the system which is weighted by the size of the load. For example if two calls of size 1 and 2, respectively, stay in the system for 3 time units each, the accumulated delay equals $(1+2)\cdot 3 = 9$ time units. This interpretation differs from the standard definition of response time in queuing networks [17] that defines the delay for single arrivals independently of their size. For this reason, we use the term *delay* rather than *response time*.

To explain the delay process $D(t)$, consider as a simple example a system where a single service call of size 2 arrives at time t and has a delay of Δ. Nothing else happens in the system. Then $D(\tau) = 0$ for $\tau < t$ and $D(\tau) = 2\Delta$ for $\tau \geq t$. I.e., at time τ the delay of the arriving call is immediately added to $D(t)$. This is the result one obtains when the integration over the curve defined by $D(t)$ is performed after exchanging x- and y-axis. We denote this as vertical integration. Before vertical integration can be presented, we first define the pseudo-inverse of $A(t)$ and $C(t)$ as

$$A^{-1}(x) = \inf\{\tau | A(\tau) \geq x\} \text{ and } C^{-1}(x) = \inf\{\tau | C(\tau) \geq x\}. \tag{1}$$

If ($A(t)$ (resp. $C(t)$) is strictly increasing and continuous, then the pseudo-inverse becomes an inverse, i.e., $A^{-1}(A(t)) = t$. In general $A(A^{-1}(x)) \geq x$ and $A^{-1}(A(t)) \leq t$ [5]. For completeness we define for $A(t) = 0$ for all $t > 0$, $A^{-1}(x) = \infty$ for all $x > 0$ and vice versa. Additionally, we define

$$d^{-1}(x) = C^{-1}(x) - A^{-1}(x),$$

the delay for load level x in the system. The accumulated delay $D(t)$ is then given by

$$D(t) = \int_0^{A(t)} d^{-1}(x)dx = \int_0^{A(t)} C^{-1}(x)dx - \int_0^{A(t)} A^{-1}(x)dx. \tag{2}$$

$D(t)$ can also be represented in terms of the functions $A(t)$ and $C(t)$ as

$$\begin{aligned} D(t) &= \int_0^t (A(\tau) - C(\tau))\, d\tau + \int_0^{d(t)} (A(t) - C(t+\tau))\, d\tau \\ &= \int_0^t A(\tau)d\tau + d(t)A(t) - \int_0^{t+d(t)} C(\tau)d\tau. \end{aligned}$$

Function $D(t)$ measures the delay accumulated up to time t. In most situations the delay depends on the load that is offered to the system. Therefore we define a function $F(x)$ that quantifies the delay depending on the load.

$$F(x) = \int_0^x d^{-1}(y)dy = \int_0^x C^{-1}(y)dy - \int_0^x A^{-1}(y)dy. \tag{3}$$

Of course, $F(A(t)) = D(t)$. The derivative of F is d^{-1}. In a similar way the accumulated buffer filling in $[0, t]$ equals

$$B(t) = \int_0^t b(t)dt = \int_0^t (A(t) - C(t))\, dt.$$

The derivative of $B(t)$ is $b(t)$.

Functions $D(t)$ and $F(x)$ describe the accumulated delay over sequences of service calls, the delays of single calls are weighted by the size of the calls. They include no information about the delay of a single call. However, in SLAs usually conditions on the delay of single calls are formulated. To assure such conditions, assumptions about the scheduling of calls in the service have to be made. We usually assume FCFS service such that a call arriving at time t has a delay of $d(t)$. In this case $D(t)$ and $F(x)$ are directly connected to the behavior of single service calls. The assumption of FCFS service does not strictly hold in a SOA with parallel activities but it is in most cases a sufficiently accurate approximation of the behavior.

If $b(t)$ and $d(t)$ are bounded by $b_{\max}$ and $d_{\max}$, respectively, then $D(t)$ and $F(x)$ can be upper-bounded by a linear function. We assume that upper bounds for backlog and delay exist because otherwise the SOA is unstable with delays potentially growing over all limits.

Before we continue the analysis of the behavior of service calls by defining bounds, the theoretical framework for the following results will be introduced. We consider a set $\mathcal{F}$ of causal wide-sense increasing functions [5] where a function $f : \mathbb{R} \to \mathbb{R}$ is causal if $f(t) = 0$ for $t < 0$ and it is wide-sense increasing if $f(a) \leq f(b)$ for all $a < b$. The functions $A(t)$, $C(t)$, $D(t)$ and $F(x)$ are causal and wide sense increasing.

For $f, g \in \mathcal{F}$, the following operations are defined

$$\begin{aligned} (f + g)(t) &= f(t) + g(t) \quad \text{(pointwise sum)}, \\ (f \vee g)(t) &= f(t) \vee g(t) \quad \text{(pointwise maximum)}, \\ (f \wedge g)(t) &= f(t) \wedge g(t) \quad \text{(pointwise minimum)}. \end{aligned}$$

If two arrival processes $A_1(t)$ and $A_2(t)$ arrive concurrently at some service, then $A_{12}(t) = (A_1 + A_2)(t)$ is the joint arrival process. Similarly if $A_1(t), \ldots, A_n(t)$ are potential arrival processes for some service, then an upper bound and a lower bound for the arrival stream can be computed as $A^L(t) = (A_1 \wedge A_2 \wedge \ldots \wedge A_n)(t)$ and $A^U(t) = (A_1 \vee A_2 \vee \ldots \vee A_n)(t)$, respectively.

Two additional operations are required for our analysis approach, namely min /+ convolution and deconvolution [5]. For $f, g \in \mathcal{F}$ the min /+ convolution and deconvolution are defined as

$$\begin{array}{rcll} (f \underline{\otimes} g)(t) & = & \inf_{0 \le s \le t} \{f(t-s) + g(s)\} & (\min /+ \text{ convolution}) \\ (f \underline{\oslash} g)(t) & = & \sup_{0 \le u} \{f(t+u) - g(u)\} & (\min /+ \text{ deconvolution}) \end{array}$$

In general the deconvolution is not the inverse of the convolution operation and it is not even closed in $\mathcal{F}$. However, a duality between convolution and deconvolution exists which justifies the name. For $f, g, h \in \mathcal{F}$ $f \underline{\oslash} g \le h$ if and only if $f \le g \underline{\otimes} h$. For further details about the operations we refer to the literature [5].

In a similar way convolution and deconvolution can be defined for max /+ algebra as

$$\begin{array}{rcll} (f \overline{\otimes} g)(t) & = & \sup_{0 \le s \le t} \{f(t-s) + g(s)\} & (\max /+ \text{ convolution}) \\ (f \overline{\oslash} g)(t) & = & \inf_{0 \le u} \{f(t+u) - g(u)\} & (\max /+ \text{ deconvolution}) \end{array}$$

For a functions $f \in \mathcal{F}$, the pseudo-inverse function f^{-1} is defined as in (1).

$$f^{-1}(x) = \inf \{t | f(t) \ge x\} \tag{4}$$

We extend the pseudo-inverse to general non-negative function g with $g(t) = 0$ for $t < 0$ by defining

$$g^{-1}(x) = \int_0^\infty \delta(g(t) < x) dt \tag{5}$$

where $\delta(g(t) < x) = 1$ for $g(t) < x$ and 0 else. For $g \in \mathcal{F}$ both definitions coincide.

We are often interested in specific classes of *good* functions, namely *sub-additive* and *super-additive* functions. A function $f \in \mathcal{F}$ is sub-additive if and only if $f(s+t) \le f(s) + f(t)$ for any $s, t \ge 0$ and it is super-additive if and only if $f(s+t) \ge f(s) + f(t)$ for all $s, t \ge 0$. A concave function f with $f(0) \ge 0$ is sub-additive. Similarly, a convex function f with $f(0) \le 0$ is super-additive. It can be shown that a function f is sub-additive if and only if $f \le f \underline{\otimes} f$ and similar it is super-additive if and only if $f \ge f \overline{\otimes} f$. If $f(0) = 0$ the inequality becomes an equality in both cases. Let $\delta_T(t)$ be the step function with $\delta_T(t) = 0$ for $t < T$ and $\delta_T(t) = +\infty$ for $t \ge T$ and define for $f \in \mathcal{F}$ $f^{(0)} = \delta_0$, $f^{(1)} = f$ and $f^{(n)} = f \underline{\otimes} f^{(n-1)}$. Then $\overline{f} = \inf_{n \ge 0} \left\{ f^{(n)} \right\}$ is the sub-additive closure. In a similar way the super-additive closure $\underline{f}$ can be defined. The relations $\overline{f} \le f$ and $\underline{f} \ge f$ hold. If f is already sub-additive, then $\overline{f} = f$ and for super-additive functions $\underline{f} = f$. For further details of both operations we refer to [5]. Algorithms to realize the operations for piecewise linear functions can be found in [7, 34].

In an SLA, bounds for $A(t)$ and $D(t)$ or $F(x)$ have to be formulated. Usually this is done by defining long term average arrival rates and delays and short term deviations from the long term averages. We use a similar but slightly more formal way. Our model is a fluid model where we assume that load arrives continuously to the system and delay is continuously produced by the system. An upper bound for the load arriving to the system is defined by a sub-additive function $\alpha^U(t)$ and an upper bound for the delay at time is defined by a sub-additive function $\Psi^U(t)$. For the delay at load level x, the sub-additive function $\Phi^U(x)$ is

an upper bound. We denote the functions as upper arrival and delay curve, respectively. Consequently, an upper arrival or delay curve is a specific function that defines an upper bound for the arrival or delay process. An arrival stream $A(t)$ is conform to an upper arrival curve $\alpha^U(t)$ if and only if for every $0 \leq s < t$, $A(t) - A(s) \leq \alpha^U(t-s)$ or equivalently $A \leq A \underline{\otimes} \alpha^U$. Since we have discrete arrivals $\alpha^U(0) \geq a_{\max}$ where $a_{\max}$ is the maximal size of an arriving service call. Similarly the delay function $D(t)$ is conform to the upper bound $\Psi^U(t)$ if and only if for every $0 \leq s < t$, $D(t) - D(s) \leq \Psi^U(t-s)$ or equivalently $D \leq D \underline{\otimes} \Psi^U$ and $F(x)$ is conform to the upper bound $\Phi^U(x)$, if and only if $F(x) - F(y) \leq \Phi^U(x-y)$ (for $0 \leq y \leq x$) or equivalently $F \leq F \underline{\otimes} \Phi^U$. The upper arrival and delay curves can be substituted by their sub-additive closures $\overline{\alpha}^U$, $\overline{\Psi}^U$ and $\overline{\Phi}^U$, if the functions are not sub-additive. To avoid an overloading of notation we usually do not print the bars and assume that the functions are sub-additive.

For α^U, Ψ^U and Φ^U usually simple functions are used to keep the definition understandable and allow an efficient analysis. The class of piecewise linear functions is easy to handle and can be used to approximate arbitrary concave functions. To specify piecewise linear curves we use a specification similar to the one used in the RTC-Toolbox [35] and define a sequence of segments (x_i, y_i, s_i) $(i = 1, \ldots, L)$. We assume $x_1 = 0$ and $x_i \leq x_{i+1}$ specifying the values at the x axis. $y_i \geq 0$ specify the corresponding values $f(x_i) = y_i$ and $s_i \geq 0$ is the slope of the linear segment starting at x_i. For some $x \geq 0$ $f(x) = \max_{i \in \{1,\ldots,L\}, x_i \leq x} \{y_i + s_i(x - x_i)\}$ and $f'(x) = s_i$ is the slope at x. Although $s_i \geq 0$ functions need not be non-decreasing since we may have the situation that $y_i < y_{i-1} + s_{i-1}(x_i - x_{i-1})$ which means that the functions makes a step downwards at x_i. Such functions occur if bounds for the output of a service are considered where calls that arrive later have a smaller delay than calls arriving earlier which means that there is some overtaking inside the service. A piecewise linear function is continuous if $y_{i+1} = y_i + s_i(y_{i+1} - y_i)$ for all $1 \leq i < L$. A continuous function is concave if $s_i \geq s_{i+1}$ and it is convex if $s_i \leq s_{i+1}$.

For the specification of quantitative properties of SLAs it is sufficient to consider piecewise linear functions with finitely many segments. This implies that for $x \geq x_L$ the slope of the function equals s_L, i.e., the function becomes linear. Most operations we use result in piecewise linear functions with a finite number of linear segments, if applied to those functions. This does not necessarily hold for the sub-additive or super-additive closure. Both functions may result in picewise linear functions with an infinite number of segments. In this case, it is possible to use upper or lower bounds, consisting of finitely many segments, with some loss of tightness. In principle the approach can also be used for functions where piecewise linear segments are periodically concatenated [7, 34] which enlarges the class of functions. Periodic functions are important for real time systems and some computer networks where a detailed specification of the behavior of a system is available. However, for SLA modeling the simpler piecewise linear functions with a finite number of segments are sufficient.

We assume that α^U, Φ^U and Ψ^U are specified by continuous concave piecewise linear functions and consider first bounds for the arrival process resulting from a mixture of affine arrivals curves [5].

Usually either Ψ^U or Φ^U are used in an SLA. If Ψ^U is used, then a contract on the delay per time is made. As long as the arrival process is conform to the upper arrival bound $\alpha^U(t)$, the accumulated delay in $[0, t)$ is not larger than $\Psi^U(t)$. The definition of Ψ^U implies that for a low load, the delay per load unit can grow without violating the delay contract. Alternatively, Φ^U puts a bound on the delay per load unit which is often more appropriate.

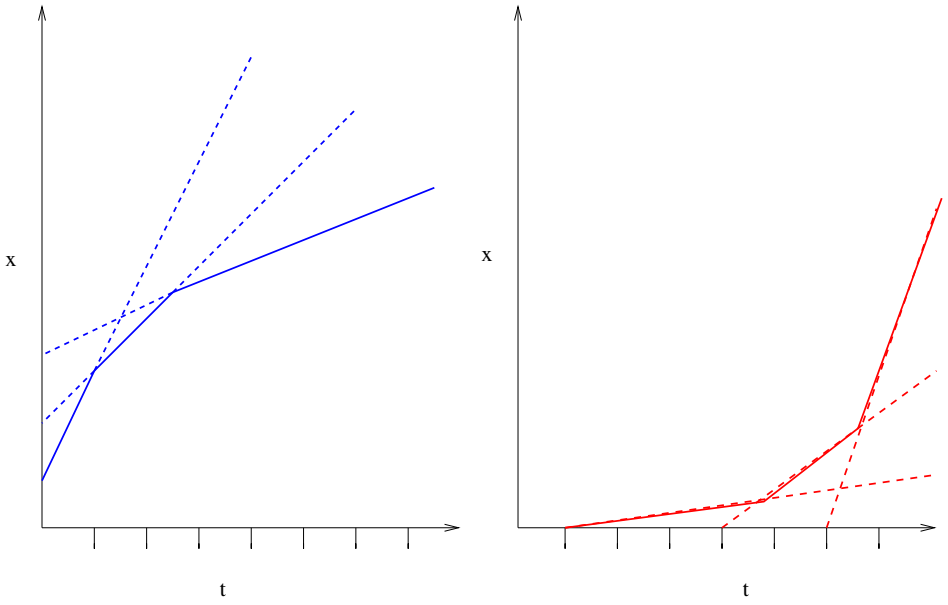

Figure 3: Upper bounding curve and lower bounding curve with three segments.

In this case, the system keeps at least its speed if the load shrinks. If Φ^U is defined for the maximum load α^U and the delay is proportional to the offered load, then $\Phi^U(\alpha^U(t)) = \Psi^U(t)$ is the corresponding load dependent delay function.

Example 1. *A deterministic arrival process which generates one service call of size* 1 *every* Δ *time units can be described by* $\alpha^U = (0, 1, \Delta^{-1})$. *If the arrival process has a jitter of at most* σ *(*$\sigma < \Delta$*), then it can be described by two segments, namely,*
$\alpha^U = \big((0, 1, (\Delta - \sigma)^{-1}), (\Delta - \sigma, 2, \Delta^{-1})\big)$.
The worst case in this situation occurs if the first call arrives at time 0 *and the remaining calls arrive as early as possible which means that the* i*th call arrives at time* $i \cdot \Delta - \sigma$.

If we assume that the calls arrive every Δ *time units but the sizes of calls are independently and uniformly distributed in* $[0.5, 1.5]$, *then a curve that considers the worst case is defined by* $(0, 1.5, 1.5/\Delta)$. *However, this curve is very pessimistic because a long term arrival rate of* 1.5 *load units is assumed whereas the average rate is* 1. *Thus, it is usually sufficient to assume that only the first* i *arrivals are of size* 1.5 *and consider afterwards the average load of* 1. *This results in the arrival curve* $\big((0, 1.5, 1.5/\Delta), (i\Delta, 1.5i, \Delta^{-1})\big)$. *Of course, with some probability, the arrival process may exceed the upper bound. This occurs if the sum of* k $(> i)$ *uniformly* $[0.5, 1.5]$ *distributed random variables is larger than* $1.5i + (k - i)$. *Since the sum converges towards a normal distribution with mean* k *and standard deviation* $\sqrt{k/12}$ *due to the central limit theorem, the probability can be approximated. For* $k \to \infty$, *the upper bound is exceeded with a probability of almost* 0.5 *for any* i. *However, in SLAs one usually allows some violation of the SLA, if it occurs rarely and with a very small probability. Thus, the time window* $k\Delta$ *and a small probability for violating the upper bound can be defined. In this case, the value of* i *can be computed from the quantiles of the standard normal distribution.*

Bounds for the delay process are defined by the same kind of function. The accumulated delay is then bounded by a piecewise linear function Ψ^U and $\psi(t) = (\Psi^U)'(t)$ the delay in the

system for load arriving at t.

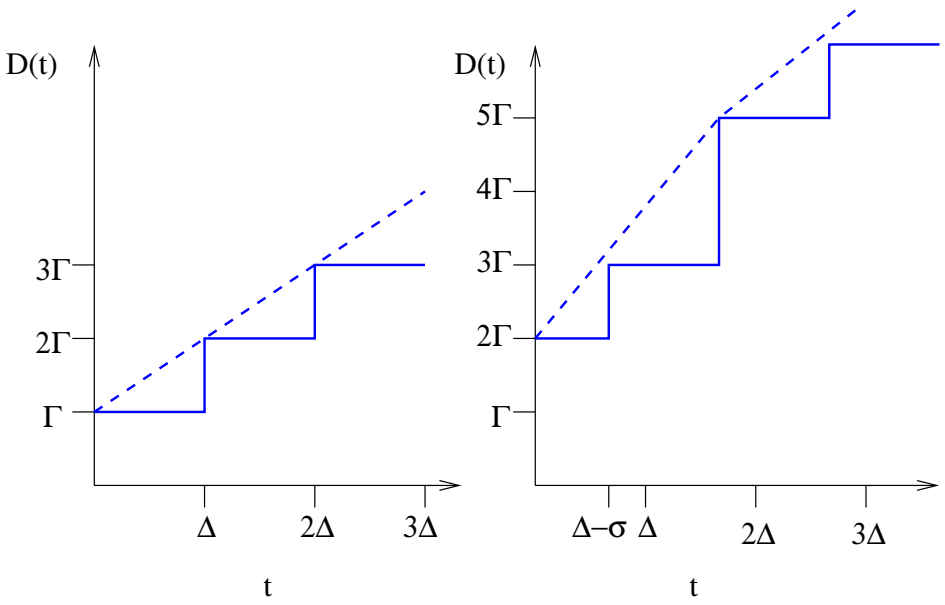

Figure 4: Delay curves for the two examples

Example 2. *For an arrival process where every Δ time units a call of size 1 arrives and leaves the system with a delay of Γ, the delay curve can be bounded by $(0, \Gamma, \Gamma/\Delta)$. The corresponding curves are shown on the left side of Figure 4. If the delay is proportional to the load, we obtain $\Phi^U(x) = (1+x)\Gamma$.*

As a second example we consider a periodic arrival process with period Δ and jitter σ ($\sigma < \Delta/2$). Arriving calls have an alternating delay of Γ and 2Γ time units. The maximal delay curve of this system is shown as the solid line in the right graph of Figure 4. A piecewise linear upper bounding curve Ψ^U for this process is given by $\left(\left(0, 2\Gamma, \frac{3\Gamma}{2\Delta-\sigma}\right), \left(2\Delta-\sigma, 3\Gamma, \frac{5\Gamma}{2\Delta}\right)\right)$. This curve is shown by the dashed line in the right graph of Figure 4.

Available semi-formal and even informal specifications of SLAs bounding the arrival process and the delays of service calls can be transformed into piecewise linear functions in a natural way. It is also possible to derive bounding curves from measurements of arrivals and delays. A conservative approach generates the curves as an upper bound for all measurements, more sophisticated approach allow some violations of the SLAs. First ideas to generate arrival and delay curves for SLAs from measured data can be found in [31] and are based on statistical techniques.

Apart from upper bounds, one can also define lower bounds. A natural lower bound for the arrival stream and the delay is 0. However, in some cases, like in periodic systems, one can also define a lower arrival and delay curve. Like for upper bounds, piecewise linear functions are used but we now assume that the functions are concave and $y_1 = 0$. Figure 3 shows on the right side a lower bounding curve with 3 segments. We denote the corresponding curves by α^L, Ψ^L and Φ^L, respectively. Observe that we assume $\alpha^L(0) = 0$.

If α^L and α^U describe a lower and an upper arrival curve, then $\alpha^L(t) \leq \alpha^U(t)$ for all $t \geq 0$ which implies $\left(\alpha^L\right)'(t) \leq \left(\alpha^U\right)'(t)$ if α^L is convex and α^U concave.

Example 3. *We describe lower arrival curves for the first examples given above. For a deterministic arrival process that generates arrivals of size* 1 *every* Δ *time units, the lower curve equals* $\alpha^L = \big((0,0,0),(\Delta,0,\Delta^{-1})\big)$. *The curve becomes non-zero at time* Δ. *If the arrival process has a jitter of at most* σ *(*$\sigma < \Delta$*), we obtain* $\alpha^L = \big((0,0,0),(\Delta+\sigma,0,\Delta^{-1})\big)$. *If calls arrive every* Δ *time units and the sizes are specified by independent uniform* $[0.5, 1.5]$ *distributed random variables. Then* $\alpha^L = ((0,0,0),(\Delta,0,0.5/\Delta))$ *is a conservative lower bound. With the same arguments as for the upper bound, one can also define a lower bounding curve that switches to an average slope of* 1 *after* i *arrivals.*

A service generates a departure process $C(t) \leq A(t)$. We now compute bounds for the departure process from the bounds of the arrival process and the delay. Let α^L, α^U, Ψ^L and Ψ^U be the lower and upper bounds of the arrival process and the delay. We assume that the upper bounds are strictly increasing, continuous, concave and the lower bounds are non-decreasing continuous convex. This assumption is not restrictive in systems which potentially run for an infinite time such that one can assume an increasing number of arrivals and an increasing accumulated delay over time. In this case, the pseudo-inverse of the upper bounds becomes an inverse and the same holds for the part of the lower bounds where the function becomes non-zero.

For some non-decreasing piecewise linear function (x_i, y_i, s_i) $(1 \leq i \leq l)$, the pseudo-inverse equals (y_i, x_i, s_i^{-1}) where $s_i^{-1} = 0$ for $s_i = 0$. The inverse of the concave upper bound is the convex lower bound and the inverse of the convex lower bound is the concave upper bound. Since $\alpha^L(t) \leq A(t) \leq \alpha^U(t)$ for all $t \geq 0$, also $\left(\alpha^L\right)^{-1}(x) \geq A^{-1}(x) \geq \left(\alpha^U\right)^{-1}(x)$ has to hold for all $x \geq 0$ which also implies $\int_0^x \left(\alpha^L\right)^{-1}(y)dy \geq \int_0^x A^{-1}(y)dy \geq \int_0^x \left(\alpha^U\right)^{-1}(y)dy$ for all $x \geq 0$. Since $\alpha^U(t) \geq A(t+u) - A(u) \geq \alpha^L(t)$ for every $u \geq 0$, $\left(\alpha^U\right)^{-1}(x) \leq A^{-1}(x+y) - A^{-1}(y) \leq \left(\alpha^L\right)^{-1}(x)$ for every $y \geq 0$ and the same relations hold for the integrals over the functions. By definition we have $\Psi^U(t) \geq D(t+u) - D(u) \geq \Psi^L(t)$ for all $u \geq 0$. The derivatives $\psi^U(t) = \left(\Psi^U\right)'(t)$ and $\psi^L(t) = \left(\Psi^L\right)'(t)$ of the delay functions describe the bounds for the delay of a call arriving at time t. Since the functions are piecewise linear, delays are piecewise constant. Since Ψ^U is concave, ψ^U is non-increasing and since Ψ^L is convex, ψ^L is non-decreasing.

The accumulated delay at time t is computed using (2), (3) allows us to represent the delay depending on the load x. This representation is used in the following equation where a lower bound for the integral of $\int_z^{z+x} C^{-1}(y)dy$, i.e., the time when load x leaves the service, is computed.

$$\begin{aligned}
&\inf_{z\geq 0}\left\{\int_z^{z+x}\left(C^{-1}(y)-A^{-1}(z)\right)dy\right\} &&= \\
&\inf_{z\geq 0}\left\{D(A^{-1}(x+z))-D(A^{-1}(z))+\int_z^{x+z}\left(A^{-1}(y)-A^{-1}(z)\right)dy\right\} &&\geq \\
&\inf_{z\geq 0}\left\{D(A^{-1}(x+z))-D(A^{-1}(z))\right\}+\inf_{z\geq 0}\left\{\int_z^{x+z}\left(A^{-1}(y)-A^{-1}(z)\right)dy\right\} &&\geq \\
&\Psi^L\left(\left(\alpha^U\right)^{-1}(x)\right)+\int_0^x\left(\alpha^U\right)^{-1}(y)dy
\end{aligned} \tag{6}$$

The relation holds due to (2) because

$$D\left(A^{-1}(x+z)\right)-D\left((A^{-1}(z)\right)= \int_z^{z+x} C^{-1}(y)dy - \int_z^{z+x} A^{-1}(y)dy\ .$$

If z is fixed, $A^{-1}(z)$ is constant and the derivative can be computed.

$$C^{-1}(x) \geq \psi^L\left(\left(\alpha^U\right)^{-1}(x)\right) + \left(\alpha^U\right)^{-1}(x) \tag{7}$$

The previous equations introduce a lower bound for the inverse of the departure process. If Ψ^L and α^U are piecewise linear, the lower bound for the integral is a quadratic function and the lower bound for $C^{-1}(x)$ is piecewise linear. Let (x_i, y_i, s_i) be the resulting lower bound for $C^{-1}(x)$, then (y_i, x_i, s_i^{-1}) is an upper bound for $C(t)$.

In a similar way, a lower bound for $C(t)$ can be computed if we assume that α^L is non-zero, otherwise the lower bound of the departure process becomes zero too.

$$\begin{array}{ll} \sup_{z\geq 0}\left\{\int_z^{z+x}\left(C^{-1}(y) - A^{-1}(z)\right)dy\right\} & = \\ \sup_{z\geq 0}\left\{D(A^{-1}(x+z)) - D(A^{-1}(z))\int_z^{x+z}\left(A^{-1}(y) - A^{-1}(z)\right)dy\right\} & \leq \\ \sup_{z\geq 0}\left\{D(A^{-1}(x+z)) - D(A^{-1}(z))\right\} + \sup_{z\geq 0}\left\{\int_z^{x+z}\left(A^{-1}(y) - A^{-1}(z)\right)dy\right\} & \leq \\ \Psi^U\left(\left(\alpha^L\right)^{-1}(x)\right)) + \int_0^x \left(\alpha^L\right)^{-1}(y)dy & \end{array} \tag{8}$$

Such that

$$C^{-1}(x) \leq \psi^U\left(\left(\alpha^L\right)^{-1}(x)\right) + \left(\alpha^L\right)^{-1}(x). \tag{9}$$

Since we can assume that $\alpha^L(0) = 0$, the lower bound can be computed for every $x \geq 0$ using the equation. Again the resulting bound for $C^{-1}(x)$ is piecewise linear if the bounds for the arrival process and accumulated delay are piecewise linear. Let (x_i, y_i, s_i) be the upper bound for $C^{-1}(x)$ and its inverse $(y_i, x_i, s_i^{-1}))$ is a lower bound for $C(t)$.

Bounds based on Φ^L and Φ^U are computed in exactly the same way as

$$\begin{array}{l} \inf_{z\geq 0}\left\{\int_z^{z+x}\left(C^{-1}(y) - A^{-1}(z)\right)dy\right\} \geq \Phi^L(x) + \int_0^x \left(\alpha^U\right)^{-1}(y)dy, \\ C^{-1}(x) \geq \phi^L(x) + \left(\alpha^U\right)^{-1}(x) \end{array} \tag{10}$$

and

$$\begin{array}{l} \sup_{z\geq 0}\left\{\int_z^{z+x}\left(C^{-1}(y) - A^{-1}(z)\right)dy\right\} \leq \Phi^U(x) + \int_0^x \left(\alpha^L\right)^{-1}(y)dy, \\ C^{-1}(x) \leq \phi^U(x) + \left(\alpha^L\right)^{-1}(x) \end{array} \tag{11}$$

where $\phi^L = \left(\Phi^L\right)'$ and $\phi^U = \left(\Phi^U\right)'$. Again we assume that ϕ^L is non-decreasing and ϕ^U is non-increasing. The bounds (10) and (11) are usually much tighter than the bounds computed via (6) and (8) since the functions Φ^L and Φ^U define a correspondence between load and delay, whereas Ψ^L and Ψ^U define a time-dependent delay, independently of the load, only a maximum load is used. In the sequel we use the bounds Φ^L and Φ^U for the computations.

The bounds for the departure process $C(t)$ are denoted as γ^{L-} and γ^{U+}, respectively.

$$\left(\gamma^{U+}\right)^{-1}(x) = \phi^L(x) + \left(\alpha^U\right)^{-1}(x) \text{ and } \left(\gamma^{L-}\right)^{-1}(x) = \phi^U(x) + \left(\alpha^L\right)^{-1}(x). \tag{12}$$

If Φ^L and Φ^U are piecewise linear functions, then ϕ^L and ϕ^U are constants which, however, may depend on x, i.e. $\phi^L(x) = (\Phi^L)'(x)$ and $\phi^U(x) = (\Phi^U)'(x)$. To compute the functions

γ^{L-} and γ^{U+}, $\left(\gamma^{L-}\right)^{-1}$ and $\left(\gamma^{U+}\right)^{-1}$ have to be inverted. As long as the functions are non-decreasing, the pseudo-inverse can be easily computed for piecewise linear functions.

If Φ^L is piecewise linear and convex, as assumed here, then ϕ^L is non-decreasing and the same holds for $\left(\alpha^U\right)^{-1}$ such that the pseudo inverse of $\left(\gamma^{U+}\right)^{-1}$ can be computed. The function is usually not continuous if $\lim_{y \nearrow x} \phi^L(y) \neq \phi^L(x)$ (i.e., at points where the slope of the linear segments building Φ^U changes). Function γ^{U+} describes departures from the system starting with an empty system and arrivals specified by α^U. Arrivals $\alpha^U(0)$ have to be delayed by ϕ^L according to the lower delay bound. This means that in the interval $[0, \phi^L(0))$ no departures at all occur and then the arrivals at time 0 leave the system.

The situation is different for $\left(\gamma^{L-}\right)^{-1}$. If Φ^U is concave, as assumed here, and consists of more than one segment, ϕ^U is decreasing at those points where the slope of Φ^U changes. Thus, the pseudo-inverse cannot be computed with (4). However γ^{L-} can be computed with (5) because $\gamma^{L-}(t)$ equals the load that has left the system under arrival stream α^L with delay $\phi^U(x)$ for load x. Let $d(t) = \phi^U\left(\alpha^L(t)\right)$, the delay of load arriving at time t under the assumption that α^L specifies the arrivals. $d(t)$ is like ϕ^U piecewise constant and non-increasing in t. For each $t \geq 0$ exists a finite number of disjoint intervals $T_i^t = (t_i^-, t_i^+]$ such that $\tau + d(\tau) < t$ if and only if $\tau \in T_i^t$ for some $i \in \{1, \ldots, I^t\}$, i.e. load that arrived in T_i has left the system at time t. Let I^t be the number of intervals for time t, then

$$\gamma^{L-}(t) = \sum_{i=1}^{I^t} \alpha^L(t_i^+) - \alpha^L(t_i^-). \tag{13}$$

For piecewise linear functions α^L also γ^{L-} is piecewise linear.

Example 4. *As an example we consider a service with the bounding curves* $\alpha^L = ((0,0,0),(1,0,1),(2,1,2))$, $\alpha^U = ((0,1,0),(1,1,3),(2,4,2))$, $\Phi^L = ((0,0,1),(1,1,2))$ *and* $\Phi^U = ((0,0,3),(1,3,2))$. *Then* $\phi^L = ((0,1,0),(1,2,0))$ *and* $\left(\alpha^U\right)^{-1} = ((1,1,\frac{1}{3}),(4,2,\frac{1}{2}))$ *such that*
$\left(\gamma^{U+}\right)^{-1} = ((0,1,0),(1,1,0),(1,3,\frac{1}{3}),(4,4,\frac{1}{2}))$ *and*
$\gamma^{U+} = ((0,0,0),(1,0,0),(1,1,0),(3,1,3),(4,4,2))$[1].

For the computation of γ^{L-}, *we have* $\left(\gamma^{L-}\right)^{-1}(x) = \phi^U(x) + \left(\alpha^L\right)^{-1}(x)$ *with* $\left(\alpha^L\right)^{-1} = (0,1,1),(1,2,\frac{1}{2})$ *and* $\phi^U = ((0,3,0),(1,2,0))$. *This results in* $\left(\gamma^{L-}\right)^{-1} = ((0,4,1),(1,4,\frac{1}{2}))$, *a function which makes a steps downwards at* $x = 1$ *such that the pseudo inverse cannot be computed with (4. By applying (5), we obtain* $\gamma^{L-} = ((0,0,0),(4,0,3),(5,3,2))$. *At time* $t = 4 + \epsilon$ *(*$0 \leq \epsilon < 1$*), load departs that arrived at time* $1 + \epsilon$ *(and had a delay of* 3*) and load that arrived at* $2 + \epsilon$*) (and had a delay of* 2*). At time* $5 + \nu$ *(*$0 \leq \nu$*) load departs that arrived at time* $3 + \nu$ *with a delay of* 2.

The functions γ^{U+} and γ^{L-} describe bounds for possible departure processes that can be observed for the system with an arrival process bounded by α^L and α^U. For the departure process $C(t)$ the relation $\gamma^{L-}(t) \leq C(t) \leq \gamma^{U+}(t)$ holds. If γ^{L-} is super-additive, then it is a lower departure curve since it considers the worst case, an empty system, a minimal arrival process and a maximal delay $\phi^U(0)$ ($\geq \phi^U(x)$ for all $x \geq 0$). The situation is different for $\gamma^{U+}(t)$. The function delays the input by $\phi^L(0)$ ($\leq \phi^L(x)$ for all $x \geq 0$) which means that in

[1] We show explicitly subsequent steps like $(1,0,0)$ and $(1,1,0)$ to indicate where the curve jumps from one level to a higher one. Usually this information is redundant.

the interval $[0, \phi^L(0))$ no departures occur at all and it does not consider the cases that calls are backlogged.

To obtain an appropriate upper bound for the departure process, we have to introduce an additional assumption. We assume the the system shows a monotonic behavior. Thus, if $A_1(t)$ and $A_2(t)$ are two arrival processes and $D_1(t)$, $D_2(t)$ are the corresponding delay processes, then $A_1(t) - A_1(s) \geq A_2(t) - A_2(s)$ for all $t \geq s \geq 0$ implies $D_1(t) \geq D_2(t)$ for all $t \geq 0$. Furthermore, for the system with arrival process $\alpha^U(t)$ backlog $b(t)$ and delay $d(t)$ are maximal, i.e., no arrival process $A \leq A \underline{\otimes} \alpha^U$ exists where $b_A(t) > b_{\alpha^U}(t)$ or $d_A(t) > d_{\alpha^U}(t)$ for some t. $b_A(t)$, $d_A(t)$ and $b_{\alpha^U}(t)$, $d_{\alpha^U}(t)$ are the backlog and delay under arrival process A and α^U, respectively. Similarly we assume that no arrival process $A \geq \alpha^L \underline{\otimes} A$ exists such that $b_A(t) < b_{\alpha^L}(t)$ or $d_A(t) < d_{\alpha^L}(t)$. Both assumptions are reasonable and hold in most systems.

With these assumptions, backlog and delay can be bounded and an upper bound for the departure process can be computed by considering the system with arrival process α^U. Load arriving at time s will be for sure in the system at some time t ($s < t$) if $s + \phi^L(\alpha^U(s)) \geq t$, it will be potentially in the system if $s + \phi^U(\alpha^U(s)) \geq t$. By assumption ϕ^L and ϕ^U are piecewise constant. Since ϕ^L is non-decreasing for each t exists some value $\phi^-(t)$ such that for all $s \in [\phi^-(t), t)$: $s + \phi^L(s) \geq t$. Thus, $b^-(t) = \alpha^U(t) - \alpha^U(t - \phi^-(t))$ is a lower bound for the backlog of the system with arrival process α^U. The situation is more complex for the upper bound. Since ϕ^U is non-increasing we may have the situation that calls arriving later have a smaller delay than calls arriving earlier and leave the system before. However, for each $t \geq 0$ exists a finite number of disjoint intervals $\mathcal{I}_i^t = [\phi_i^L(t), \phi_i^U(t))$ $(i = 1, \ldots, I_t)$ such that for each $s \in \mathcal{I}_i^t$, $s + \phi^U(s) \geq t$. Then

$$b^+(t) = \sum_{i=1}^{I_t} \left(\alpha^U(\phi_i^U(t)) - \alpha^U(\phi_i^L(t)) \right)$$

is an upper bound for the backlog. Thus, at time t the backlog will be in the interval $[b^-(t), b^+(t)]$ and $b^+(t) - b^-(t)$ is the load that can immediately leave the system. $b^+(t)$ is the backlog of the system under arrival process α^U and delay process Φ^L which results in departure process γ^{U+}. Similarly $b^-(t)$ is the backlog of the system under arrival process α^U and delay process Φ^U resulting in departure process

$$\left(\gamma^U\right)^{-1}(x) = \phi^U(x) + \left(\alpha^U\right)^{-1}(x). \tag{14}$$

Then $b^+(t) - b^-(t) = \gamma^{U+}(t) - \gamma^U(t)$. Let

$$t^* = \sup_t \left\{ t : \forall s \leq t, s + \phi^U(\alpha^U(s)) \geq t \right\}. \tag{15}$$

At time t^* the first load leaves the system with arrival process α^U delay process Φ^U. We now show that the difference $b^+(t) - b^-(t)$ becomes maximal for $t = t^*$ such that γ^{U+} shifted by t^* becomes the upper departure curve. Let

$$\eta^L(t) = \inf \left\{ s : s + \phi^L(s) \geq 0 \right\} \text{ and } \eta^U(t) = \inf \left\{ s : s + \phi^U(s) \geq 0 \right\}.$$

η^L and η^U are non-decreasing because $s + \phi^{L/U} \geq t \Rightarrow s + \phi^{L/U} \geq t - \epsilon$ for $\epsilon > 0$ and $\eta^U(t) \leq \eta^L(t)$ because $\phi^L(s) \leq \phi^U(s)$ for all $s \geq 0$. Furthermore the difference $\eta^L(t) - \eta^U(t)$ is non-decreasing for $\eta^U(t) > 0$ because the difference $\phi^U(t) - \phi^L(t)$ is non-increasing. If

$s + \phi^L(s) \geq t \Rightarrow s + \epsilon + \phi^L(s+\epsilon) \geq t$ because $\phi^L(s)$ is non-decreasing. However, this does not hold for $\eta^U(t)$ because $\phi^U(t)$ is non-increasing. Let $\delta(s+\phi^U(s) \geq t) = 1$ for $s+\phi^U(s) \geq t$ and 0 otherwise. Then

$$\begin{aligned} b^-(t) &= \int_{\eta^L(t)}^{t} \alpha^U(s)ds \\ b^+(t) &= \int_{\eta^U(t)}^{t} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds = \int_{\eta^L(t)}^{t} \alpha^U(s)ds + \int_{\eta^U(t)}^{\eta^L(t)} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds. \end{aligned}$$

The last equality holds because $\phi^U(s) \geq \phi^L(s)$. This implies

$$b^+(t) - b^-(t) = \int_{\eta^U(t)}^{\eta^L(t)} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds$$

For $\eta^U(t) > 0$

$$\int_{\eta^U(t)}^{\eta^L(t)} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds \geq \int_{\eta^U(t)}^{\eta^L(t)} \delta(s+\epsilon+\phi^U(s+\epsilon) \geq t)\alpha^U(s+\epsilon)ds$$

such that the difference $b^+(t) - b^-(t)$ is non-increasing if $\eta^U(t) > 0$. For $\eta^U(t) = 0$ we have $\phi^U(0) > t + \epsilon$ and

$$\begin{aligned} &\int_{0}^{\eta^L(t)+\epsilon} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds = \\ &\int_{0}^{\eta^L(t)} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds + \int_{\eta^L(t)}^{\eta^L(t)+\epsilon} \delta(s+\phi^U(s) \geq t)\alpha^U(s)ds \end{aligned}$$

which shows that the difference is non-decreasing. This implies that the largest backlog is achieved for $t = t^*$ and an upper departure curve equals

$$\tilde{\gamma}^{U+}(t) = \begin{cases} 0 & \text{for } t < 0, \\ \gamma^{U+}(t+t^*) & \text{for } t \geq 0. \end{cases} \tag{16}$$

Since $b^-(t)$ is non-decreasing, for a concave upper arrival curve the maximal backlog of load that can leave the system immediately equals $b^+(t^*) - b^-(t^*) = \gamma^{U+}(t^* - \epsilon)$ for an arbitrarily small $\epsilon > 0$. The latter equality holds since $\gamma^{U+}(t) = 0$ for $t < t^*$ by definition of t^*.

The departure process γ^U of the service under maximal delay has been defined in (14). To define a valid upper departure curve bound we use the result from [5, Theo. 1.2.2]) that for an output stream $R(t)$, $R \oslash R$ is an upper curve and even more, it is the smallest upper curve. Thus, we define

$$\tilde{\gamma}^U = \gamma^U \oslash \gamma^U. \tag{17}$$

For completeness we define also the minimal departure process under minimal delay as

$$\left(\gamma^L\right)^{-1} = \phi^L(x) + \left(\alpha^L\right)^{-1}(x). \tag{18}$$

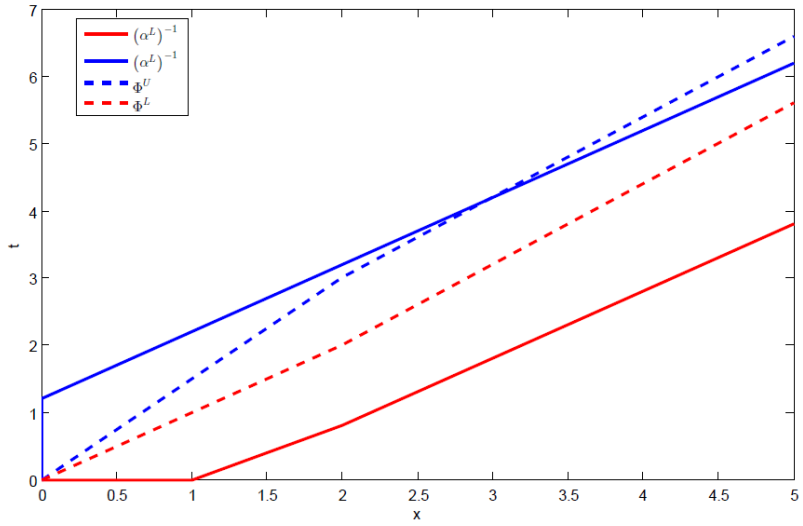

Figure 5: Arrival and delay curves.

Example 5. *We consider a system where calls of an identical size of* 1 *arrive with a periodic stream with an inter-arrival time of* $\Delta = 1$ *and a jitter of* 0.2*. A possible upper arrival curve is* $\alpha^U = ((0,1,1.25),(0.8,2,1))$*. A lower arrival curve is given by* $\alpha^L = ((0,0,0),(1.2,0,1))$*. Let* $\Phi^U = ((0,0,1.5),(2,3,1.2))$ *and* $\Phi^L = ((0,0,1),(2,2,1.2))$ *be the upper and lower delay curves, respectively.*

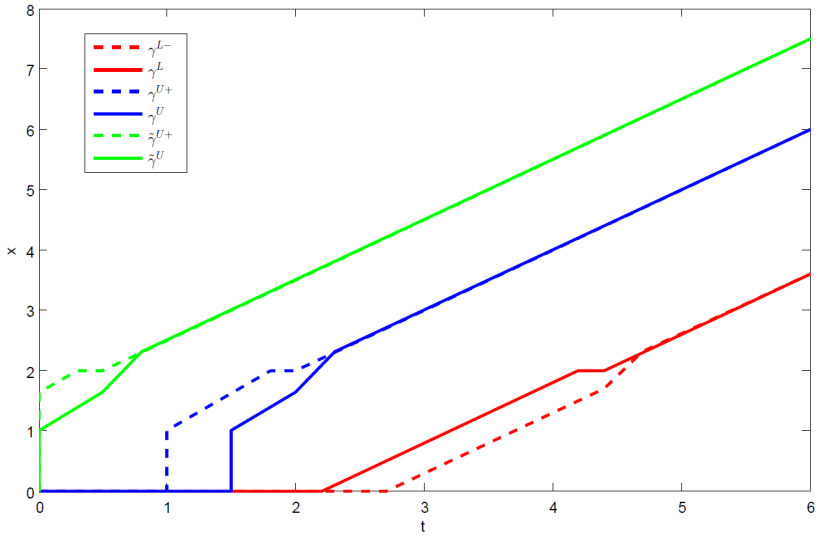

Figure 6: Bounds for the departure processes.

Figure 5 shows the (pseudo-)inverse arrival curves and the delay curves for the example. Using (12,14,18) the inverse curves for the departure process can be computed which are inverted to obtain γ^L, γ^{L-}, γ^U, γ^{U+}, $\tilde{\gamma}^U$ *and* $\tilde{\gamma}^{U+}$*. The curves for the departure process are shown in Figure 6. The departure curves* γ^L *and* $\tilde{\gamma}^U$ *are super- respectively sub-additive.*

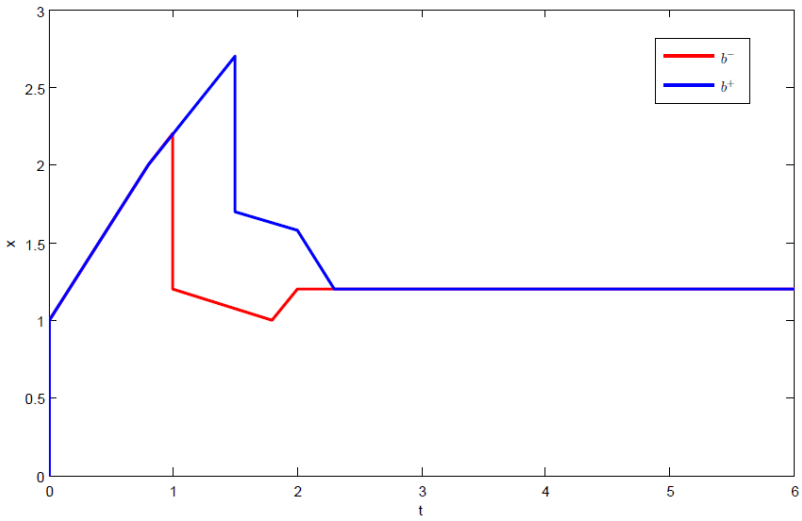

Figure 7: Bounds for the backlog under a maximal arrival process.

Bounds for the backlog of the system under the maximal arrival process defined by α^U *are shown in Figure 7. An upper bound for the backlog is* 2.7 *and the maximal amount of load that can immediately leave the system is* 1.75 *(maximal difference between* b^+ *and* b^-*). The backlog under maximal arrivals converges towards* 1.2 *which is the amount of load that arrives during the delay of* 1.2*. The lower bound for the backlog in the system is* 0 *since the maximal time without arrivals is* 1.2 *and the minimal delay is also* 1.2*.*

3 The User Perspective of Composed SLAs

After the basic method to specify SLAs for services has been defined, we now show how to use the approach from the user's side of a SOA. In the subsequent section the viewpoint of a provider will be taken.

An arrival process $A(t)$ is conform to an upper arrival curve α^U if and only if $A(t) \leq (A \underline{\otimes} \alpha^U)(t)$ for all $t \geq 0$ [5]. If the arrival process is not conform to the upper arrival curve but the SLA allows sufficient capacity (i.e, $\lim_{t\to\infty} \left(A(t) - \alpha^U(t)\right) \leq \Delta < \infty$), then the arrival stream can be smoothed using what is called in the network world a greedy shaper [5]. Greedy shapers with piecewise linear shaping curves can be realized by the concatenation of leaky buckets. The behavior of a greedy shaper is as follows. If a service call arrives which is conform to the upper arrival limit, it is immediately passed to the service. If an arriving service call violates the upper bound, it is delayed by a minimum amount of time such that the stream which is passed to the service observes the arrival bounds. Calls in the shaper are served in FCFS order. The departure process of the shaper is then given by $C(t) = (A \underline{\otimes} \alpha^U)(t)$. It is conform with the upper arrival curve α^U of the service. However, the price is an additional delay

$$d(t) = C^{-1}\left(A(t)\right) - t \ . \tag{19}$$

If a user wants to use the service of a SOA with arrival bounds (α^L, α^U) and delay bounds (Φ^L, Φ^U) he or she first has to check whether the arrival stream $A(t)$ lies in the given bounds.

If the arrival stream is not conform to the SLA, a greedy shaper like element can be introduced with an additional delay that has to be added to the delay of the service. If the arrival stream exceeds the capacity of the service, then either another service has to be chosen or the stream has to be distributed over several services as shown below. The lower bound is usually not a problem, since services rarely require a minimum number of calls. If this is the case, artificial calls can always be generated.

Often the user has no complete knowledge about his or her arrival process but knows some bounds. Let (α_0^L, α_0^U) be a pair of bounds. We assume that the bounds are specified by piecewise linear functions, which are convex and concave. If a service with arrival bounds (α_1^L, α_1^U) is used, then the arrival stream can be passed immediately to the service if $\alpha_1^L \leq \alpha_0^L \underline{\otimes} \alpha_1^L$ and $\alpha_0^U \leq \alpha_0^U \underline{\otimes} \alpha_1^U$. In this case bounds for the departure process of the service can be computed with (13,16) if the minimal or maximal departure process under varying delay is analyzed and with (18,17) if the minimal and maximal departure process under the minimal and maximal delay is analyzed. We concentrate on the latter step to determine bounds for the delay of composed services.

If the lower bound is not conform, α_0^L is substituted by $\alpha_0^{L+} = \alpha_0^L \underline{\otimes} \alpha_1^L$ which means that artificial service calls are generated. If the upper bound is not conform, a shaper has to be used to delay incoming service calls. The shaper has a service curve α_1^U such that

$$\alpha^U = \alpha_0^U \underline{\oslash} \alpha_1^U \tag{20}$$

is an upper bound for the departure process [5, Theo. 1.4.3]. If the buffer of the shaper is empty at time 0 and α_0^U, α_1^U specify the arrival and service process, respectively, then [5, Eq. 1.12]

$$\alpha = \alpha_0^U \underline{\otimes} \alpha_1^U \tag{21}$$

is the departure process of the shaper.

The lower bound for the additional delay is zero (i.e., $\Phi_0^L = 0$) if $\alpha_0^{L+} \leq \alpha_0^{L+} \underline{\otimes} \alpha^U$. If this is not the case, the lower bound for the input process produces

$$\alpha^L = \alpha_0^{L+} \underline{\otimes} \alpha_1^U$$

as a lower bound for the departure process [5] of the shaper. The computation of lower and upper bounds for the delay in the shaper are introduced in Section 4 (Eqs. 29-31) where the computation of delay bounds from known lower and upper bounds for arrival and service processes are introduced. For a shaper, the service process is exactly known, i.e., lower and upper bound are identical. The delay of the shaper has to be added to the delay of the service such that $\Phi_{01}^L + \Phi_1^L$ and $\Phi_{01}^U + \Phi_1^U$ are the lower and upper delay bounds for the service including additional delays to smooth the input which are necessary to meet the SLA for the input process.

Often a user builds her or his service from the concatenation of available services and the SLA of the concatenated service has to be derived from the SLAs of the constituting services. We start with the sequential concatenation of two services with arrival curves (α_1^L, α_1^U), (α_2^L, α_2^U) and delay curves (Φ_1^L, Φ_1^U), (Φ_2^L, Φ_2^U) (see Fig. 8). After composition, the resulting service should be described by arrival curves $(\alpha_{12}^L, \alpha_{12}^U)$ and delay curves $(\Phi_{12}^L, \Phi_{12}^U)$. The composed service can then be used like a single service as described above.

There is, of course, an interdependence between the bounds for the arrival process and the resulting delay bounds. A pair of lower and upper arrival bounds $(\alpha_{12}^L, \alpha_{12}^U)$ can be used

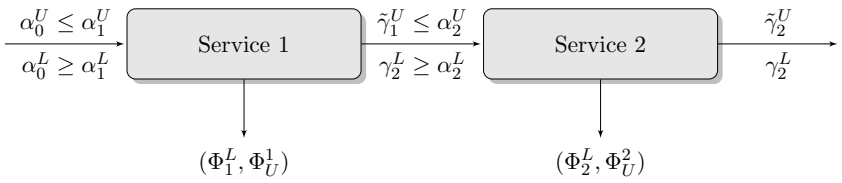

Figure 8: Sequential composition of two services.

as input for the composed service, if $\lim_{t\to\infty} \alpha_{12}^U(t) - \lim_{t\to\infty} \left(\min\left\{\alpha_1^U(t), \alpha_2^U(t)\right\}\right) < \infty$. The delay bounds become $\Phi_{12}^L = \Phi_1^L + \Phi_2^L$ or $\Phi_{12}^U = \Phi_1^U + \Phi_2^U + \Phi_{1+}^U + \Phi_{2+}^U$. Φ_{1+}^U is the additional delay which is necessary to smooth the input for the first service to be consistent with the upper arrival bound of the first service. To analyze the maximal delay in a sequence of services, we have to consider the output of the first service under a maximal arrival process with maximal delay as input process for the second service. Let $\tilde{\gamma}_1^U$ be the upper departure curve of the first service computed via (17) using $\alpha_{12}^U \underline{\otimes} \alpha_1^U$ as an upper arrival curve. Then Φ_{2+}^U is the maximal delay of a smoother with arrival process $\tilde{\gamma}_1^U$ and service curve α_2^U (computed with (31,32) below). If the arrival and delay curves are piecewise linear, then the same holds for the curves for the composed service. However, the minimum and maximum of convex or concave curves results not necessarily in a convex or concave curve. This implies that the resulting arrival curves for the composed service may not be convex and concave. In this cases, it is possible to substitute them by piecewise linear convex and concave lower and upper bounds. Similarly the upper delay bound may not be concave since Φ_{i+}^U might not be concave.

A service resulting from a sequential concatenation can be interpreted as a single service with arrival and delay contract and can then be used in further compositions with other services. In this way, sequential concatenation can be applied to compose an arbitrary number of services.

Example 6. *We consider as a simple example the concatenation of two services. Analysis is restricted to upper bounds that are usually more important than lower bounds. Service* 1 *has an upper arrival curve* $\alpha_1^U = ((0,1,2),(2,5,1))$ *and an upper delay curve* $\Phi_1^U = ((0,0,6),(5,30,3))$. *For the second service arrivals are bounded by the arrival curve* $\alpha_2^U = (0, 2.5, 1.5)$ *and the delay is bounded by the upper delay curve* $\Phi_2^U = (0,0,2)$. *The lower bounds of the arrival and delay process are all zero (i.e.,* $(0,0,0)$*). We assume that an arrival process with bounds* $\alpha_{12}^L = \alpha_1^L$ *and* $\alpha_{12}^U = \alpha_1^U$ *is fed to the composed service.*

Figure 9 shows the upper arrival curves for both services, the departure process of the first service under maximal load and minimal delay γ_1^U *and the upper departure curve* $\tilde{\gamma}_1^U$. *In this case* $t^* = 5$ *because at this time, the first load leaves the system under a maximal arrival stream and maximal delay. It can be seen that the upper departure curve of the first service is not conform with the upper arrival curve of the second service (i.e., the blue line lies initially above the red line) which implies that a shaper has to be put between the first and second service. The delay of the shaper corresponds to the area between the red and blue curve in Figure 9. Figure 10 shows the upper delay curves for the two services, the shaper between service* 1 *and* 2 *and the composed service. Since the arrival process bounds equal the arrival curve for the first service, no shaper in front of the first service is necessary.*

Apart from sequential composition, services may also be composed in parallel. We can distinguish between two possible parallel compositions of services which are shown in Figure

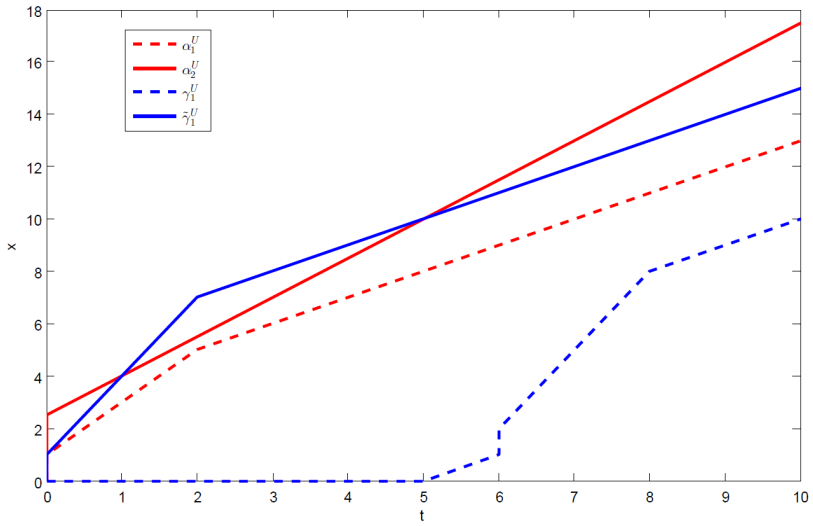

Figure 9: Upper bounds for the arrival and departure processes.

11 and 12 for two composed services. The composition is described in the figures by a circle with the symbol & or ‖. Both compositions occur as pairs. An opening node with one input and two outputs corresponds to a closing node with two inputs and one output. In this way, a composed service has the same interface as a simple service consisting of one input and one output. Between opening and closing node a complex network of composed or simple services can be used (see Fig. 15 for a simple example).

We begin with the &-composition. The semantics of a circle with a & symbol is that the amount of incoming load or fluid is removed from the incoming arc and the same amount is put onto each outgoing arc. In the closing node load from the arc where more load arrived is buffered until load is available on the other arc The construct describes a sort of *fork-join* parallelism.

If for the arrival curve $\alpha_0^U \leq \alpha_1^U \overline{\otimes} \alpha_0^U$ and $\alpha_0^U \leq \alpha_2^U \overline{\otimes} \alpha_0^U$ hold, then no smoothing of the input process is necessary. If α_0^U exceeds temporarily α_1^U or α_2^U, then a smoother is integrated before the corresponding service to delay service calls that violate the upper arrival bound. We assume that the additional delay is implicitly added to the delay defined for this service. If ϕ_i^L, ϕ_i^U $(i = 1, 2)$ are the derivatives of Φ_i^L, Φ_i^U, then ϕ_i^L, ϕ_i^U describe the delay of load arriving at level x. Then

$$\phi_{12}^L = \phi_1^L \vee \phi_2^L, \Phi_{12}^L = \int_0^x \phi_{12}^L(y)dy \text{ and } \phi_{12}^U = \phi_1^U \vee \phi_2^U, \Phi_{12}^U = \int_0^x \phi_{12}^U(y)dy \tag{22}$$

are the lower and upper delay curves of the composed service. I.e., the maximum of both delays determines the delay of the call in the composed service. For the departure process the following relations hold

$$\gamma_{12}^L(x) = \gamma_1^L(x) \wedge \gamma_2^L(x) \text{ and } \gamma_{12}^U(x) = \gamma_1^U(x) \wedge \gamma_2^U(x) \tag{23}$$

The ‖-composition describes the situation that the arriving service call may be routed to one of both sub-services. The choice of an appropriate sub-service might be completely in the hand of the user or it might be determined by some service parameters. In the latter situation,

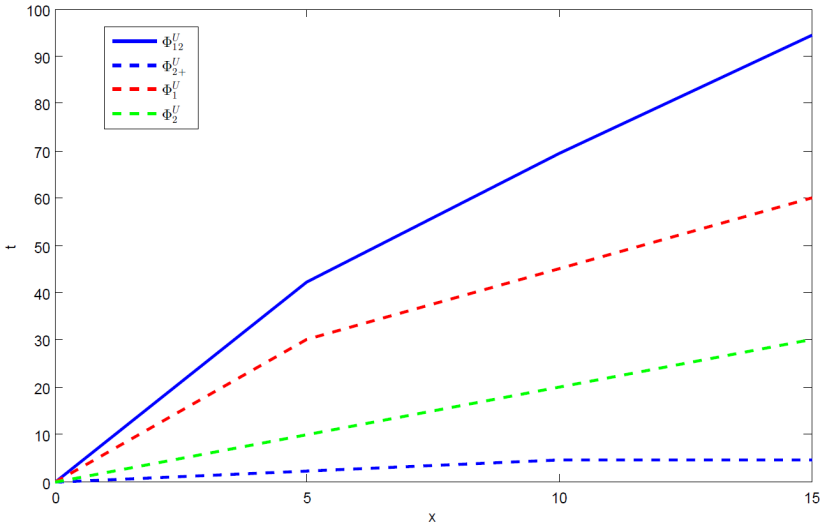

Figure 10: Lower and upper delay curves for the composed service.

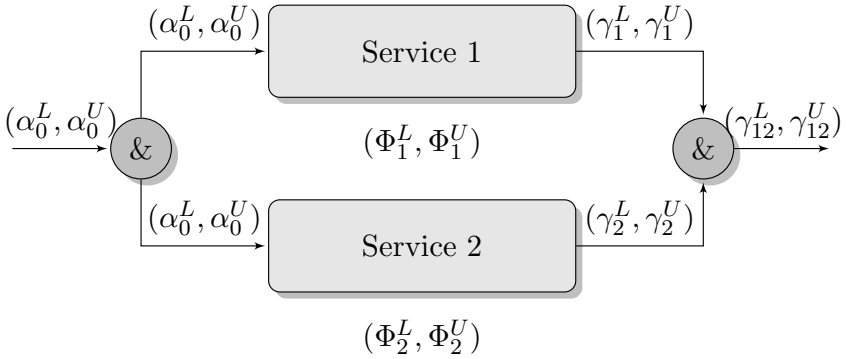

Figure 11: Parallel &-composition of two services.

the worst case behavior is given whenever the slower of both services is used. In this case, the new departure and delay bound equals the bounds for the &-composition. This implies that there is no gain in having two choices. In general, however, better bound can be computed. Ideally, one can define two arrival curves $(\alpha_{01}^L, \alpha_{01}^U)$ and $(\alpha_{02}^L, \alpha_{02}^U)$ such that $\alpha_0^L = \alpha_{01}^L + \alpha_{02}^L$ and $\alpha_0^U = \alpha_{01}^U + \alpha_{02}^U$. $(\alpha_{0i}^L, \alpha_{0i}^U)$ is then fed into service i $(i = 1, 2)$. It is often not possible to decompose an input stream into two streams such that the sum of both streams matches exactly the original stream. Often the relations $\alpha_0^L \geq \alpha_{01}^L + \alpha_{02}^L$ and $\alpha_0^U \leq \alpha_{01}^U + \alpha_{02}^U$ are used to define arrival bounds for the sub-services.

We consider two cases, namely a choice by the user and a probabilistic choice by the system. We begin with the choice by the user. In an ideal situation where load arrives as fluid, it is possible to divide the input stream arbitrarily among the two services. However, service calls arrive as discrete portions which bring some load into the system and a single call has to be transferred to one service and cannot be split. Thus, if $\alpha_0^U = (x_i, y_i, s_i)$ $(1 \leq i \leq L)$, then y_1 usually describes the maximum size of a single call which has to be routed to one service. The distribution of arriving service calls among the services is strongly

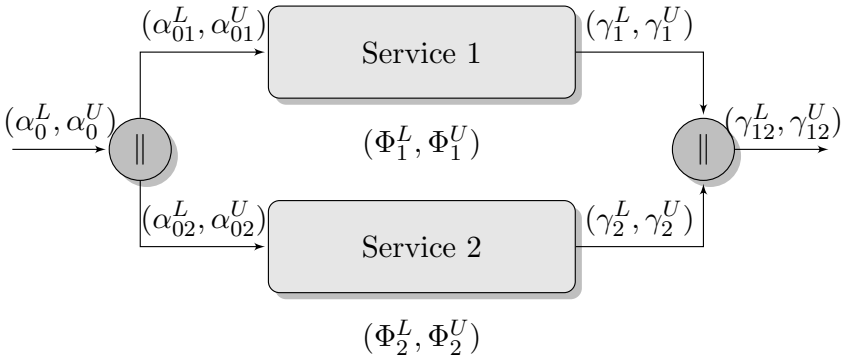

Figure 12: Parallel $\|$-composition of two services.

related to packet-by-packet generalized processor sharing (PGPS) [24] which is a widely used multiplexing scheme. In PGPS we have packets of different classes arriving at a server and the server has to decide which packet to serve next. Here we have packets of one class arriving at a $\|$-element and it has to be decided to which service the call is routed. We consider only the case of two services and assume that w_1 and w_2 are the non-negative weights of the services and load should be distributed proportional to these weights. To describe the scheme, we first define $p = w_1/(w_1 + w_2)$ the portion of calls to be routed to service 1 and $1 - p$ is the portion of calls to be routed to service 2. Let $A(t^-)$ be the load that has been arrived in $[0, t)$ and let $A_1(t^-)$ and $A_2(t^-)$ ($A(t^-) = A_1(t^-) + A_2(t^-)$) be the amount that has been routed to service 1 and 2, respectively. If a new call arrives at time t it is routed to service 1 if $A_1(t^-) < pA(t^-)$, otherwise it is routed to service 2. The following lemma is required to compute arrival curves.

Lemma 1. *If the size of arriving calls is bounded by Ω and the above scheme is used to distribute calls to the services, then*

$$(p-1)\Omega \leq pA(t) - A_1(t) \leq p\Omega \text{ and } -p\Omega \leq (1-p)A(t) - A_2(t) \leq (1-p)\Omega.$$

for all $t > 0$.

Proof. It is sufficient to prove the result for service 1 because by exchanging the number 1 and 2 and p and $(1 - p)$ the proof for the second service follows.

We assume that at time t a call of size ω ($\leq \Omega$) arrives and that $(p-1)\Omega \leq pA(t^-) - A_1(t^-) \leq p\Omega$.

If $pA(t^-) - A_1(t^-) > 0$, then the call is routed to service 1 and immediately after the arrival $A_1(t) = A_1(t^-) + \omega$, $A(t) = A(t^-) + \omega$. Then

$$pA(t) - A_1(t) = p(A(t^-) + \omega) - A_1(t^-) - \omega = pA(t^-) - A_1(t^-) + (p-1)\omega.$$

A lower bound for this equation equals

$$pA(t^-) - A_1(t^-) + (p-1)\omega > (p-1)\Omega$$

since $\omega \in [0, \Omega]$. For the upper bound

$$pA(t^-) - A_1(t^-) + (p-1)\omega \leq p\Omega$$

holds.

If $pA(t^-) - A_1(t) \leq 0$, then the call is routed to service 2 and immediately after the arrival $A_1(t) = A_1(t^-)$, $A(t) = A(t^-) + \omega$. Then

$$pA(t) - A_1(t) = p(A(t^-) + \omega) - A_1(t^-) = pA(t^-) - A_1(t^-) + p\omega.$$

The lower bound equals

$$pA(t) - A_1(t) \geq (p-1)\Omega$$

and the upper bound

$$pA(t) - A_1(t) \leq p\Omega.$$

Since the relation holds at time 0 and it holds after every arrival, it holds for every $t \geq 0$ since the load is modified only at arrival instances of calls. □

Now let $\alpha_0^U = \left(x_i^U, y_i^U, s_i^U\right)$ $(1 \leq i \leq L)$ be an upper arrival curve and define two arrival processes $\alpha_{0j}^U = \left(x_i^{U_j}, y_i^{U_j}, s_i^{U_j}\right)$ $(1 \leq i \leq L,\ j = 1, 2)$ where

$$\begin{array}{lll} x_i^{U_1} = x_i^U, & s_i^{U_1} = ps_i^U, & y_i^{U_1} = py_i^U + (1-p)y_1^U, \\ x_i^{U_2} = x_i^U, & s_i^{U_1} = (1-p)s_i^U, & y_i^{U_2} = (1-p)y_i^U + py_1^U. \end{array}$$

It is easy to show that $\alpha_{01}^U(t) + \alpha_{02}^U(t) = \alpha_0^U(t) + y_1^U$ $(t \geq 0)$ holds, i.e. the combined upper bound requires one additional service call of maximal size that can instantaneously arrive at each sub- service. For some $t \geq 0$ which belongs to the linear segment i, we have

$$\begin{array}{lll} \alpha_{01}^U(t) - p\alpha_0^U(t) & = & (t - x_i^{U_1})s_i^{U_1} + y_i^{U_1} - p\left((t - x_i^U)s_i^U + y_i^U\right) = \\ & & (t - x_i^U)ps_i^U + py_i^U + (1-p)y_1^U - p\left((t - x_i^U)s_i^U + y_i^U\right) = \\ (1-p)y_1^U. & & \end{array}$$

If we choose $\Omega = y_1^U$ and $A(t) = \alpha_0^U(t)$ in Lemma 1, it can be seen that $\alpha_{01}^U(t)$ is a valid upper bounding curve according to the PGPS like scheme of routing calls to services. In a similar way valid lower bounding curves can be computed for the routing scheme.

The parameters of the composed service depend on the weights. Let w_1 and w_2 be the weights used to route calls to the services and let $p_1 = w_1/(w_1 + w_2)$ and $p_2 = 1 - p_1$. Furthermore let Ω be the maximal size of a call. Then an arrival curve α does not exceed the upper arrival curve α_j of service j if

$$p_j\alpha + p_j\Omega \leq \alpha_j^U \Rightarrow \alpha \leq \frac{\alpha_j^U}{p_j} - \Omega$$

such that

$$\alpha_{12}^U = \frac{\alpha_1^U}{p_1} - \Omega \wedge \frac{\alpha_2^U}{p_2} - \Omega.$$

Similarly, the lower bound is respected by an arrival curve α, if

$$\alpha_j^L \leq p_j\alpha - (1-p_j)\Omega \wedge 0 \Rightarrow \alpha \geq \frac{\alpha_j^L + (1-p_j)\Omega}{p_j} \wedge 0$$

such that

$$\alpha_{12}^L = \frac{\alpha_1^L + (1-p_1)\Omega}{p_1} \vee \frac{\alpha_2^L + (1-p_2)\Omega}{p_2} \vee 0.$$

For the delay bounds we have to consider the maximal and minimal delay which can occur at load level x. Since Φ_i^U is concave, delays are non increasing which means that for a minimal load that arrived to a service the upper delay is maximized. If load x arrived to the composed service, then $p_i(x-\Omega)^+$ is the minimum of the load that arrived to service i ($i=1,2$) and $\phi_i^U(p_i(x-\Omega)^+)$ is the corresponding delay. For the lower delay bound we observe that Φ_i^L is non-decreasing such that again the smallest load has to be considered to compute the lower delay bound. Thus, the lower delay bound becomes $\phi_i^L(p_i(x-\Omega)^+)$. Together this results in the following computation of the delay bounds.

$$\begin{array}{ll} \phi_{12}^L(x) = \min\left\{\phi_1^L(p_1(x-\Omega)^+), \phi_2^L(p_2(x-\Omega)^+)\right\}, & \Phi_{12}^L(x) = \int_0^x \phi_{12}^L(y)dy \\ \phi_{12}^U(x) = \min\left\{\phi_1^U(p_1(x-\Omega)^+), \phi_2^U(p_2(x-\Omega)^+)\right\}, & \Phi_{12}^U(x) = \int_0^x \phi_{12}^U(y)dy \end{array} \tag{24}$$

For piecewise linear arrival and delay curves of the services, the arrival and delay curves of the composed service are also piecewise constant. However, the delay curves of the composed service need not be concave or convex even if this holds for the delay curves of the sub-services.

The output process of the composed service is given by the superposition of the output processes of the sub-services. This implies that the sum of the lower or upper departure curves defines a lower and an upper bound for the departure process of the composed service. Furthermore, the output process is bounded by the input process because calls have to depart after their arrival. This results in the following bounds.

$$\gamma_{12}^L = \left(\gamma_1^L + \gamma_2^L\right) \wedge \alpha_0^L \text{ and } \gamma_{12}^U = \left(\gamma_1^U + \gamma_2^U\right) \wedge \alpha_0^U \tag{25}$$

Example 7. *We consider an example with two parallel services and the bounding curves* $\alpha_1^L = ((0,0,0),(1,0,1),(3,2,1.5))$, $\alpha_1^U = ((0,1,3),(2,7,2.5))$, $\Phi_1^L = ((0,0,1),(2,2,2))$, $\Phi_1^U = ((0,0,3),(1,3,2))$ *for service* 1 *and* $\alpha_2^L = ((0,0,0),(0.5,0,1),(3,2.5,1.5))$, $\alpha_2^U = ((0,2,3.5),(4,16,2))$, $\Phi_2^L = ((0,0,0),(2,0,1.5))$, $\Phi_2^U = ((0,0,2.5))$ *for service* 2.

Both services are combined with an &-composition and the arrival bounds are chosen as the minimum of the two upper arrival curves and the maximum of the two lower arrival curves resulting in $\alpha_{12}^L = ((0,0,0),(0.5,0,1),(3,2.5,1.5))$ *and* $\alpha_{12}^U = ((0,1,3),(2,7,2.5),(12,32,2))$ *to avoid the use of a smoother. The delay curves are given by* $\Phi_{12}^L = ((0,0,1),(2,2,2))$ *and* $\Phi_{12}^U = ((0,0,3),(1,3,2.5))$.

For a $\|$*-composition we first determine the possible values for* p_1 *which depend on the bounds for the arrival process* (α_0^L, α_0^U). *To meet the long term arrival bounds of the sub-services* 1 *and* 2, *the long term rate of the arrival process has to be between* 3 *and* 4.5. *Assume that* $\alpha_{12}^L = ((0,0,0),(1,0,2),(2,2,4))$ *and* $\alpha_{12}^U = ((0,2,6),(1,8,4))$, *then* $p_1 \in [0.5, 0.625]$. *For values outside the interval, the lower or upper bounds for at least one of the sub-services are violated. We analyze the system for* $p_1 = 0.5$, $p_1 = 0.625$ *and* $\Omega = 1$. *The corresponding lower and upper delay bounds are shown in Fig. 13. It can be seen that the curves for* $p = 0.625$ *are above the curves fro* $p = 0.5$.

For calls that are routed with some probability to one of the two services, a strict upper bound corresponds to the above mentioned worst case scenario with no gain of the parallel service composition. If a predefined small probability to exceed the bound in some finite interval $[0,T]$ is accepted, then much better bounds can be computed. The concrete bounding curves depend on the probabilities with which calls are routed to the services and on the probability distribution of the size of calls (see also the following example).

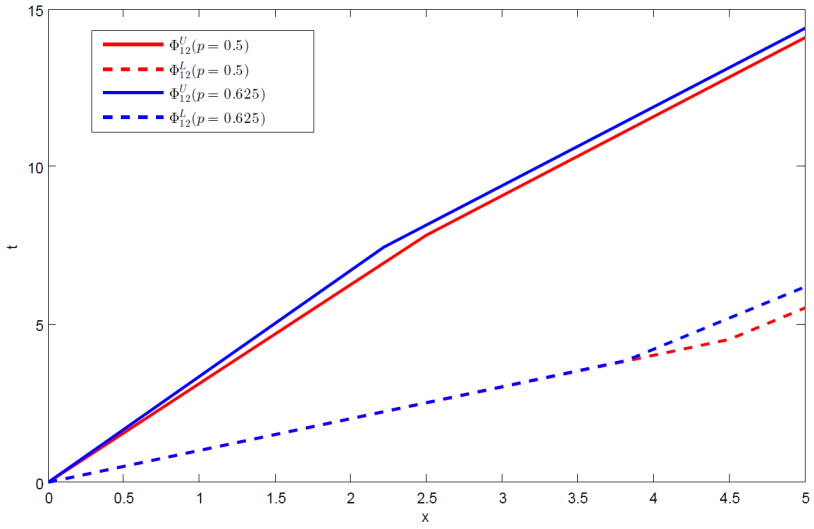

Figure 13: Lower and upper delay curves for the composed services and different values of p_1.

Example 8. *Consider an arrival stream where calls of size* 1 *arrive periodically every* Δ *time units with a jitter of* σ $(< \Delta)$*. It has already been shown that* $\big((0,1,(\Delta-\sigma)^{-1}),(\Delta-\sigma,2,\Delta^{-1})\big)$ *describes a valid upper arrival curve. If we assume that calls can be processed by two services with equal capacity, then* $((0,1,0.5/(\Delta-\sigma)),(\Delta-\sigma,1.5,0.5/\Delta))$ *is an upper arrival curve if calls are equally distributed among the services (i.e.,* $p = 0.5$*).*

Now consider an arrival stream with the same parameters as before but calls request with probability 0.5 *to the first service and with probability* 0.5 *the second service. If we choose the previous arrival curves for the services, then the bound is exceeded by the second arrival with probability* 0.5*, namely in all cases where two subsequent arrivals choose the same service. If we use instead an arrival curve with* $\Big((0,1,\frac{1}{\Delta-\sigma}),(\Delta-\sigma,2,\frac{1}{\Delta}),(k\Delta-\sigma,k,\frac{0.5}{\Delta})\Big)$*, then a service is overloaded only if out of* K $(> k)$ *calls more then* $0.5K + k$ *calls request one of the services. The corresponding probability equals*

$$2\sum_{h=k+1}^{K}\binom{K}{h}0.5^K.$$

Thus, for a given probability that the upper bound is not exceeded within K *arrivals, an appropriate value of* k *can be computed.*

With the presented results a user of a SOA can compose his or her services from available sub-services that are combined sequentially or in parallel. For the resulting composed service arrival and delay contracts can be computed easily from the arrival and delay contracts of the sub-services. This allows one to compose complex services and obtain SLAs for the composed service from the SLAs of the sub-services in a completely formalized way.

4 The Provider Perspective of SLAs

A provider is interested in the evaluation of the relationship between the load, the guaranteed delay and the necessary service capacity. We assume that the service can be represented by a function $S(t) \in \mathcal{F}$ that describes the load processed in the interval $[0, t)$ under the condition that load to process is available for the whole period. Usually one assumes a work conserving strategy which implies that the system starts working as soon as load becomes available. We consider the case of a single request stream.

To analyze SOAs we assume that the processing capacity of a service $S(t)$ is bounded by two functions σ^L and σ^U such that

$$\sigma^L(t-s) \leq S(t) - S(s) \leq \sigma^U(t-s)$$

for all $0 \leq s \leq t$. This kind of bound is denoted as a strict service curve in *Network Calculus* [6]. The upper bound is assumed to be sub-additive and the lower bound super-additive. Again we usually assume that the upper bound is given by a piecewise linear concave function and the lower bound by a piecewise linear convex function.

If $S(t)$ is available, then for the output process $C(t)$ the following relation holds [5].

$$(A \underline{\otimes} S)(t) \leq C(t) \leq (A \underline{\oslash} S)(t) \tag{26}$$

If we have bounds for the arrival and service process, in the form of lower and upper arrival and service curves, (α^L, α^U) and (σ^L, σ^U), then the following bounding curves can be computed for the departure process (for a proof see [34])

$$\begin{array}{ll} \gamma^L & = \min\left\{\left(\alpha^L \underline{\oslash} \sigma^U\right) \underline{\otimes} \sigma^L, \sigma^L\right\} \\ \gamma^U & = \min\left\{\left(\alpha^U \underline{\otimes} \sigma^U\right) \underline{\oslash} \sigma^L, \sigma^U\right\}. \end{array} \quad \geq \alpha^L \underline{\otimes} \sigma^L \tag{27}$$

For piecewise linear functions, γ^L and γ^U are easy to compute and piecewise linear.

To process the load that is brought to the system some of the available service capacity is required. The remaining service capacity can then be used to serve other customers. Let $\hat{\sigma}^L$ and $\hat{\sigma}^U$ be a lower and upper bound for the remaining service capacity which be computed from the following equations [34]

$$\hat{\sigma}^L = \left(\sigma^L - \alpha^U\right) \overline{\otimes} 0 \text{ and } \hat{\sigma}^U = \left(\sigma^U - \alpha^L\right) \overline{\oslash} 0 \tag{28}$$

Again the functions are piecewise linear for piecewise linear bounding curves of the arrival and service process.

For the provider of a SOA two questions arise.

- Is the available service sufficient to fulfill the SLA?
- What is the minimal service capacity needed to fulfill the SLA?

We begin with a provider who knows (σ^L, σ^U) the bounds for the available service capacity. Since in a SOA a service usually uses different resources, it is quite natural to describe the available capacity in the form of bounds.

To compute bounds for the departure process and delay from the arrival and service curves, we make the following assumption: If $A_1(t)$ and $A_2(t)$ are two arrival processes with $A_1(t) \geq A_2(t)$ for all $t \geq 0$ to some system $\mathcal{S}$ and $C_1(t)$, $C_2(t)$ are the two departure processes

from the system when $A_1(t)$ and $A_2(t)$ are fed to the system, then $C_1(t) \geq C_2(t)$ and $A_1(t) - C_1(t) \geq A_2(t) - C_2(t)$ for all $t \geq 0$. Similarly, if an arrival process $A_(t)$ is fed to a system with two possible service processes $S_1(t)$ and $S_2(t)$ with $S_1(t) \geq S_2(t)$ for all $t \geq 0$ and $C_1(t)$, $C_2(t)$ are the corresponding departure processes, then $C_1(t) \geq C_2(t)$ and $A(t) - C_1(t) \leq A(t) - C_2(t)$ for all $t \geq 0$. These properties define again some kind of monotonic behavior of a system. If we assume monotonic behavior and know the arrival curves (α^L, α^U) and the service curves (σ^L, σ^U), then the system $\mathcal{S}_{low}$ with arrival process α^L and service process σ^U produces the smallest delay and the system $\mathcal{S}_{up}$ with arrival process α^U and service process σ^L produces the largest delay. Let

$$b^L = \inf_{u \geq 0} \left\{ \alpha^L(u) - \sigma^U(u) \right\} \text{ and } b^U = \sup_{u \geq 0} \left\{ \alpha^U(u) - \sigma^L(u) \right\}$$

be the minimum and maximum backlog of $\mathcal{S}_{low}$ and $\mathcal{S}_{up}$, respectively. For both systems bounds for the departure process can be computed using (27). Such that

$$\gamma_{low} = \min \left\{ \left(\alpha^L \underline{\otimes} \sigma^U \right) \underline{\oslash} \sigma^U, \sigma^U \right\} \tag{29}$$

for the upper bound of the departure process of $\mathcal{S}_{low}$ and

$$\gamma_{up} = \min \left\{ \left(\alpha^U \underline{\oslash} \sigma^L \right) \underline{\otimes} \sigma^L, \min \left\{ \sigma^L, \alpha^U \right\} \right\} \tag{30}$$

is the lower bound for $\mathcal{S}_{up}$. Consequently, $(\gamma_{low})^{-1}$ is an lower bound for the pseudo-inverse of the departure process of $\mathcal{S}_{low}$ and $(\gamma_{up})^{-1}$ is a upper bound for the pseudo-inverse of the departure process of $\mathcal{S}_{up}$.

If $\lim_{t \to \infty} \frac{\alpha^U(t)}{t} > \lim_{t \to \infty} \frac{\gamma^U(t)}{t}$, then the delay becomes $((0, \infty, \infty))$. Otherwise upper and lower bounds for the delay for load x are then given by

$$\phi^U(x) = \gamma_{up}^{-1}(x) - \left(\alpha^U \right)^{-1}(x) \text{ and } \phi^L(x) = \gamma_{low}^{-1}(x) - \left(\alpha^L \right)^{-1}(x) \tag{31}$$

and

$$\Phi^U(x) = \int_0^x \phi^U(y) dy \text{ and } \Phi^L(x) = \int_0^x \phi^L(y) dy. \tag{32}$$

If Φ^U, Φ^L are not concave or convex, respectively, they may be substituted by concave upper and convex lower bounds. The computed bounds can then be compared with bounds required by the SLA to prove whether the available service capacity is sufficient to fulfill the SLA.

We now compute bounds (σ^L, σ^U) for the required service capacity. Since we assume a monotonic behavior of the system, the largest delay results from the system with arrival process α^U and service process σ^L. The output process of this system has to be larger or equal to γ^U, the output process of the system with arrival process α^U defined in (14). Since $\alpha^U \underline{\otimes} \sigma^L$ is as lower bound for the output process of the system with arrival process α^U and service process σ^L, $\gamma^U \geq \alpha^U \underline{\otimes} \sigma^L$ implies that the service capacity is sufficient to meet the upper delay bound under all arrival process that are conform to the SLA. With the relation between $\underline{\otimes}$ and $\underline{\oslash}$ the lower service curve can be computed as

$$\gamma^U \leq \alpha^U \underline{\otimes} \sigma^L \;\Leftrightarrow\; \gamma^U \underline{\oslash} \alpha^U \leq \sigma^L \tag{33}$$

such that $\sigma^L = \gamma^U \underline{\oslash} \alpha^U$ can be chosen.

For the computation of an upper bound for the service process we can in principle proceed similarly. Due to the monotonicity the shortest delay will be observed in the system with

the smallest arrival process α^L and the largest service process σ^U. This system has to have a delay process of at least Φ^L. However, if $\alpha^L = (0,0,0)$, then the delay bound cannot be met and in several other cases, the upper bound becomes too small as outlined below. To compute an upper bound, we first compute γ^L from (18). With similar arguments as above, the upper bound for the service process can be computed as

$$\gamma^L \leq \alpha^L \underline{\otimes} \sigma^U \;\Leftrightarrow\; \gamma^L \underline{\oslash} \alpha^L \geq \sigma^U \tag{34}$$

Thus, $\sigma^U = \gamma^L \underline{\oslash} \alpha^L$ is the required upper bound. Often $\sigma^U \geq \sigma^L$ will not hold, especially if α^L is small and the service has to be very slow to meet the lower delay bound. In this case we set $\sigma^U = \sigma^U \wedge \sigma^L$ and delay calls that are finished too early until they meet the lower delay bound.

Example 9. *We consider a system with the input and delay bounds as shown in Figure 5. Assume that the available service capacity is bounded from below by the service curve* $\sigma^L = ((0,0,0),(1.5,0,2))$. *For the lower bound this means that for a period of length* 1.5 *no load is served and afterwards the service continues with rate* 2. *The lower output bound, which is necessary to meet the upper delay bound under the maximal arrival stream (and therefore under all allowed arrival streams that are bounded by* α^U*) is*
$\gamma^U = ((0,0,0),(1.5,1,1.25),(2,1.625,2.25),(2.3,2.3,1))$.
With service process σ^L *and arrival process* α^U *we obtain an output process*
$\gamma^1 = ((0,0,0),(1.5,0,2),(2.7,2.4,1))$.
Fig. 14 shows the curves and it becomes clear that $\gamma^1 \leq \gamma^U$ *which implies that the service capacity is not sufficient. If we use instead a server with lower service curve* $\sigma^L = ((0,0,0),(1,0,2))$*, then the lower output bound becomes*
$\gamma^2 = ((0,0,0),(1,0,2),(2.2,2.4,1))$
and $\gamma^2 \geq \gamma^U$.

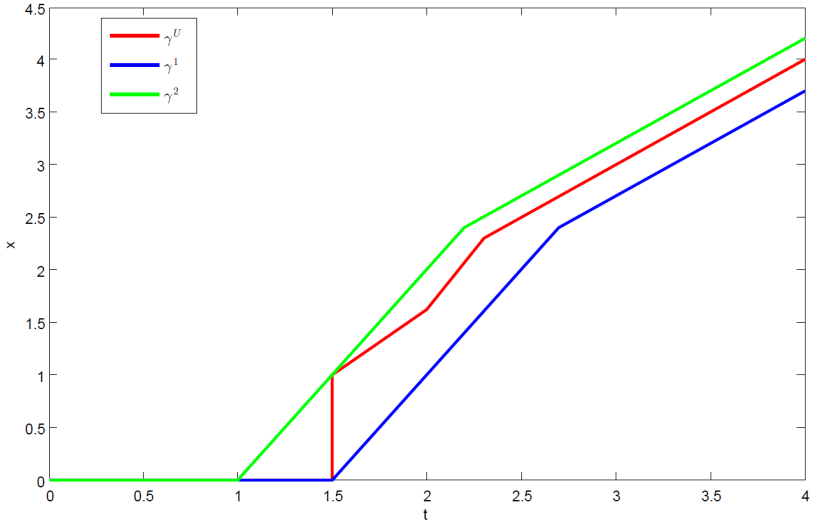

Figure 14: Bounds for the departure process.

The minimal service curve computed with (33) becomes
$\sigma^L = ((0,0,0),(1.2,0,1),(1.5,1,1.25),(2,1.625,2.25),(2.3,2.3,1))$.

	Provider 1		Provider 2	
	α^U	Φ^U	α^U	Φ^U
A	(0,5,5),(2,15,2)	(0,0,4),(2,8,2)	(0,1,3),(2,7,2)	(0,0,3),(2,6,2)
B	(0,1,8),(3,25,2)	(0,0,6),(4,24,4)	(0,1,3),(3,10,2)	(0,0,5),(3,15,3)
C	(0,2,3),(4,14,2)	(0,0,3),(3,9,2)	(0,1,2)	((0,0,3),(2,6,1)
D	(0,1,3),(3,10,2)	(0,0,4),(5,20,2)	(0,1,2)	(0,0,3),(5,15,1)
E	(0,1,2),(2,5,1)	(0,0,6),(2,12,1)	(0,1,1)	(0,0,2),(2,4,1)

Table 1: Bounding curves for the service parameters offered by the two providers.

To meet the lower delay bound under the minimal arrival process we first compute
$\gamma^L = ((0,0,0),(2.2,0,1),(4.2,2,0),(4.4,2,1))$
using (34) we obtain
$\sigma^U = ((0,0,0),(1,0,1),(3,2,0),(3.2,2,1))$
such that $\sigma^U \geq \sigma^L$ does not hold, i.e., σ^U has to be modified and calls have to be delayed, if necessary.

5 An Example

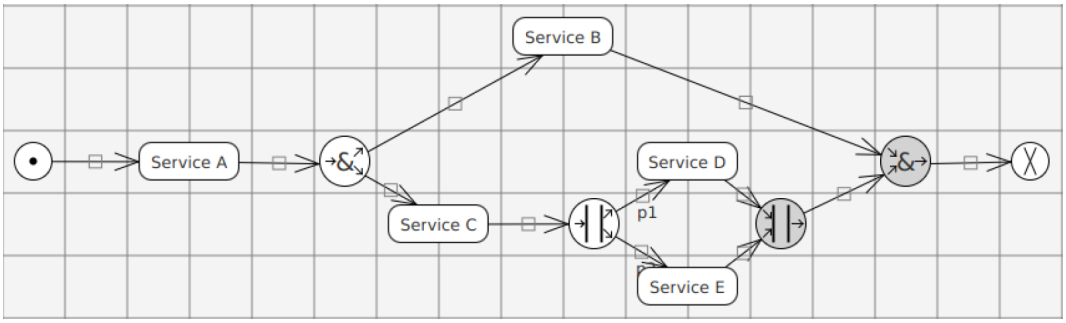

Figure 15: Example service composition.

We consider the analysis of a composed service from the user's perspective. The structure of the model follows the examples presented in [20] where, however, only mean durations and costs are considered. Figure 15 shows the service that is composed of 5 sub-services $A, \ldots, E$. We assume that the $\|$-composition between D and E is defined by a probabilistic choice such that 75% of the load goes to D and the maximal size of a call is 1. The sub-services are provided by two different providers. The upper arrival and delay curves are shown in Table 1. We assume that all lower arrival and delay curves are zero, i.e., $(0,0,0)$.

The services of both providers have the same arrival bounds and delays for large t. Provider 1 allows larger arrival batches and temporarily larger arrival rates at the beginning but the price for this flexibility is a longer delay for the first load units. We analyze two configurations where all services are provided by provider 1 or 2, respectively. Φ_1^U and Φ_2^U are the corresponding upper delay curves of the composed service.

The system is first analyzed under an input process with upper arrival curves $\alpha_0^U = ((0,1,10),(4,41,4),(6,49,1))$. Fig. 16 shows the upper delay bounds for both variants. Vari-

ant 2 is better in this case, although it has a slightly larger delay at the begining.

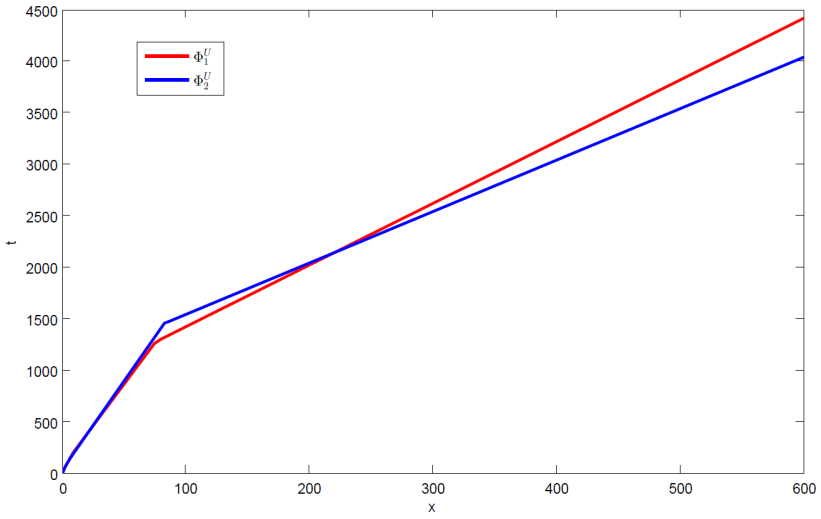

Figure 16: Upper delay bounds for the first arrival process.

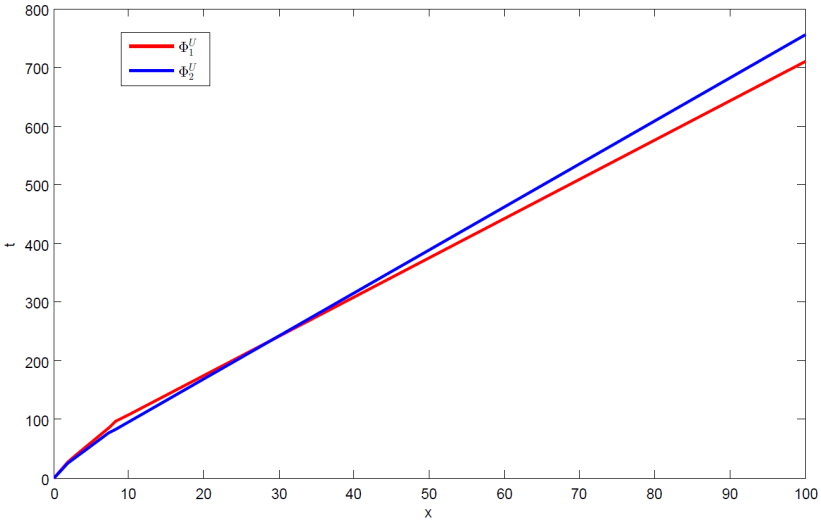

Figure 17: Upper delay bounds for the second arrival process.

If we change the arrival process to a process with upper bound $\alpha^U = ((0, 1, 5), (2, 11, 2))$, the situation changes. In this case, variant 1 is better than variant 2. The corresponding upper delay curves are shown in Fig. 17.

6 Conclusion

In this report we present a new approach to analyze large software architectures based on the information about the quantitative behavior available in the SLAs describing the functional

and non-functional properties of the system. With the available information on bounds for the load and the delay, bounds on the quality of service of composed services as well as requirements on the performance of the underlying architecture can be derived. Like with product form queuing networks, systems can be analyzed with a minimal amount of information, only simple piecewise linear functions are used to specify the parameters. Therefore necessary computations are usually efficient. The approach considers mainly worst case behavior but this is what is commonly specified in SLAs.

The presented approach is a first step in using min/plus algebra and the concept of *Network* or *Real Time Calculus* for the performance analysis of software systems. It is possible to extend the approach by adopting additional results from these areas. In particular multiple classes and the integration of cost functions are interesting topics for future research.

References

[1] D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez, and W. Wang. Quality-of-service in cloud computing: modeling techniques and their applications. *J. Internet Services and Applications*, 5(1), 2014.

[2] F. Bause and P. Buchholz. SLA tool. In R. German, K. J. Hielscher, and U. R. Krieger, editors, *Measurement, Modelling and Evaluation of Computing Systems - 19th International GI/ITG Conference, MMB 2018, Erlangen, Germany, February 26-28, 2018, Proceedings*, pages 302–306, 2018.

[3] F. Bause, P. Buchholz, and J. May. A tool supporting the analytical evaluation of service level agreements. In W. Binder, V. Cortellessa, A. Koziolek, E. Smirni, and M. Poess, editors, *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering, ICPE 2017, L'Aquila, Italy, April 22-26, 2017*, pages 233–244. ACM, 2017.

[4] S. Bondorf and J. B. Schmitt. The DiscoDNC v2 - A comprehensive tool for deterministic network calculus. In *8th International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2014, Bratislava, Slovakia, December 9-11, 2014*, 2014.

[5] J. L. Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, volume 2050 of *Lecture Notes in Computer Science*. Springer, 2001.

[6] A. Bouillard, L. Jouhet, and E. Thierry. Service curves in network calculus: dos and don'ts. Rapport de recherche 7094, INRIA, 2009.

[7] A. Bouillard and E. Thierry. An algorithmic toolbox for network calculus. *Discrete Event Dynamic Systems*, 18(1):3–49, 2008.

[8] P. Buchholz and S. Vastag. Toward an analytical method for SLA validation. *Software and System Modeling*, 17(2):527–545, 2018.

[9] G. Casale, N. Mi, L. Cherkasova, and E. Smirni. Dealing with burstiness in multi-tier applications: Models and their parameterization. *IEEE Trans. Software Eng.*, 38(5):1040–1053, 2012.

[10] C.-S. Chang. *Performance Guarantees in Communication Networks*. Springer, 2000.

[11] V. Cortellessa, A. D. Marco, and P. Inverardi. *Model-Based Software Performance Analysis*. Springer, 2011.

[12] R. L. Cruz. A calculus for network delay, part I: network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, 1991.

[13] R. L. Cruz. A calculus for network delay, part II: network analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, 1991.

[14] J. Eckert, K. Pandit, N. Repp, R. Berbner, and R. Steinmetz. Worst-case performance analysis of web service workflows. In *IWAS'2007 - The Ninth International Conference on Information Integration and Web-based Applications Services, 3-5 December 2007, Jakarta, Indonesia*, pages 67–77, 2007.

[15] G. Franks, T. Omari, C. M. Woodside, O. Das, and S. Derisavi. Enhanced modeling and solution of layered queueing networks. *IEEE Trans. Software Eng.*, 35(2):148–161, 2009.

[16] Y. Jiang and Y. Liu. *Stochastic Network Calculus*. Springer, 2008.

[17] L. Kleinrock. *Queueing Systems*, volume 1. Wiley, 1975.

[18] H. Koziolek. Performance evaluation of component-based software systems: A survey. *Perform. Eval.*, 67(8):634–658, 2010.

[19] H. Li, W. Theilmann, and J. Happe. SLA translation in multi-layered service oriented architectures: status and challenges. Interner bericht, Universität Karlsruhe, Fakultät für Informatik, 2009.

[20] D. A. Menascé. Composing Web services: A QoS view. *IEEE Internet Computing*, 8(6):88–90, 2004.

[21] D. A. Menascé. Mapping service-level agreements in distributed applications. *IEEE Internet Computing*, 8(5):100–102, 2004.

[22] D. A. Menasce and V. A. F. Almeida. *Capacity Planning for Web Services: metrics, models, and methods*. Prentice Hall, 2001.

[23] D. A. Menascé, H. Ruan, and H. Gomaa. Qos management in service-oriented architectures. *Perform. Eval.*, 64(7-8):646–663, 2007.

[24] 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. Netw.*, 1(3):344–357, 1993.

[25] J. F. Pérez and G. Casale. Assessing SLA compliance from palladio component models. In *15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2013, Timisoara, Romania, September 23-26, 2013*, pages 409–416. IEEE Computer Society, 2013.

[26] J. B. Schmitt and U. Roedig. Sensor network calculus - A framework for worst case analysis. In *Distributed Computing in Sensor Systems, First IEEE International Conference, DCOSS 2005, Marina del Rey, CA, USA, June 30 - July 1, 2005, Proceedings*, volume 3560 of *Lecture Notes in Computer Science*, pages 141–154. Springer, 2005.

[27] W. J. Stewart. *Probability, Markov Chains, Queues, and Simulation.* Princeton University Press, 2009.

[28] L. Thiele, S. Chakraborty, and M. Naedele. Real-time calculus for scheduling hard real-time systems. In *Proceedings. ISCAS 2000*, pages 100–104, 2000.

[29] M. Tribastone, P. Mayer, and M. Wirsing. Performance prediction of service-oriented systems with layered queueing networks. In *Leveraging Applications of Formal Methods, Verification, and Validation - 4th International Symposium on Leveraging Applications, ISoLA 2010, Heraklion, Crete, Greece, October 18-21, 2010, Proceedings, Part II*, pages 51–65, 2010.

[30] S. Vastag. Modeling quantitative requirements in SLAs with network calculus. In *5th International ICST Conference on Performance Evaluation Methodologies and Tools Communications, VALUETOOLS '11, Paris, France, May 16-20, 2011*, pages 391–398, 2011.

[31] S. Vastag. Arrival and delay curve estimation for SLA calculus. In *Winter Simulation Conference, WSC '12, Berlin, Germany, December 9-12, 2012*, 2012.

[32] S. Vastag. A calculus for SLA delay properties. In *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance - 16th International GI/ITG Conference, MMB & DFT 2012, Kaiserslautern, Germany, March 19-21, 2012. Proceedings*, pages 76–90, 2012.

[33] S. Vastag. *SLA Calculus.* PhD thesis, Department of Computer Science, TU Dortmund, 2014.

[34] E. Wandeler. *Modular Performance Analysis and Interface-Based Design for Embedded Real-TimeSystems.* PhD thesis, ETH Zürich, 2006.

[35] E. Wandeler and L. Thiele. Real-Time Calculus (RTC) Toolbox. `http://www.mpa.ethz.ch/Rtctoolbox`, 2006.

[36] K. Xiong and H. G. Perros. Service performance and analysis in cloud computing. In *2009 IEEE Congress on Services, Part I, SERVICES I 2009, Los Angeles, CA, USA, July 6-10, 2009*, pages 693–700, 2009.