Modeling Time-Triggered Service Intermittence In Network Calculus

Jonathan Falk
jonathan.falk@ipvs.uni-stuttgart.de
University of Stuttgart

Frank Dürr
frank.duerr@ipvs.uni-stuttgart.de
University of Stuttgart

Kurt Rothermel
kurt.rothermel@ipvs.uni-stuttgart.de
University of Stuttgart

ABSTRACT

Network elements (e.g., switches), which intermit service, i.e., stop forwarding and transmission of data according to a repeating schedule, can be found in many real-time capable communication networks, e.g., communication networks with TDMA, Ethernet with Time-aware Shapers or low-power wireless networks. The behavior of those network elements depends on the (stationary) properties of the network elements, their schedule, and the current time, as well as the offered traffic load. If a networked real-time system generates traffic flows which are not synchronized to the schedules of the network elements, formal frameworks such as Network Calculus (NC) are highly valuable to derive deterministic guarantees for the communication.

In this paper, we show the fundamental implications of modeling time-triggered network elements with service intermittence in NC. We identify two archetypes of network elements with intermittent service, and propose time-variant and time-invariant approaches to derive service curve formulations to model them. We evaluate the differences between time-variant and time-invariant service curves with respect to the overestimation of worst-case backlog and worst-case delay, and we identify schedule properties which influence the tightness of the derived bounds.

CCS CONCEPTS

• **Networks** → **Network performance analysis**; • **Computer systems organization** → *Real-time systems.*

ACM Reference Format:
Jonathan Falk, Frank Dürr, and Kurt Rothermel. 2019. Modeling Time-Triggered Service Intermittence In Network Calculus. In *27th International Conference on Real-Time Networks and Systems (RTNS 2019), November 6–8, 2019, Toulouse, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3356401.3356411

1 INTRODUCTION

Many real-time systems are distributed systems, where the individual components are connected by a communication network. For example, Cyber-physical Systems (CPS), which control physical processes through a set of software components, are often distributed due to the spatial dimensions of the physical objects they interact with. This applies, e.g., for automotive systems, "smart cities", Industrial Internet of Things (Industry 4.0), etc. Being a distributed real-time system, real-time communication networks guaranteeing *deterministic bounds on the network delay* between distributed components (real-time communication) are required.

There are many scenarios in which asynchronously generated traffic, e.g., event-triggered traffic which is subject to real-time requirements, is being sent through a communication network comprised of network elements with time-triggered service intermittence (service being the forwarding and transmission of data in the network elements). For example, in converged networks traffic synchronized to the schedules in network elements (often called time-triggered traffic) is transported as well as asynchronous traffic (e.g., shaped real-time traffic as known from IEEE 802.1 AVB, or rate-constrained traffic in TTEthernet [11]). Here, all data flows, including the non-time triggered ones, can be affected by TDMA-like time-aware shaping, where the bandwidth available to data flows varies over time according to the time slots of the schedule. Time-aware shaping as specified in the IEEE 802.1Q standard [3] can restrict the access of each queue storing outgoing traffic of a certain traffic class to its associated outgoing line to precisely defined time intervals according to a cyclic schedule.

In addition, time-triggered service intermittence in network elements is not always primarily motivated by real-time constraints. Schedules for service intermittence can also be incorporated into CPS to increase their efficiency, e.g., by anticipating external constraints. Stated informally: do not offer service when it is not strictly required, or when it is useless (e.g., due to environmental reasons). For instance, the communication link of a traffic sensor could be turned off outside of rush hours. This motivation can be found in applications where transmission schedules in the network elements are applied with the goal of power-conservation in energy-constraint devices [7, 23, 26], e.g., in wireless applications where data is buffered when the link is turned off. Nevertheless, for these wireless applications with time-triggered transmission intervals, there are many scenarios which require bounds on the time of data delivery [19] and the buffer usage, e.g., to avoid packet loss.

In this paper, we therefore focus on the class of network elements which are controlled by a repeating schedule where the schedule entries trigger intermittence or resumption of the service process at specific times to the effect that there is some service for some intervals and no service for other intervals. We refer to such a network element as time-triggered network element with intermittent service. The service offered by these network elements does not only depend on the offered traffic load and the properties of the network (e.g., link bandwidth, cyclic schedule), but additionally on the *current time.* Thus, these network elements, and by extension the whole network, exhibit a *time-varying behavior.*

Since it is non-trivial to give deterministic, provable real-time guarantees for the asynchronous traffic in these scenarios, formal frameworks are highly useful, because they provide a "systematic"

RTNS 2019, November 6–8, 2019, Toulouse, France
2019. ACM ISBN 978-1-4503-7223-7/19/11...$15.00
https://doi.org/10.1145/3356401.3356411

way to prove guarantees on the end-to-end delay and other metrics such as the maximum queue length in network elements. Envelope-based approaches such as Network Calculus (NC) [25], which describe data flows with accumulative functions and network elements with service curves, are commonly used to analyze real-time systems. Thus, it is no surprise that NC has already been applied not only to analyze networks implementing non-time-triggered QoS mechanisms such as Weighted Fair Queuing of Integrated Services (IntServ) or TSN credit-based shaping as used, for instance, in multimedia communication [17, 18, 24], but also for time-triggered network elements with intermittent service (cf. Section 2). However, the research, often targeted at very specific technologies, is predominantly considering scenarios where the underlying service process ultimately offers constant-rate service, i.e., the data is forwarded with constant rates during service intervals.

But what happens if you abandon the assumption of service processes with ultimately constant-rate service (e.g., in case of wireless MIMO links where the number of streams [1, 2], e.g., influenced by the number of active antennas, is changed)? And how bad is it, to use time-invariant functions to model time-variant systems? These questions are still open. Therefore, we focus on the fundamental aspects of the analysis of such network elements in this paper. In detail, we make the following contributions: **1)** We identify two archetypes of time-triggered network elements with intermittent service. Additionally, we present type-specific time-variant approaches for deriving service curves in the NC framework since in absence of specific restrictions (e.g., constant-rate) on the service process of these network elements, distinct service curve formulations are required. **2)** We present approaches for deriving time-invariant service curves for these two archetypes of network elements. In particular, we show that a leftover service curve approach is not applicable to obtain valid time-invariant service curves for both types, and provide an alternative approach to model time-triggered network elements with intermittent service with time-invariant service curves. **3)** We investigate the difference of worst-case backlog and worst-case delay between time-variant and time-invariant service curve for the two types of time-triggered systems, using arrival-curve independent metrics.

In the remainder of this paper, we discuss the related work in Section 2, introduce the basic concepts of NC in Section 3, describe the two types of time-triggered network elements with intermittent service in Section 4, and present the service curve formulations in Section 5. We evaluate the difference between time-variant and time-invariant approaches for modeling the service curves in Section 6, and conclude with a summary and outlook in Section 7.

2 RELATED WORK

NC has been applied to some time-triggered network elements with intermittent service. TDMA systems have been targeted in [15, 22, 29]. In [22, 29], insights from the NC analysis of a TDMA system (distributed embedded system with bus-interconnection [29], wireless sensor network [22]) are used as input for the optimization of the TDMA schedule. In [15], the tightness of worst-case bounds is improved, taking into account various scheduling policies and packetization of data. More recently, Ethernet-based real-time networks (TTEthernet, IEEE 802.1 TSN) have been subject to NC analysis [31–33]. Concerning the analysis of Ethernet-based real-time networks, the goal is to derive worst-case bounds for the traffic factoring in the impact of various, implementation-specific integration effects. For example, for the analysis of IEEE 802.1 TSN networks, the integration effects of Credit-based Shapers, Time-aware Shapers, and Frame-preemption is considered in [33], and in [31] the combination of Time-aware Shapers, Length-aware Scheduling, and strict-priority frame-selection policy is considered.

All of the work above has in common that the service curve formulations presented in these papers assume one service interval per cycle and all of the service curve formulations are time-invariant, and can—eventually—be traced back to a peak-rate service process with constant service rate. Besides explicitly anticipating more intricate schedules with multiple service intervals per cycle in our service curve formulations, we show that these approaches are not sufficient for service curves for time-triggered network elements with intermittent service processes with ultimately non-constant service rate.

A different approach is pursued in [7], where *stochastic*, time-variant NC is combined with a measurement-based approach to model cellular sleep scheduling. Service intermittence due to "sleeping" is considered a random process in [7], whereas we assume a repeating, deterministic schedule for service intermittence. In [6, 7], also the notion of regenerative service processes (i.e., the service process is restarted at regeneration points) is explicitly mentioned. While we will also start with time-variant service curve formulation (albeit using deterministic NC with cyclic schedules), we cover systems with and without "restarting" service processes, and compare them to time-invariant service curve formulations.

3 NETWORK CALCULUS

Deterministic Network Calculus [9, 14] is a modeling framework to derive worst-case bounds on performance metrics such as backlog and delay. One of the core concepts of NC is the description of properties of the arrival processes by arrival curves and the description of queuing network elements by service curves. Arrival curves and service curves are cumulative, non-decreasing functions in

$$\mathcal{F}_{t.i.} = \left\{ f : f(t) \geq f(\tau) \geq 0, \forall t \geq \tau \geq 0; \text{else } f(t) = 0 \right\} \quad (1)$$

for time-invariant (t.i.) functions [20], and respectively we have

$$\begin{aligned}\mathcal{F}_{t.v.} = \{\ & f : f(s,t) \geq 0, \forall t \geq s; \text{else } f(s,t) = 0 \\ & \wedge \forall u \leq v \leq w : f(u,v) + f(v,w) = f(u,w)\ \}\end{aligned} \quad (2)$$

for time-variant (t.v.) functions [4, 16]. We consider a continuous time-domain, i.e., (s and) $t \in \mathbb{R}$. Arrival curves indicate the accumulated amount of arriving traffic for a time interval. Conversely, service curves describe the accumulated amount of offered service for a time interval. In the following, arrival curves and service curves are left-continuous in each time variable. NC defines operations such as convolution and deconvolution (using min-plus / max-plus algebra) on arrival and service curves, and allows concatenating systems in a way similar to common system theory [25].

In this paper, we derive both, time-variant service curves and time-invariant service curves for time-triggered network elements. The time-variant service curve can be expected to provide higher

Table 1: Basic NC operations on $\mathcal{F}_{t.i.}$ /$\mathcal{F}_{t.v.}$.

operation	time-invariant $f, g \in \mathcal{F}_{t.i.}$
pointwise min. $\oplus$	$f \oplus g(t) = \inf[f(t), g(t)]$
convolution $\otimes$	$f \otimes g(t) = \inf_{0\leq\tau\leq t}[f(t-\tau)+g(\tau)]$
left-deconv. $\oslash$	$f \oslash g(t) = \sup_{\tau\geq 0}[f(t+\tau)-g(\tau)]^+$
operation	**time-variant $f, g \in \mathcal{F}_{t.v.}$**
pointwise min. $\oplus$	$f \oplus g(s,t) = \inf[f(s,t), g(s,t)]$
convolution $\otimes$	$f \otimes g(s,t) = \inf_{s\leq\tau\leq t}[f(s,\tau)+g(\tau,t)]$
left-deconv. $\oslash$	$f \oslash g(s,t) = \sup_{\tau\leq s\leq t}[f(\tau,t)-g(\tau,s)]^+$

fidelity regarding the modeled system. However, this comes at a price: a) evaluating properties of a system with a time-variant description is more difficult (cf. Equation 6-8 in Section 3.4), b) important algebraic properties are lost with time-variant analysis compared to time-invariant analysis. Most prominently, the convolution operation is not commutative anymore, which would simplify the analysis of multi-hop scenarios. Moreover, regarding the literature, time-invariant NC has attracted far more attention. This has the practical consequence that none of the major toolboxes and libraries for NC (DiscoDNC [8], CyNC [28], RTC [30], RTaW Pegase [27], etc.) advertise support for time-variant formulations. Therefore, we investigate time-invariant service curves, too.

Next, we provide formal definitions for arrival and service curves, and the NC operations.

3.1 Min-Plus Operations

NC uses the algebraic structure $\{\mathcal{F}, \oplus, \otimes\}$ [12, 16, 21, 25]. The basic operations on functions in $\mathcal{F}$ used in this paper (cf. Table 1, with $[x]^+ = \max(0, x)$) comprise the pointwise-minimum $\oplus$, the convolution $\otimes$, and the left-deconvolution $\oslash$ [12, 16] .

3.2 Arrival Curves

Assume that the cumulative arrivals at time t are given by $A(t)$. Then the arrivals are constrained by arrival curve α iff

$$\forall s \leq t : A(t) - A(s) \leq \begin{cases} \alpha(t-s) & \alpha \text{ is t.i.} \\ \alpha(s,t) & \alpha \text{ is t.v.} \end{cases}, \tag{3}$$

or alternatively using the convolution

$$\begin{cases} A(t) \leq A \otimes \alpha(t) & \text{t.i} \\ A_{t.v.}(s,t) = [A(t) - A(s)]^+ \leq A \otimes \alpha(s,t) & \text{t.v.} \end{cases}. \tag{4}$$

3.3 Service Curves

Assume that the cumulative departures of a particular system are given by $R(t)$. Then that system offers a (simple) service curve β to the flow with arrivals $A(t)$ iff

$$\forall t \geq 0\ \exists s \leq t : R(t) - A(s) \geq \begin{cases} \beta(t-s) & \beta \text{ is t.i.} \\ \beta(s,t) & \beta \text{ is t.v.} \end{cases}. \tag{5}$$

An important, more restrictive type of service curve is the so-called strict service curve [5, 10]. A strict service curve has to satisfy

$$\forall \text{ backlogged intervals } (s,t] : R(t) \geq R(s) + \begin{cases} \beta(t-s) & \beta \text{ is t.i.} \\ \beta(s,t) & \beta \text{ is t.v.} \end{cases}.$$

The strict service curve definition is related to the work-conserving property since it provides service guarantees over *every* continuously backlogged time interval, i.e., intervals where continuously $R(t) < A(t)$. In contrast to that, a system with the more general, simple service curve might idle during such time-intervals.

3.4 Deterministic Worst-Case Bounds

The goal of NC analysis is to calculate system properties such as worst-case bounds on delay and the amount of backlog. Arrival curve and service curve are required to this end (disregarding special cases, such as the availability of minimum and maximum services curves [25]). Given arrival curve α and service curve β, the maximum backlog (the biggest amount of data being buffered in the system) [20] can be computed by

$$\text{backlog}(\alpha, \beta) \leq \begin{cases} \alpha \oslash \beta(0) & \alpha,\beta \text{ t.i.} \\ \sup_{t\geq 0}[\alpha \oslash \beta(t,t)] & \alpha,\beta \text{ t.v.} \end{cases}. \tag{6}$$

The maximum backlog can be thought of as the maximal vertical distance between the arrival curve constraining the arrivals in the system and the service curve of the system.

Equivalently, in the time-invariant case the maximum virtual delay [20] is given by $\text{v.delay}(\alpha, \beta) \leq \inf[w \geq 0 : \alpha \oslash \beta(-w) \leq 0]$

$$= \inf\left[w \geq 0 : \sup_{\tau\geq 0}\{\alpha(\tau) - \beta(\tau + w)\} \leq 0\right], \tag{7}$$

and in the time-variant case by

$$\text{v.delay}(\alpha, \beta) \leq \sup_{t\geq 0}\left[\inf\left\{w \geq 0 : \alpha \oslash \beta(t+w, t) \leq 0\right\}\right] \tag{8}$$

$$= \sup_{t\geq 0}\left[\inf\left\{w \geq 0 : \sup_{\tau\leq t+w}[\alpha(\tau,t) - \beta(\tau, t+w)] \leq 0\right\}\right].$$

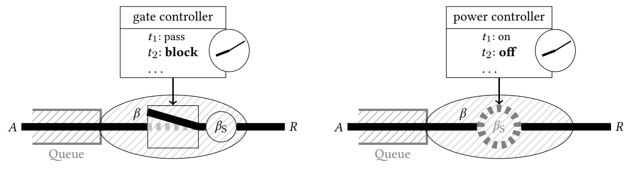
The maximum virtual delay can be visually interpreted as the maximal horizontal distance between arrival curve and service curve, which is the maximal time needed for the value of the departures to reach the same value as the arrivals [17]. In case of FIFO, i.e., data serviced in the order of its arrival, the maximal virtual delay becomes the maximum waiting time for data in the system.

4 NETWORK ELEMENTS: SYSTEM MODELS

Next, we will introduce the system models for the two different time-triggered network elements with intermittent service (cf. Figure 1). We will use them to highlight the fundamental principles and capabilities of the different service curve approaches in Section 5. Going on, we use the fluid model where data can arrive and receive service in arbitrarily sized quantities, i.e., packetization is ignored. We assume initially empty systems with no prior arrivals before $t = 0$ and no queue overflow.

In this paper, we focus on a single network element[1]. Data arriving at the network element is modeled by arrival process A with arbitrary, known arrival curve α. We have no influence on arrival process A, i.e., in the remainder of this paper, we consider arrival process A externally given. Data from arrival process A can be queued at the network element. We assume that enqueuing and dequeuing at the network element occurs instantaneously. During times controlled by the time-triggered controller, service process S (defined by β_S) offers service to the arrivals of arrival process A. Cumulative departures at the network element are denoted by R.

[1]The special case of a single network element can already model an idealized communication bus or a network where every host is directly attached to one central switch.



(a) Time-triggered blocking. **(b) Time-trigg. halt and restart.**

Figure 1: Service curve β describes a composite network element comprised of a service process S with service curve β_S manipulated by a time-triggered mechanism.

The service curve β_S of the service process S (in isolation) is not to be confused with the service curve β of the complete network element. The service curve β describes the service offered to arrivals from arrival process A, and depends on β_S, the type of time-triggered mechanism, and the schedule. The time-triggered mechanism operates according to a cyclic schedule which contains entries for the time instances when the controller "enables" the service process S to offer service to the arrivals of arrival process A ("enabling time"), and the corresponding time instances when the controller "disables" service process S from offering service to arrivals of arrival process A ("disabling time"). To this end, the time-triggered mechanism queries the current time t_{curr} from the clock of the network element. Before explaining the meaning of enabling and disabling for the two types of network elements, we first specify the parameters of the repeating schedule (cf. Table 2), with

$t_{er,i}$:= ith enabling time of S *relative* to the start of the cycle,
$t_{dr,i}$:= ith disabling time of S *relative* to the start of the cycle,
t_{cycle} := period of the schedule (cycle length).

The number of enabled intervals per cycle is given by $(i_{\max} + 1)$. The enabled intervals must not overlap, and we require $0 \le t_{er,0} < t_{dr,0} < t_{er,1}$ and $t_{dr,i_{\max}} \le t_{\text{cycle}}$.

Table 2: Repeating schedule with cycle length t_{cycle}.

Interval i:	t_{enabling}	$t_{\text{disabling}}$
0	$t_{er,0}$	$t_{dr,0}$
1	$t_{er,1}$	$t_{dr,1}$
...	. . .	

Service process S is enabled at t_{curr} if $\exists i \in [0, i_{\max}]$ such that

$$\left(t_{\text{curr}} \bmod t_{\text{cycle}} > t_{er,i}\right) \bigwedge \left(t_{\text{curr}} \bmod t_{\text{cycle}} \le t_{dr,i}\right) \qquad (9)$$

and disabled otherwise.

Regarding the time-triggered mechanisms, we consider two cases (cf. Figure 1) where enabling and disabling of service process S are implemented differently.

In the first case (Figure 1a), there is an additional element—a gate—in between queue and service process S. The service process S has a strict, time-invariant service curve β_S and starts operating, (i.e., "is powered on") at time $t = 0$, and runs continuously. Starting at each enabling time, the gate controller "opens" the gate, i.e., data in the queue is transparently *passed-through* to the service process S until the next disabling time. At each disabling time, the gate controller "closes" the gate, i.e., service process S is *blocked* from accessing the arrivals of arrival process A until the next enabling time. We will refer to this type of network element as time-triggered network element with *blocking*. Note that the service process S is only disabled from the perspective of the arrival process A. While the service process is blocked for arrivals from A, the service process S could service data from other arrival processes, i.e., $\beta_S(\tau)$ is the total amount of service the service process S has offered up to time $\tau = t$ of which only the fraction inside enabled intervals is available to arrivals from arrival process A, and t is the time in the domain of the network element. This behavior can, e.g., occur in TSN networks, where gates restrict access of the different queues of an output port to the link.

In the second case (Figure 1b), the service process S is manipulated directly by a so-called power controller. At each enabling time, the power controller (re-)starts the service process S. If the service process S is active, it offers the strict, time-invariant service curve β_S to the arrivals of arrival process A until the next disabling time when the power controller halts the service process S. If the service process S is disabled, it offers no service at all. We will refer to this type of network element as time-triggered network element with *halt-restart*. Here, β_S has only temporary meaning, because $\beta_S(\tau)$ is the amount of service the service process S can offer $\tau = t - t_{\text{enabling}}$ time units into an enabled interval. Thus, in case of halt-restart-behavior, time instances where the service process S is restarted can be considered as renewal-points [6] of the service process S.

5 DERIVING SERVICE CURVES

After presenting the basic NC background and introducing the system models in the previous sections, next we derive time-variant service curves for time-triggered network elements with intermittent service in Section 5.1. The time-invariant service curves are subsequently introduced in Section 5.2.

In the remaining sections, we will use the following notation: The corresponding index in the cyclic schedule which belongs to the overall n-th enabled interval is given by $\varphi(n) = n \bmod (i_{\max} + 1)$. For each enabled interval n we introduce the symbols

$$\text{and} \quad \left.\begin{aligned} t_{e,n} &= t_{er,\varphi(n)} \\ t_{d,n} &= t_{dr,\varphi(n)} \end{aligned}\right\} + \left\lfloor \frac{n}{i_{\max} + 1} \right\rfloor \cdot t_{\text{cycle}}$$

where $t_{e,n}$ is the start, and $t_{d,n}$ is the end of the overall n-th enabled interval (i.e., $t_{er,i}, t_{dr,i}$ are relative to the beginning of a schedule cycle, $t_{e,n}, t_{d,n}$ are absolute times).

5.1 Time-Variant Service Curve

The service offered to incoming data in a certain time interval depends on the position of that time interval on the time-axis. This observation suggests using a time-variant formulation [12, 13].

Depending on the type of time-triggered network element (cf. Section 4), the time-variant direct service curve is either given by the formula in Theorem 5.1 or Theorem 5.2.

Theorem 5.1 (Time-triggered blocking of service process). *Let $\beta_S(t)$ be the time-invariant, strict service curve of the continuously running service process S. In a time-triggered queuing system where access to this service process S is blocked outside of enabled intervals according to a given repeating schedule, the time-variant service curve is given by*

$$\beta(s,t) = \sum_{n=0}^{\infty} \left[\beta_S\left(\min\left(t, t_{d,n}\right)\right) - \beta_S\left(\max\left(s, t_{e,n}\right)\right)\right]^{+}. \qquad (10)$$

Proof. It holds that $\beta(s,t) \in \mathcal{F}_{t.v.}$, since $\beta(s,t)$ is the sum of $f(t), f \in \mathcal{F}_{t.i.}$. Let $A(t)$ and $R(t)$ be the arrival function and departure function, respectively. Considering an arbitrary backlogged interval $(s,t]$, we show that $R(t) - A(s) \geq \beta(s,t)$. We can distinguish the following cases, starting with those cases where the interval intersects with at most one gate open interval, before we cover the case where $(s,t]$ intersects with multiple gate open intervals of the flow of interest.

If no enabled interval is included in $(s,t]$ at all, then for all enabled intervals before $(s,t]$, we have $\left[\beta_S(t_{d,n}) - \beta_S(s)\right]^+$ and, conversely, for all enabled intervals after $(s,t]$, we have

$$\begin{aligned}&\left[\beta_S(t) - \beta_S(t_{e,n})\right]^+ \\ =& \left[\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n}))\right]^+ = 0.\end{aligned}$$

Since for $(s,t]$ which do not intersect with any enabled interval $\min(t, t_{d,n}) \leq \max(s, t_{e,n})$ and $\beta_S \in \mathcal{F}_{t.i.}$, $R(t) - A(s) \geq 0$. Thus, enabled intervals not included in $(s,t]$ contribute nothing to $\beta(s,t)$. We need $[\cdot]^+$ in Equation 10, since it is possible that $\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n})) < 0$.

If the interval $(s,t]$ starts in the enabled interval, but the enabled interval ends before t (cf. enabled interval $n = 0$, $(s_1, t_1]$ in Figure 2), we have $R(t) - A(s) \geq \beta_S(t_{d,n}) - \beta_S(s)$.

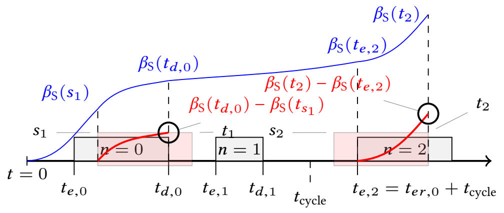


Figure 2: Start or end of enabled interval included in $(s,t]$.

In case, the interval $(s,t]$ starts before the enabled interval, but ends before the enabled intervals ends again (cf. enabled interval $n = 2$, $(s_2, t_2]$ in Figure 2), we have $R(t) - A(s) \geq \beta_S(t) - \beta_S(t_{e,n})$.

If the enabled interval includes the interval $(s,t]$ (cf. enabled interval $n = 0$, $(s_1, t_1]$ in Figure 3), we have $R(t) - A(s) \geq \beta_S(t) - \beta_S(s)$.

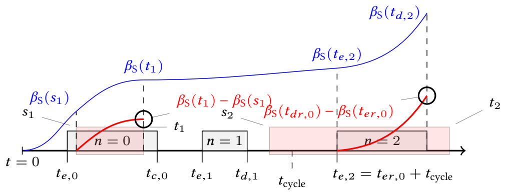


Figure 3: Enabled interval contains $(s,t]$ or vice versa.

If the interval $(s,t]$ includes the enabled interval (cf. enabled interval $n = 2$, $(s_2, t_2]$ in Figure 3), we have $R(t) - A(s) \geq \beta_S(t_{d,n}) - \beta_S(t_{e,n})$.

If the interval $(s,t]$ intersects with multiple enabled intervals (cf. Figure 6, where $(s,t]$ intersects with three enabled intervals), we define $n_s = \max\{n : t_{d,n} \leq s\} + 1$ the first enabled interval intersecting with $(s,t]$, likewise, the last intersecting interval is $n_t = \min\{n : t_{e,n} > t\} - 1$. According to the behavior of the time-triggered element with blocking, the value of R increases only when the service process S is enabled to offer service to the arrivals of A. For every $n \in [n_s, n_t]$ the lower bound of the amount of departures (i.e., arrivals which have received service from S) is given by $\left[\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n}))\right]^+$, and the lower bound of departures over the whole interval $(s,t]$ is then given by accumulating the amount of service

$$R(t) - A(s) \geq \sum_{n=n_s}^{n_t} \left[\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n}))\right]^+$$

in the individual enabled intervals in $[n_s, n_t]$. Considering, that

$$\sum_{n=0}^{n_s - 1} \left[\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n}))\right]^+ = 0,$$

$$\text{and} \sum_{n=n_t+1}^{\infty} \left[\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n}))\right]^+ = 0,$$

we can write

$$R(t) - A(s) \geq \sum_{n=0}^{\infty} \left[\beta_S(\min(t, t_{d,n})) - \beta_S(\max(s, t_{e,n}))\right]^+,$$

thus $R(t) - A(s) \geq \beta(s,t)$. □

Theorem 5.2 (Time-triggered halting and restarting of service process). *Let $\beta_S(t)$ be the time-invariant, strict service curve of a service process S. In a time-triggered queuing system where service process S is restarted at the beginning of each enabled interval and halted at the end of each enabled interval according to a given repeating schedule, the time-variant service curve is given by*

$$\beta(s,t) = \sum_{n=0}^{\infty} \beta_S(\min(t, t_{d,n}) - \max(s, t_{e,n})). \tag{11}$$

Proof. We prove Theorem 5.2 analogously to the proof of Theorem 5.1. It holds that $\beta(s,t) \in \mathcal{F}_{t.v.}$ (cf. proof of Theorem 5.1). Considering an arbitrary backlogged interval $(s,t]$, we show that $R(t) - A(s) \geq \beta(s,t)$, with $A(t)$ and $R(t)$ the arrival function and departure function, respectively. We can distinguish the same cases as before:

If no gate enabled interval is included in $(s,t]$ at all, then for all enabled intervals before $(s,t]$, we have $t_{d,n} - s \leq 0$. For all enabled intervals after $(s,t]$, we have $t - t_{e,n} \leq 0$. Therefore, $R(t) - A(s) \geq \beta_S(\min(t, t_{d,n}) - \max(s, t_{e,n})) = 0$, since $\beta_S \in \mathcal{F}_{t.i.}$.

If the interval $(s,t]$ starts in the enabled interval, but the enabled interval ends before t (cf. enabled interval $n = 0$, $(s_1, t_1]$ in Figure 4), we have $R(t) - A(s) \geq \beta_S(t_{d,n} - s)$.

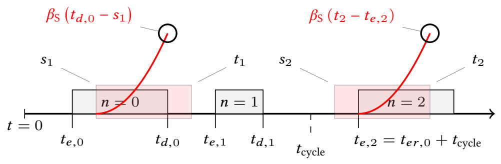


Figure 4: Start or end of enabled interval included in $(s,t]$.

If the interval $(s,t]$ starts before the enabled interval, but ends before the enabled interval ends again (cf. enabled interval $n = 2$, $(s_2, t_2]$ in Figure 4), we have $R(t) - A(s) \geq \beta_S(t - t_{e,n})$.

If the enabled interval includes the interval $(s,t]$ (cf. enabled interval $n = 0$, $(s_1, t_1]$ in Figure 5), we have $R(t) - A(s) \geq \beta_S(t - s)$.

If the interval $(s,t]$ includes the enabled interval (cf. enabled interval $n = 2$, $(s_2, t_2]$ in Figure 5), we have

$$R(t) - A(s) \geq \beta_S(t_{d,n} - t_{e,n}) = \beta_S\left(t_{dr,\varphi(n)} - t_{er,\varphi(n)}\right).$$

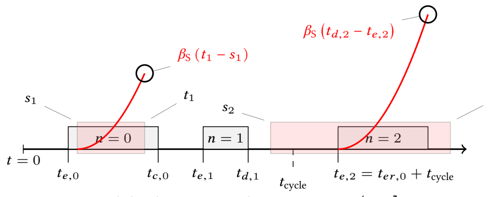


Figure 5: Enabled interval contains $(s,t]$ or vice versa.

If the interval $(s,t]$ intersects multiple enabled intervals (cf. Figure 6, where $(s,t]$ (partially) covers three enabled intervals), the argumentation is analogous to the proof of Theorem 5.1. The main difference is that during each enabled interval n in $(s,t]$ the service process S of the network element with halt-restart generates at least $\beta_S(\min(t,t_{d,n})-\max(s,t_{e,n}))$ units of departing traffic. The lower bound of the total departures in any interval $(s,t]$ is the accumulated amount of the departures in the enabled intervals, i.e.,

$$R(t)-A(s)\geq\sum_{n=0}^{\infty}\beta_S\left(\min(t,t_{d,n})-\max(s,t_{e,n})\right)=\beta(s,t).$$

□

In Equation 10 and Equation 11, respectively, the summands

$$\begin{cases}\left[\beta_S(\min(t,t_{d,n}))-\beta_S(\max(s,t_{e,n}))\right]^+,\\ \beta_S(\min(t,t_{d,n})-\max(s,t_{e,n}))\end{cases}$$

describe the service that can be offered by server S with service curve β_S during the n-th enabled interval in the time interval $(s,t]$. For $\beta_S(t)=[C\cdot t]^+$ the two theorems yield the same service curve β, since $\beta_S(t-s)=\beta_S(t)-\beta_S(s)$. Therefore, we will use this choice of β_S in Figure 6 and Figure 7 to illustrate the construction of $\beta(s,t)$ for $(s,t]$ overlapping more than one enabled interval and the time-variant property for both theorems. The dashed curve depicted in Figure 6 results from fixing s in $\beta(s,t)$. Shifting the whole interval $(s,t]$ from Figure 6 on the time axis yields a differently shaped service curve, as shown in Figure 7.

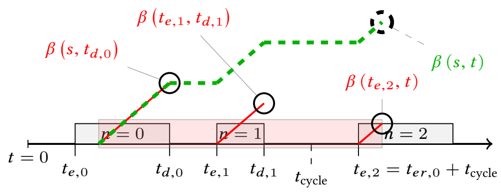


Figure 6: Offered service over multiple enabled intervals.

It is easy to see that static time-triggered scheduling results in a time-variant service curve, even if β_S is time-invariant, since $\exists\delta$ for non-trivial schedules such that $\beta(s,t)\neq\beta(s+\delta,t+\delta)$.

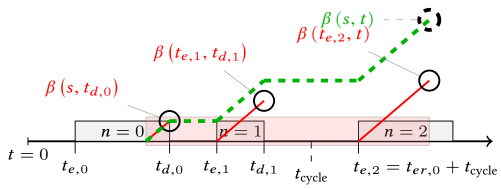


Figure 7: Offered service for shifted interval $(s,t]$.

We want to point out, that in general the service offered by time-triggered network elements with blocking differs massively from time-triggered network elements with halt-restart even if the service process S has the same service curve β_S. For example, consider a service process S with some kind of initialization "cost", i.e. some latency or lower service rate at $t=0$. For network elements with blocking, this initialization has to be accounted for *only once*, whereas for networks elements with halt-restart, this initialization has to be accounted *for each enabled interval*. This necessitates the distinction between blocking behavior and halt-restart behavior in general.

5.2 Time-Invariant Service Curves

After having developed formulations for the time-variant service curve, we are going to provide two formulations for time-invariant service curves using a leftover approach in Section 5.2.1, and a direct approach in Section 5.2.2.

5.2.1 Leftover Service Curve with Instant Arrivals. The key idea of the leftover service curve involves the subtraction of the amount of service that is offered to *another* arrival process A^v (which might be an aggregate of multiple arrival processes) from the total service offered by the service process S. Hence arrival process A can receive only that amount of service from S that is "left-over" by A^v (hence the name). The leftover service approach is well-known in NC, and has been used in [32, 33] to model time-triggered blocking behavior. The particular leftover service curve approach we present here follows the principles of the approach presented in [32, 33]. However, we will show that in general the leftover service curve approach is not applicable to time-triggered network elements with halt-restart (cf. Theorem 5.2).

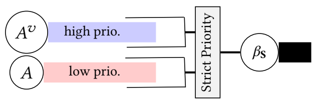


Figure 8: Artificial arrival process A^v occupies service process S during disabled intervals of actual arrival process A.

For the leftover service curve approach, an additional (virtual) arrival process A^v is constructed, which is prioritized by the service process S over the arrival process A (cf. Figure 8). The *arrivals* of the (virtual) arrival process A^v are scheduled according to the *inverse schedule* of the network element, i.e., at each disabling time in the schedule of the network element, there are arrivals for A^v. The amount of arrivals of the virtual arrival process A^v is chosen to occupy the service process S (at least) until the corresponding disabling time for arrival process A^v, which is equivalent to the next enabling time in the schedule of the network element for the actual arrival process A. Thus, A^v effectively "disables" the service process S from servicing any data of arrival process A during the disabled intervals of the original schedule for arrivals from arrival process A.

To construct A^v in accordance with our system model (cf. Section 4), we construct the arrival curve for the arrival process A^v from a set of intermediate arrival curves $a^v_{\text{imd},n}$. The intermediate arrival curve

$$a^v_{\text{imd},n}(t)=\sum_{k=n}^{\infty}\beta\left(t^v_{e,k},t^v_{d,k}\right)\cdot\mathbb{1}_{\{t>t^v_{e,k}\}}\tag{12}$$

with $\mathbb{1}_{\{t>T\}}=\{1:t>T;0:\text{otherwise}\}$ and

$$\beta\left(t^v_{e,k},t^v_{d,k}\right)\text{ with }\left.\begin{array}{l}t^v_{e,k}=t^v_{er,\varphi(k)}\\ t^v_{d,k}=t^v_{dr,\varphi(k)}\end{array}\right\}+\left\lfloor\frac{k}{i^v_{\max}+1}\right\rfloor\cdot t^v_{\text{cycle}}\tag{13}$$

models the instantaneous arrival of the maximum amount of data from arrival process A^v that can receive service during the k-th enabled interval of the arrival process A^v *in the inverse schedule*, immediately at the beginning of the k-th enabled interval of the arrival process A^v. The first enabled interval to be included in each intermediate arrival curve $a^v_{\mathrm{imd},n}$ is the overall n-th interval in the *inverse schedule*. We construct the artificial arrival curve for arrival process A^v

$$\alpha^v(t) = \sup_{n \in [0, n^v_{\max}]} \left(a^v_{\mathrm{imd},n} \left(t + t^v_{e,n} \right) \right), \tag{14}$$

by taking the supremum of the shifted intermediate arrival curves. To account for the worst-case over all times when the service process S is disabled for the actual arrival process A, each intermediate arrival curve used in the construction of the arrival curve for the (virtual) arrival process A^v is shifted such that the first arrival of data of the respective intermediate arrival curve occurs at the time-origin. This artificial arrival curve a^v thus emulates the effect of the service process S being not available during disabled intervals. The arrival curve a^v for A^v given in Equation 14 is artificial in the sense that it is introduced only for modeling purposes and does not necessarily reflect the actual arrival curve of any other arrival process (e.g., other traffic flows at time-triggered network elements with blocking). Note, that if we do not know anything about the behavior of β_S, then $n^v_{\max} = \infty$. For $n^v_{\max} = \infty$, it is infeasible to evaluate Equation 14 computationally. However, if β_S conforms to one of the asymptotic behaviors[2] given in Table 3, $n^v_{\max}$ can be bounded by $\left\lfloor \frac{n^v_{\max}}{i^v_{\max}+1} \right\rfloor \cdot t^v_{\mathrm{cycle}} \geq T + t_{\mathrm{cycle}}$.

Table 3: Asymptotic behaviors for time-invariant service functions with bounded $n^v_{\max}$.

const. rate [4]	$\exists \sigma, \rho \in \mathbb{R}, \forall t : f(t) = \rho t + \sigma$ or $\forall t : f(t) = \infty$
ultimately const. rate [4]	$\exists T, \exists \sigma, \rho \in \mathbb{R}, \forall t > T : f(t) = \rho t + \sigma$ or $\forall t > T : f(t) = \infty$
pseudo-periodic [4]	$\exists c, \exists d \in \mathbb{R} \setminus 0, \forall t : f(t+d) = f(t) + c$
ult. pseudo-periodic [4]	$\exists T, \exists c, \exists d \in \mathbb{R} \setminus 0, \forall t > T : f(t+d) = f(t) + c$
accelerating	$\forall a, d \in \mathbb{R}^+, \forall t : f(t+d) - f(t) \leq f(t+d+a) - f(t+a)$
ult. accelerating	$\exists T, \forall a, d \in \mathbb{R}^+, \forall t > T : f(t+d) - f(t) \leq f(t+d+a) - f(t+a)$

Subtracting the arrival curve $\alpha^v(t)$ from the service curve of the service process β_S, yields the leftover service curve

$$\beta(t) = \left[\sup_{s \leq t} \left(\beta_s(s) - \alpha^v(s) \right) \right]^+ . \tag{15}$$

The $[\ldots]^+$ expression is necessary due to the model with instantaneous arrivals $\beta_s(t) - \alpha^v(t) \notin \mathcal{F}_{t.i}$.

Table 4: Example schedule and corresponding inverse schedule for A^v with $i_{\max} = i^v_{\max} = 2$ and $t_{\mathrm{cycle}} = t^v_{\mathrm{cycle}} = 8$.

Interval i:	Original Sched.		Inverse Sched. for A^v	
	t_{enabling}	$t_{\mathrm{disabling}}$	t^v_{enabling}	$t^v_{\mathrm{disabling}}$
0	0	1	1	2
1	2	4	4	6
2	6	7	7	8

[2] [4] uses the term affine instead of constant-rate. In the related work, usually an affine or ultimately affine service function is chosen for β_S for which Equation 15 can be used in practice since $n^v_{\max}$ is finite.

The shifted intermediate arrival curves, the artificial arrival curve, and the leftover service curve using the cyclic schedules from Table 4 are shown in Figure 9. A proof for the leftover service curve has already been provided in [32, 33], and can be similarly applied to our modified arrival curve for the (virtual) arrival process A^v.

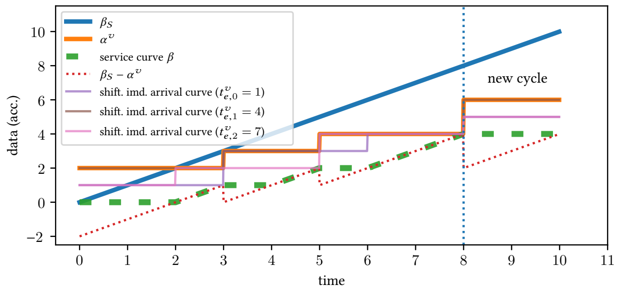


Figure 9: Leftover service for cyclic schedule in Table 4.

However, in the context of time-triggered network elements with intermittent service, the leftover service curve approach from Equation 15 is in general not applicable to time-triggered network elements with halt-restart (cf. Theorem 5.2). Consider a scenario with $\beta_S(t) = \{t^2 : t \geq 0, 0 \text{ otherwise}\}$, and a schedule where disabled intervals and enabled intervals of length 1 alternate, starting with a disabled interval for arrival process A. Then, according to Equation 13 and Theorem 5.2 we get $\beta\left(t^v_{e,k}, t^v_{d,k}\right) = 1$ per disabled interval of arrival process A. However, e.g., for $t = 3$, according to Equation 15, arrival process A will have been offered at least $\beta(3) = \beta_S(3) - 2 = 7$ units of service. This is a contradiction since even in the best case, arrival process A will have observed only two complete, enabled intervals in each of which it would have been offered 1 unit of service. While this choice of $\beta_S(t)$ is quite contrived, with current wireless technology it is possible to vary the bandwidth per time interval in a controlled way, e.g., by adjusting the number of streams of MIMO links [1, 2] or 5G resource blocks assigned to a connection.

5.2.2 *Direct Service Curve.*

Here, we provide an alternative to the leftover service curve approach from Section 5.2.1 that yields a valid time-invariant service curve for time-triggered network elements with halt-restart. For this approach, we again reuse the time-variant service curve, more precisely Equation 10 and Equation 11, respectively, to "directly" construct a time-invariant service curve.

The main distinction between time-variant service curve and time-invariant service curve is that the former describes the lower bound of offered service in any *specific* interval, whereas the latter describes the lower bound of offered service *irrespective of the position* of the interval. Therefore, the basic idea to derive the direct time-invariant service curve from the time-variant service curve is to get the overall lower bound of offered service by considering *all* intervals of the time-variant service curve.

Theorem 5.3. *The time-invariant, direct service curve for a time-triggered network element is given by the infimum*

$$\beta(t) = \inf_{n \in [0, n_{max}]} \left(\begin{cases} \beta(0, t) & n = 0 \\ \beta\left(\left(t_{offset,n}\right), \left(t + t_{offset,n}\right)\right) & n > 0 \end{cases} \right), \tag{16}$$

$$t_{offset,n} = t_{dr,\varphi(n)-1} + \left\lfloor \frac{n-1}{i_{max}+1} \right\rfloor \cdot t_{cycle} \tag{17}$$

where $\beta(s, t)$ is the time-variant, direct service curve for the time-triggered network element with given repeating schedule and service process S with service curve β_S. For time-triggered network elements with halt-restart (cf. Equation 11), n_{max} can be set to $i_{max} + 1$, since in all cycles the same amount of service is being offered. For time-triggered network elements with blocking (cf. Equation 10), we have $n_{max} = \infty$ in general.

Proof. Recalling the definition of the service curves from Section 3.3, the infimum of offered service in any backlogged interval of length t equals the value of the time-invariant service curve at time t. Due to the cyclic properties of the time-triggered mechanisms, this lower bound can be found by evaluating the time-variant direct service curve $\beta(s, s+t)$ for n_{max} intervals with length t using properly chosen starting points s. If the starting point s of the interval with length t for which we evaluate the time-variant service curve $\beta(s, s + t)$ is equal to the time when an enabled interval has just ended, no service is offered until the next enabled interval. This is potentially a worst-case. As an intermediate step, we define the intermediate service curves

$$b_{\text{imd},n}(t) = \begin{cases} \beta(0, t) & n = 0 \\ \beta\left(\left(t_{\text{offset},n}\right), \left(t + t_{\text{offset},n}\right)\right) & n > 0 \end{cases}. \tag{18}$$

The values of $b_{\text{imd},n}(t)$ are obtained by evaluating $\beta(s, t)$ for a fixed s starting at the disabling times of the different enabled intervals n. To make the step from the intermediate service curves to the time-invariant service curve, each intermediate service curve $b_{\text{imd},n}(t)$ is first shifted in the time domain (cf. Figure 10) to the left by the distance of the last disabling time before $t_{e,n}$ from the origin. In the second step, we apply the infimum over all shifted intermediate service curves. Thus given arrival function $A(t)$ and departure function $R(t)$, $R(t) \geq A(s) + \beta(t - s)$. □

Note, that for network elements where the service process S gets blocked according to the cyclic schedule (cf. Theorem 5.1) this service curve is again infeasible to evaluate computationally since $n_{\text{max}} = \infty$. But for asymptotic behaviors of β_S as defined in Table 3, n_{max} can be bounded to a finite value similarly as in the leftover service curve approach. The basic idea of taking the lower bound of several intermediate service curves can be recovered from [31], which considers the special case where the intermediate service curves are derived from the traditional TDMA service curve with constant-rate service.

In Figure 10, we illustrate the construction of the direct, time-invariant service curve for the schedule in Table 4. Figure 10 already indicates that the cost for reducing the bivariate time-variant service curve function to an univariate time-invariant service curve is an under-approximation of the offered service due to the inf-term in Equation 16.

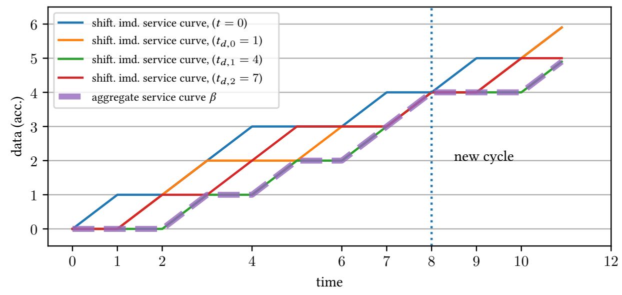


Figure 10: Direct, time-invariant service curve for the cyclic schedule from Table 4.

6 EMPIRICAL EVALUATION:

In this section, we investigate the difference between time-variant and time-invariant service curves with respect to the underestimation of offered service. We think this aspect is worth exploring, because both—time-variant and time-invariant Network Calculus—have their own merits and drawbacks. Since the derivation of the time-invariant service curves (cf. Equation 15-16) involves the combinatorial problem of finding extrema of the respective intermediate curves, we use a numerical approach to assess the impact of the schedule on the difference between the time-variant and time-invariant service curves for the time-triggered network elements.

Since service processes with ultimately constant-rate service are used the most often, we use the most simple service curve $\beta_S = t$ with constant-rate behavior for the service process S of the time-triggered network elements to figure out "how bad it is, to use time-invariant functions to model time-variant systems", focusing on the tightness of the worst-case bounds. Before continuing, we explain the performance metrics, which we use for the comparison of the service curve formulations.

Arrival-Curve-Oblivious Metrics. Since service curves can be interpreted as lower bounds on the actually offered service, the less under-approximation of offered service a service curve exhibits, the less worst-case bounds on backlog and delay are overestimated. Assume service curve β^a and service curve β^b are both supposed to model the same system. We consider service curve β^b more pessimistic than service curve β^a, if β^a exceeds β^b for some time, but never falls below β^b. More precisely, in the time-variant case, β^b is more pessimistic than β^a if $\forall s, t : \beta^b(s, t) \leq \beta^a(s, t)$, and $\exists s, t : \beta^b(s, t) < \beta^a(s, t)$.

The service curve can be regarded as means to describe the system, whereas the arrival curve is foremost a description of the arriving data traffic. The definition of the arrival curve is tied to a specific application and network scenario and is not within the scope of this paper. However, the shape of the arrival curve determines how much the different ways of modeling the service impact the computed delay and backlog bounds: Figure 11 depicts an example where $\text{delay}_b - \text{delay}_a = \Delta h$. In this scenario, the largest horizontal distance between the two service curves is the difference in the derived worst-case delay, and can be computed with the equations for the virtual delay (Equation 7,8), replacing the actual arrival curve α with the less pessimistic service curve β^a, and β with the more pessimistic service curve β^b, i.e., $\Delta h = \text{v.delay}(\beta^a, \beta^b)$. Analogously, we have $\text{backlog}_b - \text{backlog}_a = \Delta v$ in Figure 11, i.e., if we use β^b instead of β^a, we overestimate the worst-case backlog of the system by Δv. In this case, we compute the difference in the derived worst-case backlog with the backlog equation (Equation 6), replacing α with the less pessimistic service β^a curve and β with the more pessimistic service curve β^b, i.e., $\Delta v = \text{backlog}(\beta^a, \beta^b)$.

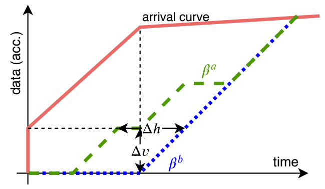


Figure 11: Δh and Δv are the difference in worst-case performance bounds for β^a and β^b.

Similarly, depending on the arrival curve there exist scenarios where the maximum delay and backlog are equal for both service curves even though service curve β^b is more pessimistic than service curve β^a. Using Δh and Δv, we can express the possible difference of the delay and backlog bounds of two service curves in absence of any specific arrival curve. Therefore, our evaluation is not subject to a particularly "good" or "bad" choice of the arrival curve or limited to a specific scenario.

6.1 Results

With the Δh and Δv metric, we compare the following two combinations of pairs of service curves that can be mapped to the two types of network elements :

time-triggered blocking: We compare the time-variant service curve from Equation 10 in Section 5.1 to the time-invariant leftover service curve from Equation 15 in Section 5.2.1.

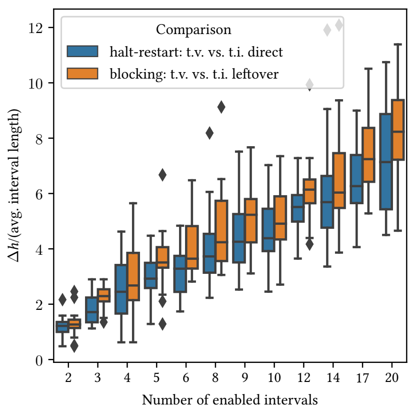
time-triggered halt-restart: We compare the time-variant service curve from Equation 11 in Section 5.1 to the time-invariant direct service curve from Equation 16 in Section 5.2.2.

Note, that Equation 16 can also be used for time-triggered network elements with blocking [31], but for brevity, we use the network element types to label the combinations throughout this section. To compare a time-variant service curve with a time-invariant service curve, we artificially extend each time-invariant service curve to a time-variant function $\beta_{\text{pseudo t.v.}}(s, t) = \beta_{\text{t.i}}(t - s)$. The evaluation of Δh and Δv is done as described previously with the time-variant versions of the equations for backlog bound (Equation 6) and virtual delay bound (Equation 8). Considering the derivation of the time-invariant service curve formulations for both types of network elements, the time-invariant service curves are either equally or more pessimistic than the time-variant service curves (cf. Figure 11: β^a equals the time-variant service curve, β^b equals the time-invariant service curve). We use our own prototype implementation of NC operations in Python for the evaluations in this section. For the computation of Δh and Δv, we apply a time discretization with 100 steps per time unit and evaluate the service curve functions with randomly created schedules with a granularity of one time unit. The service curves are evaluated over a window of length t_{cycle} since the offered service per cycle is equal for all service curves for our choice of $\beta_S = t$.

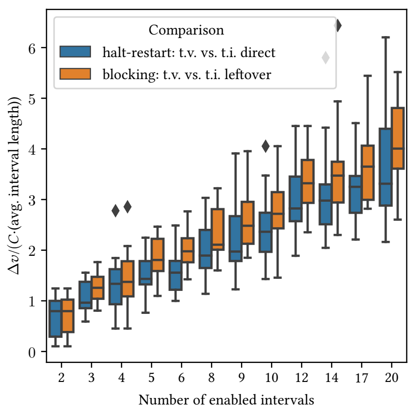
Schedules with Varying Number of Enabled Intervals of Random Length. The schedules for the evaluations presented in Figure 12a and 12b are created by randomly drawing the length of the enabled intervals and disabled intervals from the discrete range $[1, 100]$ with uniform probability. The cycle time is set to the sum of the lengths of the enabled intervals and disabled intervals. The number of enabled intervals ranges from 2 to 20, and we evaluate 20 unique schedules for each number of enabled intervals and each type of network element. Due to the random length of the individual enabled intervals, it is not meaningful to interpret the absolute values of Δh and Δv. Therefore, we normalize Δh to the average interval length per schedule. Analogously, we normalize Δv to the amount of data that can be served in the average schedule interval. Again, this normalization is applied per schedule. In both figures (Figure 12a, Figure 12b) we can observe that overall the difference (i.e., the normalized values of Δh and Δv) between the time-variant and time-invariant service curve increases with the number of entries of the cyclic schedule. For the evaluated schedules, Δh is on average bigger than the size of an average interval. The outliers in Figure 12a where Δh exceeds the average interval multiple times are from schedules where the enabled intervals are shorter than the difference of the lengths of the disabled intervals. Thus, the time-invariant service curves can possibly lead to overestimating the worst-case delay in the order of multiple interval lengths. The different modeling approaches have a larger impact on Δh than on Δv, because here all service curves have the same slope in the enabled intervals, which is determined by β_S.

Schedules with Varying Ratio of Enabled Intervals to Cycle Time. While the evaluations in the previous section used random schedules where only the number of schedule entries is fixed, we now fix the percentage $p_{e.d.}$ of the cycle time where the service process is enabled. To be exact, we create 20 unique schedules per $p_{e.d.} = \sum_i (t_{dr,i} - t_{er,i})/t_{\text{cycle}}$ with $p_{e.d.} \in \{0.1, 0.2, 0.4, 0.8\}$, each with $t_{\text{cycle}} = 400$ and 10 enabled intervals. The results are presented in Figure 12c and show that the mean of Δh reduces with increasing $p_{e.d.}$. This means, the less time the enabled intervals take up per cycle, the bigger the benefit of employing the time-variant service curve instead of the time-invariant service curve. The behavior shown in Figure 12c supports the observation from the evaluation in the with random interval length in the previous section that the time-invariant approaches perform particularly bad if the schedule consists of long disabled intervals with short interspersed enabled intervals (e.g., for $p_{e.d.} = 0.1$).

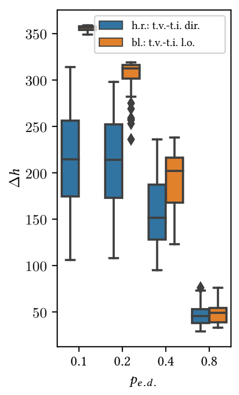
Schedules with Equal Mean Interval Length and Varying Variance of the Interval Lengths. To evaluate the effect of the variance of the enabled intervals, we compute Δh for schedules where all interval lengths are drawn from one of the two binomial distributions, $B_{HV}(N = 1000, p = 0.1)$, or $B_{LV}(N = 125, p = 0.8)$. The parameters are chosen such that the interval lengths have equal mean value $\mu_{HV} = \mu_{LV} = 100$, but different variance $\sigma^2_{HV} = 90$ and $\sigma^2_{LV} = 20$. Effectively, B_{HV} yields schedules with 4.5 times higher variance of the interval lengths compared to B_{LV}. Each schedule comprises 20 enabled intervals for each gate, and we evaluate 16 schedules per configuration. The results are shown in Figure 12d. The values of Δh are much higher for the data sets with higher variance of the interval lengths. Therefore, we conclude that not only the length of the schedule intervals, but also the variance of the schedule intervals is a major influence with respect to the difference of the service curves.



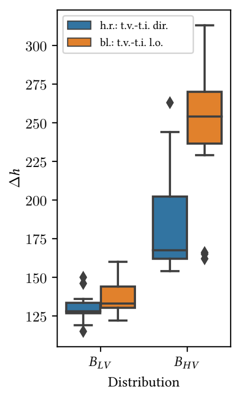
(a) Δh relative to average schedule interval length: different $i_{\max}$.



(b) Δv relative to data transmittable in the average schedule interval: different $i_{\max}$.



(c) Δh: different ratio of interval types per cycle.



(d) Δh: different variance of interval length.

Figure 12: Evaluation for random schedules with different creation scheme.

Summary of the evaluation. In all of our evaluations, the time-invariant service curve approach can lead to vast overestimations of the worst-case delay bounds and the worst-case backlog bounds. The more entries the cyclic schedule contains and the more the interval lengths vary, the stronger can the difference with regards to the worst-case bounds emerge in the analysis of a system. This can be attributed to the construction of the time-invariant service curves where a large variance of the schedule intervals can increase the under-approximation of offered service (cf. Figure 9 and Figure 10). This is intuitively clear, if we look at the extreme case where all schedule intervals have the same length. Then the variance of the lengths of the schedule intervals is zero, and the schedule can be reduced to a schedule with just one enabled interval and a cycle length of two enabled intervals. In this case, the difference between time-variant and time-invariant service curve is limited to one interval length, i.e., much smaller than the values observed in Figure 12. In the evaluations, the overall difference between time-variant service curve and the time-invariant service curve for the time-triggered elements with blocking is larger than the difference between time-variant service curve and the time-invariant service curve for the time-triggered elements with halt-restart. This is caused by the instantaneous arrivals of the data (cf. Equation 12) that emulate the disabled intervals for the actual arrival process A for the construction of the time-invariant leftover service curve. The instantaneous arrivals can result in service being attributed to the virtual arrival process A^v too early. Due to our particular choice of $\beta_S = t$, which is less affected by the cyclic schedule compared to other service curves (cf. rate-latency, where latency $\geq$ interval length), we expect these effects to be even more pronounced for most other service processes.

6.2 Extension To Multi-Hop?

"In theory", NC already provides the means to evaluate multi-hop scenarios, because we can compute the service curve of a series of network elements with $\beta_1 \otimes \beta_2 \otimes \ldots$ (β_x being the single-hop service curve of the x-th network element). However, in practice an analytic (symbolic) approach is required to evaluate expressions with multiple steps or operands, unless the expression can be computed by pointwise evaluation of the operands. This is due to the search for extrema (inf, sup) over large time-intervals as required for operations such as $\otimes$ which, if no analytic knowledge can be exploited, becomes computationally very expensive, effectively limiting a numerical approach to a single hop. While for time-invariant NC (restricted to piecewise-linear, ultimately pseudo-periodic functions), at least the algorithms are published [4], research yet has to provide similarly powerful algorithms for time-variant NC to make the evaluation of multi-hop scenarios feasible.

7 CONCLUSION

In this paper, we investigated service curves for network elements with service intermittence and resumption triggered at times derived from a cyclic schedule. In detail, we looked at time-triggered network elements with blocking, and time-triggered network elements with halt-restart behavior. Since there exists a tight coupling between the type of network element and the service curve formulation, we presented distinct time-variant formulations for each type of time-triggered network element and also showed how to obtain the respective time-invariant service curves. We showed that for a generic service process S, the computation of the actual value of the time-invariant service curve of the network element might involve evaluating an arbitrary number of expressions. In our numerical evaluations, we observed that the time-invariant service curves result in large over-estimations of the worst-case bounds compared to the time-variant service curve formulation and identified influencing factors.

Ultimately, our findings highlight some challenges for analyzing time-triggered network elements with intermittent service with NC: time-invariant service curves suffer from under-approximation when compared to the time-variant service curve, and, depending on β_S, might even be infeasible to evaluate computationally. Time-variant service curves offer tighter bounds, but their use, e.g., for large multi-hop scenario, is impeded by the lack of efficient algorithms for time-variant NC. Besides the development of these algorithms, the use of NC to jointly design shaping mechanisms and service schedules for real-time applications in networks with intermittent service is an interesting extension of this work.

ACKNOWLEDGMENTS

This work was supported by the German Research Foundation (DFG) under the research grant "Integrated Controller Design Methods and Communication Services for Networked Control Systems (NCS)" (RO 1086/20-1).

The authors also want to thank the anonymous reviewers of this paper for their valuable feedback.

REFERENCES

[1] 2016. IEEE Standard for Information Technology—Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks—Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (Dec. 2016), 1–3534. https://doi.org/10.1109/IEEESTD.2016.7786995

[2] 2018. IEEE Standard for Air Interface for Broadband Wireless Access Systems. *IEEE Std 802.16-2017 (Revision of IEEE Std 802.16-2012)* (March 2018), 1–2726. https://doi.org/10.1109/IEEESTD.2018.8303870

[3] 2018. IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks. *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)* (July 2018), 1–1993. https://doi.org/10.1109/IEEESTD.2018.8403927

[4] Anne Bouillard. 2014. *Algorithms and Efficiency of Network Calculus.* Habilitation à Diriger des Recherches. https://www.di.ens.fr/~bouillar/Publis/HDR.pdf

[5] Anne Bouillard, Laurent Jouhet, and Eric Thierry. 2009. *Service Curves in Network Calculus: Dos and Don'ts.* Technical Report RR-7094. INRIA. https://hal.inria.fr/inria-00431674/document

[6] Nico Becker and Markus Fidler. 2015. A Non-Stationary Service Curve Model for Performance Analysis of Transient Phases. *arXiv:1506.04657 [cs]* (June 2015). arXiv:cs/1506.04657

[7] Nico Becker and Markus Fidler. 2016. A Non-Stationary Service Curve Model for Estimation of Cellular Sleep Scheduling. *arXiv:1608.04024 [cs]* (Aug. 2016). arXiv:cs/1608.04024

[8] Steffen Bondorf and Jens B. Schmitt. 2014. The DiscoDNC - A Comprehensive Tool for Deterministic Network Calculus. In *Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS '14).*

[9] Jean-Yves Le Boudec. 1996. *Network Calculus Made Easy.* Technical Report. L'Ecole Polytechnique Fédérale de Lausanne (EPFL).

[10] Anne Bouillard, Laurent Jouhet, and Éric Thierry. 2010. Comparison of Different Classes of Service Curves in Network Calculus. *IFAC Proceedings Volumes* 43, 12 (Jan. 2010), 306–311. https://doi.org/10.3182/20100830-3-DE-4013.00051

[11] Marc Boyer, Hugo Daigmorte, Nicolas Navet, and Jörn Migge. 2016. Performance Impact of the Interactions between Time-Triggered and Rate-Constrained Transmissions in TTEthernet. In *8th European Congress on Embedded Real Time Software and Systems*. Toulouse, France.

[12] Cheng-Shang Chang and Rene L. Cruz. 1999. A Time Varying Filtering Theory for Constrained Traffic Regulation and Dynamic Service Guarantees. In *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '99)*, Vol. 3. 63–70. https://doi.org/10.1109/INFCOM.1999.749253

[13] Cheng-Shang Chang, Rene L. Cruz, Jean Yves Le Boudec, and Patrick Thiran. 2002. A Min, + System Theory for Constrained Traffic Regulation and Dynamic Service Guarantees. *IEEE/ACM Transactions on Networking* 10, 6 (Dec. 2002), 805–817. https://doi.org/10.1109/TNET.2002.804824

[14] Rene L. Cruz. 1991. A Calculus for Network Delay. Part I.+II. *IEEE Transactions on Information Theory* 37, 1 (Jan. 1991), 114–141.

[15] Dang Dinh Khanh and Ahlem Mifdaoui. 2014. Timing Analysis of TDMA-Based Networks Using Network Calculus and Integer Linear Programming. In *2014 IEEE 22nd International Symposium on Modelling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS '14).* IEEE, Paris, France, 21–30. https://doi.org/doi.ieeecomputersociety.org/10.1109/MASCOTS.2014.12

[16] Mábia Daniel-Cavalcante and Rafael Santos-Mendes. 2013. Modular Methodology for the Network Calculus in a Time-Varying Context. *IEEE Transactions on Information Theory* 59, 10 (Oct. 2013), 6342–6356. https://doi.org/10.1109/TIT.2013.2272870

[17] Joan Adrià Ruiz De Azua and Marc Boyer. 2014. Complete Modelling of AVB in Network Calculus Framework. In *Proceedings of the 22nd International Conference on Real-Time Networks and Systems (RTNS '14).* ACM, New York, NY, USA, 55:55–55:64. https://doi.org/10.1145/2659787.2659810

[18] Jonas Diemer, Daniel Thiele, and Rolf Ernst. 2012. Formal Worst-Case Timing Analysis of Ethernet Topologies with Strict-Priority and AVB Switching. In *7th IEEE International Symposium on Industrial Embedded Systems (SIES '12).* 1–10. https://doi.org/10.1109/SIES.2012.6356564

[19] Benazir Fateh and Manimaran Govindarasu. 2015. Joint Scheduling of Tasks and Messages for Energy Minimization in Interference-Aware Real-Time Sensor Networks. *IEEE Transactions on Mobile Computing* 14, 1 (Jan. 2015), 86–98. https://doi.org/10.1109/TMC.2013.81

[20] Markus Fidler. 2010. Survey of Deterministic and Stochastic Service Curve Models in the Network Calculus. *IEEE Communications Surveys Tutorials* 12, 1 (Feb. 2010), 59–86. https://doi.org/10.1109/SURV.2010.020110.00019

[21] Francois Baccelli, Guy Cohen, Geert Jan Olsder, and Jean-Pierre Quadrat. 2001. *Synchronization and Linearity An Algebra for Discrete Event Systems* (web ed.). https://www.rocq.inria.fr/metalau/cohen/documents/BCOQ-book.pdf

[22] Nicos Gollan and Jens B. Schmitt. 2007. Energy-Efficent TDMA Design Under Real-Time Constraints in Wireless Sensor Networks. In *Proceedings of the 2007 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '07).* IEEE Computer Society, Washington, DC, USA, 80–87. https://doi.org/10.1109/MASCOTS.2007.23

[23] Qing He, Di Yuan, and Anthony Ephremides. 2018. Optimal Link Scheduling for Age Minimization in Wireless Systems. *IEEE Transactions on Information Theory* 64, 7 (July 2018), 5381–5394. https://doi.org/10.1109/TIT.2017.2746751

[24] Jean-Yves Le Boudec. 1998. Application of Network Calculus to Guaranteed Service Networks. *IEEE Transactions on Information Theory* 44, 3 (May 1998), 1087–1096. https://doi.org/10.1109/18.669170

[25] Jean-Yves Le Boudec and Patrick Thiran (Eds.). 2001. *Network Calculus A Theory of Deterministic Queuing Systems for the Internet.* Number 2050 in Lecture Notes in Computer Science. Springer Berlin Heidelberg.

[26] Maria Rita Palattella, Nicola Accettura, Luigi Alfredo Grieco, Gennaro Boggia, Mischa Dohler, and Thomas Engel. 2013. On Optimal Scheduling in Duty-Cycled Industrial IoT Applications Using IEEE802.15.4e TSCH. *IEEE Sensors Journal* 13, 10 (Oct. 2013), 3655–3666. https://doi.org/10.1109/JSEN.2013.2266417

[27] RealTime-at-Work (RTaW). [n. d.]. RTaW-Pegase Helps Design Safe and Optimized Critical Embedded Networks. http://www.realtimeatwork.com/software/rtaw-pegase/

[28] Henrik Schioler, Hans P. Schwefel, and Martin B. Hansen. 2007. CyNC: A MATLAB/SimuLink Toolbox for Network Calculus. In *Proceedings of the 2nd International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS '07).* ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 60:1–60:10. http://dl.acm.org/citation.cfm?id=1345263.1345340

[29] Ernesto Wandeler and Lothar Thiele. 2006. Optimal TDMA Time Slot and Cycle Length Allocation for Hard Real-Time Systems. In *Asia and South Pacific Conference on Design Automation, 2006.* 479–484. https://doi.org/10.1109/ASPDAC.2006.1594731

[30] Ernesto Wandeler and Lothar Thiele. 2006. Real-Time Calculus (RTC) Toolbox. http://www.mpa.ethz.ch/Rtctoolbox. http://www.mpa.ethz.ch/Rtctoolbox

[31] Luxi Zhao, Paul Pop, and Silviu S. Craciunas. 2018. Worst-Case Latency Analysis for IEEE 802.1Qbv Time Sensitive Networks Using Network Calculus. *IEEE Access* 6 (2018), 41803–41815. https://doi.org/10.1109/ACCESS.2018.2858767

[32] Luxi Zhao, Paul Pop, Qiao Li, Junyan Chen, and Huagang Xiong. 2017. Timing Analysis of Rate-Constrained Traffic in TTEthernet Using Network Calculus. *Real-Time Systems* 53, 2 (March 2017), 254–287. https://doi.org/10.1007/s11241-016-9265-0

[33] Luxi Zhao, Paul Pop, Zhong Zheng, and Qiao Li. 2018. Timing Analysis of AVB Traffic in TSN Networks Using Network Calculus. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '18).* 25–36. https://doi.org/10.1109/RTAS.2018.00009