Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projekt 1 Bericht

Jonas Engler

Worst-Case-Timing unter AVB mit Network Calculus

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Jonas Engler

Worst-Case-Timing unter AVB mit Network Calculus

Projekt 1 Bericht eingereicht im Rahmen der PJ1 Prüfung

im Studiengang Master Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Korf

Eingereicht am: 2. April 2015

Inhaltsverzeichnis

1 Einleitung

Die Technik moderner Fahrzeuge beinhaltet verschiedene Infotainment-Bereiche und greift auf zunehmend viele Kontroll-Mechanismen zu. Durch die steigende Anzahl der Steuergeräte (ECUs) wächst die verwendete Bandbreite in den Netzwerken der Fahrzeuge kontinuierlich. Das treibt die alten Feldbus-Systeme (CAN, MOST, LIN, FlexRay) an ihre Grenzen, da diese keine ausreichend große Bandbreite anbieten können. Ohne neue Netzwerk-Techniken führen hinzugefügte Systeme zu einer stetigen Zunahme der Bussysteme, was positive sowie negative Folgen hat. Zum einen können isolierte Anwendungsbereiche übersichtlicher betrachtet werden (beispielsweise je ein Feldbus, der nur bestimmte Medien-Streams verarbeitet), zum anderen hat die erhöhte Anzahl einen größeren Verkabelungsaufwand und eine verschlechterte Wartbarkeit zur Folge, die sich auch aus der Notwendigkeit von feldbusübergreifenden Kommunikations-Gateways ergibt. Die ursprüngliche Entwicklung der Feldbusse hatte das Ziel, diese negative Seite der aktuellen Entwicklung – also erhöhten Verkabelungsaufwand – zu unterbinden. Dementsprechend beschreibt die aktuelle Entwicklung im Fahrzeug-Vernetzungs-Bereich einen ähnlichen Weg, mit dem Ziel, ein System für alle Übertragungen zu verwenden und damit einen gemeinsamen Kommunikations-Backbone für das gesamte Fahrzeug zu verwenden.

Um erhöhte Bandbreite sowie Kompatibilität zu vielen Endgeräten zu erreichen, ist Ethernet ein praktikabler Kandidat für die Nachfolge der aktuellen Feldbussysteme. Ethernet hat jedoch Nachteile bezüglich mangelhaftem Quality-of-Service (QoS). Für zwangsweise niedrige Latenzen, beispielsweise für sicherheitskritische Anwendungen, sind blockierende Pakete ein immenses Problem. Um die ursprünglich FIFO-gesteuerten Ausgangs-Queues der Switches auf prioritätsgesteuertes Forwarding zu erweitern, ist unter IEEE 802.1Q [1] eine Anpassung der Switches standardisiert. Eine zusätzliche Erweiterung existiert mit AVB [2]. AVB erweitert das priorisierte Scheduling um einen dynamischen Kredit für definierte AVB-Klassen. Obwohl die AVB-Klassen die höchsten Prioritäten des Netzwerkes erhalten, ermöglicht der Kredit ein faires Scheduling, da so auch niedrigere Prioritäten (Best-Effort-Pakete) die Möglichkeit zum Senden erhalten. Durch die Prioritäten können vergleichsweise kleine Latenzen garantiert werden und das Fairness-Prinzip ermöglicht ein Ankommen aller Pakete.

Zur Vermeidung von Sicherheitsrisiken, ausgelöst durch verzögerte Kommunikation, müssen Garantien bezüglich der Übertragungszeit von sicherheitsrelevanten Daten gegeben werden. Für diese Garantien muss ein Netzwerk analysiert werden. Die Analysen in dieser Arbeit beziehen sich auf die größte mögliche Verzögerung (worst case timing) einzelner Streams und zeigen damit die maximal möglichen Latenzen der Verbindungen von einzelnen End-Knoten zu jeweils anderen End-Knoten.

Um das worst case timing eines Netzwerk-Modelles zu erhalten, ist ein theoretisches Analyse-Verfahren notwendig. Eine extensive Simulation der Netzwerk-Abläufe ist dafür nicht ausreichend, da nicht zwangsläufig alle Konstellationen des Netzwerks durch eine Simulation abgedeckt werden. In dieser Arbeit wird daher das Analyse-Verfahren Network-Calculus [3] verwendet, das auf die Verwendung mit AVB angepasst wurde [4]. Ziel ist dabei die Berechnung der größtmöglichen Ende-zu-Ende-Latenz eines AVB-Netzwerks unter gezeigten Topologien und Konfigurationen. Die Theorie des Network-Calculus ist ein deterministisches Analyse-Verfahren, das variable Einflüsse, wie mögliche Paketverluste, Teilausfälle des Systems

und Redundanzkonzepte, nicht berücksichtigt. Unter Berücksichtigung aller Konfigurations-Einflüsse weist es dafür eindeutige Ergebnisse auf. Diese Arbeit wird den Analyse-Vorgang detailliert aufzeigen und auf Teilaspekte der Berechnung sowie offene Fragestellungen eingehen. Nachfolgend werden in Kapitel 2 die Grundlagen der Netzwerk-Techniken erklärt, in Kapitel 3 werden Aufbau und Vorgehensweise des Network-Calculus beleuchtet. Anschließend wird in Kapitel 4 das Network-Calculus-Modell für AVB vorgestellt und in Kapitel 5 eine Studie an mehreren Modellen mit Einordnung der Ergebnissen vorgenommen. Abgeschlossen wird die Arbeit mit einem Fazit und einem Ausblick.

2 Netzwerk-Grundlagen

Dieses Kapitel dient als Einstieg und zeigt die Entwicklung und die technischen Ansätze von AVB und geplante Entwicklungen des Standards.

2.1 AVB

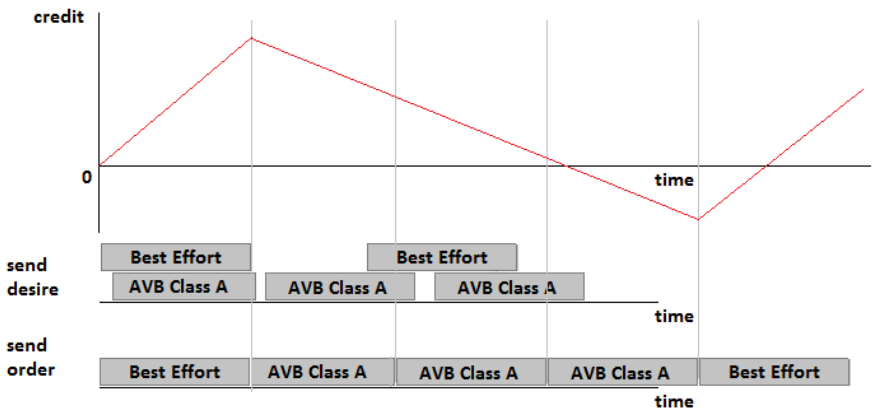


Abbildung 1: Kreditverlauf bei beispielhafter Paketsendungen unter AVB

Ursprünglich wurde AVB entwickelt, um die Übertragung von Audio- und Video-Signalen zu digitalisieren und über Ethernet laufen zu lassen. Da moderne Fahrzeuge audiovisuelle Daten für ihre Infotainment-Segmente übertragen müssen, fand AVB auch im Automotive-Bereich Beachtung. Da AVB – wie jeder andere Ethernet-Standard – mit QoS-Problemen haushalten muss, steht die Echtzeitfähigkeit des Standards im aktuellen Zustand zur Debatte. Diese Schwäche soll mit der Weiterentwicklung zu TSN beseitigt werden.

AVB steht für Audio/Video Bridging und setzt sich aus vier Standards [5–8] der IEEE zusammen. Diese ermöglichen die zeitlich synchronisierte, niedrig-latente Übertragung von priorisierten Streams in IEEE802-Netzen (vgl. [2]). Die Grundlage für Prioritäts-gesteuerte Ausgangsqueues an den Switches bildet der Standard 802.1Q [1], auf dem AVB mit einem

zusätzlichen Kredit-System aufbaut. Das Kredit-System wird für definierte AVB-Pakete, namentlich *AVB Class A* und *AVB Class B*, verwendet. Der Kredit nimmt bei Versendung einer dieser Klassen ab und steigt, wenn ein Paket der Klassen nicht senden kann bzw. warten muss. Class-A- und Class-B-Pakete können nur dann gesendet werden, wenn ihr Kredit größer als 0 ist. Ein beispielhafter Verlauf der Sendung von verschiedenen Paketen wird in Abb. 1 veranschaulicht. Durch den Kredit wird gewährleistet, dass auch niedrigpriore Pakete (beispielsweise Best Effort) versendet werden können.

Im Gegensatz zu TDMA-Verfahren weist AVB den Vorteil der dynamischen Bandbreiten-Reservierung auf. Es ist nicht nötig, alle Streams bei dem Entwurf des Netzwerks festzulegen, weil über AVB eine Reservierung bei laufendem Betrieb möglich ist. Da ein zusätzlicher TDMA-Ansatz für die nachfolgende Version des Standards geplant ist, wird diese zukünftige Entwicklung im nachfolgenden Abschnitt erläutert.

2.2 TSN

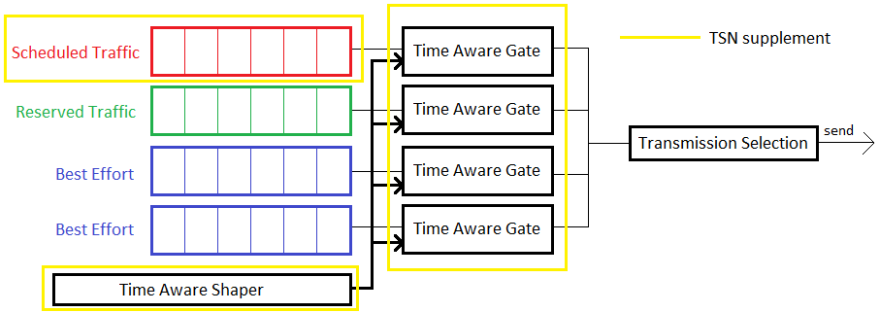


Abbildung 2: Anpassung der Ausgangsqueues unter TSN

Time **S**ensitive **N**etworking [9] beschreibt die nächste Generation von AVB. Die Arbeitsgruppe um TSN hat sich zum Ziel gesetzt, automotive- und industrial-needs zu erfüllen, indem die nächste AVB-Generation auch sicherheitsrelevante niedrige Latenzen aufweisen kann.

Unter Ethernet bilden verzögernde Pakete das größte Hindernis für niedrige Latenzen. Mit Blick auf die zukünftige Entwicklung von Ethernet umfasst TSN zwei Möglichkeiten der Verringerung der Latenz-Zeiten. Zum einen mit Präemption. Im aktuellen Zustand bietet Ethernet keine Präemption von Sendungen. Dieser Umstand soll sich in naher Zukunft, in Zusammenarbeit mit der Task Group IEEE 802.3 [10], ändern und für eine verkürzte Verzögerung durch niedrigpriore Pakete sorgen. Zum anderen mit Hilfe von TDMA. Time Division Multiple Access (TDMA) sorgt dafür, dass Pakete feste Zeitslots zugewiesen bekommen und sehr niedrige Latenzen aufweisen. Unter TSN wird der TDMA-Ansatz „Time-Aware-Shaper“ (TAS) genannt. In Abb. 2 werden die angepassten Queues unter TSN gezeigt. Der TAS schaltet sich über das Kredit-System von AVB und verhindert so die Versendung von Nicht-TDMA-Paketen in zuvor festgelegten Zeitslots. Für diese Arbeit werden Modelle der aktuellen AVB-Generation analysiert.

3 Network Calculus

Der Network Calculus „ist das Ergebnis kürzlicher Entwicklungen, die den tiefen Einblick in Flow-Probleme in Netzwerken liefern. [...] Mit dem Network Calculus sind wir in der Lage, fundamentale Eigenschaften von integrated services networks, window flow control, Scheduling und Buffer- oder Delay-Dimensionierungen zu verstehen“(vgl. [3]). Grundsätzlich untersucht der Network Calculus innerhalb von Komponenten, wie Ethernet-Switches, die Data-Flows (oder auch Streams) über die Zeit. Mit eingehenden und ausgehenden Flows können Rückschlüsse auf Verzögerungen und notwendige Queue-Größen innerhalb der Komponenten getroffen werden. Ob es sich bei den Flows um diskrete oder kontinuierliche Funktionen handelt, ist für die Rechnung nicht von Belang, da jede kontinuierliche Funktion $R(t)$ durch Sampling (mit Zeitslot-Größe δ) in eine diskrete Funktion $S(n), n \in \mathbb{N}$ eingesetzt werden kann:

$$S(n) = R(n\delta) \textbf{ (1)}$$

Die betrachteten Flows beschreiben den Datenverkehr in den Komponenten. Da ein Fluss (Flow) nur in eine Richtung fließt, sind die Network-Calculus-Funktionen zwingend monoton. In diesem Fall (nicht streng) monoton steigend, da die betrachteten Funktionen die bisher angekommenen bzw. schon weitergeleiteten Daten beschreiben. In Abb. 3 wird ein Beispiel für ein System S mit der Input-Funktion $R(t)$ und Output-Funktion $R^*(t)$ gezeigt.

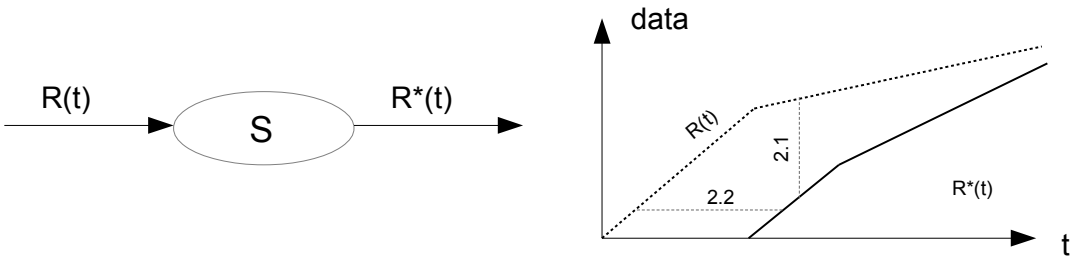


Abbildung 3: Network Calculus: Ein System S mit Input-Funktion $R(t)$, Output-Funktion $R^*(t)$ und dem zeitlichen Verlauf der gesendeten Daten dieser Flows

3.1 Flow-Eigenschaften

Aus $R(t)$ und $R^*(t)$ ergeben sich folgende Definitionen für Systemeigenschaften:

Der *backlog* zu Zeitpunkt t ist $R(t) - R^*(t)$ **(2.1)**
Der *virtual delay* zu Zeitpunkt t ist $d(t) = \inf\{\tau \geq 0 : R(t) \leq R^*(t+\tau)\}$ **(2.2)**

Der *backlog* ist die Menge an Daten (in Bits), die im System bzw. der Komponente gehalten werden. Wäre das System beispielsweise ein Puffer, entspräche der *backlog* der Anzahl verbleibender Elemente in der Queue. Der *backlog* entspricht der vertikalen Differenz beider Funktionen (s. Abb. 3: 2.1).

Als $virtual\ delay$ wird die Zeitspanne bezeichnet, um die ein Bit verzögert wird, wenn alle Bits, die vorher angekommen sind, zuerst gesendet werden. Der $virtual\ delay$ entspricht der horizontalen Differenz beider Funktionen (s. Abb. 3: 2.2). Durch den Operator inf erhält man das Infimum, also die größte untere Schranke, einer Menge. Mit den beiden Eigenschaften 2.1 und 2.2 lassen sich zwei zentrale Fragen beantworten:

1. Wie groß müssen die Queues innerhalb des Systems sein ($backlog$)?
2. Wie groß ist die maximale Verzögerung eines Pakets ($virtual\ delay$)?

Frage 1 ist für diese Arbeit uninteressant, Frage 2 zielt auf den Kern der Untersuchung von Worst-Case-Timings ab. Der $virtual\ delay$ ist für die Berechnung von Ende-zu-Ende-Latenzen wichtig.

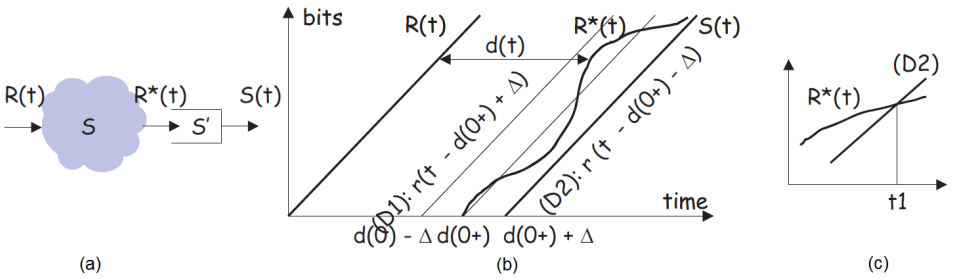


Abbildung 4: Network Calculus: Der Playout Buffer [3])

Die Output-Funktion eines Systems ist nicht zwingendermaßen linear und lässt sich deshalb nicht mit einer festen Rate beschreiben. Um für die Berechnung des $virtual\ delay$s eine linearisierte Output-Funktion bereitstellen zu können, wird in [3] der Playout Buffer (s. Abb. 4) vorgestellt. Er überführt $R(t)$ und $R^*(t)$ eines Systems S mithilfe von S' in die linearisierte Ausgangs-Rate $S(t)$ (s. Abb. 4(a)). $S(t)$ schätzt $R^*(t)$ so ab, dass $S(t)$ zu jedem Zeitpunkt t unter $R^*(t)$ liegt und dabei eine lineare Steigung hat. $S(t)$ entspricht der in Abb. 4(b) gezeigten Funktion $D2$.

Die beiden Funktionen $D1$ und $D2$ beschreiben die maximale bzw. minimale Rate, die das System durch $R^*(t)$ unterstützt (mit einer auf Δ begrenzten Delay-Schwankung). Nur durch die Verwendung der minimalen Rate $D2$ wird sichergestellt, dass der Puffer zu keinem Zeitpunkt t in $R^*(t)$ durch eine zu hohe Rate leer ist (was dem Fall von Zeitpunkt $t1$ in Abb. 4(c) entspräche). Wären leere Puffer vorhanden, könnte dieses System nicht mehr die ursprüngliche Output-Funktion $R^*(t)$ abbilden. In weiteren Berechnungen des $virtual\ delay$s werden linearisierte Output-Funktionen verwendet.

3.2 Arrival- und Service-Curve

Die Arrival- und die Service-Curve sind das Herzstück der Network-Calculus-Analyse. Sie begrenzen Ein- und Ausgangsverkehr einer Komponente mit Bezug auf den analysierten Flow.

Arrival Curve

Für Worst-Case-Berechnungen benötigt die Input-Funktion $R(t)$ eines Systems ein Limit aller ankommenden Daten. Folgende Definition gilt:

Für eine monoton steigende Funktion α, definiert für $t \geq 0$, ist ein Flow R dann und nur dann eingeschränkt von α wenn für alle $s \leq t : R(t) - R(s) \leq \alpha(t - s)$ **(3)**
In diesem Fall hat R α als eine Arrival Curve oder R ist α-$smooth$.

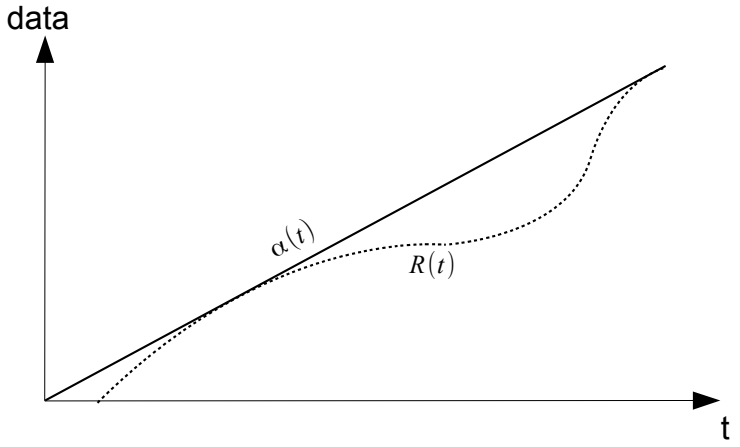


Abbildung 5: Network Calculus: Arrival Curve $\alpha(t)$ und Input-Funktion $R(t)$

Abb. 5 zeigt eine beispielhafte Arrival Curve. $R(t)$ ist niemals oberhalb von $\alpha(t)$, deswegen ist $\alpha(t)$ eine Arrival Curve für $R(t)$.

Für ein switched network ist die erweiterte Definition der „affine arrival curve“ hilfreich. Eine affine Arrival Curve ist eine affine Funktion und wird folgendermaßen definiert:

$$\gamma_{r,b} = \begin{cases} rt + b & \text{wenn } t \geq 0 \\ 0, & \text{sonst} \end{cases} \text{ für } r \geq 0 \text{ (die Rate) und } b \geq 0 \text{ (Der Burst)} \textbf{ (4)}$$

Ist beispielsweise $\alpha(t) = rt$ (wobei r die Kapazität der Input-Ports ist), wird die beschränkende Arrival Curve durch eine Rate r beschrieben. In einem switched network ist dies durch die maximale Kapazität eines Links gegeben, da niemals mehr Bits im System ankommen können, als die Input-Ports mit ihrer Bandbreite verarbeiten können.

Service Curve

Als Gegenstück zur Begrenzung des Inputs gibt es Garantien für den Output. Diese drücken aus, was das System dem Flow an Output-Bandbreite garantieren kann und finden ihre Entsprechung im Network Calculus in der Service Curve. Per Definition:

Man nehme ein System S mit einem Flow s durch S mit den Input- bzw. Output-Funktionen R und R^*. Wir definieren, dass β für den Flow s eine Service Curve ist, dann und nur genau dann, wenn β monoton steigend, $\beta(0) = 0$ und $R^* \geq R \otimes \beta$ ist **(5)**

Hat ein Flow mit Output-Funktion $R^*(t)$ eine Service Curve β, ist zu jedem Zeitpunkt t der Übertragung garantiert, dass $R^*(t) \geq \beta(t)$ (s. Abb. 6).

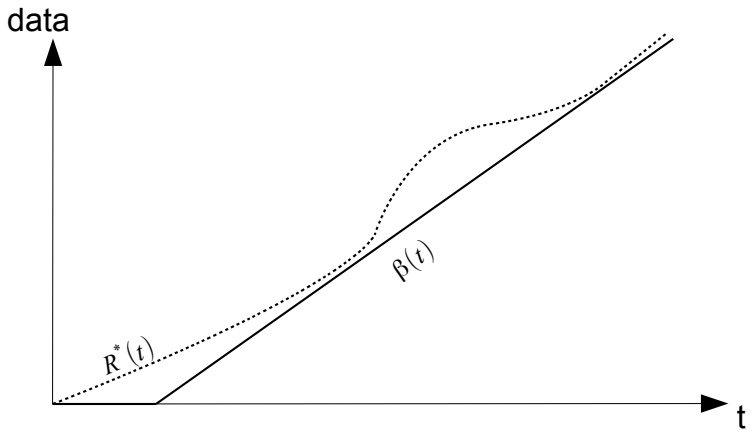


Abbildung 6: Network Calculus: Service Curve $\beta(t)$ und Output-Funktion $R^*(t)$

Die Operation $\otimes$ ist Bestandteil der Min-Plus-Algebra und bewirkt eine Min-Plus-Faltung, die im Anschluss (Abschnitt 3.3) erläutert wird. Die Gleichung $R^* \geq R \otimes \beta$ kann in konventioneller Algebra als $R^*(t) \geq \inf[R(s) + r(t-s)], 0 \leq s \leq t$ geschrieben werden, wobei r für die Rate der Service Curve steht.

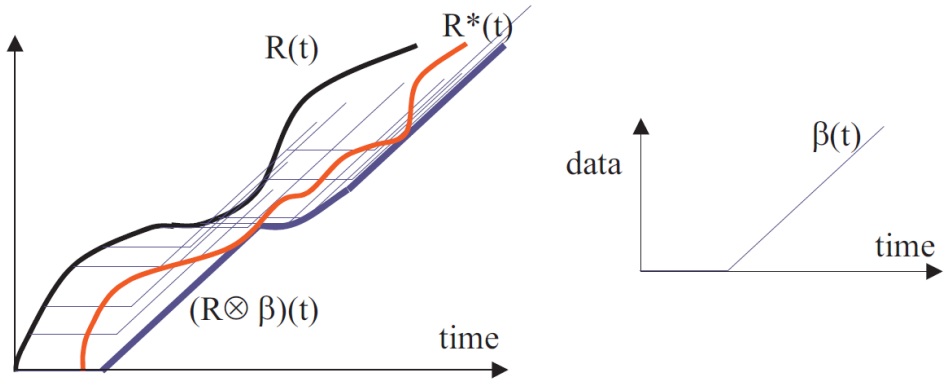


Abbildung 7: Network Calculus: Service Curve [3])

Die untere Begrenzung der Output-Funktion wird durch die in Abb. 7 gezeigte Funktion $(R \otimes \beta)(t)$ gebildet. Sie bildet die Minimal-Funktion von $R(t)$ und $\beta(t)$.

Ein Beispiel für eine Service Curve in einem switched network ist der verwaltete Traffic eines Traffic Shapers. Erlaubt der Traffic Shaper einem Flow die Benutzung eines Teils der Bandbreite auf dem Output-Port, so entspricht die Service Curve der Rate dieses Bandbreiten-Teils. Ein Paket des Flows darf auf jeden Fall senden und das mindestens mit der Rate, die der Traffic Shaper garantiert hat. Wenn kein anderes Paket senden will, wird der Flow noch schneller als mit der garantierten Rate gesendet. Damit liegt die garantierte Bandbreite des Traffic Shapers garantiert unter oder auf der Rate des Flows und erfüllt die Definition 5.

AVB arbeitet mit dem Credit Based Shaper, der den $idleSlope$ der Klassen regelt. Der $idleSlope$ reguliert die Kredite der Klassen und legt somit Bandbreiten für die Klassen fest. Dieser $idleSlope$ gibt Garantien für verwendbare Bandbreiten und entspricht ebenfalls der Definition 5.

3.3 Min-Plus-Algebra

In der Min-Plus-Algebra [11] gelten andere Definitionen für binäre Operatoren als in der konventionellen arithmetischen Algebra. Für die Min-Plus-Operatoren $\oplus$ und $\otimes$ gelten die in Tabelle 1 angegebenen Definitionen der konventionellen Algebra. Die Operation $3 + 5 = 8$ der konventionellen Algebra ist unter Min-Plus mit $3 \oplus 5 = 3$ definiert. Tabelle 1 zeigt nur die für

Konventionelle Algebra	Min-Plus-Algebra
$min\{a, b\}$	$a \oplus b$
$a + b$	$a \otimes b$

Tabelle 1: Min-Plus-Algebra-Definitionen

die Ausführung des Network Calculus notwendigen Operatoren. Die Änderung der binären Operatoren formt den Teilaspekt $(\mathbb{R}, +, \times)$ der konventionellen Algebra in $(\mathbb{R} \cup \{+\infty\}, \wedge, +)$ um. $\wedge$ wird als binärer Operator des Minimums zweier Parameter definiert, $\vee$ als Maximum.

Min-Plus-Faltung

Die in der Funktionsanalyse verwendete Faltung (auch Konvolution genannt) zweier Funktionen (f und g) liefert in konventioneller Algebra ein Produkt ($f * g$) und ist definiert durch:

$$(f * g)(t) := \int_0^t f(t-s)g(s)ds \quad \text{für } f(t<0) = g(t<0) = 0 \textbf{ (6)}$$

Die resultierende Funktion beschreibt den gewichteten Mittelwert der Funktion g mit der Gewichtung f abhängig von s. Die Faltung in der Min-Plus-Algebra ist definiert durch:

$$(f \otimes g)(t) := \inf\{f(t-s) + g(s)\}, 0 \leq s \leq t \textbf{ (7)}$$

Die Min-Plus-Faltung zweier Funktionen ergibt für jeden Zeitpunkt t das Minimum beider Funktionen zum Zeitpunkt t. Diese Faltung ist für den Network Calculus besonders in Bezug

auf die Service Curve wichtig, da diese immer die minimal garantierte Ausgangs-Bandbreite eines Flows darstellt.

Der Operator $\otimes$ wurde bereits in Tabelle 1 definiert und drückt aus, dass statt einer Multiplikation eine Addition der Funktionen stattfindet. Das Integral der konventionellen Faltung entspricht in der Min-Plus-Faltung dem Infimum der Menge dieses Zeitintervalls, da Additionen zu Minimumsfindungen (oder Infimumsfindungen, wenn kein Minimum existiert) werden.

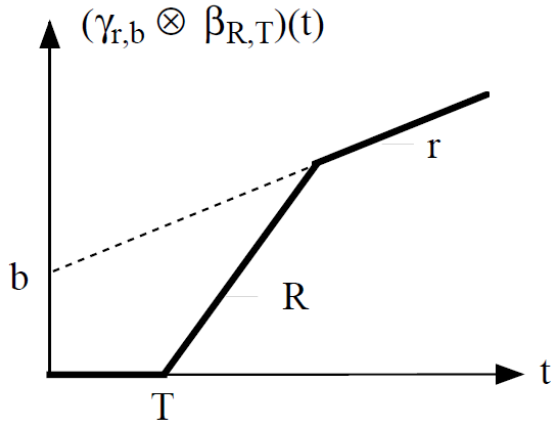


Abbildung 8: Network Calculus: Faltung in der Min-Plus-Algebra [3])

Für ein Beispiel der Faltung in der Min-Plus-Algebra werden die beiden Funktionen $\gamma_{r,b}$ und $\beta_{R,T}$ betrachtet. Das Ergebnis der Faltung ist in Abb. 8 veranschaulicht. Die Funktion $R[s-T]^+$ ist die Rate-Latency-Funktion und steht für $\beta_{R,T}$. T beschreibt den maximalen Delay. Die Rate-Latency-Funktion übernimmt diesen Delay, indem erst ab Zeitpunkt T Daten übertragen werden. $\gamma_{r,b}$ wurde bereits in Abschnitt 3.2, Formel 4, definiert. Zuerst die Faltung für $0 \leq t \leq T$:

$$
\begin{aligned}
&(\gamma_{r,b} \otimes \beta_{R,T})(t) \\
&= \inf_{0 \leq s \leq t} \{\gamma_{r,b}(t-s) + \beta_{R,T}(s)\} \\
&= \inf_{0 \leq s \leq t} \{\gamma_{r,b}(t-s) + R[s-T]^+\} \\
&= \inf_{0 \leq s \leq t} \{\gamma_{r,b}(t-s) + 0\} = \gamma_{r,b}(0) + 0 = 0 + 0 = 0 \quad \textbf{(8.1)}
\end{aligned}
$$

Die Funktionen sind beide für die Bereiche $t \leq 0$ mit dem Wert 0 definiert. Das Infimum für $\gamma_{r,b}(t-s)$ ist damit 0. $R[s-T]^+$ ist für alle Bereiche $t \leq T$ ebenfalls mit 0 definiert und ergibt für den Zeitraum $0 \leq t \leq T$ den Wert 0. Nachfolgend die Berechnung für den Zeitraum $t > T$:

$$
\begin{aligned}
&(\gamma_{r,b} \otimes \beta_{R,T})(t) \\
&= \inf_{0 \le s \le t} \{\gamma_{r,b}(t-s) + \beta_{R,T}(s)\} \\
&= \inf_{0 \le s \le T} \{\gamma_{r,b}(t-s) + R[s-T]^+\} \wedge \inf_{T \le s < t} \{\gamma_{r,b}(t-s) + R[s-T]^+\} \wedge \inf_{s=t} \{\gamma_{r,b}(t-s) + R[s-T]^+\} \\
&= \inf_{0 \le s \le T} \{b + r(t-s) + 0\} \wedge \inf_{T < s < t} \{b + r(t-s) + R[s-T]\} \wedge \{0 + R[t-T]\} \\
&= \{b + r(t-T)\} \wedge \{b + rt - RT + \inf_{T \le s \le t} \{(R-r)s\}\} \wedge \{R[t-T]\} \\
&= \{b + r(t-T)\} \wedge \{b + r(t-T)\} \wedge \{R[t-T]\} \\
&= \{b + r(t-T)\} \wedge \{R[t-T]\} \quad \textbf{(8.2)}
\end{aligned}
$$

Die Faltung sorgt dafür, dass die entstandene Funktion an allen Punkten einen Wert aufweist, der dem Infimum (Minimum) der gefalteten Funktionen entspricht. Im weiteren Verlauf dieser Arbeit wird, außer bei expliziter Gegenaussage, mit konventioneller Algebra gerechnet.

4 NC-Modell für Switched Networks

Georges et al. [12] haben in ihrer Arbeit eine Adaption des Network Calculus auf switched networks vorgenommen. Die dabei entstandene Formel zur Untersuchung des Delays eines Pakets innerhalb eines Switches (s. Formel 13) wurde durch Rene Queck in [4] so parametrisiert, dass die Untersuchung von AVB-Netzwerken möglich ist. Die Bestandteile der Delay-Formel werden hier nachfolgend erklärt, veranschaulicht und in Abschnitt 5.1 an dem Fahrzeug-Modell aus [4] und einem weiteren Beispiel-Modell (5.2) angewendet.

4.1 Arrival- und Service-Curve

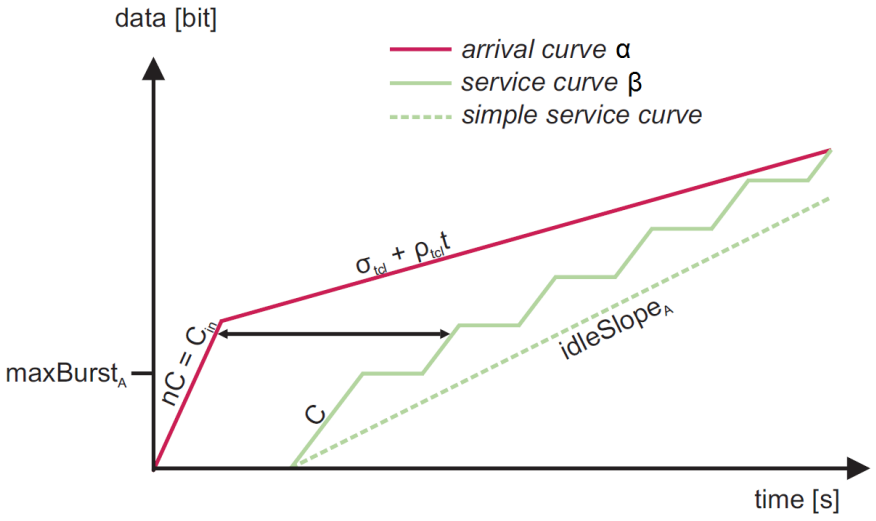


Abbildung 9: Network Calculus: Arrival- und Service-Curve (urspr. aus [4])

Abb. 9 zeigt Arrival- und Service-Curve (die linearisierte Form wird für die Rechnungen verwendet) für ein switched network. Die Form der Kurven bleibt für die Anwendung unter AVB grundsätzlich erhalten. Für den $virtual\ delay$ eines Systems S – für ein switched network ist S ein Switch – muss die horizontale Differenz von Arrival- und Service-Curve betrachtet werden. Die Arrival-Curve des Systems ist folgendermaßen beschrieben:

$$\alpha(t) = min\{Ct, \sigma + \rho t\} \textbf{ (9)}$$

C steht für die Kapazität eines Links, σ für einen initialen Burst und ρ für die Bandbreite des Streams. Die Arrival-Curve (s. Abb. 9) hat anfangs durch den Burst einen starken Anstieg. Durch die Begrenzung der Kapazitäten entspricht die Arrival-Curve bis zu dem Zeitpunkt der vollständigen Verarbeitung des Bursts der gesamten Kapazität C des Links. Ab dem Zeitpunkt der vollständigen Verarbeitung des Bursts beschreibt die Arrival-Curve die Rate der Stream-Bandbreite ρ. Für die Service-Curve des Systems gilt:

$$\beta_{tcl} = R_{tcl}(t - T_{tcl})^{+} \textbf{ (10)}$$

Der Index tcl beschreibt unterschiedliche Traffic-Klassen. Diese Indexierung erlaubt unter AVB unterschiedliche Service-Curves für AVB Klasse A, Klasse B und Best-Effort-Verkehr. Rene Queck wählte in [4] den $idleSlope$ der AVB-Klassen als Service Curve. Wie in den Beispielen der Service Curve in 3.2 bereits gezeigt, kann die Bandbreitenregulierung eines Traffic Shapers eine Service Curve sein. Der $idleSlope$ unter AVB entspricht der Regulierung durch einen Shaper und ist eine gute Modellierung der Service Curve.

4.2 Delay innerhalb eines AVB-Switches

Der Delay eines Systems ist von verschiedenen Parametern abhängig. Für ein switched network, auch für AVB, gibt es folgende Parameter:

- T_{tcl}[s] — Der maximale zu erwartende Delay einer Traffic-Class tcl
- σ_{tcl}[bits] — Der initiale Burst vor der Versendung des Streams
- ρ_{tcl}[bit/s] — Die Bandbreite des betrachteten Streams
- C_{in}[bit/s] — Die Summe der Bandbreiten der Input-Ports
- R_{tcl}[bit/s] — Entspricht der Service Curve ($idleSlope$)
- τ_{tcl}[s] — Die Zeitspanne des initialen Bursts

Diese Parameter wurden in [12] in den $virtual\ delay$ eines Systems folgendermaßen eingesetzt:

$$d_i(t) = \inf_{\Delta \geq 0} \left\{ min\{C_{in}t, \sigma_i + p_i t\} = R(t + \Delta - T)^{+} \right\} \textbf{ (11)}$$

Der maximale Delay und damit der Worst-Case-Delay innerhalb eines Switches wird mit $\overline{D_i} = max_{t \geq 0} d_i(t)$ angegeben. Analog zu der in Abschnitt 3.3 gezeigten Faltung wird $\overline{D_i}$ umgestellt in ($\vee$ ist definiert als das Maximum zweier Werte):

$$\overline{D_i} = d(0) \vee \underset{0<t<\tau_i}{max}\, d(t) \vee \underset{t\geq\tau_i}{max} d(t) \textbf{ (12)}$$

Durch Umformungen mit Berücksichtigung der Eigenschaften der o.g. Parameter ergibt sich für dieses System die folgende Formel für den maximalen Delay eines Flows innerhalb des Systems:

$$\overline{D_{tcl}} = (T_{tcl} - \tau_{tcl}) + \frac{\sigma_{tcl}+\rho_{tcl}\tau_{tcl}}{R_{tcl}}$$
$$\tau_{tcl} = \frac{\sigma_{tcl}}{C_{in}-\rho_{tcl}} \textbf{ (13)}$$

Um die Ende-zu-Ende-Latenz eines Flows auf seinem Weg über alle Switches zu erhalten, werden die Delays dieser Switche aufsummiert. Nicht berücksichtigt in dieser Formel ist die Zeitverzögerung durch die Übertragung des Pakets. Konsequenterweise muss die vollständige Formel zur Berechnung des Ende-zu-Ende-Delays $D_{total,max}$ bei einer Anzahl von n Switches also lauten:

$$D_{total,max} = (\sum_{k=0}^{n} \overline{D_{i_k}}) + (n+1)\frac{\eta}{C} \textbf{ (14)}$$

η steht für die Paketgröße des Flows. Formel 14 liefert den Worst-Case-Wert der gesamten Ende-zu-Ende-Latenz. Der Zusatz $(n+1)\frac{\eta}{C}$ beschreibt die Übertragungszeit des Pakets über die Anzahl der Links.

4.3 Initialer Delay

Der initiale Delay T_{tcl} des Flows wird mit einer Zeitspanne angegeben. Diese setzt sich aus den Verzögerungen durch andere Pakete oder Traffic-Shaper-Beeinflussungen zusammen. In [4] wird der initiale Delay eines Flows abhängig von der Traffic-Class (in nachfolgender Formel als i) des Pakets angegeben:

$$T_i = \delta_{i,hold} + \delta_{i,burst}$$
$$\delta_{i,hold} = \frac{max_{0<j<i}\{L_{j,max}\}}{C}$$
$$\delta_{i,burst} = \frac{\sum_{k=i+1}^{i_{max}} \sigma_k + min\{\sigma_A, \overline{\sigma}_{i,A}\} + min\{\sigma_B, \overline{\sigma}_{i,B}\}}{R_i} \textbf{ (15)}$$

Der Wert von $\delta_{i,hold}$ entspricht der Zeitspanne, die das größte Paket aller anderen Traffic-Klassen zur Versendung benötigt. Nach dem Zeitraum des größten Pakets anderer Klassen ist es möglich, dass ein Burst höher priorisierter Nachrichten die Versendung des Flows weiter verzögert. Diese Zeitspanne wird mit $\delta_{i,burst}$ abgebildet.

Für ein switched Network mit Prioritäten sind $\delta_{i,hold}$ und $\delta_{i,burst}$ zutreffend. Für AVB muss unter $\delta_{i,hold}$ zusätzlich der Aspekt des Traffic-Shapings miteinbezogen werden. Der Credit Based Shaper sorgt dafür, dass ein Paket nur dann senden kann, wenn der Kredit der zugehörigen Klasse einen Wert über 0 aufweist. Ist der Wert des Kredits minimal ($loCredit$), muss das Paket im schlimmsten Fall die Steigerung des Kredits auf einen Wert ≥ 0 abwarten und bei dem Kreditwert von -1 einem Paket den Vortritt lassen, das die Zeitspanne von $\delta_{i,hold}$ in Anspruch nimmt.

Um die Zeitspanne zu modellieren, die ein Paket warten muss, bis sein Kredit wieder den Wert 0 erreicht, wird zunächst der kleinstmögliche Kreditwert ($loCredit$) benötigt (C=Linkkapazität):

$$loCredit_{tcl} = maxFrameSize_{tcl} \cdot \frac{sendSlope_{tcl}}{C} \textbf{ (16.1)}$$
$$sendSlope_{tcl} = (idleSlope_{tcl} - C)$$
$$idleSlope_{tcl} = \textit{Die für Traffic-Class tcl reservierte Bandbreite}$$

Als nächstes folgt die Zeitspanne die $loCredit$ für das Erreichen des Wertes 0 benötigt:

$$Credit_{low \to 0} = \frac{-loCredit}{idleSlope} \textbf{ (16.2)}$$

Da ein Erreichen von 0 eine Versendung des Pakets erlauben würde, wird die Zeitspanne bis zum Erreichen des Wertes -1 verwendet. Unter Berücksichtigung dieser Zeitspanne wird als Anpassung von Formel 15 die folgende Version angeboten:

$$\delta_{i,hold,neu} = \frac{max_{0<j<i}\{L_{j,max}\}}{C} + \frac{-(loCredit_i+1)}{idleSlope_i} \textbf{ (17)}$$

Mit T_i aus Formel 15 lässt sich nun der Traffic-Klassen-spezifische initiale Delay für AVB angeben. Anschließend werden die in [4] angegebenen Definitionen (alt) von T_A und T_B um die in Formel 17 vorgeschlagenen Anpassungen erweitert (neu):

$$T_{A,alt} = \frac{max\{L_{B,max}, L_{max}\}}{C} \textbf{ (18.1)}$$
$$T_{A,neu} = \frac{max\{L_{B,max}, L_{max}\}}{C} + \frac{-(loCredit_A-1)}{idleSlope_A} \textbf{ (18.2)}$$

Ein AVB-Klasse-A-Paket muss somit nicht nur auf ein Paket anderer Klassen mit maximaler Größe warten, sondern auch auf das Ansteigen des Kreditwertes von $loCredit$ auf -1.

Für T_B ergeben sich die folgenden Varianten:

$$T_{B,alt} = \frac{L_{max}}{-sendSlope_A} + \frac{L_{A,max}}{C} \textbf{ (19.1)}$$
$$T_{B,neu} = \frac{L_{max}}{-sendSlope_A} + \frac{L_{A,max}}{C} + \frac{-(loCredit_B-1)}{idleSlope_B} \textbf{ (19.2)}$$

Für AVB-Klasse-B-Pakete gibt es somit ein ähnliches $\delta_{i,hold}$ wie für Klasse A. Der Wert von $\delta_{i,burst}$ ist für Klasse A aufgrund fehlender höherer Prioritäten unwichtig, für Klasse B enthält dieser die maximale Zeitspanne, die „burstende“ Pakete der Klasse A verursachen können.

In nachfolgenden Berechnungen (inbesondere in Kapitel 5) ist die Unterscheidung von $T_{i,alt}$ und $T_{i,neu}$ wichtig, daher wird diese Kennzeichnung weiterhin verwendet.

4.4 Burst

Der initiale Burst σ_{tcl} vor der Versendung eines Streams unterscheidet sich je nach Konfiguration des Netzwerks und seiner Streams. Dem Burst zugehörig sind Pakete der gleichen Klasse, im Fall von Klasse-A-Paketen ebenfalls Klasse-A-Pakete. Die im Burst enthaltenen Pakete kommen zeitlich gesehen vor dem analysierten Stream in der Ausgangs-Queue des Switches an, daher müssen sie zeitlich vor dem analysierten Stream versendet werden. Der Burst setzt sich im Worst-Case-Szenario aus allen anderen Streams zusammen, die an der selben Ausgangs-Queue wie der analysierte Stream senden wollen (s. Formel 20).

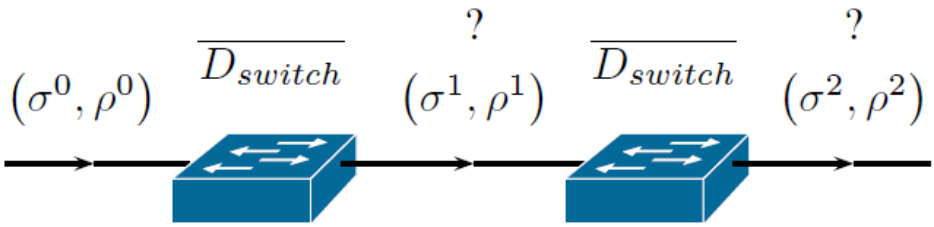


Abbildung 10: Network Calculus: Burst-Entwicklung im Netzwerk [12]

$$\sigma_{tcl} = \sum_{t=0}^{m} \rho_t \textbf{ (20)}$$

Der Burst eines Streams auf dem Weg durch das Netzwerk weist den in Formel 20 angegebenen Wert nur im ersten Switch auf, im nachfolgenden Switch wird dieser errechnete Wert mit den Delay-Werten der vorherigen Switches konkateniert. Das führt zu dem in Abb. 10 gezeigten Verhalten, das die Berechnung des Bursts in in- und out-Komponenten aufteilt.

$$\sigma_{out} = \sigma_{in} + \rho_{in}\overline{D}, \quad \rho_{out} = \rho_{in} \textbf{ (21)}$$

Für die in Abb. 10 gezeigte Verbindung zwischen den beiden Switches gilt nach Formel 21:

$$(\sigma^1, \rho^1) = (\sigma^0 + \rho^0\overline{D_{tcl}}, \rho^0) \textbf{ (22)}$$

Bei der Berechnung des Ende-zu-Ende-Delays eines Streams im Netzwerk steigt der Burst-Wert für jeden weiteren Switch auf dem Weg des Streams. Der Anstieg resultiert aus dem zusätzlichen Burst, der am nächsten Switch anliegt. Dieser besteht aus den Bits, die in der Zeitspanne $\overline{D_{tcl}}$ von dem analysierten Stream versendet werden könnten. Rene Queck bietet in seiner Arbeit eine Adaption der Formel auf die limitierte Burst-Fähigkeit eines Streams unter AVB an ($\overline{\sigma}$ ist der maximale Burst-Wert):

$$\overline{\sigma}_{i,A} = \frac{R_A \cdot max_{j \neq i}\{L_{B,max}, L_{j,max}\}}{-sendSlope_A} + L_{A,max} \textbf{ (23.1)}$$
$$\overline{\sigma}_{i,B} = C \cdot \frac{T_B \cdot R_B - loCredit_B}{-sendSlope_B} \textbf{ (23.2)}$$

$$\sigma_{A,out'} = \sigma_{A,in} + \rho_{A,in}\overline{D_{tcl}}, \quad \rho_{A,out} = \rho_{A,in},$$
$$\sigma_{A,out} = min\{\overline{\sigma}_{i,A}, \sigma_{A,out'}\} \textbf{ (24)}$$

Unter AVB sind ausgehende Bursts durch den CBS limitiert. Die maximalen Werte sind durch Formel 23.1 bzw. 23.2 definiert. Der ausgehende Burst kann diesen Maximalwert nie überschreiten. In der Sektion 5.3 wird auf die Burst-Fortpflanzung im Zusammenhang mit AVB und den Ergebnissen aus [4] eingegangen.

4.5 Werte der Parameter

Die Parameter des Delays setzen sich aus statischen und dynamischen Komponenten zusammen, die in Tabelle 2 gezeigt werden. Die Werte der dynamischen Komponenten ändern sich je nach Aufbau des Netzwerks und den verwendeten Streams.

Parameter	Typ – Zusammensetzung	Wert
$T_{tcl,neu}$	dynamisch – Maximal-Frame + AVB-Verzögerung	s. Formel 18.2
σ_{tcl}	dynamisch – AVB-Frames in höherer Queue-Position	dynamisch
ρ_{tcl}	dynamisch – Verwendete Bandbreite des Streams	dynamisch
C_{in}	statisch – Summe der Bandbreiten der Input-Ports	nC
R_{tcl}	dynamisch – Service Curve	$idleSlope_{tcl}$

Tabelle 2: Delay-Parameter unter AVB

4.6 Veranschaulichung

Um die einzelnen Bestandteile der Delay-Formel einordnen zu können, zeigt dieser Abschnitt, wie sich die einzelnen Parameter auf den Delay auswirken. In Abb. 11 wird gezeigt, wie sich der Delay eines Streams aus den Bestandteilen der Formel 13 ($\overline{D_{tcl}}$) zusammensetzt.

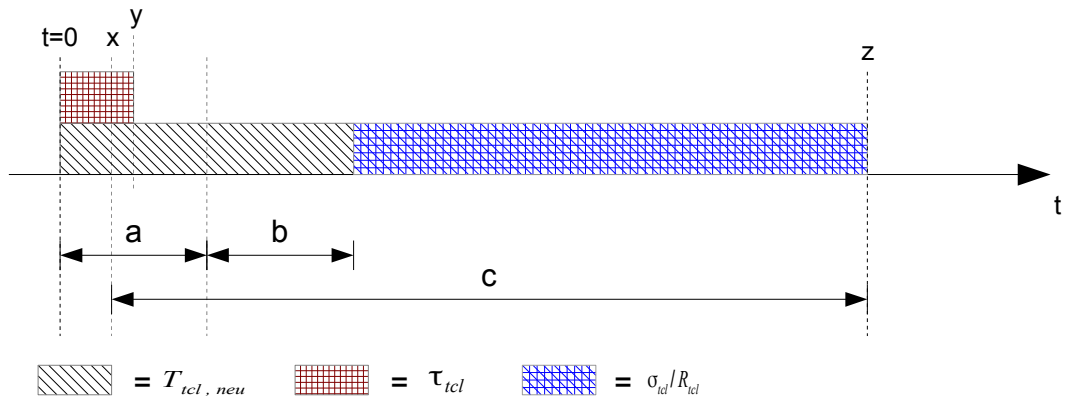


Abbildung 11: Komponenten des Delays

Zum Verständnis der einzelnen Zeitabschnitte werden an dieser Stelle die markierten Abschnitte mit den Bestandteilen der Delay-Formel (Formel 13) in Verbindung gebracht. Der zu analysierende Stream wird nachfolgend als γ bezeichnet. Der Zeitpunkt $t = 0$ beschreibt den Moment, in dem der Zeitraum des initialen Delays ($T_{tcl,neu}$) beginnt. Der Abschnitt, der den Zeitkonsum von $T_{tcl,neu}$, beschreibt, ist in mehrere Sub-Segmente unterteilt. Die Zeitabschnitte a und b beschreiben die Zeiträume, die γ auf das Ansteigen des Kredits auf einen Wert ≥ 0 (a) bzw. auf einen Maximal-Ethernet-Frame warten muss (b).

Das Segment a fällt zeitlich gesehen teilweise mit dem Zeitraum von τ_{tcl} zusammen. Der Zeitraum τ_{tcl} beschreibt die Zeit, die alle Ports des Switches brauchen, um den Burst σ auf die Ausgangs-Queue zu legen. Der Start dieses Zeitraums stimmt mit dem Start von $T_{tcl,neu}$ überein, da der Wert des Kredits zum Zeitpunkt $t = 0$ den niedrigsten Wert $loCredit$ aufweist.

Um diesen niedrigen Wert zu erreichen, muss vorher eine AVB-Nachricht versendet worden sein, die den Kredit des End-Knotens, der sie verschickt hat, so stark verringert hat, dass die Sendung eines weiteren Pakets (das zum Burst gehören wird) erst wieder zum Zeitpunkt $t = 0$ möglich ist.

Im Zeitraum τ_{tcl} sind alle Ports mit der Weiterleitung der Burst-Pakete beschäftigt, daher kann kein Paket auf der Ausgangs-Queue ankommen und somit keine Sendung eines Pakets zur Abnahme des Kredits führen, also steigt der Kredit in diesem Zeitraum. Alle Burst-Pakete müssen vor γ an der Ausgangs-Queue anliegen, aus diesem Grund muss das Ankommen von γ in den Zeitraum τ_{tcl} fallen. Für die Berechnung des Delays muss deshalb von Segment a der Zeitraum τ_{tcl} abgezogen werden, da γ erst an dessen Ende eintreffen kann. Da in Teilen des Zeitraums τ_{tcl} Ressourcen für die Weiterleitung von γ verwendet werden müssen (Zeitraum $y-x$), wird nicht der gesamte Zeitraum τ_{tcl} von Segment a abgezogen, sondern nur $x-(t=0)$. Von $T_{tcl,neu}$ werden demnach nur Segment b und Segment a abzüglich x in den Delay mit eingerechnet.

Der Abschnitt $\frac{\sigma_{tcl}}{R_{tcl}}$ beschreibt die Zeit, die der Burst σ braucht, um auf der Ausgangs-Queue versendet zu werden. Die Pakete des Bursts sind vor γ in der Ausgangs-Queue und müssen alle vor γ versendet werden. γ muss auf alle Pakete des Bursts warten.

Der gesamte Delay innerhalb eines Switches setzt sich aus dem verbliebenen Zeitraum $T_{tcl,neu}-x$ und der Forwarding-Zeit des Bursts auf der Ausgangs-Queue, $\frac{\sigma_{tcl}}{R_{tcl}}$, zusammen. Der Gesamt-Delay entspricht dem Zeitraum c. Nachfolgend wird Formel 13 umgestellt, um die einzelnen Abschnitte genauer zu beleuchten:

$$\begin{aligned}\overline{D_{tcl}} &= (T_{tcl} - \tau_{tcl}) + \tfrac{\sigma_{tcl}+\rho_{tcl}\tau_{tcl}}{R_{tcl}} \\ &= T_{tcl} - \tfrac{\tau_{tcl}R_{tcl}}{R_{tcl}} + \tfrac{\sigma_{tcl}+\rho_{tcl}\tau_{tcl}}{R_{tcl}} \\ &= T_{tcl} - \tfrac{\tau_{tcl}R_{tcl}}{R_{tcl}} + \tfrac{\sigma_{tcl}}{R_{tcl}} + \tfrac{\rho_{tcl}\tau_{tcl}}{R_{tcl}} \\ &= T_{tcl} + \tfrac{\sigma_{tcl}}{R_{tcl}} + \tfrac{\tau_{tcl}(\rho_{tcl}-R_{tcl})}{R_{tcl}} \quad \textbf{(25)}\end{aligned}$$

Die ersten beiden Summanden aus Formel 25 beschreiben den Zeitraum $z-0$ aus Abb. 11. Der Zeitraum $x-(t=0)$ entspricht in Formel 25 dem Ausdruck $\frac{\tau_{tcl}(\rho_{tcl}-R_{tcl})}{R_{tcl}}$ und muss von z abgezogen werden, da dieser Ausdruck immer negativ ist. Der gesamte Delay entspricht Segment $c = z - x$.

5 Analyse

In diesem Kapitel werden die obenstehenden Formeln auf Automotive-Modelle angewendet. Das von Rene Queck in [4] vorgestellte Automotive-Modell wird nachvollzogen und analysiert und die Ergebnisse mit der Analyse dieser Arbeit verglichen. Anschließend wird ein Minimal-Modell betrachtet, um die Ergebnisse einordnen zu können.

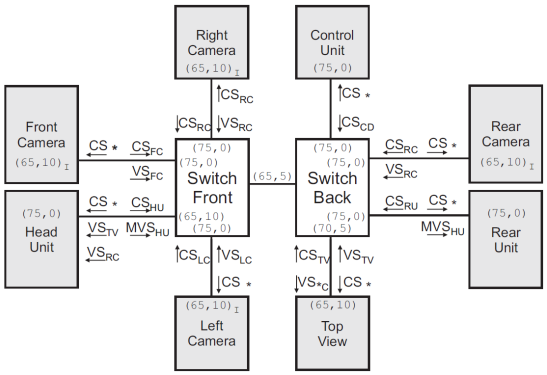
5.1 Beispiel-Modell 1

Rene Queck beschreibt in seiner Arbeit „Analysis of Ethernet AVB for Automotive Networks using Network Calculus“ [4] eine Automotive-Topologie, die auf die Latenzen ihrer einzelnen Streams untersucht wird. Dieses Modell wird zum Vergleich auch für diese Arbeit analysiert. Verwendet werden die drei verschiedenen Nachrichtentypen

1. Control Signals (CS) – AVB-Class A,
2. Video Signals (VS) – AVB-Class B und
3. MVideo Signals (MS) – Best Effort.

Diese unterscheiden sich in ihren Paketgrößen und dem daraus resultierenden Burst (s. dazu Abb. 12(b)). Der Aufbau des Netzwerkes und die Eigenschaften der Streams werden für die Analyse per Network Calculus verwendet und liefern Worst-Case-Latenzen für die einzelnen Streams.

Netzwerk-Konfiguration



(a) Netzwerk-Konfiguration aus [4]

Signal	ρ [Mbit/s]	σ [byte]	Frame [byte][a]	P
Control Signal (CS)	0.512	640	64	3-A
Video Signal (VS)	16.97	70012	1522	2-B
MVideo Signal (MS)	15.70	65446	1522	7-1

(b) Stream-Eigenschaften aus [4]

Abbildung 12: AVB-Aufbau aus [4]

In Abb. 12 werden Topologie und Konfiguration des Netzwerks gezeigt. Es besteht aus acht Endknoten, die über zwei Switches miteinander verbunden werden. Eine Nachricht läuft über maximal drei Hops. In der Analyse werden die Streams der CS- und VS-Signale betrachtet. In Abb. 12(b) werden einige Eigenschaften der Streams genannt, für die Berechnung der Ende-zu-Ende-Latenzen werden die fehlenden Parameter in Tabelle 3 ergänzt.

Der Parameter T_A ist Klasse-A-spezifisch berechnet und C_{in} aus der Topologie begründet. Der Parameter σ_A ist durch Abb. 12(b) vorgegeben und bedeutet mit seiner Größe, dass der Burst aus zehn Nachrichten der gleichen Klasse besteht. Insgesamt sind in der Konfiguration acht

Parameter	Wert	Erklärung
$T_{A,alt}$	$121.76\mu s$	s. Formel 18.1
$T_{A,neu}$	$207.372\mu s$	s. Formel 18.2
σ_A	640 Byte	s. Abb. 12 (b)
ρ_A	0.512 Mbit/s	s. Abb. 12 (b)
C_{in}	400 Mbit/s	s. Tabelle 2
R_A	5.632 Mbit/s	s. Tabelle 2

Tabelle 3: Parameter für CS-Pakete

verschiedene Nachrichten des Typs CS vorhanden, nur sieben davon würden sinnvollerweise an einen End-Knoten zugestellt werden und einer davon ist der analysierte Stream. Der Burst-Wert müsste also eine Größe von 384 Bytes aufweisen. Der Parameter ρ_A beschreibt die verwendete Bandbreite des analysierten Streams. Unter AVB erhält jedes Klasse-A-Paket einen Sendenzeitraum von 125 μs, innerhalb dieser Zeit wird das Paket versendet. Das bedeutet, dass jedes Klasse-A-Paket 8000 mal in der Sekunde verschickt wird. Berücksichtigt man die Größe des Frames, kann die benötigte Bandbreite folgendermaßen berechnet werden:

$$\frac{64Bytes}{0.000125s} = \frac{512Bits}{0.000125s} = \frac{4096000Bits}{s} = 4.096Mbit/s$$

Der Burst-Wert und die Bandbreite des Streams aus Tabelle 3 sind somit unklar, werden aber in den Berechnungen dieses Modells verwendet.

Der Wert von R_A ist durch den idleSlope dieser Klasse festgelegt. Ausgehend von der Annahme, dass der Burst aus zehn Paketen besteht, muss Bandbreite für zehn (+ 1 für den analysierten Stream) Pakete reserviert werden. Der Wert von R_A entspricht der elffachen Bandbreite eines CS-Pakets.

Parameter	Wert	Erklärung
$T_{B,alt}$	$189.437\mu s$	s. Formel 19.1
$T_{B,neu}$	$426.399\mu s$	s. Formel 19.2
σ_B	70012 Byte	s. Abb. 12 (b)
ρ_B	16.97 Mbit/s	s. Abb. 12 (b)
C_{in}	400 Mbit/s	s. Tabelle 2
R_B	33.94 Mbit/s	s. Tabelle 2

Tabelle 4: Parameter für VS-Pakete

Für VS-Pakete werden in Tabelle 4 die Werte der Parameter angegeben. Die Bandbreite ρ_B hat Auswirkungen auf R_B, der bei zwei Paketen dem obigen Wert entspricht. C_{in} ist gegenüber den Werten für Klasse-A-Pakete unverändert. Der Burst der VS-Pakete ist mit dem 46-fachen Wert eines Pakets beziffert. Die Zusammensetzung dieses Werts ist unklar, wird aber in den folgenden Berechnungen verwendet.

Network-Calculus-Ergebnisse

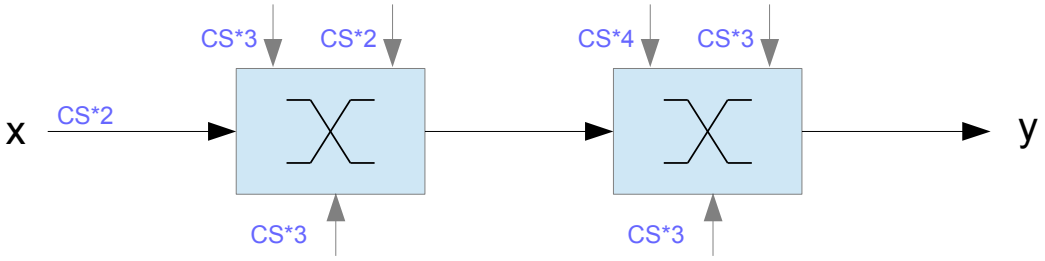


Abbildung 13: Analysierter CS-Stream im Netzwerk. Die Ende-zu-Ende-Latenz des CS-Streams ist die benötigte Zeit zwischen x und y. Alle eingehenden Ports senden den Burst des Switches zu möglichst gleichen Teilen

In diesem Abschnitt werden die Ergebnisse der Network-Calculus-Berechnungen dargestellt und analysiert. Der Weg des analysierten CS-Streams entspricht dem Weg zwischen x und y in Abb. 13. VS-Pakete nehmen den selben Weg, werden aber durch andere VS-Pakete (und CS-Pakete) blockiert.

Für CS-Pakete gilt:

1. Gehören zu AVB-Klasse-A
2. $\rho_A = 0.512 Mbit/s$
3. Route über 3 Hops
4. Frame-Größe: 64 Byte
5. VL-Bandbreiten (C): $100 Mbit/s$
6. Burst pro Switch: 640 Byte

Für VS-Pakete gilt:

1. Gehören zu AVB-Klasse-B
2. $\rho_B = 16.97 Mbit/s$
3. Route über 3 Hops
4. Frame-Größe: 1522 Byte
5. VL-Bandbreiten: $100 Mbit/s$
6. Burst pro Switch: 70012 Byte

Die (aufgerundeten) Werte für den maximalen ausgehenden Burst (relevant für die „Fortpflanzung“ von Burst) in dieser Konfiguration lauten:

$\overline{\sigma}_{i,A} = 155$ Byte
$\overline{\sigma}_{i,B} = 4261$ Byte

Zum Vergleich der Werte zeigt Tabelle 5 die Worst-Case-Ende-zu-Ende-Latenzen dieser Konfiguration aus [4].

CS-Pakete $[ms]$	VS-Pakete $[ms]$
4.25	5.67

Tabelle 5: Queck [4] Ergebnisse der CS- und VS-Pakete

In der Berechnung gibt es für den Parameter T_{tcl} die Alternativen $T_{tcl,neu}$ (Warten auf $loCredit$) und $T_{tcl,alt}$ (nur ein Maximal-Frame). Zusätzlich wird die Fortpflanzung des Bursts (σ-Evolution – s. Abschnitt 4.4) entweder gar nicht, in der Variante „analysierter Stream“ (nur der analysierte Stream wird für die Fortpflanzung des Bursts berücksichtigt) oder der Variante „alle Streams“ (alle Streams der gleichen Klasse werden berücksichtigt) in die Berechnung miteinbezogen. Formel 21 definiert nicht eindeutig, welche der beiden Varianten angewendet werden muss. Zusätzlich ergibt sich für AVB die Fragestellung, ob eine Fortpflanzung des Bursts möglich ist (s. dazu Abschnitt 5.3). Das $\sigma_{t,out}$-Maximum mit Auswirkung auf die Burst-Fortpflanzung wird probeweise teilweise nicht berücksichtigt.

#	T_{tcl}	σ-Evolution	$\sigma_{t,out}$-Maximum	CS-Pakete $[ms]$	VS-Pakete $[ms]$
1	neu	keine	–	2.226	32.76
2	neu	analysierter Stream	nein	2.324	40.50
3	neu	analysierter Stream	ja	2.253	32.88
4	**neu**	**alle Streams**	**nein**	**3.317**	**48.24**
5	neu	alle Streams	ja	2.253	32.88
6	alt	keine	–	2.054	32.29
7	alt	analysierter Stream	nein	2.145	39.91
8	*alt*	*analysierter Stream*	*ja*	*2.081*	*32.41*
9	alt	alle Streams	nein	3.060	47.54
10	*alt*	*alle Streams*	*ja*	*2.081*	*32.41*

Tabelle 6: Network-Calculus-Ergebnisse

Tabelle 6 zeigt die Ergebnisse für die höchstmöglichen Ende-zu-Ende-Latenzen aller CS- sowie VS-Pakete. Die hervorgehobene Zeile 4 der Ergebnisse weist die höchsten Latenz-Werte auf.

Die in Zeile 4 gewählte Berechnung bewirkt durch den sich fortpflanzenden Burst und eine fehlende Regulierung (kein maximaler Ausgangs-Burst) eine Aufsummierung aller bisherigen Delays ($\sum_{i=1}^{n} \overline{D_{tcl}} \cdot i$, n = Anzahl der Switches) für weitere Switches. Wenn die Route mehr als zwei Switches umfassen würde, geriete der Unterschied noch größer. Wenn der Ausgangs-Burst begrenzt wird, ist eine solche Aufsummierung nicht möglich. Da das Ausgangs-Burst-Maximum bereits den Ausgangs-Burst des ersten Switches begrenzt, scheint es bei dieser initialen Burst-Größe irrelevant, ob die Burst-Fortpflanzung nur den analysierten Stream oder alle Streams der Klasse umfasst. Diese Auswirkung ist in den Zeilen 3 und 5 oder 8 und 10 zu erkennen.

Um den Vergleich mit den Ergebnissen aus Tabelle 5 anzustellen, ist die Berechnung so auszuwählen, dass sie der Vorgehensweise aus [4] entspricht. Dafür werden $T_{tcl,alt}$, eine aktive Burst-Fortpflanzung sowie ein aktives Ausgangs-Burst-Maximum verwendet (vgl. Zeile 8 und 10 aus Tabelle 6). Der Vergleich dieser Werte mit den Ergebnissen aus Tabelle 5 zeigt:

- Die CS-Pakete weisen eine deutlich geringere Worst-Case-Latenz auf
- Die VS-Pakete weisen eine deutlich höhere Worst-Case-Latenz auf

Auch andere Werte aus Tabelle 6 liefern keine Ergebnisse, die eine zu vernachlässigende Differenz zu den Werten aus Tabelle 5 aufweisen.

Eine mögliche Erklärung dieser Diskrepanzen lässt sich in der Konfiguration des Netzwerkes finden. Die gegebenen Parameter für die Berechnungen (s. Abb. 12(b)) sind eindeutig. Allerdings könnte R_{tcl} einen anderen Wert aufweisen, was die Ergebnisse der Berechnungen entscheidend beeinflussen würde. Die Änderung

$$R_A = 2.56 Mbit/s \text{ (fünffache Bandbreite des analysierten Streams)}$$

beispielsweise würde den Worst-Case-Latenz-Wert der Zeilen 8 und 10 in Tabelle 6 auf $4.279 ms$ ändern. Abzüglich der Übertragungszeit, die ein Paket durch seine Größe beansprucht, kommt dies dem Ergebnis aus Tabelle 5 nahe. Für VS-Pakete hat sogar die Ansetzung des $idleSlopes$ auf das Maximum (= $75 Mbit/s$) nicht zur Folge, dass die Übertragung im zeitlichen Rahmen des Wertes aus Tabelle 5 liegt. Der große Burst-Wert lässt keine kleineren Delays zu.

Für CS-Pakete lässt sich mit Anpassung der Parameter ein vergleichbares Ergebnis erzielen, für VS-Pakete nicht. Im folgenden Abschnitt wird ein weiteres Netzwerk betrachtet, analysiert und ein Vergleich zu diesem Modell gezogen.

5.2 Beispiel-Modell 2

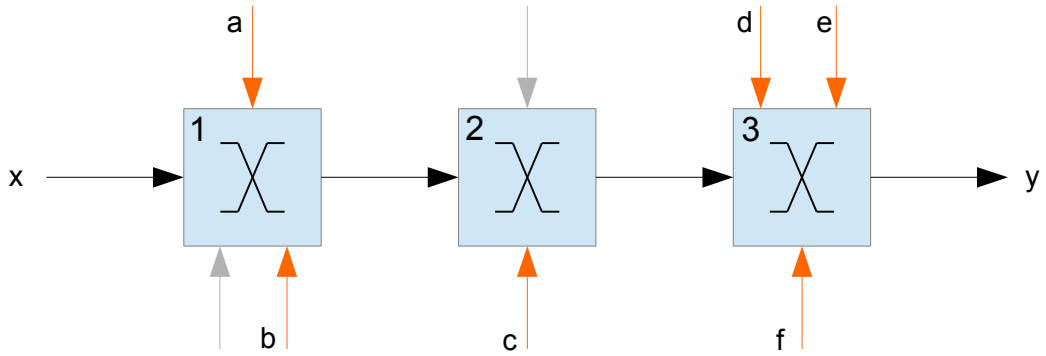


Abbildung 14: Der analysierte Stream nimmt die Route zwischen x und y. Links mit störendem Verkehr sind orange markiert. Graue Links leiten nur Best-Effort-Pakete weiter.

Diese Sektion zeigt neben dem in Abschnitt 5.1 vorgestellten Modell ein weiteres Fahrzeug-Modell, das mithilfe des auf AVB angepassten Network Calculus untersucht wird. Es wird ein Stream der AVB-Klasse-A mit den folgenden Eigenschaften untersucht:

1. $\rho_A = 14.08 Mbit/s$
2. Route über 4 Hops
3. Frame-Größe: 220 Byte
4. VL-Bandbreiten (C): $100 Mbit/s$

Der Weg des analysierten Streams wird in Abb. 14 zwischen x und y gezeigt. Die in Abb. 14 gezeigten Pakete sind AVB-Klasse-A-Pakete mit den folgenden Eigenschaften:

Bez.	Größe [Byte]	ρ_A [Mbit/s]
a	120	7.68
b	140	8.96
c	100	6.4
d	360	23.04
e	240	15.36
f	120	7.68

Tabelle 7: Eigenschaften der anderen AVB-Pakete im Netzwerk

Andere Pakete des Netzwerkes sind ausschließlich Best-Effort-Pakete mit einer maximalen Größe von 800 Byte. Für den Stream ergeben sich in den drei Switches unterschiedliche

Parameter für D_{tcl}, die in Tabelle 8 beleuchtet werden. $T_{tcl,neu}$ weist einen kleinen Wert auf, da kein Best-Effort-Paket mit maximaler Größe (1522 Byte) versendet wird.

Switch #	$T_{tcl,neu}$ [μs]	σ_A [Byte]	ρ_A [Mbit/s]	C_{in} [Mbit/s]	R_A [Mbit/s]
1	103.66	260	14.08	400	30.72
2	132.29	100	14.08	300	20.48
3	83.06	720	14.08	400	60.16

Tabelle 8: Delay-Parameter des Streams innerhalb der Switches

Im Gegensatz zu dem in Abschnitt 5.1 vorgestellten Modell sind alle Eigenschaften des Traffics bekannt. Die Berechnungen werden mit folgender Konfiguration (vgl. Tabelle 6) ausgeführt:

1. $T_{tcl} = T_{tcl,neu}$
2. σ-Evolution: Nur der analysierte Stream
3. $\sigma_{t,out}$-Maximum: ja

Bereich	$\overline{D_A}$ [μs]	σ_{out} [Byte]	σ_{out}-Max [Byte]
Switch 1	168.45	296	574
Switch 2	284.42	425	425
Switch 3	217.17	–	–
Gesamt	740.44	–	–

Tabelle 9: Ergebnisse der Analyse

Die Ergebnisse für den Delay der einzelnen Switches und die gesamte Ende-zu-Ende-Latenz werden in Tabelle 9 gezeigt. Es zeigt sich in dem Modell, dass nicht hoher Burst für große Verzögerungen sorgt, sondern ein niedriger $idleSlope$. Trotz des kleinsten Burst-Werts weist Switch 2 den größten Delay auf. Durch niedrige $idleSlope$s wird die Sendezeit des Bursts erhöht. Die Fortpflanzung des Bursts wird im zweiten Switch durch das Ausgangs-Burst-Maximum verringert, ohne Limitierung würde der Burst einen höheren Wert aufweisen. Die Limitierung des Ausgangs-Bursts verhindert einen kontinuierlichen Anstieg des Bursts.
Für die Route zwischen x und y (s. Abb. 14) benötigt der Stream 740.44μs. Im Gegensatz zu Beispiel-Modell 1 liegt dieser Wert in dem Bereich der Latenz-Garantie der AVB-Standards (maximal 2ms über 7 Hops).

Im nächsten Abschnitt werden Fragestellungen behandelt, die sich im Laufe dieser Arbeit ergeben haben.

5.3 Offene Fragestellungen

Aus der Betrachtung des Network Calculus und dessen Erweiterung für switched networks ergeben sich einige Fragen, die in dieser Sektion genauer beleuchtet werden.

Burst auf Ingress-Ports

Fragestellung: „**Warum startet der Burst-Eingang τ_{tcl} erst zu Zeitpunkt t=0?**“

Die Delay-Formel (13) enthält den Parameter τ_{tcl}. Dieser drückt eine Zeitspanne zu Beginn der Übertragung aller Frames aus. Sie definiert die Zeit, die der gesamte Burst des Switches benötigt, um durch die Ingress-Ports auf die Ausgangs-Queue gelegt zu werden (s. Abb. 11).

Laut Formel 13 muss der Zeitabschnitt τ_{tcl} vom gesamten Delay abgezogen werden, was bedeutet, dass der Burst-Eingang frühestens zu Zeitpunkt t=0 anfängt und während seiner Übertragung andere AVB-Versendungen verhindert. Unklar ist, warum dieser Zeitabschnitt nicht bereits zu einem Zeitpunkt t<0 beginnt.

Vorläufige Erklärung:
Ein Blick auf die Eigenschaften des Netzwerkes zu den Zeitpunkten $t < 0$ gibt eine Hilfestellung. Zu Zeitpunkt t=0 entspricht der Kredit dem Wert $loCredit$. Das bedeutet, dass bis zu Zeitpunkt t=0 ein AVB-Frame (der größte AVB-Frame des Systems) übertragen worden ist. Dieser Frame muss Bestandteil des Bursts sein, da alle AVB-Frames (bis auf den analysierten) zum Burst gehören. Einer der Endknoten, der für Burst-Pakete sorgt, muss für diesen Frame verantwortlich sein. Durch die Größe des Frames ist der Kredit dieses Endknotens niedrig, was eine sofortige Versendung eines neuen Frames verhindert.

Problem der Erklärung: Sie ist zu ungenau. Diese Annahme rechtfertigt nicht, dass τ_{tcl} niemals vor t=0 beginnt. Weiterhin lässt sich keine anschauliche exakte Erklärung für ein solches Verhalten von τ_{tcl} finden.

Burst-Fortpflanzung

Fragestellung: „**Muss die Burst-Fortpflanzung bei AVB angewendet werden?**“

Die in Abschnitt 4.4 eingeführte Fortpflanzung des Bursts (σ-Evolution) ist ein Bestandteil des Switched-Network-NC-Modells. Durch diese Fortpflanzung werden aufgetretene Bursts eines Switches auch im nächsten Switch (teilweise) berücksichtigt. Unklar ist, ob sich der Ausgangs-Burst aus dem analysierten Stream oder allen Paketen der zugehörigen Traffic-Class zusammensetzt. Unter AVB sorgt der Credit Based Shaper (CBS) dafür, dass Bursts sich nicht unendlich „aufstauen“ können. Nach Formel 24 wächst der Ausgangs-Burst proportional zum Delay. Diese Ansichtsweise impliziert, dass ein Stream (mindestens teilweise) auf sich selber wartet, was im Gesamtkontext unlogisch erscheint. Nicht zu vernachlässigen ist das Ausgangs-Burst-Maximum, dass das Verhalten des CBS in die Burst-Fortpflanzung limitierend einbringt.

Vorläufige Erklärung:
Durch den Ausgangs-Port des Switches werden aufgrund des Delays viele Daten versendet. Ein Teil dieser Daten könnte auch im nächsten Switch noch auf die Versendung warten, wenn der analysierte Stream an der Ausgangs-Queue des nächsten Switches anliegt. Die Burst-Fortpflanzung kann ein Versuch sein, dieses Verhalten abzubilden.

Problem der Erklärung: Auch diese Lösungs-Hypothese lässt sich mithilfe der Formel nicht nachvollziehen, daher bleibt dies nur eine mögliche Erklärung.

6 Fazit und Ausblick

Diese Arbeit ist ausgerichtet auf das Worst-Case-Timing von Ende-zu-Ende-Latenzen in AVB-Netzwerken. Für eindeutige und nachvollziehbare Aussagen bezüglich dieser Latenzen wird ein theoretisches Analyse-Verfahren benötigt. Verwendet wird innerhalb dieser Arbeit der Network-Calculus, der auf die Analyse von Flows, beispielsweise Daten-Streams, ausgerichtet ist. Zur Analyse von AVB-Netzwerken wird eine existierende Adaption des Network-Calculus verwendet. Zwei Beispiel-Modelle werden untersucht. Für Modell 1 liegen Vergleichswerte vor, Modell 2 wird innerhalb dieser Arbeit entworfen. Die Ergebnisse für Modell 1 weichen, verursacht durch unterschiedliche Parametrisierungen der Netzwerk-Konfiguration, von den Vergleichswerten ab. Von den Ergebnis-Differenzen abgesehen deutet Modell 1 darauf hin, dass Latenz-Garantien des AVB-Standards unter dieser Konfiguration nicht eingehalten werden. Modell 2 erzeugt Analyse-Ergebnisse, die innerhalb der Garantie-Beschränkungen liegen.

Um die Parametrisierungen der Modelle und die daraus resultierenden Ergebnisse zu evaluieren, müssen die Ergebnisse im nächsten Schritt nachvollzogen werden. Zu diesem Zweck können die Modelle simuliert werden. Die Simulation des Modells muss so konfiguriert werden, dass das Szenario auftritt, das die Worst-Case-Latenz in der Analyse verursacht. Dieser Evaluierungsprozess wird in einer weiterführenden Arbeit ausgeführt.

Ebenfalls für die weiterführende Arbeit wird eine weitere Adaption des Network-Calculus-Modells vorgenommen. Die Entwicklung des AVB-Standards wird unter dem Namen TSN eine TDMA-Erweiterung enthalten, die für geringere Latenzen sorgen soll. Das Network-Calculus-Modell wird die Kombination aus den bisherigen Paket-Klassen von AVB und der neuen TDMA-Klasse enthalten. Zu berücksichtigen ist, dass die TDMA-Pakete in der Priorität über Klasse-A-Paketen stehen werden und damit eine neue Hierarchie einführen. Die Analyse wird an einem Fahrzeug-Modell durchgeführt, das bereits für andere Arbeiten die Forschungsgrundlage bildete. Ein Simulation dieses Fahrzeug-Modells wird den Evaluierungsprozess für die Network-Calculus-Adaption ausführen. Die Ziele der Arbeit werden die Konfiguration eines Fahrzeug-Modells mit TDMA- und normalem AVB-Verkehr, eine Analyse mit anschließender Simulation und die daraus resultierende Möglichkeit, Latenz-Garantien zu geben, umfassen.

Literaturverzeichnis

[1] Institute of Electrical and Electronics Engineers, Inc., "*IEEE 802.1: 802.1Q - Virtual LANs*." http://www.ieee802.org/1/pages/802.1Q.html. zuletzt abgerufen am 23.12.2014.

[2] Institute of Electrical and Electronics Engineers, Inc., "*IEEE 802.1 AV Bridging Task Group*." http://ieee802.org/1/pages/avbridges.html. zuletzt abgerufen am 06.05.2013.

[3] P. Thiran and J.-Y. L. Boudec, *Network Calculus - A Theory of Deterministic Queuing Systems for the Internet*. Berlin Heidelberg: Springer Science & Business Media, 2001. aufl. ed., 2001.

[4] R. Queck, "Analysis of Ethernet AVB for automotive networks using Network Calculus," in *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on*, pp. 61–67, 2012.

[5] "IEEE standard for local and metropolitan area networks - virtual bridged local area networks amendment 12: Forwarding and queuing enhancements for time-sensitive streams," *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, pp. C1–72, 2009.

[6] "IEEE standard for local and metropolitan area networks—virtual bridged local area networks amendment 14: Stream reservation protocol (srp)," *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)*, pp. 1–119, 2010.

[7] "IEEE standard for local and metropolitan area networks - timing and synchronization for time-sensitive applications in bridged local area networks," *IEEE Std 802.1AS-2011*, pp. 1–292, 2011.

[8] "IEEE standard for local and metropolitan area networks–audio video bridging (avb) systems," *IEEE Std 802.1BA-2011*, pp. 1–45, 2011.

[9] Institute of Electrical and Electronics Engineers, Inc., "*IEEE 802.1 Time Sensitive Networking Task Group*." http://www.ieee802.org/1/pages/tsn.html. zuletzt abgerufen am 23.12.2014.

[10] Institute of Electrical and Electronics Engineers, Inc., "*IEEE 802.3 ETHERNET WORKING GROUP*." http://www.ieee802.org/3/. zuletzt abgerufen am 23.12.2014.

[11] Distributed Computing Group, "*Event Systems Chapter 7 - Network Calculus*." http://disco.ethz.ch/lectures/ws0405/eventsystems/lecture/chap7q.pdf. zuletzt abgerufen am 22.03.2015.

[12] J.-P. Georges, T. Divoux, and E. Rondeau, "Strict Priority versus Weighted Fair Queueing in Switched Ethernet Networks for Time Critical Applications," in *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, pp. 141–141, April 2005.