Determinism for Ethernet flows in industrial networks

Lukas Grillmayer
Advisor: Fabien Geyer
Seminar Innovative Internet Technologies and Mobile Communications (IITM) SS2014
Chair for Network Architectures and Services
Department of Informatics, Technische Universität München
Email: lukas.grillmayer@tum.de

ABSTRACT

Due to technological advances, network architectures must handle a growing amount of exchanged data and number of connections. This results in increasingly complex systems. In the domain of industrial systems, it is necessary to provide performance guarantees for large networks. This includes bounded end-to-end latency, bounded end-to-end jitter, and maximum buffer usage.

This paper presents the analytic approach of Network Calculus by providing a basic theoretical background and applying it to simple systems. Finally it is shown how Network Calculus can be applied to Ethernet and AFDX (Avionics Full Duplex Switched Ethernet) networks.

Keywords

Determinism, Industrial Networks, Network Calculus, Ethernet, AFDX

1. INTRODUCTION

Networks enable communication between numerous computational devices and are omnipresent in today's world. As technology advances the amount of electronic communication partners and quantity of exchanged data increases drastically. This results in new challenges and requirements for a network infrastructure as the competition for network resources increases. Within the scope of this paper a network is considered to be a system which can be divided into sub systems such as servers or buffers. The effects of lacking resources are easily observable in the domain of real-time applications, where computer science differentiates between soft and hard systems. For example VoIP internet telephony is considered a soft system as a loss of frames will only result in degraded service quality, even system failure does not impose a major problem. In contrast, an aircraft autopilot is a hard real-time system. If a deadline is missed and sensor data arrives too late at the autopilot, the entire system might fail and endanger human lives as well as the physical environment [8].

Industrial networks often deploy custom software and hardware to achieve compliance with a set of requirements, needed for hard real-time applications. In other words, industrial networks must be deterministic. In order to provide guarantees, these networks need to be analyzed towards their worst-case behavior. Common approaches are Network Calculus, queuing networks simulation, and model checking [9].

This paper uses deterministic Network Calculus as described by Jean-Yves Le Boudec [3], who also provided two short tutorials on network calculus [1, 2].

The following section presents Network Calculus, a collection of theorems which forms a tool to analytically determine performance bounds for systems. This section introduces fundamental properties of networks and provides background information of the proposed mathematical framework. Section 3 discusses how Ethernet technology can be adapted to suit hard real-time applications and shows how Network Calculus can be applied to Avionics Full Duplex Switched Ethernet (AFDX).

2. NETWORK CALCULUS

Network calculus is an algebraic framework to describe and analyze fundamental properties of integrated service networks. It provides insight into network flow problems, to form an understanding of e.g. window flow control, scheduling, and buffer dimensioning. Using network calculus a system can be analyzed towards compliance to given performance constraints, i.e. required for real-time applications [3].

2.1 Basics

In contrast to traditional system theory network calculus operates on the Min-Plus algebra (also referred to as Min-Plus dioid, Min-Plus semiring or $< min, + >$) where addition is the computation of the minimum and multiplication is addition [1].

Convolution and deconvolution of two functions f and g are defined as follows [1]:

$$(f \otimes g)(t) = \inf_{0 \le s \le t} \{f(t-s) + g(s)\} \tag{1}$$

$$(f \oslash g)(t) = \sup_{u \le 0} \{f(t+u) - g(u)\} \tag{2}$$

A data flow in a system is measured in data units per time interval, e.g. bit/s. It is characterized by a non-decreasing function $R(t) \in \mathcal{F}$, whereas $t \in \mathbb{R}^+$ denotes time [1, 4]:

$$\mathcal{F} = \left\{ f : \mathbb{R}^+ \to \mathbb{R}^+, \forall t \ge s : f(t) \ge f(s), f(0) = 0 \right\} \tag{3}$$

As an entire network is hard to model, it can be abstracted as a system consisting of numerous smaller subsystems, which are easier to analyze. Every system has an input flow $R(t)$, and an output flow $R^*(t)$. While processing data it imposes

doi: 10.2313/NET-2014-08-1_10

a (variable) delay $d(t)$ on an input flow. The amount of data traversing a system at time t is called backlog $R(t) - R^*(t)$. For a loss-less system this is defined as follows whereas τ denotes the time window width [4, 3]:

$$d(t) = \inf \{\tau \geq 0 : R(t) \leq R^*(t + \tau)\} \quad (4)$$

An example for a simple system $\mathcal{S}$ is a FIFO queue (or buffer), which experiences a constant input rate r, thus the input flow is characterized by $R(t) = rt$. The output flow is denoted as $R^*(t) = R(t + d(t))$. Figure 1 illustrates this with $r = 10Mbit/s$ and $d(t) = 5s$.

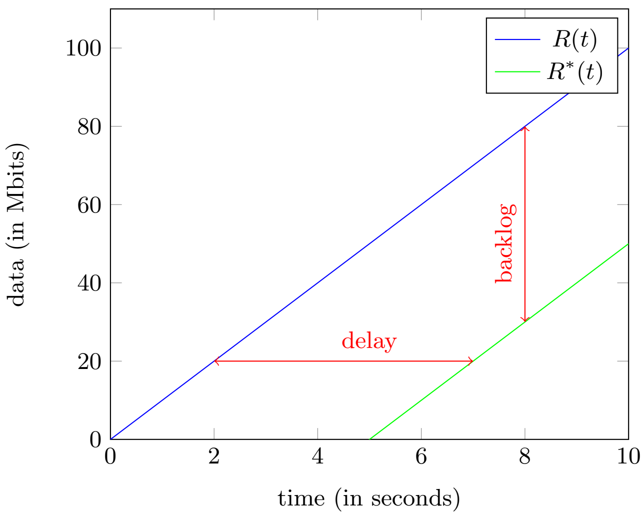

Figure 1: Illustration of input and output flow

2.2 Arrival Curves

Requirements for industrial networks commonly ask for guarantees for data flows. For a data flow $R(t)$ this can be modeled by an arrival curve $\alpha(t) \in \mathcal{F}$ with $t \geq 0$. $R(t)$ is limited by $\alpha(t)$ if for all $s \leq t$ [3]:

$$R(t) - R(s) \leq \alpha(t - s) \quad (5)$$

This is equal to:

$$R(t) \leq \inf_{0 \leq s \leq t} \{\alpha(t - s) + R(s)\} = (\alpha \otimes R)(t) \quad (6)$$

For example let $R(t)$ have an arrival curve of $\alpha(t) = rt$ then the flow is "peak rate limited" by $r\tau$ for any time window τ. In case of a constant arrival curve $\alpha(t) = b$ the flow will not accept more than b data ever.

As this representation assumes a loss-less system, affine arrival curves are commonly used to model the concept of leaky buckets. In this analogy a flow is represented as a bucket, which is leaking fluid (data) at a constant rate when not empty. Data is poured into the bucket. If additional data causes the bucket to overflow, the excess data is marked as "non-conformant" and will be discarded or buffered. An affine arrival curve $\gamma_{r,b}$ with burst tolerance (maximal number of parallel incoming bits) b and rate r is defined as follows [1]:

$$\gamma_{r,b} = \begin{cases} rt + b & t > 0 \\ 0 & t \leq 0 \end{cases} \quad (7)$$

In the domain of Integrated services framework of the Internet (Interserv) additional parameters for maximum packet size M and sustainable rate r are introduced to calculate the arrival curve. As a result the traffic specification (T-SPEC) can be given as (p, M, r, b) [1]:

$$\alpha(t) = \min \{M + pt, rt + b\} = \gamma_{p,M}(t) \wedge \gamma_{r,b}(t) \quad (8)$$

2.3 Service Curves

In addition to arrival curves, which have put constraints on a flow, systems must also offer guarantees to a flow. Generally this is realized by packet schedulers. Network calculus represents schedulers as service curves. A system $\mathcal{S}$ with input flow $R(t)$ and output flow $R^*(t)$ offers a service curve β to the flow if $\forall t \geq 0$ $\exists t_0 \geq 0$ with $t_0 \leq t$ to satisfy the following [1]:

$$R^*(t) - R(t_0) \geq \beta(t - t_0) \quad (9)$$

This is equal to:

$$R^*(t) \geq \inf_{0 \leq s \leq t} \{\beta(t - s) + R(s)\} = (\beta \otimes R)(t) \quad (10)$$

2.4 Bounds for simple networks

Service and arrival curves can be used to simplify some expressions. For lossless networks with service guarantees following theorems can be applied to analyze a system towards guarantees concerning bounded end-to-end latency, bounded end-to-end jitter, and maximum buffer usage [3].

2.4.1 Backlog Bound

Theorem 1.4.1 (Backlog Bound): Assume a flow, constrained by arrival curve α, traverses a system that offers a service curve of β. The backlog $R(t) - R^*(t)$ for all t satisfies:

$$R(t) - R^*(t) \leq \sup_{0 \leq s \leq t} [R(t) - R(t - s) + \beta(s)] \quad (11)$$

$$\leq \sup_{0 \leq s \leq t} [\alpha(s) + \beta(t - s)] \quad (12)$$

2.4.2 Delay Bound

Theorem 1.4.2 (Delay Bound): Assume a flow, constrained by arrival curve α, traverses a system that offers a service curve of β. The virtual delay $d(t)$ for all t satisfies:

$$d(t) \leq h(\alpha, \beta) \quad (13)$$

2.4.3 Output Flow

Theorem 1.4.3 (Output Flow): Assume a flow, constrained by arrival curve α, traverses a system that offers a service curve of β. The output flow is constrained by the arrival curve

$$\alpha^* = \alpha \otimes \beta \quad (14)$$

doi: 10.2313/NET-2014-08-1_10

2.5 Applied Network Calculus

This section shows how the network calculus methods presented above can be applied to a small network.

2.5.1 Example 1 - Single System

Consider a network consisting of one server $\mathcal{S}$ with service curve $\beta_{10,10}$, and one traversing flow with arrival curve $\gamma_{5,25}$.

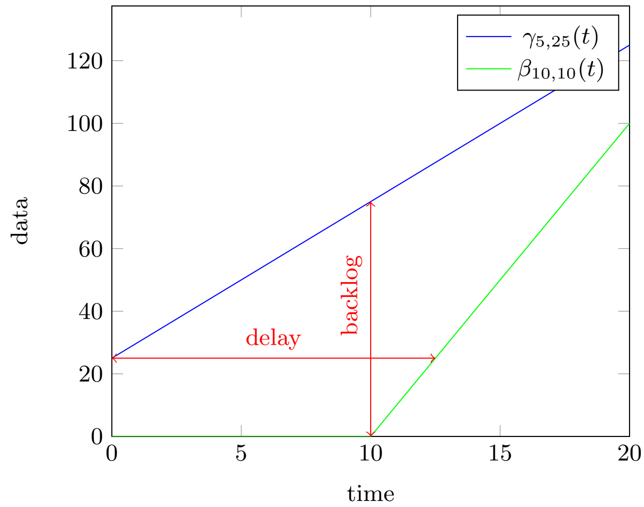

Figure 2: Arrival and service curve of a system

Figure 2 shows the system's arrival curve and service curve. The required backlog (or buffer size) of the server can be computed as presented in subsection 2.4.1. For all t the backlog bound must satisfy:

$$\begin{aligned} R(t) - R^*(t) &\leq \sup_{0 \leq s \leq t} [R(t) - R(t-s) + \beta(s)] \\ &\leq \sup_{0 \leq s \leq t} [\alpha(s) + \beta(t-s)] \\ &\leq \sup_{s \geq 0} [\alpha(s) - \beta(s)] \\ &\leq \sup_{s \geq 0} [\gamma_{5,25}(s) - \beta_{10,10}(s)] = 75 \end{aligned}$$

The maximum delay (or latency) of the flow can be computed according to subsection 2.4.2. For all t the delay bound must satisfy:

$$\begin{aligned} d(t) &= \inf \{\tau \geq 0 : \alpha(t) \leq \beta(t+\tau)\} \\ &= \inf \{\tau \geq 0 : \gamma_{5,25}(t) \leq \beta_{10,10}(t+\tau)\} \\ &\leq h(\alpha, \beta) \\ &\leq h(\gamma_{5,25}, \beta_{10,10}) \\ &\leq \sup_t \{d(t)\} = 12.5 \end{aligned}$$

Following equation 5 the constraint of the output flow (arrival curve) is as follows. For all t the arrival curve $\alpha^*(t)$ must satisfy:

$$\begin{aligned} \alpha^*(t-s) &\geq R^*(t) - R^*(s) \\ &\geq R^*(t) + 75 = 5t + 75 \end{aligned}$$

2.5.2 Example 2 - Concatenated Systems

Consider a network consisting of two servers $\mathcal{S}_1$ and $\mathcal{S}_2$ with service curve $\beta_{10,10}$, and one flow with arrival curve $\gamma_{5,25}$ traversing first $\mathcal{S}_1$ and then $\mathcal{S}_2$.

Some results from subsection 2.5.1 can be reused as the system $\mathcal{S}$ is equal to $\mathcal{S}_1$ now. Thus, the curve of the flow $R_1(t)$ after it has traversed $\mathcal{S}_1$ is

$$R_1(t) = 5t + 75$$

The curve of the flow after it has traversed $\mathcal{S}_2$, that is the entire system $\mathcal{S}$:

$$R^*(t) = R_1(t) + 75 = 5t + 150$$

The latency of the flow after $\mathcal{S}_1$:

$$\begin{aligned} D_1 &= \frac{b}{R_1} + T_1 \\ &= \frac{25}{10} + 10 = 12.5 \end{aligned}$$

The latency of the flow after $S2$:

$$\begin{aligned} D_2 &= \frac{b + rT_1}{R_2} + T_2 \\ &= \frac{25 + 5 * 10}{10} + 10 = 17.5 \end{aligned}$$

The resulting end-to-end delay D:

$$\begin{aligned} D &= D_1 + D_2 \\ &= 12.5 + 17.5 = 30 \end{aligned}$$

According to the *Pay Bursts Only Once* theorem [3] the end-to-end delay D_0 of concatenated systems can be improved as follows. Let $R = \min_i(R_i)$ and $T_0 = \sum_i T_i$:

$$\begin{aligned} D_0 &= \frac{b}{R} + T_0 \\ &= \frac{25}{10} + 20 = 22.5 \end{aligned}$$

3. ETHERNET

Ethernet was first standardized in 1983 as IEEE 802.3 and is the most wide spread transmission technology for local area networks today.

In early Ethernet networks a single coaxial cable was used as shared broadcast media in a bus architecture. If a connected host wanted to send a message, it had to wait until the medium is idle. All hosts were part of the same collision domain and if two hosts sent messages at the same time a collision occurred which was detected. After a suitable back-off strategy a host may retransmit the message. This strategy is referred to as CSMA/CD (Carrier Sense, Multiple Access, Collision Detection). At this time Ethernet was not suited for industrial networks as there was no central coordination for broadcast media access and the back-off interval increased exponentially plus a randomly chosen number to time slots, which results in non-deterministic behavior [7].

Over the years the coaxial cables were replaced by twisted pair copper cables. Common category 5 UTP cables include two twisted pairs of copper wire (Tx and Rx). In half-duplex mode the issues discussed in previous paragraph are not resolved since transmissions can still collide and an exponential back-off algorithm can result in very large transmission delays [7].

The usage of switches reduced the size of collision domains significantly. In Full-duplex Mode Ethernet there are dedicated wires for each directions of transmission, thus allowing simultaneous transmitting and receiving without collisions. Figure 3 shows an example for full-duplex, switched Ethernet, in which collisions are impossible but other issues need to be considered. In theory the Rx and Tx buffers can overflow, which results in packet loss. Using Network Calculus the buffers can be sized adequately to form a loss-less system. Furthermore, packets may be delayed due to congestion within the switch, which results in an unpredictable amount of jitter [7].

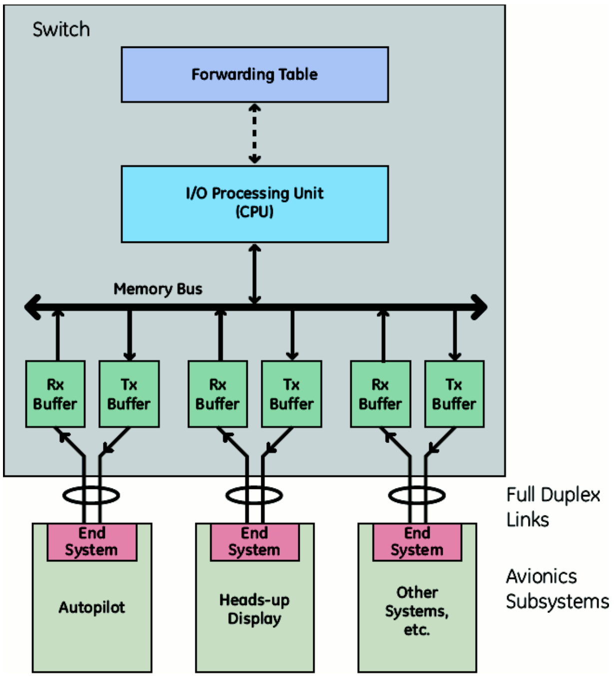

Figure 3: Full-Duplex Switched Ethernet [7]

In order to apply Network Calculus to actual hardware a model needs to be derived. To model an Ethernet switch we need to split the switch into smaller subsystems, which represent various functionality aspects of an Ethernet switch.

Being overly simplistic we are assuming omniscient knowledge about the network. Now, an unmanaged full-duplex Ethernet switch with sufficient backplane bandwidth could be modeled as a single system which adds a (variable) delay to an input signal. This is the time required to map the destination MAC address to a port and redirect the Ethernet packet correctly. Of course, this could also be split into more subsystems. The arrival curve would be determined by the maximum network speed and the maximum Ethernet packet size.

3.1 AFDX

In the domain of avionic networks, the Airbus Group developed an Ethernet derivative based on IEEE 802.3 called Avionics Full-Duplex Switched Ethernet (AFDX), which is defined in ARINC 664 section 7 [6, 7]. An example of AFDX deployment is the Airbus A380. The major difference to regular Ethernet is the usage of Virtual Links (VL). Portions of the available bandwidth of the physical link are dedicated to Virtual Links as illustrated in figure 4. Virtual Links are unidirectional fixed communication paths from one end system to one or more end systems.

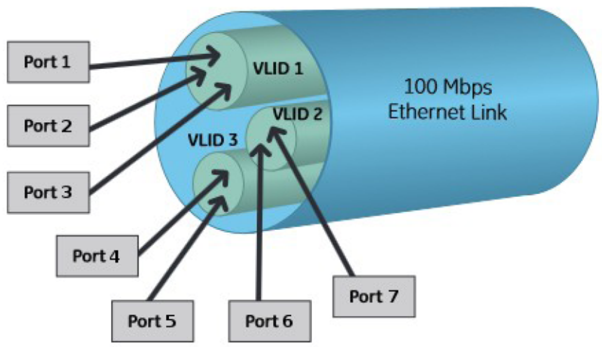

Figure 4: Three Virtual Links in a physical link [7]

For every VL the maximum packet size s_{min} and minimum time between two packets BAG is known. This allows for the formulation of guarantees by means of network calculus [6, 7]. Exemplary, an arrival curve $\alpha(t)$ for AFDX is defined as follows:

$$\alpha(t) = \min\left\{t, s_{min} + \frac{s_{min}}{BAG}t\right\}$$

4. TOOL: DISCO DNC

There are only few tools available for network calculus. The interested reader might want to look at the *DISCO Deterministic Network Calculator*. This Java library supports network analysis by means of deterministic network calculus presented above [5].

5. CONCLUSION

In the domain of industrial networks it is important to formulate performance guarantees. In addition to hardware constraints this requires the usage to deterministic protocols. As the development and certification of custom application specific standards is very expensive a current trend is to adapt Ethernet to fit the needs. An example for an Ethernet based derivative is AFDX, which adds central coordination by dedicating network bandwidth to certain communication paths. Using Network Calculus upper bounds can be calculated for required buffer size, end-to-end delay, and end-to-end jitter. It is apparent that upper bounds can only exist if the underlying model of the system has deterministic characteristics. Using traffic shapers even non-deterministic aspects can be modeled in an abstract manner to a certain extend although this might decrease the significance of the result. Network Calculus should only be used to analyze or model a system with hard requirements. The presented approach might be overly pessimistic as only the worst-case behavior is considered though it might be very unlikely. In

doi: 10.2313/NET-2014-08-1_10

order to evaluate the average system performance one should resort to network simulations.

6. REFERENCES

[1] J.-Y. Le Boudec, P. Thiran: *A SHORT TUTORIAL ON NETWORK CALCULUS - I: FUNDAMENTAL BOUNDS IN COMMUNICATION NETWORKS*, In IEEE International Symposium on Circuits and Systems, pages IV-93 - IV95, IEEE, Lausanne, Switzerland, 2000

[2] J.-Y. Le Boudec, P. Thiran, S. Giordano: *A SHORT TUTORIAL ON NETWORK CALCULUS - II: MIN-PLUS SYSTEM THEORY APPLIED TO COMMUNICATION NETWORKS*, IEEE, Lausanne, Switzerland, 2000

[3] J.-Y. Le Boudec, P. Thiran: *Network Calculus: A Theory of Deterministic Queuing Systems for the internet*, Springer, Berlin, Germany, 2001

[4] J.-P. Georges, T. Divoux, É. Rondeau: *Network Calculus: Application to switched real-time networking*, In Proceedings of the 5th International ICST Conference on Performance Evaluation Methodologies and Tools, pages 399-407, ICST, Vandœuvre lès Nancy, France, 2011

[5] *DISCO Deterministic Network Calculator*, Technische Universität Kaiserslautern, `http://disco.informatik.uni-kl.de/index.php/projects/disco-dnc` (accessed on 2014/07/01)

[6] C. Fuchs: *The Evolution of Avionics Networks From ARINC 429 to AFDX*, In Proceedings of the Seminars Future Internet (FI), Innovative Internet Technologies and Mobile Communication (IITM) and Aerospace Networks (AN), Summer Semester 2012, p.65-76, Technische Universität München, Munich, Germany, 2012

[7] General Electric: *AFDX/ARINC 664 Protocol: Tutorial*, General Electric, Fairfield, CT, USA

[8] Dr. C. Buckl: *Echtzeitsysteme WS 2013/14*, Technische Universität München, Munich, Germany, 2013, `http://www6.in.tum.de/Main/TeachingWs20123chtzeitsysteme` (accessed on 2014/07/01)

[9] H. Charara, J.-L. Scharbarg, J. Ermont, C. Fraboul: *Methods for bounding end-to-end delays on an AFDX network*, In Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS'06), Toulouse, France, 2006

doi: 10.2313/NET-2014-08-1_10