

Tightening end to end delay upper bound for AFDX network with rate latency FIFO servers using network calculus

Marc Boyer, Christian Fraboul

IRIT Lab.
ENSEEIHHT / University of Toulouse

WFCS 2008
Mai, 21th, 2008

Outline

Network calculus

What is network calculus

A very short introduction to networks calculus

When to use it ?

The AFDX network

Modeling AFDX in NC

Our contribution : get the “pay bust only once” benefit

Conclusion

Outline

Network calculus

What is network calculus

A very short introduction to networks calculus

When to use it ?

The AFDX network

Modeling AFDX in NC

Our contribution : get the “pay bust only once” benefit

Conclusion

What is network calculus ?

- ▶ A theory to get guaranteed upper bounds for buffers and delays in communication networks [Cruz91][Le Boudec01])
To provide guaranteed results, it needs
 - ▶ guarantees on bandwidths and nodes services
 - ▶ guarantees on maximal input flows
- ▶ Based on the $(\wedge, +)$ theory
 $\wedge \equiv \min$
- ▶ Mainly used for Internet flows
- ▶ Adapted to embedded (complex) systems, like AFDX

Three basic mathematics operations

Network calculus mainly uses three operators on functions from the $(\wedge, +)$ dioid.

$$\text{Convolution} \quad (f \otimes g)(t) = \inf_{0 \leq u \leq t} (f(t-u) + g(u))$$

$$\text{Deconvolution} \quad (f \oslash g)(t) = \sup_{0 \leq u} (f(t+u) - g(u))$$

$$\text{Sub-additive closure} \quad \bar{f} = \delta_0 \wedge f \wedge (f \otimes f) \wedge (f \otimes f \otimes f) \wedge \dots$$

Three basic notions

- ▶ A flow is defined by its *cumulative function* R
 - ▶ $R(t)$: the total number a bits up to time t
 - ▶ in general, a flow is defined by its throughput r

$$R(t) = \int_0^t r(u) du$$

- ▶ A flow R has an *arrival function* $\alpha \iff R \leq R \otimes \alpha$

$$\forall t, s \geq 0 : R(t + s) - R(t) \leq \alpha(s)$$

- ▶ A network element S offers to its input flow R a *service curve* β , producing an output flow $R' \iff R' \geq R \otimes \beta$

Two famous curves

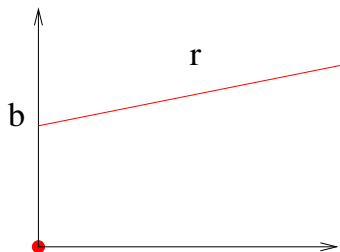


FIG.: A leaky bucket arrival curve :
 $\gamma_{r,b}$

$$\gamma_{r,b}(t) = \begin{cases} 0 & \text{if } t < 0 \\ rt + b & \text{otherwise} \end{cases}$$

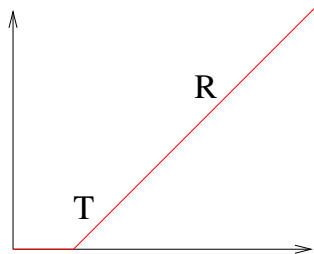
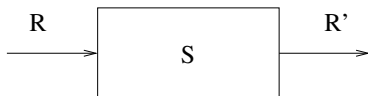


FIG.: A rate-latency service curve :
 $\beta_{R,T}$

$$\beta_{R,T}(t) = \begin{cases} 0 & \text{if } t < T \\ R(t - T) & \text{otherwise} \end{cases}$$

Two main results



- ▶ The bounds on delays (h) and buffer (v) can be computed

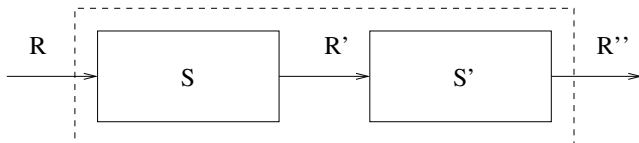
$$h(\alpha, \beta) = \sup_{s \geq 0} (\inf \{ \tau \geq 0 \mid \alpha(s) \leq \beta(s + \tau) \})$$

$$v(\alpha, \beta) = \sup \{ \alpha(x) - \beta(x) \}$$

- ▶ The computation can be propagated : we can compute α' , an arrival curve for R'

$$\alpha' = \overline{\alpha \otimes \beta}$$

A famous result : pay burst only once



The result : The sequence of two nodes S, S' with service curves β, β' is equivalent to a single node with service curve $\beta \otimes \beta'$.

Main interest : Gives a better end-to-end bound, because a burst can be in S , in S' , but not in both at the same time.

“pay burst only once”

When to use it ?

Network calculus :

- ▶ could seem complex (mathematics : dioid, convolution...)
- ▶ is a generic theory : in specific cases, specific algorithms are better, often based on scheduling theory

But :

- ▶ no need of high mathematics : enough well known results
 - ▶ the leaky bucket and periodic flows are modelled by a $\gamma_{r,b}$ arrival curve
 - ▶ the rate-latency service is modelled by a $\beta_{R,T}$ curve

$$\overline{\gamma_{r,b} \otimes \beta_{R,T}} = \gamma_{r,b+rT}$$

- ▶ the computational complexity is low

Moreover, it often is the only solution in multiple hop topology :

- ▶ computation can be propagated hop by hop
NC gives bounds *and* arrival curve of the output flow
- ▶ the “pay burst only once” principle gives good results

Outline

Network calculus

- What is network calculus

- A very short introduction to networks calculus

- When to use it ?

The AFDX network

Modeling AFDX in NC

Our contribution : get the “pay bust only once” benefit

Conclusion

AFDX : a switched Ethernet Backbone

Goals of the avionics manufacturers :

- ▶ reduce to weight of the plane
- ▶ reduce the connecting complexity
- ⇒ have a shared backbone
 - ▶ use a well known, low cost hardware
- ⇒ Ethernet
 - ▶ keep determinist upper bound
- ⇒ remove the medium access indeterminacy : full duplex links
- ⇒ compute upper bounds for delays in switch queues : network calculus

Virtual Link

- ▶ To use NC, the traffic must respect some constraint.
- ▶ A Virtual Link is a static mono-sender multicast flow.
- ▶ It is constrained by
 - ▶ a minimal frame size s^{min}
 - ▶ a maximal frame size s^{max}
 - ▶ a minimal interval between two frames (BAG)
- ▶ links and switch queues are shared by different VLs



Outline

Network calculus

What is network calculus

A very short introduction to networks calculus

When to use it ?

The AFDX network

Modeling AFDX in NC

Our contribution : get the “pay bust only once” benefit

Conclusion

A simple modeling [Grieu03]

- ▶ A VL has an arrival curve $\gamma_{\frac{s^{max}}{BAG}, s^{max}}$
- ▶ The sum of some VL_{*i*} has an arrival curve $\gamma_{\sum \frac{s_i^{max}}{BAG_i}, \sum s_i^{max}}$
- ▶ A switch output queue offers the service curve $\beta_{R, T}$ with R the throughput of the Ethernet link and T the commutation time

It is simple, but the computed delays do not satisfy the A380 applications constraints.

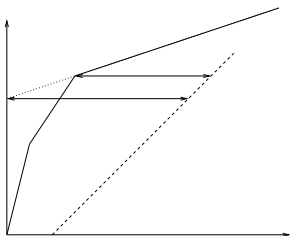
A better modeling [Grieu03] : the shaping of links

Principle whatever the applicative throughput is, it is bounded by the Ethernet throughput

NC VL modeling a VL in the network is constrained by : $(\gamma_{\frac{s^{max}}{BAG}, s^{max}}) \wedge \gamma_{D,0}$

NC and sum of VL the sum of the link-shaped VLs is a concave piecewise linear (CPL) function

Results this modelling was, on an A380 configuration, 40% better than the simple one



Outline

Network calculus

What is network calculus

A very short introduction to networks calculus

When to use it ?

The AFDX network

Modeling AFDX in NC

Our contribution : get the “pay bust only once” benefit

Conclusion

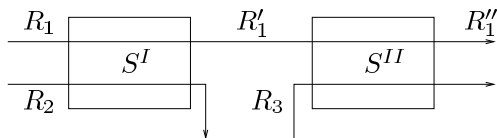
Our contribution : get the “pay bust only once” benefit

- ▶ The previous modelling was used to certify the network

but

- ▶ The upper bounds computed are very high compared to what is measured in reality
 - ▶ The next generation plane could have higher traffic needs
- ⇒ could we get better bounds with NC ?
- ▶ For some technical reasons, the previous modelling computes only local bounds (ie in each switch)
 - ▶ Trying to get the “pay burst only once” (PBOO) benefit ?

The problem



- ▶ To apply the PBBO, the services β_1^I and β_1^{II} offered by the networks elements to R_1 must be computed
- ▶ This computation is pessimistic
- ▶ Then the end-to-end service $\beta_1^I \otimes \beta_1^{II}$ can be computed, and gives the PBOO benefit

The equations to be solved

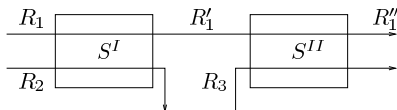
In our context, the arrival curves α_i are piecewise linear concave function. Two equations gives the service curve β_1^l and the output arrival curve α_1^l .

$$\beta_1^\theta(t) = [\beta_{R,T}(t) - \alpha(t - \theta)]^+ 1_{\{t > \theta\}} \quad (1)$$

$$\alpha_1^l = \inf_{\theta \geq 0} (\alpha_1 \circ \beta_1^\theta) \quad (2)$$

We do not solve the equations, but have some upper approximations.

The results



- ▶ The new results have been compared with the other ones on one topology with some values
 - ▶ Local delays (R_2, R_3) : the new method is worst than the one with link shaping and better than the one without link shaping
 - ▶ End-to-end delay (R_1) : the new method is often better than the others
- ▶ Interpretation
 - ▶ Modelling link shaping is important
 - ▶ Very small end-to-end flow : the under approximation introduced by computing the individual service destroys the benefit of the PBOO
 - ▶ PBOO : very efficient for long path

Outline

Network calculus

What is network calculus

A very short introduction to networks calculus

When to use it ?

The AFDX network

Modeling AFDX in NC

Our contribution : get the “pay bust only once” benefit

Conclusion

Conclusion

- ▶ In NC, there is no modelling better than others (up to now?)
- ▶ Ongoing works
 - ▶ Applying the results to a realistic configuration
 - ▶ An open tool for NC
 - ▶ Getting a better upper approximation
 - ▶ Taking into account other aspects (minimal frame size)