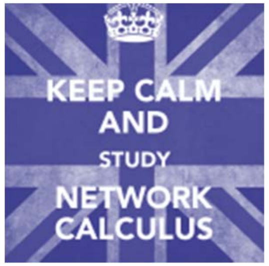


Applicability of Network Calculus to DetNet

Jean-Yves Le Boudec[1,2,3]

EPFL

IETF 100

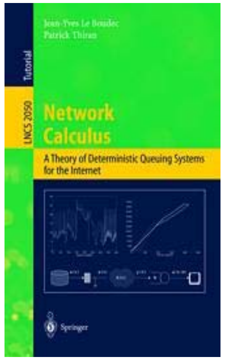
[1] https://people.epfl.ch/105633/research

[2] http://smartgrid.epfl.ch

[3] IP Parallel Redundancy Protocol https://github.com/LCA2-EPFL/iprp

What is Network Calculus ?

A theory and tools to compute bounds on queuing delays, buffers, burstiness of flows, etc



C.S. Chang, R. Cruz, JY Le Boudec, P. Thiran, …

For deterministic networking, per-flow and per-class queuing

Arrival curve, Service curve, Shapers, Concatenation

Where could it be applied to DetNet ?

Which parameters for describing the contribution of a DetNet node to the end-to-end delay bounds ?

More generally, how to describe the parameters of interest of a Detnet node without imposing an implementation ?

How to prove delay bounds for a detnet node ? For a detnet network (e.g. UNI to UNI) ?

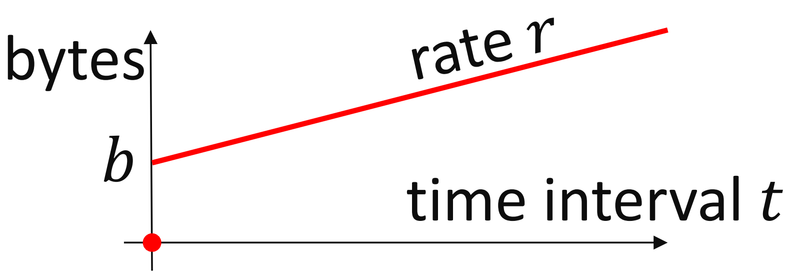
Simplification of Path Computation.

Arrival Curve

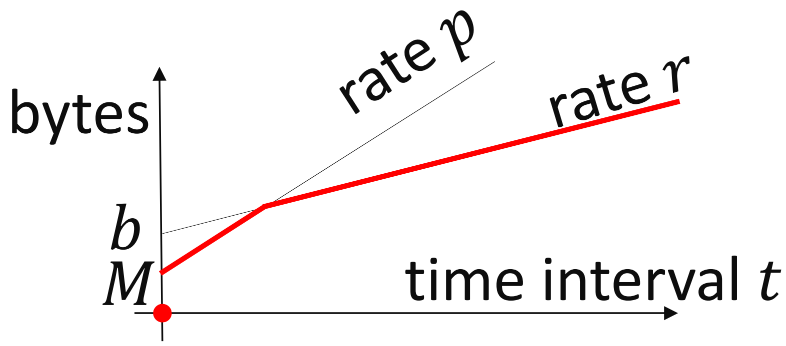
For a flow, at an observation point

Flow is constrained by arrival curve $\alpha()$ iff the amount of basic data units (e.g. bytes) observed in *any interval* of duration t is $\leq \alpha(t)$

token bucket with rate r and burst b: $\alpha(t) = rt + b$

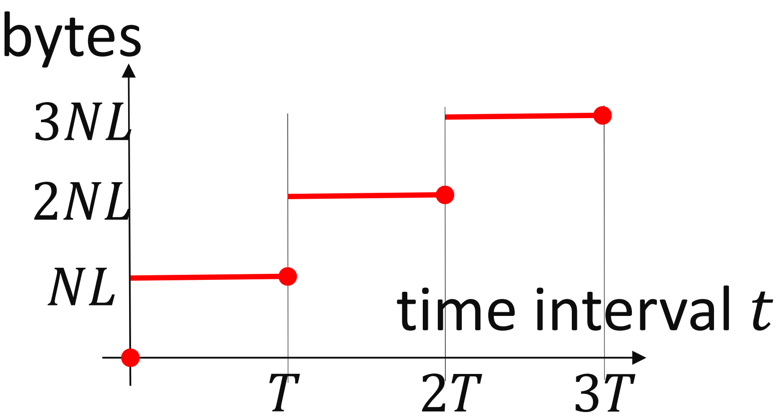


token bucket + peak rate p and MTU M: $\alpha(t) = \min(pt + M, rt + b)$



The Arrival Curve implied by detnet-architecture-03, 4.3.2

At most N transmissions of size at most L in T seconds

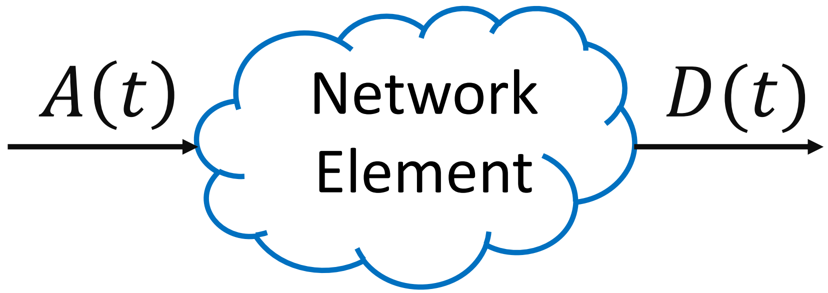


Any arrival curve can be assumed sub-additive

sub-additive:
$\alpha(s+t) \leq \alpha(s)+\alpha(t)$

concave $\Rightarrow$ subadditive

Service Curve

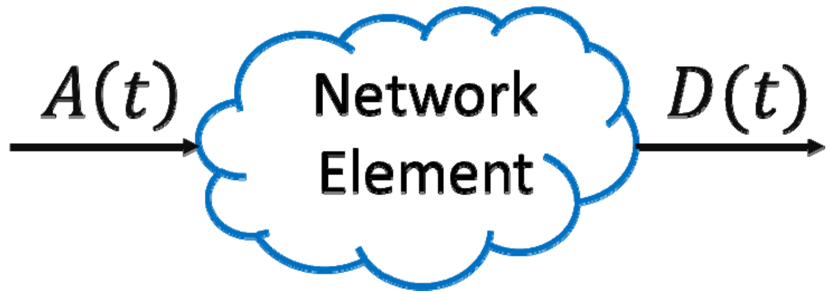


$A(t), D(t)$: amount of basic data units observed in $[0, t]$

Network element offers to this flow a service curve $\beta()$ if

$$\forall t \geq 0, \exists s \in [0, t]: D(t) \geq A(s) + \beta(t - s)$$

Service Curve Example



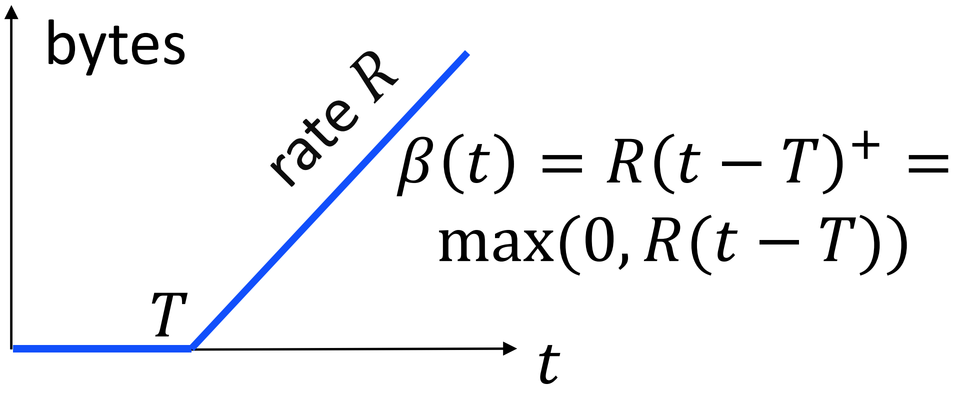
Rate-latency service curve :

$$\beta(t) = R(t-T)^{+}$$

Models many schedulers: DRR, PGPS, RFC 7006, etc.

Example: service received by a high priority flow (no pre-emption):

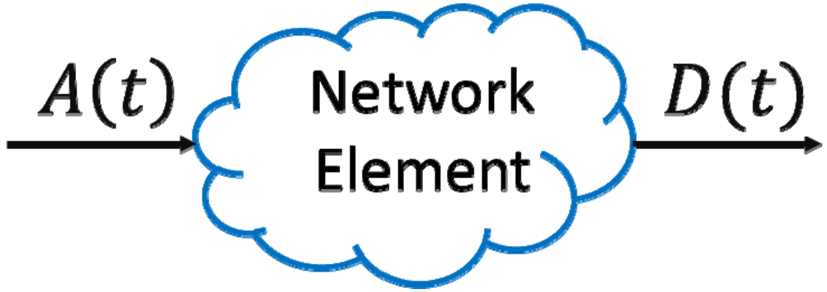
R = line rate
$RT = MTU$ of low priority packets



$$D(t) = A(s) + \beta(t-s)$$

s = beginning of busy period

Service Curve Example: Bounded Delay

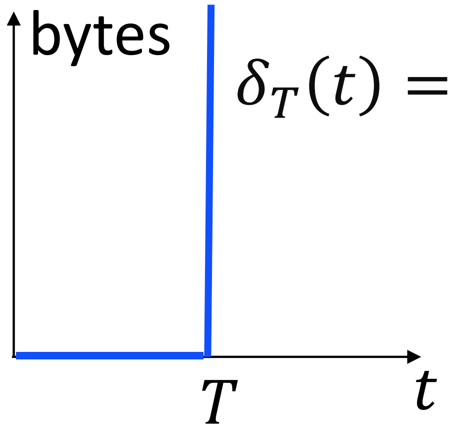


For a FIFO per-flow system:

delay is $\leq T$

$\Leftrightarrow$

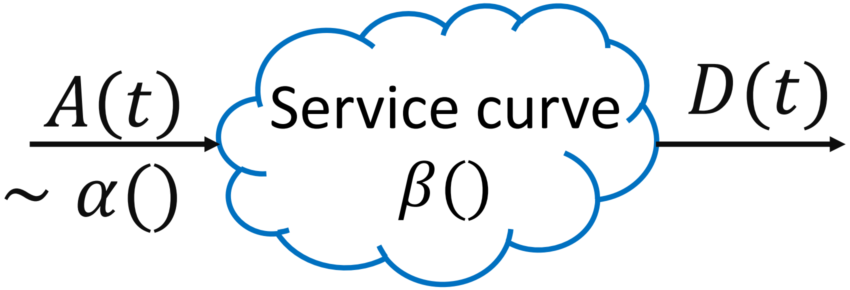
system offers to this flow a service curve equal to the delay function $\beta(t) = \delta_T(t)$



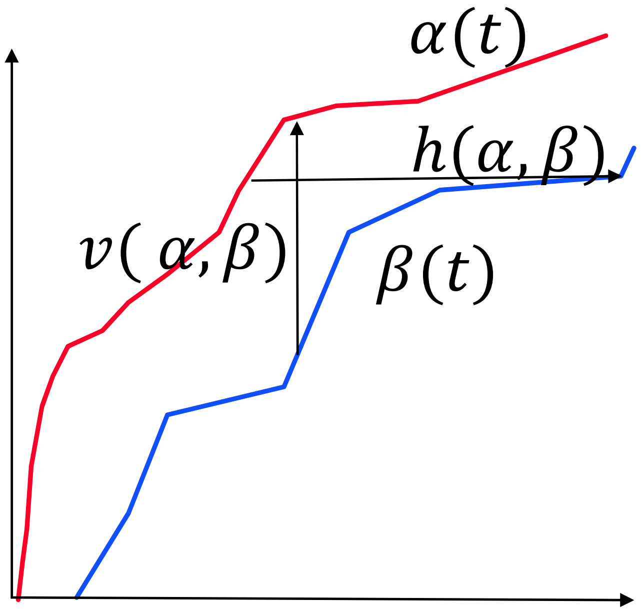
$$\delta_T(t) = \begin{cases} 0 \text{ if } t \leq T \\ +\infty \text{ else} \end{cases}$$

$$D(t) \geq A(t - T)$$

Basic Results: 3 Tight Bounds

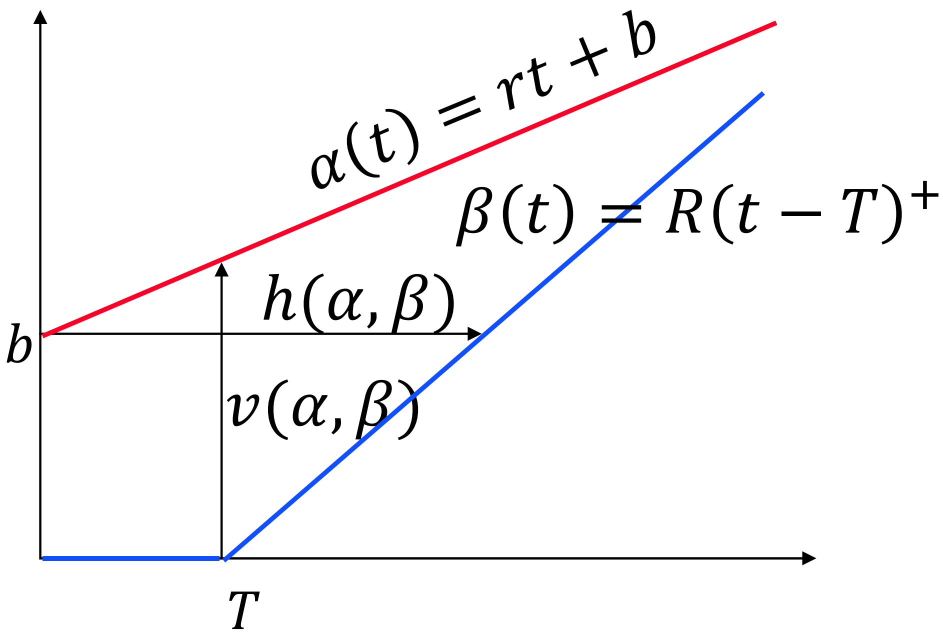


Flow is constrained by arrival curve $\alpha()$; served in network element with service curve $\beta()$. Then



1. backlog $\leq v(\alpha,\beta) = \sup_t \big(\alpha(t) - \beta(t)\big)$
2. if FIFO per flow, delay $\leq h(\alpha,\beta)$
3. output is constrained by arrival curve $\alpha^*(t) = \sup_{u \geq 0} \big(\alpha(t+u) - \beta(u)\big)$

Example



One flow, constrained by one token bucket is served in a network element that offers a rate latency service curve.

Assume $r \leq R$

Backlog bound: $b + rT$

Delay bound: $\frac{b}{R} + T$

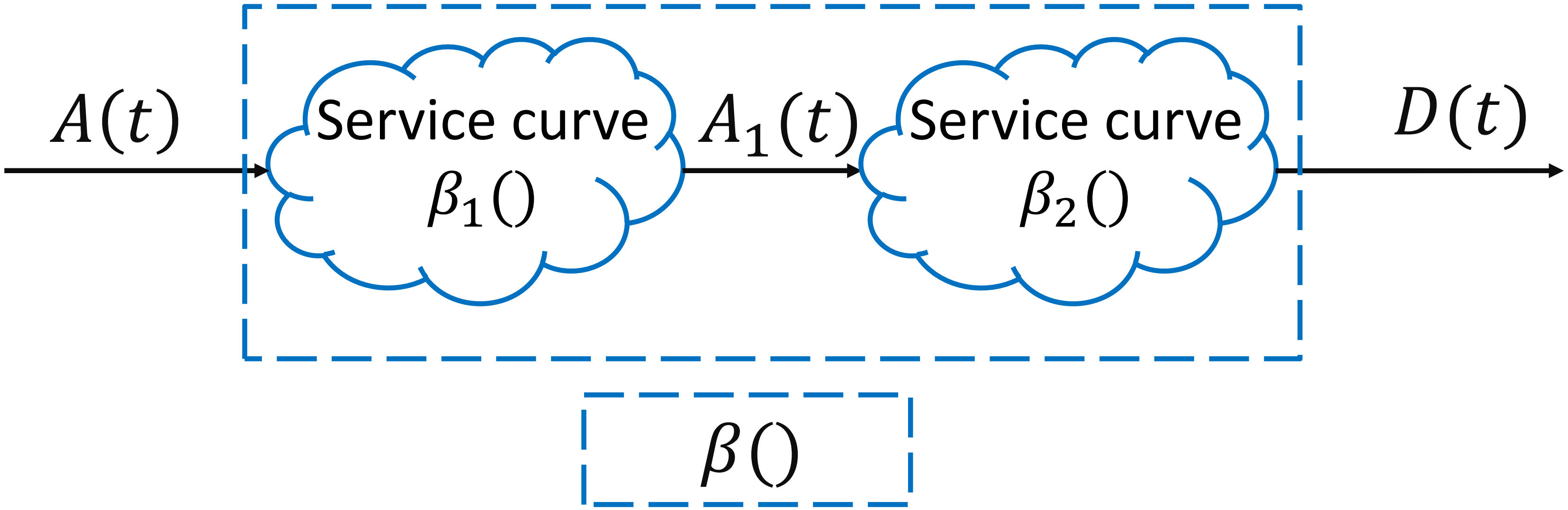
Output arrival curve:

$$\alpha^*(t) = rt + b^*$$

with $b^* = b + rT$

(burstiness b is increased by rT)

Concatenation



A flow is served in series, network element i offers service curve $\beta_i()$.
The concatenation offers the service curve $\beta()$ defined by

$$\beta(t) = \inf_{s \geq 0} \left(\beta_1(s) + \beta_2(t - s)\right)$$

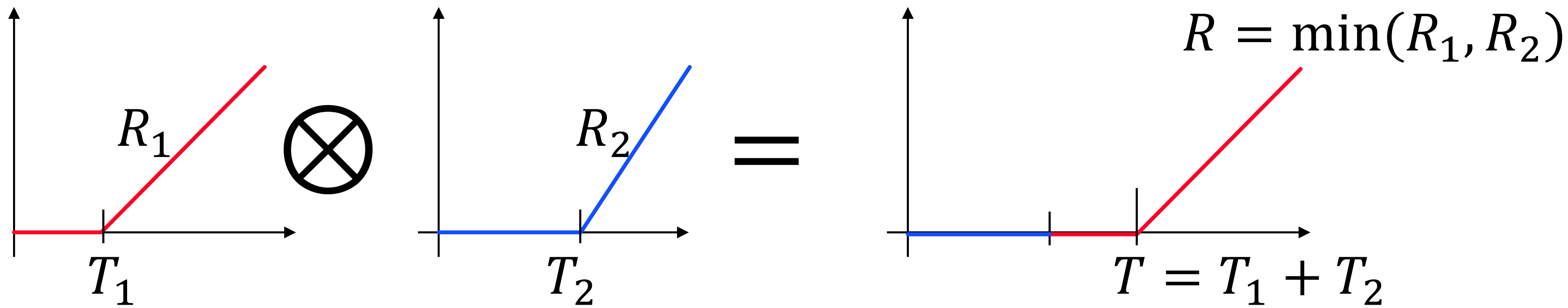
Min-Plus Convolution

$$\beta(t) = \inf_{s \geq 0} \left(\beta_1(s) + \beta_2(t-s)\right)$$
$$\beta = \beta_1 \otimes \beta_2$$

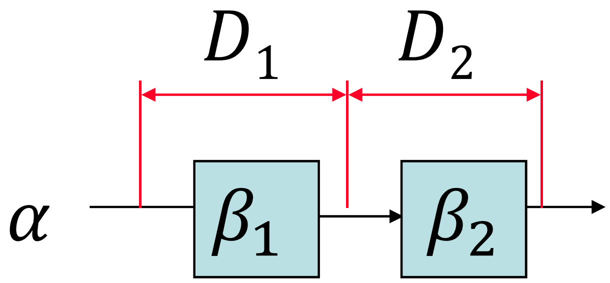
This operation is called *min-plus convolution*. It has the same nice properties as usual convolution; e.g.

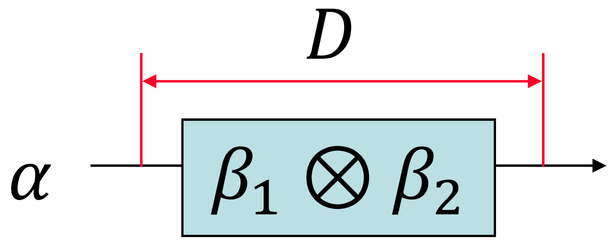
$$(\beta_1 \otimes \beta_2) \otimes \beta_3 = \beta_1 \otimes (\beta_2 \otimes \beta_3)$$
$$\beta_1 \otimes \beta_2 = \beta_2 \otimes \beta_1$$

It can be computed easily: e.g.



Pay Bursts Only Once





$$\alpha(t) = rt + b$$
$$\beta_1(t) = R(t - T_1)^+$$
$$\beta_2(t) = R(t - T_2)^+$$
$$r \leq R$$

one flow constrained *at source* by $\alpha()$

end-to-end delay bound computed *node-by-node* (also accounting for increased burstiness at node 2):

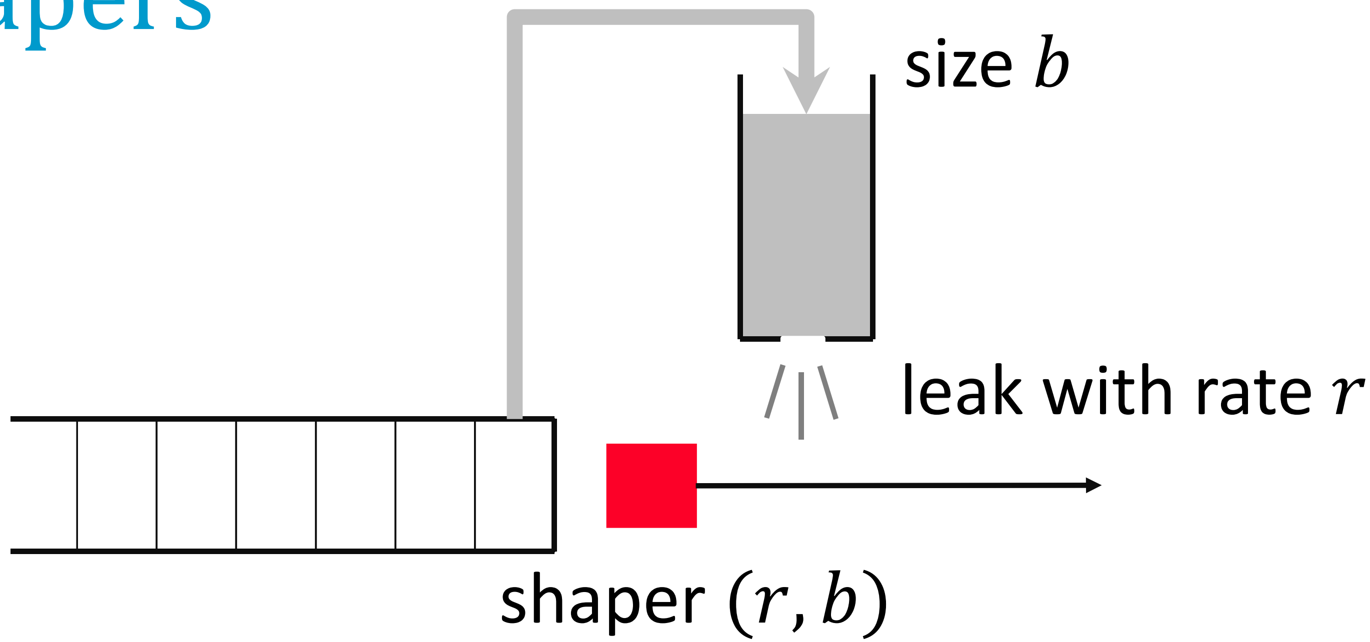
$$D_1 + D_2 = \frac{2b + RT_1}{R} + T_1 + T_2$$

computed *by concatenation:*

$$D = \frac{b}{R} + T_1 + T_2$$

i.e. worst cases cannot happen simultaneously – concatenation captures this !

Shapers

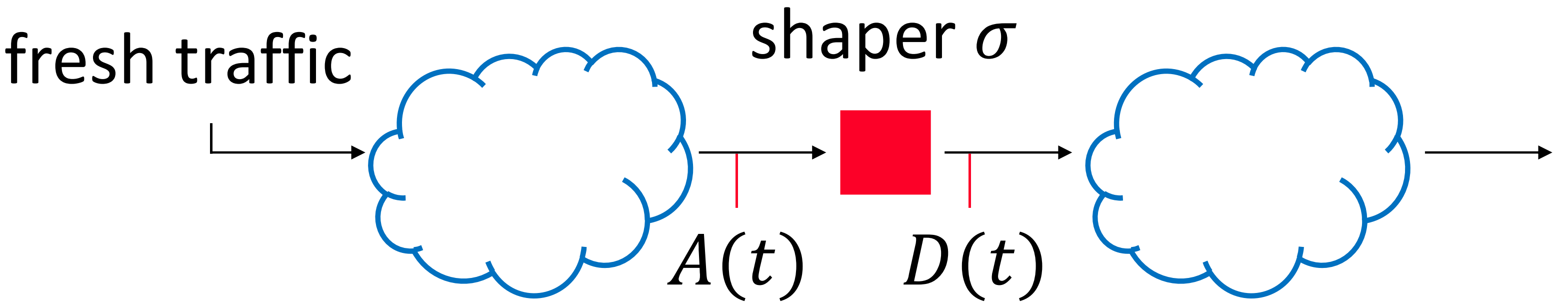


Burstiness increases as flows traverse network elements

Shapers are used to reduce burstiness

Example: leaky bucket shaper (r, b) releases a packet only if there is space to put an equivalent amount of fluid into bucket

The Mathematics of Shapers



A shaper

forces output to be constrained by arrival curve $\sigma()$

stores data in a buffer if needed

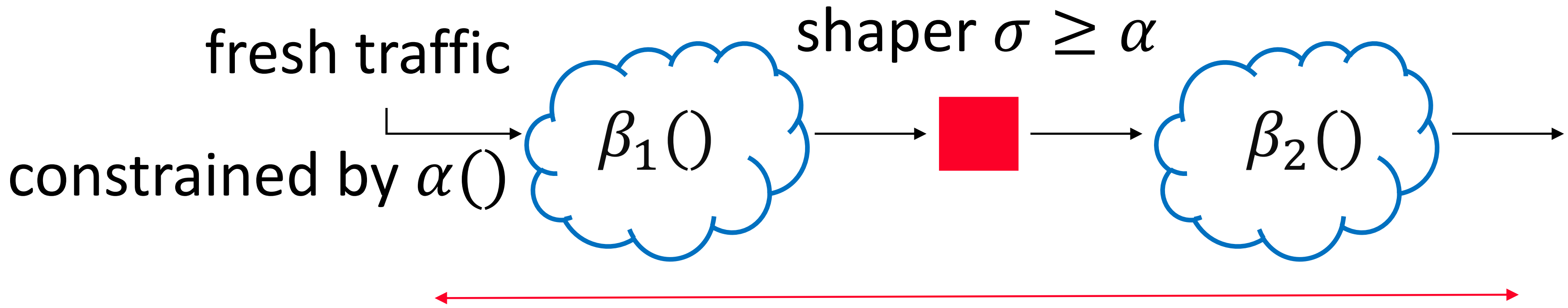
Leaky bucket shaper: $\sigma(t) = rt + b$

Output of shaper is $D(t) = (\sigma \otimes A)(t)$

$\Rightarrow$ Shaper is a service curve element with $\beta() = \sigma()$

Properties of Shapers

Re-shaping does not increase worst-case end-to-end delay



same end-to-end delay bound with or without shaper

with shaper : $D' = h(\alpha, \beta_1 \otimes \sigma \otimes \beta_2)$

without shaper : $D = h(\alpha, \beta_1 \otimes \beta_2)$

$$D' = h(\alpha, \beta_1 \otimes \sigma \otimes \beta_2) = h(\alpha, \sigma \otimes \beta_1 \otimes \beta_2) = h(\alpha, \beta_1 \otimes \beta_2) = D$$

Other Bells and Whistles

Variable and Fixed Delays

can be handled separately

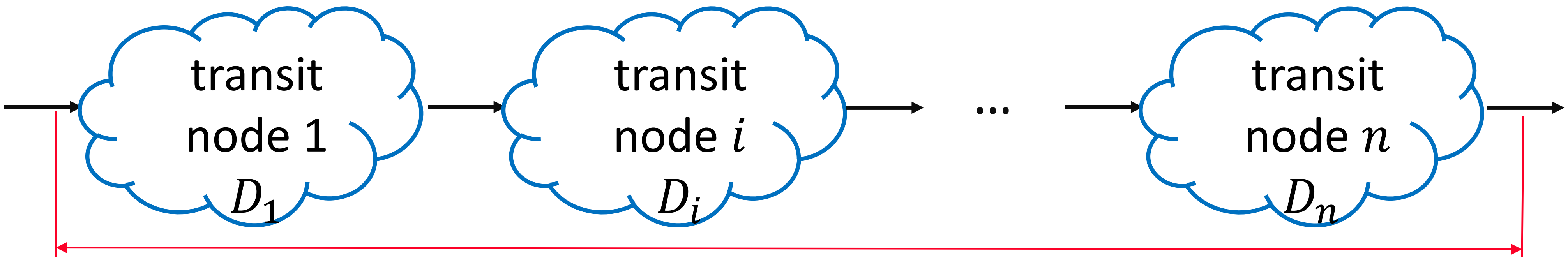
fixed delays can be excluded from service curves [Le Boudec-Thiran 2001, Section 1.6.3]

Packetization Delays

[Le Boudec-Thiran 2001, Section 1.7.2]

Implications for Path Computation

In TSN / SRP, end-to-end delay bound is sum of local delay-bounds computed at every node.



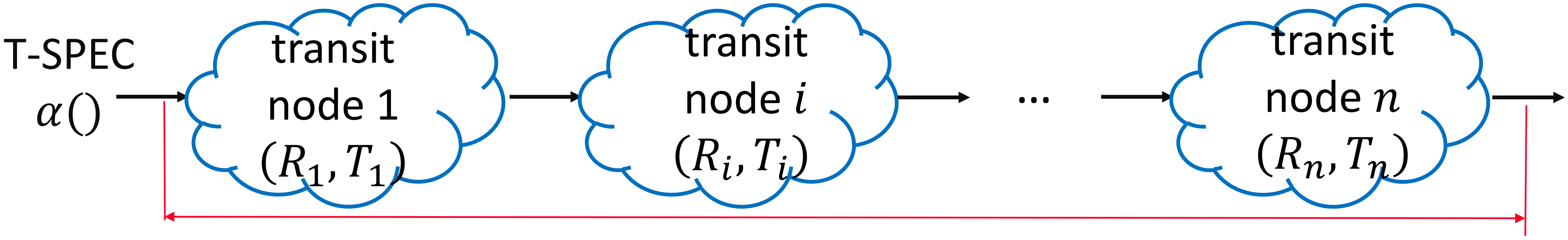
path is accepted if $\sum_i D_i \leq D_{\text{target}}$

This may be suboptimal because of "Pay Bursts Only Once". The end-to-end delay bound may be smaller than $\sum_i D_i$.

Alternative for Path Computation

Assume every transit nodes exports to PCE / SRP a description of a service curve it can guarantee to this flow.

For example, here, using a rate-latency service curve (R_i, T_i)



path is accepted if $h\left(\alpha, \min_i R_i, \sum_i T_i\right) \leq D_{\text{target}}$

The improvement on delay bound is:

(delay due to burstiness + re-shaping delay) $\times (n-1)$

A Simple Distributed Path Setup Procedure

Assume path is pre-computed (e.g. by with Widest Path routing).

Path setup message from source contains TSPEC + an object for accumulated service curve e.g. accSerCur = (type='rate-latency', R, T)

Say node i on the path accepts reservation and agrees to offer a rate-latency service curve with parameters (R_i, T_i). This node updates accSerCur in path setup message as:

accSerCur.R = min (accSerCur.R, R_i)

accSerCur.T = accSerCur.T $+ T_i$

Destination receives the proposed end-to-end service curve and T-SPEC and computes accurate end-to-end delay bound.

Centralized Joint Path Selection and Setup

Central PCE can compute a path and reserve resources in one shot.

Problem is : given a TSPEC and a delay bound D_{target} find a path and the service curve elements at every node on the path such that the end-to-end delay bound is $\leq D_{\text{target}}$.

[Frangioni et al 2014]: assume arrival curve is affine, service curves are rate-latency with linear dependence of latency on rate.
The problem is NP-hard but can be cast as a Mixed-Integer Second Order Cone Program (MISOCP), which can be solved efficiently in real-time.

Conclusions

Network Calculus can

- help understand some physical properties of Deterministic Networking (e.g. pay bursts only once, reshaping does not increase end-to-end delay bound),
- simplify end-to-end computations using simple abstractions,
- provide formal guarantees on extreme delays that are hard to reach by simulation or by ad-hoc analysis,
- provide a simple language to abstract a DetNet node without prescribing an implementation.

Future Work ?

Obtain service curve characterization of TSN/other schedulers and shapers.

Formally prove end-to-end bounds.

Quantify of improvement to end-to-end delay-bounds by exporting service curves instead of per-node delay-bounds.

Explore implications for path computation and setup (distributed, centralized).

Propose and test abstract node models.

References

Textbook: [Le Boudec-Thiran 2001] Le Boudec, Jean-Yves, and Patrick Thiran. Network calculus: a theory of deterministic queuing systems for the internet. Vol. 2050. Springer Science & Business Media, 2001, legally and freely available online at http://ica1www.epfl.ch/PS_files/NetCal.htm

Ultra-short tutorial: [Le Boudec-Thiran 2000] Le Boudec, J-Y., and Patrick Thiran. "A short tutorial on network calculus. I. Fundamental bounds in communication networks." *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*. Vol. 4. IEEE, 2000, also at http://www.cse.cuhk.edu.hk/~cslui/CSC6480/nc_intro1.pdf

Path Computation [Frangioni et al 2014] Frangioni, A., Galli, L., & Stea, G. (2014). Optimal joint path computation and rate allocation for real-time traffic. *The Computer Journal*, *58*(6), 1416-1430.

References

Avionics: [Boyer et al 2012b] Boyer, Marc, Nicolas Navet, and Marc Fumey. "Experimental assessment of timing verification techniques for AFDX." *6th European Congress on Embedded Real Time Software and Systems*. 2012.

Automotive: [Queck 2012] Queck, Rene. "Analysis of ethernet avb for automotive networks using network calculus." *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on*. IEEE, 2012.

Worst case bounds for class based queuing: [Bondorf et al, 2017], Bondorf, Steffen, Paul Nikolaus, and Jens B. Schmitt. "Quality and Cost of Deterministic Network Calculus: Design and Evaluation of an Accurate and Fast Analysis." *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1.1 (2017) and arXiv:1603.02094v3