Performance evaluation of real-time traffic in aggregate-scheduling networks

Giovanni Stea

Computer Networking Group
Department of Information Engineering
University of Pisa, Italy

UNIVERSITÀ DI PISA

Computer Networking Group
Architectures, Protocols and Performance Evaluation

Melbourne, AU, August 2009

My coauthors

- Joint work with colleagues @ CNG, Pisa
  - Luca Bisti, Luciano Lenzini, Enzo Mingozzi
- Research activity started in 2003
  - 7 conference papers, 4 journal papers

Outline

- Introduction
  - Real-time applications and e2e delay bounds
  - Aggregation in the Internet
  - Problem statement and previous work
- Our method
  - Network Calculus (brief intro)
  - The Least Upper Delay Bound methodology
  - Evaluation: scalability, accuracy
- Conclusions

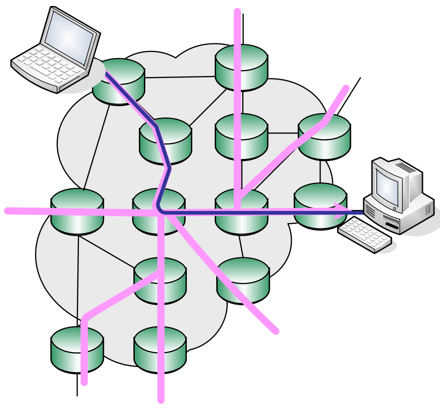
Real-time traffic

- Expected to represent the bulk of the traffic in the Internet soon
  - Skype users: 10^7-10^8
  - Cisco white paper: video traffic volume to surpass P2P in 2010
- Revenue-generating only if reliable
- Reliability boils down to “packets meeting deadlines”
  - End-to-end delay bounds are required

Traffic aggregation

- Aggregation as "the" solution for <u>scalable</u> provisioning of QoS in core networks
- Internet:
  - **Differentiated Services**
  - **MPLS**

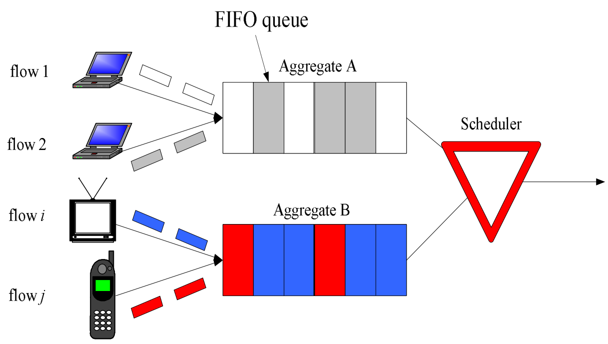
Per-flow resource management (e.g., packet scheduling) <u>just doesn't scale</u>



Dipartimento di Ingegneria della Informazione

Per-aggregate scheduling

- Packets of an aggregate normally queued FIFO
- Arbitration (scheduling) **among** aggregates
  - Forwarding guarantees **for the aggregate**

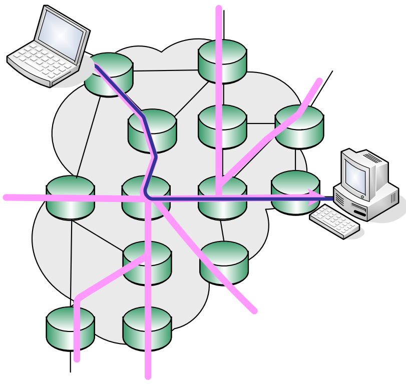


Dipartimento di Ingegneria della Informazione

Dipartimento di Ingegneria della Informazione

Traffic aggregation (cont.)

- Aggregates change along a path
  - Flows join and leave at the nodes
  - Per-node guarantees for **different sets of flows** at each node



Dipartimento di Ingegneria della Informazione

Performance evaluation problem

- Users care about *<u>their flows</u>*, not *aggregates*
- Users want *<u>e2e delay bounds</u>*, not *per-node forwarding guarantees*

How to compute
per-flow **end-to-end** delay bounds
from ***per-aggregate*** **per-node** guarantees?

Building on the e2e delay bound

Good **end-to-end delay bounds**
for **single flows**
under aggregate scheduling

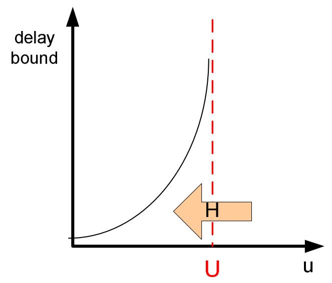
- Admission control
- Optimal aggregation criteria
- Manage Service Level Specifications
- Provisioning and traffic engineering

for real-time traffic

Theoretical background

- Le Boudec, Charny (2001)
  - Generic FIFO network, H hops
  - A delay bound exists
    - For small utilization factors: $u < U = \frac{1}{H-1}$
    - proportional to

$$\frac{1}{1-u/U}$$



  - Better bounds **require more hypotheses**

Our approach

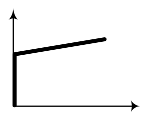
Dipartimento di Ingegneria della Informazione

- More hypotheses
  - An administrator *does* know the topology of its network
  - Networks are often feed forward
    - Stable up to 100% utilization
  - Compute **better** bounds in practical scenarios
- Tool used: Network Calculus
  - Won't use formulas (promise!)

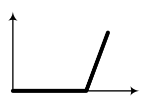
Crash course in Network Calculus (1)

Dipartimento di Ingegneria della Informazione

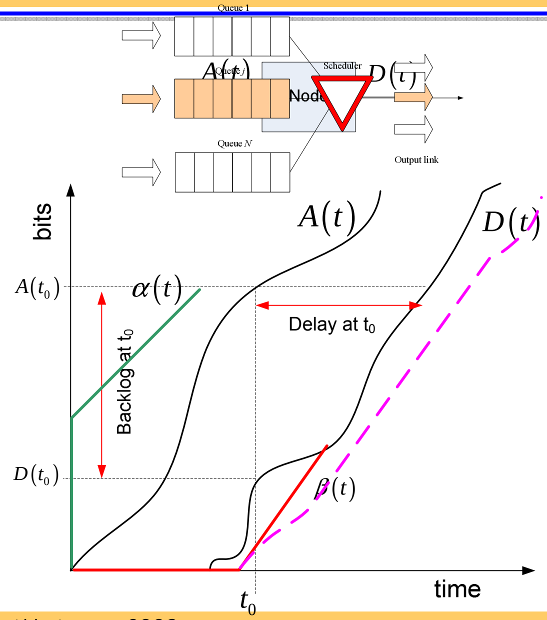
- Model network entities as functions of time or operators on these
  - **Traffic shaper → Arrival curve**
    - Upper bound on the arrivals of a flow
    - E.g., a leaky-bucket



  - **Scheduler → Service curve**
    - Lower bound on the departures of a flow
    - Per-node forwarding guarantee
    - E.g., a Fair Queuing or Priority scheduler

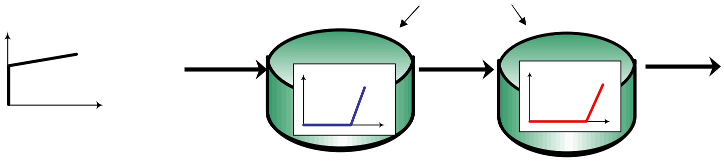


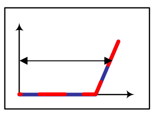
Crash Course in Network Calculus (2)



Crash Course in Network Calculus (3)

Both SCs offered
to the **same flow**



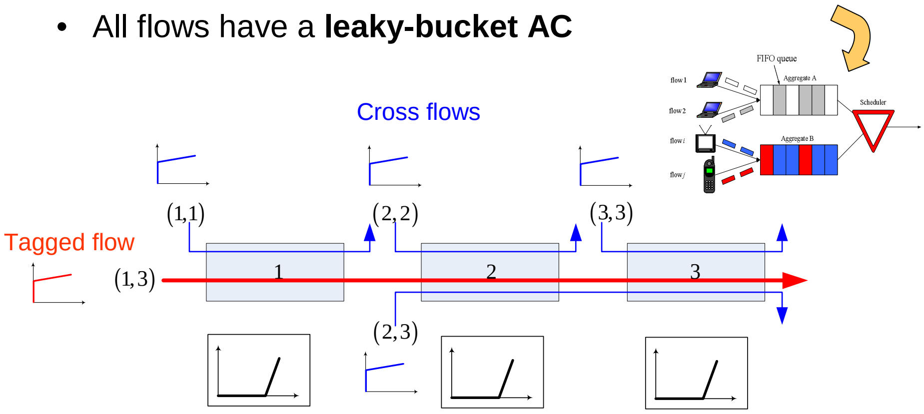


Min,+ convolution ⊗

e2e delay bound

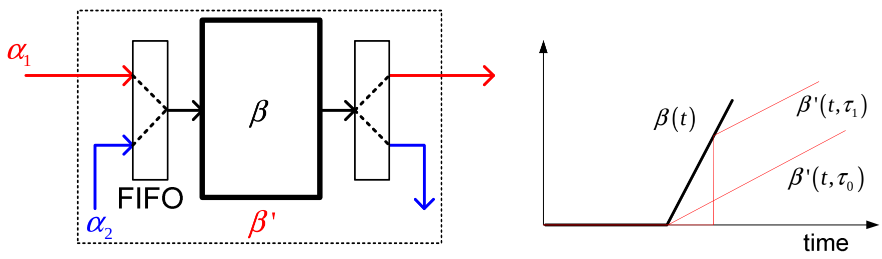
Back to our problem – system model

- **Tandem network** of FIFO rate-latency elements
- All nodes have a **rate-latency SC** for the aggregate
- All flows have a **leaky-bucket AC**

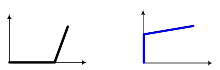


The starting point

- Th. **FIFO Mux - Minimum Service Curves** [Cruz, '98]



$$\beta'(t,\tau)=\left[\beta(t)-\alpha_2(t-\tau)\right]^+\cdot 1_{\{t>\tau\}},\ \tau\ge 0$$

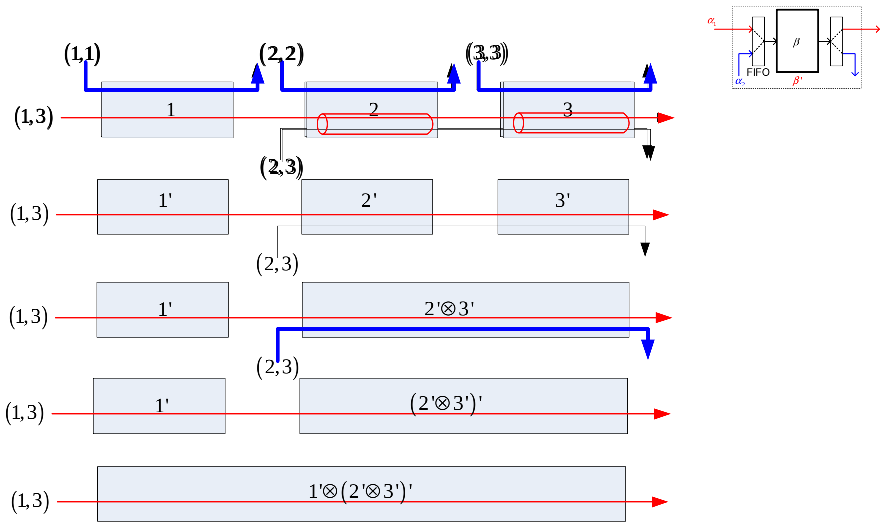


The LUDB methodology

- LUDB: Least Upper Delay Bound

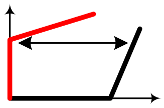
- <u>Step 1:</u>
  - Apply the FIFO Mux theorem iteratively so as to "remove" all cross-flows

The LUDB methodology (2)



The LUDB methodology (3)

- An **n-dimensional infinity** of e2e SCs for the tagged flow
  - n = # of cross-flows
- Delay bound = fn. of n parameters



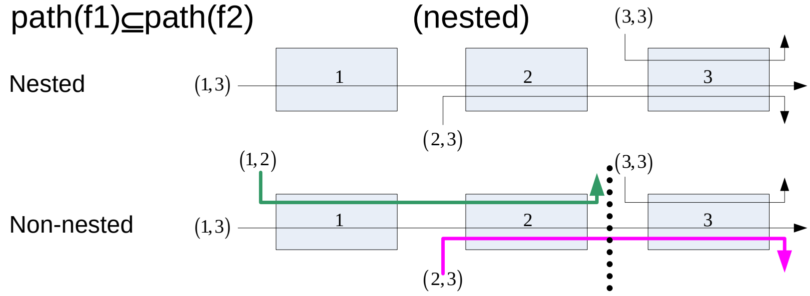
- Step 2
  - Solve an **optimization problem**

$$LUDB = \min_{\tau_i \geq 0} \left\{ DB\left(\tau_1, \ldots, \tau_n\right) \right\}$$

  - The minimum is the **best** delay bound

Nested vs. non-nested tandems

- **Nested iff**
  - path(f1)∩path(f2) =∅ (disjoint)
  - path(f1)⊆path(f2) (nested)



  - You can **only** compute an e2e SC for the tagged flow in a **nested tandem**

Two important points

- We show that this method:
  1. Is scalable enough
     - You can use it in paths of 30+ nodes
  2. Yields accurate bounds
     - Close to a flow's **Worst-Case Delay**
     - (Sometimes)
- We start from nested tandem, then move to non-nested ones

Solving the LUDB problem

- Nested tandems

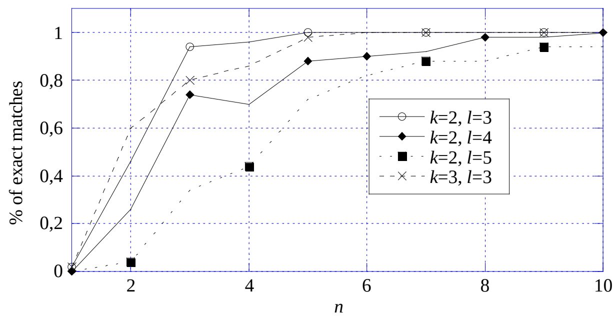
$$LUDB = \min_{\tau_i \geq 0} \left\{ DB(\tau_1, \ldots, \tau_n) \right\}$$

  - **Piecewise linear** optimization problem
    - Objective function is piecewise linear
    - Constraints are linear
  - We don't know it to be convex
  - Equivalent to (a large number of) **simplexes**

Scalability: nested tandems

Dipartimento di Ingegneria della Informazione

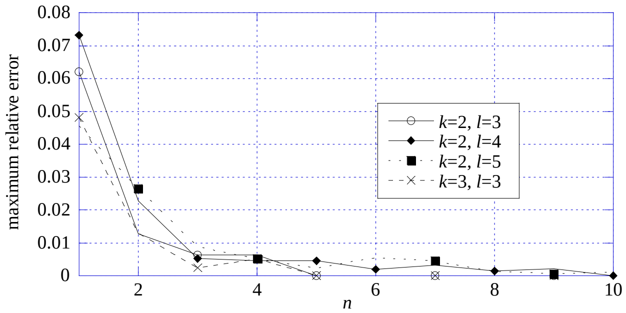
- 30 nodes: hours, too many simplexes
- Heuristic: try to figure out which (few) simplexes yield the smallest delay bound



Scalability: nested tandems

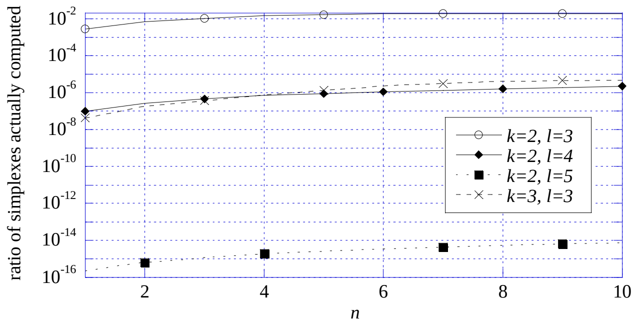
Dipartimento di Ingegneria della Informazione

- 30 nodes: hours, too many simplexes
- Heuristic: try to figure out which (few) simplexes yield the smallest delay bound



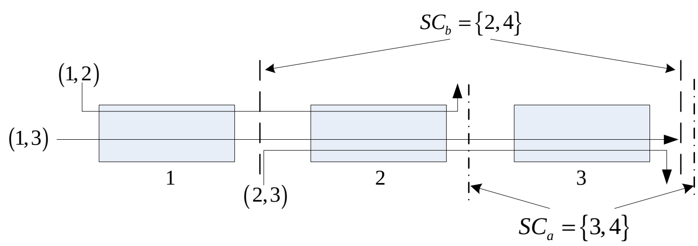
Scalability: nested tandems

- 30 nodes: hours, too many simplexes
- Heuristic: try to figure out which (few) simplexes yield the smallest delay bound



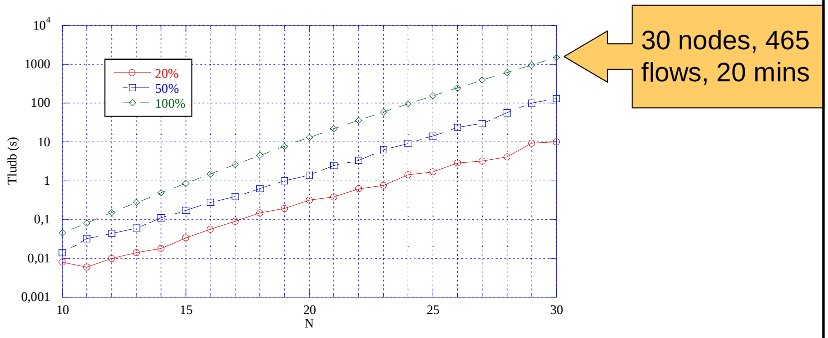
Scalability: non-nested tandems

- Cross-flows are inter-dependent
  - **Cut** the tandem into *nested* sub-tandems
    - Compute the LUDB for the 1st sub-tandem
    - Move on to the 2nd sub-tandem and do the same
    - **Sum** all partial (per sub-tandem) delay bounds
- Problem: many ways to cut a tandem



Scalability: non-nested tandems (2)

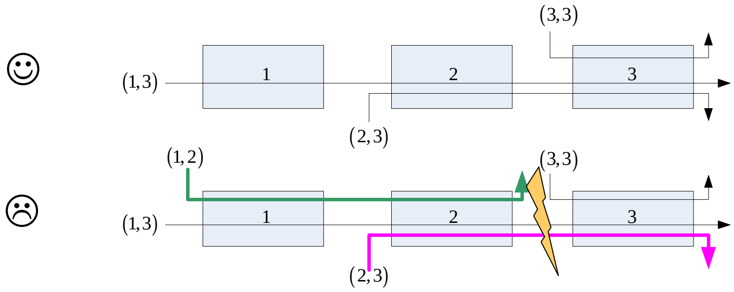
- Identify only **good** cuts
  - Cache and reuse partial LUDB computations



  - Computation times for **very nasty** non-nested tandems

Accuracy

- Delay bounds are as useful as they are **tight**, i.e. close to the Worst-Case Delay
  - WCD unknown (to date)
- End-to-end analysis is fundamental

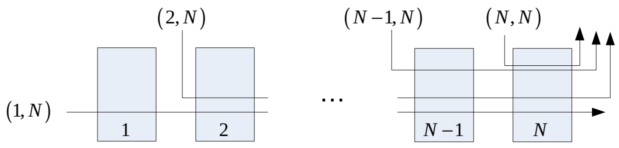


Accuracy (2)

- e2e analysis is however **not sufficient**
  - **Sink trees**: nested tandems, all paths are (j,N)

WCD = LUDB

  - Tune flows so that a bit experiences a delay = **LUDB**



  - However, in other nested tandems

WCD < LUDB

Assessing the accuracy

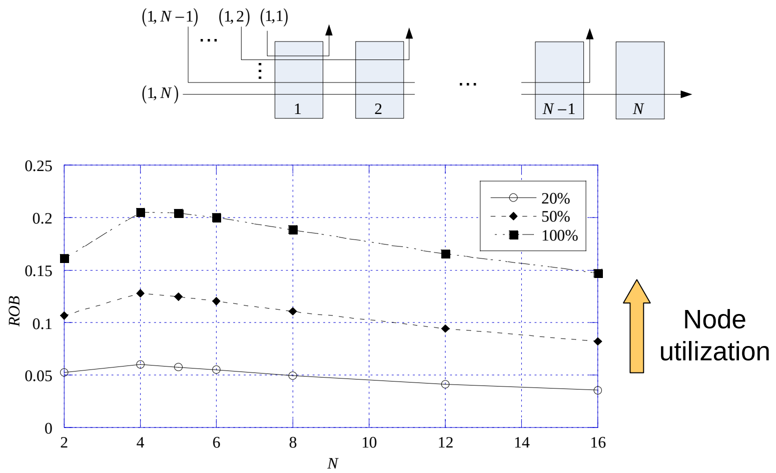
- Compute a (heuristic) <u>**lower**</u> bound
  - Tune flows carefully so as to “maximize” delay
- Compare the lower bound and the LUDB

$$WCD \in [LowerBound, LUDB]$$

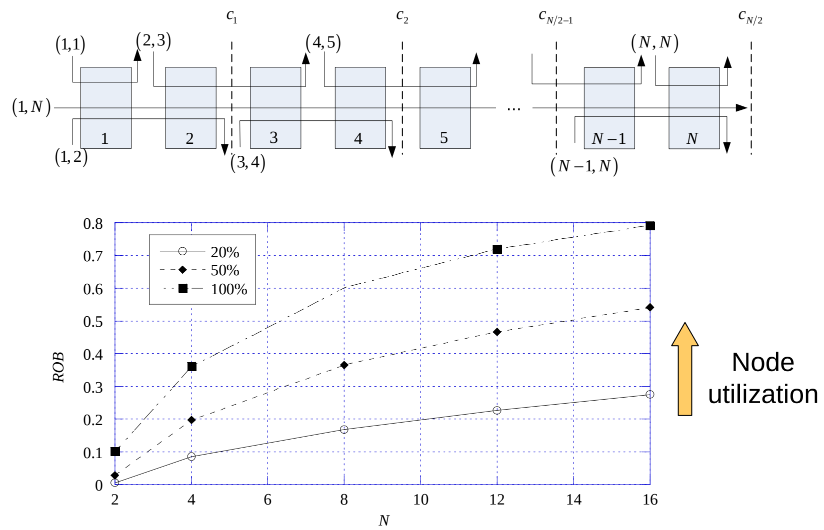
  - The smaller the interval, the tighter the bound
- Relative Overrating Bound (ROB):

1 - LowerBound/LUDB

Case Studies – nested tandems



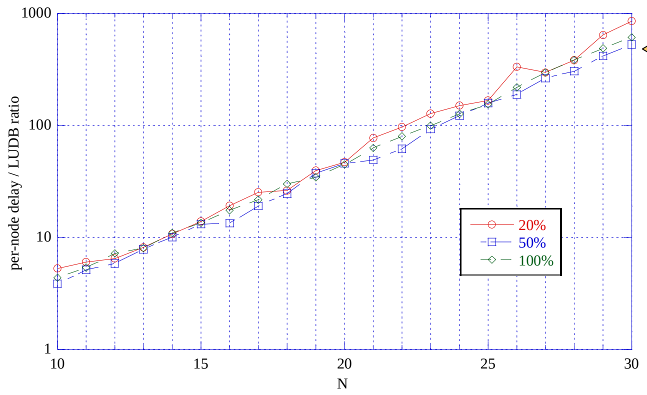
Case studies: non-nested tandems



Dipartimento di Ingegneria della Informazione

Still better than the rest

- Compared to (naïve) per-node analysis



500-1000
times smaller

Dipartimento di Ingegneria della Informazione

A publicly available tool

- DEBORAH
  - DElay BOund Rating AlgoritHm
  - Includes
    - LUDB computation for nested and non-nested tandems
    - Heuristics for LUDB computation
    - Computation of the *lower* bounds for comparison
    - C++, source code available
- http://cng1.iet.unipi.it/wiki/index.php/Deborah

Dipartimento di Ingegneria della Informazione

Conclusions

- A method for computing end-to-end delay bounds in FIFO tandem networks
- The method is
  - **Practicable**:
    - analysis of edge-to-edge paths in today ISP's networks requires seconds to minutes
  - **Accurate**:
    - Definitely so in nested tandems
    - Not so much in non-nested tandems

Dipartimento di Ingegneria della Informazione

(some) References

1. L. Bisti, L. Lenzini, E. Mingozzi, G. Stea, "Computation and Tightness Assessment of End-to-end Delay Bounds in FIFO Tandems Using Network Calculus", submitted to Elsevier Performance Evaluation, 2008
2. L. Lenzini, E. Mingozzi, G. Stea, "A Methodology for Computing End-to-end Delay Bounds in FIFO-multiplexing Tandems" Elsevier Performance Evaluation, 65 (2008) 922-943
3. L. Lenzini, L. Martorini, E. Mingozzi, G. Stea, "Tight End-to-end Per-flow Delay Bounds in FIFO Multiplexing Sink-tree Networks", Elsevier Performance Evaluation, Vol. 63, Issues 9-10, October 2006, pp. 956-987
4. L. Lenzini, E. Mingozzi, G. Stea, "Delay Bounds for FIFO Aggregates: a Case Study", Elsevier Computer Communications 28/3, February 2005, pp. 287-299
5. L. Bisti, L. Lenzini, E. Mingozzi, G. Stea, "Estimating the Worst-case Delay in FIFO Tandems Using Network Calculus", Proceedings of VALUETOOLS'08, Athens, Greece, October 21-23, 2008
6. L. Lenzini, E. Mingozzi, G. Stea, "End-to-end Delay Bounds in FIFO-multiplexing Tandems", Proc. of VALUETOOLS'07, Nantes (FR), October 23-25, 2007.
7. L. Lenzini, E. Mingozzi, G. Stea, "Delay Bounds for FIFO Aggregates: a Case Study", Proceedings of the 4th COST263 International Workshop on Quality of Future Internet Services (QoFIS '03), Stockholm, Sweden, October 1-3, 2003 LNCS 2811/2003, pp. 31-40

Thank you for listening

- Questions?
- Comments?

Dipartimento di Ingegneria della Informazione