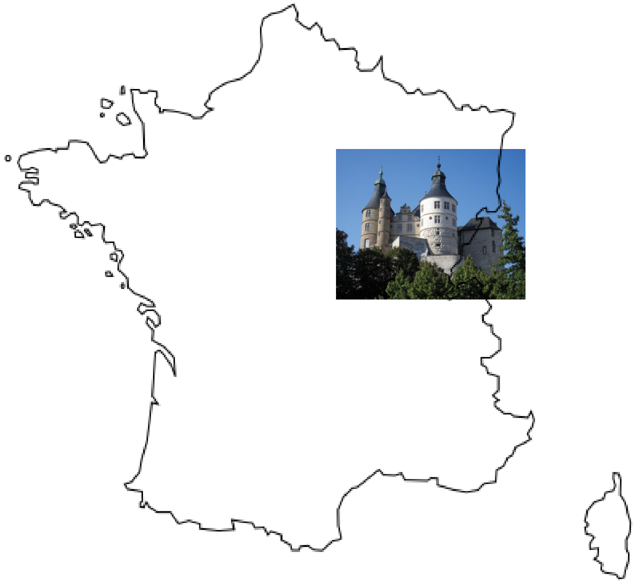
An analysis of congestion controls in centralized control systems

Eugen Dedu, Grégory Bise, Julien Bourgeois
FEMTO-ST, DISC department, OMNI team
Montbéliard, France

NetGCooP conference
Avignon, France
30 Nov. 2012



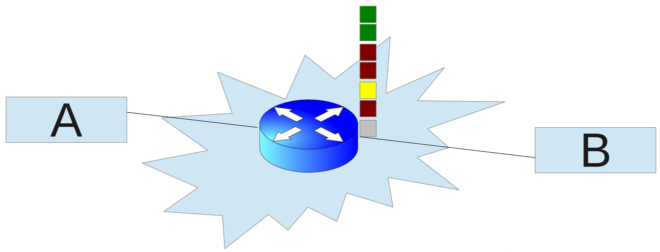
Background on CS

- Control systems (CS): devices which manage the behaviour of systems
  - open-loop control: controller and actuator
  - **closed-loop control**: sensor, controller and actuator (+ feedback)
  - **centralised**: one controller
  - decentralised: several controllers
  - **networked**: sensors and actuators are connected through a network

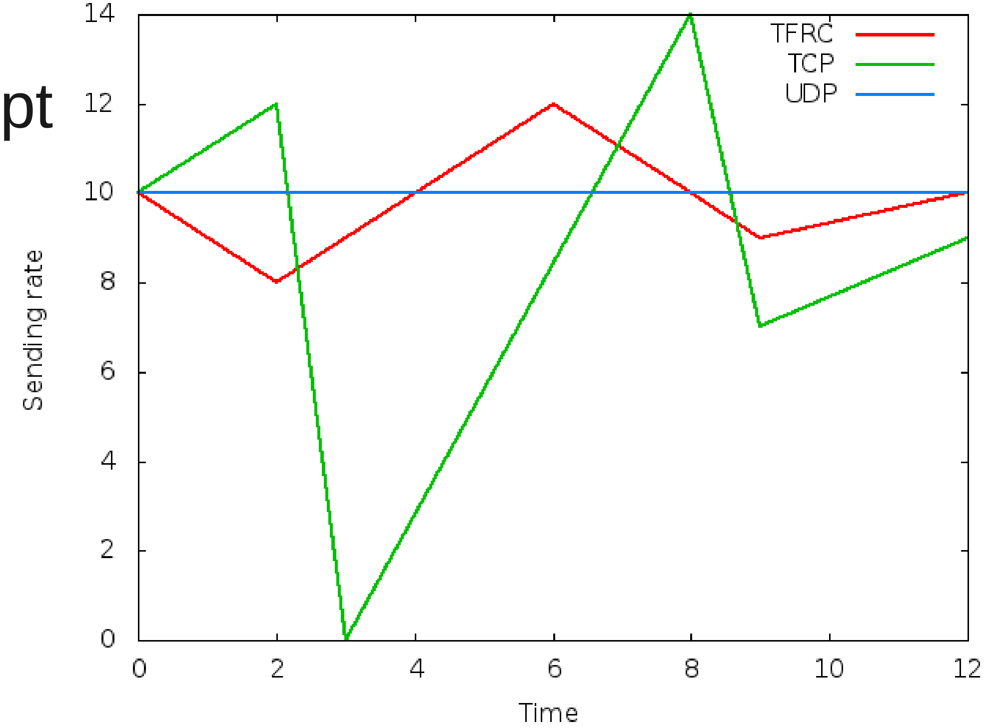
Brakes: Foot -> wheel

Radiator: Temp sensor -> Controller -> temp update
Manned plane control
Nuclear plants
etc.

Background on CC



- Network can be congested if packet rate exceeds network bandwidth => lost packets
- Congestion control (CC) aims to adapt sending rate to network
- CC of TCP: window-based, abrupt changes, 100% reliability
- TFRC: equation-based, smooth changes
- DCCP, can choose a CC, no reliability
- UDP, no CC



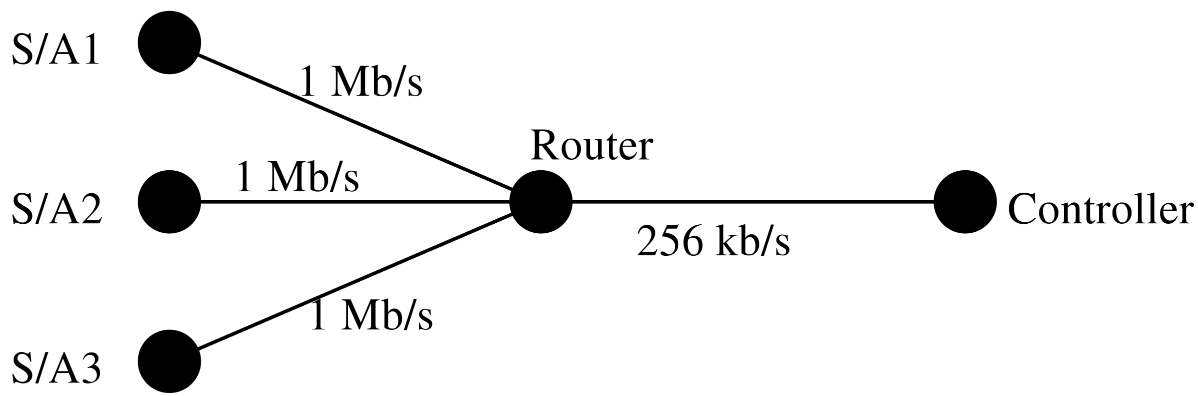
Motivations

- TCP already deeply analysed in Internet context
- Control systems generally take into account physical layer, so do not cope with congestion in network
- Intersection of the two: How current congestion control algorithms work in control system constraints?
- Some differences between Internet and CS:
  - network size/complexity: very complex for Internet, simple for CS
  - **data generation**: some data needs to be transported, users add randomly new data to be sent for Internet, regular data sending for CS
  - etc.
- **In this talk we analyse various CC results from centralised CS with regular data sending**
  - we do not try to improve them
- We use simulations: TrueTime (in CS community) vs **ns2** (in network community)
  - no modelisation

Plan

- Network topology used
- Simulation results for various congestion controls:
  - TCP
  - DCCP/TFRC
  - DCCP/TCP-like
  - UDP
- Discussion
- Conclusions

Network topology used

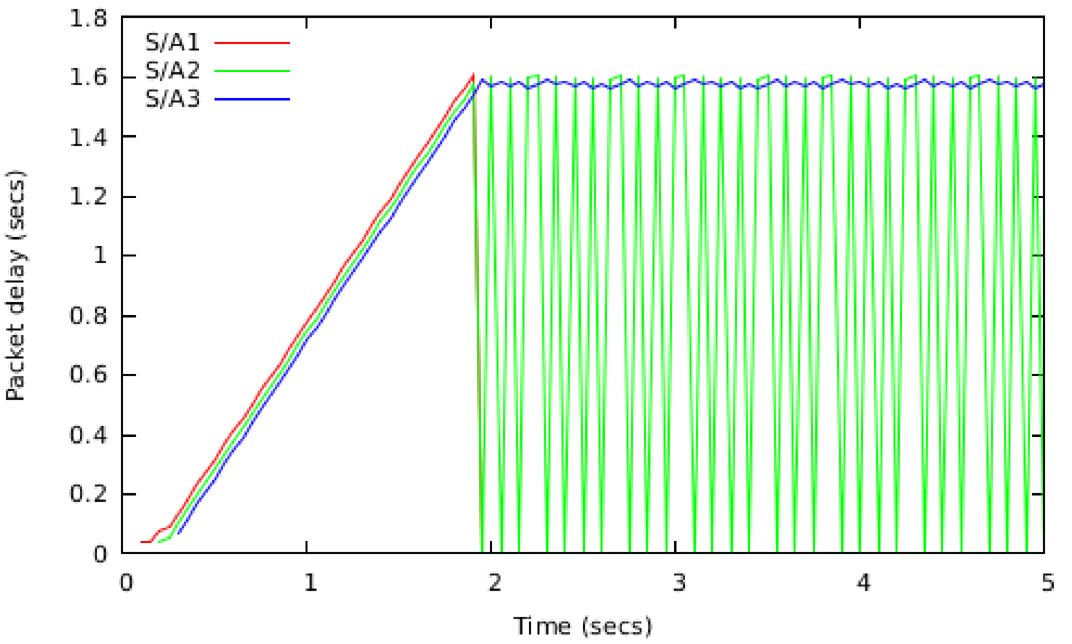


- Each S/A sends to Controller 1 packet of 1024 bytes of data each 50 ms
- Controller answers with a 200 bytes packet
- Router uses DropTail (when queue is filled, drop packets)
- => Congestion on right link from Router to Controller

  (~20 kB/s * 3 = 480 kb/s generated)
- Question: how do various CC cope with it (latency, throughput)?

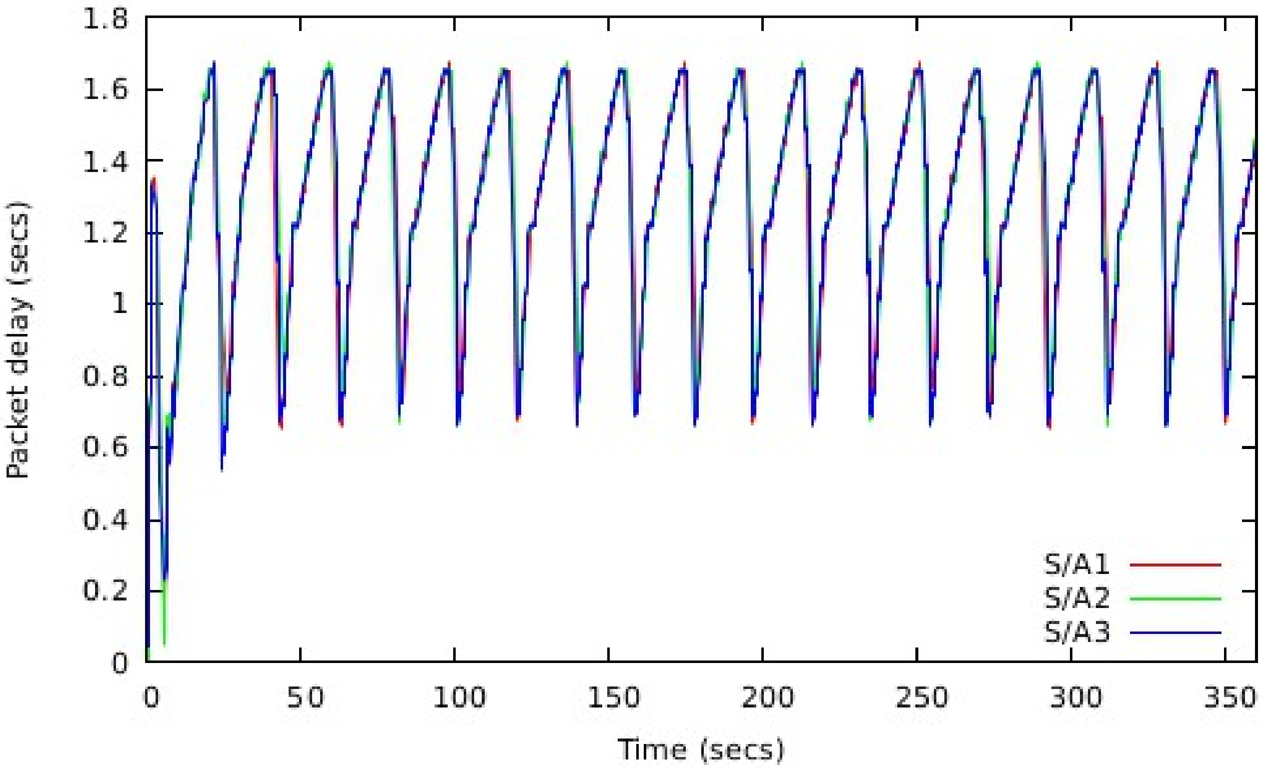
UDP results

- For t <= 2s, all packets arrive and delay of each S/A increases
- Unfairness:
  - S/A1 loses all packets
  - S/A2 switches between 0 and 1.6s
  - S/A3 has 1.6s delay
  - timing issue, two causes:
    - DropTail and UDP



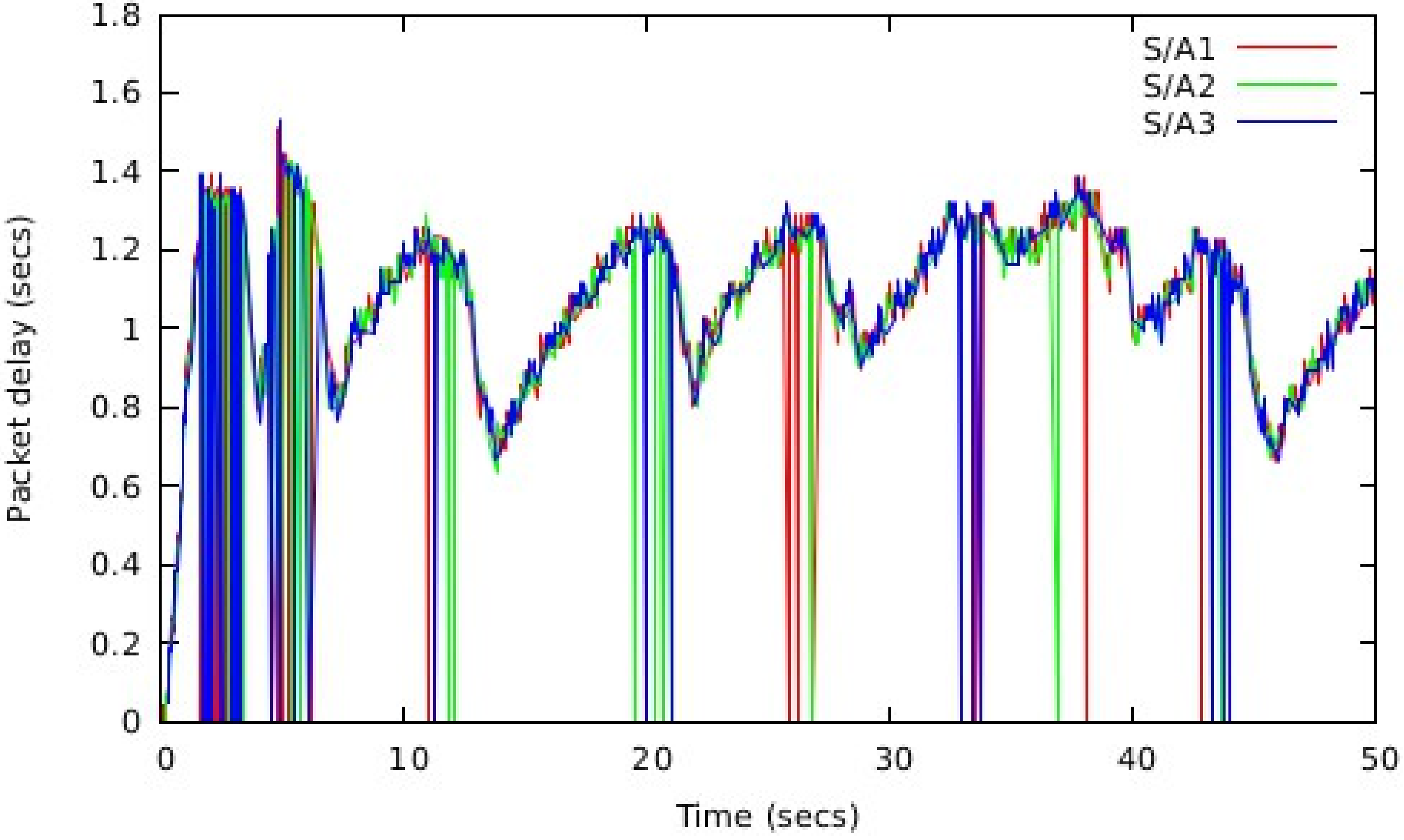
TCP results

- Fair balancing
- Well-known saw teeth-like curve can be seen



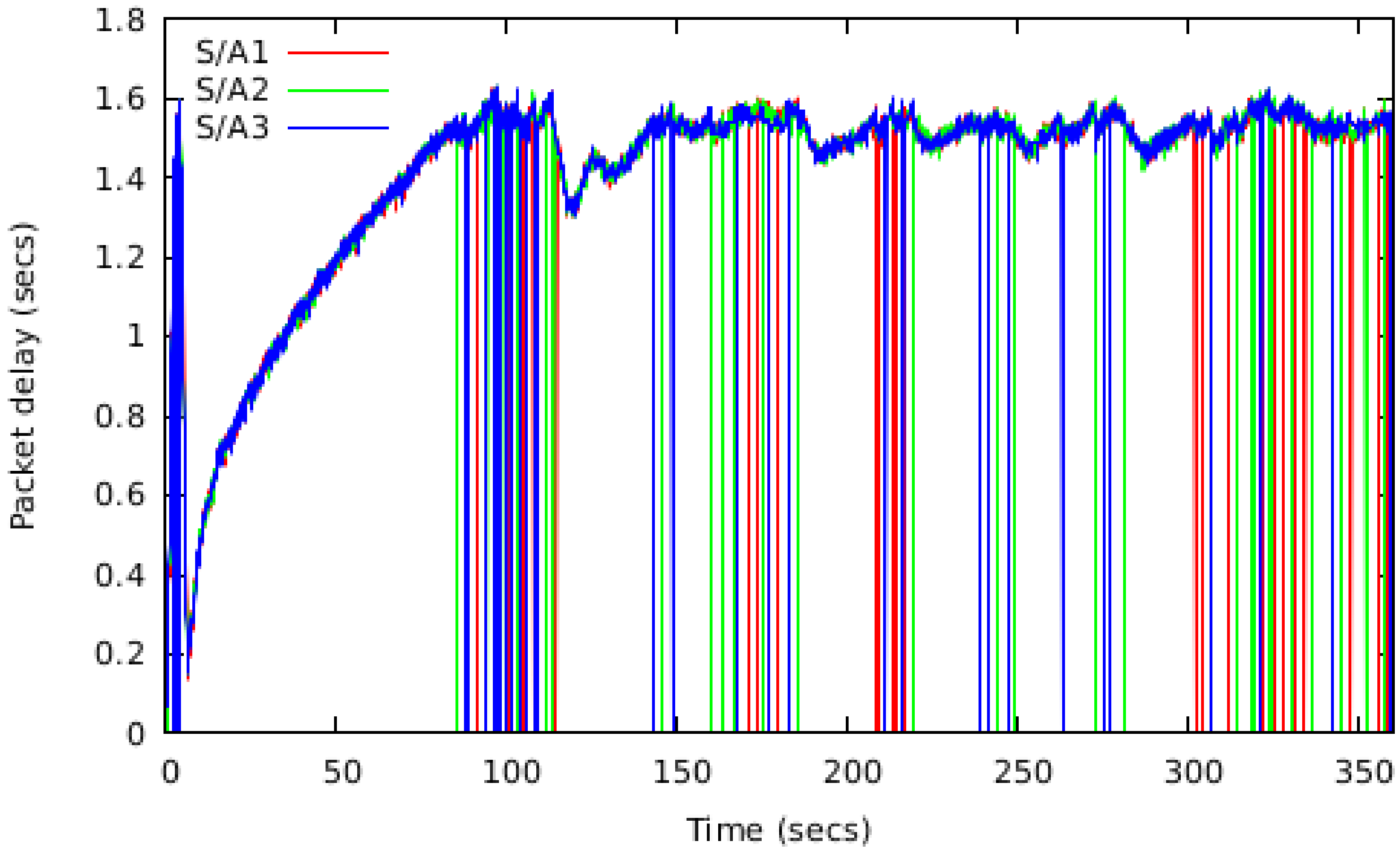
DCCP/TCP-like results

- Shape similar to TCP, as expected

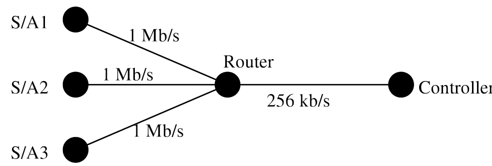


DCCP/TFRC results

- Smoother than TCP, as expected



Further results + Discussion



Protocol	Sensor	Packets				Delay	
		generated	lost on sensor	lost on network	received	highest	average
UDP	1	7199	0	7163	36	1.61	1.35
	2	7197	0	3432	3765		
	3	7195	0	0	7195		
TCP	1	7199	4425	0 (26 retr)	2774	1.68	1.38
	2	7197	4380	0 (26 retr)	2817		
	3	7195	1577	0 (11 retr)	5618		
DCCP/TCP-like	1	7199	2276	115	4808	1.53	1.09
	2	7197	2510	108	4579		
	3	7195	2235	124	4836		
DCCP/TFRC	1	7199	3496	60	3643	1.63	1.41
	2	7197	3184	54	3959		
	3	7195	3193	60	3942		

- Congestion => lost packets
  - on network: when no CC (UDP)
  - on sensor: when CC (TCP, DCCP), because data generation higher than available bw
- Type of CC influences delay: TCP-like maintains lower queue filling
- All CC received similar nb of packets: CC smooths data, but has no effect when data is generated regularly (similar to video streaming)!

Conclusions

- UDP has crucial synchronisation issues, solvable using a mechanism other than DropTail
- In terms of data received, no CC is definitely better, and CC = without CC
- In terms of delay, DCCP/TCP-like gives best results
- Perspectives:
  - decentralised system
  - data generation rate adaptation based on network conditions