Coupled congestion control for RTP media

draft-welzl-rmcat-coupled-cc-00 (01)

Michael Welzl
michawe@ifi.uio.no

RMCAT, 86th IETF Meeting
11. 11. 2012

Context

- Addresses "identifying and controlling groups of flows"
  - "Identifying" trivial for some rtcweb flows, else requires Shared Bottleneck Detection (SBD) – working on it, hopefully first draft @ IETF87

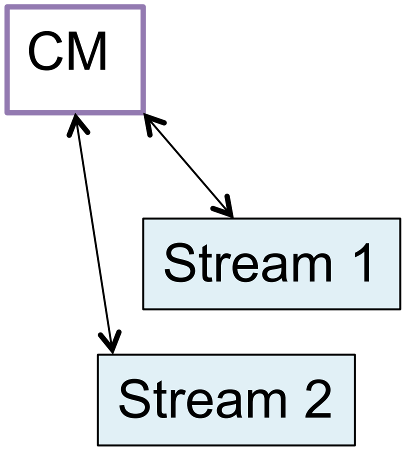
- Has been tried in the past (CM (RFC 3124), RFC 2140) – what has gone wrong?
  - Not able to address the "identifying" part
  - Too hard to implement (note: easy-to-implement, non-critical parts of RFC 2140 are actually implemented AFAIK, e.g. sshthresh caching (+ sharing?))

⇒Try to make it as easy to implement + simple as possible

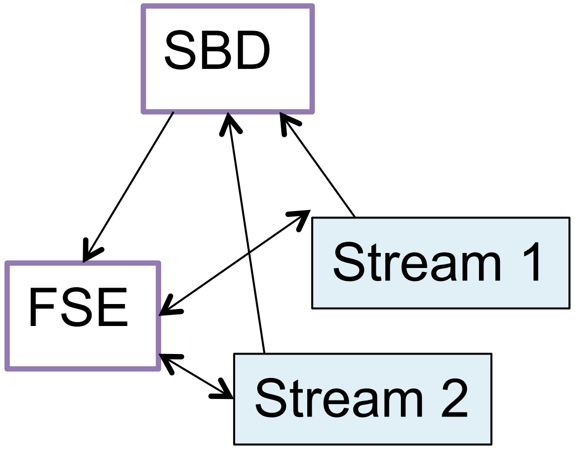
“Flow State Exchange” (FSE)

- The result of searching for minimum-necessary-standardization: passive storage, only define what goes in / out + what to do with the information
  - So far, sender-side only
  - Could reside in a single app (e.g. browser) and/or in the OS

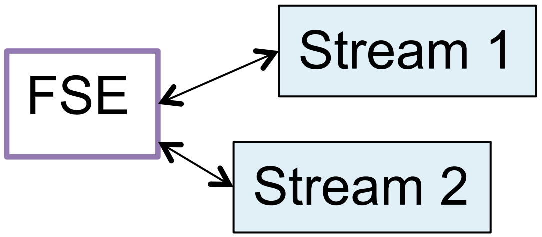
Traditional CM



FSE based flow coordination



FSE based flow coordination
For some rtcweb flows (trivial SBD)



“Flow State Exchange” (FSE) /2

- Flows update it when they start and stop
- Flows query+update whenever they update their rate
  - flow’s used rate = UPDATE(CC-calculated rate (CR), desired rate (DR))

Example state:

Stored

#	**FG**	**P**	**CR**	**DR**	**S_CR**	*Rate*
1	1	1	8	2	10	2
2	1	0.5	3	3	11	9

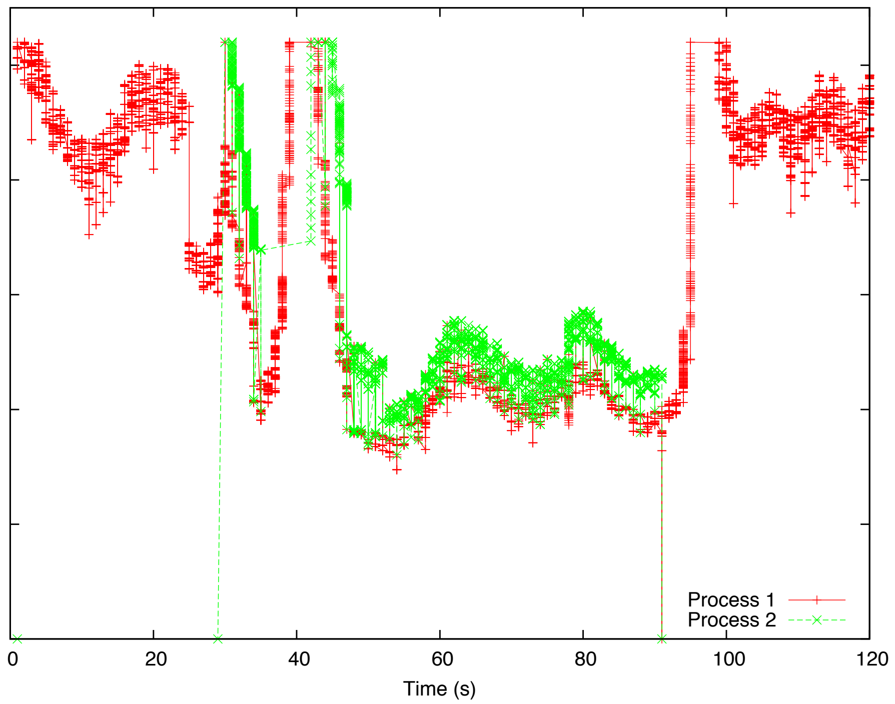
Algorithm in the draft

- Just an example – could be changed, as long as the same is used for all flows
  - Perhaps best to implement with the FSE, not with each flow

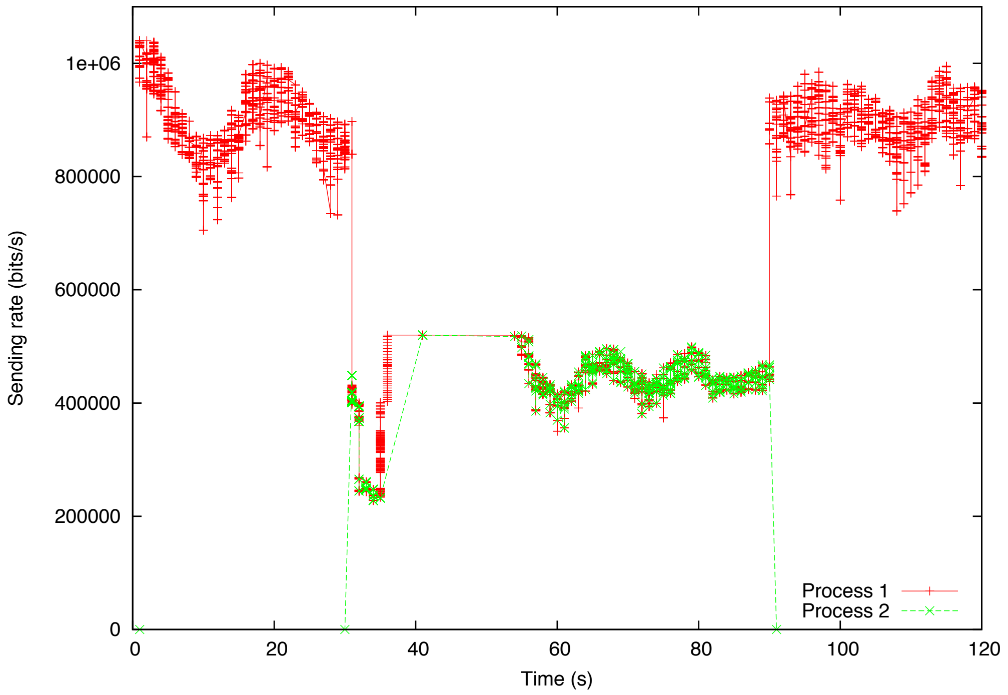
- Goals of the example algorithm:
  - Realize fairness with priorities
  - Good capacity usage: always use all the available bandwidth that congestion controls have found
  - Reduce delay: N flows should not probe N times
  - Let greedy flows immediately use unused bandwidth of non-greedy or terminated ones

Example using older variant of the alg.: 2 vic instances w / TFRC + FSE

Without



With



Current state of things

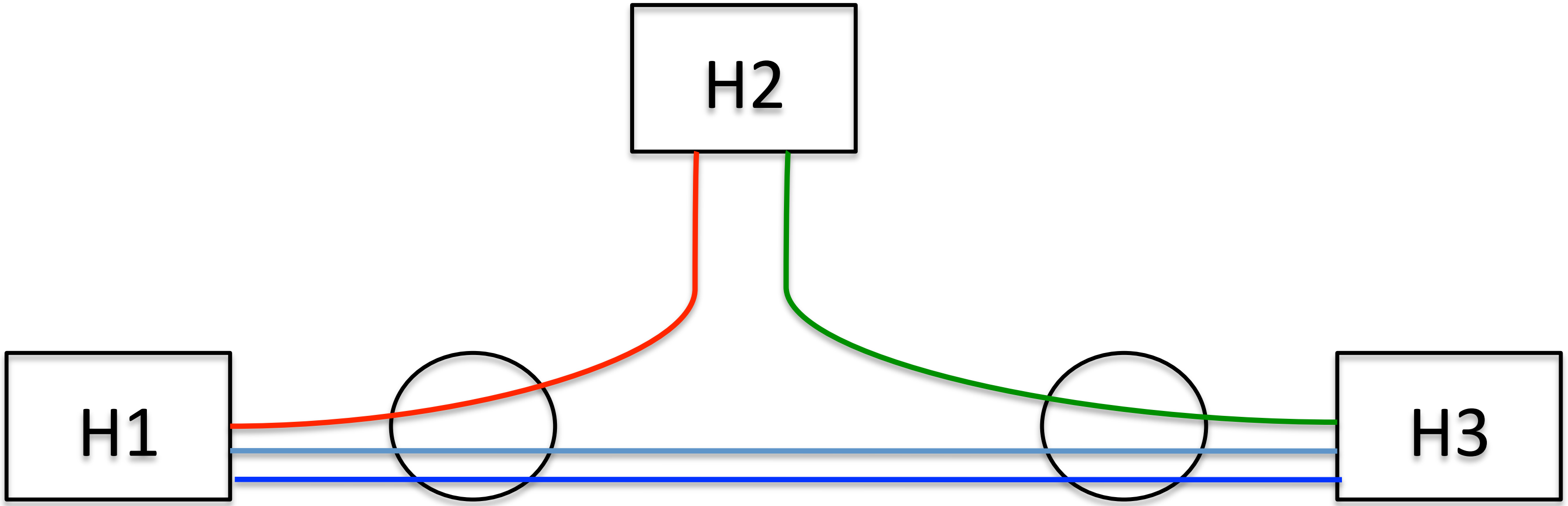
- Playing with the current algorithm
  - Working on 2 main FSE problems:

1. A flow is told to use a rate that is not what the congestion controller has determined
   - Smaller: can cc. cope with non-greedy sources? (note: on-the-wire effect probably close to greedy)
   - Larger: really unusual

   ⇒ Need to check congestion controls one by one
2. Problems could arise when flows are highly asynchronous (async. RTTs, ..)
   - Negative impact can certainly be bounded

Really only on the sender side?

- Reducing feedback frequency would require a message to the receiver based on information from the FSE (similar to async. RTT)
- Shared Bottleneck Detection needs signaling of measurement results
  - A tricky problem by itself… but not fully solving it yields false negatives, which are not too problematic (limits FSE benefits)



Thank you!

Questions?