Estimation of Queueing Delay for Packet Scheduling

Johanna Antila and Marko Luoma

Networking Laboratory, Helsinki University of Technology

{jmantti3, mluoma}@netlab.hut.fi

Outline

- Motivation
- Contribution
- Adaptive Scheduling
- Delay Estimators
- Simulation Results
- Conclusions
- Future Work

Motivation

- Adaptivity has become a key word in network provisioning
  - self-configurable networks
- Scheduling is a crucial component in adaptive provisioning
  - adjust the class resources dynamically
    - use of measurements

Contribution

- Starting point:
  - Service differentiation is based on DiffServ architecture
- We study delay-based adaptive provisioning
  - Delay-bounded HPD (DBHPD scheduler)
- We develop estimator algorithms for DBHPD’s delay estimation problem
- By simulations we evaluate
  - Filtering properties of the proposed estimators
    - Several traffic mixes

Adaptive Scheduling

- Conventional scheduling approaches:
  - divide the link capacity statically between the service classes, e.g. WFQ, DRR
- We base the scheduling decisions on measurements of short term and long term packet delays
- delay-bounded HPD (DBHPD) algorithm
  - provides absolute and proportional delay differentiation

Notations

- Let us use the following notations
  - δ_i = differentiation parameter
  - $d_i(m)$ = queuing delay of the m'th packet in class i
  - $w_i(m)$ = normalized head waiting time of class i when m packets have departed
  - g = constant
  - γ_i = filtering coefficient for class i
  - s_i = mean packet size of class i
  - q_i = maximum queue size for class i
  - C = link capacity

Delay-bounded HPD

- delay-bounded HPD scheduler
  - aims to provide proportional delay differentiation and a delay bound.
  - the algorithm first checks whether a packet in the best class is about to violate its deadline with the following equation:

$$t_{in} + d_{\max} < t_{curr} + t_{safe}$$

Delay-bounded HPD

- If delay violation is not occurring, the algorithm tries to maintain the delay ratios between consecutive classes
- this is achieved by selecting for transmission a packet from a backlogged class *j* with maximum normalized hybrid delay:

$$j = \arg\max(g\,\overline{d_i}(m)/\delta_i + (1-g)w_i(m)/\delta_i)$$

Delay-bounded HPD

- We have already evaluated the basic version of the DBHPD algorithm
  - comparisons with static DRR and adaptive DRR in a simple setup
  - comparison with static DRR in a network setup
- In recent work, the focus has been on queueing delay estimation for DBHPD

Delay Estimators

- Estimation of the long term delay is a crucial part of the DBHPD algorithm
  - the head-of-line packet delay and this estimated delay together determine the time-scale at which DBHPD operates
- Possible estimators
  - Simple Sum
  - EWMA variants
  - Kalman filters and filters based on neural networks

Delay Estimators

- Simple Sum estimator:
  - sum of the queueing delays divided by the departed packets
  - disadvantages
    - impossible to implement
    - ”infinite” history

$$\overline{d}_i(m) = \frac{\sum_{m=1}^{|D_i(m)|} d_i(m)}{|D_i(m)|}$$

Delay Estimators

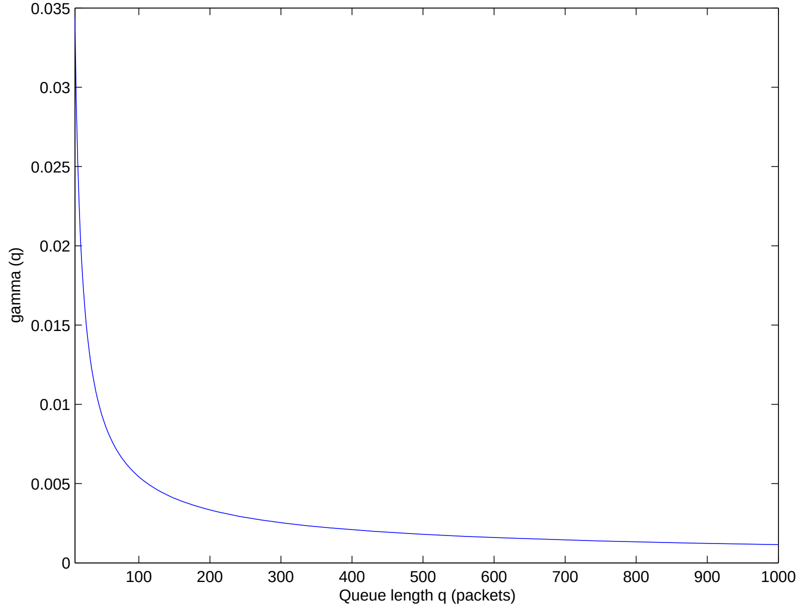
- Simple EWMA estimator
  - update long term delay with exponential smoothing
    - eliminate the overflow problem
  - traffic characteristics of the classes are very different
    - use of separate filtering coefficient for each class

$$\overline{d_i}(m) = \gamma_i d_i(m) + (1 - \gamma_i)\overline{d_i}(m-1)$$

$$\gamma_i(q_i) = \frac{1}{N * \sqrt{q_i} * \ln(q_i)}$$

Delay Estimators

- The gamma-function behaves as follows (assuming four traffic classes):



Delay Estimators

- EWMA estimator with restart (EWMA-r)
  - EWMA filter is restarted as the queue becomes active after an idle period when a certain threshold of packets have arrived in the queue
  - filter is restarted if estimator has been idle for a time

$$cycle_i = \frac{abs\_factor_i * q_i * s_i}{C}$$

Delay Estimators

- EWMA estimator based on proportional error of the estimate (EWMA-pe)
  - also the EWMA filtering coefficient is adapted
    - adaptation is based on the proportional error of the estimate

$$\overline{d_i}(m) = n * \gamma_i d_i(m) + (1 - n * \gamma_i)\overline{d_i}(m-1)$$

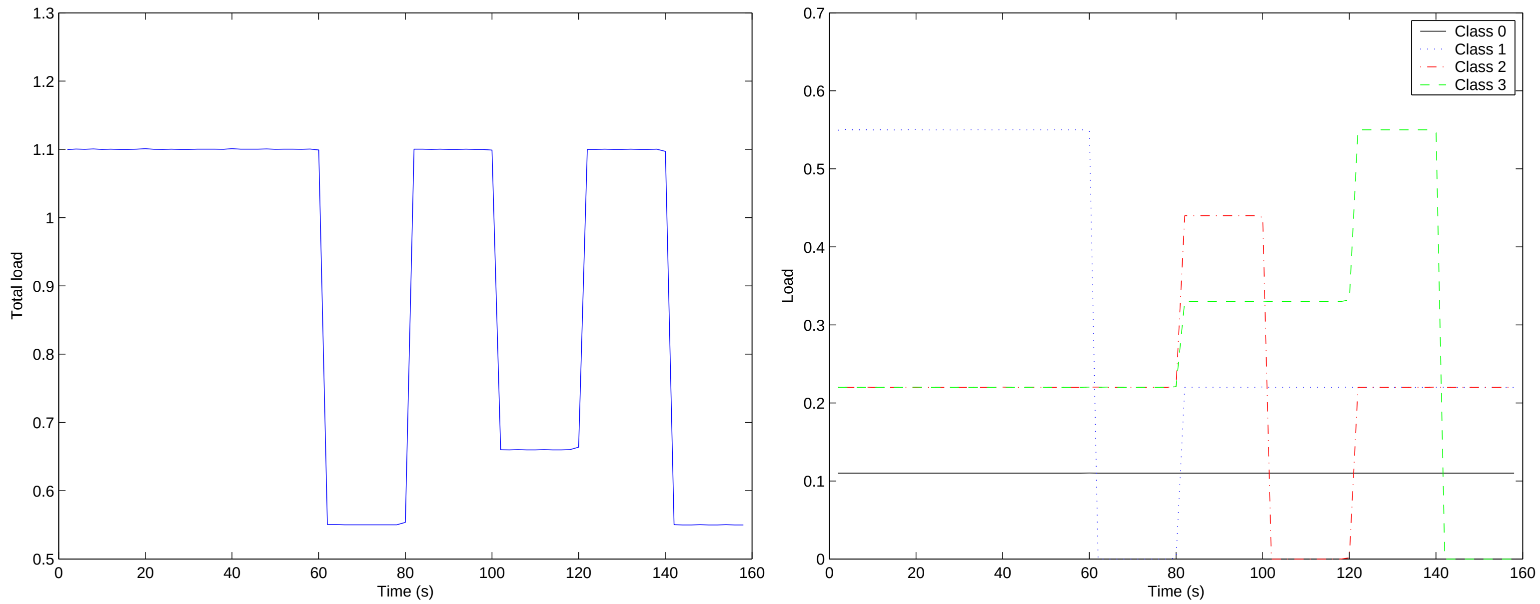
$$n = f\left(\frac{d_i(m)}{\overline{d_i}(m)}\right)$$

Simulation Scenarios

- Three different traffic mixes were used:
  - pure CBR
  - a multiplex of Pareto ON-OFF sources
  - traffic from ”real” applications
- Only some examples of the results are shown here

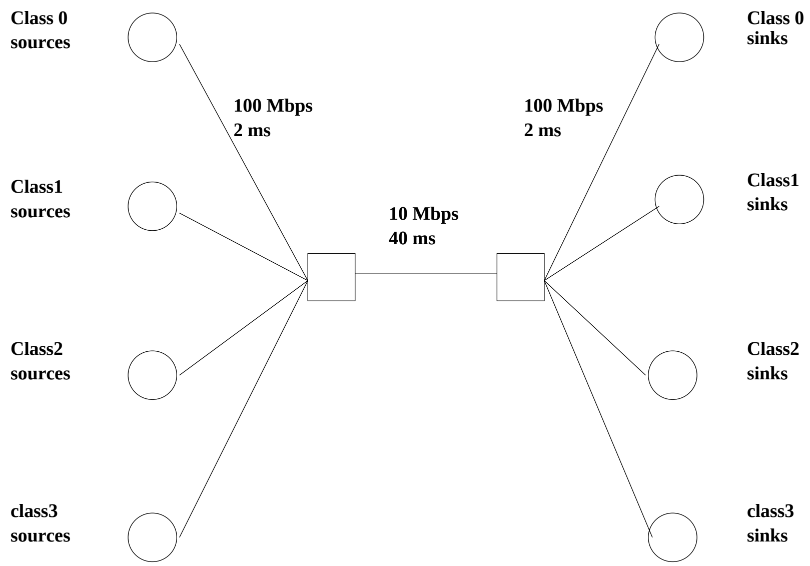
Simulation Scenarios

- CBR loads (total load and the loads in different traffic classes) used in traffic mix 1



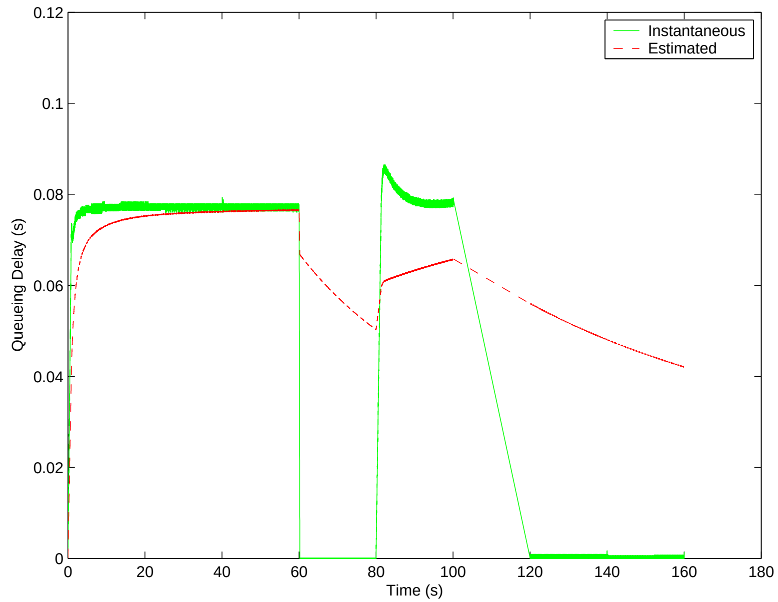
Simulation Scenarios

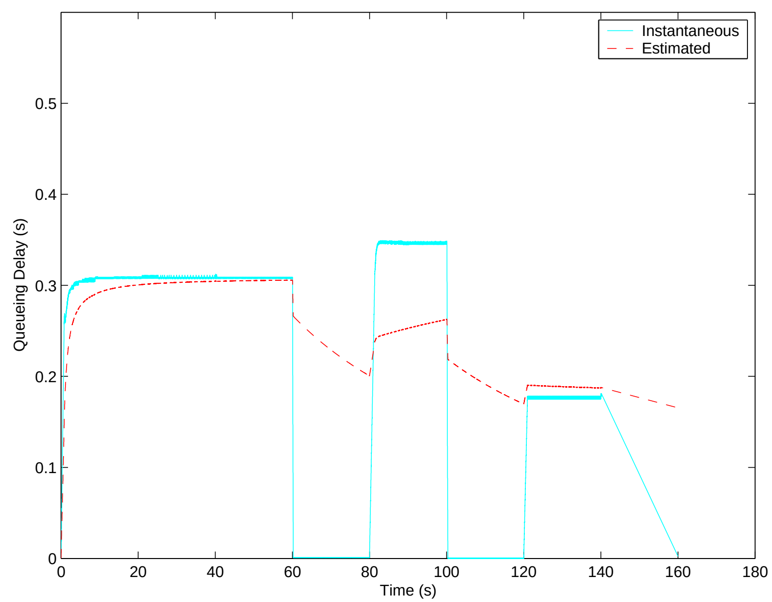
- Simple topology was used
  - complex topology not required when the filtering properties are tested



Simulation Results

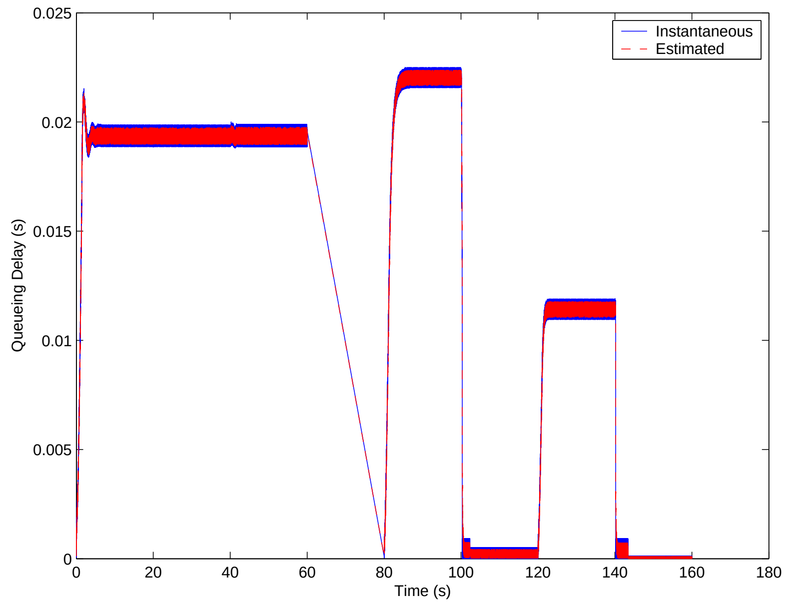
- Simple Sum estimator with pure CBR-traffic

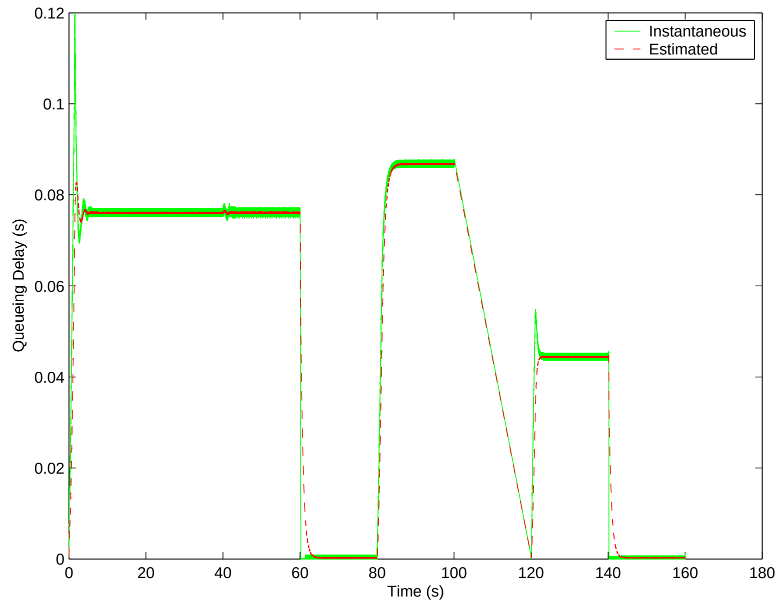




Simulation Results

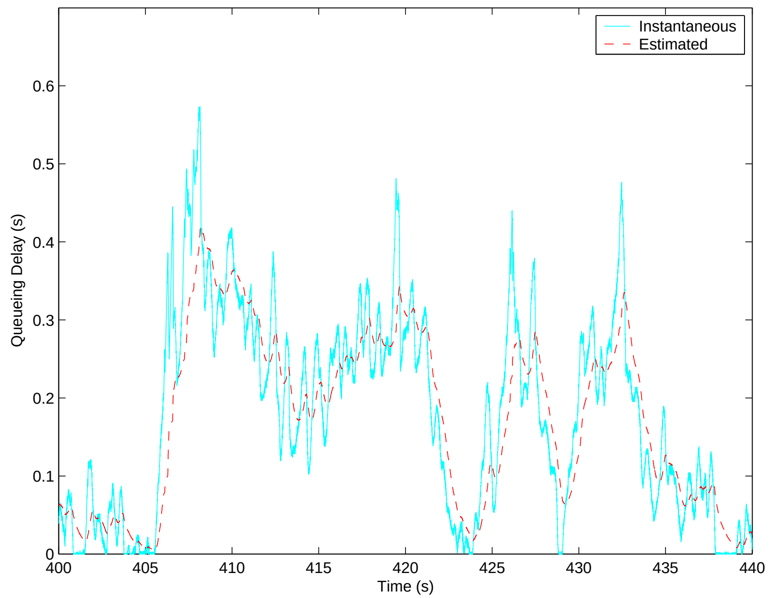
- EWMA-r estimator with pure CBR-traffic

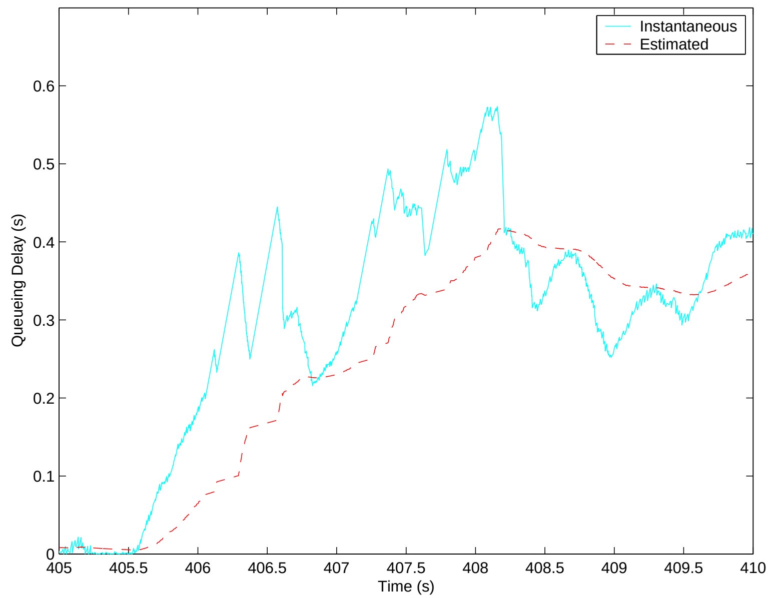




Simulation Results

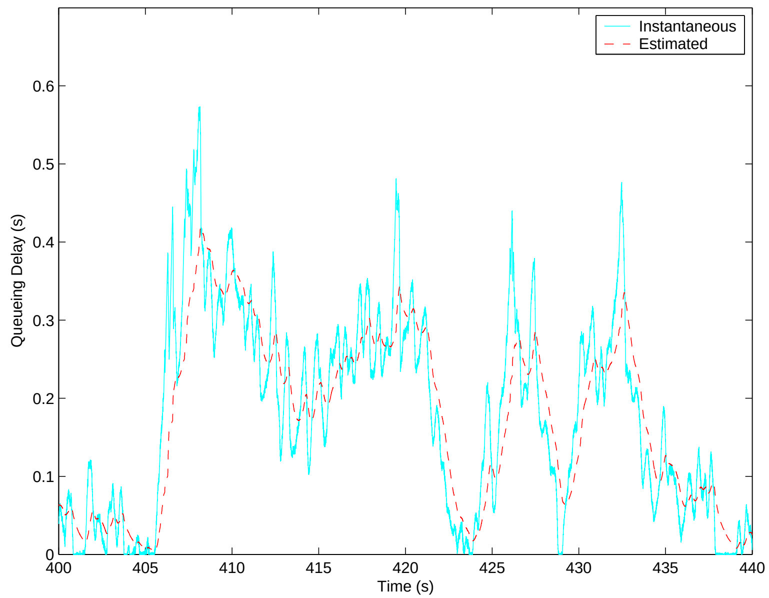
- EWMA-r estimator with Pareto ON OFF traffic

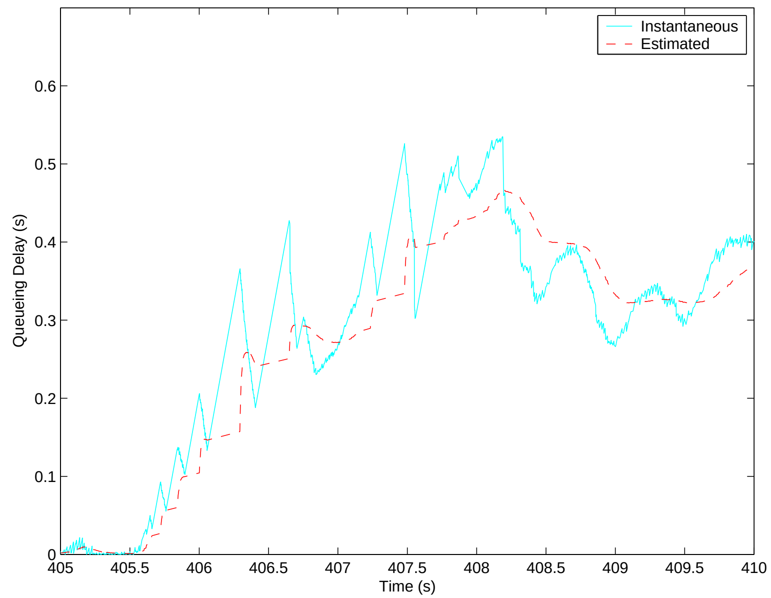




Simulation Results

- EWMA-pe estimator with Pareto ON OFF traffic

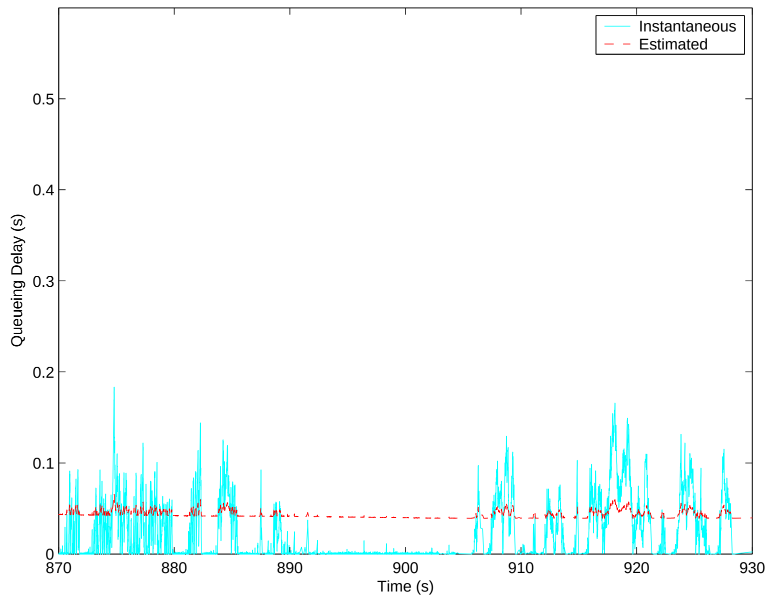


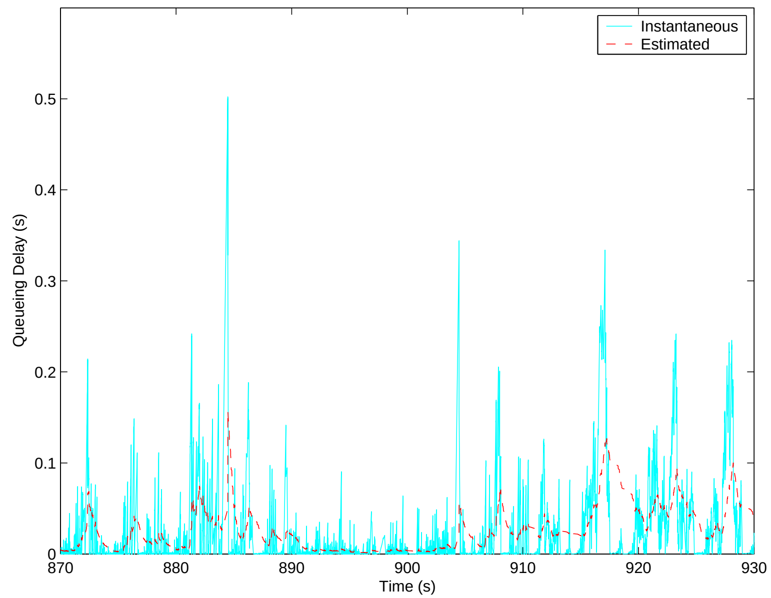


HELSINKI UNIVERSITY OF TECHNOLOGY

Simulation Results

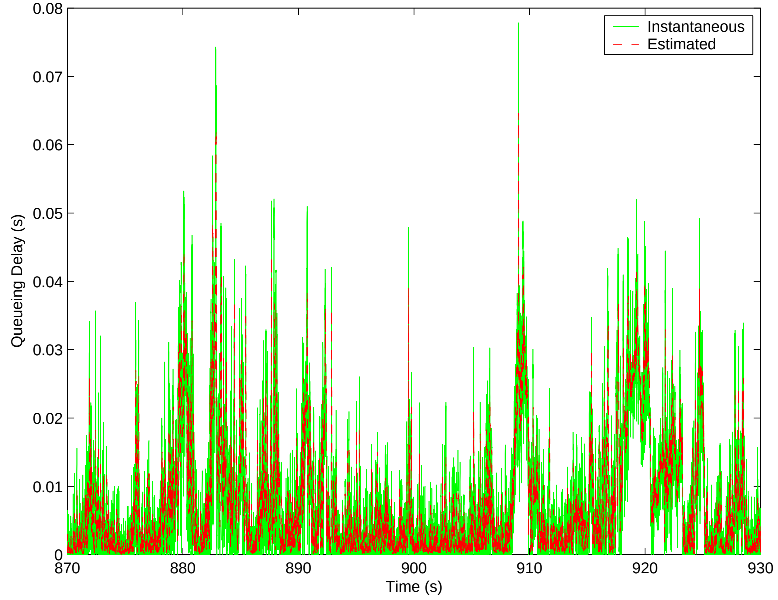
- Simple Sum and EWMA-r estimator with FTP traffic

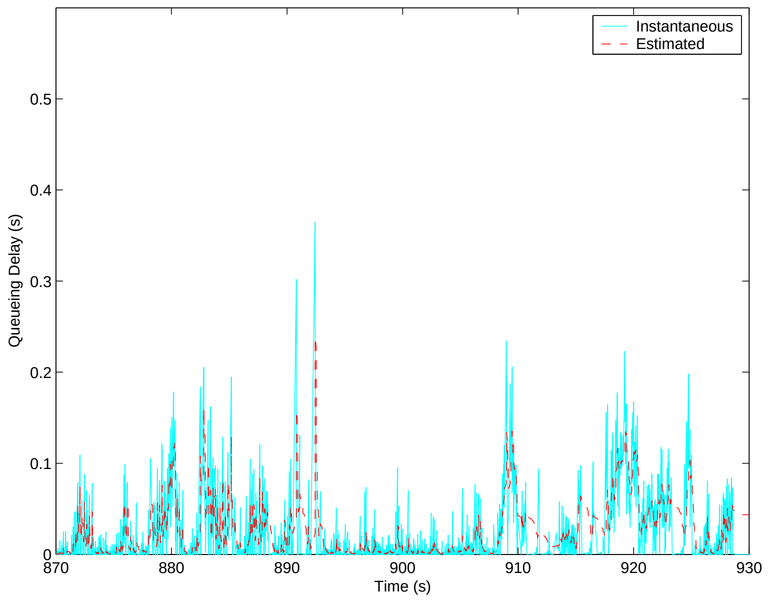




Simulation Results

- EWMA-pe estimator with HTTP and FTP traffic





Conclusions

- Lessons learned from the simulations
  - Simple Sum and Simple EWMA estimators lead to false scheduling decisions
    - not suitable for the delay estimation problem
  - EWMA-r and EWMA-pe seem to be promising
    - good filtering properties
    - simple to implement
  - Especially EWMA-pe operated well with very different types of traffic mixes

Future Work

- Network-wide performance analysis of the DBHPD algorithm with the new estimators
  - investigation of e.g. packet loss patterns and possible oscillatory behavior of TCP
  - refinement of the estimator functions if required
- Implementation of the estimators in an existing prototype version of DBHPD
  - measurements