Reformulating the monitor placement problem: Optimal Network-wide Sampling

Gianluca Iannaccone
Intel Research @ Cambridge

Joint work with:
G. Cantieni, P. Thiran (EPFL)
C. Barakat (INRIA), C. Diot (Intel)

Motivation

- Router-embedded monitoring functionalities are commonly used in small and large ISPs
  - e.g., Cisco's NetFlow
  - provide visibility over the entire network
  - level of details is "good enough" (for now...)

- Challenge:
  how to configure a network-wide monitoring infrastructure with hundreds of viewpoints?

Why is it a hard challenge?

- “Configure” means setting the sampling rates on all individual interfaces
- Sampling rates needs to be low to reduce stress on routers
- Aggregate volume of information collected from the routers should be kept under control
- Measurement task unknown a priori and a single fixed layout does not perform well
  – e.g., PoP-level traffic matrix estimation
    → all edge routers with low sampling rates.
  – e.g., focusing on specific prefix “below the radars”
    → few monitors, relatively higher sampling rates.

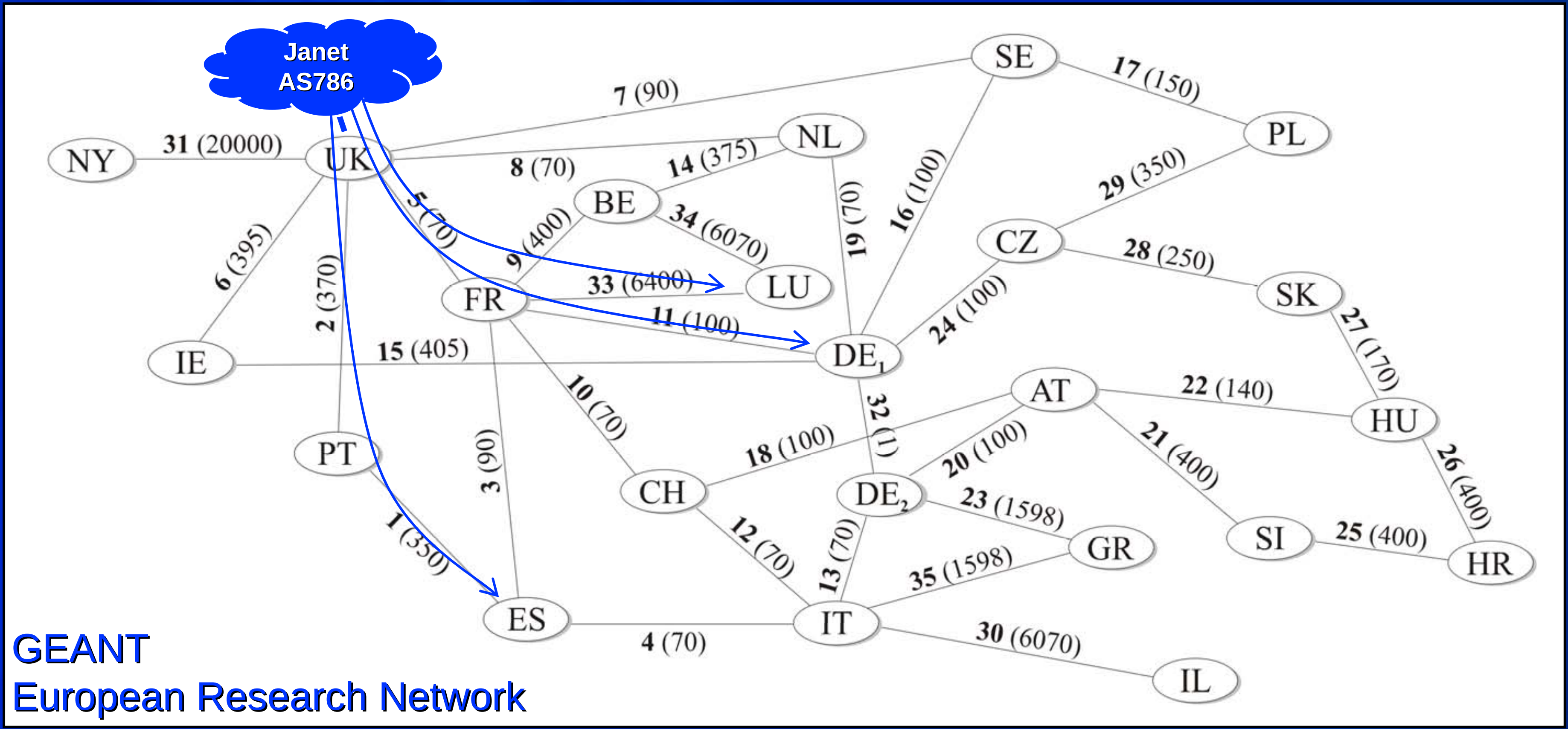
intel

Our objective

- Given a measurement task and a target accuracy, find a method that:
  - sets the sampling rates on all interfaces
  - guarantees optimal use of resources (in terms of processed packets)
  - requires minimum configuration
  - can adapt quickly to changes in the traffic
- Method should apply to a general class of measurement tasks

Picking a measurement task...

- Estimate amount of traffic flowing among a subset of origin-destination pairs
- Common task for traffic engineering apps



Problem formulation

Choose vector of sampling rates **p** that maximizes

utility function for OD pair k

"effective" sampling rate for OD pair k

$$\sum_{k\in\mathcal{F}} M\left(\rho_k\left(\boldsymbol{p}\right)\right),$$

sampling rate on link i

$$\begin{aligned} p_i &\geq 0 && \text{for all } i \in \mathcal{L} \\ p_i &\leq \alpha_i && \text{for all } i \in \mathcal{L} \\ \sum_{i\in\mathcal{L}} p_i U_i &\leq \theta, \end{aligned}$$

max sampling rate for link i

packets traversing link i

system capacity (in packets)

- Effective sampling rate approximated by sum of sampling rates
- All constraints are linear and define a convex solution space
- Unique maximizer exists as long as ***M()*** is strictly concave

Algorithm

- Solve system defined by KKT conditions
  - select set active/inactive constraints (equivalent to switching off/on a link monitor)
  - use gradient projection method to explore space
  - use KKT conditions to check optimality of solution
- Selection of active/inactive constraints is NP-hard → no guarantee of convergence
- Limit algorithm runs to 2,000 iterations → 98.6% optimum found (for our task)

The utility function

- Measures quality of sampling an OD pair
- "Well behaved" to make the algorithm run fast
- Mean square relative error good candidate
  - $E[SRE] = E[((X/\rho - S) / S)^2]$
  - actually $1 - E[SRE]$
    → mean square relative "accuracy"
- $M(\rho) = 1 - E[1/S] * (1/\rho - 1)$;
  - minor tweaking to force it to be zero when $\rho = 0$
  - needs $E[1/S]$ where S is the size of the OD pair

Evaluation

- Consider NetFlow data from GEANT
  - Collected using Juniper's Traffic Sampling
  - 1/1000 periodic sampling
  - We scale the measurement by 1000
    (we just need a realistic mix of OD pair sizes)
- Results based on one run of the algorithm
  - One five minute snapshot of the network traffic
  - Compute OD pair sizes and link loads
  - Assume E[1/S] is known

intel

Results highlights

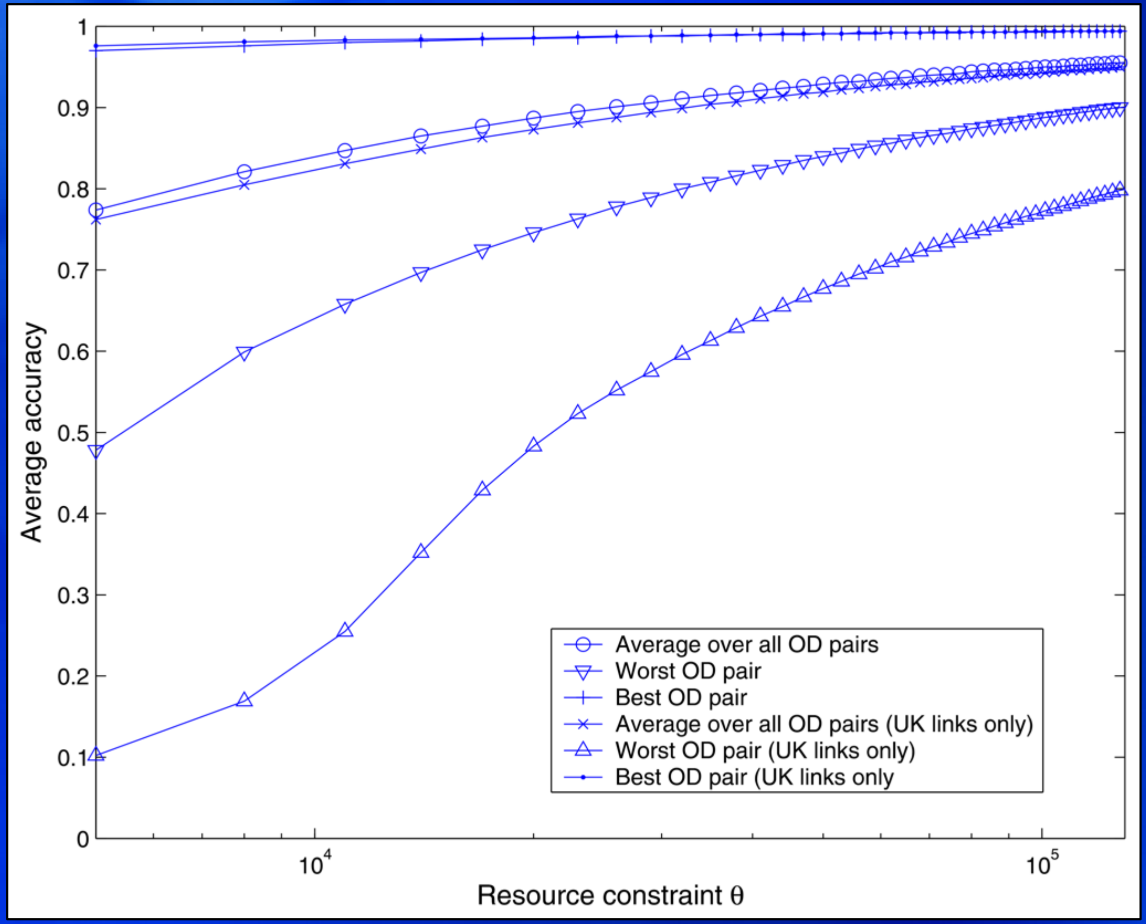
- Measuring relative accuracy
  - Defined as one minus relative error (not squared)
  - Allows to validate manipulation of utility function and the use of effective sampling rate
- Accuracy is in the range 89-99%
  - Worst accuracy for JANET – LU (it has just 20 pkts/sec)
- Measurement spread across 10 links
- Max sampling rates is 0.92% (lightly loaded links)
  - Most links are around 0.1%
  - No OD pair is monitored on more than two links
  - Effective sampling rate (sum of sampling rates) is a good approximation of actual sampling rate

Comparing to “naive” solutions

- Why not just monitoring JANET access link?
  - All the monitored traffic would be relevant!
  - To achieve same accuracy over all OD pairs we need ~1% sampling rate
    → 70% more packets are processed
  - It’s not always possible to monitor both directions of access links
- Why not just monitoring all UK links?
  - There are just 6 links leaving the UK
  - Straightforward algorithm to set sampling rate (each OD pair is present on just one link), but...

intel

Monitoring all UK links

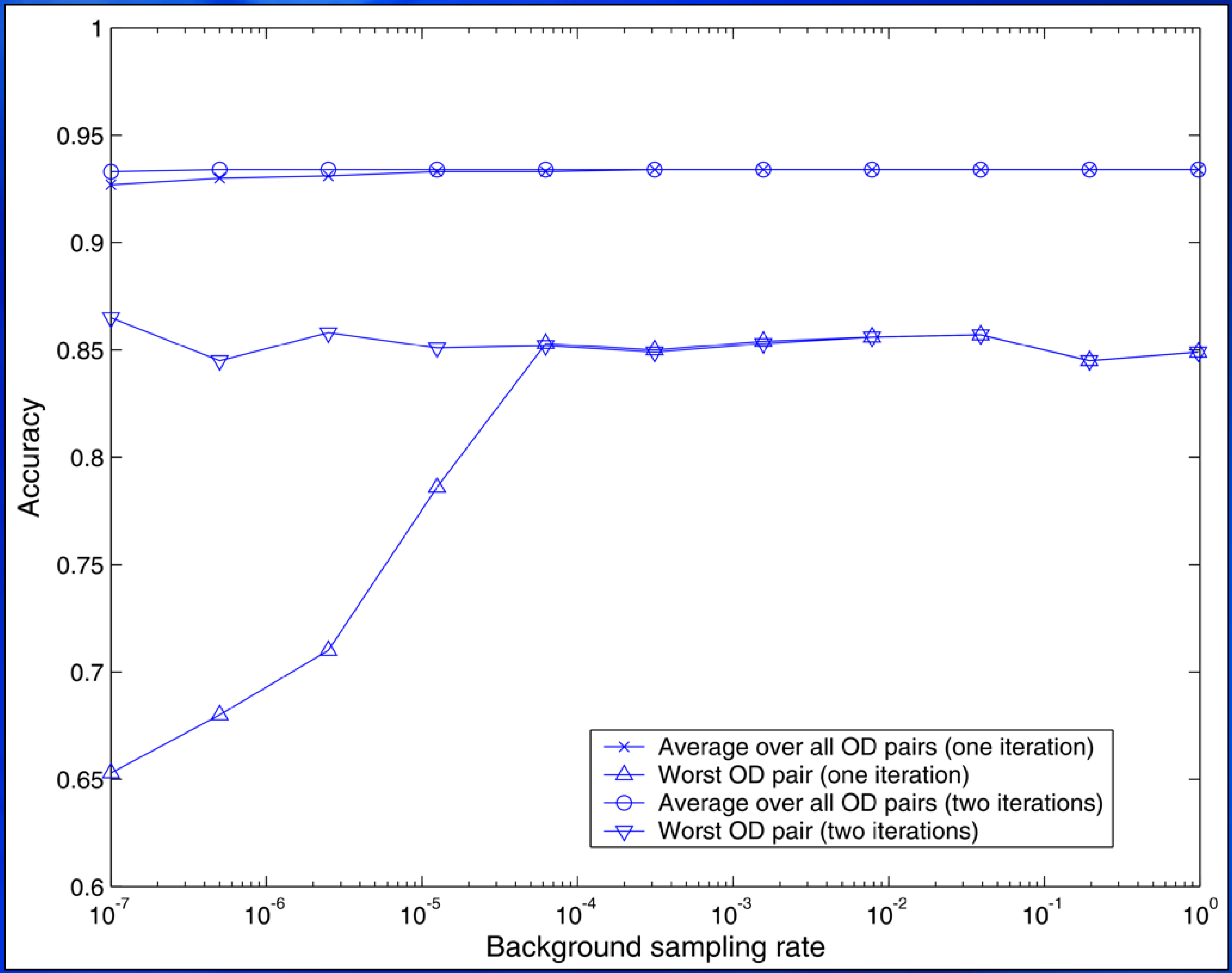


- Why does our method work better?
  - It looks across the entire network to find where small OD pairs manifest themselves without hiding behind large flows

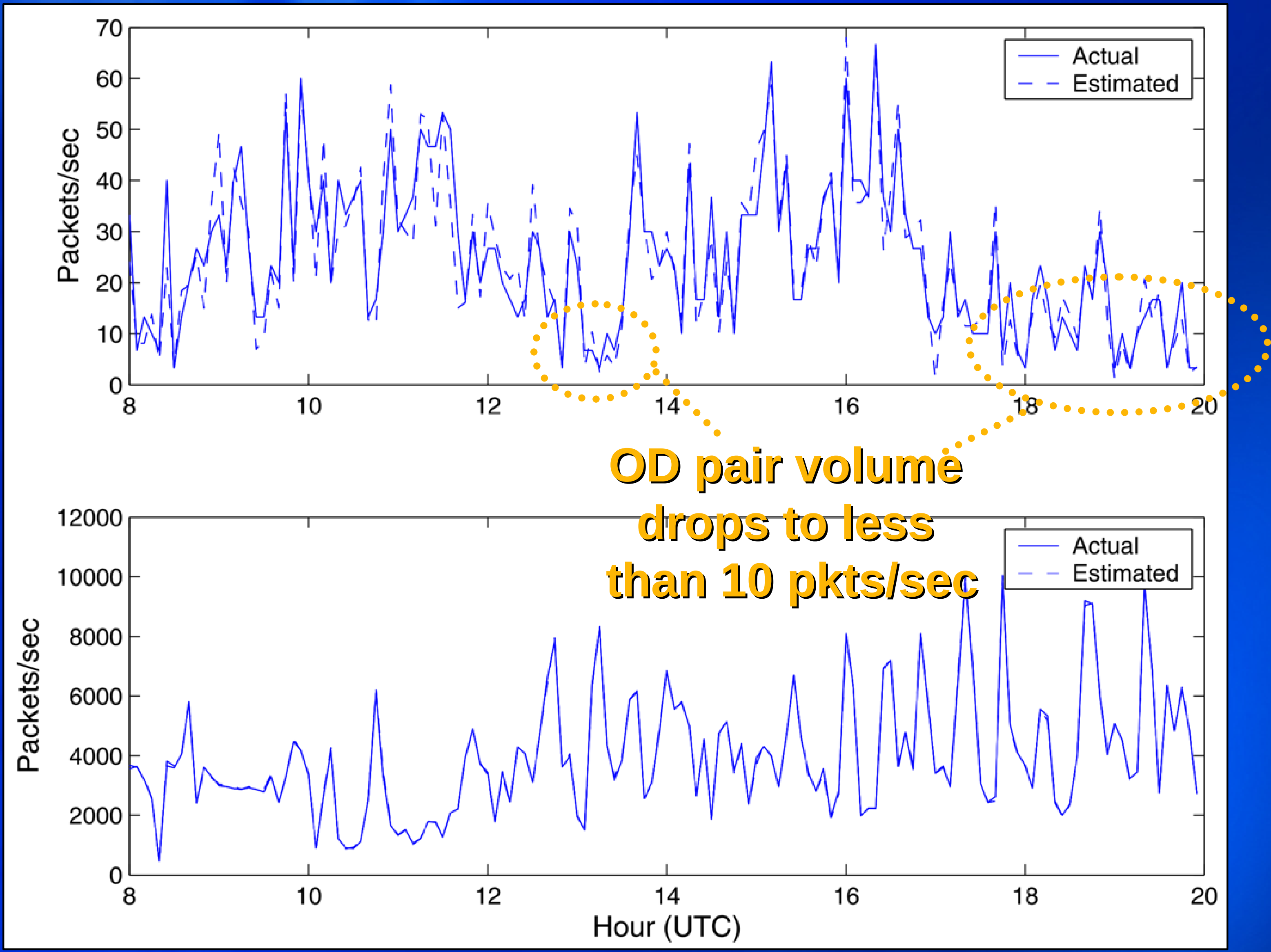
Deployment on real networks

- Two aspects need to be addressed
- What prior knowledge about the network does the method need?
  - need routing information
  - need estimate of E[1/S] for each OD pair
    - → bootstrapping phase
- How does the method perform over time?
  - time of day effect change E[1/S] and U_i
  - routing event change path taken by OD pairs
    - → adapt sampling rates

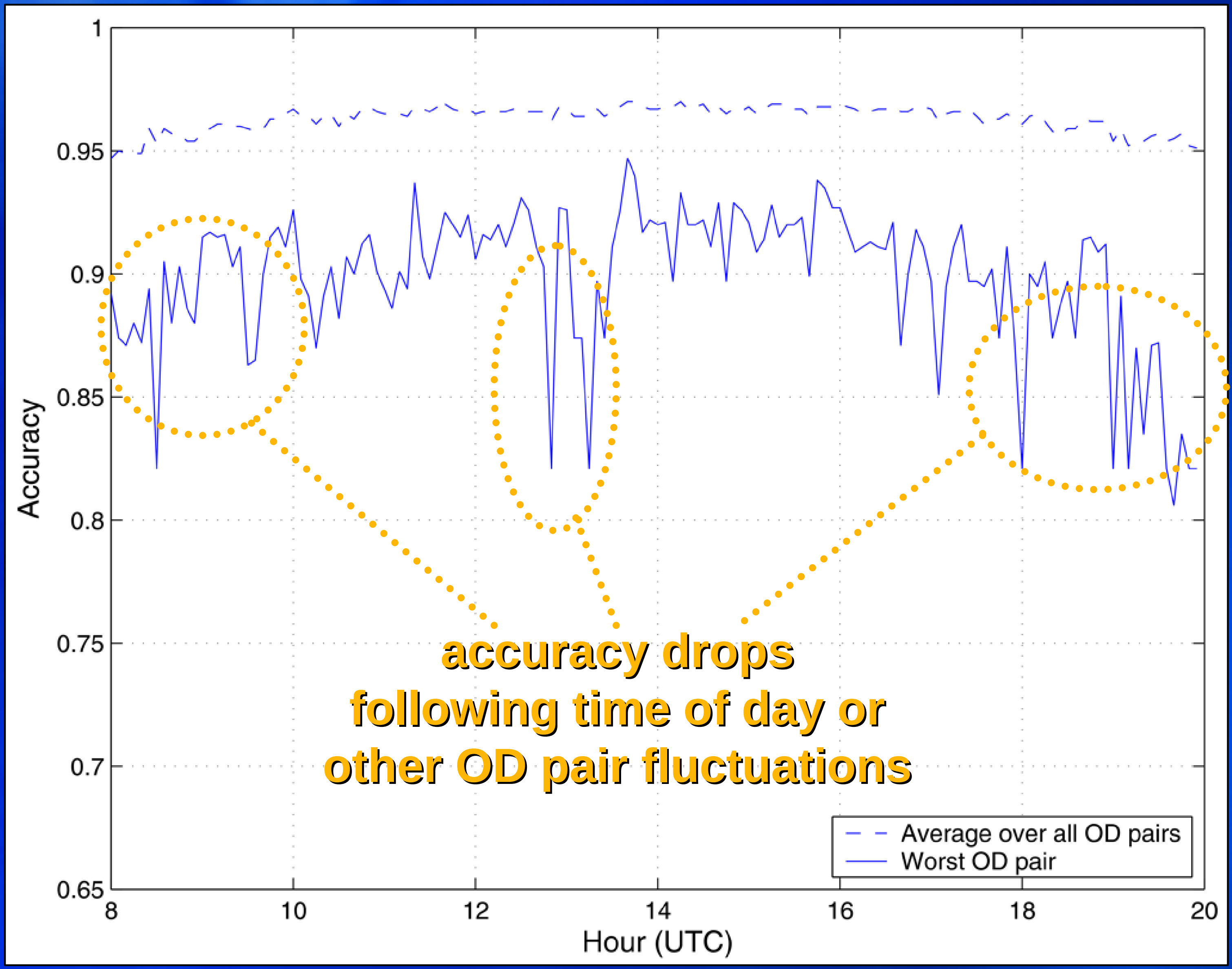
Bootstrapping phase



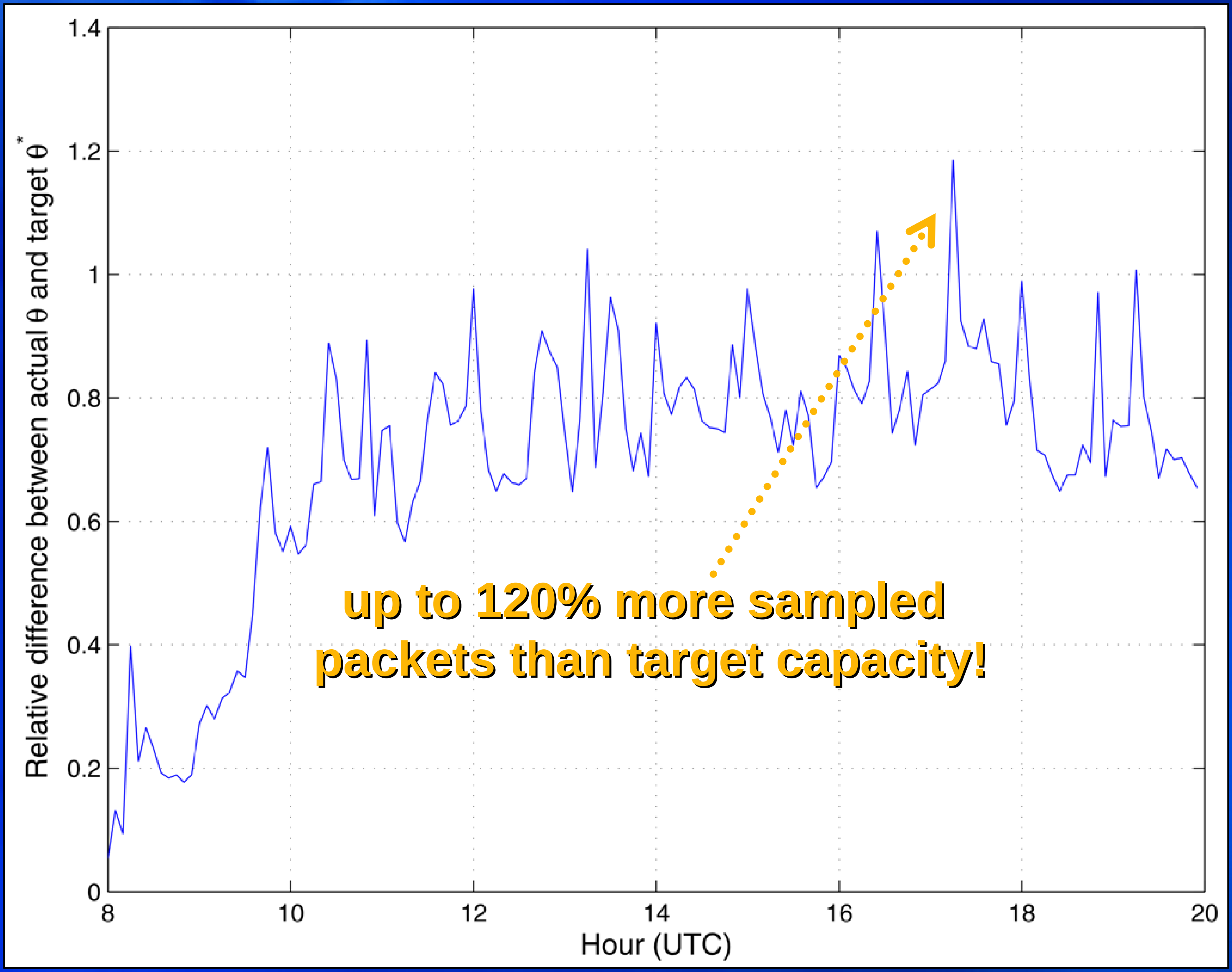
Performance over time



Performance over time (cont'd)



Performance over time (cont'd)



Adapting to traffic fluctuations

- Three different cases that require different approaches
- Link load increases
  - more sampled packets, exceeding capacity

    → find new sampling rates to enforce target capacity
- OD pair decreases in volume
  - poor accuracy because of bad E[1/S] estimate

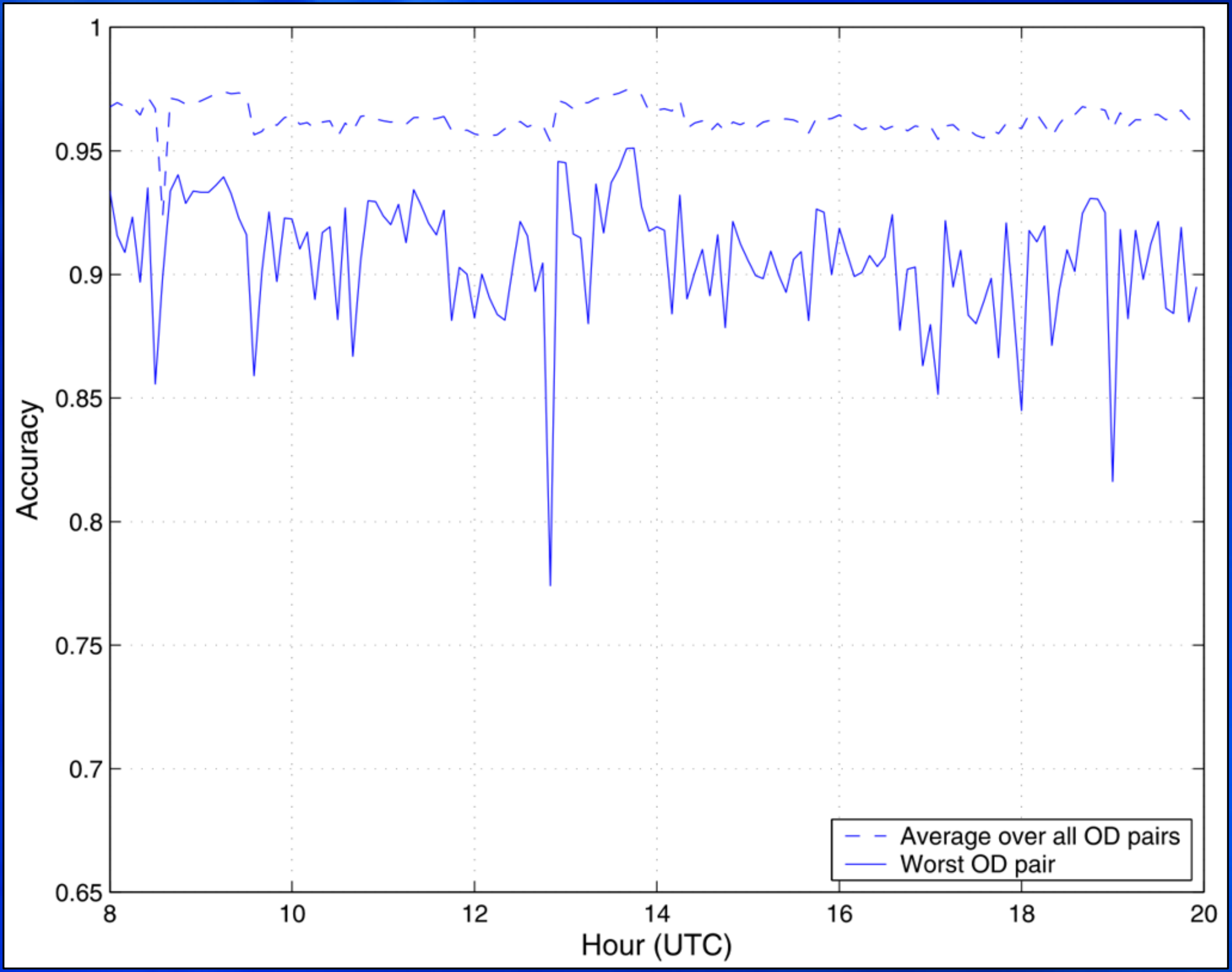
    → adapt capacity Θ to keep target accuracy
- OD pair traverses different set of links
  - missing entire OD pair

    → monitor routing updates and “re-bootstrap” the algorithm

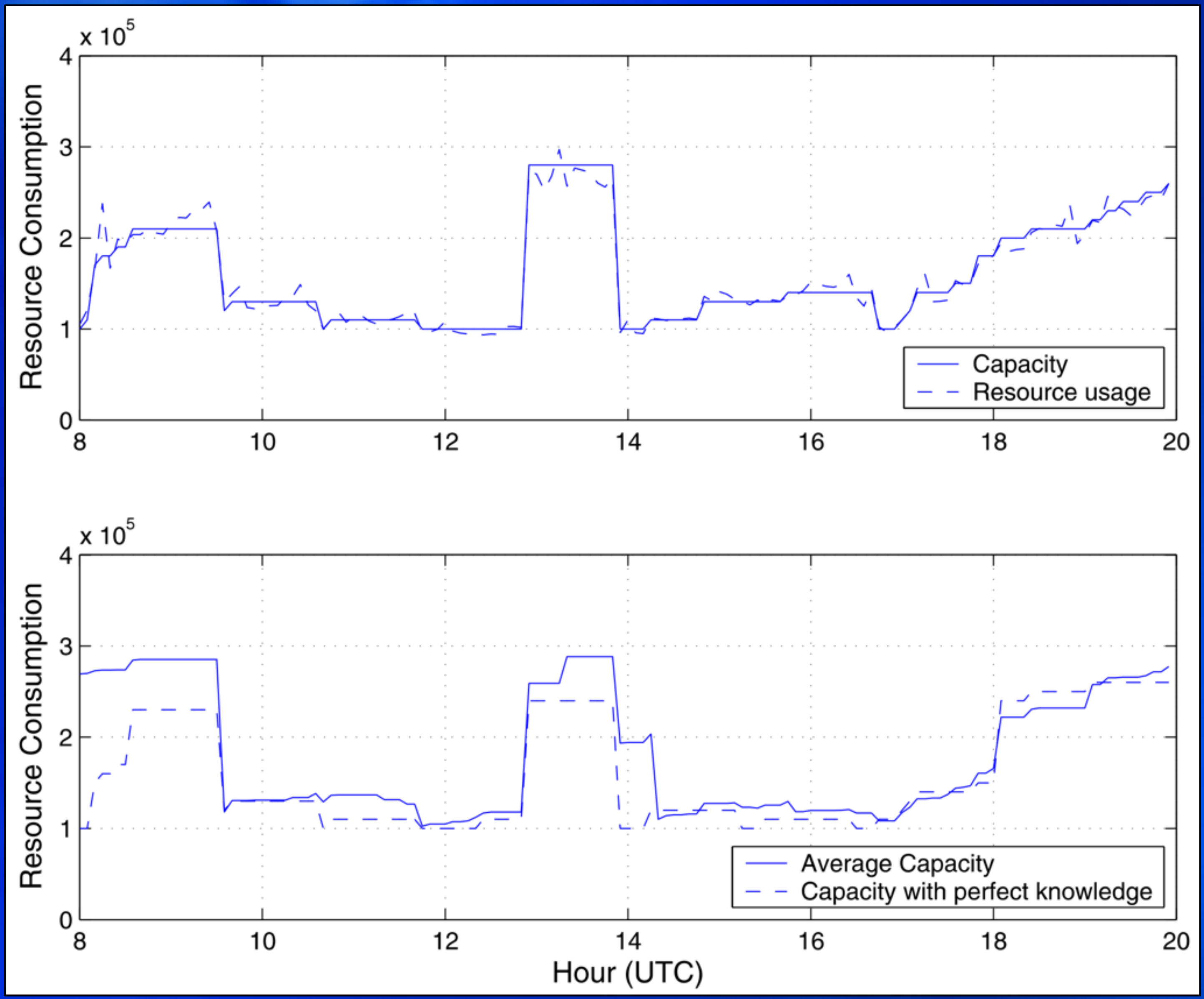
Fluctuations in OD pairs

- Monitoring accuracy of OD pairs
  - This is not trivial. Accuracy is not known.
  - Need to estimate E[1/S] from sampled data.
  - Use simplest method → Current size of OD pair
- Compute new sampling rates when estimated accuracy drops below target
- If the estimated accuracy is still below target, increase capacity by 10%
- Decrease capacity if estimated accuracy is above target for more than one hour

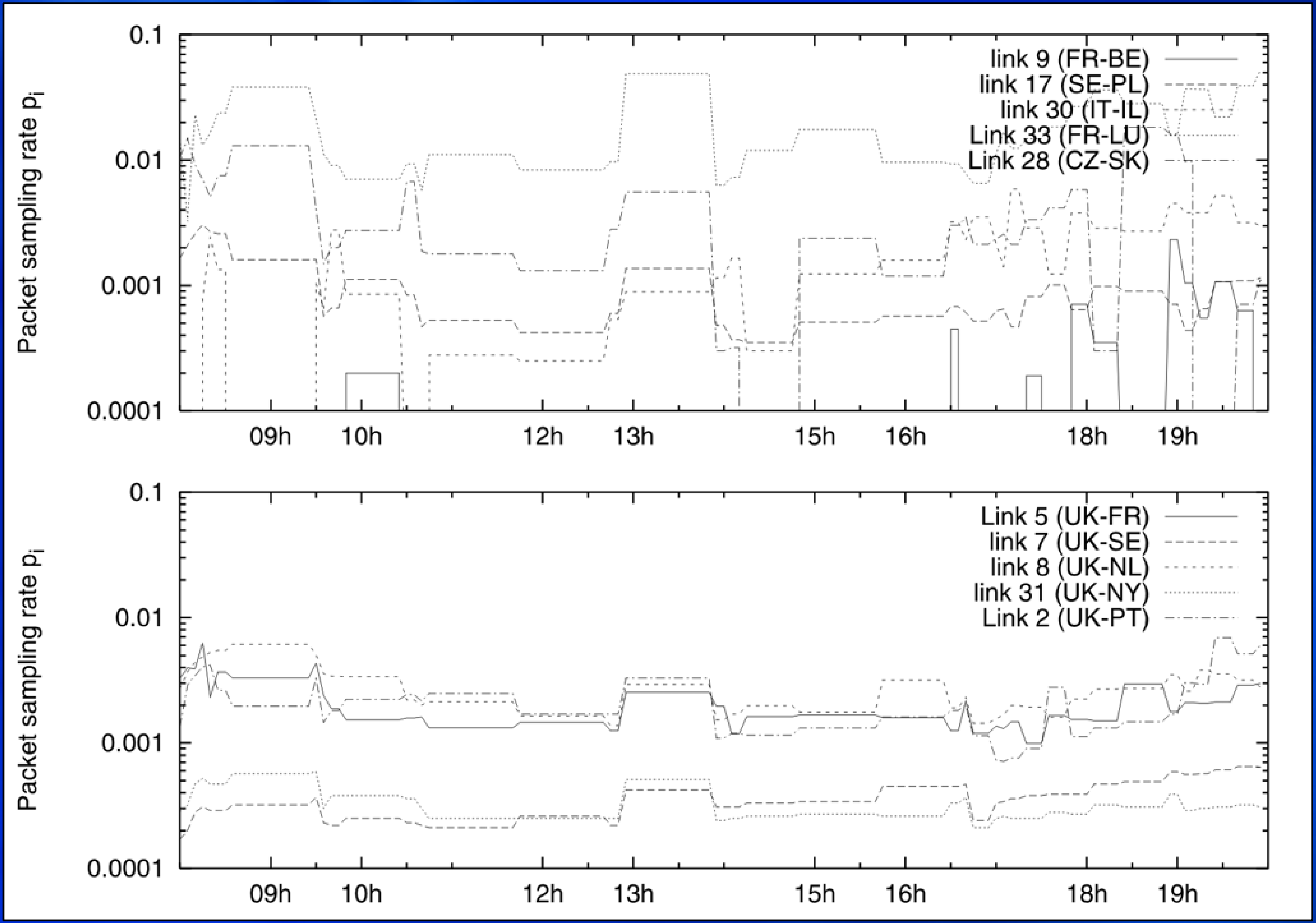
Fluctuations in OD pairs (cont'd)



Fluctuations in OD pairs (cont'd)



Fluctuations in OD pairs (cont'd)



Related work

- Passive monitoring
  - Suh et al, “Locating Network Monitors...”, Infocom 2005
    - two phase approach: select the monitors then optimize sampling
    - near-optimal solutions
- Active monitoring
  - Bejerano, Rastogi, “Robust monitoring of link delays”, Infocom 2003
  - Jamin et al., “On the placement of Internet instrumentation”, Infocom 2000
- Improving NetFlow
  - Estan et al, “Building a better NetFlow”, Sigcomm 2004
  - Baek-Yong et al. “... Adaptive Sampling...”
- TM estimation work
  - really a different problem setting

Conclusion & Future work

- Set sampling rates of a network of monitors.
- General enough framework for large class of measurement tasks

- Working on finding new utility functions
- Looking into using better predictors for E[1/S]

- Open issue
  How long does it take to reconfigure NetFlow?