Applications of Clock Synchronization: Network Telemetry
Recall two scenarios:

**Fast loop**
- Traders place orders based on market data feeds from several exchanges
  - In fact, U.S. law requires a trader investing other people’s money to find the “national best bid or offer” (NBBO)
  - Currently required to check prices at 13 U.S. stock exchanges

**Slow loop**
- Trading firm collects market data from several exchanges to run large-scale computations that determine trading strategies

In both cases it is critical to capture market data with accurate “head end” timestamps; i.e., to synchronize clocks across the trading venues

In the fast loop, it is also critical to have “fresh” market data
A trader in Singapore is comparing the market data from SGX and HKEX to place orders in Singapore.

**He needs:**
- Accurate clock sync between SG and HK to know what the HKEX timestamps mean in terms of his local clock.
- Precise monitoring of HK→SG link quality to ensure market data is not delayed.
The HK→SG link has two types of latency issues:

1. Delays due to congestion from trader’s own traffic or, more likely, from cross traffic (the link is leased)
   • Traffic-dependent, stochastic

2. Variation in the propagation time due to MPLS, as load is dynamically balanced on the link across different wavelengths
   • Periodic, step-like
MPLS-related change in RTT on link from OR→VA in:

- The change in OWD, and hence RTT, on the link shows discrete jumps of 100–150μs.
- Imperceptible to most applications, but it can throw off accurate clock sync without careful compensation.
A trader in Singapore is comparing the market data from SGX and HKEX to place orders in Singapore.

He needs:
- Accurate clock sync between SG and HK to know what the HKEX timestamps mean in terms of his local clock.
- Precise monitoring of HK→SG link quality to ensure market data is not delayed.

Accurate clock sync enables:
- Market data timestamps to be synced across HK and SG, despite changes in propagation time.
- It also enables tracking path delays and link quality at a fine-grained level.
SIMON: Simple Monitoring of Networks via Edge-based Network Reconstruction
Network Telemetry in a Data Center

The Setting
- A data center supporting applications which are running large-scale computations to devise trading strategies based on market data
- Or, just a data center supporting any apps

Key Question
- When a packet or an RPC has a large in-network transit time or got dropped, can we determine which switch/link in the data center caused it?

Telemetry via Tomography
- Using a probe mesh to synchronize clocks in the end-hosts, it is possible to “reconstruct” the delays on individual links based on delays experienced by the probes
- Details in SIMON paper: https://www.usenix.org/conference/nsdi19/presentation/geng
Network Telemetry From The Edge: Tomography

From total time in the network, determine time spent in each switch

Clock Synchronization

TX Timestamp

RX Timestamp
Algorithm

• **Input:**
  – 5-tuples of packets, for inferring network paths
  – Tx, Rx timestamps of packets

• **Basic equations**
  – For each packet:
    
    \[
    \text{One way delay} = \sum_{\text{hops}} \text{queueing delay} + \text{propagation delay}
    \]
  – Combine all packets: \( D = AQ + N \)
  – Solve for queue sizes: Use the Lasso algorithm
    \[
    \hat{Q} = \arg\min_{Q} \|D - AQ\|_2 + \lambda \|Q\|_1
    \]
Reconstruct Average Queue Length in a Recon-interval

Queue size sampled every 1us

1ms average of the queue size

Reconstruct this, much simpler and scalable
Signal Processing Explanation:
Averaging = Low-pass Filter

\[ Q_i(t) \xrightarrow{\text{Average per 1ms}} Q'_i(t) \xrightarrow{\text{Moving average over 1ms window}} \bar{Q}_i(t) \]

\[ |H(f)| \]

\[ |X(f)| \]

[Graph showing frequency response]
Power Spectral Density of the Queue Process

• Autocorrelation of the queue process
  \[ R_Q(\tau) = E[Q(t + \tau)Q(t)], \text{for } \tau \geq 0 \]

• Power Spectral Density (PSD) of the queue process
  \[ S_Q(f) = \sum_{\tau=-\infty}^{\infty} R_Q(\tau)e^{-i2\pi f \tau}, \text{for } |f\tau| < \frac{1}{2} \text{ or } |f| < 0.5 \text{MHz}(\tau) \]

\((\tau) = 1 \mu s\)
Power Spectral Density of the Queue Process (Cont’)

1 ms recon-interval preserves 97.5% of the power

Power of removed high frequency component: $(12KB)^2$
Estimates Well

$$\| \hat{Q} - Q \|_2 = 5.2KB$$
So far...

- Used probe packets to reconstruct queues

- We still don’t know
  - Link utilizations
  - Whose packets are in the queues?
  - Whose packets are using the links?

- Need to use data packet timestamps and sizes
Inference of the Journey of A Data Packet

Pkt position at any point of time!
Queue and Link Decomposition

- **L2-switch 6 --> L3-switch 6**
  - Queue length (KBytes)
  - Time (sec)
  - NS3 groundtruth
  - Rack 0-7
  - Rack 0-15
  - Rack 16-23
  - Rack 24-31

- **ToR 14 --> L2-switch 14**
  - Queue length (KBytes)
  - Time (sec)
  - NS3 groundtruth

- **L2-switch 6 --> L3-switch 6**
  - Link utilization
  - Time (sec)

- **ToR 14 --> L2-switch 14**
  - Link utilization
  - Time (sec)

RMSE = 4.14KB
RMSE = 1%