Network Tomography

CS 552

Richard Martin

What is Network Tomography?

- Derive internal state of the network from:
  - external measurements (probes)
  - Some knowledge about networks
    - Captured in simple models.

Why Perform Network Tomography?

- Can't always see what's going in the network!
  - Vs. direct measurement.
- Performance
  - Find bottlenecks, link characteristics
- Diagnosis
  - Find when something is broken/slow.
- Security.
  - How to know someone added a hub/sniffer?

This week's papers

- J. C. Bolot
  - Finds bottleneck link bandwidth, average packet sizes using simple probes and analysis.
- R. Castro, et al.
  - Overview of Tomography Techniques
- M. Coats et. al.
  - Tries to derive topological structure of the network from probe measurements.
  - Tries to find the "most likely" structure from sets of delay measurements.
- Heidemann et. Al.
  - Recent survey and techniques (as of summer 2008)

Measurement Strategy

- Send stream of UDP packets (probes) to a target at regular intervals (every δ ms)
- Target host echos packets to source
- Size of the packet is constant (32 bytes)
- Vary δ (8,20,50, 100,200, 500 ms)
- Measure Round Trip Time (RTT) of each packet.

Definitions

s_n sending time of probe n

r_n: receiving time of probe n

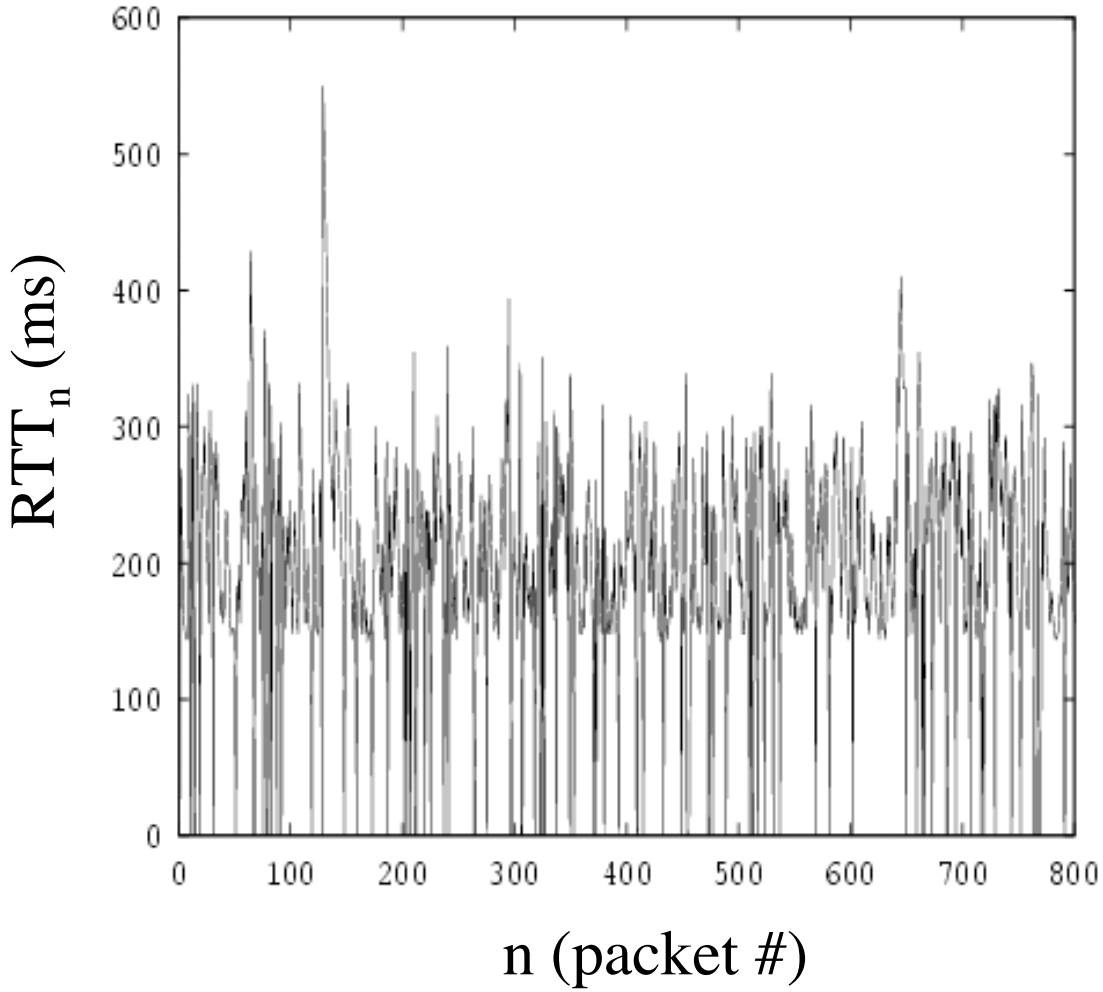
$rtt_n = r_n - s_n$: probe's RTT

δ: interval between probe sends

Lost packets: r_n undefined, define $rtt_n = 0$.

Time Series Analysis

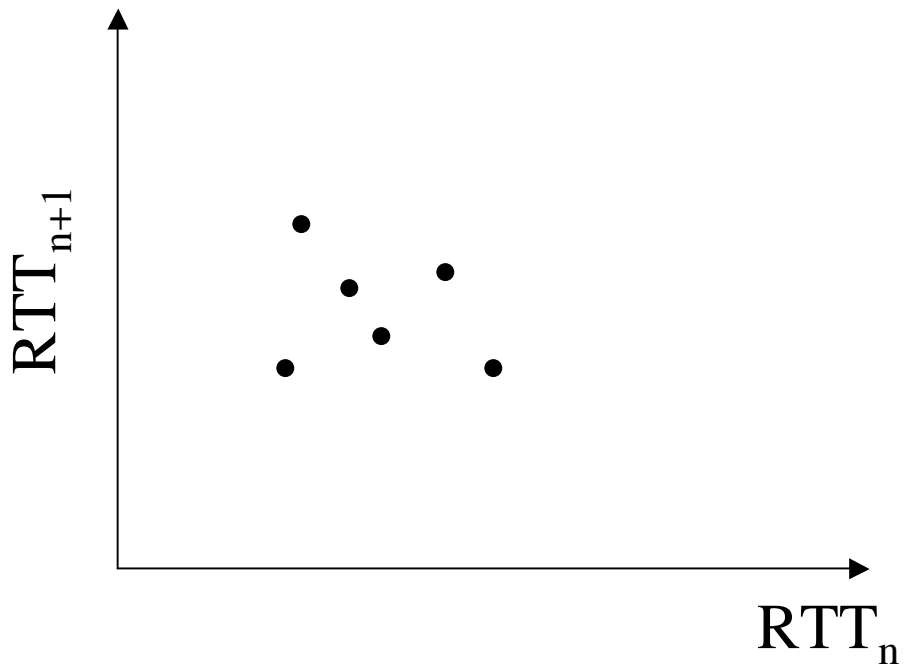
Min RTT: 140 ms
Mean RTT: ?
Loss rate: 9%



Classic Time series analysis

- Stochastic analysis
  - View RTT as a function of time (I.e. RTT as F(t))
  - Model fitting
  - Model prediction
- What do we really want from out data?
  - Tomography: learn critical aspects of the network

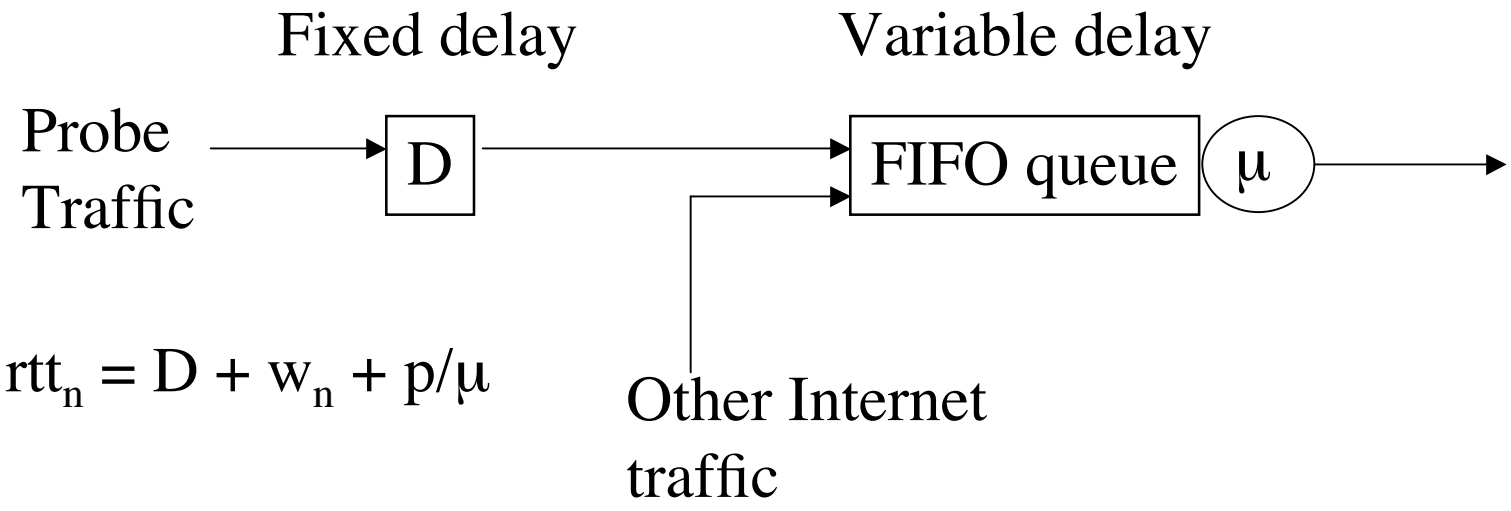
Phase Plot: Novel Interpretation



View difference between RTT's, not the RTT itself

Structure of phase plot tells us: bandwidth of bottleneck!

Simple Model



μ: bottleneck router's service rate
k: buffer size
p: size of the probe packet (bits)
w_n: waiting time for probe packet n

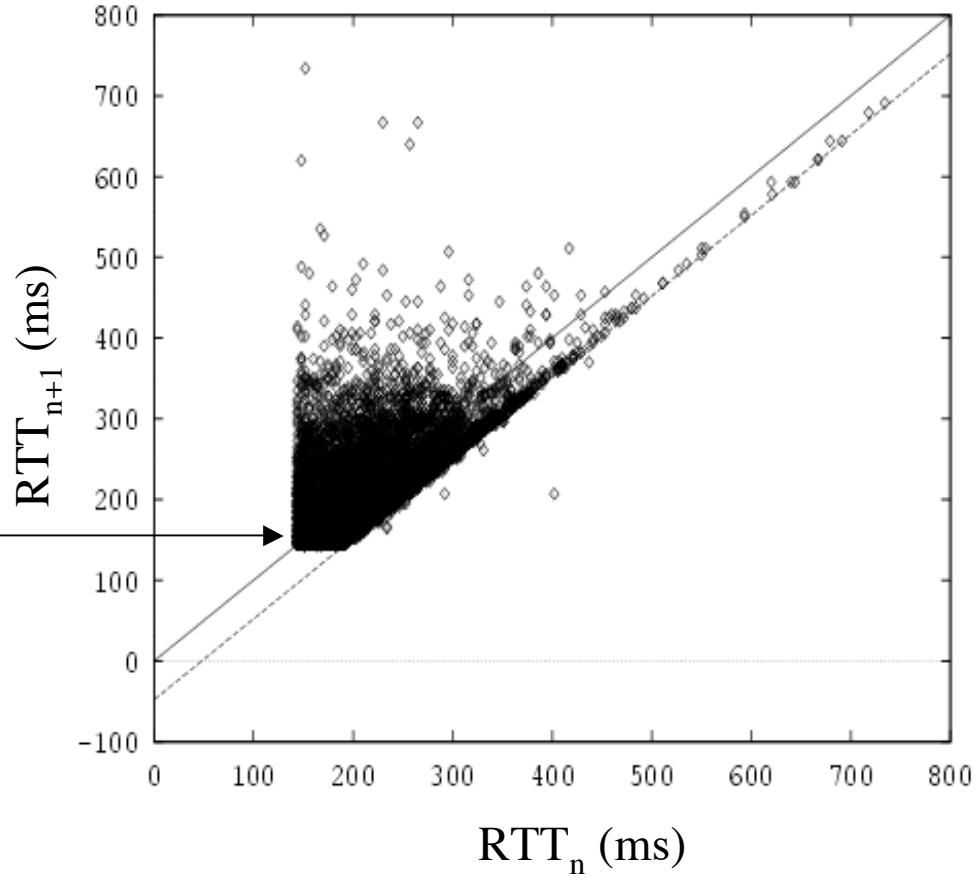
Expectation for light traffic

- What do we expect to see in the phase plot
  - when traffic is light
  - δ is large enough and p small enough not to cause load.
- $w_{n+1} = w_n$
- $rtt_{n+1} = rtt_n$
- For small p, approximate $w_n = 0$

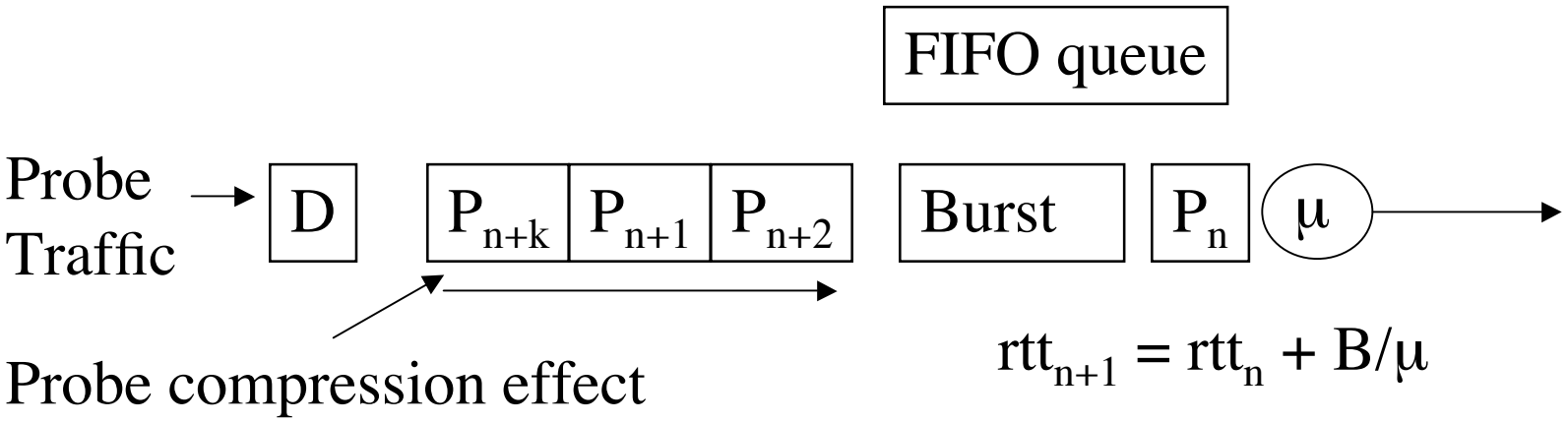
Light Traffic Example

n=800
δ=50 ms

"corner"
(D,D)
D = 140 ms



Heavy load expectation



$$rtt_{n+2} - rtt_{n+1} = (r_{n+2} - s_{n+2}) - (r_{n+1} - s_{n+1})$$
$$= (r_{n+2} - r_{n+1}) - (s_{n+2} - s_{n+1})$$
$$= p/\mu - \delta$$

Time between compressed probes

Time between probe sends

Heavy load, cont/

- What does the entire burst look like?

$$rtt_{n+3} - rtt_{n+2} = rtt_{n+k} - rtt_{n+k-1} = p/\mu - \delta$$

- Rewrite:

$$rtt_{n+1} = rtt_n + (p/\mu - \delta)$$

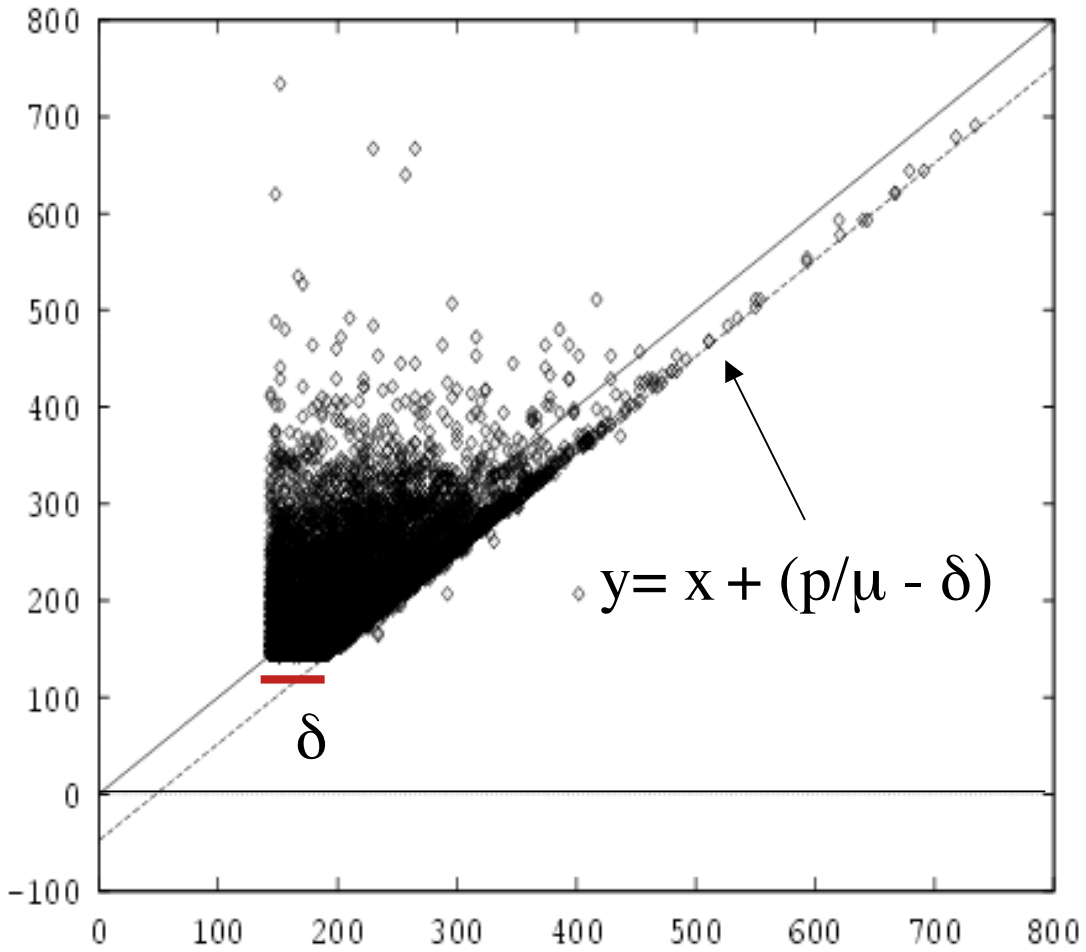
- General form:

$$y = x + (p/\mu - \delta)$$

Should observe such a line in the phase plot.

Finding the bottleneck

Find intercept.
Know p, δ, can compute μ !



Average packet size

- Can use phase data to find the average packet size on the internet.
- Idea: large packets disrupt phase data
  - Disruption from constant stream d, can infer size of the disruption.
  - Use distribution of rtt's

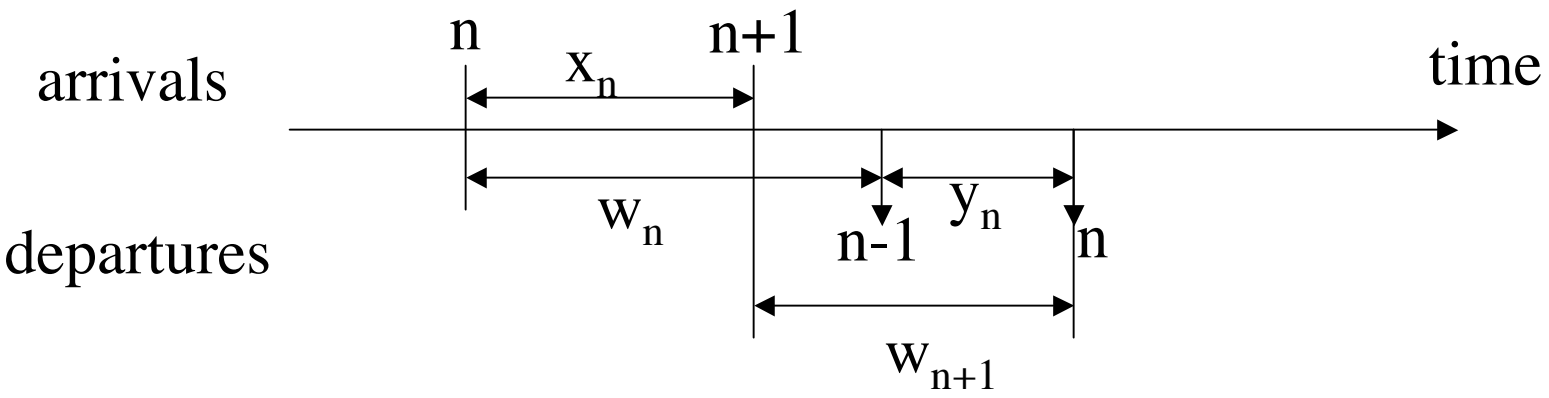
Average packet size

- Lindley's Recurrence equation
- Relationship between the waiting time of two successive customers in a queue:

  w_n: waiting time for customer n

  y_n: service time for customer n

  x_n: interarrival time between customers n, n+1



w_{n+1}= prev. packet wait + service - overlap

$$w_{n+1} = w_n + y_n - x_n, \text{ if } w_n + y_n - x_n > 0$$

Finding the burst size

- Model a slotted time of arrival where slots are defined by probe boundaries

$$wb_n = \max(w_n + p/\mu, 0)$$

- Apply recurrence:

$$w_{n+1} = w_n + (p + b_n)/\mu - \delta$$

- Substitute and solve for b_n:

Note: assume $w_n + (p + b_n)/\mu - \delta > 0$, then

$$b_n = \mu(w_{n+1} - w_n + \delta) - p$$

Distribution plot

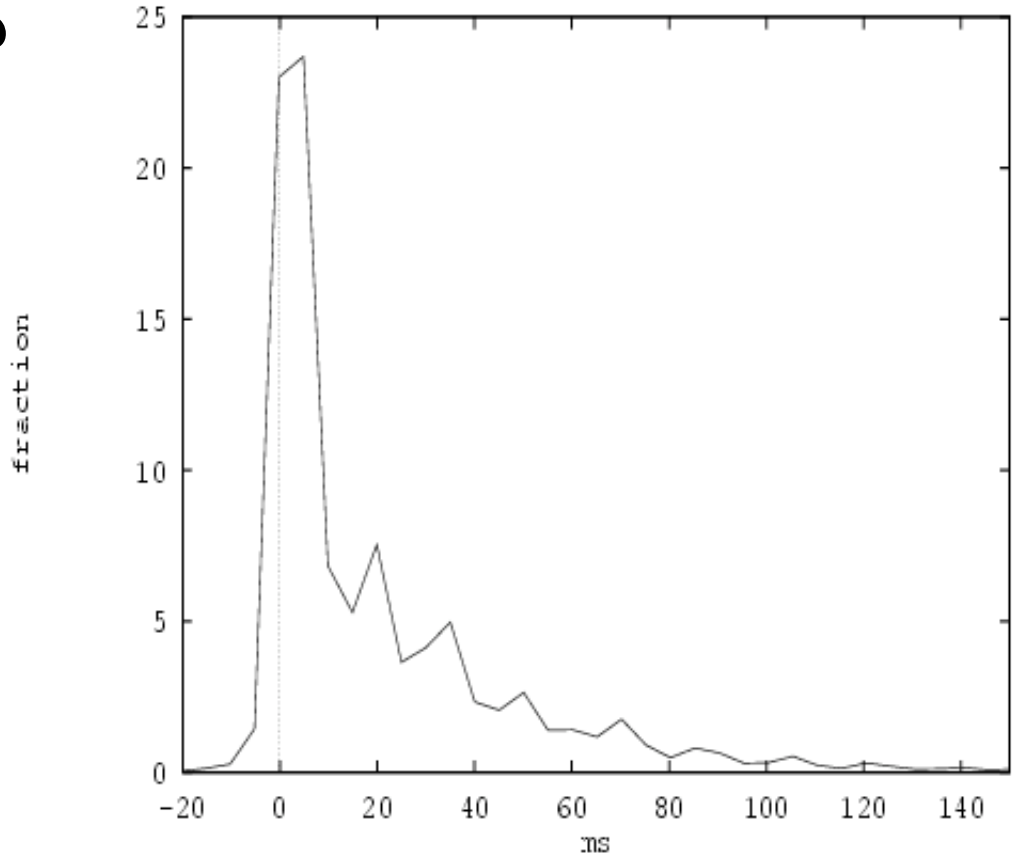
1st peak $w_{n+1}-w_n = p/\mu-\delta$

2nd: $w_{n+1}=w_n$

3rd: $b_n= \mu(w_{n+1}-w_n+\delta)-p$

know, μ, δ, p

solve for b_n



distribution of $w_{n+1} - w_n + \delta$, δ = 20 ms

Inter-arrival times

- A packet arrived in a slot if:

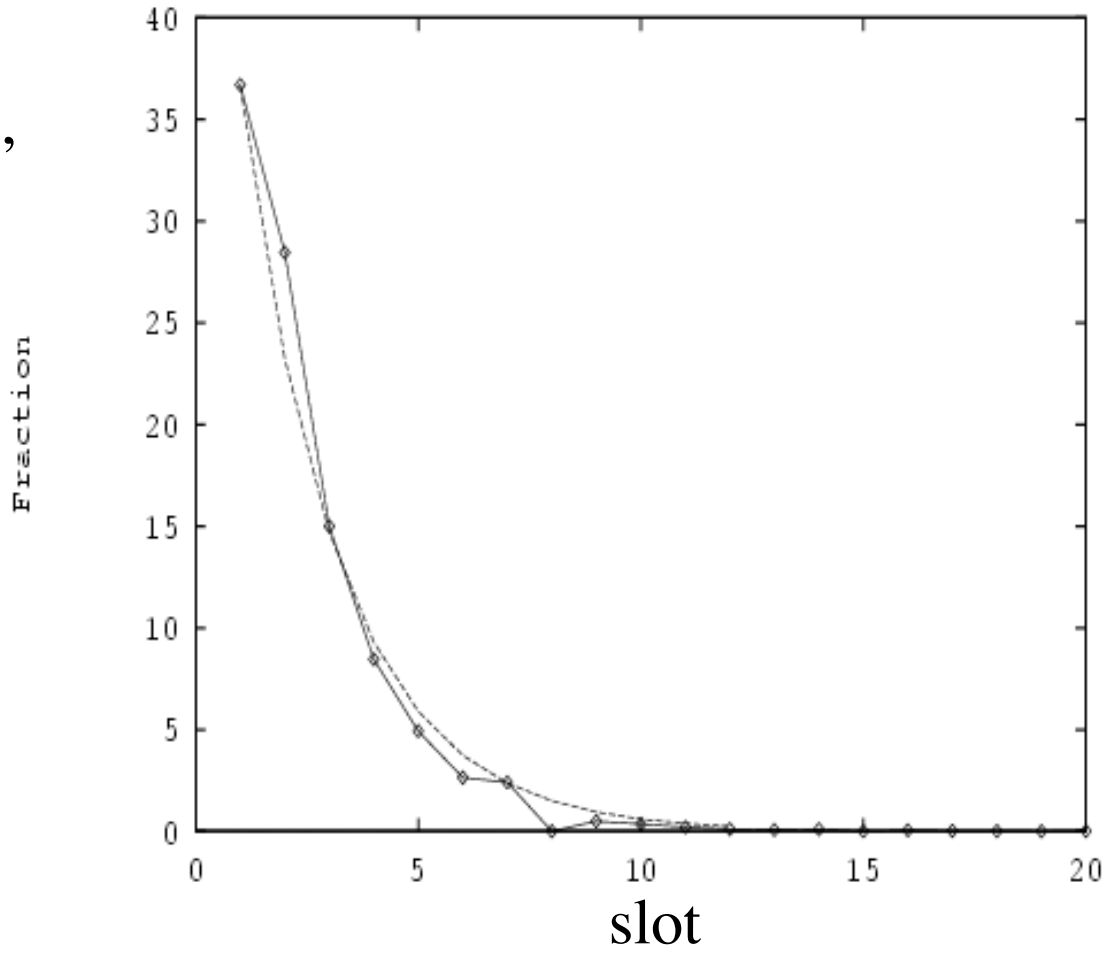
$$w_{n+1} - w_n > p/\mu - \delta$$

- Choose a small δ
- Avoid false positives
- Count a packet arrival if:

$$w_{n+1} - w_n > 0$$

Fraction of arrival slots

Fitted to $p(1-p)^{k-1}$, $p=0.37$



Packet loss

- What is unconditional likelihood of loss?
  - ulp = $P(rtt_n=0)$
- Given a lost packet, what is conditional likelihood will lose the next one?
  - clp = $P(rtt_{n+1}=0 \mid rtt_n=0)$
- Packet loss gap:
  - The number of packets lost in a burst
  - plg = 1/(1-clp)

Loss probabilities

δ(ms)	8	20	50	100	200	500
ulp	0.23	0.16	0.1	0.12	0.11	0.09
clp	0.6	0.42	0.27	0.18	0.18	0.09
plg	2.5	1.7	1.3	1.2	1.2	1.1

Tomography Overview

- Basic idea
- Methods
- Formal analysis
- Future directions

Traffic Matrix Approaches

- Cast problem of the form:
  - $Y_t = Ax_t + e_t$

Traffic Matrix example

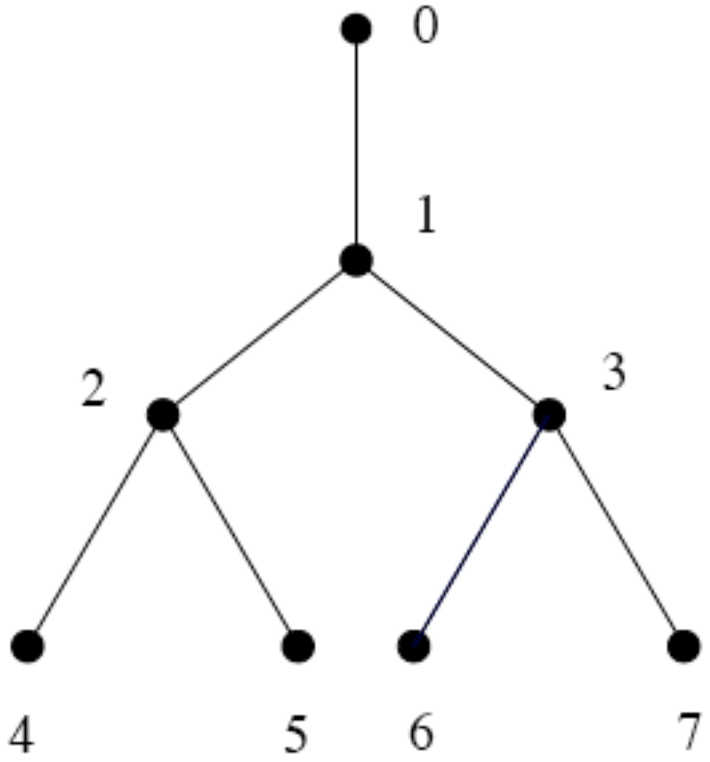
- Send multicast packet
- Measure delay of packet at receivers
- Shared paths result in shared delay
- Find the “most likely” tree given the observations

Traffic Matrix Example

Source node

Intermediate routes

Destination nodes



Problem Set-up

$$\underset{\text{Y}}{\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{pmatrix}} = \underset{\text{A}}{\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}} \underset{\text{X}}{\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_7 \end{pmatrix}}$$

End observed delay Routing matrix Link delays

Introduction

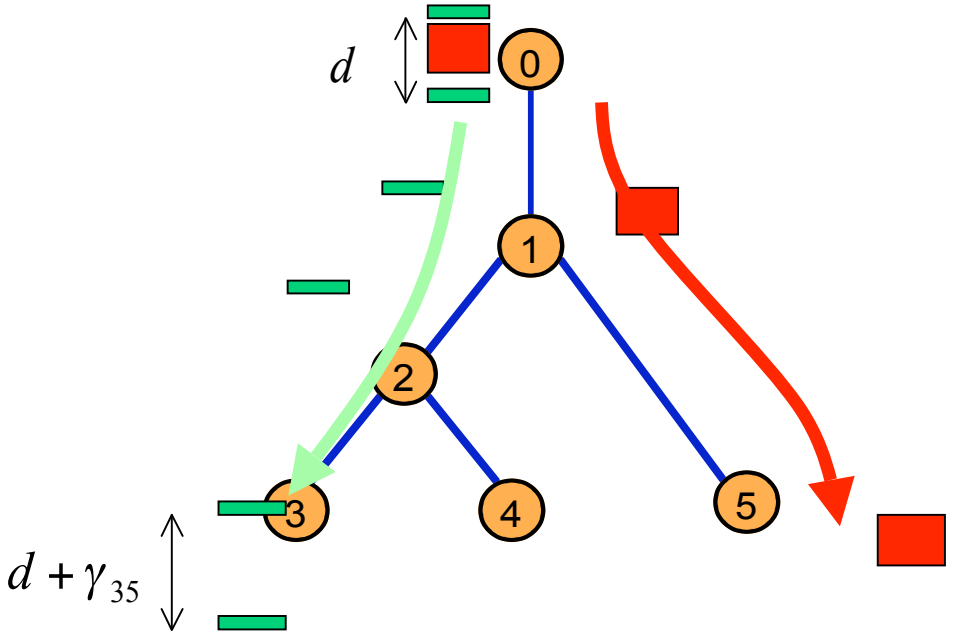
- Performance optimization of high-end applications
- Spatially localized information about network performance
  - Two gathering approaches:
  - Internal: impractical(CPU load, scalability, administration…)
  - External: network tomography
- Cooperative conditions: increasingly uncommon
- Assumption: the routers from the sender to the receiver are fixed during the measurement period

Contributions

- A novel measurement scheme based on special-purpose unicast “”sandwich” probes
  - Only delay differences are measured, clock synchronization is not required
- A new, penalized likelihood framework for topology identification
  - A special Markov Chain Monte Carlo (MCMC) procedure that efficiently searches the space of topologies

Sandwich Probe Measurements

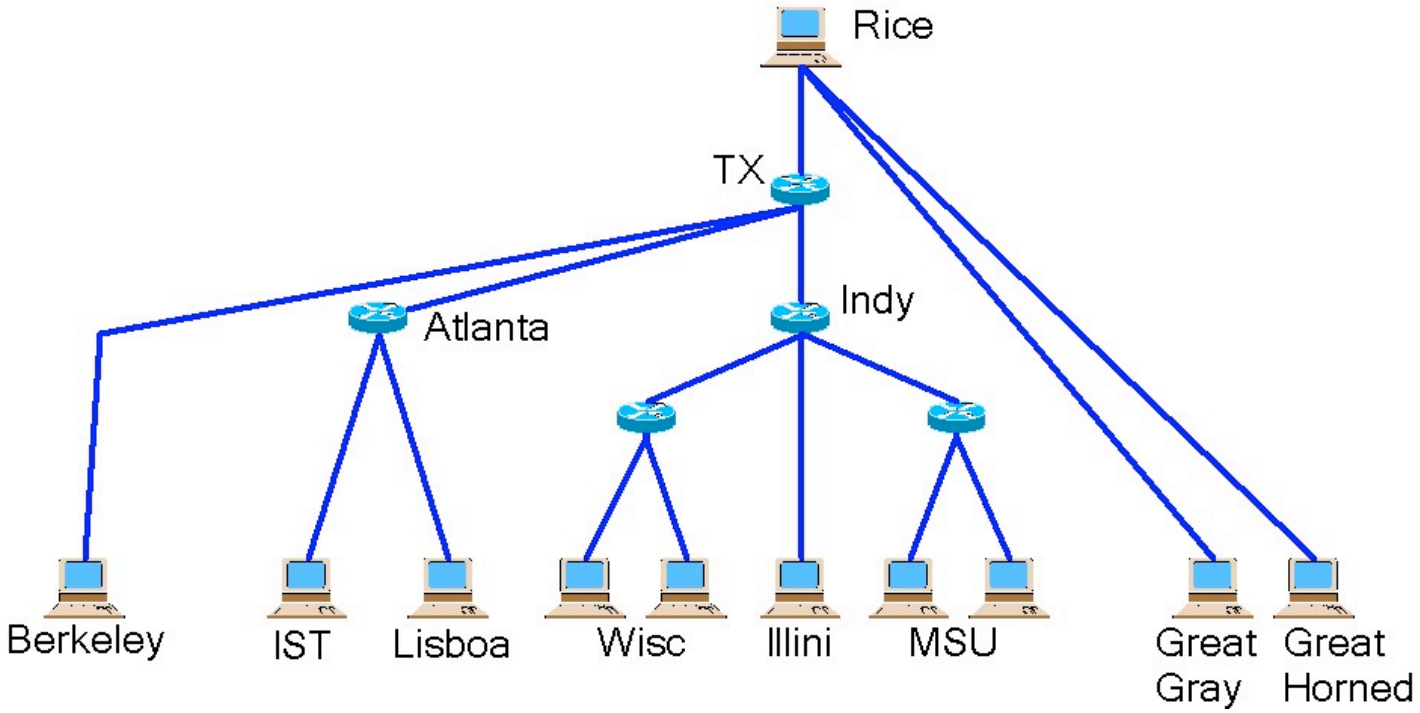
- Sandwich: two small packets destined for one receiver separated by a larger packet destined for another receiver



Sandwich Probe Measurements

- Three steps
  - End-to-end measurements are made
  - A set of metrics are estimated based on the measurements
  - Network topology is estimated by an inference algorithm based on the metric

Step 1: Measuring (**Pairwise delay measurements**)

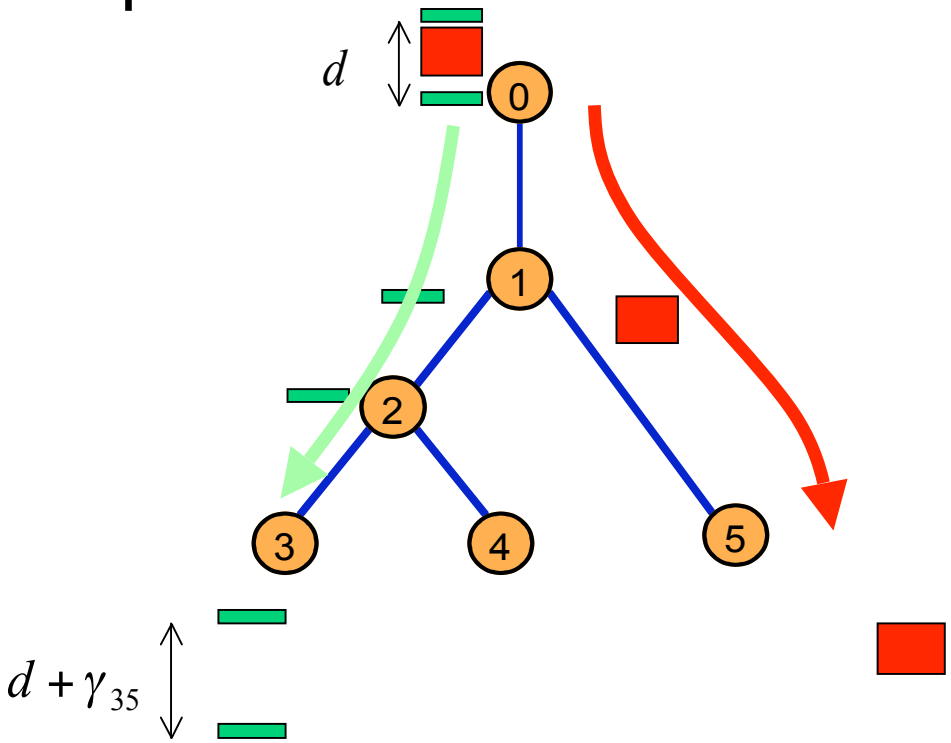


Step 1: Measuring (Continue)

- Each time a pair of receivers are selected
- Unicast is used to send packets to receivers
- Two small packets are sent to one of the two receivers
- A larger packet separates the two small ones and is sent to the other receiver
- The difference between the starting times of the two small packets should be large enough to make sure that the second one arrives the receiver after the first one
- Cross-traffic has a zero-mean effect on the measurements (d is large enough)

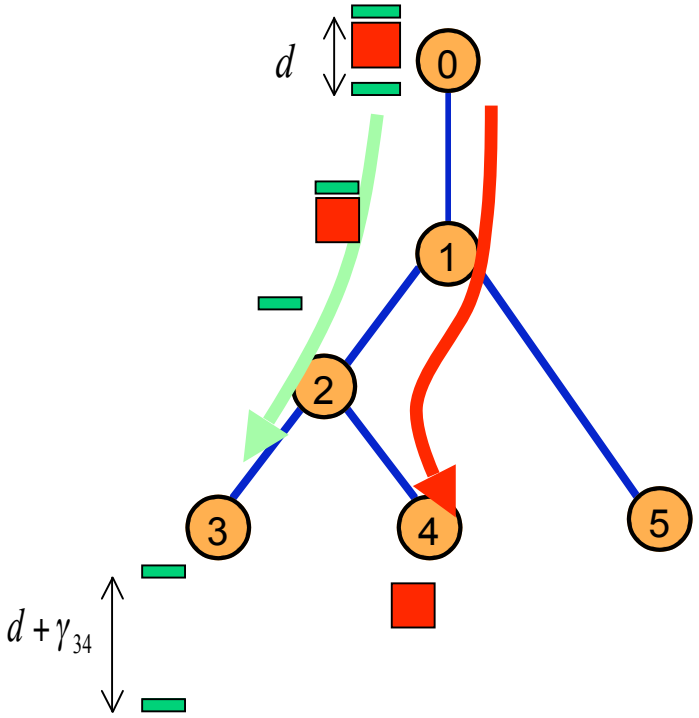
Step 1: Measuring (Continued)

- γ_{35} is resulted from the queuing delay on the shared path



Step 1: Measuring (Continued)

- More shared queues→ larger γ : $\gamma_{34} > \gamma_{35}$



Step 2: Metric Estimation

- More measurements, more reliable the logical topology identification is.
- The choice of metric affects how fast the percentage of successful identification improves as the number of measurements increases
- Metrics should make every measurement as informative as possible
- Mean Delay Differences are used as metrics
  - Measured locally
  - No need for global clock synchronization

Step 2: Metric Estimation(Continued)

- The difference between the arrival times of the two small packets at the receiver is related to the bandwidth on the portion of the path shared with the other receiver
- A metric estimation is generated for each pair of receivers.

Step 2: Metric Estimation(Continued)

- Formalization of end-to-end metric construction
  - N receivers → N(N-1) different types of measurements
  - K measurements, independent and identically distributed
  - δ(k) – difference between arrival times of the 2 small packets in the k^{th} measurement
  - Get the sample mean and sample variance of the measurement for each pair (i,j): $x_{i,j}$ and $\sigma_{i,j}^2$

(Sample mean of sample $\boldsymbol{X} = (X_1, X_2, ...)$ is

$M_n(\boldsymbol{X}) = (X_1 + X_2 + \cdots + X_n) / n$ (arithmetic mean)

Sample variance is $(1 / n)\Sigma_{i=1..n} (X_i - \mu)^2$

$E(M_n) = \mu$)

Step 3: Topology Estimation

- Assumption: tree-structured graph
- Logical links
- Maximum likelihood criterion:
  - find the true topology tree T* out of the possible trees (forest) F based on x
- Note: other ways to find trees based on common delay differences (follow references)
- Probability model for delay difference
  - Central Limit Theorem→$x_{i,j} \sim N(\gamma_{i,j}, \sigma_{i.j}/n_{i,j})$
  - $y_{i,j}$ is the the theoretical value of $x_{i,j}$
  - That is, sample mean be approximately normally distributed with mean $y_{i,j}$ and variance $s_{i.j}/n_{i,j}$
  - The larger $n_{i,j}$ is, the better the approximation is.

Step 3: Topology Estimation(Cont.)

- Probability density of x is p(x|T, μ(T)), means μ(T) is computed from the measurements x
- Maximum Likelihood Estimator (MLE) estimates the value of μ(T) that maximizes p(x|T, μ(T)), that is,
- Log likelihood of T is

$$L(\boldsymbol{x}|\mathcal{T}) \equiv \log p(\boldsymbol{x}|\mathcal{T}, \widehat{\boldsymbol{\mu}}(\mathcal{T})).$$

- Maximum Likelihood Tree (MLT) T*

T* = argmax $_{T \in F}$ $\log p(\boldsymbol{x}|\mathcal{T}, \widehat{\boldsymbol{\mu}}(\mathcal{T}))$

Step 3: Topology Estimation(Cont.)

- Over fitting problem: the more degrees of freedom in a model, the more closely the model can fit the data
- Penalized likelihood criteria:

$$L_\lambda(\boldsymbol{x}|\mathcal{T}) \;=\; \log p(\boldsymbol{x}|\mathcal{T}, \widehat{\boldsymbol{\mu}}(\mathcal{T})) \;-\; \lambda\, n(\mathcal{T})$$

  - Tradeoff between fitting the data and controlling the number of links in the tree
- Maximum Penalized Likelihood Tree(MPLT) is

$$\widehat{\mathcal{T}}_\lambda \;\equiv\; \max_{\mathcal{T}\in\mathcal{F}} L_\lambda(\boldsymbol{x}|\mathcal{T})$$

Finding the Tallest Tree in the Forest

- When N is large, it is infeasible to exhaustively compute the penalized likelihood value of each tree in F.
- A better way is concentrating on a small set of likely trees $\exp(L_\lambda(\boldsymbol{x}|\mathcal{T})) = \mathrm{e}^{-\lambda n(\mathcal{T})} p(\boldsymbol{x}\,|\,\mathcal{T},\boldsymbol{\mu}) \propto p(\mathcal{T},\boldsymbol{\mu}\,|\,\boldsymbol{x})$
- Given:

$$p(\mathcal{T}) \propto \exp(-\lambda n(\mathcal{T}))$$

- Posterior density $p(\mathcal{T},\boldsymbol{\mu}\,|\,\boldsymbol{x}) = p(\mathcal{T}) \times p(\boldsymbol{x}\,|\,\mathcal{T},\boldsymbol{\mu})$ can be used as a guide for searching F.
- Posterior density is peaked near highly likely trees, so stochastic search focuses the exploration

Stochastic Search Methodology

- Reversible Jump Markov Chain Monte Carlo
  - Target distribution: $p(\mathcal{T}, \boldsymbol{\mu} \mid \boldsymbol{x})$
  - Basic idea: simulate an ergodic markov chain whose samples are asymptotically distributed according to the target distribution
  - Transition kernel: transition probability from one state to another
  - Moves: birth step, death step and μ-step

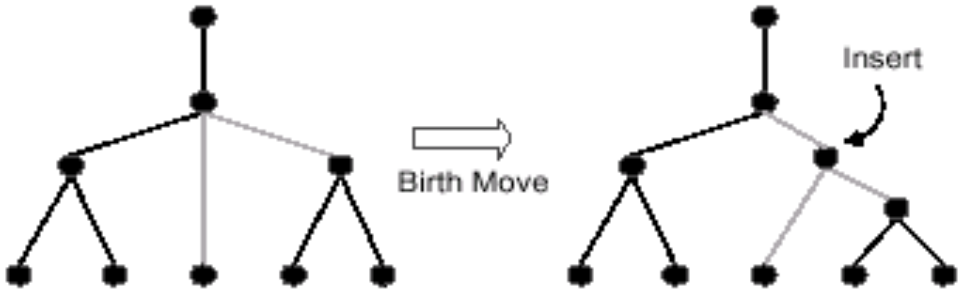
Birth Step

- A new node l* is added→ extra parameter μ_l^*
- The dimension of the model is increased
- Transformation (non-deterministic)

$\mu_l^* = r \times \min(\mu_c(l,1), \mu_c(l,2))$

$\mu'_c(l,1) = \mu_c(l,1) - \mu_l^*$

$\mu'_c(l,2) = \mu_c(l,2) - \mu_l^*$

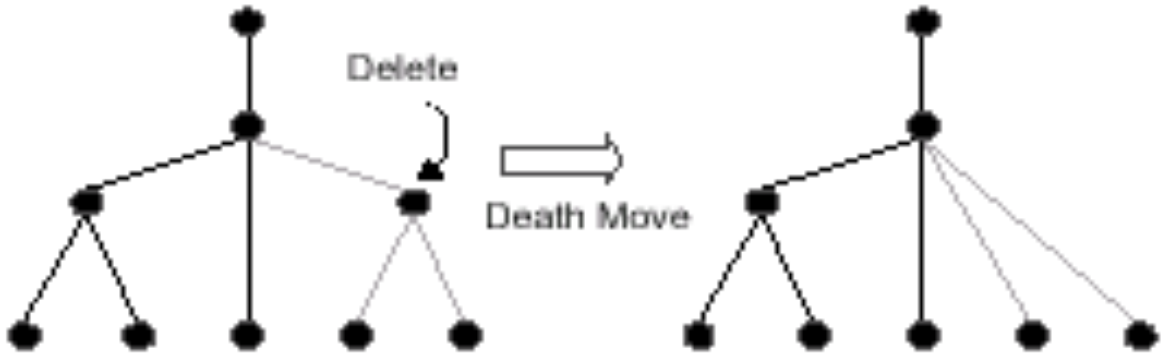


Death Step

- A node l* is deleted
- The dimension of the model is reduced by 1
- Transformation (deterministic)

$\mu_c(l,1) = \mu'_c(l,1) + \mu_{l^*}$

$\mu_c(l,2) = \mu'_c(l,2) + \mu_{l^*}$



μ –step

- Choose a link l and change the value of μ_l
- New value of μ_l is drawn from the conditional posterior distribution

The Algorithm

- Choose a starting state s_0
- Propose a move to another state s_1
  - Probability = $$\min\left\{1, \frac{p(\mathcal{T}_1, \boldsymbol{\mu}_1 \mid \boldsymbol{x}) q(s_0 \mid s_1)}{p(\mathcal{T}_0, \boldsymbol{\mu}_0 \mid \boldsymbol{x}) q(s_1 \mid s_0)} \times \mathcal{J}_{f(s_1, s_0)}\right\}$$
- Repeat these two steps and evaluate the log-likelihood of each encountered tree
- Why restart?

Penalty parameter

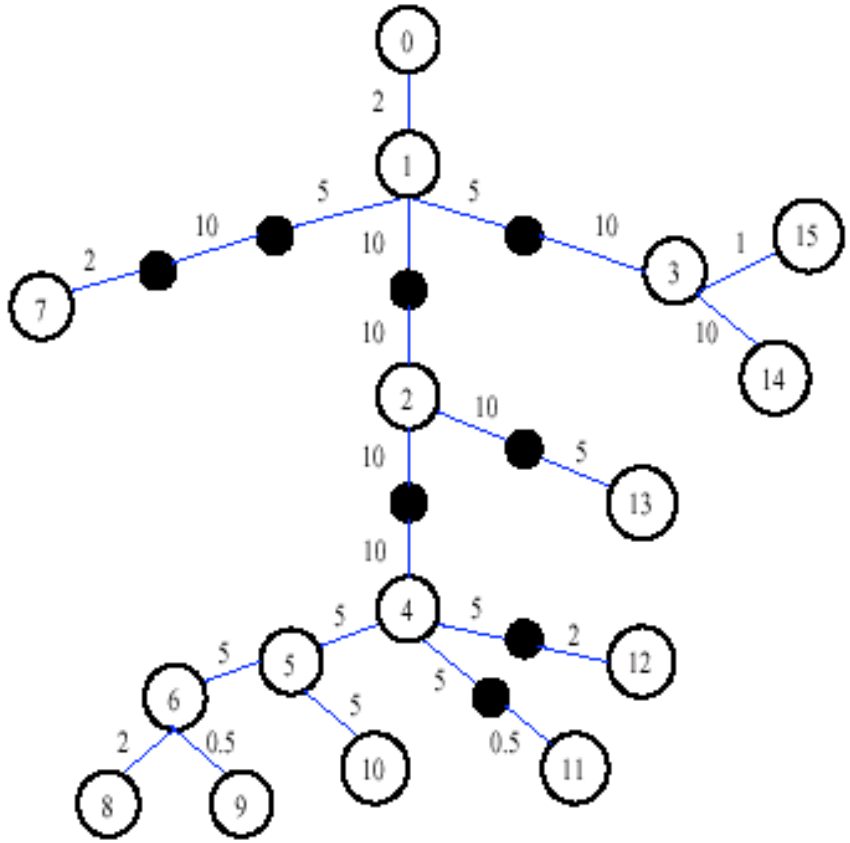
- Penalty = $1/2\log_2 N$
- N: number of receivers

Simulation Experiments

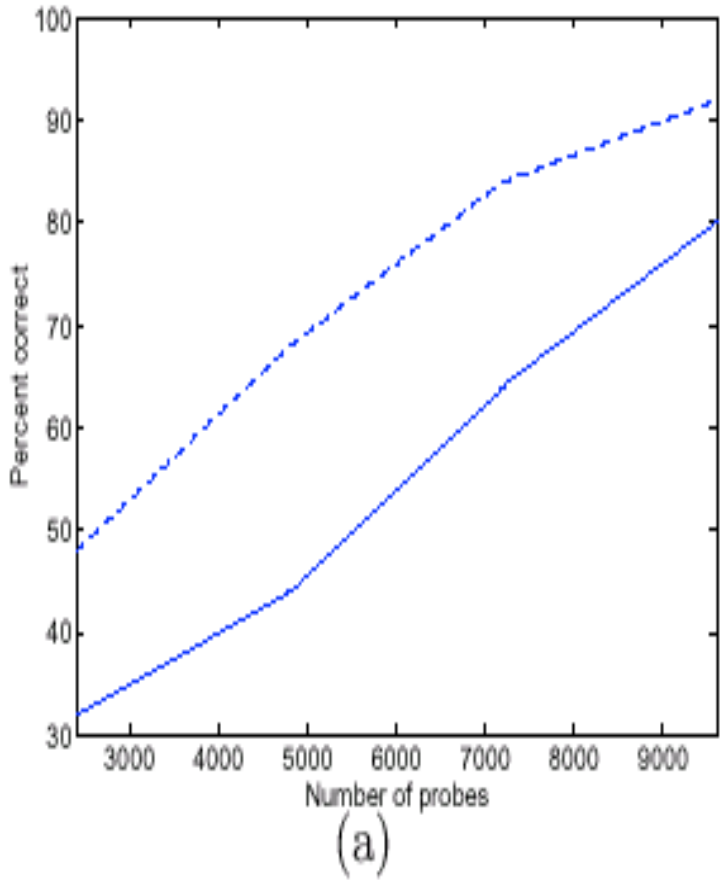
- Compare the performance of DBT(Deterministic Binary Tree) and MPLT
- Penalty = 0 (both will produce binary trees)
- 50 probes for each pair in one experiment, 1000 independent experiments
- When the variability of the delay difference measurements differ on different links, MPLT performs better than DBT
- Maximum Likelihood criteria can provide significantly better identification results than DBT

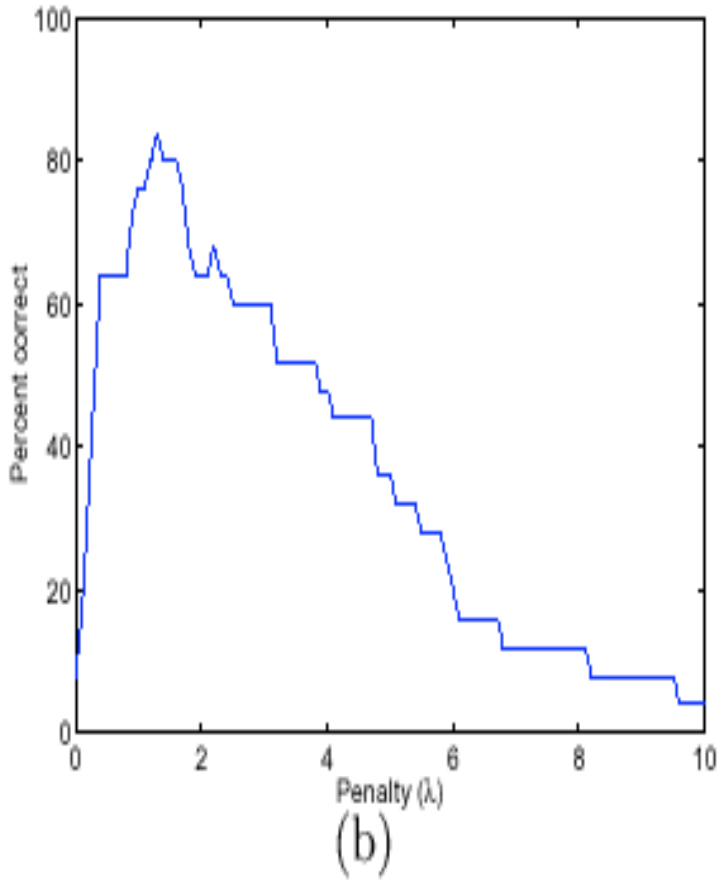
ns Experiment

- Topology used for the experiment



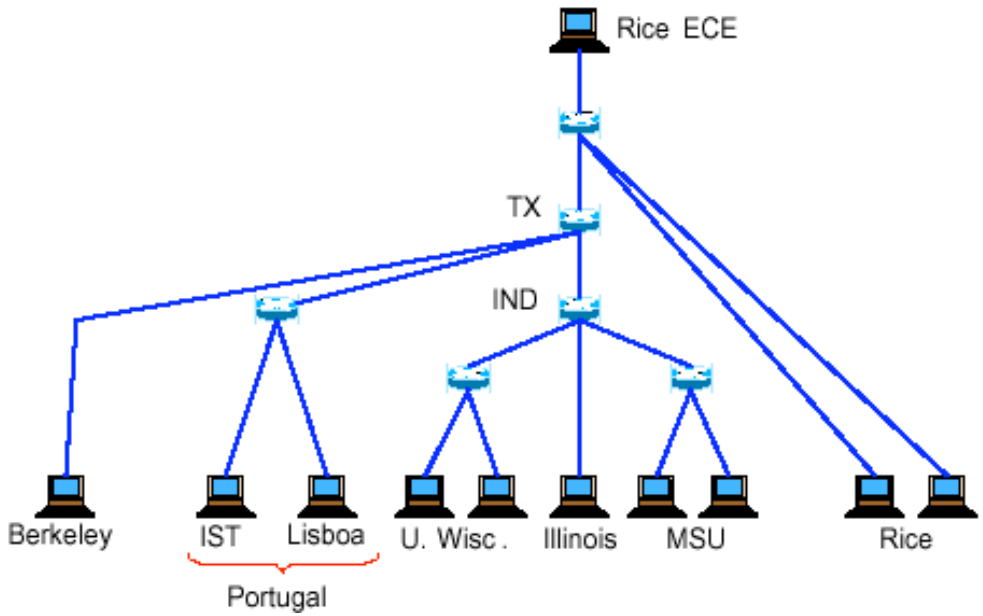
Experiment Results





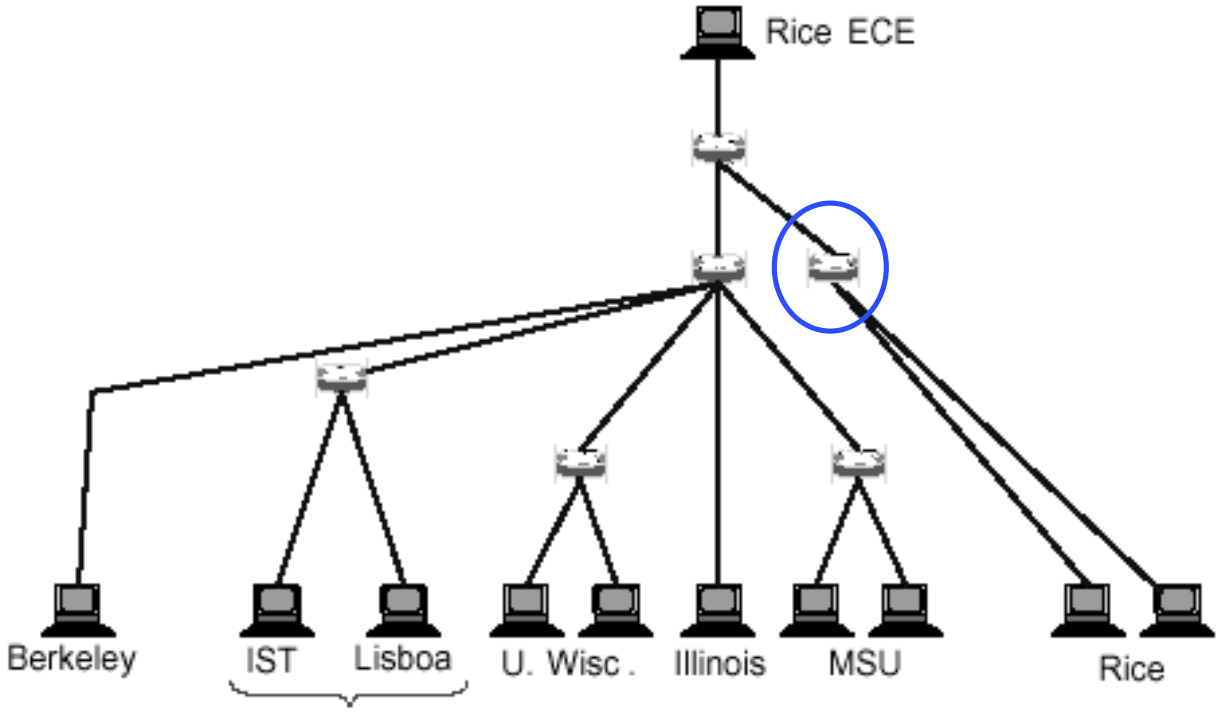
Internet Experiment

- Source host: data collection and inference
- Receivers: a low overhead receiver task
- 8 minutes/experiment, 6 independent experiments
- 1 sandwitch probe / 50ms
- Penalty = 1.7
- topology



Experiment Result

- Estimated topology



Conclusions and Future work

- Conclusions:
  - Delay-based measurement without the need for synchronization
  - MCMC algorithm to explore forest and identify maximum (penalized) likelihood tree
  - Foundation for multi-sender topology identification
  - Localization of layer-two elements
- Future work
  - Adaptive methods for selecting penalty parameter
  - Adaptivity in the probing scheme

Extra Credit

- Log into planetLab nodes
  - Use SSH with class-provided key
- Pick a set of hosts to perform the experiment
  - A set of 2 given hosts posted for the class
    - You pick 3 more:
      - East Asia -> North America
      - North America -> Europe
      - Europe -> East Asia
- Generate & record a 1 minute ping sequence with different δ (6 in all)
    - 1, 5, 15, 50, 100, 200 ms

Extra Credit (cont)

- For each trace (30 in all):
  - Plot the phase plot
  - Find the equation of the line $y = x + (p/\mu - \delta)$
  - Plot the distribution plot
  - Find the first three peaks; find b_n
- For a set of traces between 2 hosts:
  - Provide the table of ulp, clp, plg

Extra Credit (cont)

- What to hand in:
  - Short paragraph describing the experiment, and problems you had
  - Phase plots + equations
  - Distribution plots + positions of peaks, Bn
  - Probability table
  - Label plots with source, destination host names, time of experiment, length of experiment