Network measurements and monitoring

20.2.2008

Matti Siekkinen [siekkine@ifi.uio.no]

University of Oslo

Part I: The Basics

- Background and motivation
- Basic measurement principles

Part II: Targets and Techniques

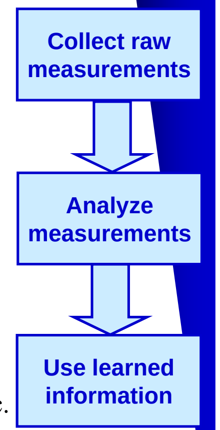
- Infrastructure measurements
- Traffic measurements
- Applications measurements

Part I: The Basics

- Background and motivation
- Basic measurement principles
  - Passive vs. active
  - On-line vs. off-line
  - Anonymization
  - Where can we collect measurements?
  - What can we measure?

Measuring/monitoring networks

- Obtaining raw measurements
  - Traffic traces etc.
  - Input for inference/analysis process
- Inference / Analysis
  - Usually also considered as part of the measurement process
  - E.g. learn that a router is congested
- Learning via inference/analysis drives other operations
  - Input for network management, protocol/application design, etc.
  - E.g. reroute part of traffic to ease the load of the congested router



Why do we need to measure networks?

- Provisioning networks
  - Over provisioning costs money
  - Under provisioning makes customers complain
  - Measurements help determining the suitable tradeoff
- Managing networks
  - Network and traffic engineering
    - Load balancing
    - Capacity planning & optimizing
    - Identify bottlenecks
  - Identify misconfigured devices
    - E.g. routers that advertise false routes
  - Autonomic networks
    - Self-* (configuring, optimizing, healing, protecting) properties in networking
    - Monitoring is imperative

Why do we need to measure networks?

- Crucial input for future development
  - Services and protocols
  - Application, TCPv947…
  - Modeling (traffic, mobility, user behavior, …)
    - Input for simulators
  - Perform empirical studies
    - Simulations are not always sufficient
- Security related issues
  - Protect and defend against malicious activities

Why is it challenging?

- Few built-in measurement mechanisms
  - Today's networks are mostly IP networks
  - Network elements are simple
  - Intelligence lies at the edges
  - ⇨ Need to use complex end-to-end methods to measure simple things (e.g. link capacity)
- The targets are constantly moving
  - Dominating services in the Internet
    - before: Web and file transfer (FTP)
    - now: P2P file sharing, Skype, social networks (MySpace, YouTube, Facebook, …)
    - tomorrow: ?
  - Internet access link capacities at home
    - a few years ago (in Europe): 512 Kbit/s
    - now: > 10Mbit/s
  - More and more mobility
  - New kinds of networks: e.g. MANETs, VANETs, sensor networks, DTNs
  - Hard to characterize "typical" behavior

Why is it challenging?

- Scale of networks can be very large
  - Traffic volumes
  - Number of nodes
  - ⇨ Measurement techniques need to be scalable too
- Data can be sensitive
  - Legal issues: privacy
    - Paul Ohm et al.: *Legal Issues Surrounding Monitoring During Network Research* (Internet Measurement Conference, October 2007)
  - Business: ISPs are reluctant to disclose any information

Basic measurement principles

- Passive vs. active measurements
- On-line vs. off-line measurements
- Anonymization
- Where can we collect measurements?
- What can we measure?

Measurements: Passive vs. active

- Passive
  - Simply record what you observe
  - E.g. for measuring traffic characteristics or application behaviour
  - ☺ Measures the real thing (no artificial component)
  - ☺ Does not perturb the network
  - ☹ No control over the measurement process
- Active
  - Inject packets to the network for measurement purposes
  - Especially usable for measuring the infrastructure
  - ☺ Full control over the measured traffic
  - ☹ Need often access to more than one measurement point at strategic locations
  - ☹ Can perturb the network
- Fused measurements
  - Combine active and passive approaches

Measurements: On-line vs. off-line

- On-line
  - Perform (at least a part of) the analysis on the observed data in a real-time manner
  - Often necessary when handling very large amounts of data
    - E.g. monitoring traffic of one Abilene Intenet2 backbone link (OC-192, 10 Gbit/s link) produced >8 MBytes/s of uncompressed packet headers
  - ☺ Data reduction, don't need to store everything
  - ☺ Results right away, can react immediately
  - ☹ Efficient solutions can be very complex to build
  - ☹ Do not necessarily have all the raw data for later analysis
- Off-line
  - Record data into persistent storage and analyze later
  - ☺ Possible to run complex time-consuming analysis
  - ☺ Simple and cheap solutions exist
  - ☹ Not applicable for time critical scenarios
  - ☹ Storage can become an issue

Measurements: Anonymization

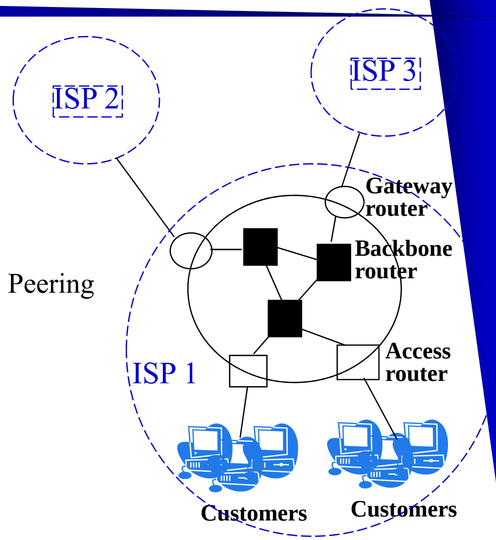
- Sharing measurements is good
  - Good scientific practice
    - Repeatable experiments, result verification…
  - Enable access for more "players" to the field
  - Everybody benefits, usually…
- Measurements can contain sensitive information
  - About individuals ⇒ privacy concerns
  - About organizations ⇒ competition
  - Complicates the sharing of measurements
- Anonymization helps to overcome obstacles in sharing measurements
  - Replace sensitive information with "bogus" information

Measurements: Anonymization

- What is anonymized? For example:
  - Packet payloads in traffic data
    - Passwords, nature of content…
  - IP addresses
    - Sometimes preserve network structure information
  - Traffic volumes
- Techniques
  - Lossless transformation
  - Semi-lossy transformation
    - E.g. keep only first 24-bits of an IP address
  - Lossy transformation
    - E.g. map strings to numbers

Where can we collect measurements?

- Vantage points in a network
  - Client
  - Backbone
    - Backbone router
  - Network entry points
    - Access router, Gateway router, Peering router



What can we measure?

- End device
  - PC, PDA, cell phone…
  - Traffic measurements with e.g. tcpdump
  - Application level measurements
  - End-to-end infrastructure measurements
- Router
  - Traffic measurements with Netflow
  - Infrastructure measurements by recording routing table entries
- Wireless sniffer
  - E.g. a strategically placed laptop listening the traffic of a wireless nw
  - Transport, network, and link layer information
- Link
  - Tap a link via a hub or optical splitter
  - Collect packet traces

Part II: Targets and Techniques

- Infrastructure measurements
  - Topology discovery
    - Example: Doubletree
  - Network coordinates
    - Example: Vivaldi
  - Bandwidth measurements
    - Example: CapProbe
- Traffic measurements
  - Traffic matrices
  - TCP
  - Anomaly detection
- Applications measurements
  - P2P
  - Web
  - Social networks
    - Example: YouTube

Topology discovery

- The art of finding out how the network is laid out
  - Not trivial knowledge in large scale networks
- Why do this?
  - Realistic simulation and modeling of the Internet
  - Correctness of network protocols typically independent of topology
  - Performance of networks critically dependent on topology
    - e.g., convergence of route information
  - Modeling of topology needed to generate test topologies

Topology discovery

- Router-level topologies
  - Reflect physical connectivity between nodes
  - Inferred using with e.g. traceroute
- AS graphs
  - Peering relationships between providers/clients
  - Inferred from inter-domain routers' BGP tables

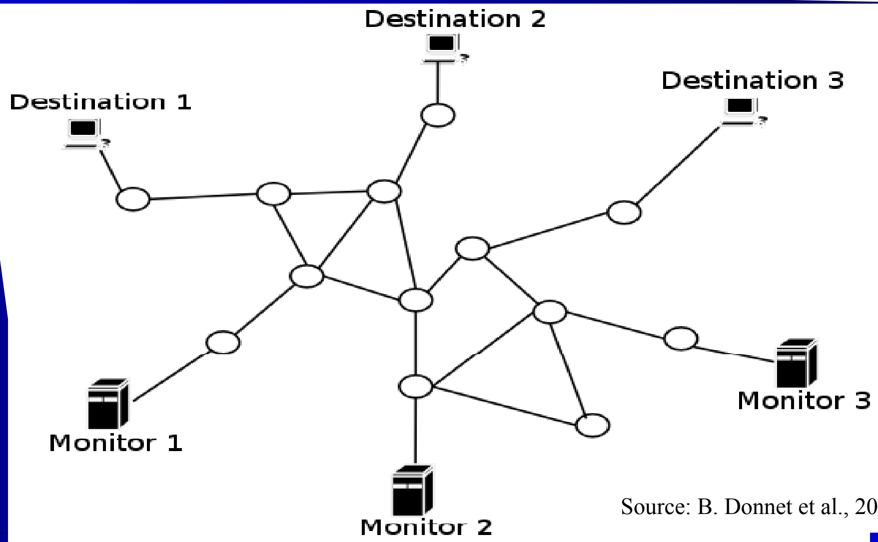
Topology discovery: some examples

- SNMP
  - Query hosts recursively
  - Access usually restricted => works only locally
- Skitter
  - ICMP ECHO-REQUEST probes (~ traceroute) with increasing TTL from 30-40 monitors to measure delay and IP path
  - Gather actively used IP addresses from a number of sources
    - Bbone packet traces, NeTraMet traces, NetGeo, CAIDA website hits…
- Oregon Route Views project
  - Provides real-time AS-level information about the global routing system
  - Operating since 1995

Large Scale Topology discovery: Doubletree

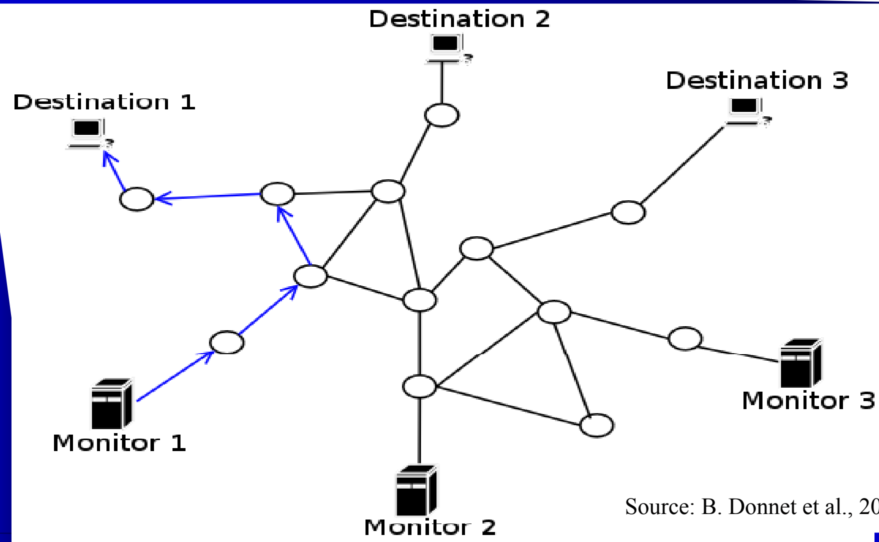
- Benoit Donnet et al.: *Efficient Algorithms for Large-Scale Topology Discovery.* (SIGMETRICS 2005)
- Probing scheme based on traceroute
- Problem: more monitors means more load on
  - network resources
  - destinations
- Two types of scaling barriers
  - Intra-monitor redundancy
  - Inter-monitor redundancy

Doubletree: Intra-monitor Redundancy



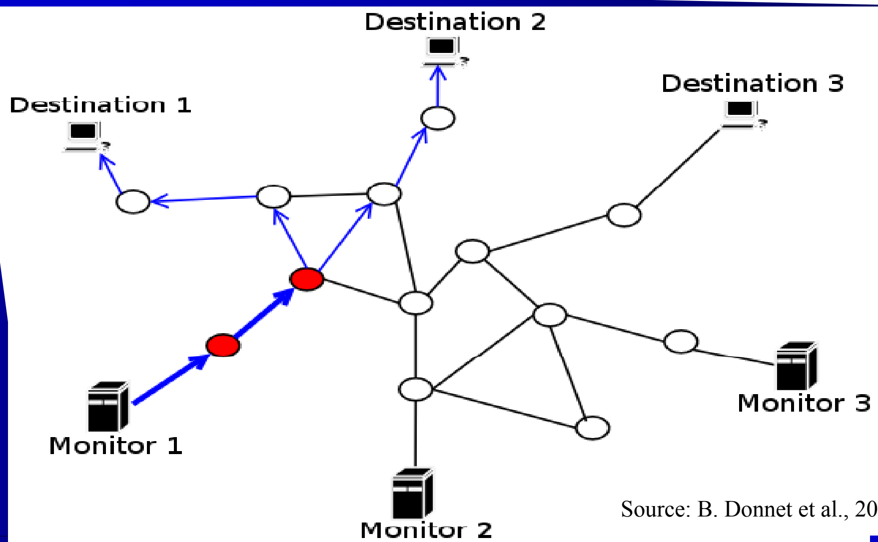
Source: B. Donnet et al., 2005

Doubletree: Intra-monitor Redundancy



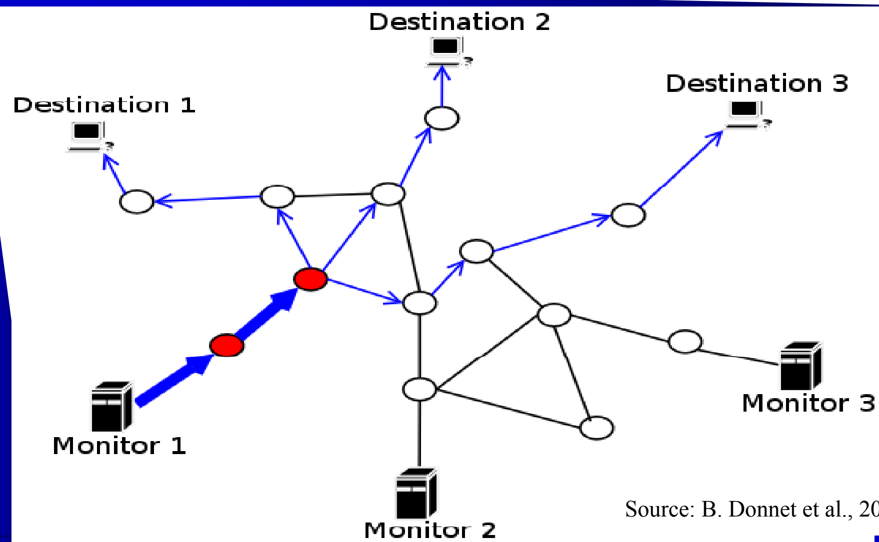
Source: B. Donnet et al., 2005

Doubletree: Intra-monitor Redundancy



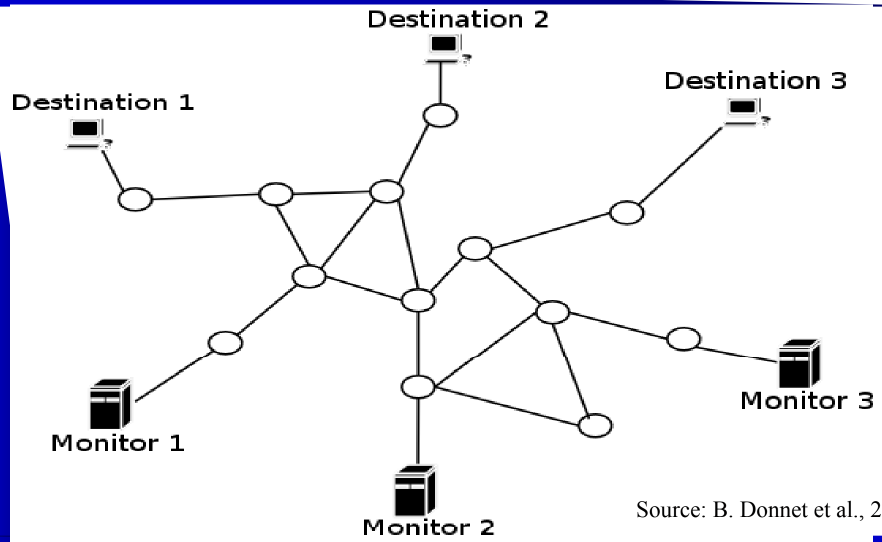
Source: B. Donnet et al., 2005

Doubletree: Intra-monitor Redundancy



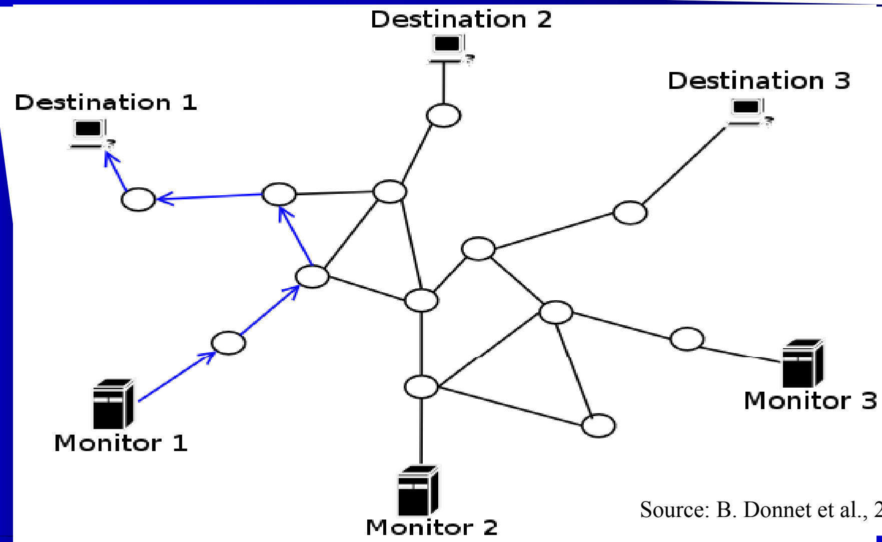
Source: B. Donnet et al., 2005

Doubletree: Inter-monitor Redundancy



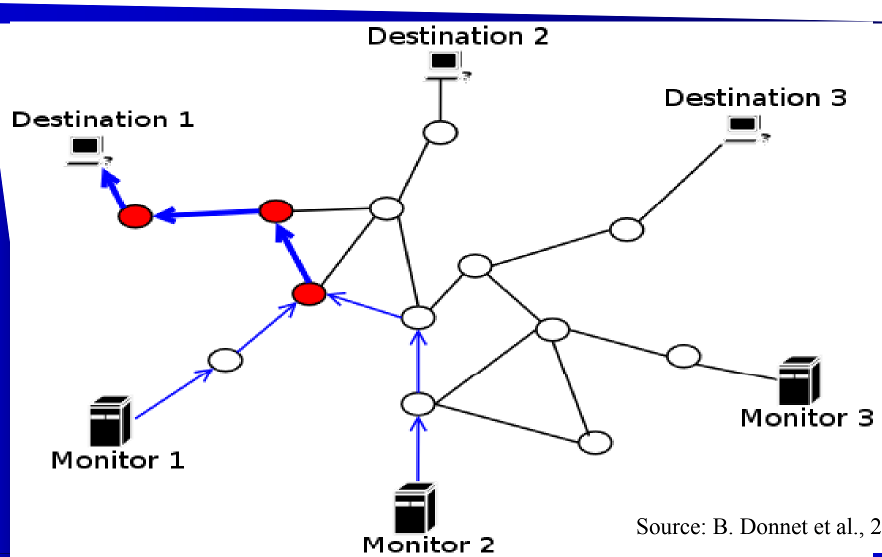
Source: B. Donnet et al., 2005

Doubletree: Inter-monitor Redundancy



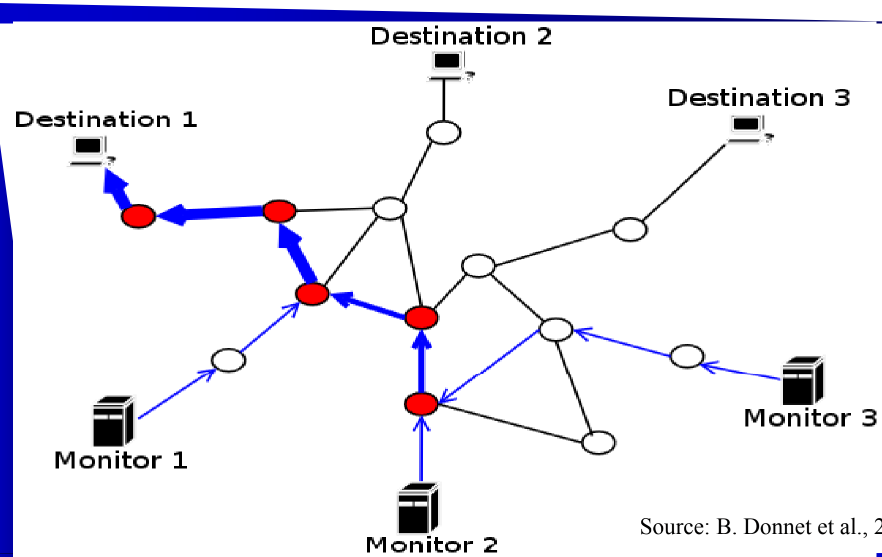
Source: B. Donnet et al., 2005

Doubletree: Inter-monitor Redundancy



Source: B. Donnet et al., 2005

Doubletree: Inter-monitor Redundancy

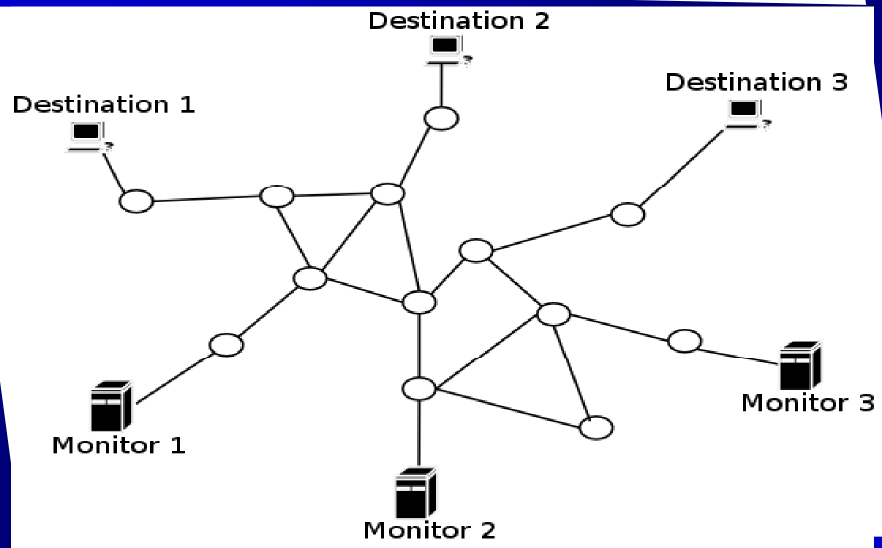


Source: B. Donnet et al., 2005

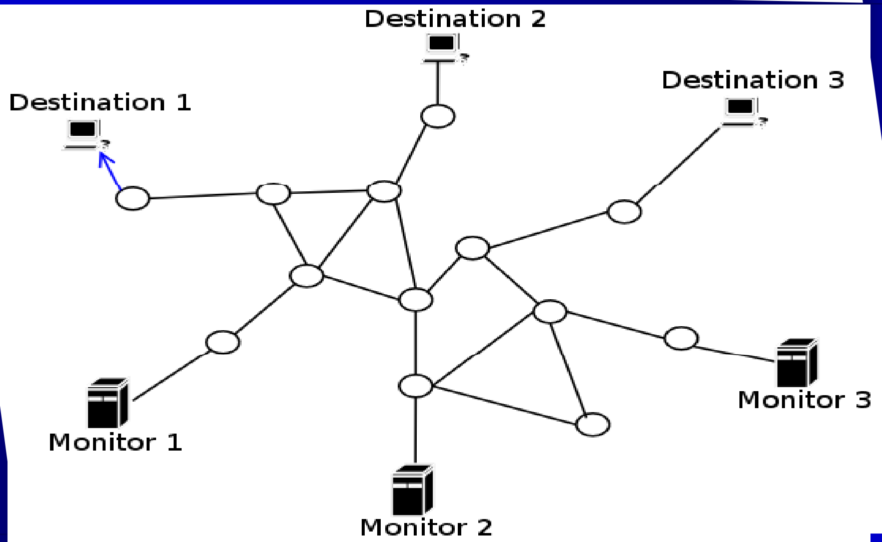
Doubletree: Two probing schemes

- Two redundancies (i.e. inter and intra) suggest two different probing schemes
  - They are based on the *tree-like structure* of routes
- Intra-monitor
  - *monitor-rooted* tree
- Inter-monitor
  - *destination-rooted* tree

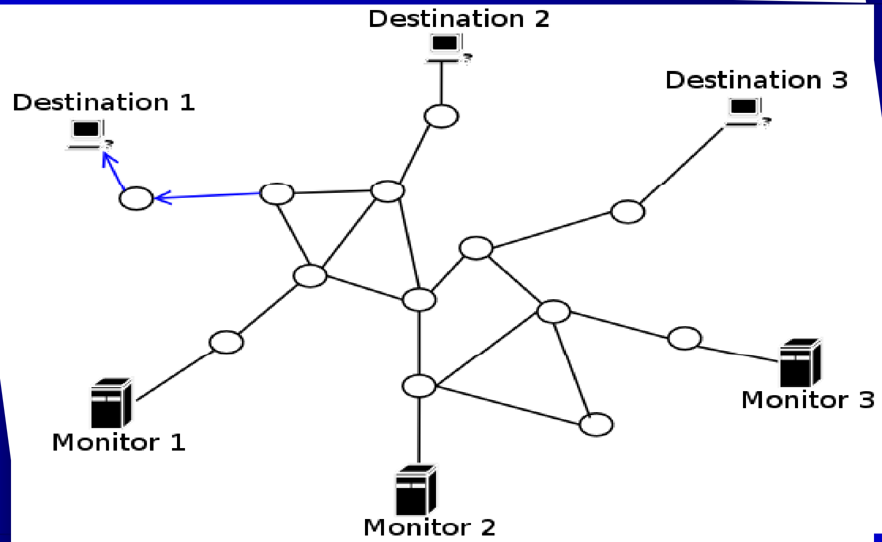
Doubletree: Monitor-rooted Tree



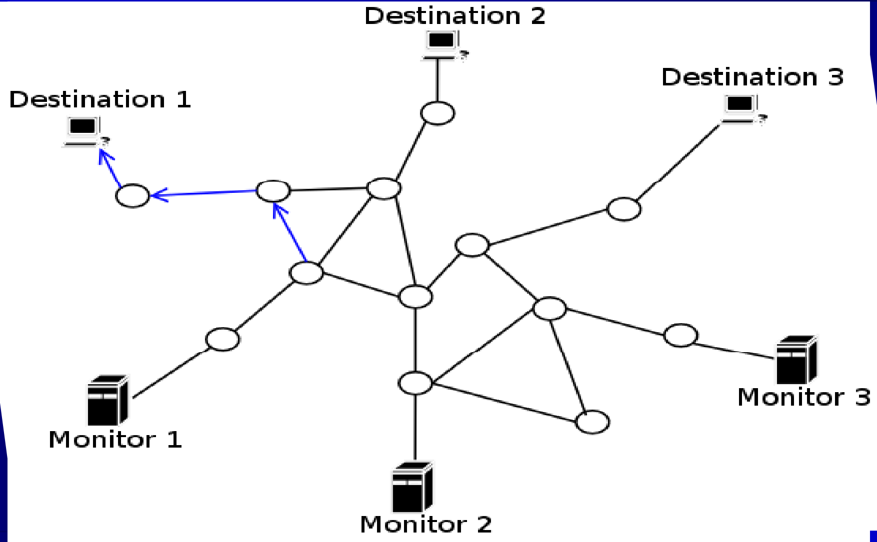
Doubletree: Monitor-rooted Tree



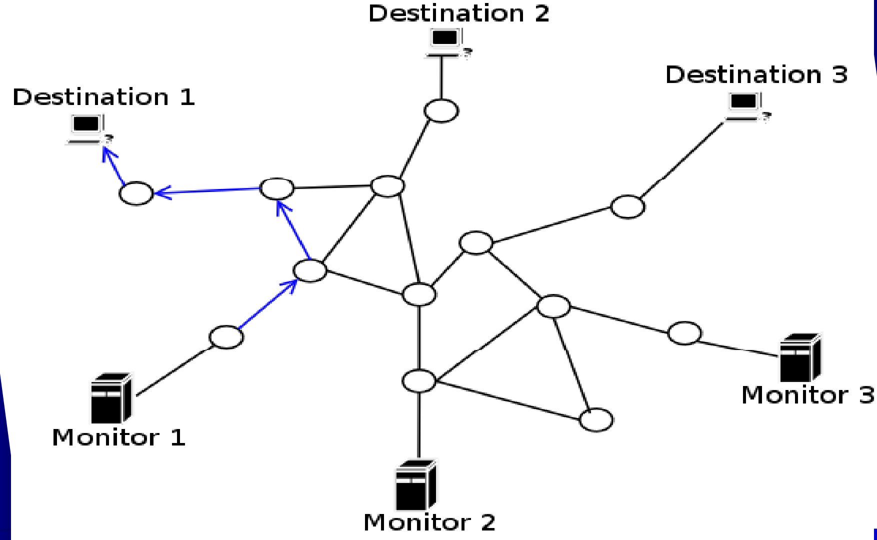
Doubletree: Monitor-rooted Tree



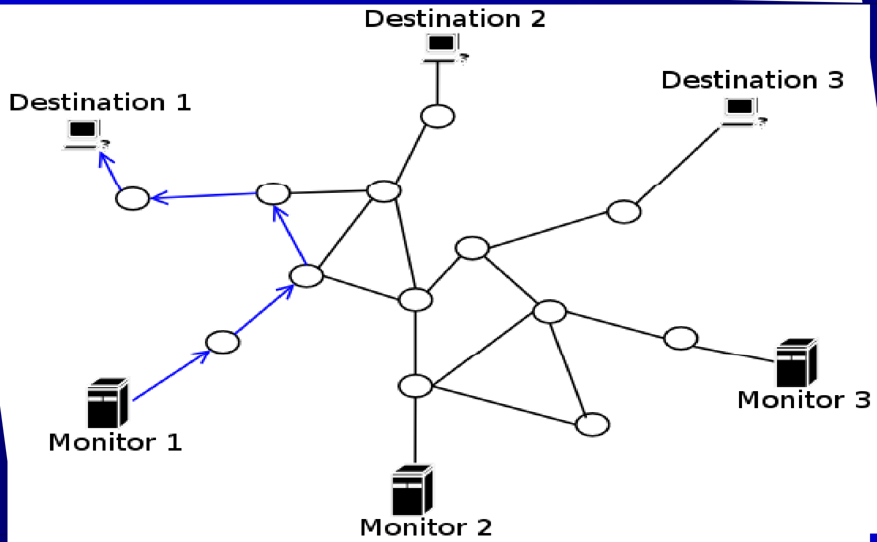
Doubletree: Monitor-rooted Tree



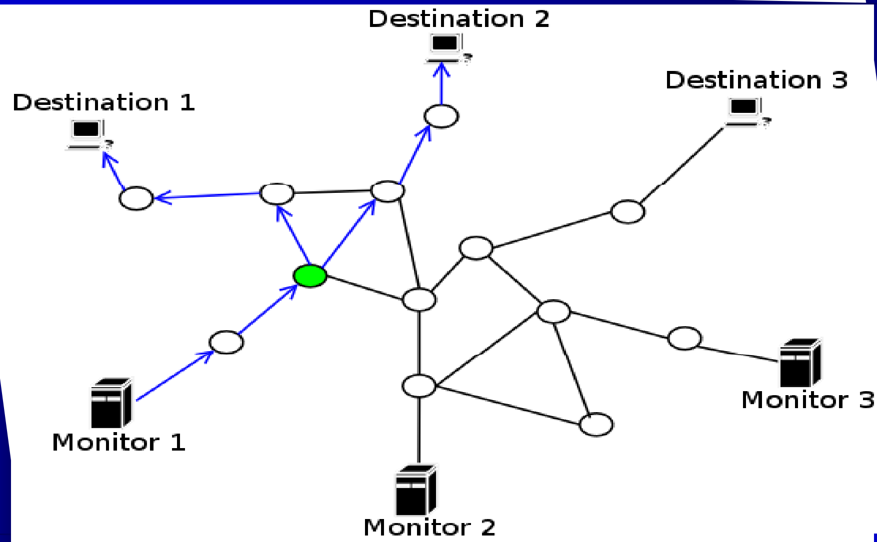
Doubletree: Monitor-rooted Tree



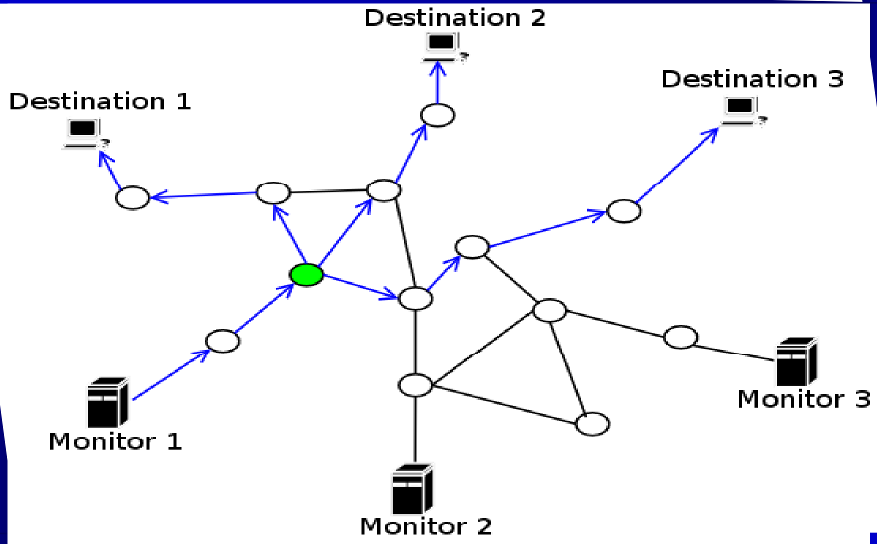
Doubletree: Monitor-rooted Tree



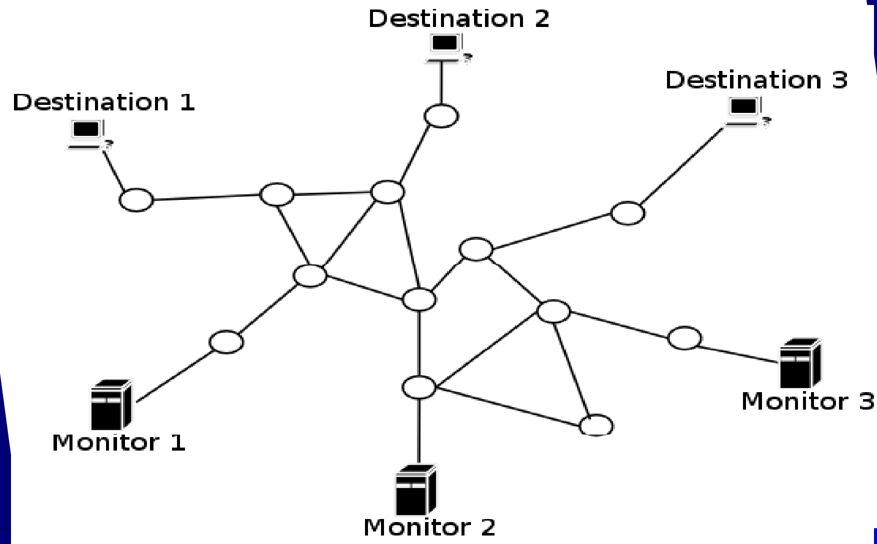
Doubletree: Monitor-rooted Tree



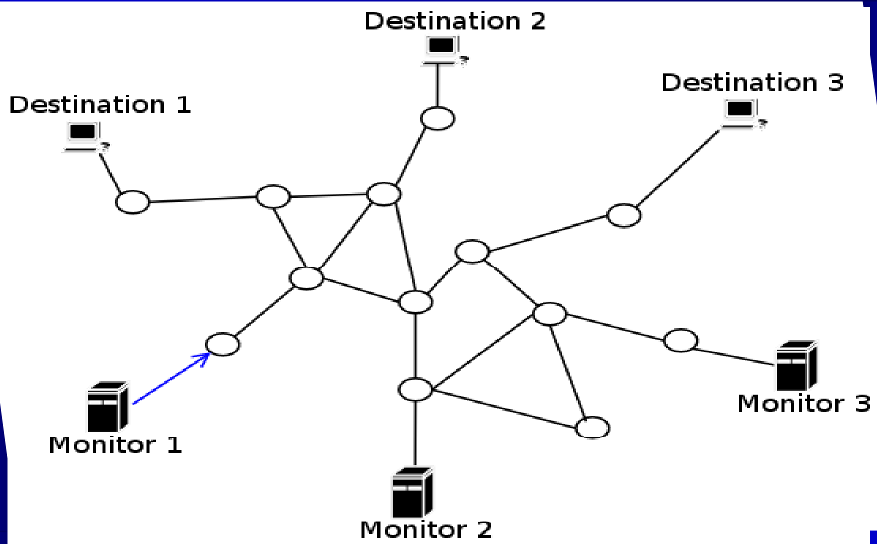
Doubletree: Monitor-rooted Tree



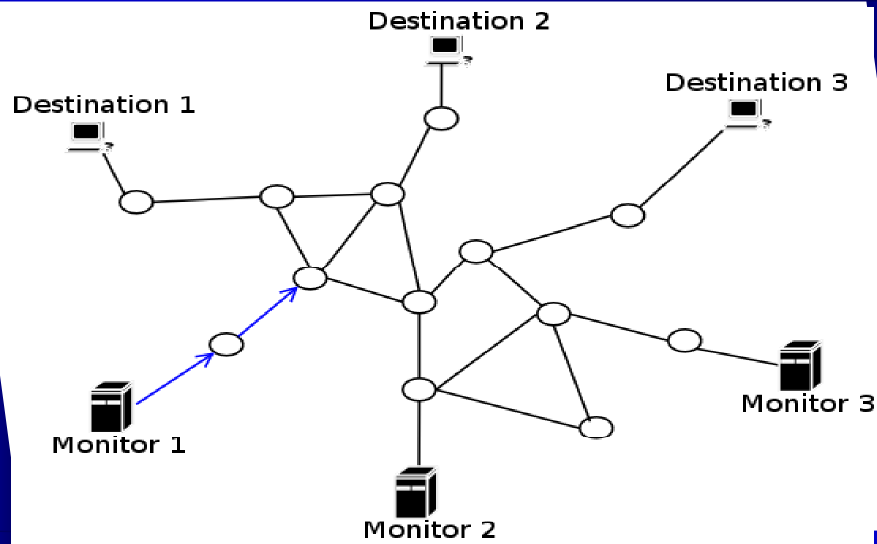
Doubletree: Destination-rooted Tree



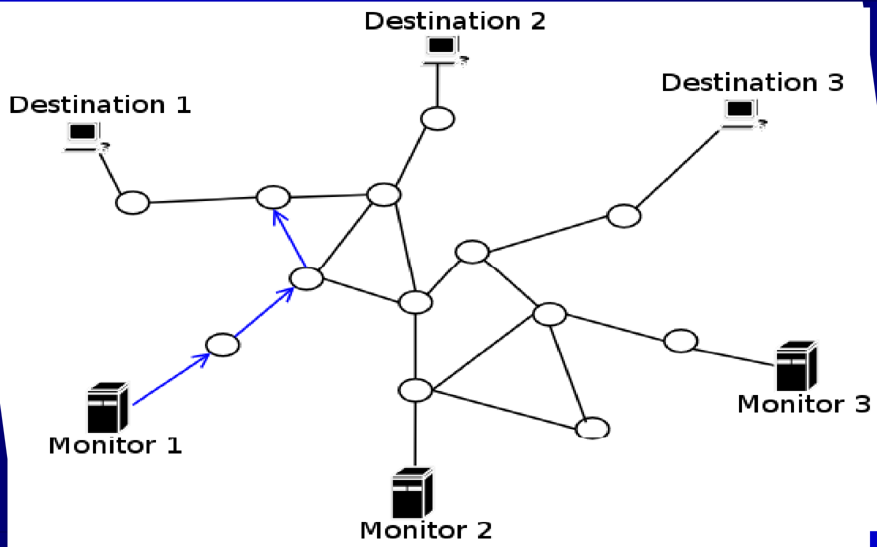
Doubletree: Destination-rooted Tree



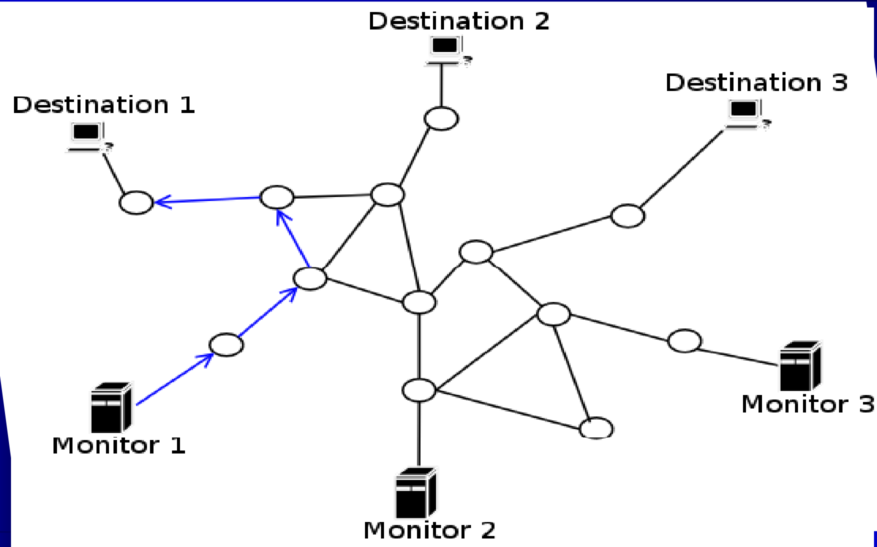
Doubletree: Destination-rooted Tree



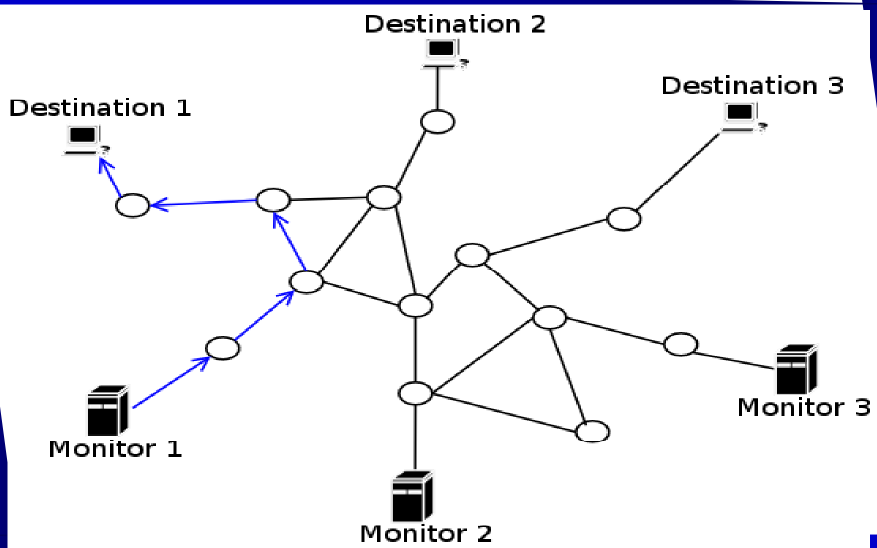
Doubletree: Destination-rooted Tree



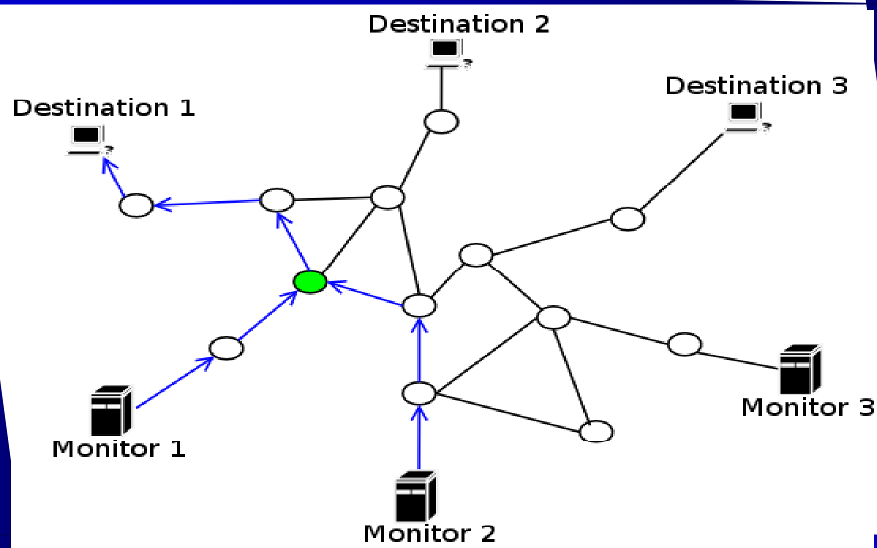
Doubletree: Destination-rooted Tree



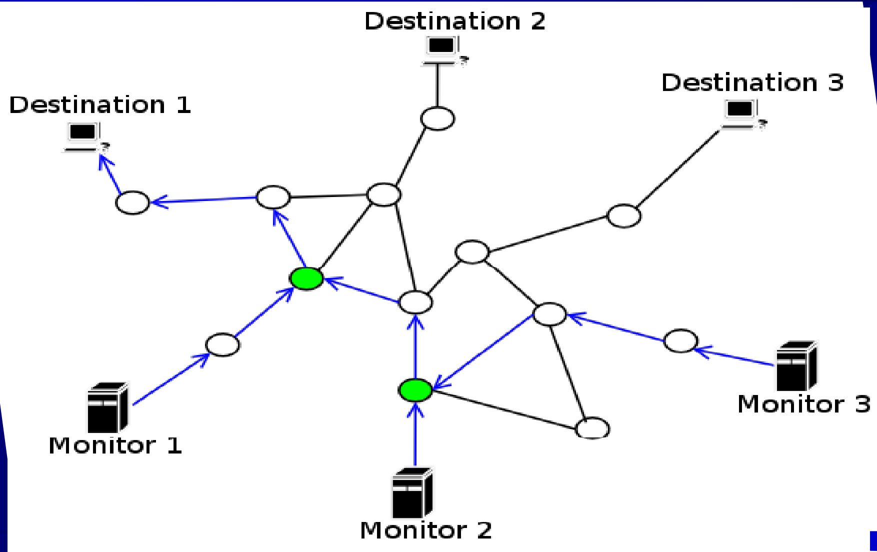
Doubletree: Destination-rooted Tree



Doubletree: Destination-rooted Tree



Doubletree: Destination-rooted Tree



Doubletree: the algorithm

- Merge the two probing schemes
  - Select some intermediate hop *h*
  - Forward probing from *h*
  - Backward probing from *h-1*
- Each monitor uses *stop sets:* {(interface, root)}
  - *Local Stop Set* B: {interface}
    - Backward probing
  - *Global Stop Set* F: {(interface, destination)}
    - Forward probing
    - Shared between monitors

  Don't need to keep track of the whole tree structure

Doubletree: the algorithm

- Parameter *p* determines *h*
  - *p* is probability to hit a responding destination at hop count *h*
  - Sets the trade off in reduction between intra- and inter-redundancy
- Can achieve good performance
  - Measurement load reduction up to 76%
  - Interface and link coverage above 90%

Part II: Targets and Techniques

- Infrastructure measurements
  - Topology discovery
    - Example: Doubletree
  - → Network coordinates
    - Example: Vivaldi
  - Bandwidth measurements
    - Example: CapProbe
- Traffic measurements
  - Traffic matrices
  - TCP
  - Anomaly detection
- Applications measurements
  - P2P
  - Web
  - Social networks
    - Example: YouTube

Network coordinates

- Express the communication latency, the "distance", in virtual coordinates
- Synthetic coordinate systems
  - Predictions, no exact coordinates
  - Due to triangular inequality violation, for instance
- Enables predicting round-trip times to other hosts without having to contact them first
  - Useful in selecting a mirror server or peers in P2P systems
- Traditional approach:
  1. Select a subset of hosts for reference points (RP)
     - Create the origin of the coordinate system
  2. Measure round-trip-time (distance) between RPs
  3. Calculate coordinates for each RP
  4. Measure RTT between host and RPs
  5. Calculate coordinates for the host
- Different proposed techniques for steps 1,3 and 5
- Reference points = landmarks, lighthouses, beacons

Example: Vivaldi

- Frank Dabek et al.: *Vivaldi: A Decentralized Network Coordinate System.* (SIGCOMM 2004)
- Properties
  - Completely decentralized
    - No landmarks
  - Efficient with low overhead
    - Minimal probe traffic

  ⇨ Scales to a large number of hosts
  - Adaptive to changing network conditions

Vivaldi: Minimize Prediction Error

- Let L_{ij} be the actual RTT between nodes *i* and *j*, and x_i be the coordinates assigned to node *i*.
- The errors in the coordinates can be characterized using a squared-error function:

$$E = \sum_i \sum_j (L_{ij} - \| x_i - x_j \|)^2$$

Goal is to minimize this error

Vivaldi: String system analogy

- Imagine strings that connect nodes
  - Rest length is the know RTT: L_{ij}
  - Current length is distance in coordinate system: $\| x_i - x_j \|$
  - Force vector that the spring between nodes i and j exerts on node i:

$$F_{ij} = (L_{ij} - \| x_i - x_j \|) \times u(x_i - x_j)$$

  - Potential energy of spring is proportional to the square of the displacement from its rest length
  - Sum of the potential energies over all springs is exactly the error function we want to minimize

Vivaldi: Algorithm - the simple version

// Node i has measured node j to be rtt ms away,
// and node j says it has coordinates x_j.
simple_vivaldi(rtt, x_j) ⟵ Called for each new RTT measurement
// Compute error of this sample. (1)
$e = rtt - \|x_i - x_j\|$
// Find the direction of the force the error is causing. (2)
$dir = u(x_i - x_j)$
// The force vector is proportional to the error (3)
$f = dir \times e$
// Move a a small step in the direction of the force. (4)
$x_i = x_i + \delta \times dir$

- Constant time step δ

Vivaldi: An Adaptive Timestep

- The rate of convergence is governed by the δ timestep
  - A small δ causes slow convergence
  - A large δ causes oscillation
- Vivaldi varies δ depending on how certain the node is about its coordinates

$$\delta = c_c \times \frac{\text{local error}}{\text{local error} + \text{remote error}}$$

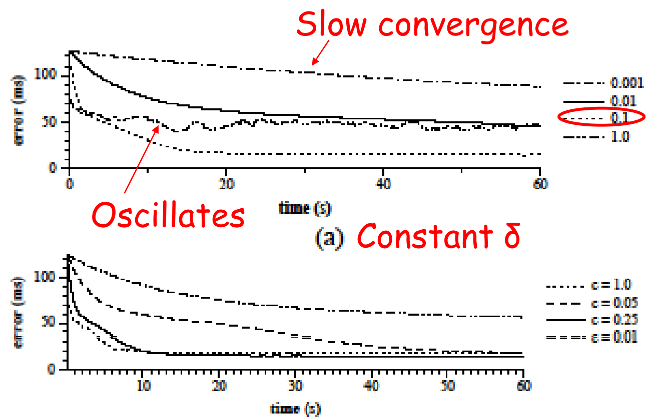
Take into account also confidence of the remote node

Each node compares new measured RTT sample with predicted RTT, and maintains local error

Vivaldi: Evaluation - Setup

- Used a packet-level network simulator running with RTT data collected from the Internet
  - PlanetLab data set: 192 hosts on the PlanetLab network testbed
  - King data set: 1740 Internet DNS servers

Vivaldi: Evaluation - Convergence



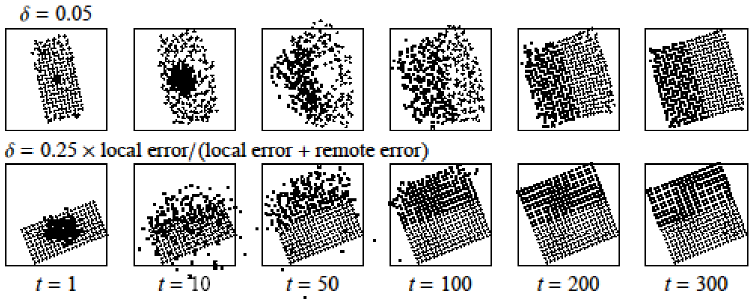
(a) Constant δ

(b) Adaptive δ

Adaptive δ leads to lower error

Vivaldi: Evaluation - Robustness

Using constant δ destroys the initial structure of the system (too much reliance on young high-error nodes)



Using adaptive δ preserves original structure

A stable 200-node network after 200 new nodes join

Part II: Targets and Techniques

- Infrastructure measurements
  - Topology discovery
    - Example: Doubletree
  - Network coordinates
    - Example: Vivaldi
  - → Bandwidth measurements
    - Example: CapProbe
- Traffic measurements
  - Traffic matrices
  - TCP
  - Anomaly detection
- Applications measurements
  - P2P
  - Web
  - Social networks
    - Example: YouTube

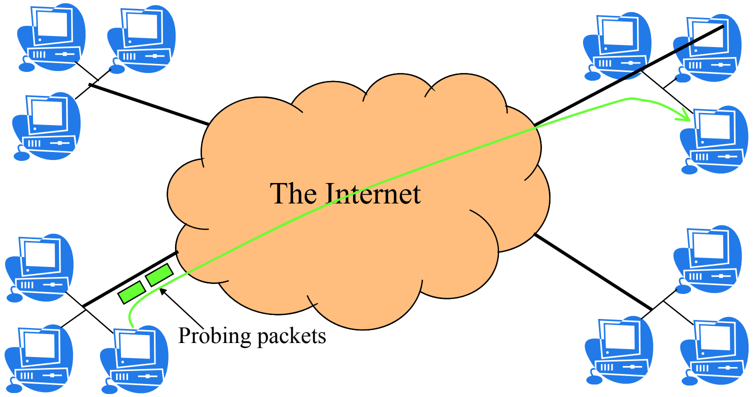
Infrastructure: Bandwidth measurements

- What?
  - Infer the bandwidth on a specific hop or on a whole path
    - **Capacity** = maximum possible throughput
    - **Available bandwidth** = portion of capacity not currently used
    - **Bulk transfer capacity** = throughput that a new single long-lived TCP connection could obtain
- Why?
  - Network aware applications
    - Server or peer selection
    - Route selection in overlay networks
  - QoS verification

Bandwidth measurements: Challenges

- Routers and switches do not provide direct feedback to end-hosts (except ICMP, also of limited use)
  - Mostly due to scalability, policy, and simplicity reasons
- Network administrators can read router/switch information using SNMP protocol
- End-to-end bandwidth estimation cannot be done in the above way
  - No access because of administrative barriers

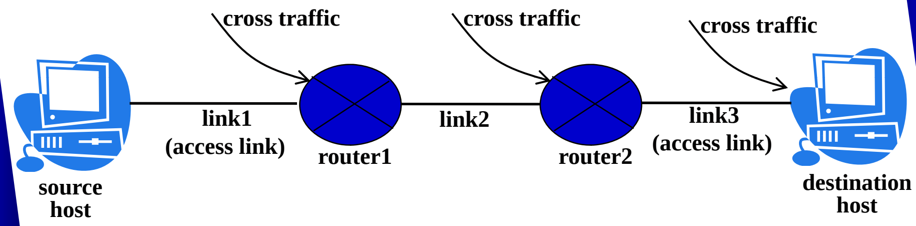
Bandwidth measurements: The Internet is a "black box"



- End-systems can infer network state through end-to-end (e2e) measurements
  - Without any feedback from routers
  - Objectives: accuracy, speed, minimal intrusiveness

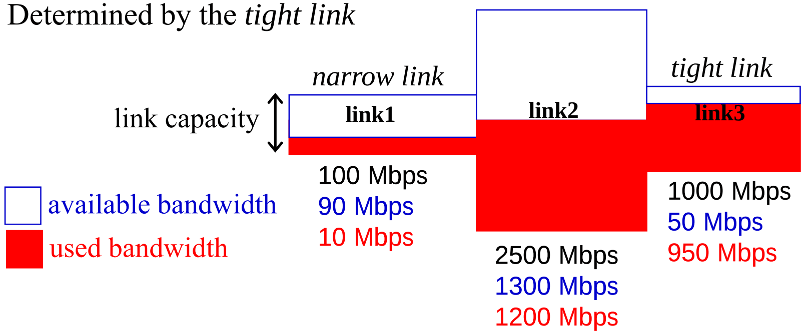
Bandwidth measurements: Metrics and definitions

- Example end-to-end path



Bandwidth measurements: Metrics and definitions

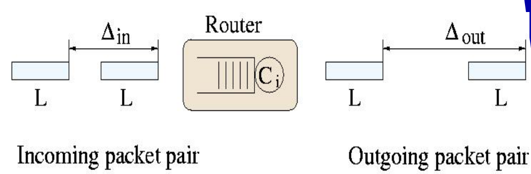
- Capacity of this path is 100 Mbps
  - Determined by the *narrow link*
- Available bandwidth of this path is 50 Mbps
  - Determined by the *tight link*



Bandwidth measurements: Techniques

- Generally use active probing
  - send packets with a specific inter-arrival pattern and observe the pattern at the other end
- Example: Packet-pair technique for capacity estimation
  - Originally, due to Jacobson & Keshav
  - Send two equal-sized packets back-to-back
    - Packet size: L
    - Packet trx time at link i: L/C_i
  - P-P dispersion: time interval between last bit of two packets
  - Without any cross traffic, the dispersion at receiver is determined by narrow link:

$$\Delta_{out} = \max\left(\Delta_{in}, \frac{L}{C_i}\right)$$



$$\Delta_R = \max_{i=1,\ldots,H}\left(\frac{L}{C_i}\right) = \frac{L}{C}$$

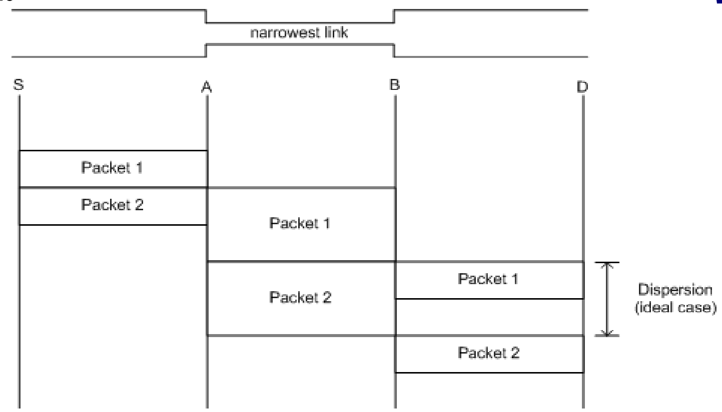
path capacity

Bandwidth estimation: CapProbe

- Rohit Kapoor et al.: *CapProbe: A Simple and Accurate Capacity Estimation Technique* (SIGCOMM 2004)
- CapProbe is a capacity estimation tool
- Takes into account effect of cross-traffic
- Cross traffic packets can affect P-P dispersion
  - P-P expansion: capacity underestimation
  - P-P compression: capacity overestimation
- Noise in P-P distribution depends on cross traffic load

CapProbe: Ideal Packet Dispersion

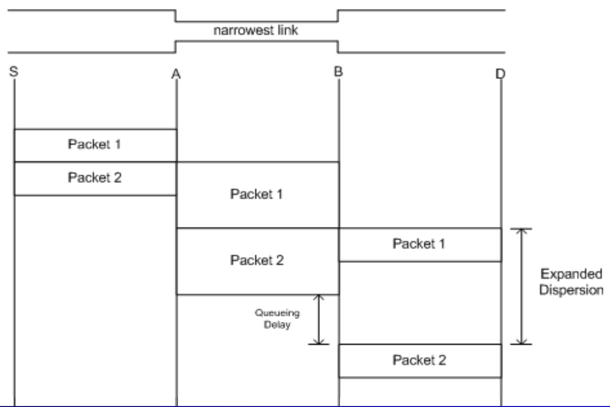
- No cross-traffic



Capacity = (Packet Size) / (Dispersion)

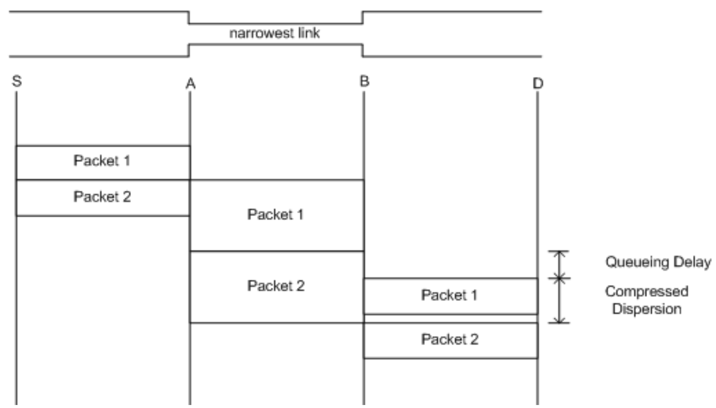
CapProbe: Expansion of Dispersion

- Cross-traffic (CT) serviced between PP packets
- Second packet queues due to Cross Traffic (CT)=> expansion of dispersion =>Under-estimation



CapProbe: Compression of Dispersion

- First packet queueing => compressed dispersion => Over-estimation

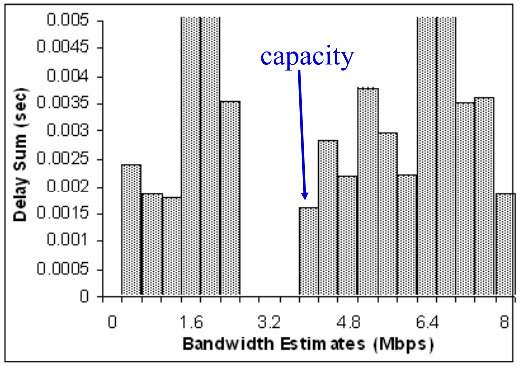


CapProbe: The approach

- Observations:
  - First packet queues more than the second
    - Compression
    - Over-estimation
  - Second packet queues more than the first
    - Expansion
    - Under-estimation
  - Both expansion and compression are the result of probe packets experiencing queuing
    - Sum of PP delay includes queuing delay
- Filter PP samples that do not have minimum queuing time
- *Dispersion of PP sample with minimum delay sum reflects capacity*

CapProbe Observation

- For each packet pair, CapProbe calculates delay sum:
  delay(packet_1) + delay(packet_2)
- A PP with the minimum delay sum points out the capacity



Bandwidth estimations: wrap-up

- Zillion of estimation tools & techniques
  - Abing, netest, pipechar, STAB, pathneck, IGI/PTR, abget, Spruce, pathchar, clink, pchar, PPrate, …
- Some practical issues
  - Traffic shapers
  - Non-FIFO queues
- More scalable methods
  - Passive measurements instead of active measurements
    - E.g. PPrate (2006) for capacity estimation: adapt Pathrate's algorithm
  - One measurement host instead of two cooperating ones
    - E.g. abget (2006) for available bandwidth estimation

Part II: Targets and Techniques

- Infrastructure measurements
  - Topology discovery
    - Example: Doubletree
  - Network coordinates
    - Example: Vivaldi
  - Bandwidth measurements
    - Example: CapProbe
- Traffic measurements
  - Traffic matrices
  - TCP
  - Anomaly detection
- Applications measurements
  - P2P
  - Web
  - Social networks
    - Example: YouTube

Traffic measurements

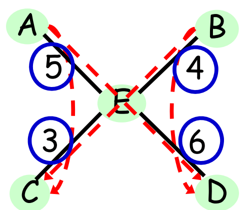
- Measure traffic on various layers
  - Application layer: P2P, On-line games, Skype, WWW…
  - Transport layer: TCP, (UDP)
  - IP layer
  - MAC layer: Wireless environments
  - Physical layer: Especially wireless links
- Objectives
  - Performance analysis
    - Network and applications
    - Modeling for simulations
    - Guide application development
  - Network engineering
    - Capacity planning
    - Troubleshooting
    - Network configuration
  - Analysis of user behavior
    - Enforce rules and regulations (RIAA)

Traffic matrices

- Traffic matrix is a network wide view of the traffic
  - Represents for every ingress point i into the network and every egress point j out of the network, the volume of traffic $T_{i,j}$ from i to j over a given time interval
  - Crucial input for traffic engineering
    - Routing, capacity planning…
- Problem: Can not measure directly
  - Flow-level measurements at ingress points can generate terabytes of data per day

⇨ Solution: Estimate

Traffic matrices (cont.)



src / dst	A	B
C	2	1
D	3	3

src / dst	A	B
C	1	2
D	4	2

AE, BE, EC, ED obtained using SNMP (=◯)
Link ED = AD + BD, Link AE = AD + AC…

⇨ We have a linear system **Y** = **AX**
  - **X** are the $T_{i,j}$ values to be estimated
  - **A** are IGP link weights
  - **Y** can be obtained using SNMP

Fundamental problem: # links < < # OD pairs

⇨ under-constrained system

⇨ infinitely many solutions

A variety of different proposed solutions

TCP

- TCP carries over 90% of the bytes in the Internet
- Modeling TCP
  - Express the performance of a TCP transfer as a function of some parameters that have physical meaning
    - Parameters: packet loss rate (p), round-trip time (RTT) of the TCP connection, the receiver advertised window, the slow start threshold, initial window size, window increase rate etc.
    - Performance metrics: Throughput, latency, fairness index etc.
  - E.g. the Square Root Formula: $Tput = \frac{MSS}{RTT}\sqrt{\frac{3}{2p}}$ (Mathis et al. 1997)
  - More advanced modeling
    - Advanced models for loss processes
    - Queuing theory
  - Models are (should be) validated through measurements

TCP (cont.)

- Empirical approach
  - Infer techniques from observations on real Internet traffic
  - More intuitive and simple models
  - Apply a tool or an algorithm on real packet traces and analyze results
  - Examples
    - Studying the burstiness of TCP traffic
      - H. Jiang, C. Dovrolis: *Why is the Internet traffic bursty in short (sub-RTT) time scales?* (SIGMETRICS 2005)
    - TCP Root Cause Analysis
      - How to identify the cause that prevents a TCP connection from achieving a higher throughput?
      - M. Siekkinen et al.: *A Root Cause Analysis Toolkit for TCP.* (Computer Networks, 2008)

Anomaly detection

- Study abnormal traffic
  - Non-productive traffic, a.k.a. Internet "background radiation"
  - Traffic that is malicious (scans for vulnerabilities, worms) or mostly harmless (misconfigured devices)
- Network troubleshooting
  - Identify and locate misconfigured or compromised devices
- Intrusion detection
  - Identify malicious activity before it hits you
  - Analyze traffic for attack signatures
- Characterizing malicious activities
  - *Honeypot*: an information system resource whose value lies in unauthorized or illicit use of that resource
    - Learn how attackers probe for and exploit a system
  - *Network telescope*: portion of routed IP address space on which little or no legitimate traffic exists

Part II: Targets and Techniques

- Infrastructure measurements
  - Topology discovery
    - Example: Doubletree
  - Network coordinates
    - Example: Vivaldi
  - Bandwidth measurements
    - Example: CapProbe
- Traffic measurements
  - Traffic matrices
  - TCP
  - Anomaly detection
- Applications measurements
  - P2P
  - Web
  - Social networks
    - Example: YouTube

P2P

- Main source of traffic in the Internet
- High relevance
  - Large impact for the network
  - Large impact for the users
  - Large impact for service providers

⇨ Lot of measurement efforts

P2P traffic identification

- Need to identify it before it can be characterized…
- Regulations and rules (RIAA)
- P2P uses TCP ports of other applications (e.g. 80)
  - Circumvent firewalls and "hide" from authorities
- Identification by well-know TCP ports
  - ☺ Fast and simple
  - ☹ May capture only a fraction of the total P2P traffic
- Search application specific keywords from packet payloads
  - ☺ Generally very accurate
  - ☹ A set of legal, privacy, technical, logistic, and financial obstacles
  - ☹ Need to reverse engineer poorly documented P2P protocols
  - ☹ Payload encryption in P2P protocols (e.g. some BitTorrent clients)

P2P traffic identification (cont.)

- Transport layer connection patterns
  - *Transport layer identification of P2P traffic.* T. Karagiannis et al. IMC 2004.
  - Observe connection patterns of source and destination IPs
  - ☺ Identify > 95% of P2P flows and bytes, 8-12% false positives
  - ☹ Limited by knowledge of the existing connection patterns
- "Early identification"
  - L. Bernaille et al.: *Early Application Identification.* (CoNEXT 2006)
  - Observe size and direction of first few packets of connection
  - Also encrypted (SSL) traffic
    - L. Bernaille et al.: *Early Recognition of Encrypted Applications.* (PAM 2007)
  - ☺ Robust: identify > 90% of unencrypted and > 85% of encrypted connections
  - ☺ Simple and fast
  - ☹ Need to train the system offline

P2P analysis

- Improve the performance of P2P applications
  - Scalability, download times, distribution efficiency
- Evaluate their impact on the network
  - What happens if a new killer P2P application emerges?
- Modeling
  - Build models for the behavior and verify by applying to real traffic
  - Mathematical model enables accurate analysis
  - E.g. D. Qiu et al.: *Modeling and performance analysis of BitTorrent-like peer-to-peer networks.* (SIGCOMM 2004)
- Empirical analysis of P2P systems
  - Study the behavior of operational P2P systems
  - Analyze observed traffic, application logs, etc.
  - E.g. K. Cho et al.: *The impact and implications of the growth in residential user-to-user traffic*. (SIGCOMM 2006)

Measuring the Web

- Still the single most popular application
- Main objective is to reduce *latency* experienced by users
  - Composed of many elements: DNS, TCP, HTTP, Web server and client delays (see the related assignment)

Measuring the Web

- What is measured?

Class	Measured property	Why measured
High-level characterization	Fraction of traffic, number of entities	Examining overall trends
Location	Presence of Web entities	Handling population distribution and mobility
Configuration	Software/hardware configuration	Load handling ability
User workload models	Access patterns	Modeling Web phenomena, shifting user populations
Traffic properties	Caching, flash crowds	Provisioning for regular and abnormal conditions
Application demands	Impact on network	Protocol improvement
Performance	Web components performance	Maintaining site popularity

(From Crovella and Krishnamurthy: *Internet Measurement. 2006.*)

Measuring the Web: Challenges

- Size
- Hidden data
  - Most servers not accessible for external measurements
    - E.g. intranet
  - At server, need to estimate/guess client properties
    - E.g. connectivity (dialup,cable…) and configurations (of browser, TCP…)
- Hidden layers
  - Redirection on several layers
  - DNS, HTTP, TCP
- Hidden entities
  - E.g. proxies

Measuring Online Social Networks

- Incredibly popular sites on the Web
  - MySpace, Facebook, YouTube, Orkut, …
- Users form an online social network
  - Powerful means of sharing, organizing, and finding content and contacts
- Opportunity for large scale studies of online social network graphs
  - Improve current systems
  - Design new applications of online social networks

Example: YouTube study

- Meeyoung Cha et al.: *I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System.* (IMC 2007)

- Try to find out:
  - Popularity distribution
  - Popularity evolution
  - P2P scalable distribution
  - Content duplication and illegal downloads

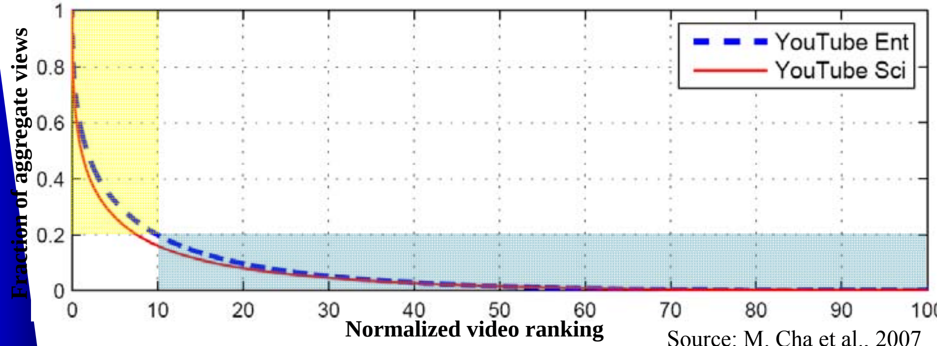
  ⇨ Implications on the design of future UGC systems

- Analyzed data
  - Crawled YouTube and other UGC systems metadata: video ID, length, views
  - Two categories: 1.6M Entertainment, 250KScience videos

User Generated Content (UGC) vs. Non-UGC

- UGC differs from non-UGC
  - Massive production scale
    - 15 days in YouTube to produce 120-y worth of movies in IMDb!
  - Extreme publishers
    - UGC user: up to 1000 uploads over few years
    - Movie director: up to 100 movies over 50 years
  - Short video length
    - 30 sec–5 min vs. 100 min movies in LoveFilm

Highly skewed popularity distribution

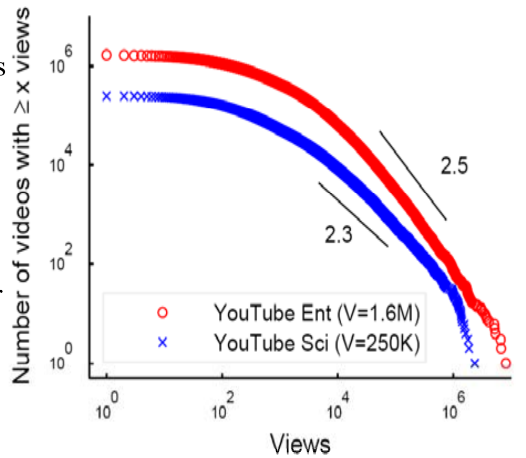


Source: M. Cha et al., 2007

- 10% popular videos account for 80% total views

CCDF of video popularity

- Power-law waist
  - *rich-get-richer* principle: If k users have already watched a video, then the rate of other users watching the video is proportional to k.
- Truncated both ends
  - Tail (popular videos) likely due to “fetch-at-most-once” behavior
    - Only view once unchanged video content
    - Popular web sites are visited many times (no tail truncation)



Source: M. Cha et al., 2007

Unpopular video distribution

- Why the truncated distribution of unpopular videos?
  - Sampling bias or pre-filters
    - Publishers tend to upload interesting videos
  - Information filtering or post-filters
    - Search results or suggestions favor popular items
- Leads to lower-than-expected popularity of niche contents
- Removing the filtering “bottleneck” could bring 40% additional views
  - Personalized recommendation
  - Enriched metadata…

Popularity evolution

- How requests on any given day are distributed across the video age?
- 6-day daily trace of Science videos
  - Step1- Group videos requested at least once by age
  - Step2- Count request volume per age group
- Some observations
  - Viewers mildly more interested in new videos
  - User preference relatively insensitive to age
    - 80% requests on >1 month old videos
  - Daily top hits mostly come from new videos
  - Some very old videos get significant amount of requests

YouTube analysis conclusions

- Paper has some more analysis results
- Results can aid in developing better strategies for
  - Marketing
  - Target advertising
  - Recommendation
  - Search engines
- Help to build more efficient UGC systems
  - Caching
  - Peer assisted VoD distribution

Wrapping up

- Lots of different activities around network measurements
- Very important for
  - Network and traffic management and engineering
  - Development of future services and protocols
- We merely scratched the surface…
- Two group assignments available on measurements
  - Web performance in practice – "Why are we waiting"? - Ten years after.
    - Dissecting experienced Web browsing latency
    - Compare results to those reported from ten years ago
  - Performance evaluation of Mercury for managing network measurements in real world
    - Deploy Mercury in Planetlab and experiment

That's all folks!