

---

# Measurements for Network Operations

Jennifer Rexford

Internet and Networking Systems

AT&T Labs - Research; Florham Park, NJ

<http://www.research.att.com/~jrex>



# Part 1: Outline

---

## ◆ Introduction

- Role of measurement in network operations
- Reacting to congestion, DoS attacks, and failures

## ◆ Passive traffic measurement

- Filtering, aggregation, and sampling
- SNMP, packet monitoring, and flow measurement

## ◆ Domain-wide traffic models

- Traffic, demand, and path matrices
- Inference, mapping, and direct observation

## ◆ Example application: traffic engineering

## ◆ Conclusions

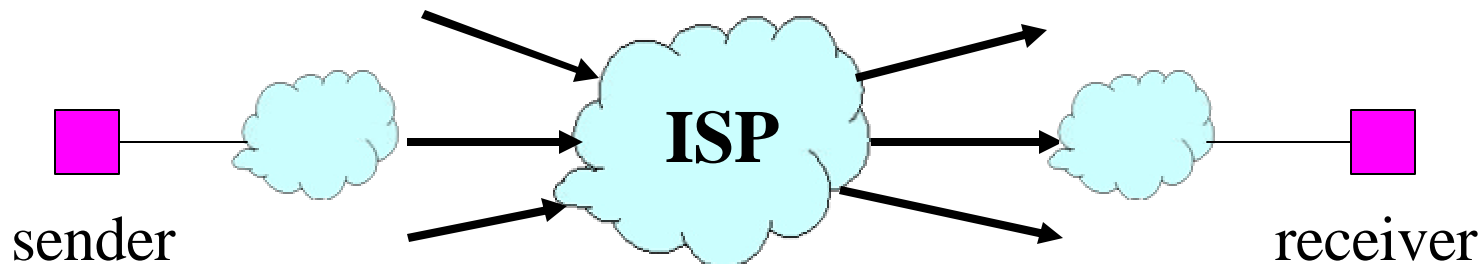
# Characteristics of the Internet

## ◆ The Internet is

- Decentralized (loose confederation of peers)
- Self-configuring (no global registry of topology)
- Stateless (limited information in the routers)
- Connectionless (no fixed connection between hosts)

## ◆ These attributes contribute

- To the success of Internet
- To the rapid growth of the Internet
- ... and the difficulty of controlling the Internet!



# Curse of the Dumb Network

---

## ◆ Don't IP networks manage themselves?

- Transport protocols (TCP) adapt to congestion
- Routing protocols adapt to topology changes

## ◆ Well yes, but...

- The network might not run all that *efficiently*
- E.g., many Web transfers sharing a single busy link

## ◆ Network operations

- Detect, diagnose, and fix problems
- Measure, model, and control the network

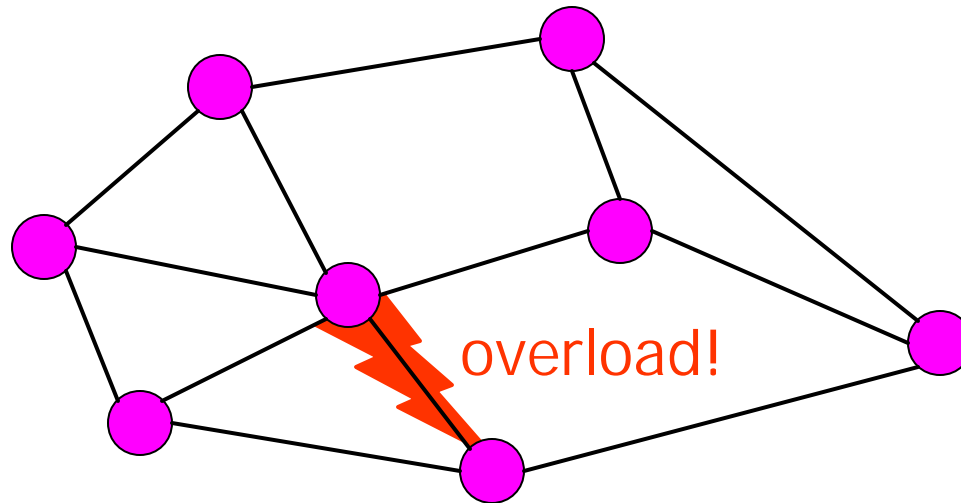
# Traffic Measurement: Science vs. Engineering

---

- ◆ **Science: characterizing the network**
  - End-to-end delay, throughput, and loss
  - Verification of congestion-control models
  - Workload models for Web users
  - Models of self-similarity/multi-fractal traffic
- ◆ **Engineering: managing the network**
  - Reports for customers and internal groups
  - Diagnosing performance and reliability problems
  - Tuning the network configuration
  - Planning outlay of new equipment

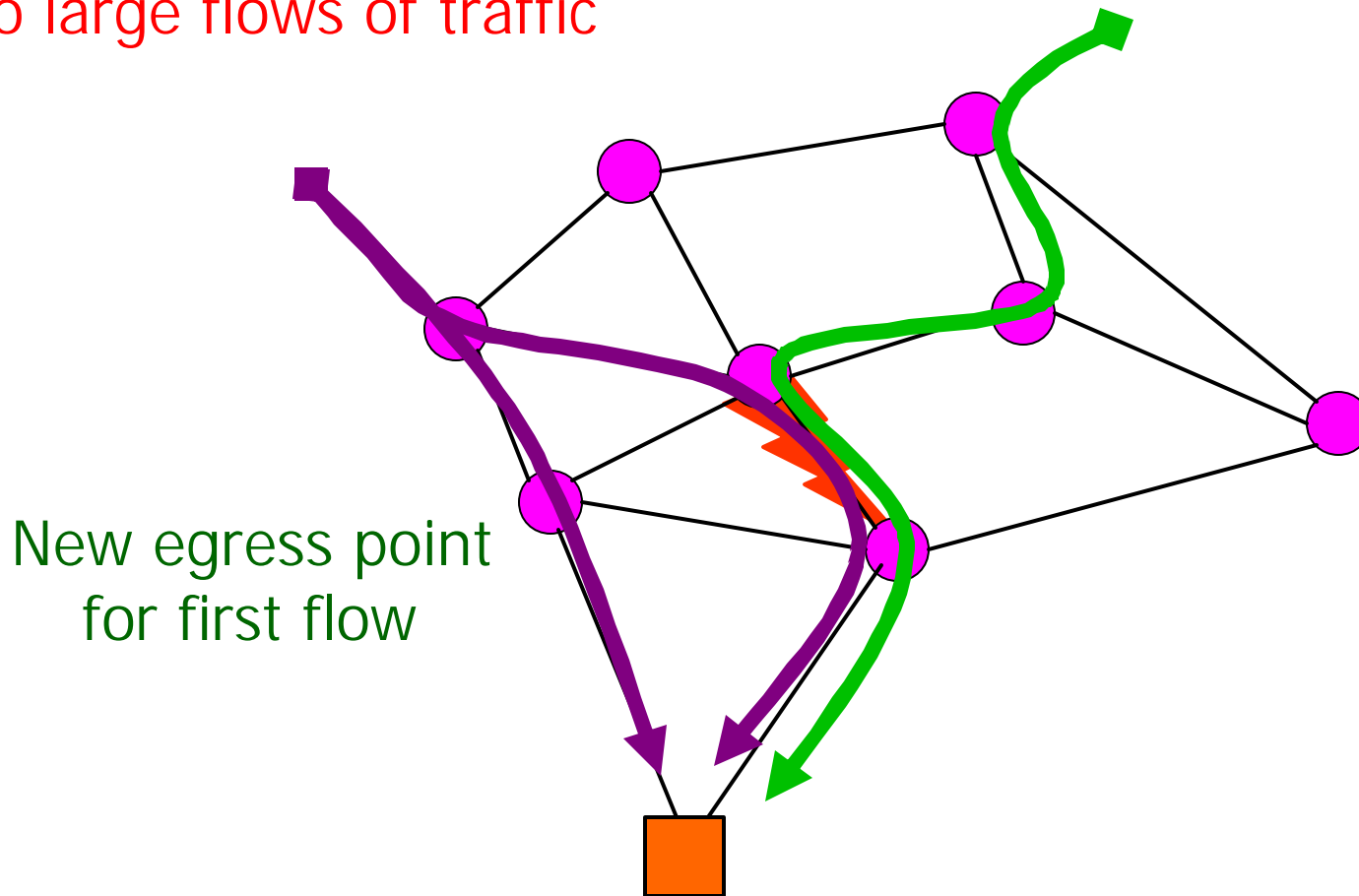
# Detecting Performance Problems

- High load or loss on the link
- Poor performance for probe traffic
- Customer complaints



# Network Operations: Excess Traffic

Two large flows of traffic

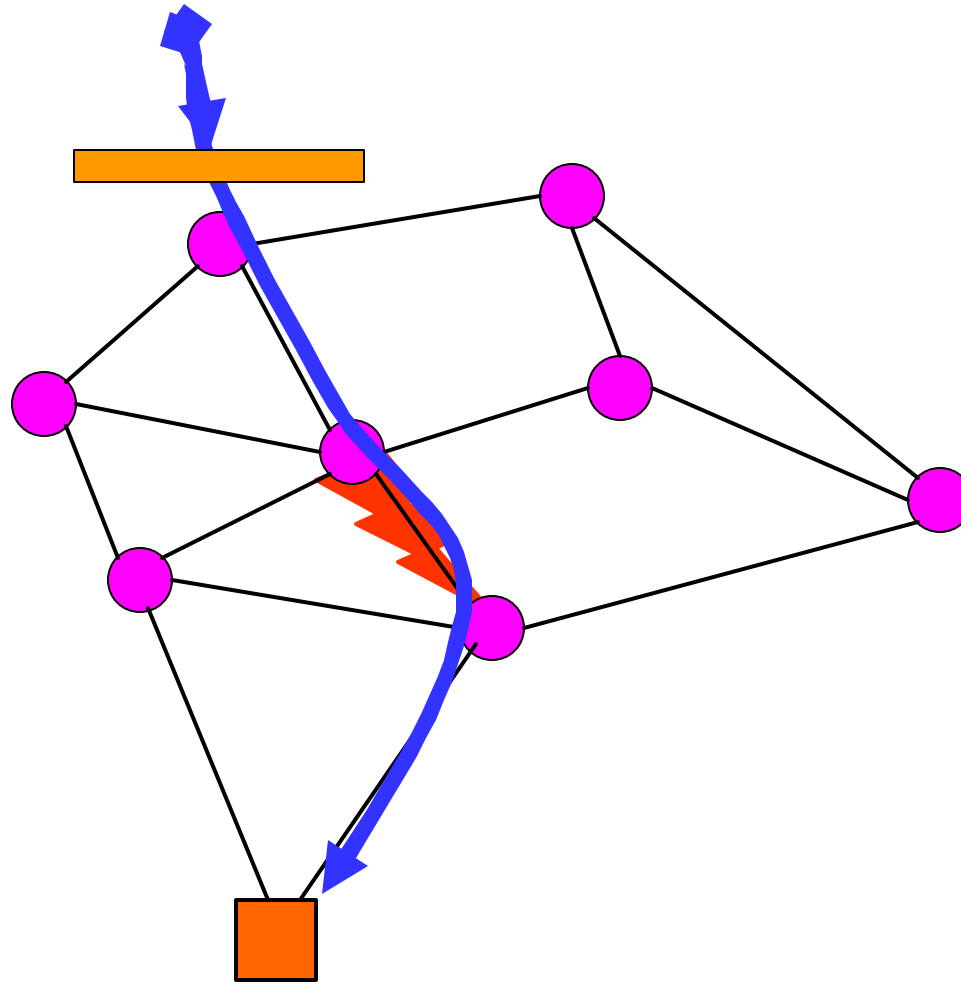


New egress point  
for first flow

**Multi-homed customer**

# Network Operations: Denial-of-Service Attack

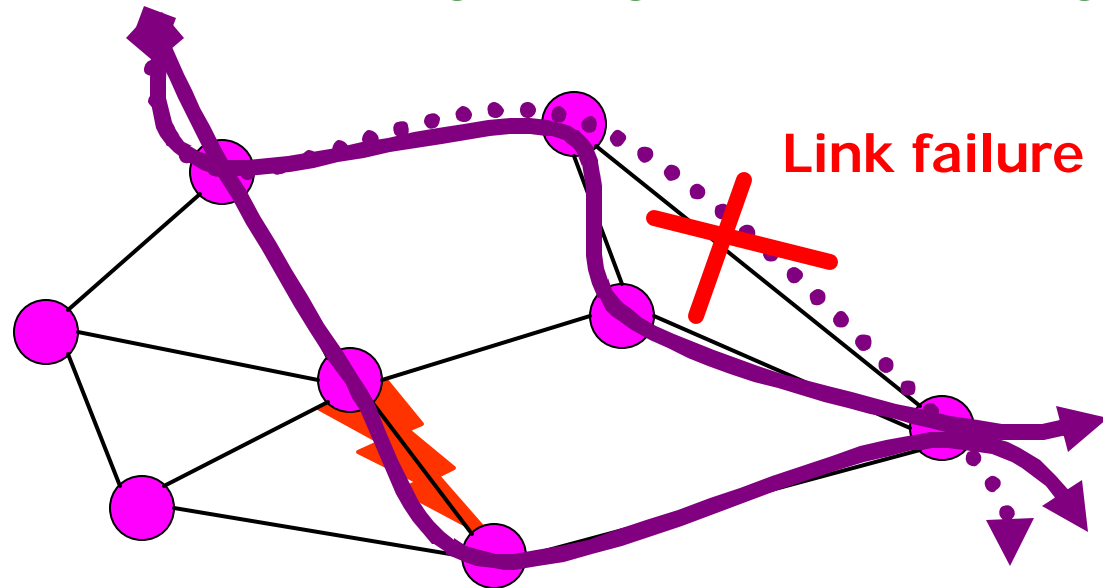
Install packet filter



Web server starts to die...

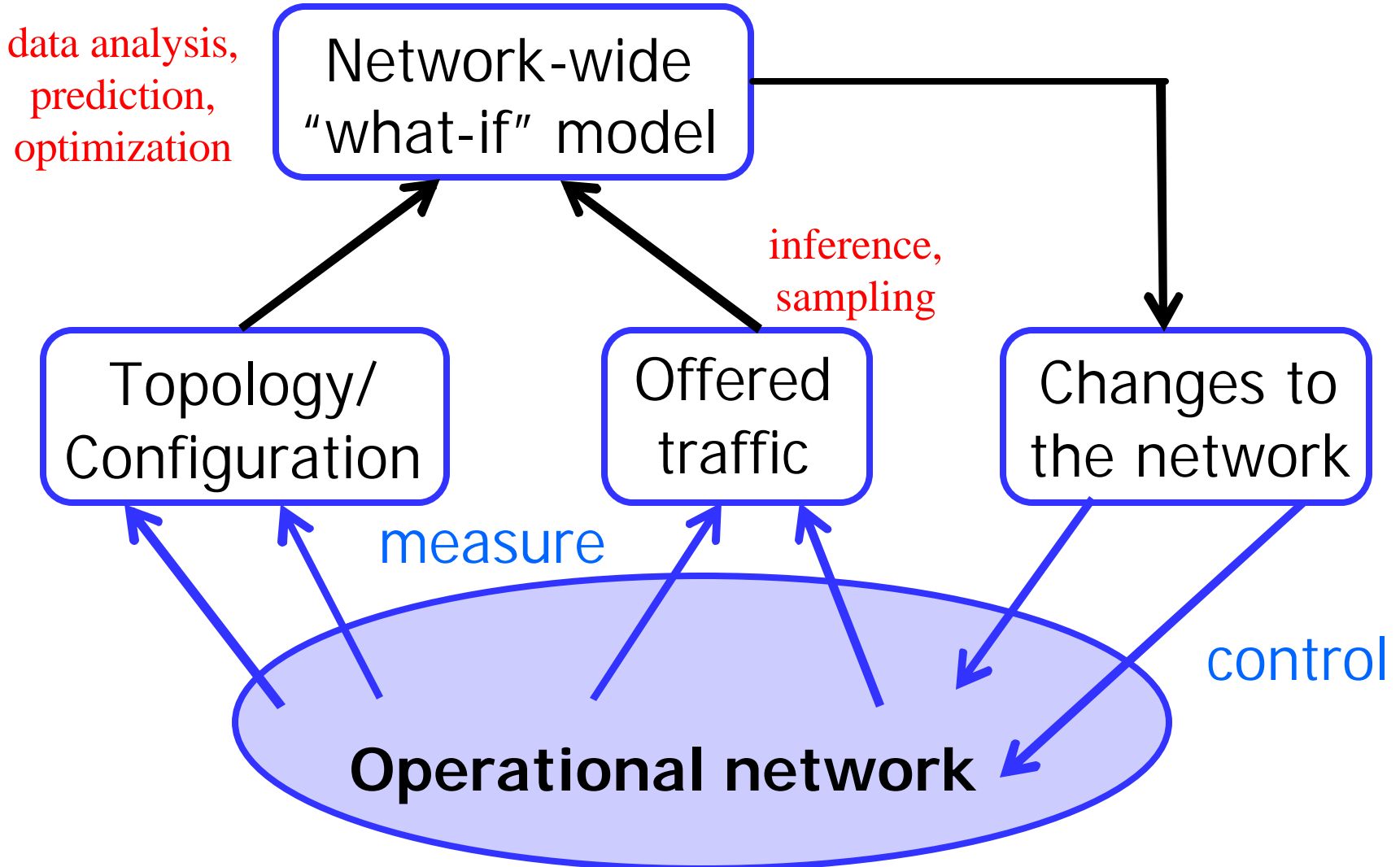
# Network Operations: Link Failure

Routing change alleviates congestion



New route overloads a link

# Network Operations: Measure, Model, and Control



# Network Operations: Time Scales

---

## ◆ Minutes to hours

- Denial-of-service attacks
- Router and link failures
- Serious congestion

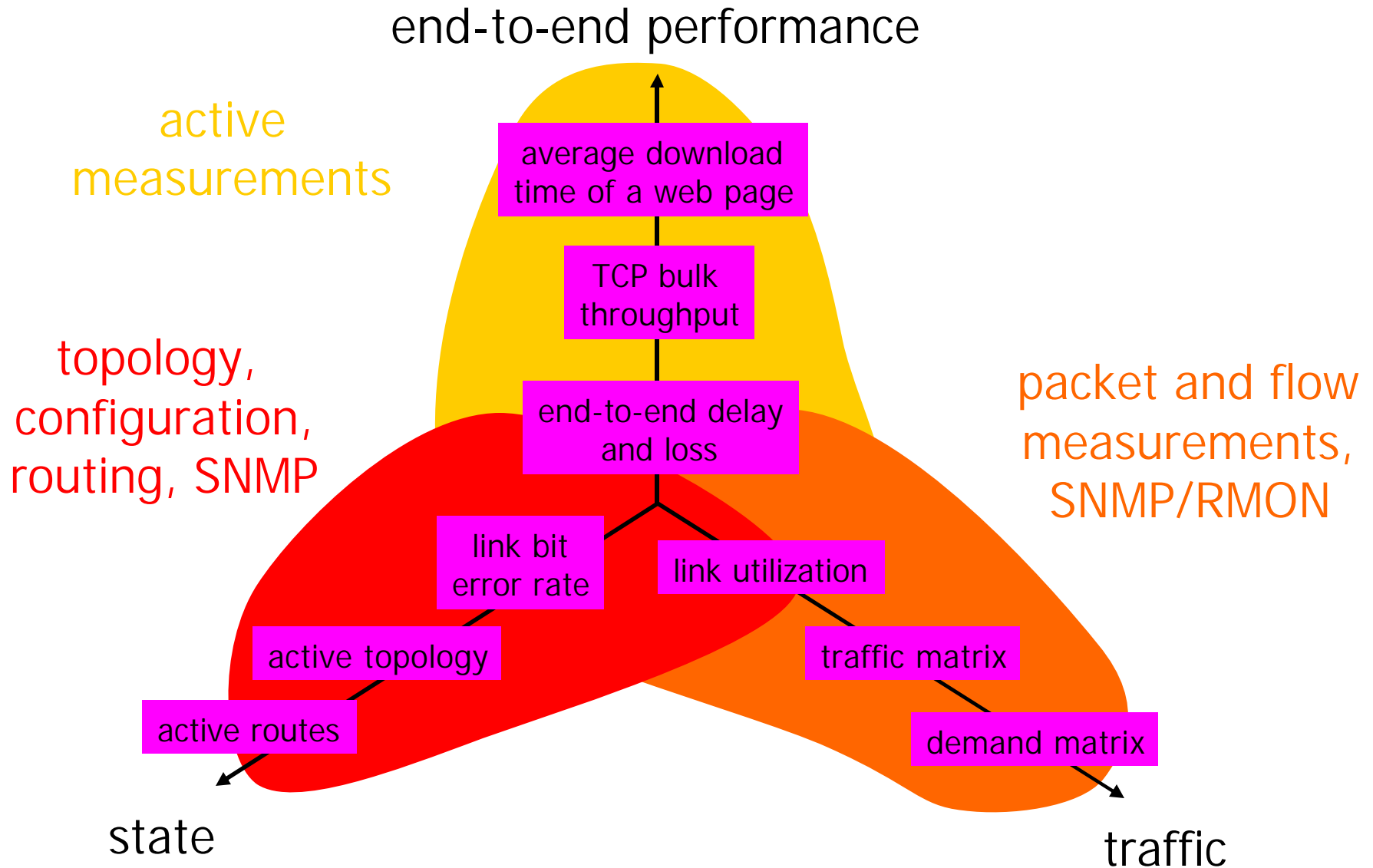
## ◆ Hours to weeks

- Time-of-day or day-of-week engineering
- Outlay of new routers and links
- Addition/deletion of customers or peers

## ◆ Weeks to years

- Planning of new capacity and topology changes
- Evaluation of network designs and routing protocols

# Terminology: Measurements vs Metrics



# Collection of Measurement Data

---

- ◆ Need to transport measurement data
  - Produced and consumed in different systems
  - Usual scenario: large number of measurement devices, and a small number of aggregation points
  - Usually in-band transport of measurement data
- ◆ Reliable transport: better data quality
  - But, device needs to maintain state and be addressable, and measurements may overload a congested link
- ◆ Unreliable transport: simpler measurement device
  - But, uncertainty due to lost measurement data, and the loss process might be hard to model

# Controlling Measurement Overhead

---

## ◆ Measurement overhead

- In some areas, could measure everything
- Information processing not the bottleneck
- Examples: geology, stock market,...
- Networking: thinning is crucial!

## ◆ Reduce measurement traffic:

- Filtering
- Aggregation
- Sampling

# Reducing Packet/Flow Measurement Overhead

---

- ◆ *Filtering*: select a subset of the traffic
  - E.g., destination address block for a customer
  - E.g., port number for an application (e.g., 80 for Web)
- ◆ *Aggregation*: grouping related traffic
  - E.g., packets with same next-hop Autonomous System
  - E.g., packets destined to a particular service (e.g., Web)
- ◆ *Sampling*: subselecting the traffic
  - Random, deterministic, or hash-based sampling
  - 1-out-of-n or stratified based on packet/flow size
- ◆ Combining filtering, aggregation, and sampling

# Basic Properties

	Filtering	Aggregation	Sampling
Precision	exact	exact	approximate
Generality	constrained a-priori	constrained a-priori	general
Local Processing	filter criterion for every object	table update for every object	only sampling decision
Local memory	none	one bin per value of interest	none
Compression	depends on data	depends on data	controlled

# Simple Network Management Protocol (SNMP)

---

## ◆ Definition

- Router CPU utilization, link utilization, link loss, ...
- Standardized protocol and naming hierarchy
- Collected from every router/link every few minutes

## ◆ Outline

- Management Information Base (MIB)
- Applications of SNMP traffic statistics
- Limitations of SNMP for network operations

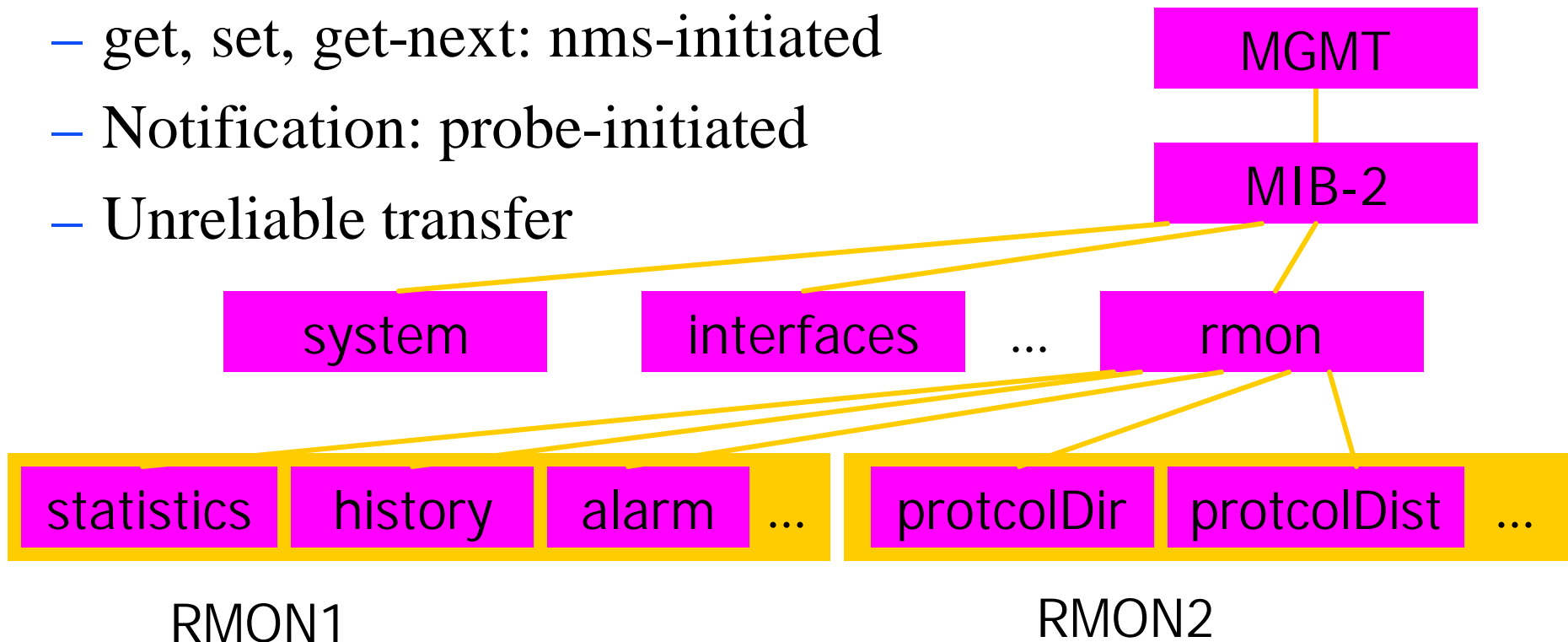
# SNMP: Management Information Base (MIB)

## ◆ Information model: MIB tree

- Naming and semantic convention between management station and agent (router)

## ◆ Protocol to access MIB

- get, set, get-next: nms-initiated
- Notification: probe-initiated
- Unreliable transfer



# SNMP: Applications of SNMP Traffic Statistics

---

- ◆ Driving wall-board at operations center
  - Complete view of every link in the network
- ◆ Usage-based billing
  - Tracking customer traffic on coarse time scale
- ◆ Alarming on significant changes
  - Detect high load or high packet loss
- ◆ Planning outlay of new capacity
  - Trending analysis to predict requirements
- ◆ Inference of the offered traffic matrix
  - ... more on this in part 2

# SNMP: Measurement Limitations

---

- ◆ **Statistics are hard-coded**
  - No local accumulation of statistics
  - No customizable alerting functions
- ◆ **Highly aggregated traffic information**
  - Aggregate link statistics (load, loss, etc.)
  - Cannot drill down into more detail
- ◆ **Protocol is simple, but dumb**
  - Cannot express complex queries over MIB

# SNMP: Conclusions

---

## ◆ SNMP link statistics

- Highly-aggregated view of every link

## ◆ Applications

- Network-wide view of aggregate traffic
- Detecting (but not diagnosing) problems

## ◆ Advantages

- Open standard that is universally supported
- Low volume of data, and low overhead on routers

## ◆ Disadvantages

- Coarse-grain view in time (e.g., 1-5 minutes)
- Coarse-grain view in space (e.g., entire link)
- Unreliable transfer of the measurement data

# Packet Monitoring

---

## ◆ Definition

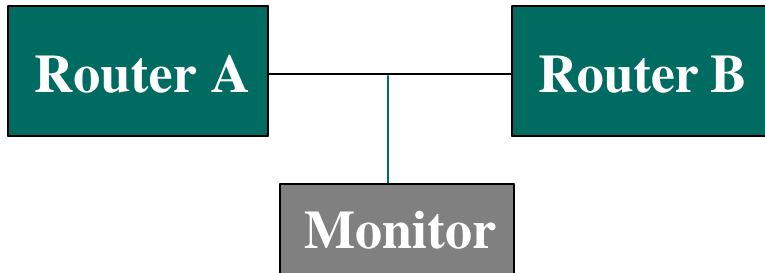
- Passively collect packets on links
- Record IP, TCP/UDP, or application-layer traces

## ◆ Outline

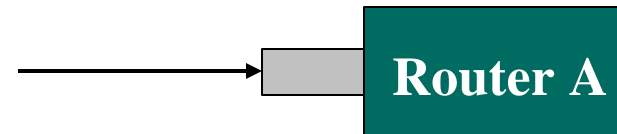
- Tapping a link and capturing packets
- Operational applications for packet traces
- Placement of the packet monitor
- Practical challenges in collecting the data

# Packet Monitoring: Tapping the WAN Link

## Splitting a point-to-point link



## Line card that does packet sampling

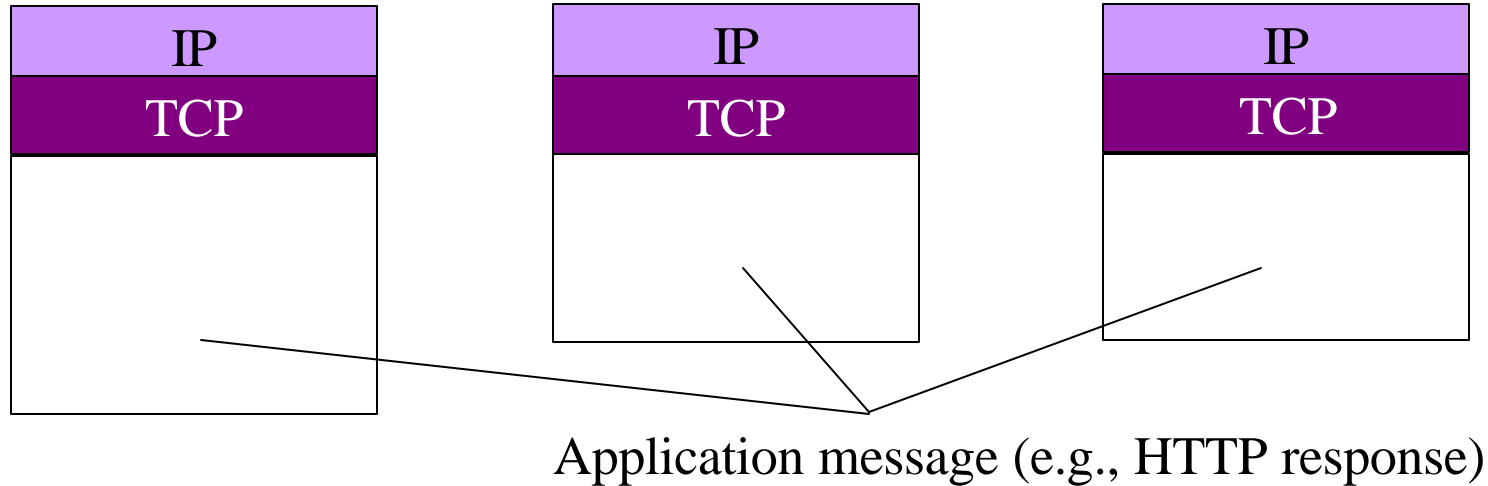


# Packet Monitoring: Selecting the Traffic

---

- ◆ Filter to focus on a subset of the packets
  - IP addresses (e.g., to/from specific machines)
  - Protocol (e.g., TCP, UDP, or ICMP)
  - Port numbers (e.g., HTTP, DNS, BGP, Kazaa)
- ◆ Collect first n bytes of packet (snap length)
  - Medium access control header (if present)
  - IP header (typically 20 bytes)
  - IP+UDP header (typically 28 bytes)
  - IP+TCP header (typically 40 bytes)
  - Application-layer message (entire packet)

# Extracting Data from IP Packets



## ◆ Many layers of information

- IP: source/dest IP addresses, protocol (TCP/UDP), ...
- TCP/UDP: src/dest port numbers, seq/ack, flags, ...
- Application: URL, user keystrokes, BGP updates, ...

# Packet Monitoring: IP Header Traces

---

- ◆ Source/destination IP addresses
  - Popular Web servers and heavy customers
- ◆ Traffic breakdown by protocol
  - Amount of traffic not using congestion control
- ◆ Packet delay through the router
  - Identification of typical delays and anomalies
- ◆ Distribution of packet sizes
  - Workload models for routers
- ◆ Burstiness of link traffic over time
  - Provisioning rules for allocating link capacity
- ◆ Throughput between src/dest pairs
  - Detection of performance problems

# Packet Monitoring: TCP Header Analysis

---

- ◆ Source and destination port numbers
  - Popular applications (HTTP, Kazaa, DNS)
  - # of parallel connections between source-dest pairs
- ◆ Sequence/ACK numbers and timestamps
  - Out-of-order/lost packets
  - Violations of congestion control
- ◆ Number of packets/bytes per connection
  - Size of typical Web transfers
  - Frequency of bulk transfers
- ◆ SYN flags from client machines
  - Unsuccessful connection requests
  - Denial-of-service attacks

# Packet Monitoring: Packet Contents

---

- ◆ **Application-layer header**
  - HTTP and RTSP request and response headers
  - FTP, NNTP, and SMTP commands and replies
  - DNS queries/responses; OSPF/BGP messages
- ◆ **Application-layer body**
  - HTTP resources (or checksums of the contents)
  - User keystrokes in Telnet/Rlogin sessions
- ◆ **Security/privacy**
  - Significant risk of violating user privacy
  - Analysis thwarted by end-to-end encryption

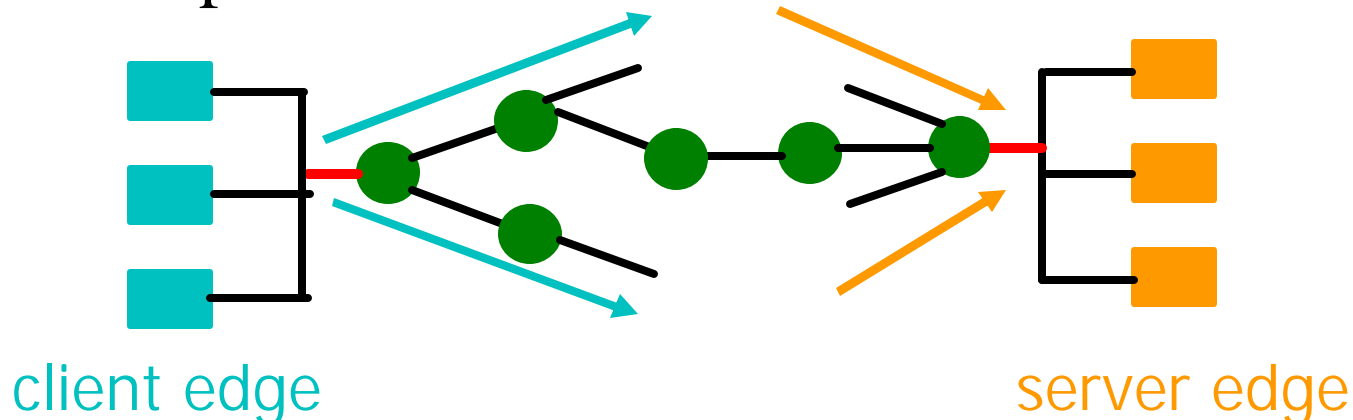
# Packet Monitoring: Monitor at the Edge

## ◆ Client edge

- Capture all traffic to/from a group of clients
- Useful for evaluating effectiveness of a proxy
- Not be representative of other clients

## ◆ Server edge

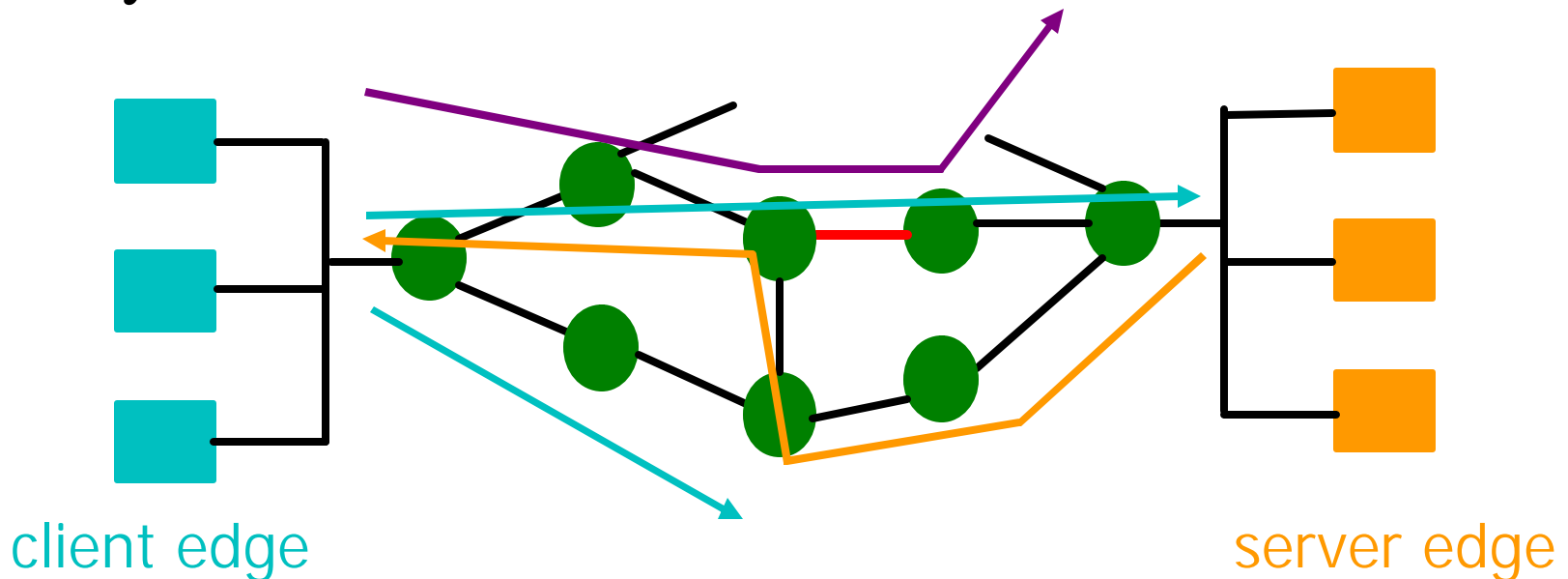
- Capture all traffic to/from a set of Web sites
- Useful for characterization of access patterns
- Not be representative of accesses to other sites



# Packet Monitoring: Monitor in the Core

## ◆ Middle of network

- Capture all traffic traversing a particular link
- Useful for capturing a diverse mix of traffic
- May not see all traffic traveling from host A to host B
- May not see the reverse traffic from host B to host A



# Packet Monitoring: System Constraints

---

## ◆ High data rate

- Bandwidth limits on CPU, I/O, memory, and disk/tape
- Could monitor lower-speed links (edge of network)

## ◆ High data volume

- Space limitations in main memory and on disk/tape
- Could do online analysis to sample, filter, & aggregate

## ◆ High processing load

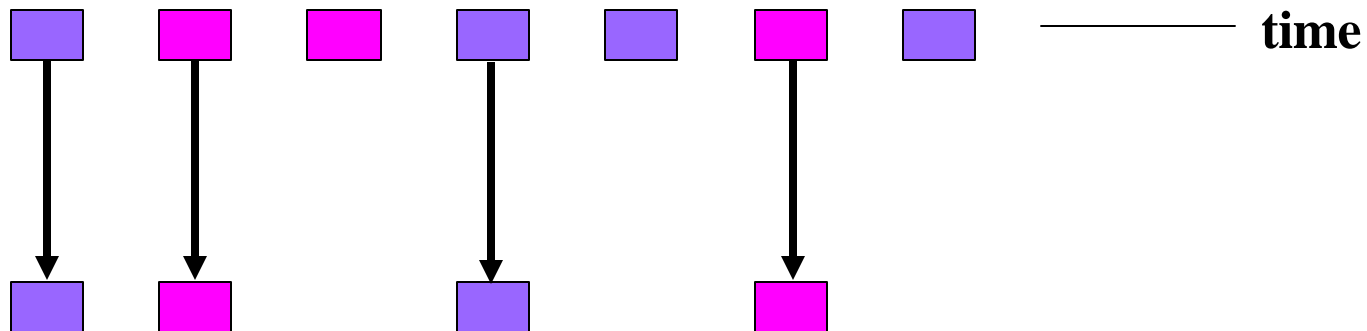
- CPU/memory limits for extracting and analyzing
- Could do offline processing for time-consuming analysis

## ◆ General solutions to system constraints

- Sub-select the traffic (addresses/ports, first n bytes)
- Operating system and interface card support
- Efficient/robust software and hardware for the monitor

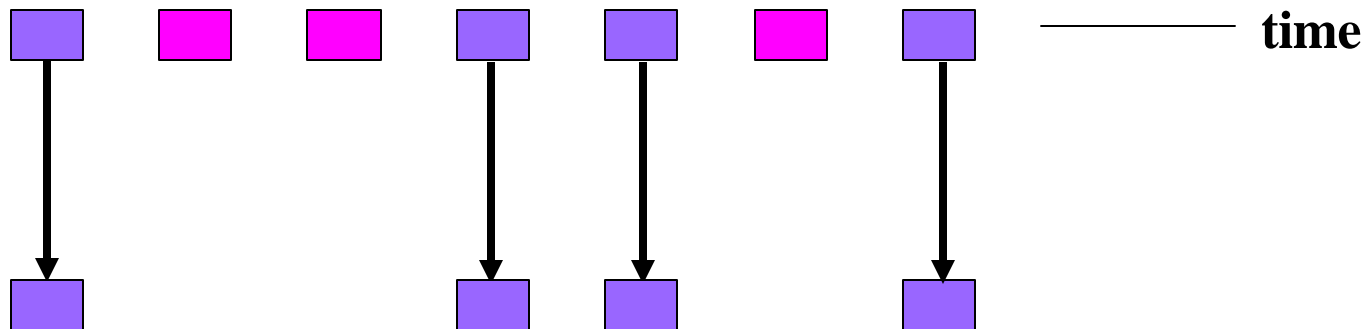
# Packet Monitoring: Packet Sampling

- ◆ Random sampling of individual packets
  - Per-packet: select 1-out-of-m packets
  - Per-byte: sample in proportion to packet length
- ◆ Appropriate for certain types of analysis
  - Traffic mix by port #, src/dest IP address, protocol, etc.
  - Packet size distribution, frequency of IP/TCP header values
  - Not appropriate for gleaning application-level information



# Packet Monitoring: Application-Layer Sampling

- ◆ Sample application-layer messages
  - Select 1-out-of-m flows (all packets or no packets)
  - Cannot randomly sample at the packet level
- ◆ Sample based on hash of the packet header
  - TCP/UDP session: addresses and port numbers
  - End-host pair: source and destination address



# Packet Monitoring: PSAMP IETF Activity

---

- ◆ Goals of the “psamp” group
  - Minimal functionality for packet-level measurement
  - Tunable trade-offs between overhead and accuracy
  - Measurement data for a variety of important applications
- ◆ Basic idea: parallel filter/sample banks
  - Filter on header fields (src/dest, port #s, protocol)
  - 1-out-of-N sampling (random, periodic, or hash)
  - Record key IP and TCP/UDP header fields
  - Send measurement record to a collection system
- ◆ References
  - <http://ops.ietf.org/lists/psamp/>

# Packet Monitoring: Conclusions

---

## ◆ Packet monitoring

- Detailed, fine-grain view of individual links

## ◆ Advantages

- Finest level of granularity (individual IP packets)
- Primary source of application-level information

## ◆ Disadvantages

- Expensive to build and deploy
- Large volume of measurement data
- Difficult to deploy over a large network
- Hard to collect on high-speed links
- Hard to reconstruct application-level info

# Flow Measurement: Outline

---

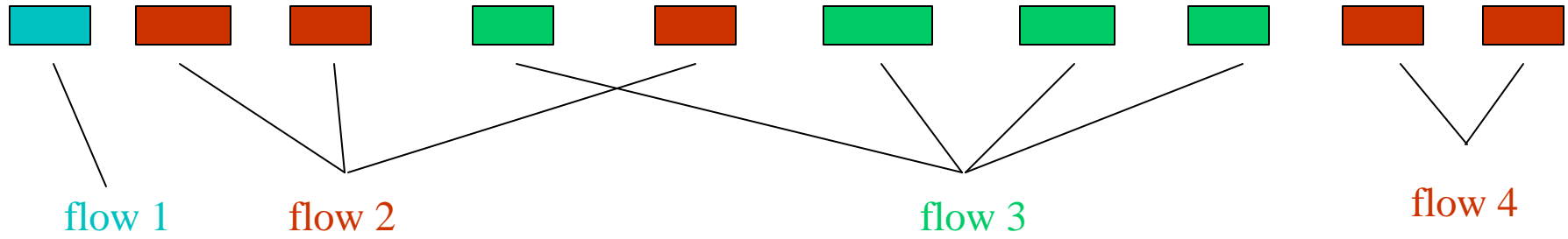
## ◆ Definition

- Passively collect statistics about groups of packets
- Group packets based on headers and time
- Essentially a form of aggregation

## ◆ Outline

- Definition of an IP “flow”
- Applications of flow measurement
- Mechanics of collecting flow-level measurements
- Reducing the measurement overheads

# Flow Measurement: IP Flows



## ◆ Set of packets that “belong together”

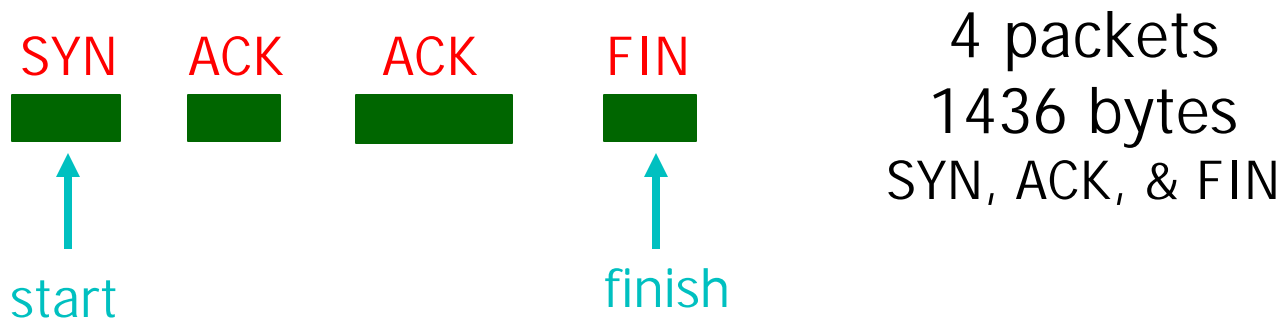
- Source/destination IP addresses and port numbers
- Same protocol, ToS bits, ...
- Same input/output interfaces at a router (if known)

## ◆ Packets that are “close” together in time

- Maximum spacing between packets (e.g., 15 sec, 30 sec)
- Example: flows 2 and 4 are different flows due to time

# Flow Measurement: Recording Traffic Statistics

- ◆ Packet header information (same for every packet)
  - Source and destination IP addresses
  - Source and destination TCP/UDP port numbers
  - Other header fields (protocol, ToS bits, etc.)
- ◆ Aggregate traffic information
  - Start and finish time (time of first & last packet)
  - Total number of bytes and number of packets in the flow
  - TCP flags (e.g., logical OR over the packets)



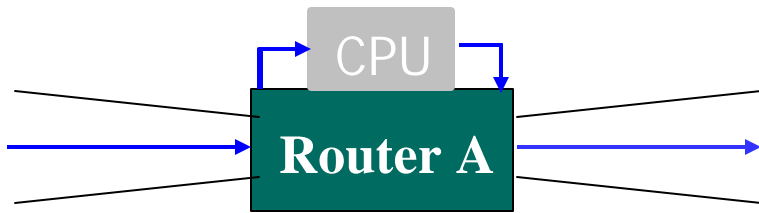
# Flow Measurement: Versus Packet Monitoring

---

- ◆ **Basic statistics (available from both)**
  - Traffic mix by IP addresses, port numbers, and protocol
  - Average packet size
- ◆ **Traffic over time**
  - Both: traffic volumes on a medium-to-large time scale
  - Packet: burstiness of the traffic on a small time scale
- ◆ **Statistics per TCP connection**
  - Both: number of packets & bytes transferred over the link
  - Packet: frequency of lost or out-of-order packets, and the number of application-level bytes delivered
- ◆ **Per-packet info (available only from packet traces)**
  - TCP seq/ack #s, receiver window, per-packet flags, ...
  - Probability distribution of packet sizes
  - Application-level header and body (full packet contents)

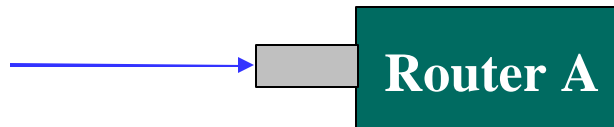
# Flow Measurement: Tapping the Link

**Router CPU that generates flow records**



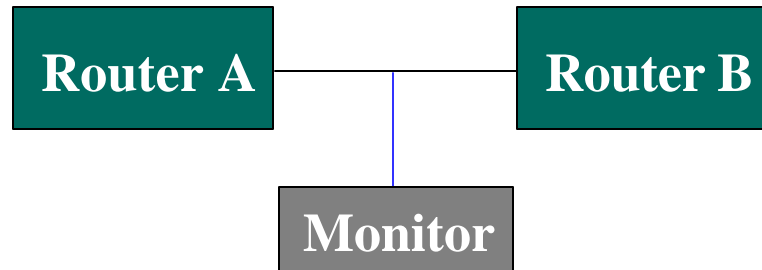
**...may degrade forwarding performance**

**Line card that generates flow records**



**...more efficient to support measurement in each line card**

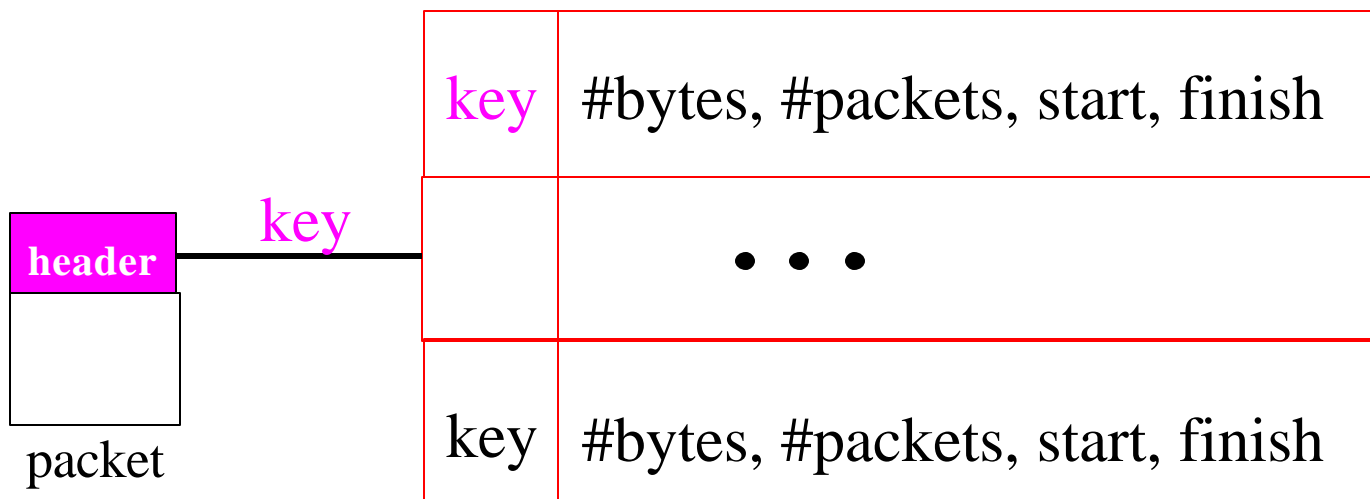
**Packet monitor that generates flow records**



**...third party**

# Flow Measurement: Mechanics of Flow Cache

- ◆ Maintain a cache of active flows
  - Storage of byte/packet counts, timestamps, etc.
- ◆ Compute a key per incoming packet
  - Concatenation of source, destination, port #s, etc.
- ◆ Index into the flow cache based on the key
  - Creation or updating of an entry in the flow cache



# Flow Measurement: Evicting Cache Entries

---

## ◆ Flow timeout

- Remove idle flows (e.g., no packet in last 60 sec)
- Periodic sequencing through the cache

## ◆ Cache replacement

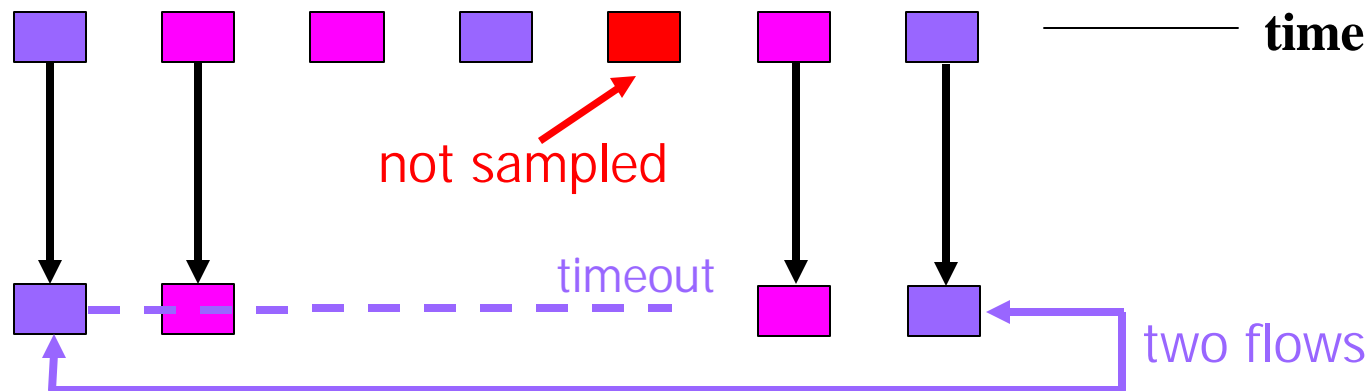
- Remove flow(s) when cache is full
- Evict existing flow(s) upon creating new entry
- Apply eviction policy (LRU, random flow, etc.)

## ◆ Long-lived flows

- Remove flow(s) that persist for a long time (e.g., 30 min)
- ... otherwise flow statistics don't become available
- ... and the byte and packet counters might overflow

# Flow Measurement: Packet Sampling

- ◆ Packet sampling before flow creation (Sampled Netflow)
  - 1-out-of-m sampling of individual packets (e.g.,  $m=100$ )
  - Create of flow records over the sampled packets
- ◆ Reducing overhead
  - Avoid per-packet overhead on  $(m-1)/m$  packets
  - Avoid creating records for a large number of **small flows**
- ◆ Increasing overhead (in some cases)
  - May split some **long transfers** into multiple flow records
  - ... due to larger time gaps between successive packets



# Flow Measurement: Flow-Level Sampling

---

- ◆ Sampling of flow records evicted from flow cache
  - When evicting flows from table or when analyzing flows
- ◆ Stratified sampling to put weight on “heavy” flows
  - Select all long flows and sample the short flows
- ◆ Reduces the number of flow records
  - Still measures the vast majority of the traffic

Flow 1, 40 bytes

← sample with 0.1% probability

Flow 2, 15580 bytes

Flow 3, 8196 bytes

Flow 4, 5350789 bytes

← sample with 100% probability

Flow 5, 532 bytes

Flow 6, 7432 bytes

← sample with 10% probability

# Flow Measurement: Aggregation

---

## ◆ Define flows at a coarser level

- Ignore TCP/UDP port numbers, ToS bits, etc.
- Source and destination IP address blocks
- Source and destination autonomous system (AS)

## ◆ Advantages

- Reduce the size of the flow cache
- Reduce the number of flow records

## ◆ Disadvantage

- Lost information for basic traffic reporting
- Impacted by the view of routing (prefixes) at this router

# IETF Standards Activity

---

## ◆ Real-Time Traffic Flow Meter (RTFM)

- Past working group on describing and measuring flows
- Meter with flow table and packet matching/handling
- Meter readers that transport usage data from the meters
- Manager for downloading rule sets to the meter
- SNMP for downloading rules and reading usage data

## ◆ Internet Protocol Flow eXport (IPFX)

- Distinguishing flows (interfaces, IP & transport header fields, ...)
- Metering (reliability, sampling, timestamps, flow timeout)
- Data export (information model, reliability, confidentiality, integrity, anonymization, reporting times)

# Flow Measurement: Conclusions

---

## ◆ Flow measurement

- Medium-grain view of traffic on links

## ◆ Advantages

- Lower measurement volume than packet traces
- Available on high-end line cards (Cisco Netflow)
- Control over overhead via aggregation and sampling

## ◆ Disadvantages

- Computation and memory requirements for flow cache
- Loss of fine-grain timing and per-packet information
- Not uniformly supported by router vendors

## Part 2: Outline

---

- ◆ Introduction
  - Role of measurement in network operations
  - Reacting to congestion, DoS attacks, and failures
- ◆ Passive traffic measurement
  - Filtering, aggregation, and sampling
  - SNMP, packet monitoring, and flow measurement
- ◆ Domain-wide traffic models
  - Traffic, demand, and path matrices
  - Inference, mapping, and direct observation
- ◆ Example application: traffic engineering
- ◆ Conclusions

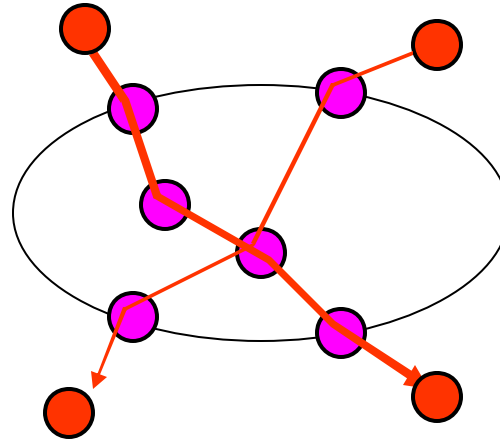
# Traffic Representations for Network Operators

---

- ◆ Network-wide views
  - Not directly supported by IP
  - Combine traffic, topology, and state
- ◆ Need to make assumptions
  - Properties of the traffic
  - Knowledge of topology and routing
  - Network support for measurement
- ◆ Models: traffic, demand, and path matrices
  - Populating the models from measurements
  - Proposals for new types of measurements

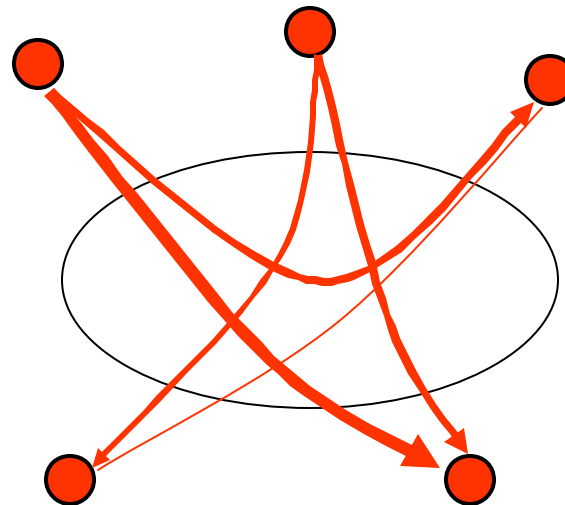
# End-to-End Traffic & Demand Models

Ideally, captures all the information about the current network **state and behavior**



path matrix = bytes per path

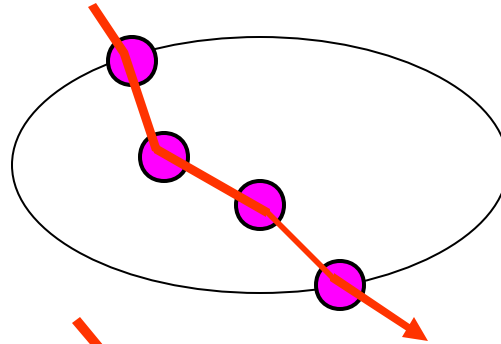
Ideally, captures all the information that is **invariant** with respect to the network state



traffic matrix = bytes per source-destination pair

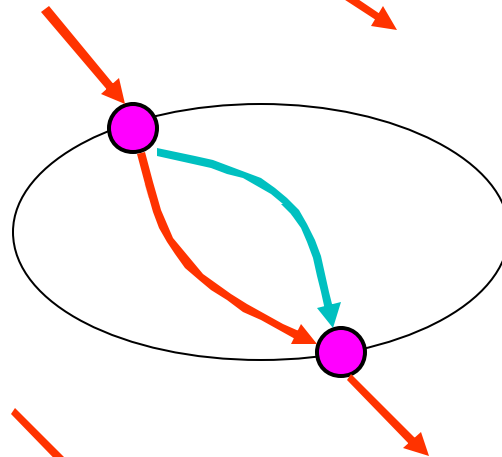
# Domain-Wide Network Traffic Models

current state &  
traffic flow



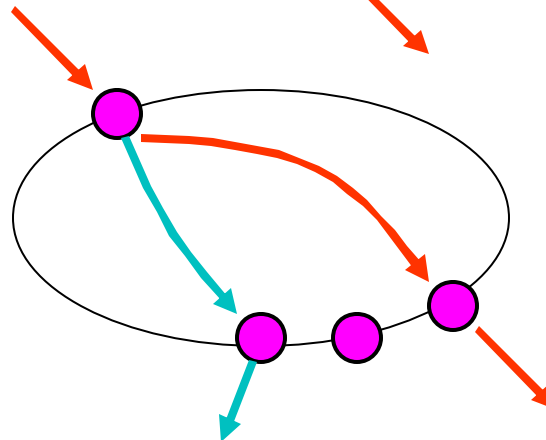
fine grained:  
**path matrix** =  
bytes per path

predicted  
control action:  
impact of intra-  
domain routing



intradomain focus:  
**traffic matrix** =  
bytes per ingress-egress

predicted  
control action:  
impact of inter-  
domain routing



interdomain focus:  
**demand matrix** =  
bytes per ingress and  
set of possible egresses

# Path Matrix: Operational Uses

---

## ◆ Congested link

- Problem: easy to detect, hard to diagnose
- Which traffic is responsible? Which traffic affected?

## ◆ Customer complaint

- Problem: customer has limited visibility to diagnose
- How is the traffic of a given customer routed?
- Where does the traffic experience loss and delay?

## ◆ Denial-of-service attack

- Problem: spoofed source address, distributed attack
- Where is the attack coming from? Who is affected?

# Traffic Matrix: Operational Uses

---

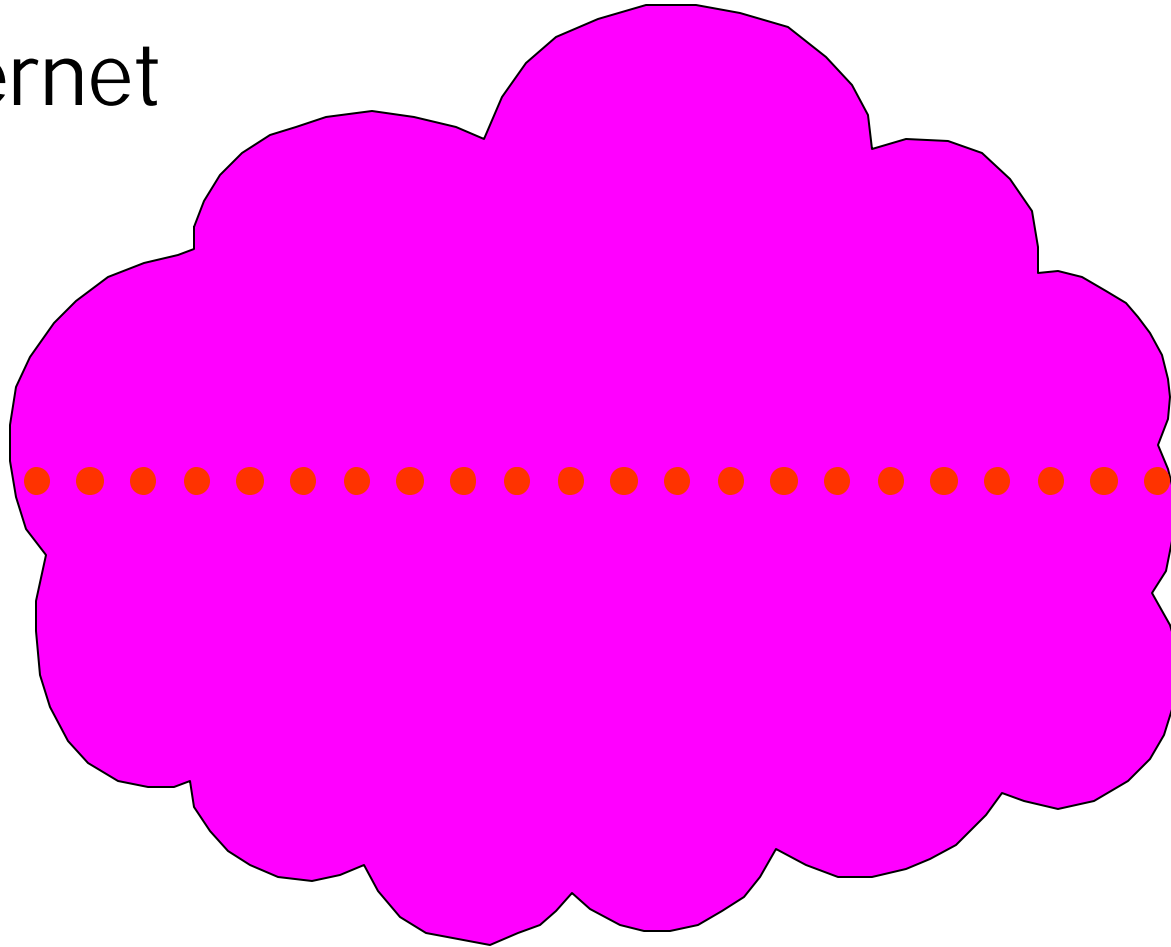
- ◆ Short-term congestion and performance problems
  - Problem: predicting link loads after a routing change
  - Map the traffic matrix onto the new set of routes
- ◆ Long-term congestion and performance problems
  - Problem: predicting link loads after topology changes
  - Map traffic matrix onto the routes on new topology
- ◆ Reliability despite equipment failures
  - Problem: allocating spare capacity for failover
  - Find link weights such that no failure causes overload

# Demand Matrix: Motivating Example

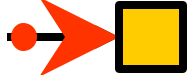
---

Big Internet

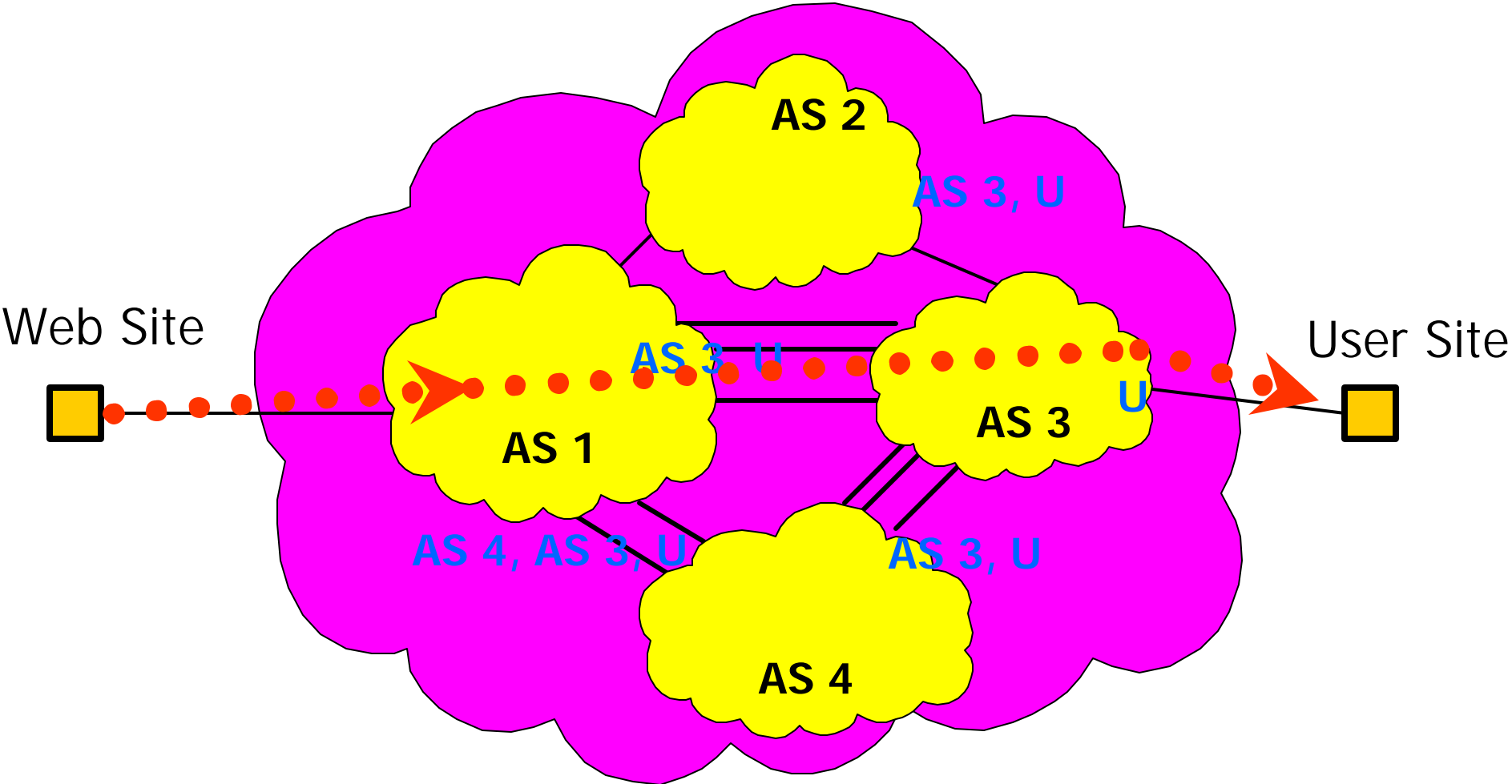
Web Site



User Site

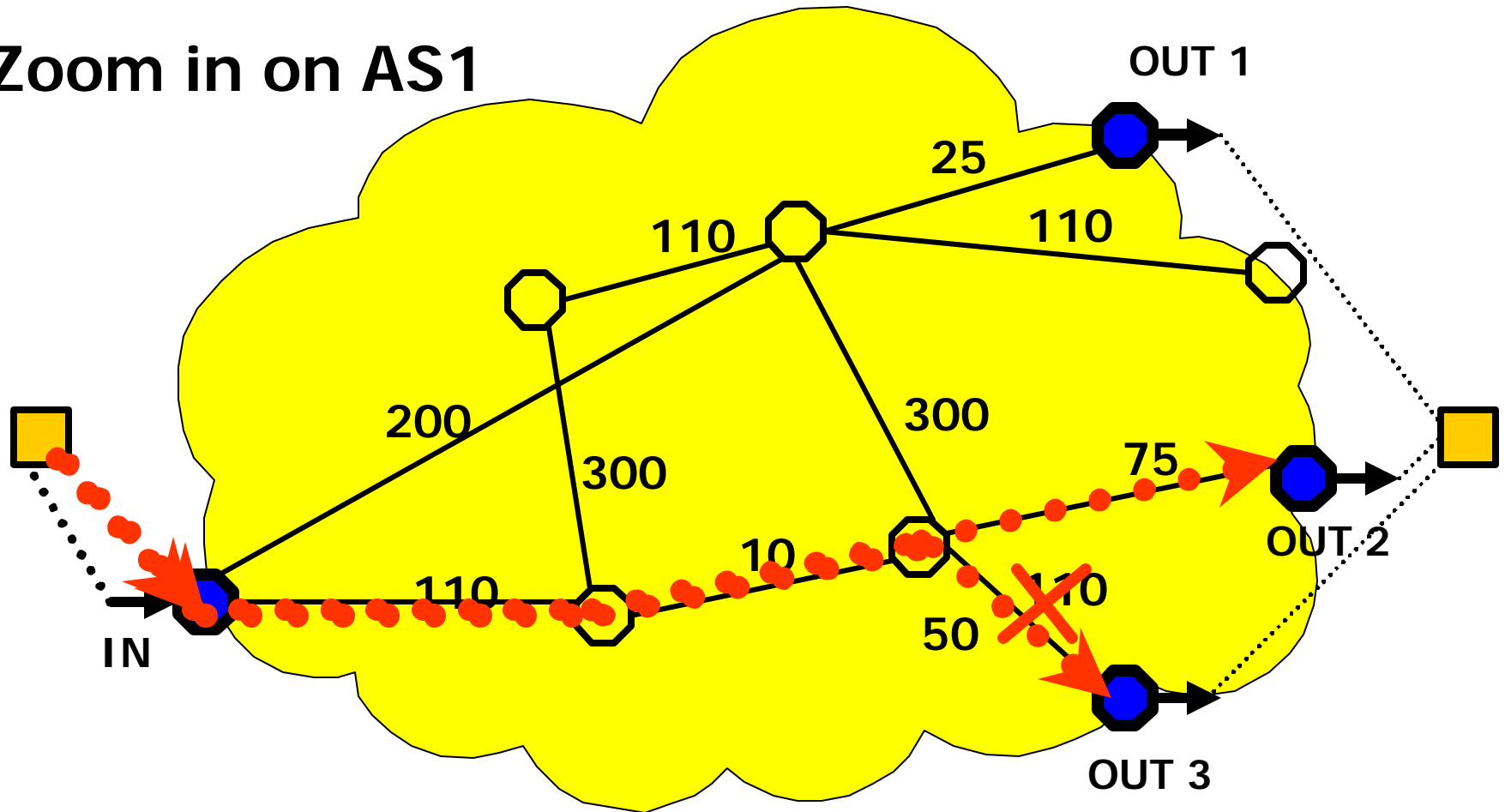


# Coupling of Inter and Intradomain Routing



# Intradomain Routing: “Hot Potato”

Zoom in on AS1



Hot-potato routing: change in **intradomain** routing (link weights) changes the traffic's **egress** point!

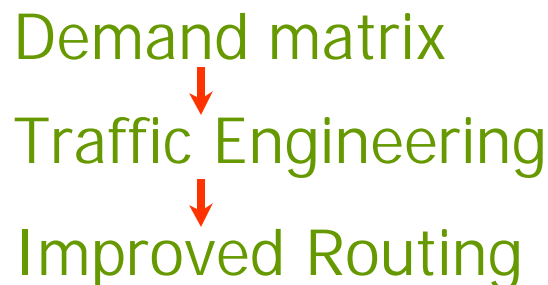
# Demand Model: Operational Uses

## ◆ Coupling problem with traffic matrix approach



## ◆ Demands: # bytes for each (in, {out\_1,...,out\_m})

- ingress link (in)
- set of possible egress links ({out\_1,...,out\_m})



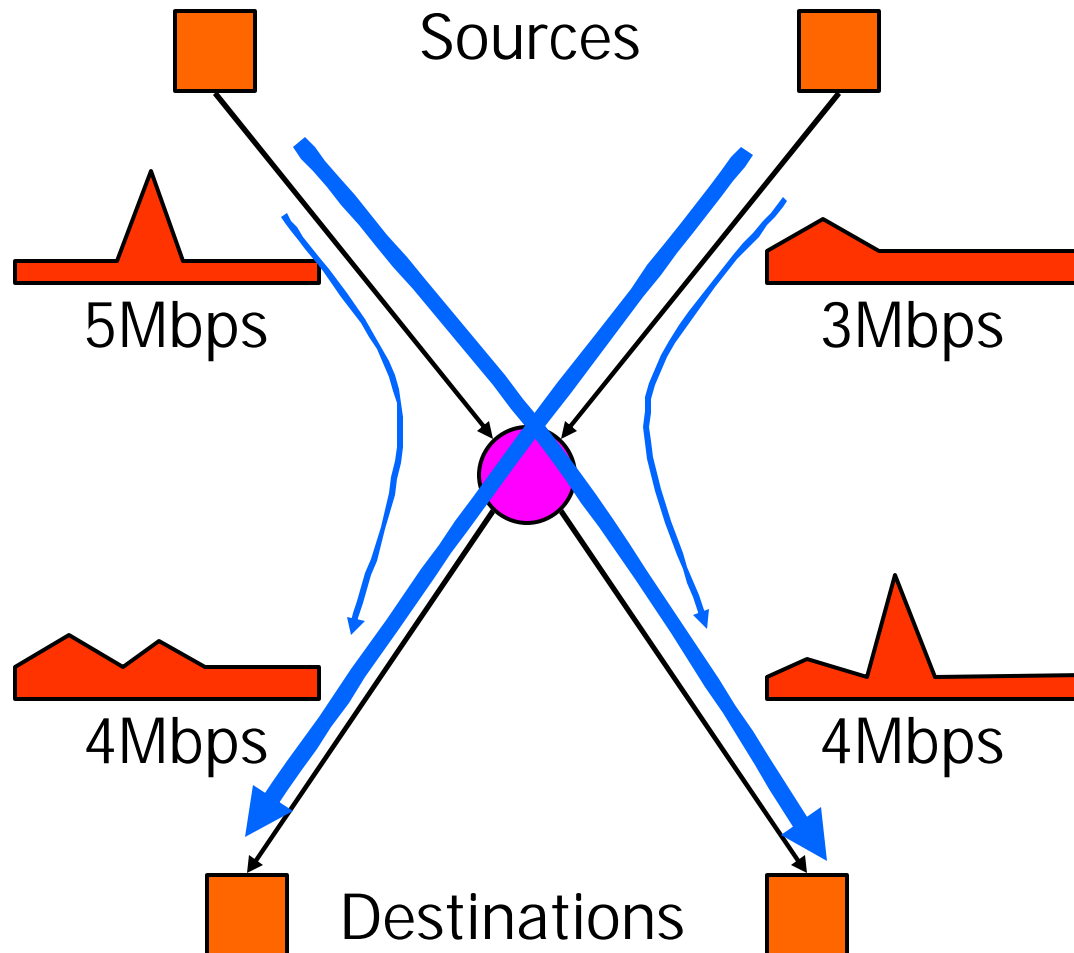
# Populating the Domain-Wide Models

---

- ◆ **Inference:** assumptions about traffic and routing
  - Traffic data: byte counts per link (over time)
  - Routing data: path(s) between each pair of nodes
- ◆ **Mapping:** assumptions about routing
  - Traffic data: packet/flow statistics at network edge
  - Routing data: egress point(s) per destination prefix
- ◆ **Direct observation:** no assumptions
  - Traffic data: packet samples at every link
  - Routing data: none

# Inference: Network Tomography

From link counts to the traffic matrix



# Tomography: Formalizing the Problem

---

## ◆ Ingress-egress pairs

- $p$  is a ingress-egress pair of nodes
- $x_p$  is the (unknown) traffic volume for this pair

## ◆ Routing

- $R_{lp} = 1$  if link  $l$  is on the path for ingress-egress pair  $p$
- Or,  $R_{lp}$  is the proportion of  $p$ 's traffic that traverses  $l$

## ◆ Links in the network

- $l$  is a unidirectional edge
- $y_l$  is the observed traffic volume on this link

## ◆ Relationship: $y = Rx$ (now work back to get $x$ )

# Tomography: Single Observation is Insufficient

---

## ◆ Linear system is underdetermined

- Number of nodes  $n$
- Number of links  $e$  is around  $O(n)$
- Number of ingress-egress pairs  $c$  is  $O(n^2)$
- Dimension of solution sub-space at least  $c - e$

## ◆ One approach: multiple observations

- $k$  independent observations (over time)
- Stochastic model with Poisson iid ingress/egress counts
- Maximum likelihood estimation to infer traffic matrix
- Vardi, “Network Tomography,” *JASA*, March 1996

# Tomography: Challenges

---

## ◆ Limitations

- Inaccurate traffic assumptions
- Significant estimation error
- High computation overhead
- Cannot handle packet loss or multicast traffic

## ◆ Directions for future work

- Partial queries over subgraphs
- Incorporating additional measurement data
- More realistic traffic assumptions...

# Promising Extension: Gravity Models

---

- ◆ **Gravitational assumption**
  - Ingress point  $a$  has traffic  $v_a^i$
  - Egress point  $b$  has traffic  $v_b^e$
  - Pair  $(a,b)$  has traffic proportional to  $v_a^i * v_b^e$
- ◆ **Incorporating hot-potato routing**
  - Combine traffic across egress points to the same peer
  - Gravity divides  $a$ 's traffic proportional to peer loads
  - “Hot potato” identifies single egress point for  $a$ 's traffic
- ◆ **Experimental results [SIGMETRICS'03]**
  - Reasonable accuracy, especially for large  $(a,b)$  pairs
  - Sufficient accuracy for traffic engineering applications

# Mapping: Remove Traffic Assumptions

---

## ◆ Assumptions

- Know the egress point where traffic leaves the domain
- Know the path from the ingress to the egress point

## ◆ Approach

- Collect fine-grain measurements at ingress points
- Associate each record with path and egress point
- Sum over measurement records with same path/egress

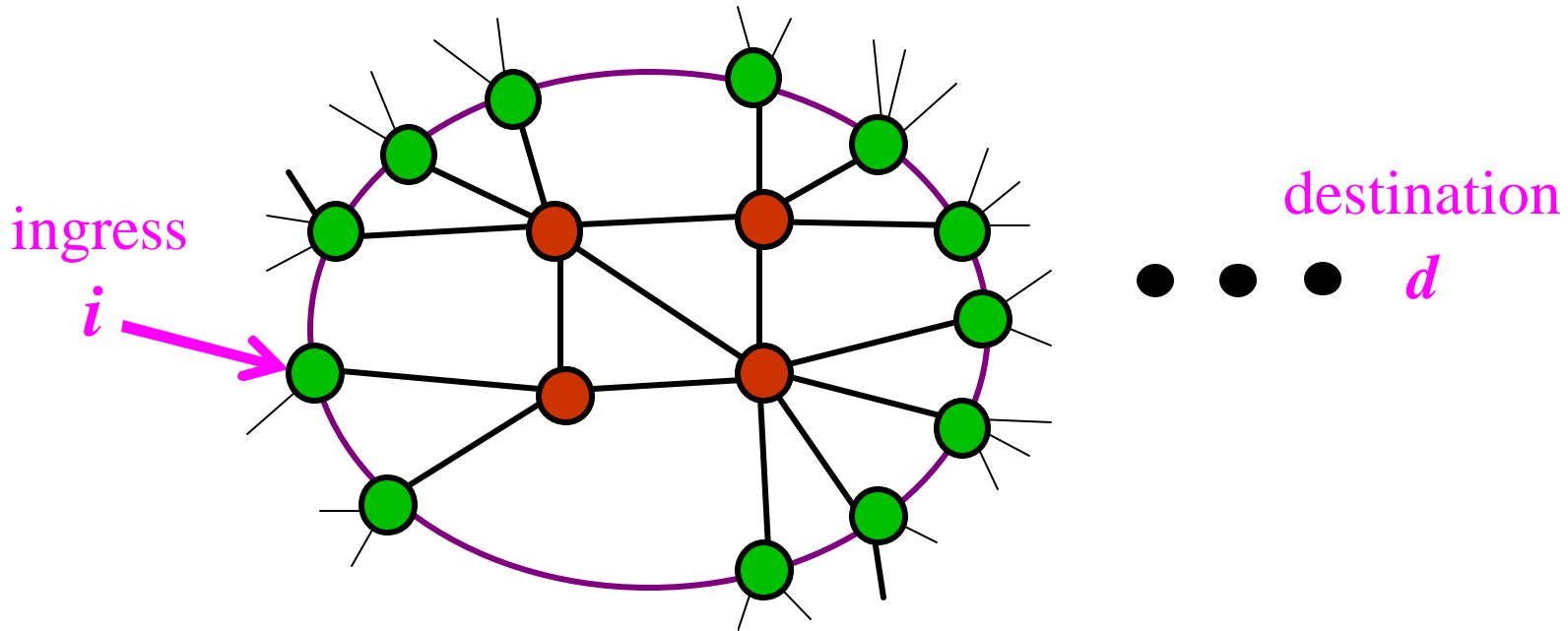
## ◆ Requirements

- Packet or flow measurement at the ingress points
- Routing table from each of the egress points

# Traffic Mapping: Ingress Measurement

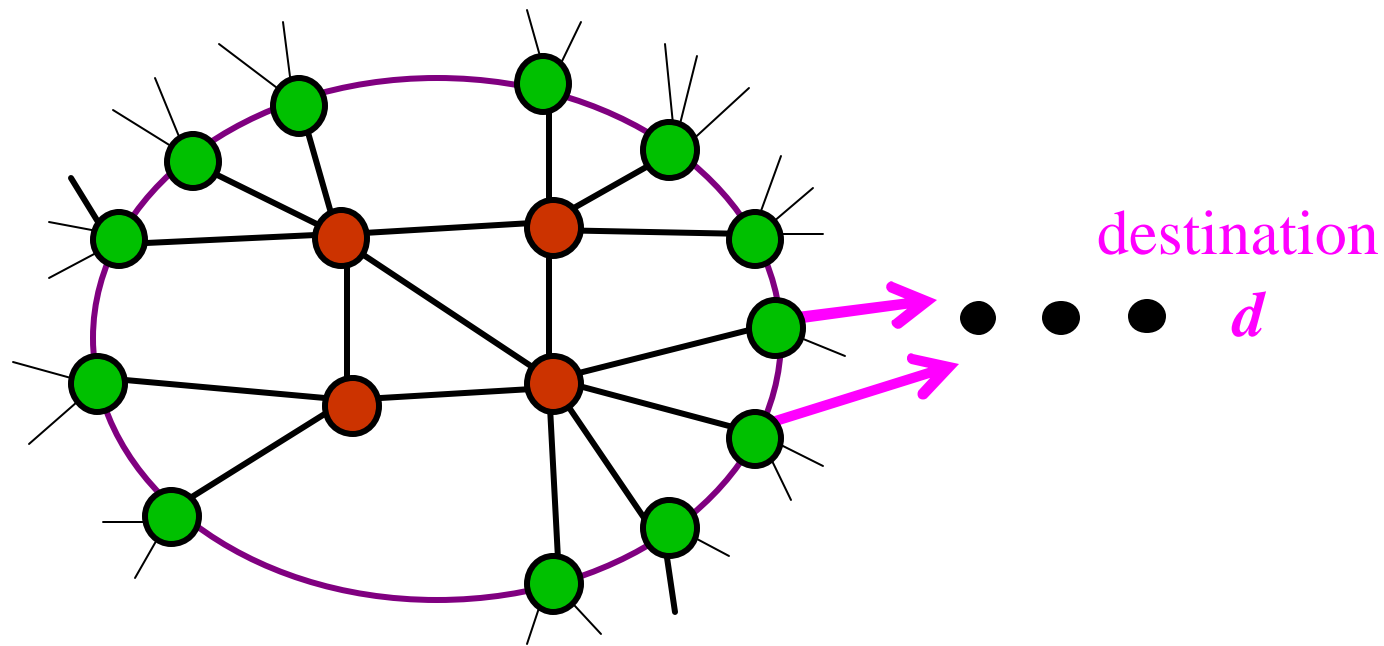
## ◆ Traffic measurement data (e.g., Netflow)

- Ingress point  $i$
- Destination prefix  $d$
- Traffic volume  $V_{id}$



# Traffic Mapping: Egress Point(s)

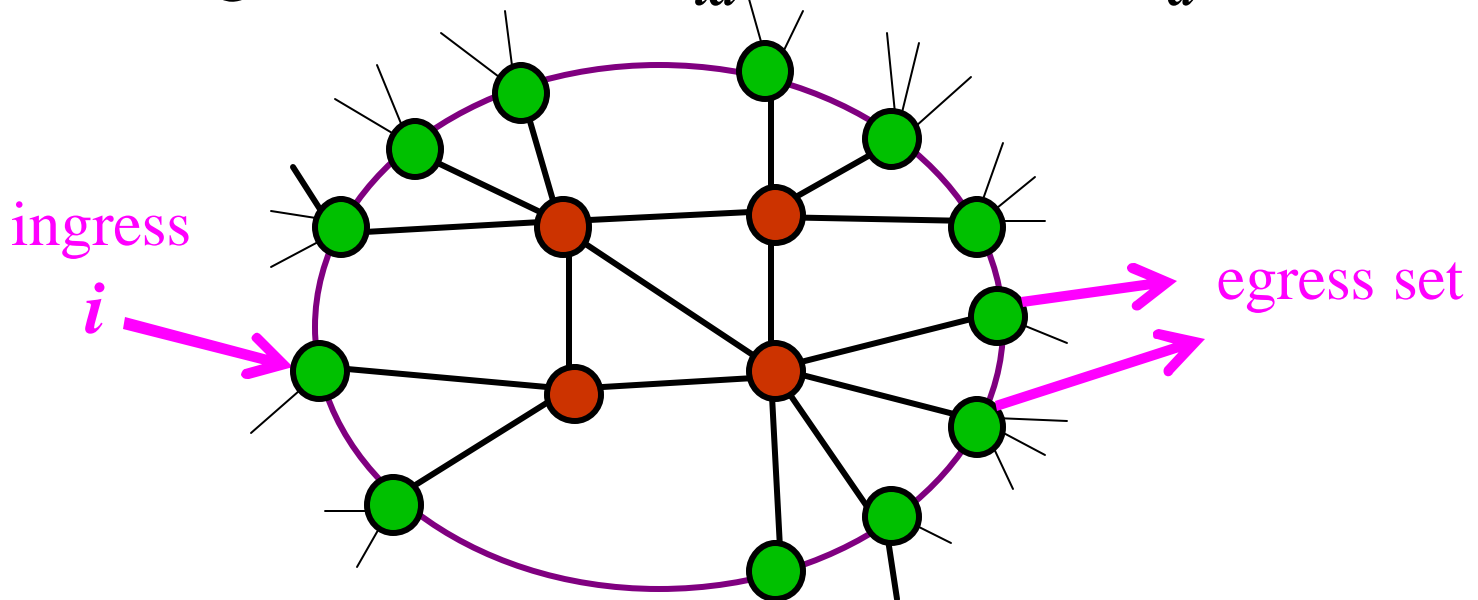
- ◆ Routing data (e.g., router forwarding tables)
  - Destination prefix  $d$
  - Set of egress points  $e_d$



# Traffic Mapping: Combining the Data

## ◆ Combining multiple types of data

- Traffic:  $V_{id}$  (ingress  $i$ , destination prefix  $d$ )
- Routing:  $e_d$  (set  $e_d$  of egress links toward  $d$ )
- Combining: sum over  $V_{id}$  with same  $e_d$



# Mapping: Challenges

---

## ◆ Limitations

- Need for fine-grain data from ingress points
- Large volume of traffic measurement data
- Need for forwarding tables from egress point
- Data inconsistencies across different locations

## ◆ Directions for future work

- Vendor support for packet measurement (*psamp*)
- Distributed infrastructure for collecting data
- Online monitoring of topology and routing data

# Direct Observation: Overcoming Uncertainty

---

## ◆ Internet traffic

- Fluctuation over time (burstiness, congestion control)
- Packet loss as traffic flows through the network
- Inconsistencies in timestamps across routers

## ◆ IP routing protocols

- Changes due to failure and reconfiguration
- Large state space (high number of links or paths)
- Vendor-specific implementation (e.g., tie-breaking)
- Multicast groups that send to (dynamic) set of receivers

## ◆ Better to observe the traffic *directly* as it travels

# Direct Observation: Straw-Man Approaches

---

## ◆ Path marking

- Each packet carries the path it has traversed so far
- Drawback: excessive overhead

## ◆ Packet or flow measurement on every link

- Combine records across all links to obtain the paths
- Drawback: excessive measurement and CPU overhead

## ◆ Sample the entire path for certain packets

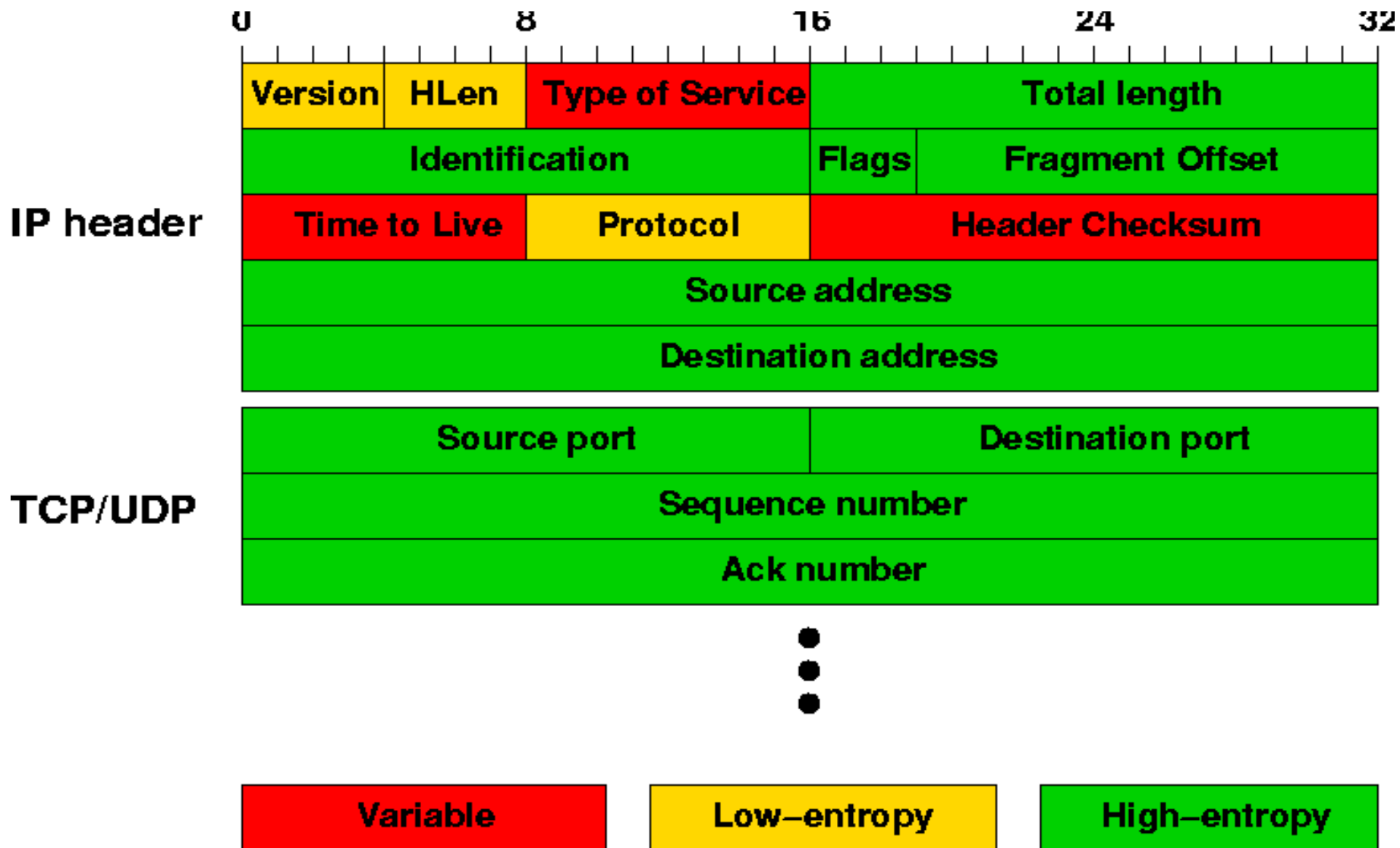
- Sample and tag a fraction of packets at ingress point
- Sample all of the tagged packets inside the network
- Drawback: requires modification to IP (for tagging)

# Direct Observation: Trajectory Sampling

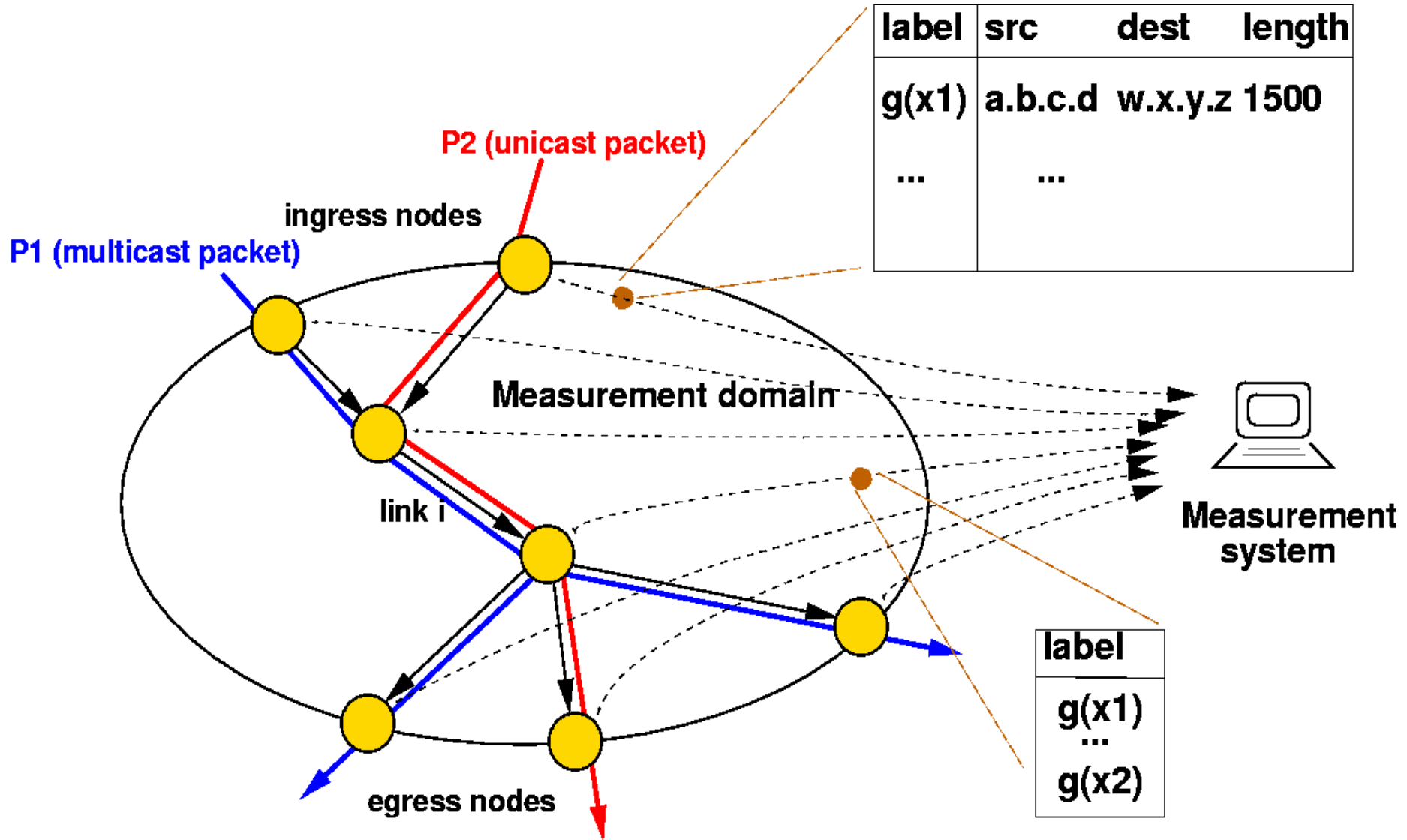
---

- ◆ Sample packets at every link without tagging
  - Pseudo random sampling (e.g., 1-out-of-100)
  - Either sample or don't sample at each link
  - Compute a hash over the contents of the packet
- ◆ Details of consistent sampling
  - $\mathbf{x}$ : subset of invariant bits in the packet
  - Hash function:  $h(\mathbf{x}) = \mathbf{x} \bmod A$
  - Sample if  $h(\mathbf{x}) < r$ , where  $r/A$  is a thinning factor
- ◆ Exploit entropy in packet contents to do sampling

# Trajectory Sampling: Fields Included in Hashes



# Trajectory Sampling



# Trajectory Sampling: Summary

---

## ◆ Advantages

- Estimation of the path and traffic matrices
- Estimation of performance statistics (loss, delay, etc.)
- No assumptions about routing or traffic
- Applicable to multicast traffic and DoS attacks
- Flexible control over measurement overhead

## ◆ Disadvantages

- Requires new support on router interface cards (*psamp*)
- Requires use of the same hash function at each hop

# Example Application: Intradomain Traffic Engineering

Route emulation,  
Objective function

Network-wide  
"what-if" model

Topology,  
Link weights

Traffic  
demands

Weight  
changes

Topology/  
Configuration

Offered  
traffic

Changes to  
the network

measure

control

Operational network

```
graph TD; subgraph Model; direction TB; A[Network-wide "what-if" model]; end; subgraph Inputs; direction TB; B[Topology/ Configuration]; C[Offered traffic]; end; subgraph Outputs; direction TB; D[Changes to the network]; end; subgraph Network; direction TB; E[Operational network]; end; B -- "Topology, Link weights" --> A; C -- "Traffic demands" --> A; A -- "Weight changes" --> D; E -- "measure" --> B; E -- "measure" --> C; D -- "control" --> E;
```

# Traffic Engineering by Tuning Link Weights

---

## ◆ Measured inputs

- Traffic demands
- Network topology and link weights

## ◆ Objective function

- Max link utilization
- Sum of  $\exp(\text{utilization})$

## ◆ “What-if” model of intradomain routing

- Select a closest exit point based on link weights
- Compute shortest path(s) based on link weights
- Capture traffic splitting over multiple shortest paths

# Weight Optimization

---

## ◆ Local search

- Generate a candidate setting of the weights
- Predict the resulting load on the network links
- Compute the value of the objective function
- Repeat, and select solution with min objective function

## ◆ Efficient computation

- Explore the “neighborhood” around good solutions
- Exploit efficient incremental graph algorithms

## ◆ Performance on AT&T's network

- Much better using link capacity or physical distance
- Quite competitive with multi-commodity flow solution

## Incorporating Operational Realities

---

- ◆ Minimize changes to the network
  - Changing just one or two link weights is often enough
- ◆ Tolerate failure of network equipment
  - Weights settings usually remain good after failure
  - ... or can be fixed by changing one or two weights
- ◆ Limit the number of distinct weight values
  - Small number of integer values is sufficient
- ◆ Limit dependence on accuracy of traffic demands
  - Good weights remain good despite random noise
- ◆ Limit frequency of changes to the weights
  - Joint optimization for day and night traffic matrices

# Conclusions

---

- ◆ Operating IP networks is challenging
  - IP networks stateless, best-effort, heterogeneous
  - Operators lack end-to-end control over the path
  - IP was not designed with measurement in mind
- ◆ Domain-wide traffic models
  - Needed to detect, diagnose, and fix problems
  - Models: path, traffic, and demand matrices
  - Techniques: inference, mapping, & direct observation
  - Optimization of routing configuration to the traffic
- ◆ <http://www.research.att.com/~jrex/papers/sfi.ps>

# Interesting Research Problems

---

## ◆ Packet/flow sampling

- Traffic and performance statistics from sampled data
- Analysis of trade-off between overhead and accuracy

## ◆ Anomaly detection

- Identifying fluctuations in traffic (and routing) data
- Analyzing the data for root cause analysis

## ◆ Populating the domain-wide models

- New techniques, and combinations of techniques
- Working with a mixture of different types of data

## ◆ Route optimization

- Influence of inaccurate demand estimates on results
- Optimization under traffic fluctuation and failures