Traffic Monitoring, Estimation, and Engineering

Nick Feamster
CS 7260
February 19, 2007

Administrivia

- Turn in project proposals by midnight tonight
- If you are still at a loss for project ideas, come see me at office hours today

Today's Topics

- Traffic monitoring
  - Flow monitoring
    - Mechanics
    - Limitations
  - Estimation / Problem formulation

- Traffic engineering
  - Intradomain (*today's paper*)
  - Interdomain

Traffic Flow Statistics

- *Flow monitoring* (*e.g.*, Cisco Netflow)
  - Statistics about groups of related packets (*e.g.,* same IP/TCP headers and close in time)
  - Recording header information, counts, and time

- More detail than SNMP, less overhead than packet capture
  - Typically implemented directly on line card

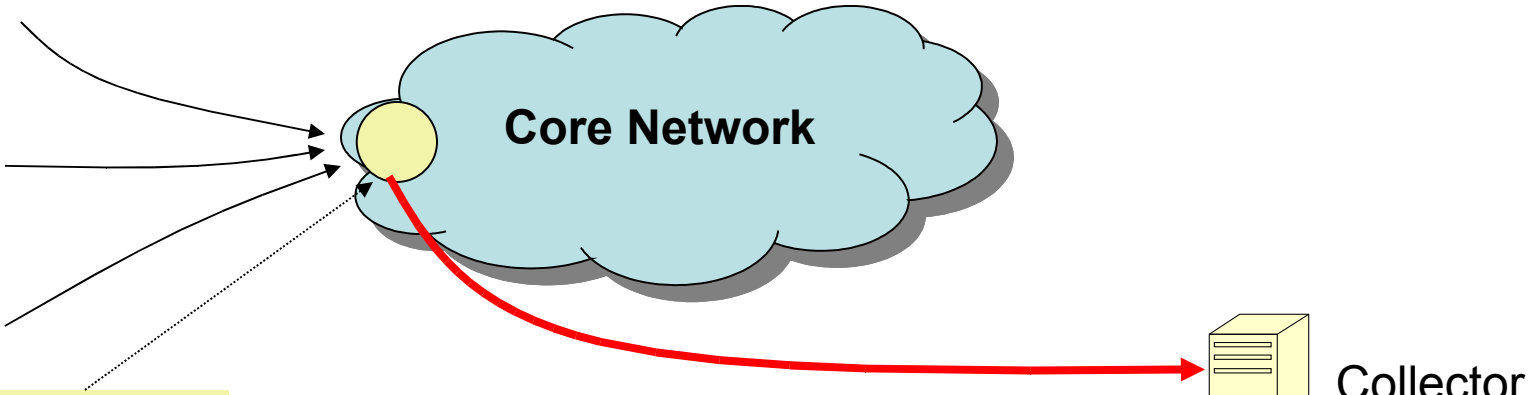
What is a flow?

- **Source IP address**
- **Destination IP address**
- **Source port**
- **Destination port**
- **Layer 3 protocol type**
- TOS byte (DSCP)
- Input logical interface (ifIndex)

Cisco Netflow

- Basic output: “Flow record”
  - Most common version is v5
  - Latest version is v10 (RFC 3917)

- Current version (10) is being standardized in the IETF (*template-based*)
  - More flexible record format
  - Much easier to add new flow record types



Flow Record Contents

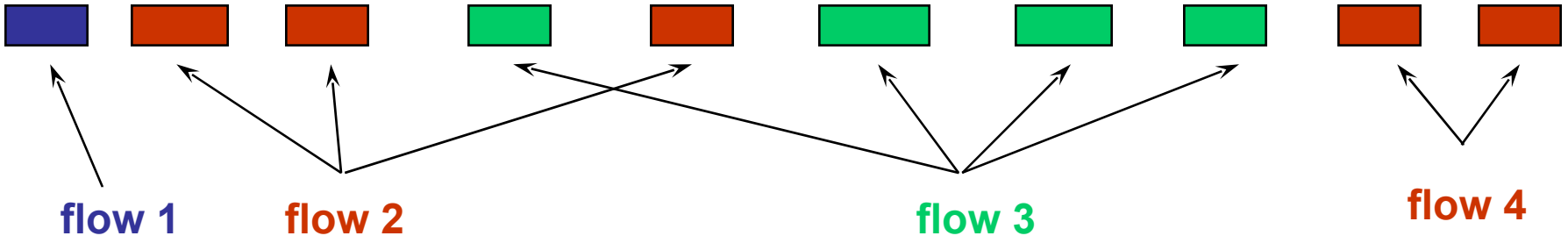
Basic information about the flow…

- Source and Destination, IP address and port
- Packet and byte counts
- Start and end times
- ToS, TCP flags

…plus, information related to routing

- Next-hop IP address
- Source and destination AS
- Source and destination prefix

Aggregating Packets into Flows



- **Criteria 1:** 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)

- **Criteria 2:** Packets that are "close" together in time
  - Maximum inter-packet spacing (e.g., 15 sec, 30 sec)
  - **Example:** flows 2 and 4 are different flows due to time

Netflow Processing

1. Create and update flows in NetFlow Cache

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	SrcPort	SrcMsk	SrcAS	DstPort	DstMsk	DstAS	NextHop	Bytes/Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1745	4
Fa1/0	173.100.3.2	Fa0/0	10.0.227.12	6	40	0	2491	15	/26	196	15	/24	15	10.0.23.2	740	41.5	1
Fa1/0	173.100.20.2	Fa0/0	10.0.227.12	11	80	10	10000	00A1	/24	180	00A1	/24	15	10.0.23.2	1428	1145.5	3
Fa1/0	173.100.6.2	Fa0/0	10.0.227.12	6	40	0	2210	19	/30	180	19	/24	15	10.0.23.2	1040	24.5	14

1. Expiration

- Inactive timer expired (15 sec is default)
- Active timer expired (30 min (1800 sec) is default)
- NetFlow cache is full (oldest flows are expired)
- RST or FIN TCP Flag

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	SrcPort	SrcMsk	SrcAS	DstPort	DstMsk	DstAS	NextHop	Bytes/Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1800	4

1. Aggregation?

No

Yes

e.g. Protocol-Port Aggregation Scheme becomes

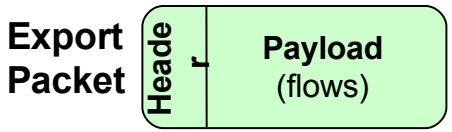
Protocol	Pkts	SrcPort	DstPort	Bytes/Pkt
11	11000	00A2	00A2	1528

1. Export Version

Non-Aggregated Flows – export **Version 5 or 9**

Aggregated Flows – export **Version 8 or 9**

1. Transport Protocol

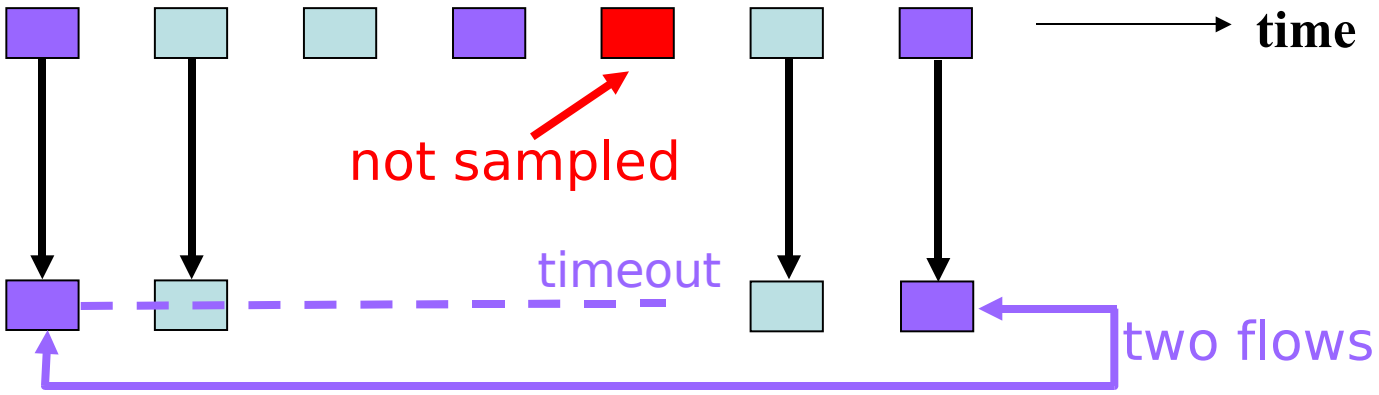


Reducing Measurement Overhead

- **Filtering:** on interface
  - destination prefix for a customer
  - port number for an application (e.g., 80 for Web)
- **Sampling:** before insertion into flow cache
  - Random, deterministic, or hash-based sampling
  - 1-out-of-n or stratified based on packet/flow size
  - *Two types:* packet-level and flow-level
- **Aggregation:** after cache eviction
  - packets/flows with same next-hop AS
  - packets/flows destined to a particular service

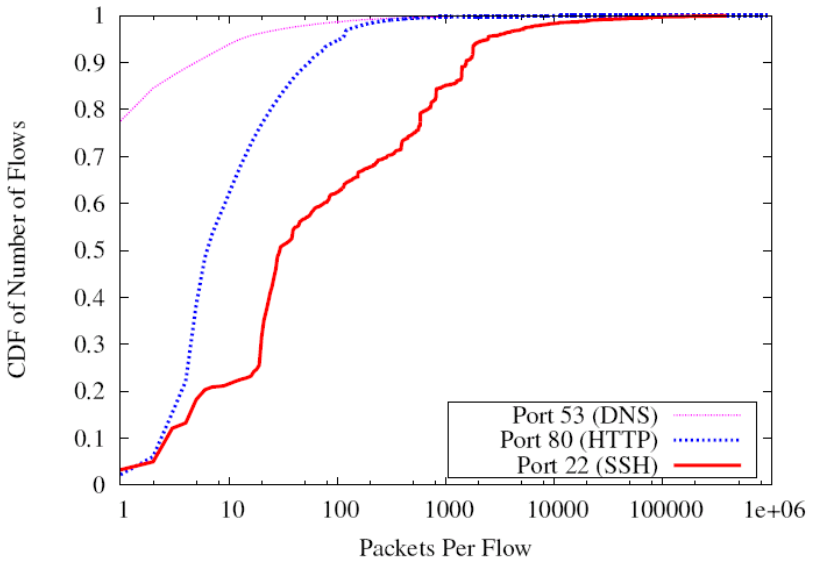
Packet Sampling

- 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



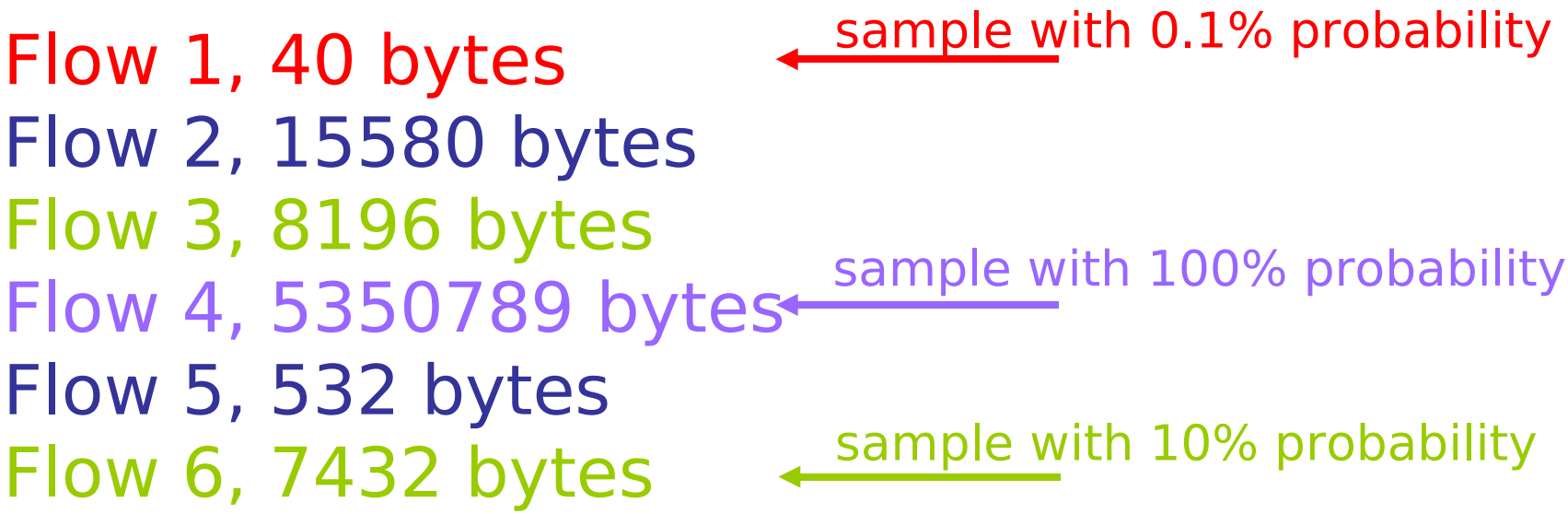
Problems with Packet Sampling

- Determining size of original flows is tricky
  - For a flow originally of size *n*, the size of the *sampled* flow follows a binomial distribution
  - Extrapoliation can result in big errors
  - Much research in reducing such errors (upcoming lectures)
- Flow records can be lost
- Small flows may be eradicated entirely



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



Accuracy Depends on Phenomenon

- Even naïve random sampling probably decent for capturing the *existence* of large flows

- Accurately measuring other features may require different approaches
  - Sizes of large flows
  - Distribution of flow sizes
  - Existence of small flows (coupon collection)
  - Size of small flows
  - Traffic “matrix”

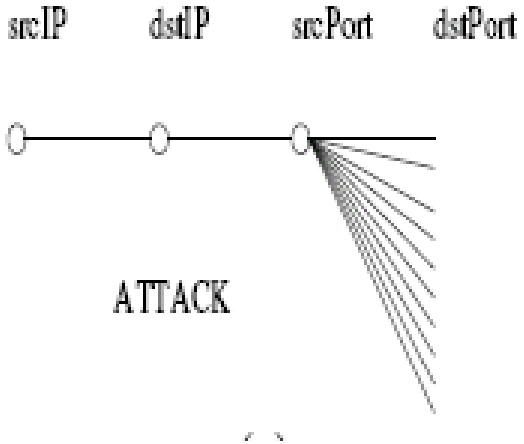
Recent Work: Targeted Sampling

- Set sampling rates based on estimate of flow sizes or classes

- **Flow sampling:** Size-dependent flow sampling (Duffield)

- **Packet sampling:** Sketch-Guided Sampling (Kumar), Flexible Sampling (Ramachandran)

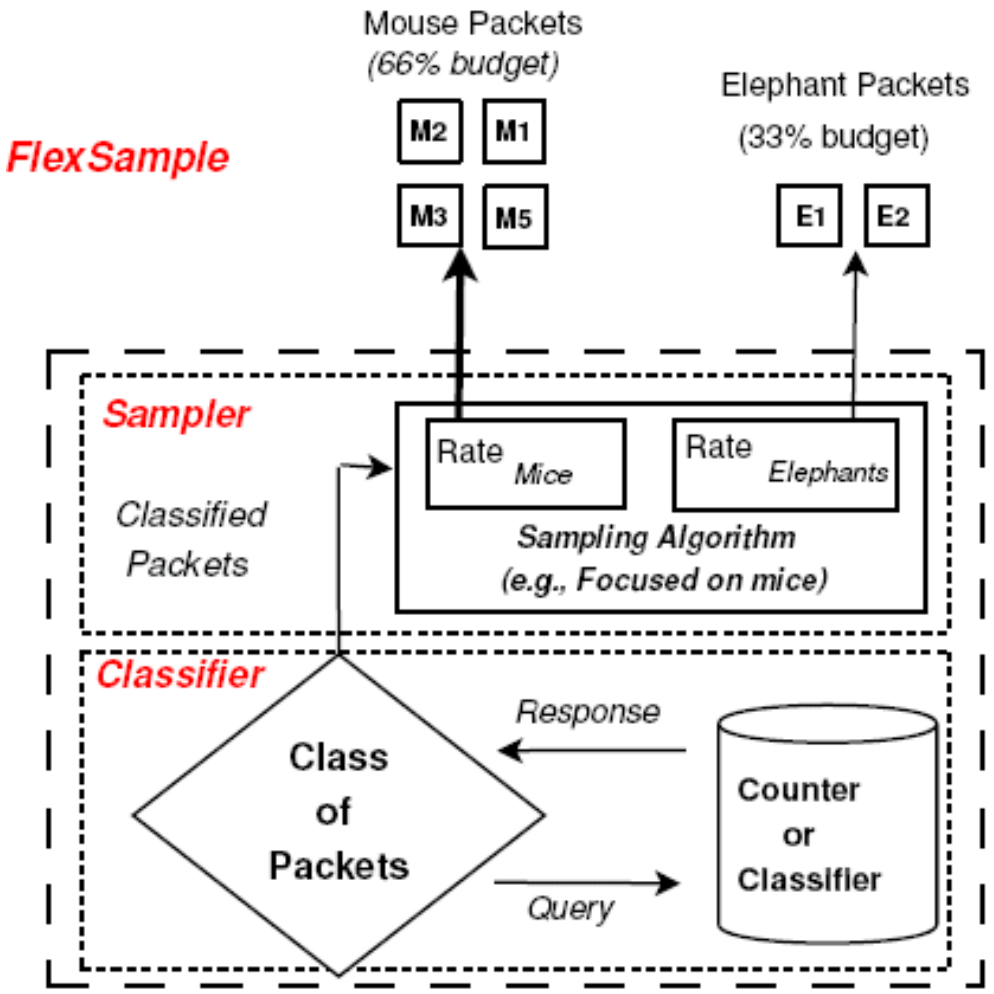
Motivation: Recover Structure



"BLINC: Multi-level Traffic Classification in the Dark", *SIGCOMM 2004*

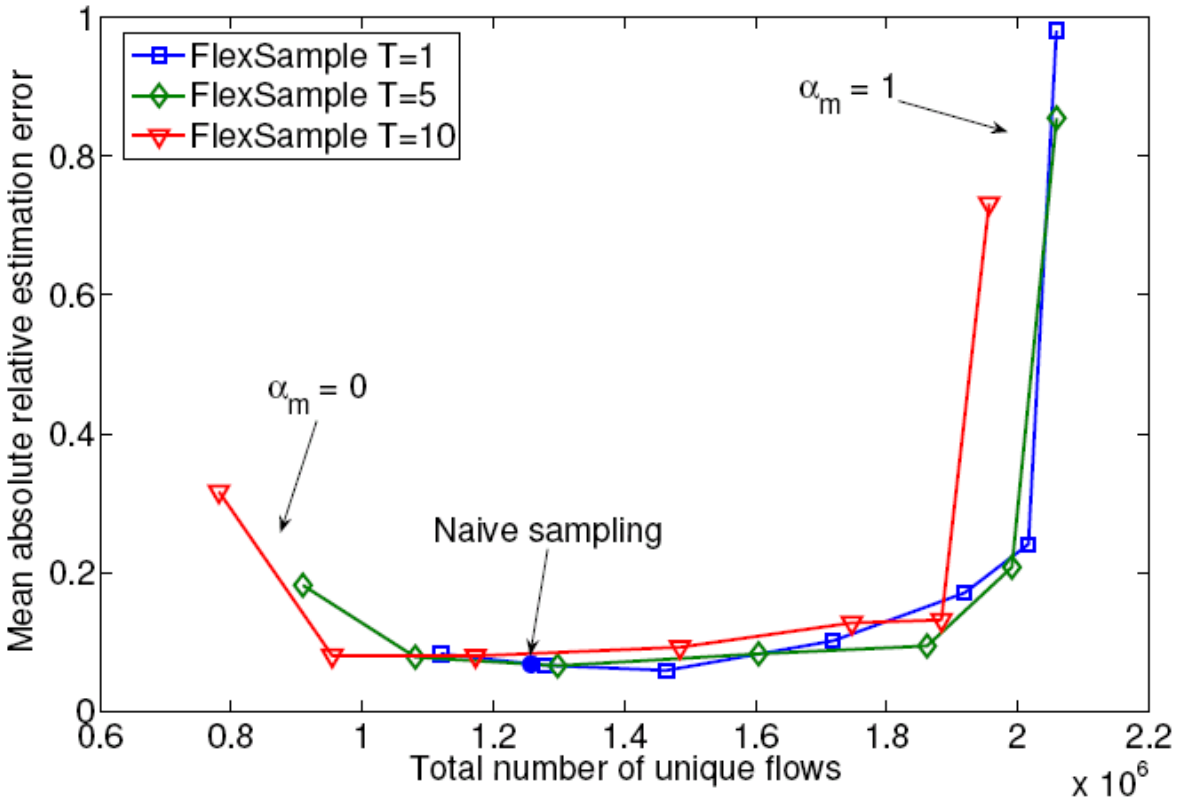
- Structure of "who's ralking to whom" can reveal helpful information about traffic patterns
- Can be difficult to get this from sampled flow records if the flows themselves are small

Idea: Augment Sampling with Counting



- Estimate, online, the *type/size* of a flow to which a packet belongs
- Adjust sampling rate according to corresponding class

Stealing Packets from Large Flows



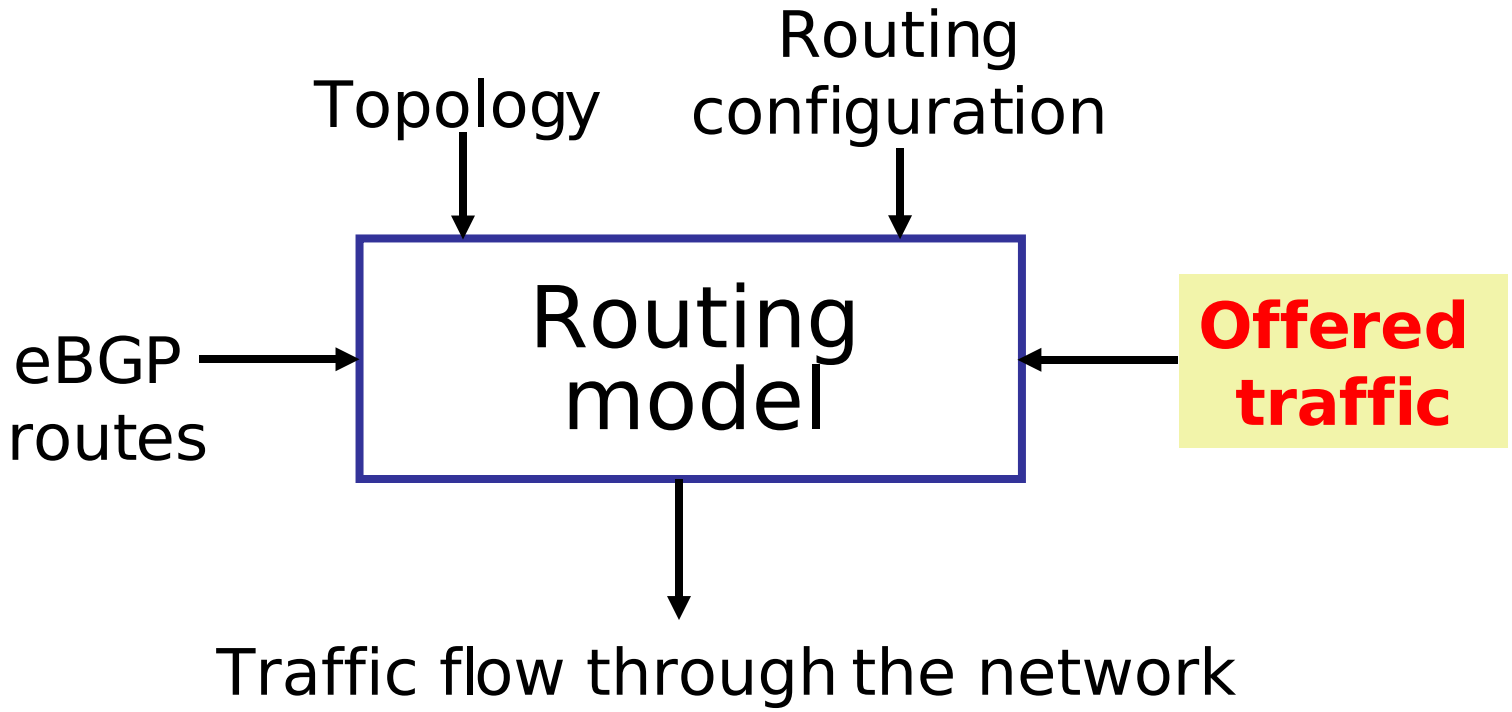
- Some packet samples can be "stolen" from large flows without incurring much estimation error
- Reallocating to small flows means more unique "conversations" captured

Traffic Engineering

Traffic Engineering Motivation

- Efficient use of resources (capacity, etc.)
- Response to dynamic network conditions
  - Changes in offered traffic load
  - Changes in capacity
  - Link failures, etc.
- Routing protocols do not (typically) automatically adapt to these changing conditions

Traffic Engineering Formulation



Two Types of Traffic Engineering

- **Intradomain:** Within a single AS
  - More “traditional” optimization problem
  - Challenges are in updating link weights

- **Interdomain:** Between multiple ASes
  - Ill-formed problem
  - Many more unknowns

Challenge: Dynamic Conditions

- Link state updates
  - High update rate leads to high overhead
  - Low update rate leads to oscillation
- Many connections are short
  - Average Web transfer is just 10 packets
  - Requires high update rates to ensure stability

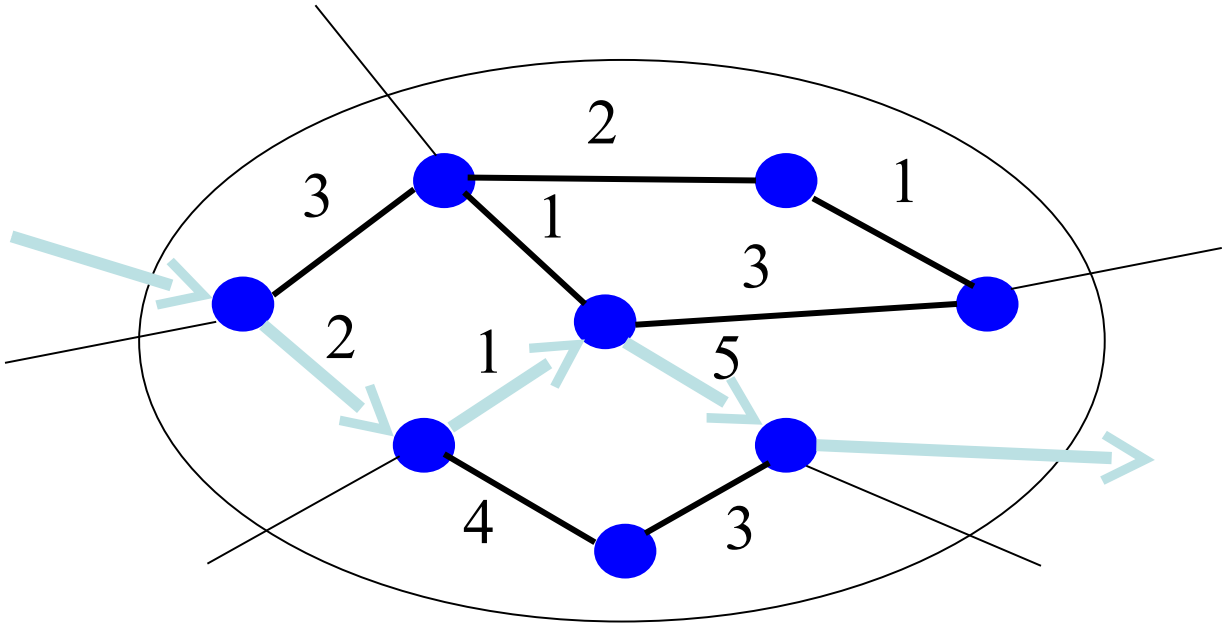
Information goes stale as network conditions change. Approaches for dealing with this?

Challenge: Cost of Reconfiguration

- Minimize number of changes to the network
  - Changing just 1 or 2 link weights is often enough
- Limit frequency of changes to the weights
  - Joint optimization for day & night traffic matrices
- Tolerate failure of network equipment
  - Weights settings usually remain good after failure
  - … or can be fixed by changing one or two weights
- Limit dependence on *accuracy* and *dynamics*

"Traditional" Intradomain TE

- Routers flood information to learn topology
  - Determine "next hop" to reach other routers…
  - Compute shortest paths based on link weights
- Link weights configured by network operator



Approaches: Local vs. Network-Wide

- **Local**
  - Proportional to physical distance
    - Cross-country links have higher weights
    - Minimizes end-to-end propagation delay
  - Inversely proportional to link capacity
    - Smaller weights for higher-bandwidth links
    - Attracts more traffic to links with more capacity
- **Network-Wide**
  - Network-wide optimization of the link weights
  - Directly minimizes metrics like max link utilization
  - Assumes accesss to network-wide information

Approaches: Offline vs. Dynamic

- Offline
  - Monitor trends and macro patterns in utilization
  - Perform offline optimization
  - **Advantage:** No protocol modification. No stability issues. Flexibility
  - **Disadvantage:** Perhaps less reactive.

- Dynamic
  - Routers themselves monitor link utilization
  - Distributed/local reoptimization of link weights and route selection
  - **Advantage**: More reactive.
  - **Disadvantage:** Protocol modifications, stability issues, etc.

Traffic Engineering Approaches

	Offline	Dynamic
Centralized	**Intradomain:** Fortz/Rexford02 **Interdomain:** Feamster03	Routing control platform (Open questions in this area)
Distributed		**Intradomain:** Katabi (TeXCP) **Interdomain:** Mahajan (NP, nexit)

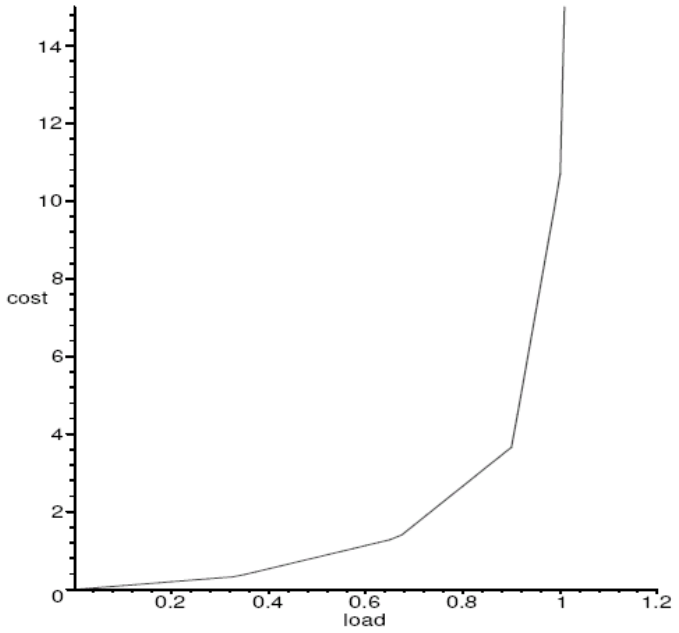
Formalization

- **Input:** Graph *G* with unidirectional links *l* and capacities for each link
- **Input:** **Objective function**
- **Input:** **traffic matrix**
  - $M_{i,j}$: traffic load from source *i* to destination *j*
  - For intradomain TE, source/destination are routers

- **Output:** setting of the link weights
  - W_l: weight on unidirectional link *l*
  - $P_{i,j,l}$: fraction of traffic from *i* to *j* traversing link *l*

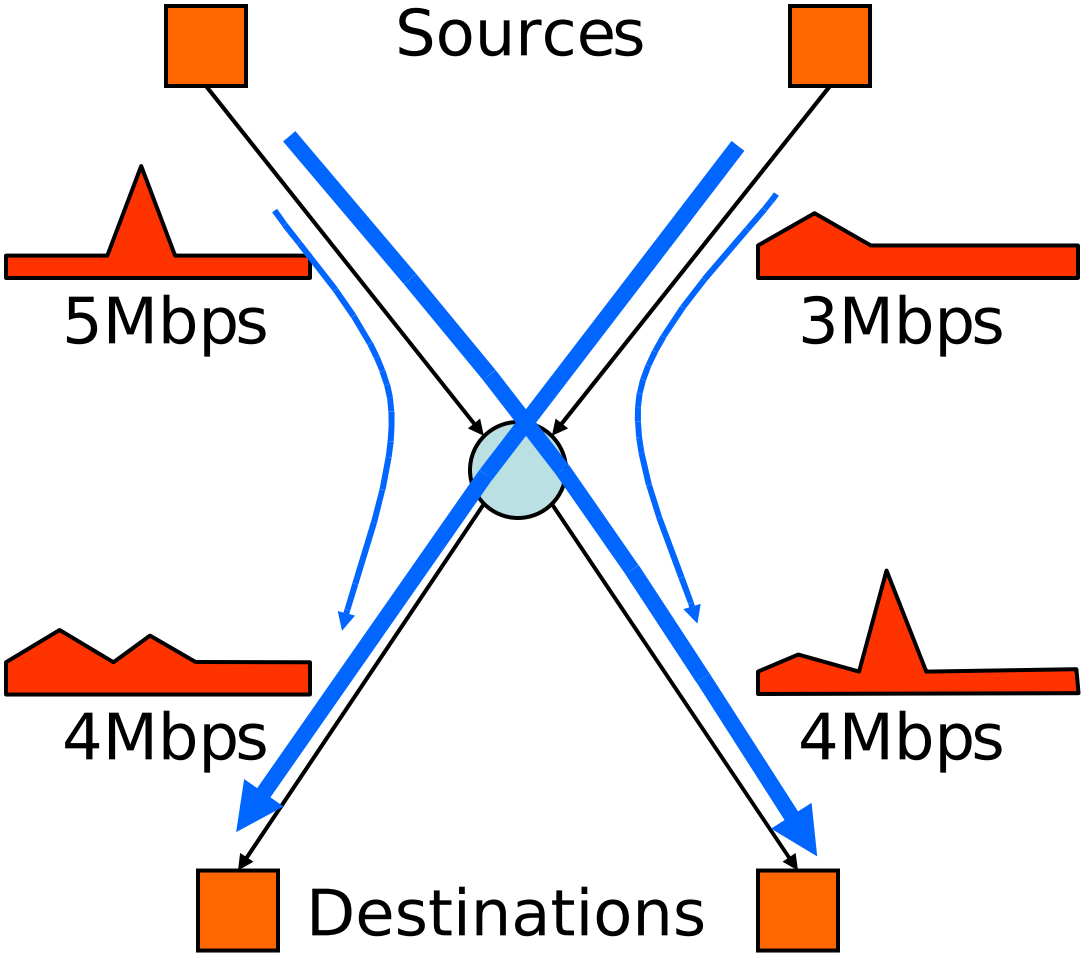
Objective Functions

- **Take 1:** Minimize maximum utilization
  - Individual bottleneck links may impose unrealistic/overly strong constraints
- **Take 2:** Fractional relation between load and total capacity



Traffic Matrix Estimation

From link counts to the traffic matrix



Formalization

- Source-destination pairs
  - ***p*** is a source-destination pair (of nodes)
  - $\boldsymbol{x}_p$ is the (unknown) traffic volume for this pair
- Links in the network
  - ***l*** is a unidirectional edge
  - $\boldsymbol{y}_l$ is the observed traffic volume on this link
- Routing
  - $\boldsymbol{R}_{lp}$ = 1 if link ***l*** is on the path for src-dest pair ***p***
  - Or, $\boldsymbol{R}_{lp}$ is the proportion of ***p***'s traffic that traverses ***l***
- $\boldsymbol{y} = \boldsymbol{R}\boldsymbol{x}$ (now work backwards to get ***x***)

Estimation is Under-Constrained

- Linear system is underdetermined
  - Number of nodes ***n***
  - Number of links ***e*** is around ***O(n)***
  - Number of src-dest pairs ***c*** is ***$O(n^2)$***
  - Dimension of solution sub-space at least ***c - e***

- Multiple observations can help
  - ***k*** independent observations (over time)
  - Stochastic model with src-dest counts Poisson & i.i.d
  - Maximum likelihood estimation to infer traffic matrix
  - Use NetFlow to augment byte counts can help constrain the problem

- **Lots of recent work on traffic matrix estimation**

Questions for Intra-Domain TE

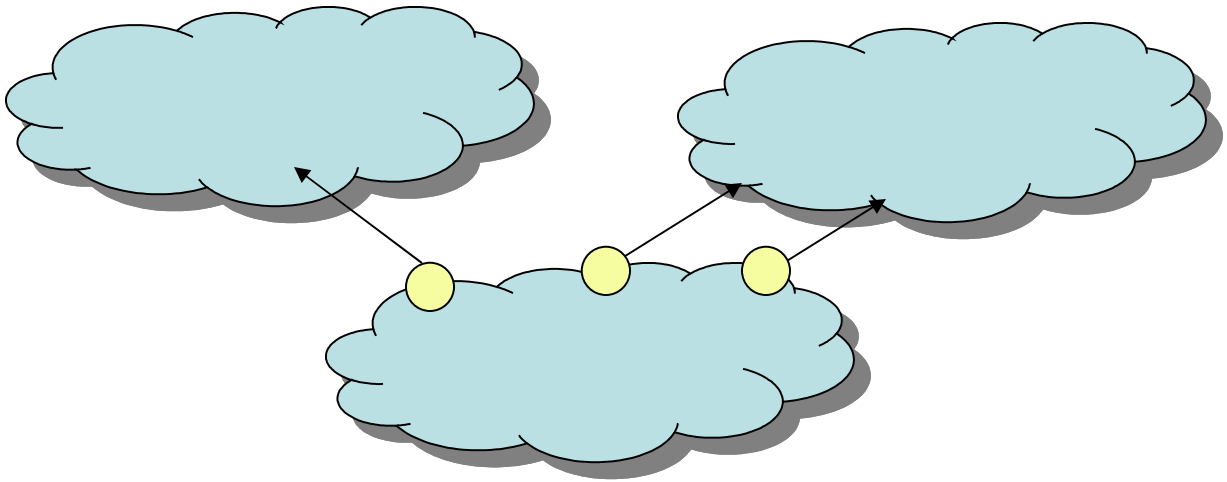
- Is the objective function reasonable? What about:
  - Resilience to failure?
  - Use of available capacity?
- How stable/fragile is the resulting optimal solution when traffic demands change?
  - Fluctuating traffic demands
- Setting link weights in this fashion is unintuitive and often not used in practice? Is there a better way to set link weights?

Interdomain Traffic Engineering

Interdomain Traffic Engineering

- Why?
  - Alleviating congestion on edge links
  - Adapting to provisioning changes
  - Achieving good end-to-end performance

- How?
  - Directing traffic to a different neighbor AS
  - Directing traffic to different links to the same neighbor



Overview of Method (Outbound)

- Change outbound traffic using BGP *import* policy

- **Requirements**
  - Minimal overhead (management and messages)
  - Predictable changes in traffic
  - No effect on neighboring ASes' routing decisions

Why Interdomain TE is Hard

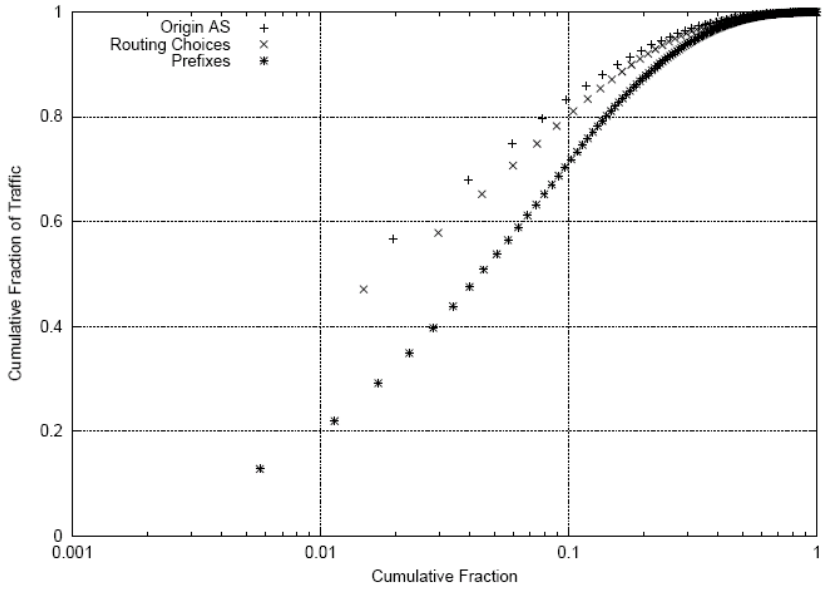
- **Scale:** Can't set independent policy for 200k+ prefixes
  - Configuration overhead
  - Traffic instability
- **Predictability:** Policy-based adjustments are indirect
- **Control:** Neighbors' behavior can affect traffic volumes in unpredictable and uncontrollable ways

Why Interdomain TE with BGP is Hard

- **Protocol** problems
  - No performance metrics in advertisement attributes
- **Configuration** problems
  - Not possible to express conjunction between attributes
  - Indirect influence
- **Route selection** problems
  - One best route per prefix per router
  - Can't split traffic to a prefix over paths of different lengths
  - Interaction with IGP
- **Commercial relationship** problems

Managing Scale

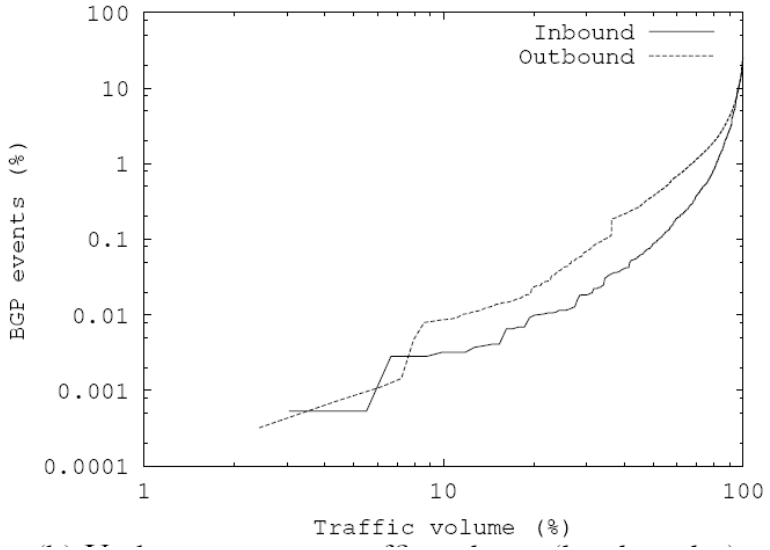
- **Problem:** Large number of prefixes
- **Solution:** Change policies for small fraction of prefixes that are responsible for most traffic



10% of prefixes responsible for 70% of traffic
More concentrated for origin ASes.

Achieving Predictability

- **Problem:** Traffic volumes change over time
- **Solution:** Change policies for prefxes that have more stable traffic volumes.



Origin ASes responsible for 10% of inbound traffic don't fluctuate by more than a factor or two between weeks.

Acheving Predictability

- **Problem:** Internal changes that are externally visible can change inbound traffic volumes
- **Solution:** Shift traffic among paths
  - To the same AS
  - To a different AS, but one with the same path length