Best Practices for Determining Traffic Matrices in IP Networks V 4.0

June 1 2008, 2:00pm-3:30pm

NANOG 43
Brooklyn, NY

Rached Blili & Arman Maghbouleh
Cariden Technologies, Inc.

cariden

Overview

Traffic Matrices

- In context of
  - Flows
  - Interface Stats
  - ASPaths
- Internal versus External
- per Customer, Per CoS, per Application

Methods

- Measurement
  - Netflow
  - RSVP
  - LDP
- Estimation via Tomogravity
- Practical Issues
- Regressed Measurements

- Notes
- Recommendations

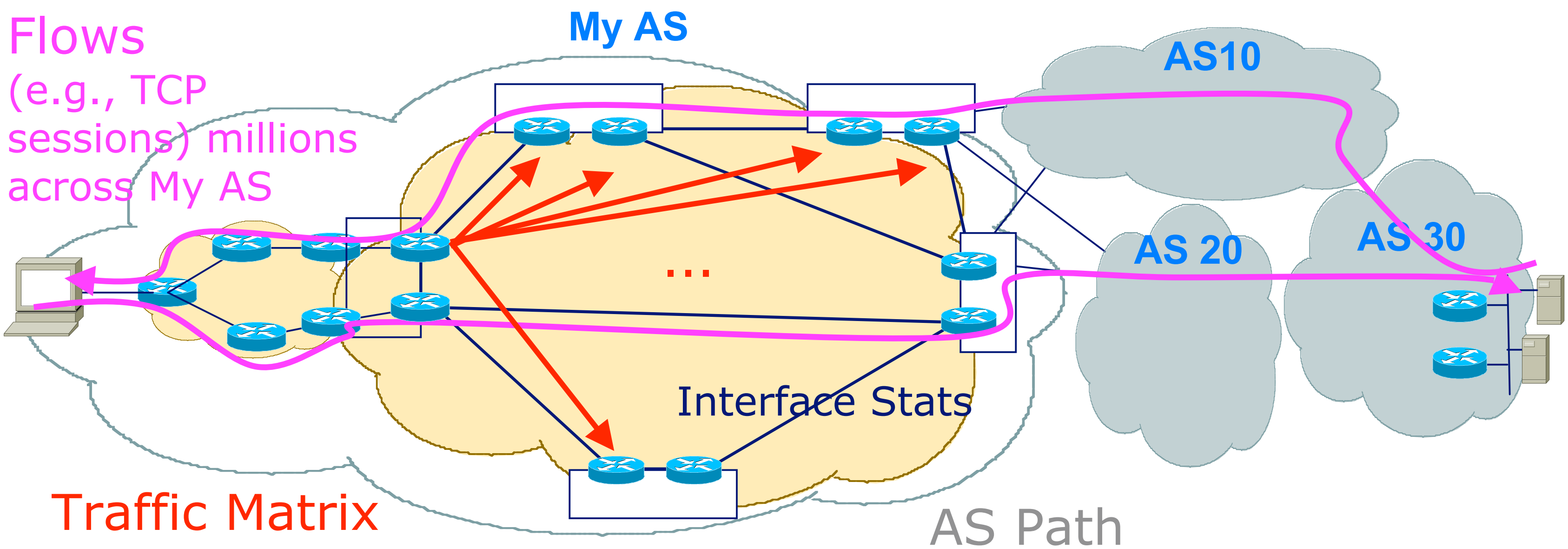
Contributors

- Thomas Telkamp, Cariden
  - Versions 1-3 of this tutorial
- Stefan Schnitter, *T-Systems*
  - MPLS/LDP, Partial topologies
- Benoit Claise, *Cisco Systems, Inc.*
  - Cisco NetFlow
- Per Gregers Bilse, Network Signature
  - Typical NetFlow statistics
- Mikael Johansson, *KTH*
  - Traffic matrix properties

Tutorial V3 to V4

- Removed Lesser-Used Methods
  - BGP Policy Accounting, DCU
- Added SNMP How-Tos for the Do-It-Yourselfers
- Expanded on the “Gotchas”
- Highlighted Regressed Measurement Separately
- Added Typical Cases and Recommendations
  - including when Traffic Matrix not needed

Terms



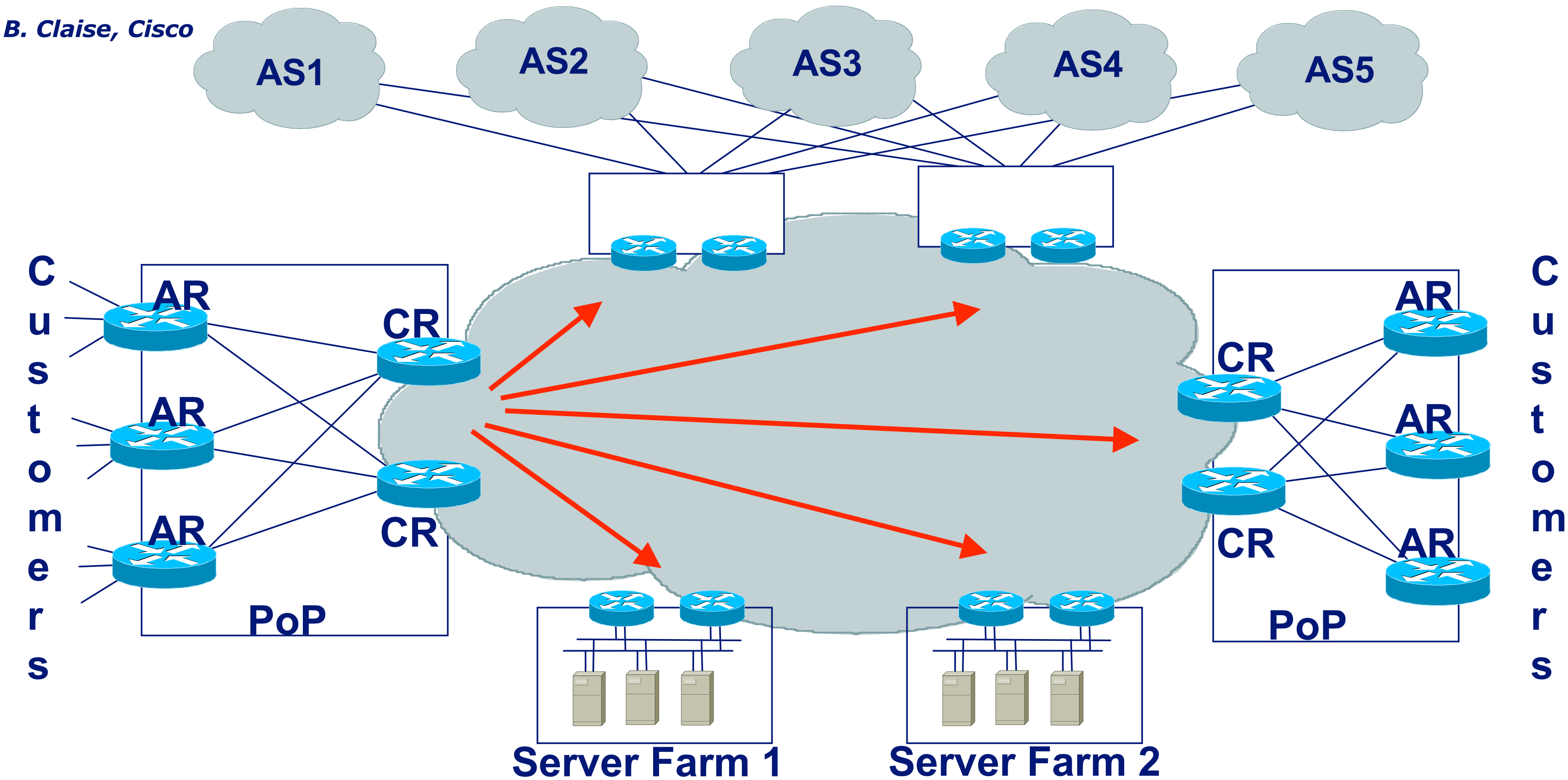
Traffic Matrix
Aggregation of flows at edge of network model.

Demand
A single element of a traffic matrix said to have "source" and "destination" in model.

AS Path
e.g., AS10;AS30 and AS20;AS30 as two possible paths for reaching the server in diagram.

Internal Traffic Matrix

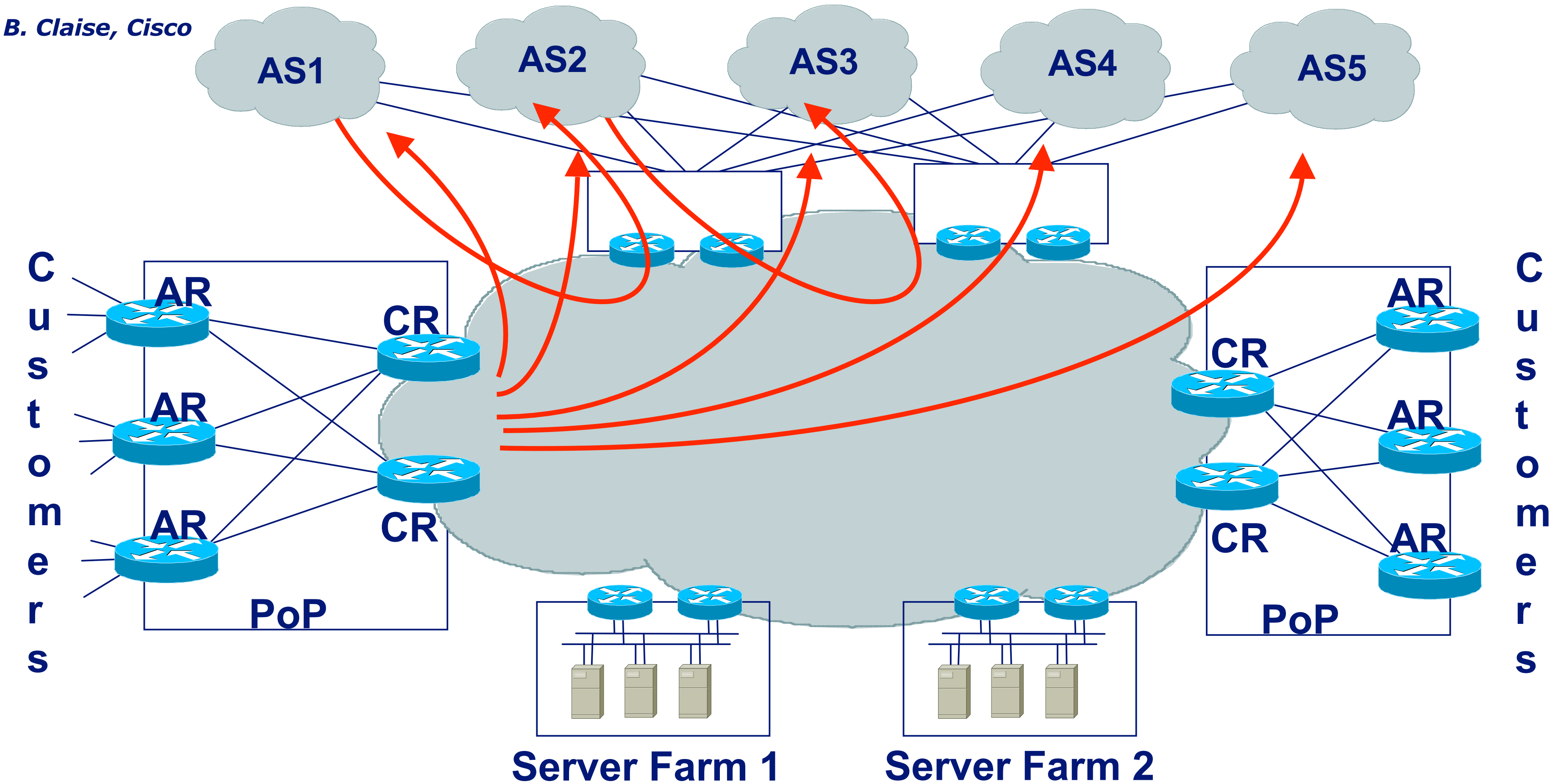
B. Claise, Cisco



Demands start and end in My AS.
Could be CR-to-CR, AR-to-AR, or ... (see T-Com example)

External Traffic Matrix

B. Claise, Cisco

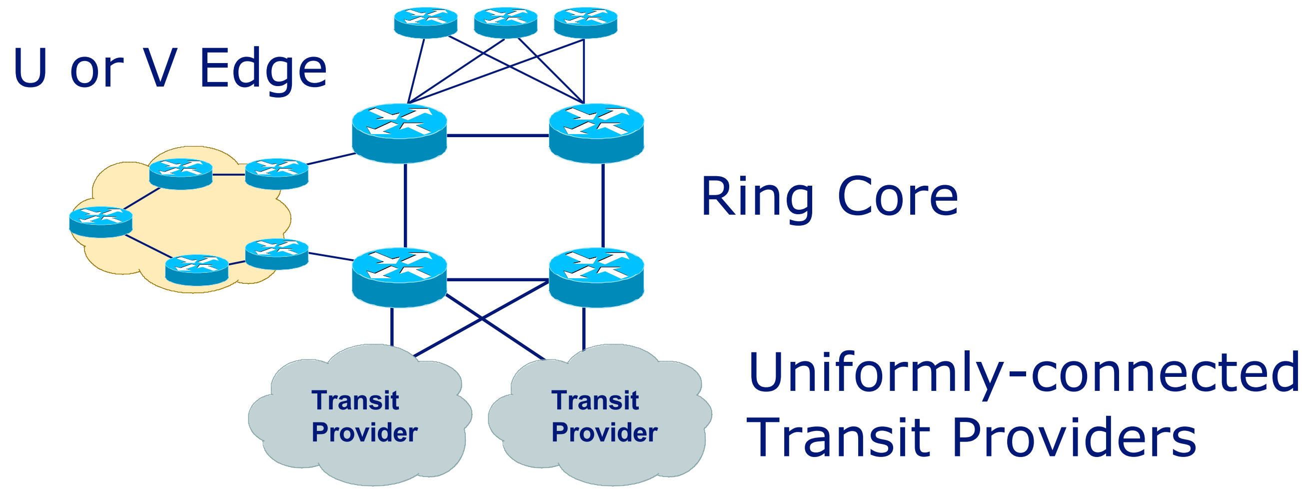


Demand end points may be in neighboring AS:
CR-to-Neighbor_AS, CR-to-CR, Neighbor_AS-to-Neighbor_AS.

Usage

- Interface Stats
  Often all that is needed for edge planning
- Individual Flows used in security analysis
- AS Paths
  Crucial for peering analysis and BGP-TE (balancing traffic on peering links)
- Traffic Matrices
  Crucial for core planning and TE (both IGP and BGP)
  - Per (VPN) Customer
    Used in troubleshooting, up-sell to customer
  - Per Class of Service
    Used in planning for products with different QoS guarantees
  - Per *application*
    Used in more enterprise like settings

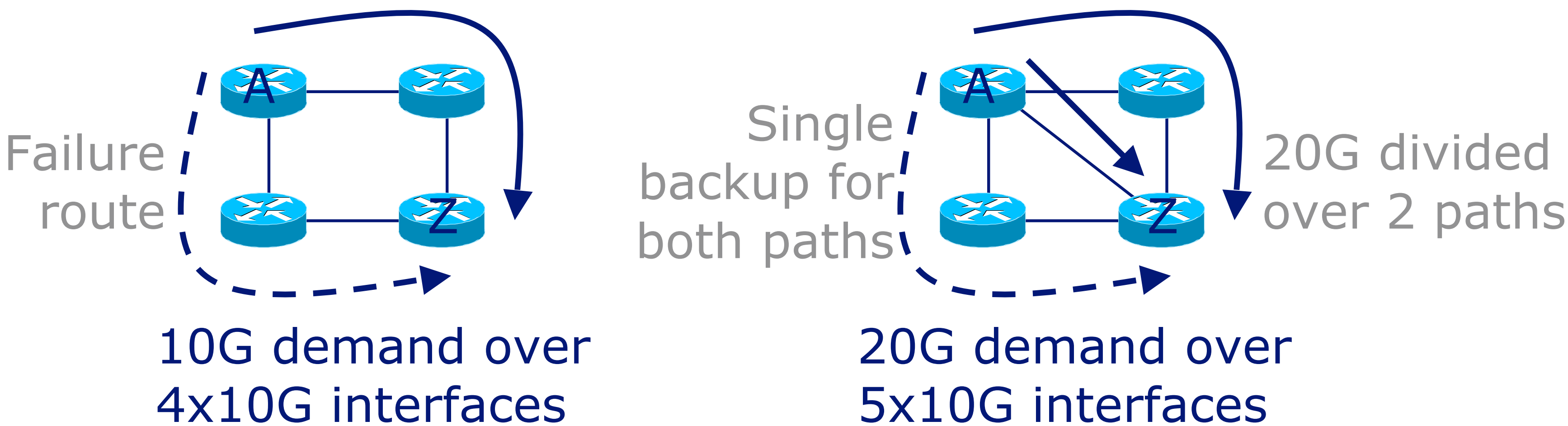
When Not Need Traffic Matrix



- Upgrade U when any link reaches 40% peak
- Upgrade V when sum of V utils reaches 80%
- Use MEDs, pre-pending, etc. to balance across transit links

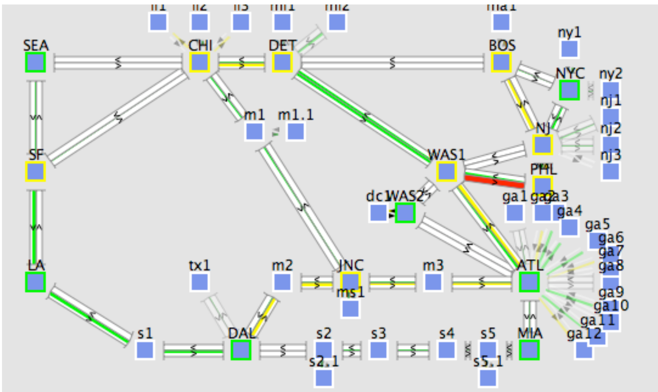
When Need Traffic Matrix

- Networks grow meshy over time
  - U's, V's and Rings (O's?) inefficient: 1:1 protection
  - Meshes allow traffic engineering and n:1 protection with n>1

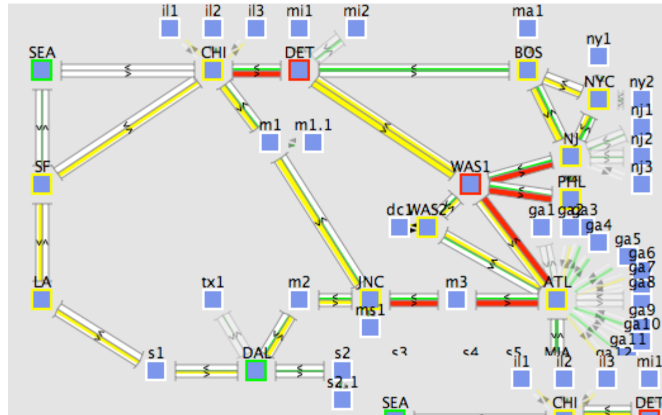


- Traffic matrix crucial for what-if analysis, planning and TE for meshy topologies

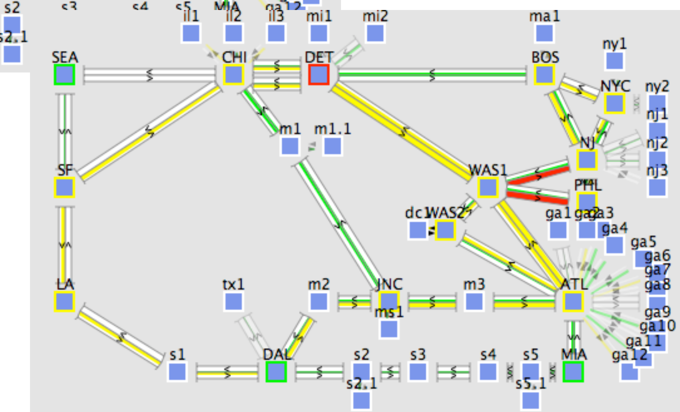
Using Traffic Matrices: Real Example*



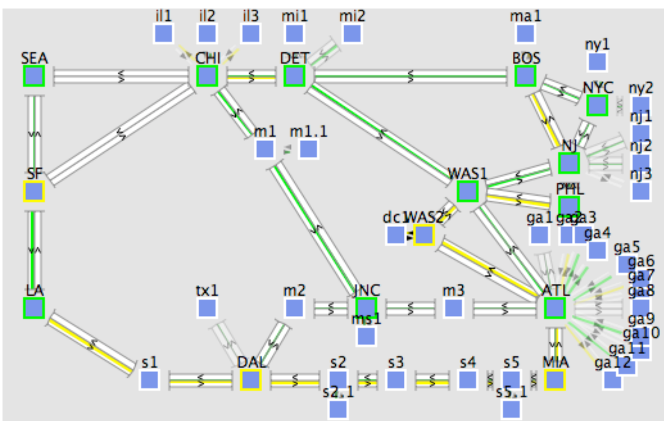
25 circuits to upgrade if use 50% rule (no TM).



8 circuits to upgrade based on failure analysis using TM.



3 circuits to upgrade if use TM for what-if analysis to determine to add bypass between CHI-DET.



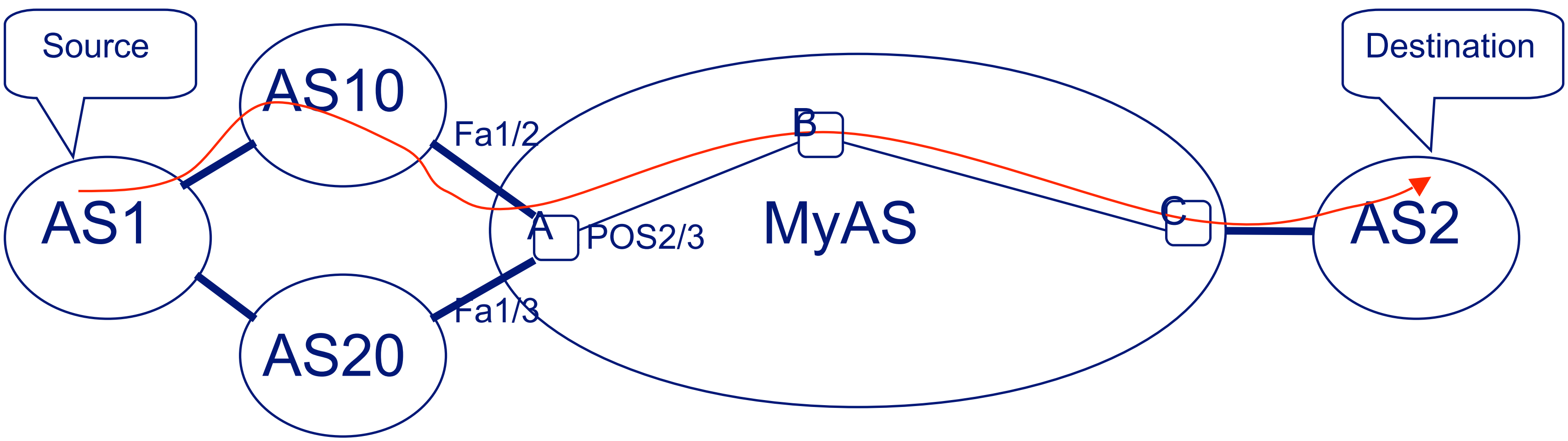
0 circuits to upgrade if use TM in offline metric-based TE for 3:1 protection.

*Details in Cariden App Note titled "Anon2 Case Study."

Measurement

Netflow Based Method(s)

Net & Flow



Source: 192.100.1.10
Destination: 192.200.2.20

TCP connection: src port 51228, dst port 80

Flow-cache

1. Create and update flows in NetFlow cache

Srclf	Srcl Padd	Dstlf	Dstl Padd	Protocol	TOS	Flgs	11000	00A2	Src Msk	Src AS	00A2	Dst Msk	Dst AS	Next Hop	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	10.023.2	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	10.023.2	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	10.023.2	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	10.023.2	1040	1745	14

2. 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

Srclf	Srcl Padd	Dstlf	Dstl Padd	Protocol	TOS	Flgs	11000	00A2	Src Msk	Src AS	00A2	Dst Msk	Dst AS	Next Hop	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	10.023.2	1528	1800	4

3. Aggregation

No

Yes

4. Export version
Non-Aggregated Flows—Export Version 5 or 9

5. Transport protocol

Export Packet | Payload (Flows)

e.g. Protocol-Port Aggregation Scheme Becomes

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

Aggregated Flows — Export Version 8 or 9

121919

Flow Export

- **On Router A**
- Flow expires after TCP FIN flag, and is exported
- Flow Cache Export data:
  - SrcIf: FA1/2
    SrcIP: 192.100.1.10
    DstIf: POS2/3
    DstIP: 192.200.2.20
    Protocol: 6
    Nexthop: 192.2.3.4 (router C)
    Start timestamp: 1210946008
    End timestamp: 1210946018
    byte count: 4500
    Src AS: 1
    Dst AS: 3
    [and more]

How to process NetFlow record

- Note:
  1. Src and Dst IP addresses are not in MyAS
  2. Ingress NetFlow accounting is needed to determine where a flow entered the network
     - Egress NetFlow would only tell you the incoming interface on the egress router, not where the flow entered your network
  3. Routers can be configured to export peer-as instead of origin-as, but this is only reliable for Dst peer-as
     - See diagram, MyAS *might* route flows towards AS1 via AS20, and hence identify AS20 as the Src peer-as for traffic *from* AS1.
  4. Use SrcIf to reliable determine neighbor/peer AS

Process NetFlow record

- For the Internal Traffic Matrix, determine ingress and egress router per flow
  - Ingress: easy (it is the exporting router)
  - Egress: lookup the nexthop field to determine which router it belong to.
  - Aggregate traffic per ingress/egress pair
    - Divide bytes by elapsed time
- External Traffic Matrix
  - Ingress: lookup Src AS for SrcIf (on exporting router)
    - Might not work in case of shared medium, e.g. IX.
    - Use Src peer-as (?)
  - Egress: lookup the nexthop field to determine which remote router it is connected to
    - In case of iBGP next-hop-self: use Dst peer-as

NetFlow

Useful:

http://www.cisco.com/en/US/docs/ios/solutions_docs/netflow/nfwhite.html

MPLS Based Methods

MPLS Based Methods

- Two methods to determine traffic matrices:
  - Using RSVP-TE tunnels
  - Using LDP statistics

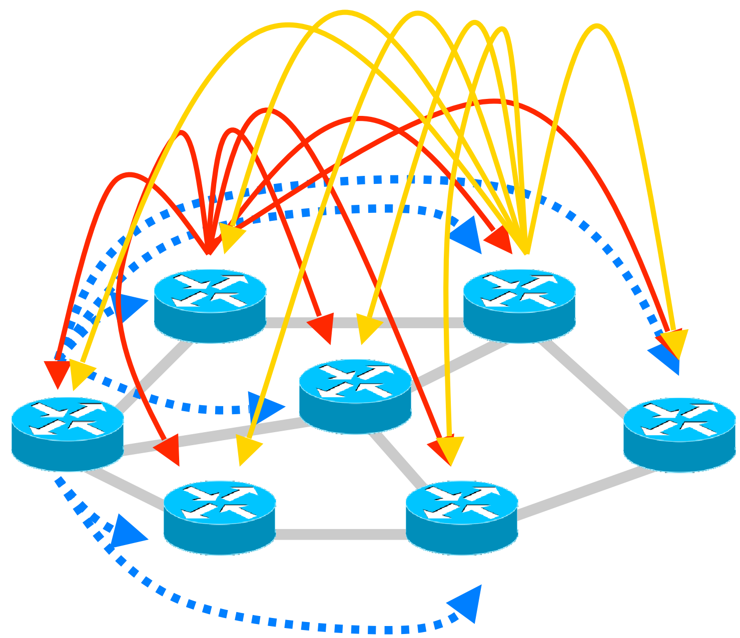
RSVP-TE in MPLS Networks

- RSVP-TE (RFC 3209) can be used to establish LSPs
- Example (IOS):

```
interface Tunnel99
 description RouterA => RouterB
 tag-switching ip
 tunnel destination 3.3.3.3
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng priority 5 5
 tunnel mpls traffic-eng bandwidth  1
 tunnel mpls traffic-eng path-option 3 explicit identifier 17
 tunnel mpls traffic-eng path-option 5 dynamic
!
ip explicit-path identifier 17 enable
 next-address 1.1.1.1
 next-address 2.2.2.2
 next-address 3.3.3.3
!
```

RSVP-TE in MPLS Networks

- Explicitly routed Label Switched Paths (TE-LSP) have associated byte counters
- A full mesh of TE-LSPs enables to measure the traffic matrix in MPLS networks directly



RSVP-TE in MPLS Networks

Pro’s and Con’s

- Advantages:
  - Method that comes closest a traffic matrix measurement
  - Easy to collect data
- Disadvantages:
  - A full mesh of TE-LSPs introduces an additional routing layer with significant operational costs;
  - Emulating ECMP load sharing with TE-LSPs is difficult and complex:
    - Define load-sharing LSPs explicitly;
    - End-to-end vs. local load-sharing;
  - Only provides Internal Traffic Matrix, no Router/PoP to peer traffic

RSVP-TE in MPLS Networks

SNMP Implementation

Juniper - All in one table

- MPLS-MIB mplsLspList (1.3.6.1.4.1.2636.3.2.3)
  - Relevant Objects:
    - mplsLspName
    - mplsLspTo
    - mplsLspState
    - mplsLspOctets
    - mplsLspAge

Other Useful objects can reveal information about LSP bandwidth, ERO, setup and hold priorities as well as affinities.

RSVP-TE in MPLS Networks

SNMP Implementation

Cisco - Must reference more than one table/MIB

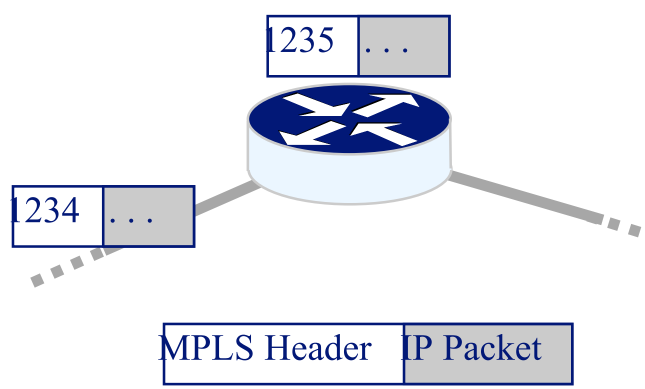
- MPLS-TE-MIB mplsTunnelTable (1.3.6.1.3.95.2.2)
  - Relevant Objects:
    - The table index (tunnelID.instanceID.srcAddr.dstAddr)
    - mplsTunnelName
    - mplsTunnelOperStatus
    - mplsTunnelIFIndex

Extra info available in MIB or crossreferenced to other tables via table pointers in MIB.

Use mplsTunnelIFIndex in IF-MIB or INTERFACES-MIB to look up the interface which should be measured for traffic.

Traffic matrices with LDP statistics

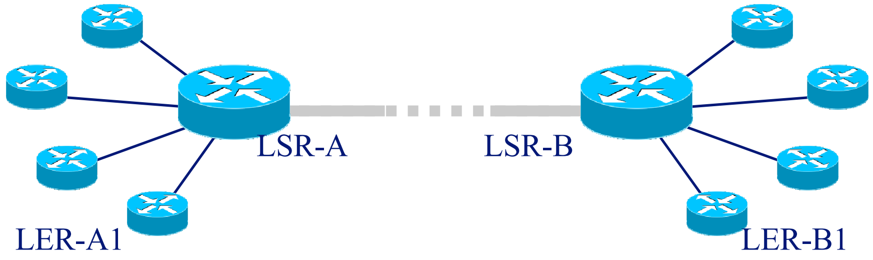
- In a MPLS network, LDP can be used to distribute label information
- Label-switching can be used without changing the routing scheme (e.g. IGP metrics)
- Many router operating systems provide statistical data about bytes switched in each *forwarding equivalence class* (FEC):



InLabel	OutLabel	Bytes	FEC	OutInt
1234	1235	**4124**	10.10.10.1/32	PO1/2
…	…	…		…

Practical Implementation
Cisco IOS

- LDP statistical data available through "show mpls forwarding" command
- Problem: Statistic contains no ingress traffic (only transit)
- If separate routers exist for LER- and LSR-functionality, a traffic matrix on the LSR level can be calculated
- A scaling process can be established to compensate a moderate number of combined LERs/LSRs.



Practical Implementation
Cisco IOS

LDP statistics on IOS:

```
router# show mpls forwarding-table
Local   Outgoing    Prefix             Bytes tag  Outgoing   Next Hop
tag     tag or VC   or Tunnel Id       switched   interface
26      26          62.225.24.184/30   0          PO10/0     point2point
        46          62.225.24.184/30   0          PO13/1     point2point
27      Pop tag     62.225.17.203/32   56529738   PO2/0      point2point
[..]
```

Martin Horneffer, NANOG33

Practical Implementation
Cisco IOS

SNMP Implementation

Limitation:
Only have measurements starting at the first hop.
No information about inbound interface.

Assumption:
All diverging LDP paths will converge

Practical Implementation
Cisco IOS

SNMP Implementation

3 tables in the MPLS-LSR-MIB (1.3.6.1.3.96.1)

- mplsInSegmentPerfTable
- mplsOutSegmentTable
- mplsXCTable : out of the OID index for this table, grab in segment ID, in lable and out segment ID

ip MIB's ipAddrTable for cataloguing all IP addresses on a router

Caveat:

This will grab ip addresses on interfaces not in the IGP, but this should still be OK since we are only doing this to identify which router an IP belongs to.

Practical Implementation
Cisco IOS

Process:

For every router:

1) gather the IP addresses of all its interfaces.
2) Get in segment, in label and out segment relationships from XCTable
3) get out segment and out label associations for all out segments
4) get IGP next hops for each outbound label.

Correlate the above… now for each path crossing this router, we know:

- inbound label
- outbound label
- IGP next hop (explicitly or implied by missing out segment)

Most entries will be transit hops along a path, however some are final hops.

- Easy to indentify since at end of path there is either no label, or label set to 3 (pop or PHP)

Practical Implementation
Cisco IOS

Process Continued:

Chain previously discovered path hops into paths.

For any given hop, we know:

inbound label, outbound label, and IP of nexthop (if applicable)

Find router which corresponds to IGP next hop.

Find path hop entry for that router that has inbound label same as this hop's outbound label, and so on.

Using recursion, do something like this:

```
Follow (List, thisHop, outLabel)
  if (thisHop.NextHop is 0.0.0.0) then return(thisHop); # End of Path
  else
    if (thisHop.NextHop is known)
       For each pathHop in List
          if (thisHop.NextHop is on pathHop and outLabel = pathHop.inLabel)
                EndHop = Follow(List, pathHop, pathHop.outLabel);
                return(EndHop);
       return(thisHop); # End of Path
```

Practical Implementation
Cisco IOS

Once complete we know:

- Inbound label L on router R_1 maps down a path leading to router R_n
- How to measure the amount of traffic received with inbound label L on R1

We can now easily:

- Measure the traffic R1 receives on each inbound label.
- Add up all the traffic for paths leading to Rn.

This tells us exactly how much traffic bound for Rn has flowed **through** R1.

This does not tell us how much traffic bound for Rn **originated** at R1.

- Must use math and/or interface measurements to arrive at solution.

Practical Implementation
Juniper JUNOS

- LDP statistical data available through “show ldp traffic-statistics” command
- Problem: Statistic is given only per FECs and not per outgoing interface
- As a result one cannot observe the branching ratios for a FEC that is split due to load-sharing (ECMP);
- Assume that traffic is split equally
- Especially for backbone networks with highly aggregated traffic this assumption is met quite accurately

Practical Implementation
Juniper JUNOS

```
user@router> show ldp traffic-statistics
FEC                  Type      Packets       Bytes  Shared
 62.225.16.134/32    Transit   1236933   103984630  No
                     Ingress         0           0  No
[..]
user@router> show route table inet.3
[..]
62.225.16.134/32    *[LDP/9] 06:08:27, metric 1
                     > via so-0/0/0.0, Push 39
[..]
```

Martin Horneffer, NANOG33

Practical Implementation
Juniper JUNOS

SNMP Implementation
(pure Juniper Environment/pre-8.1 JunOS)

JUNIPER-LDP-MIB

Only 1 table: jnxLdpStatsTable **(**1.3.6.1.4.1.2636.3.14.2)

Relevant objects (most IPv4 implementations):

- jnxLdpFec
- jnxLdpFecLength
- jnxLdpIngressOctets
- jnxLdpTransitOctets

Look for /32 FECs and find the associated routers.

Done.

Practical Implementation
Juniper JUNOS

SNMP Implementation

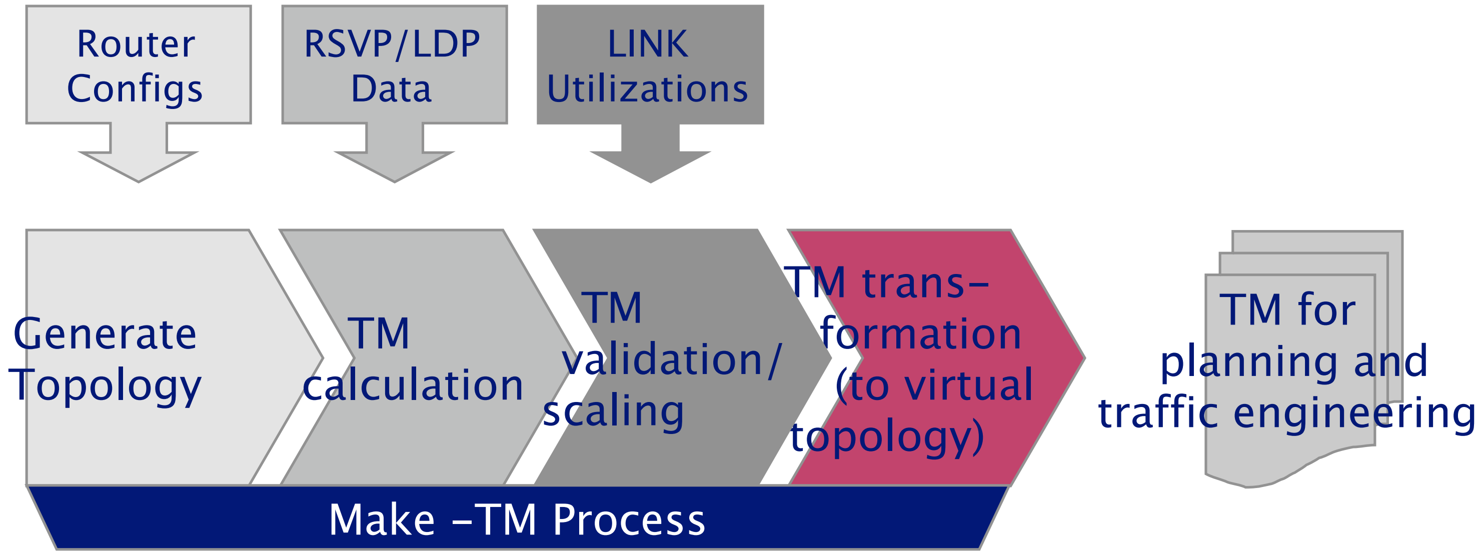
(mixed Cisco/Juniper environment or JunOS >= 8.1)

Approach very similar to Cisco-based approach, using Juniper MIBs:

JUNIPER-MPLS-MIB (1.3.6.1.4.1.2636.3.2)

JUNIPER-MPLS-LDP-MIB (1.3.6.1.4.1.2636.3.36)

Practical Implementation
Deployment Process



Conclusions for LDP method

- This method can be implemented in a multi-vendor network
- It does not require the definition of explicitly routed LSPs
- It allows for a continuous calculation
- There are some restrictions concerning
- vendor equipment
- network topology
- See Ref. [4]

Estimation based on Link Stats (e.g. Tomogravity)

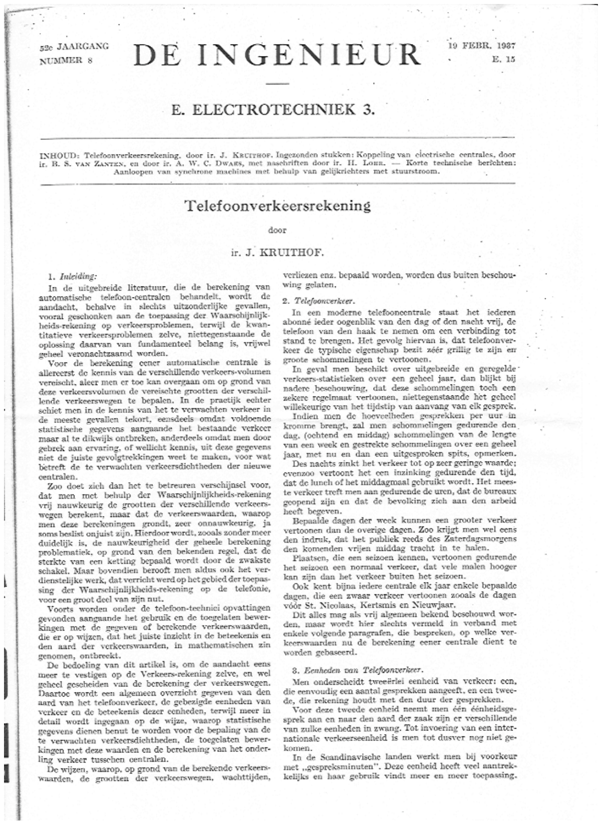
What do we want?

- Derive Traffic Matrix (TM) from easy to measure variables
  - No complex features to enable
- Link Utilization measurements
  - SNMP
  - easy to collect, e.g. MRTG
- Problem:

  Estimate point-to-point demands from measured link loads
- Network Tomography
  - Y. Vardi, 1996
  - Similar to: Seismology, MRI scan, etc.

Is this new?

- Not really…
- ir. J. Kruithof: *Telefoonverkeersrekening*, De Ingenieur, vol. 52, no. 8, feb. 1937 (!)



52e JAARGANG
NUMMER 8

DE INGENIEUR

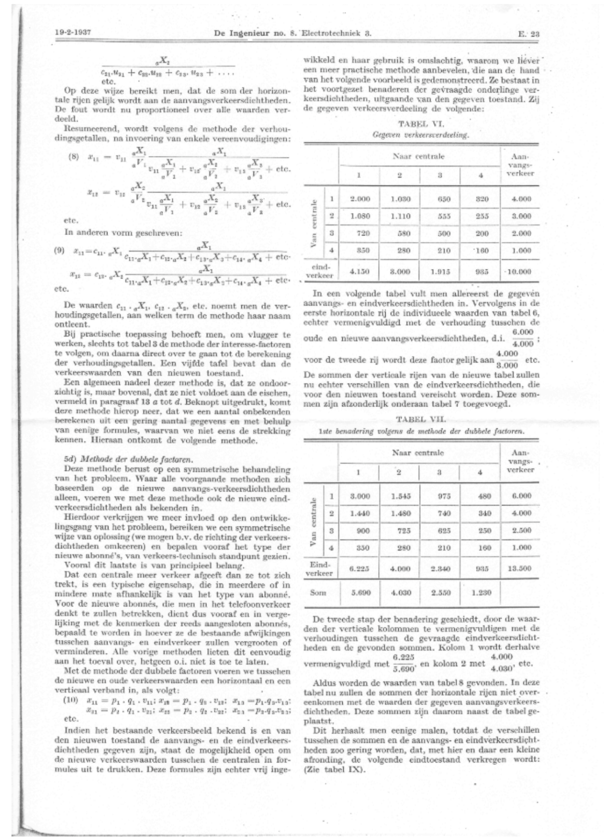
19 FEBR. 1937
E. 15

E. ELECTROTECHNIEK 3.

Telefoonverkeersrekening

door

ir. J. KRUITHOF.



19-2-1937 De Ingenieur no. 8. Electrotechniek 3. E. 23

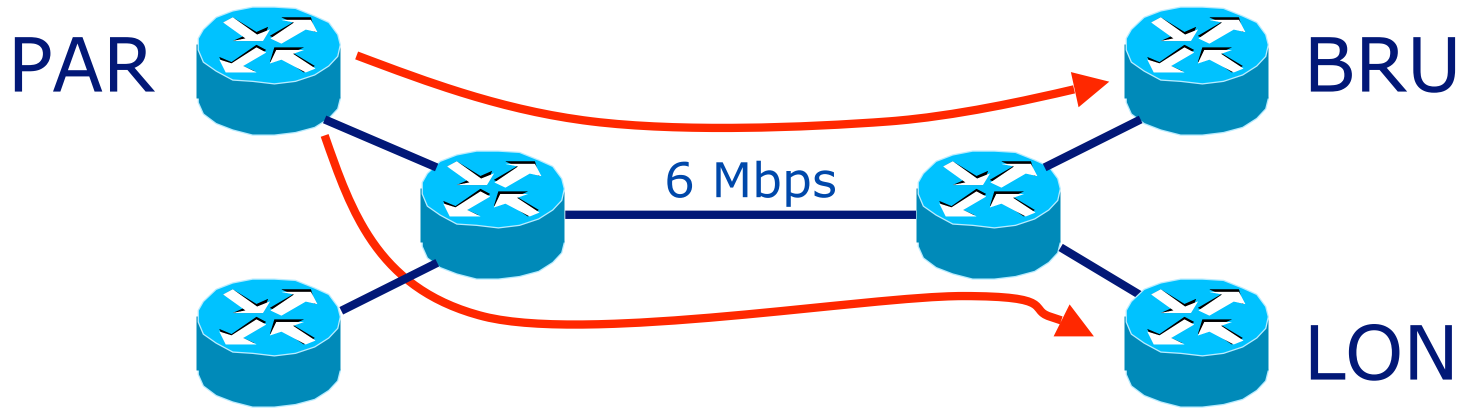
TABEL VI.

TABEL VII.

Demand Estimation

- Underdetermined system:
  - N nodes in the network
  - O(N) links utilizations *(known)*
  - O(N^2) demands *(unknown)*
  - Must add additional assumptions (information)
- Many algorithms exist:
  - *Gravity model*
  - *Iterative Proportional Fitting (Kruithof's Projection)*
  - *Maximum Likelihood Estimation*
  - *Entropy maximization*
  - *Bayesian statistics (model prior knowledge)*
  - *Etc...!*
- Calculate the **most likely** Traffic Matrix

Example

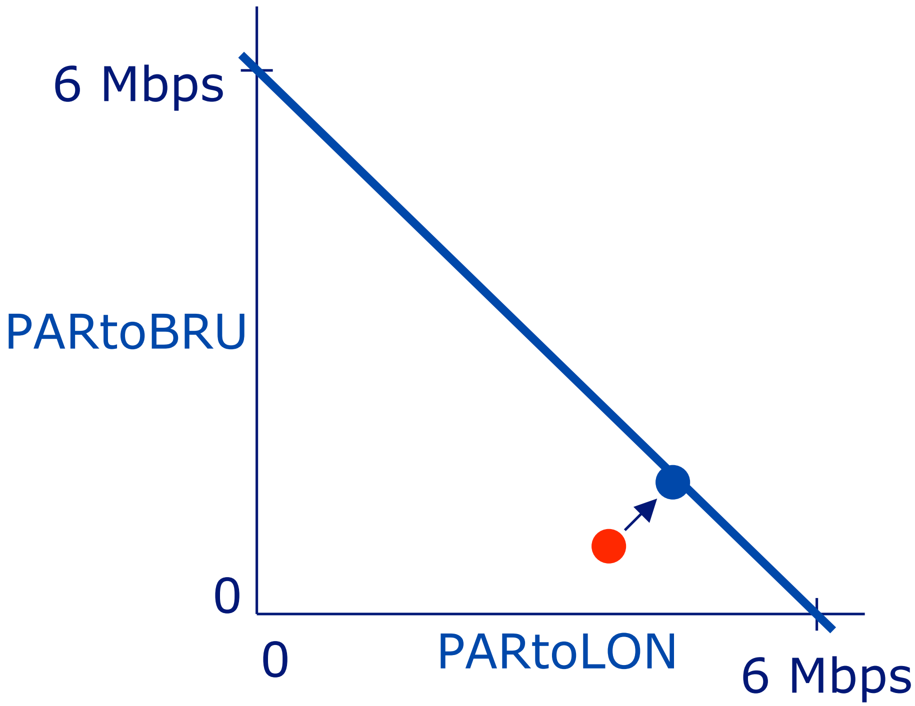


y: link utilizations
A: routing matrix
x: point-to-point demands

Solve: y = Ax
In this example: 6 = PARtoBRU + PARtoLON

Example

Solve: y = Ax -> 6 = PARtoBRU + PARtoLON



Additional information

E.g. Gravity Model (every source sends the same percentage as all other sources of it's total traffic to a certain destination)

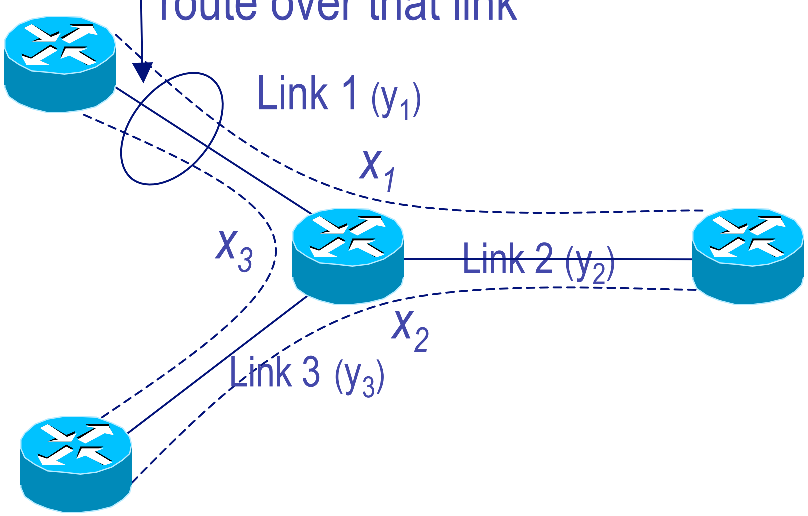
Example: Total traffic sourced at PAR is *50Mbps.* BRU sinks *2%* of total traffic, LON sinks *8%:*
PARtoBRU =1 Mbps and
PARtoLON =4 Mbps

Final Estimate: PARtoBRU = 1.5 Mbps and PARtoLON = 4.5 Mbps

General Formulation

$y_1 = x_1 + x_3$

Measured traffic, y, equals the sum of TM elements that route over that link



Interface Stats | Routing Matrix A | Traffic Matrix (as a vector)

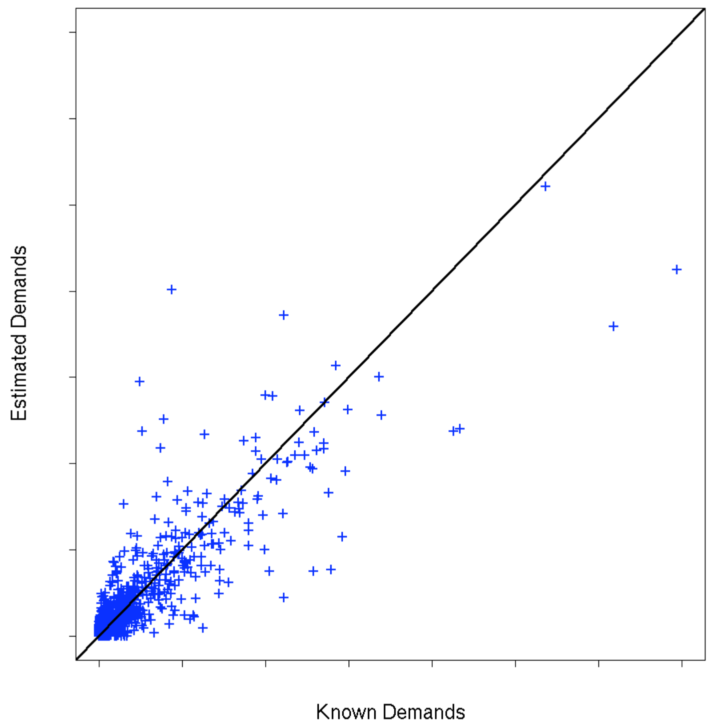
$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \varepsilon$$

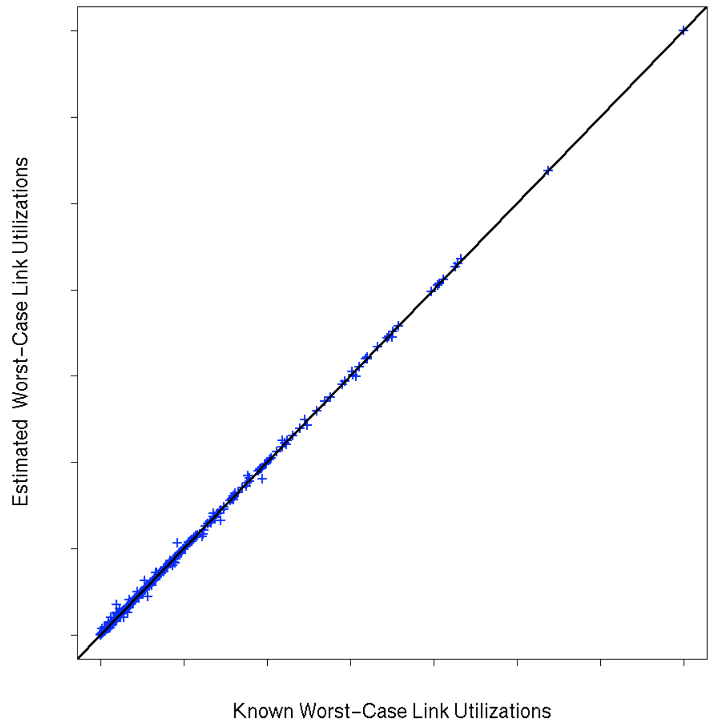
- Given Y and A solve for X (minimize ε)
- Many solutions to above
  - Pick some *likely* X
    (e.g., most conformant with gravity model)

* Zhang et al. (2004)

Estimation Results

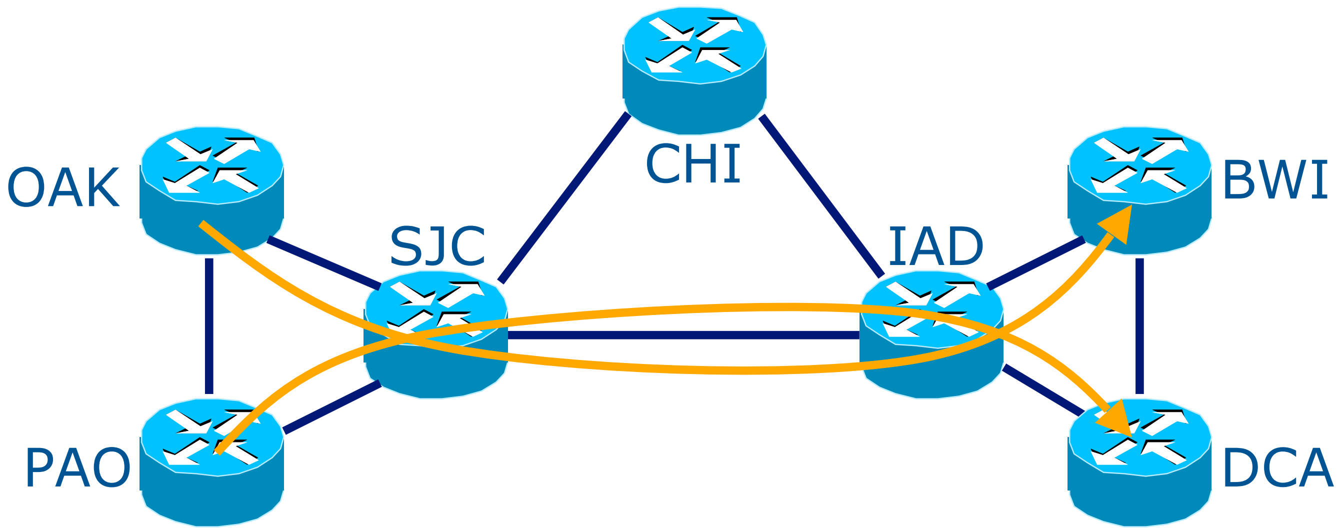
International Tier-1
IP Backbone





- Individual estimates are not accurate
- Results of using (the inaccurate) estimates in, failure analysis, for example, are accurate!

Estimation Paradox Explained



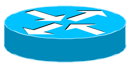
- Hard to tell apart elements
  (OAK->BWI, OAK->DCA, PAO->BWI, PAO->DCA, similar routings)
  are likely to shift as a group under failure or IP TE
  (e.g., above all shift together to route via CHI under SJC-IAD failure)

(Potential) Issues

NetFlow stats may not match link stats

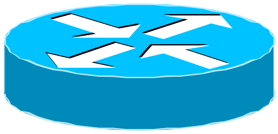
- NetFlow stats undercount but not consistently:-(

Router implementation matters!
Sampling is one cause but not always.



Typical AR

Interface	Traffic via SNMP(Mbps)	NetFlow/ SNMP
6	81	0.56
1	338	0.57
17	145	0.58
9	333	0.6
1	2210	0.61
33	4150	0.61
8	147	0.61
3	2290	0.62
32	1380	0.62
11	516	0.62
7	500	0.62
12	602	0.66
31	673	0.68



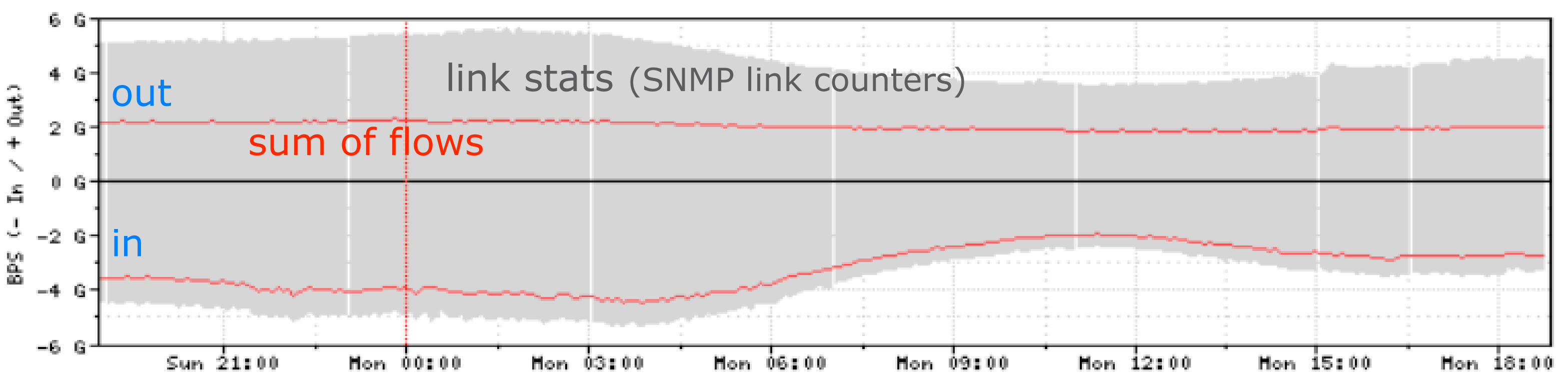
Top-of-the-line CR

Interface	Traffic via SNMP(Mbps)	NetFlow/ SNMP
45	1760	0.77
49	1730	0.78
58	1730	0.79
79	7750	0.8
74	7570	0.82
30	1350	0.85
34	7260	0.85
31	8840	0.86
61	7330	0.86
71	6310	0.86
39	1730	0.87
94	12710	0.87
98	12590	0.87
5	1760	0.88
27	11130	0.88
35	11130	0.88
36	5380	0.88
37	68	0.88
38	5320	0.89
39	71	0.89
8	1380	0.92
42	1370	0.93
57	1720	0.93
48	1720	0.94
44	1360	0.97
26	1730	0.98
29	0.396	13.31

Data from NetFlow tool in an operational ISP network

NetFlow Issues (2)

- Stats can clip at crucial times
  - NetFlow cache overflows at high traffic
  - CPU stops counting NetFlow when busy
- NetFlow and SNMP timescale mismatch
  - 10- or 15-minute typical (flows expire) vs. 2- or 5-minute SNMP link stats
- Poor implementations (e.g., bad outbound accounting)



MPLS Issues

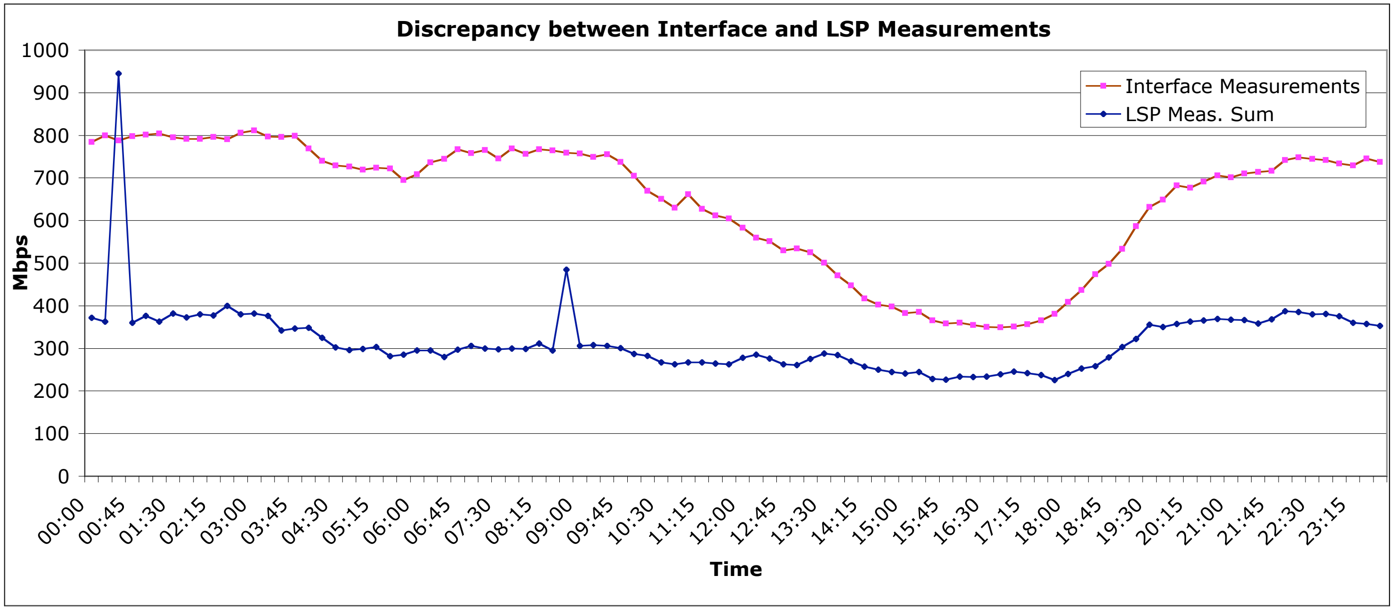
- MPLS LSPs (should be able to) provide internal traffic matrix directly
  - LDP: MPLS-LSR-MIB (or equivalent)
    - Mapping FEC to exit point of LDP cloud
    - Counters for packets that enter FEC (ingress)
    - Counters for packets switched per FEC (transit)
  - RSVP counters

- Does not provides external traffic matrix

LDP Issues

- Only transit statistics, no ingress statistic
  (on many versions of Cisco's IOS)
- Missing values
  (expected when making tens of thousands of measurements)
- Can take many minutes
  (important for tactical, quick response, TE)
- Not address external TM (of course)

RSVP Possible Issues



Data from operational network:
150 LSPs in one link

- Undercount link stats
- Not track well
- Volatile

- Also
  - Problematic counters:
    reset on path reroute on many Junos implementations
    missing all together on many Alcatel Lucent SR platforms
  - Issues with $O(N^2)$: missing values, time, ...

LSP Stats Summary

- LSP stats good enough when:
  - Only need internal traffic matrix
  - Have full mesh of LSPs
  - Not getting bitten by various platform issues
  - Long-term analysis (not quick enough for tactical Ops)

- Otherwise, if use LSP stats, need to watch out for
  - missing
  - unreliable
  - unavailable
  - inconsistent
  - slow-to-gather data

Estimation Issues

- Needs human guidance to set up
  - e.g., mesh of demands between voice routers but no traffic between VPN and voice routers
- Not fit for fine-grained traffic engineering
- Presents a leap of faith
  - Takes time for people to trust it

Regressed Measurements

Regressed Measurements Overview

- Use interface stats as gold standard
  - Traffic management policies, almost always, based on interface stats (e.g.,
    ops alarm if 5-min average utilization goes >90%
    traffic engineering considered if any link util approach 80%
    cap planning guideline is to not have link util above 90% under any single failure)

- Mold NetFlow, LSP stats, ... to match interface stats

LSP Example for Regression

- Builds on estimation. Each LSP/NetFlow/... measurement adds a row to the Y=AX
  - RSVP measurement for OAK->BWI
    $Y_{RSVP\text{-}OAK\text{->}BWI} = X_{OAK\text{->}BWI}$

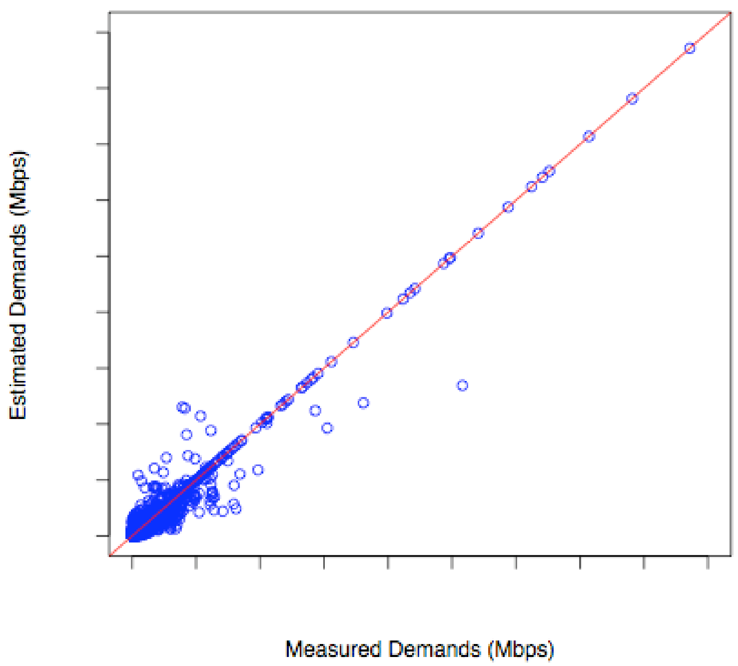
  - Transit LDP measurement for SJC->BWI:
    $Y_{Transit\text{-}SJC\text{->}DCA} = X_{OAK\text{->}DCA} + X_{PAO\text{->}DCA}$

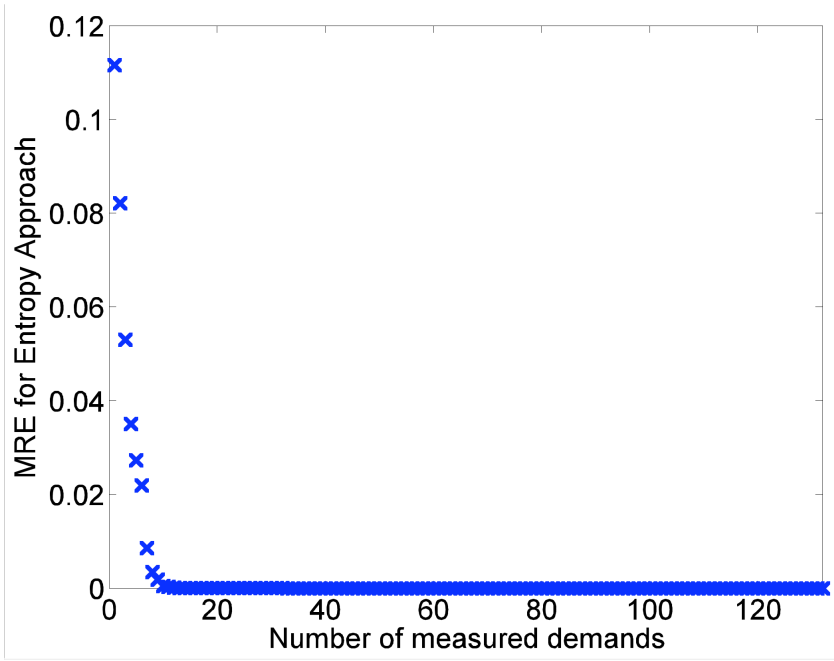
- Solve for X such that there is strict conformance with link stat Y values with other measurements matched as best possible.

Role of Netflow, LSP Stats,...

- Can improve TM estimates with just a few measurements

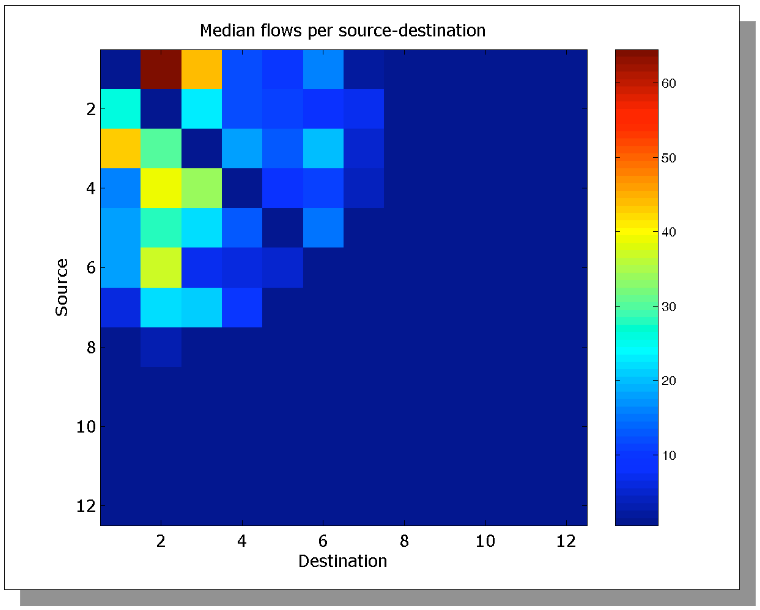
Add 160 measurements from 10 routers



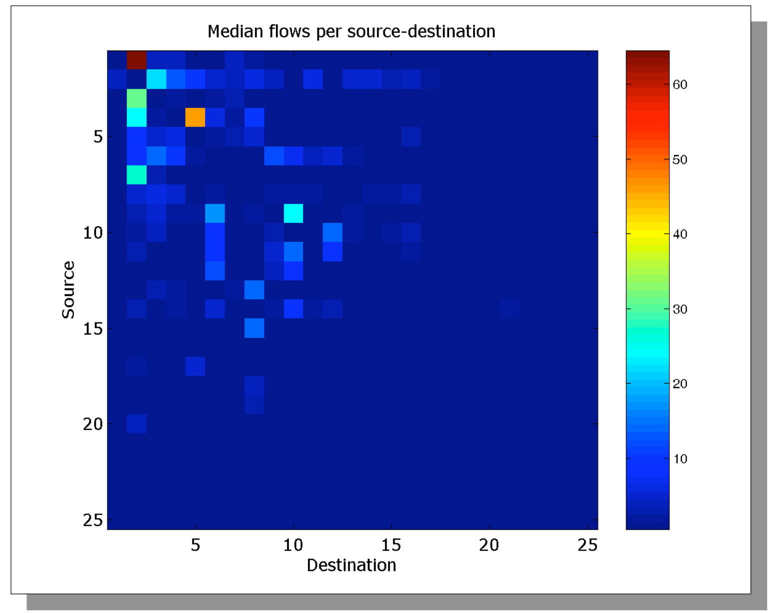


Spatial demand distributions

European subnetwork



American subnetwork



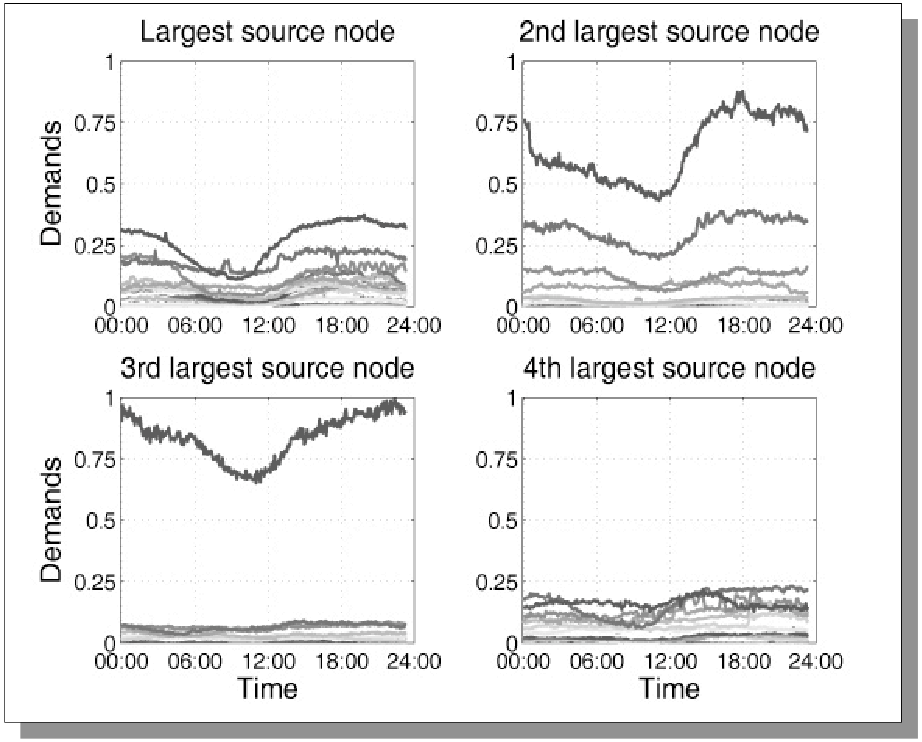
Few large nodes contribute to total traffic (20% demands – 80% of total traffic)

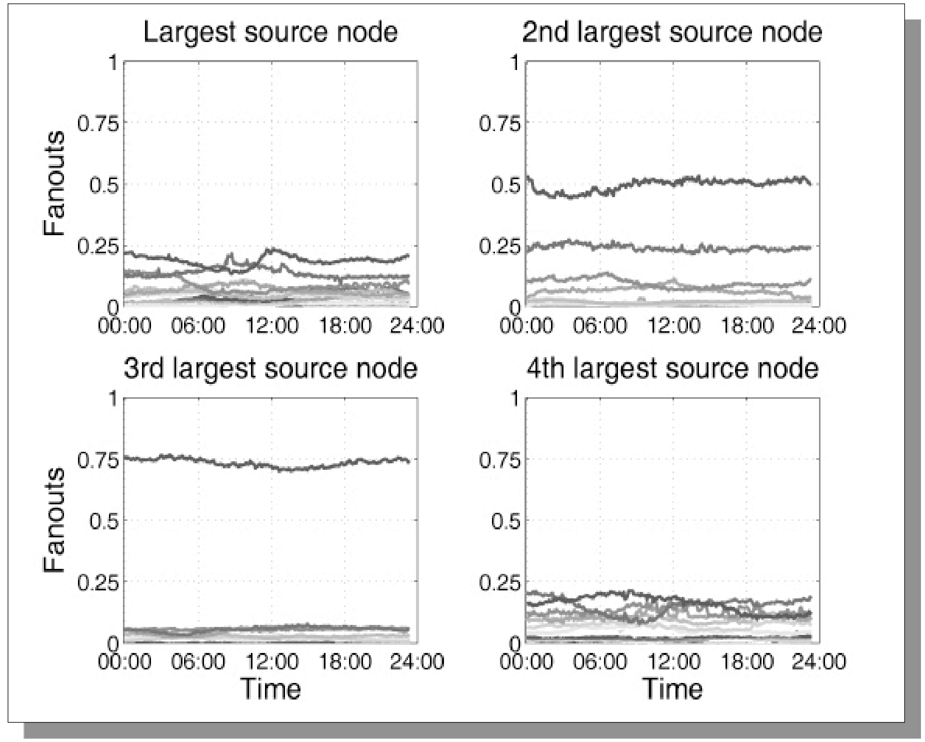
Demand Ratios (Fanouts) Are Stable

Fanout: relative amount of traffic (as percentage of total)

Demands for 4 largest nodes, USA

Corresponding fanout factors





Can use demand ratios from NetFlow or LSPs even if absolute amounts are not accurate or are outdated.

Regressed Measurements with LDP

- Topology discovery done in real-time
- LDP measurements rolling every 30 minutes
- Interface measurement every 2 minutes
- Regression* combines the above information
- Robust TM estimate available every 5 minutes
- (See the DT LDP estimation for another approach for LDP**)

*Cariden's Demand Deduction™ in this case(http://www.cariden.com)
** Schnitter and Horneffer (2004)

Regressed Measurement and NetFlow

- NetFlow can sample less frequently
  - Regressed Measurement uses the demand ratios. OK if absolute numbers not right. They will get adjusted.

- Need to process less frequently
  Missing data less important
  - Can combine hours-old Netflow data with with minute-by-minute link stats

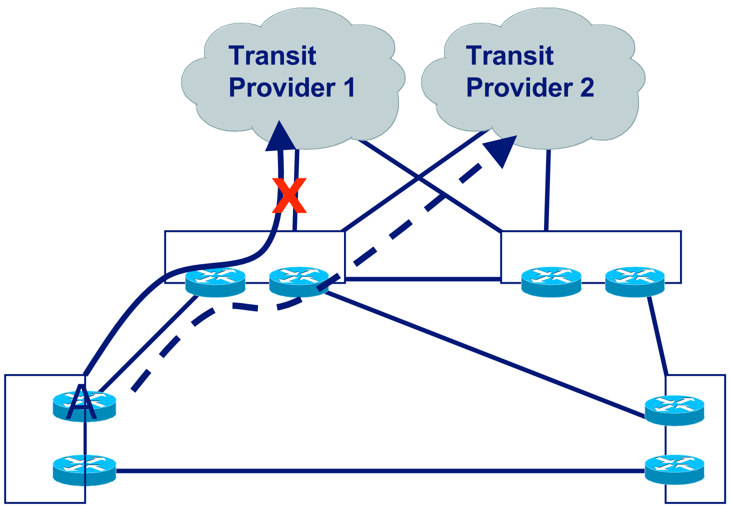
- Can use partial NetFlow Coverage
  - Recall "Few large nodes contribute to total traffic (20% demands – 80% of total traffic)"

Regressed Measurements Summary

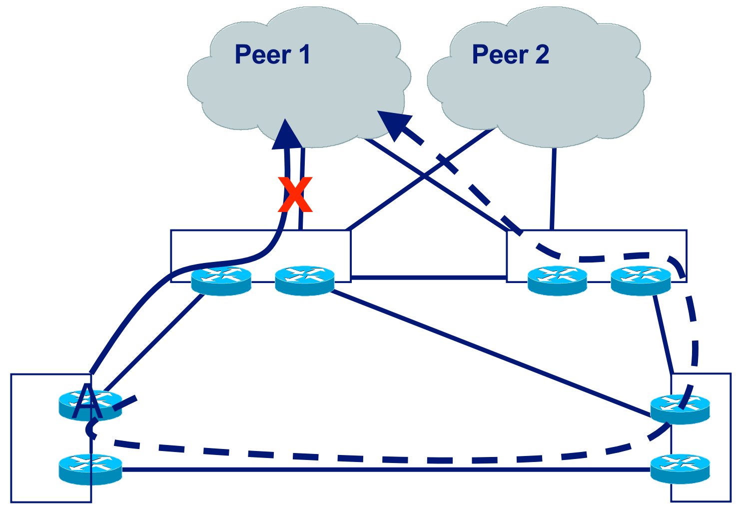
- Interface counters remain the most reliable and relevant statistics
- Collect LSP, Netflow, etc. stats as convenient
  - Can afford partial coverage (e.g., one or two big PoPs)
  - more sparse sampling (1:10000 or 1:50000 instead of 1:500 or 1:1000)
  - less frequent measurements (hourly instead of by the minute)
- Use regression (or similar method) to find TM that conforms primarily to interface stats but is guided by NetFlow, LSP stats

Notes

Internal or External TM?



Internal TM tends to be stable with transit traffic.



External TM tends to be stable for peering traffic.
(See Cariden Peering Planning studies presented at APRICOT and RIPE: leakage around 16%.)

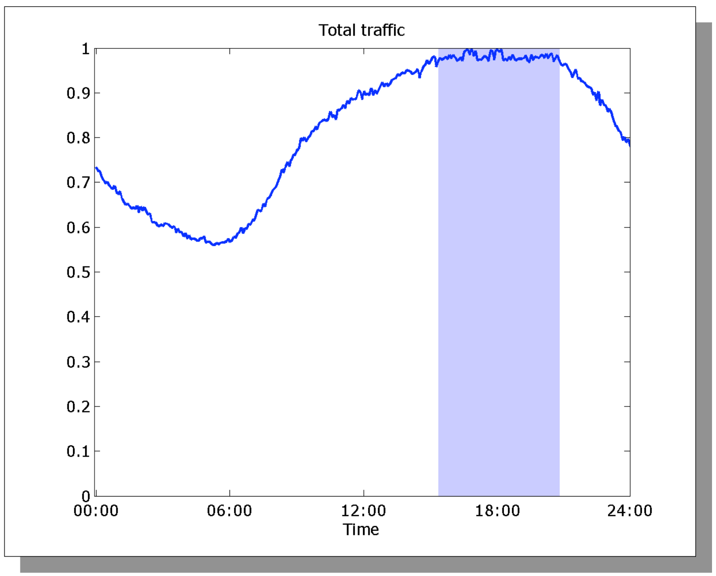
These are just guidelines. We have Tier-1 network models based on Internal TM because the shift in internal traffic matrix is not seen to be significant. (see Sprint paper for opposite case).

Peak Across Time or Peak Time?

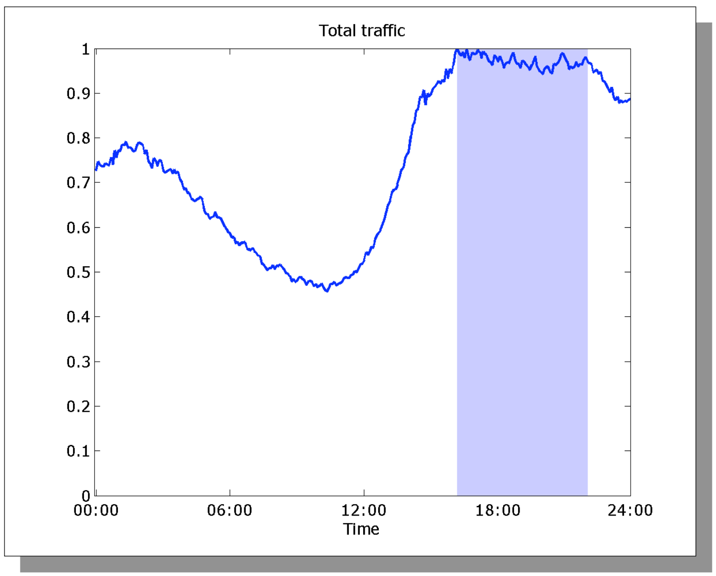
- Planning with link stats often uses P95 across week or month
- Planning with TM does better with one or two peak times.

Example of picking a peak time for a multi-continent network.

European subnetwork



American subnetwork

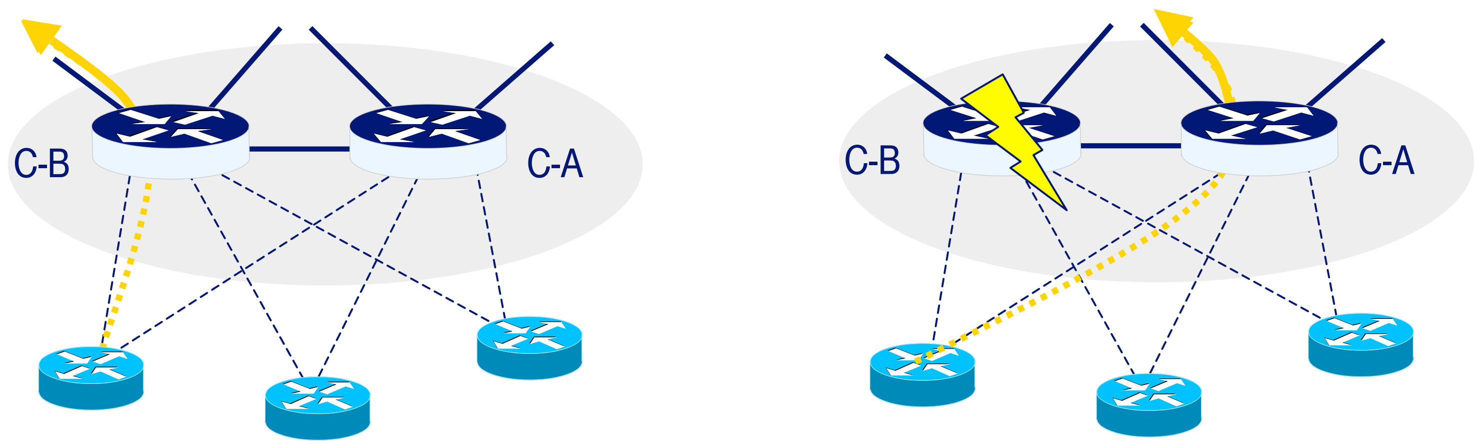


Total traffic very stable over 3-hour busy period

Traffic Matrices for Partial Topologies

Traffic Matrices in Partial Topologies

- In larger networks, it is often important to have a TM for a partial topology (not based on every router)
- Example: TM for core network (planning and TE)
- Problem: TM changes in failure simulations
- Demand moves to another router since actual demand starts outside the considered topology (red):

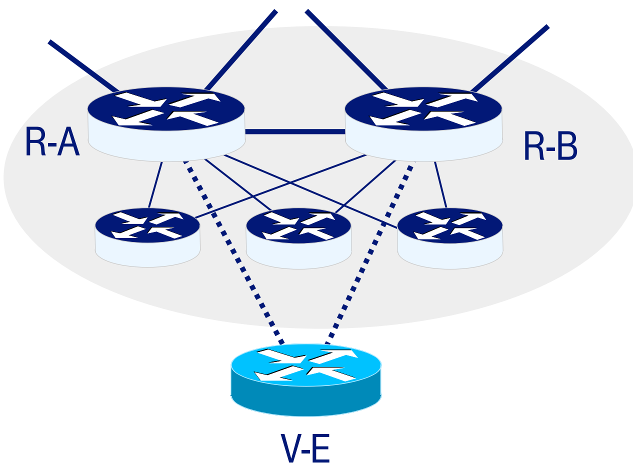


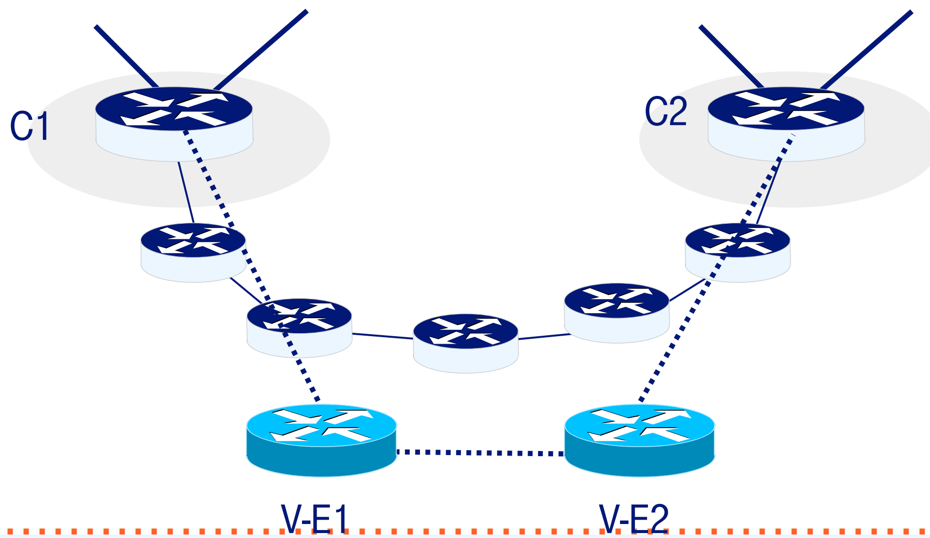
Traffic Matrices in Partial Topologies

- The same problem arises with link failures
- Results in inaccurate failure simulations on the reduced topology
- Metric changes can introduce demand shifts in partial topologies, too.
- But accurate (failure) simulations are essential for planning and traffic engineering tasks

Traffic Matrices in Partial Topologies

- Introduce virtual edge devices as new start-/endpoints for demands
- Map real demands to virtual edge devices
- Model depends on real topology
- Tradeoff between simulation accuracy and problem size.





Summary & Conclusions

Recommendations

- Divide and Conquer
    - Use interface stats for "edge" U, V topologies
    - Use TM in Core
    - Use ASPath for BGP TE

- Start Simple
    - Internal TM if have RSVP, LDP, or NetFlow with NextHopSelf on
    - Estimation if only link stats available

- Monitor Model Goodness (see how well model predicts realities after failures or network changes)
- Add Info/Procedures as Necessary
    - NetFlow (partial) etc. and Regressed Measurements.

References

1. A. Gunnar, M. Johansson, and T. Telkamp, "Traffic Matrix Estimation on a Large IP Backbone - A Comparison on Real Data", *Internet Measurement Conference 2004.* Taormina, Italy, October 2004.
2. Yin Zhang, Matthew Roughan, Albert Greenberg, David Donoho, Nick Duffield, Carsten Lund, Quynh Nguyen, and David Donoho, "How to Compute Accurate Traffic Matrices for Your Network in Seconds", NANOG29, Chicago, October 2004.
3. AT&T Tomogravity page:
   http://public.research.att.com/viewProject.cfm?prjID=133/
4. S. Schnitter, T-Systems; M. Horneffer, T-Com. "Traffic Matrices for MPLS Networks with LDP Traffic Statistics." Proc. Networks 2004, VDE-Verlag 2004.
5. Y. Vardi. "Network Tomography: Estimating Source-Destination Traffic Intensities from Link Data." J.of the American Statistical Association, pages 365–377, 1996.
6. "Simon's SNMP Hacks":
   http://www.switch.ch/misc/leinen/snmp/
7. Van Jacobson, Haobo Yu, Bruce Mah, "Route-Flow Fusion: Making traffic measurement useful." NANOG 35, October 2005.
8. T. Telkamp, "Peering Planning Cooperation without Revealing Confidential Information." RIPE 52, Istanbul, Turkey
9. GRNET Multicast Weathermap: http://netmon.grnet.gr/multicast-map.shtml