Constraint Based Resilience Analysis

Helmut Simonis
helmut.simonis@crosscoreop.com

CrossCore Optimization Ltd

Comments

- Too boring!

Comments

- Too boring!
- Is this not too specific for CP?

Comments

- Too boring!
- Is this not too specific for CP?
- Why is that an important problem?

Comments

- Too boring!
- Is this not too specific for CP?
- Why is that an important problem?
- Try something more interesting...

Comments

- Too boring!
- Is this not too specific for CP?
- Why is that an important problem?
- Try something more interesting...
- Keep it high-level

Constraint Applications in Networks

Commercial Break

Handbook of Constraint Programming, chapter 25

Overview

- The Problem

Overview

- The Problem
- Reminder: IP Networks

Overview

- The Problem
- Reminder: IP Networks
- The Applications
    - Traffic Placement
    - Secondary Paths
    - Bandwidth on Demand
    - Traffic Diversion
    - Metric Optimization
    - Network Design
    - Resilience Analysis

Overview

- The Problem
- Reminder: IP Networks
- The Applications
  - Traffic Placement
  - Secondary Paths
  - Bandwidth on Demand
  - Traffic Diversion
  - Metric Optimization
  - Network Design
  - Resilience Analysis
- The Tool: Core:Insight

The Problem

- How to provide cost effective, high quality services running an IP network?

The Problem

- How to provide cost effective, high quality services running an IP network?
- Easy to build high quality network by massive over-provisioning

The Problem

- How to provide cost effective, high quality services running an IP network?
- Easy to build high quality network by massive over-provisioning
- Easy to build consumer grade network disregarding Quality of Service (QoS)

The Problem

- How to provide cost effective, high quality services running an IP network?
- Easy to build high quality network by massive over-provisioning
- Easy to build consumer grade network disregarding Quality of Service (QoS)
- Very hard to right-size a network, providing just enough capacity

Reminder: IP Networks

- Packet forwarding
  - Connection-less

Reminder: IP Networks

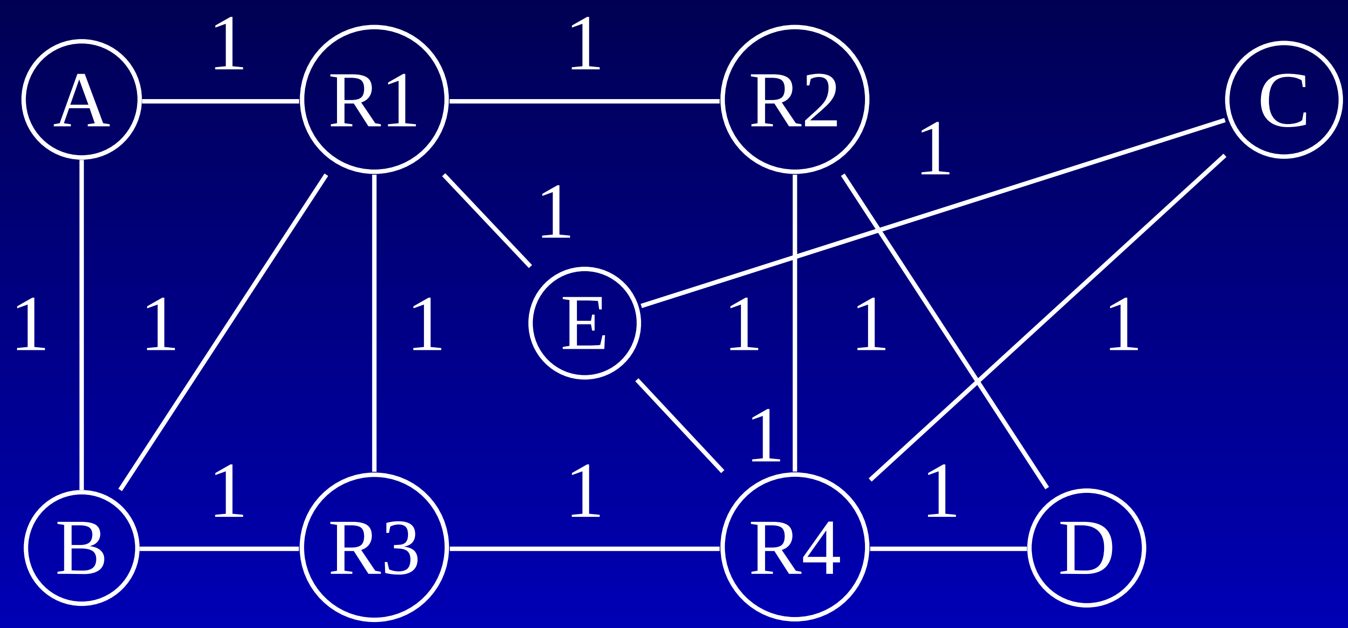
- Packet forwarding
  - Connection-less
- Destination based routing
  - Distributed routing algorithm based on shortest path algorithm
  - Routing metric determines preferred path

Reminder: IP Networks

- Packet forwarding
  - Connection-less
- Destination based routing
  - Distributed routing algorithm based on shortest path algorithm
  - Routing metric determines preferred path
- Best effort
  - Packets are dropped when there is too much traffic on interface
  - Guaranteed delivery handled at other layers (TCP/applications)

Example

Example network with uniform metric (1) and bandwidth (100), nine routers



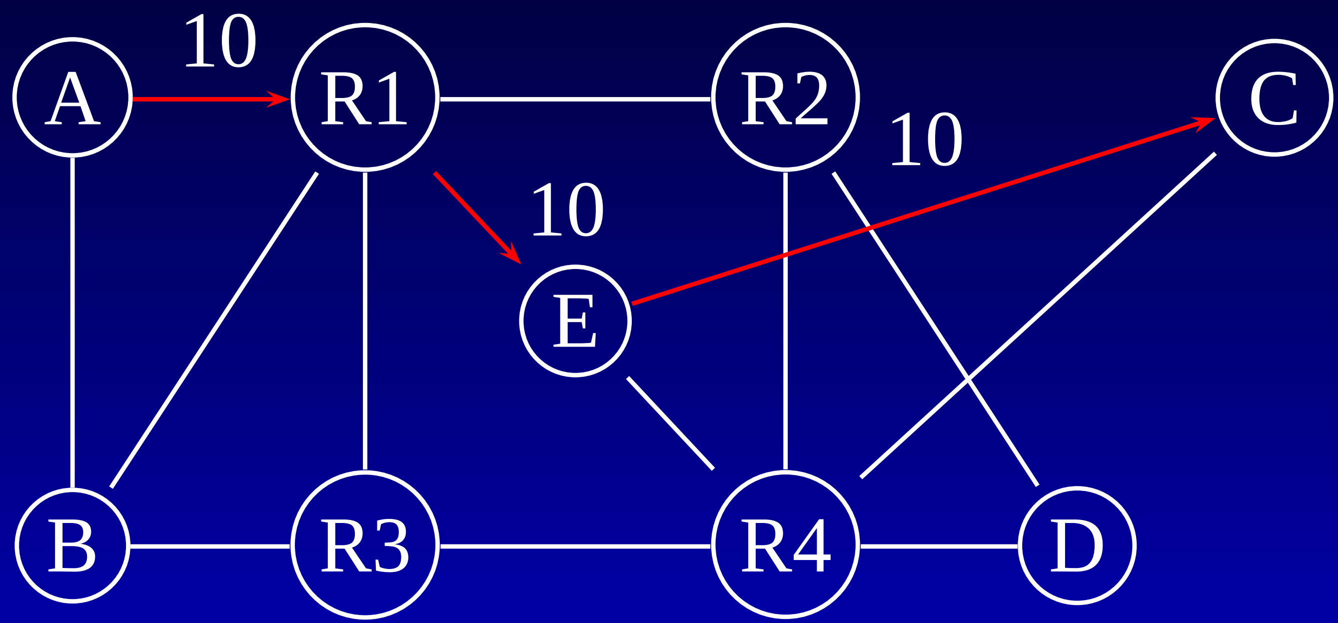
Traffic Matrix

For demonstration purposes only partially filled

	A	B	C	D	E
A	0	0	10	20	20
B	0	0	10	20	20
C	0	0	0	0	0
D	0	0	0	0	0
E	0	0	0	0	0

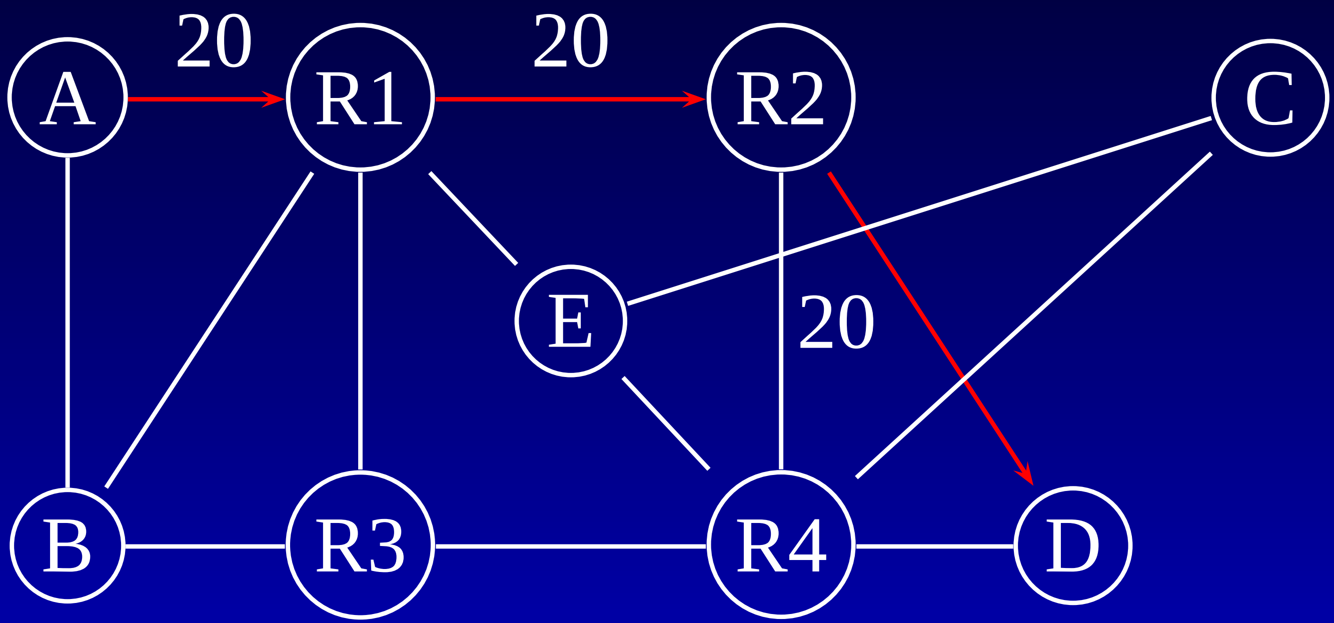
Routing

Demand AC 10



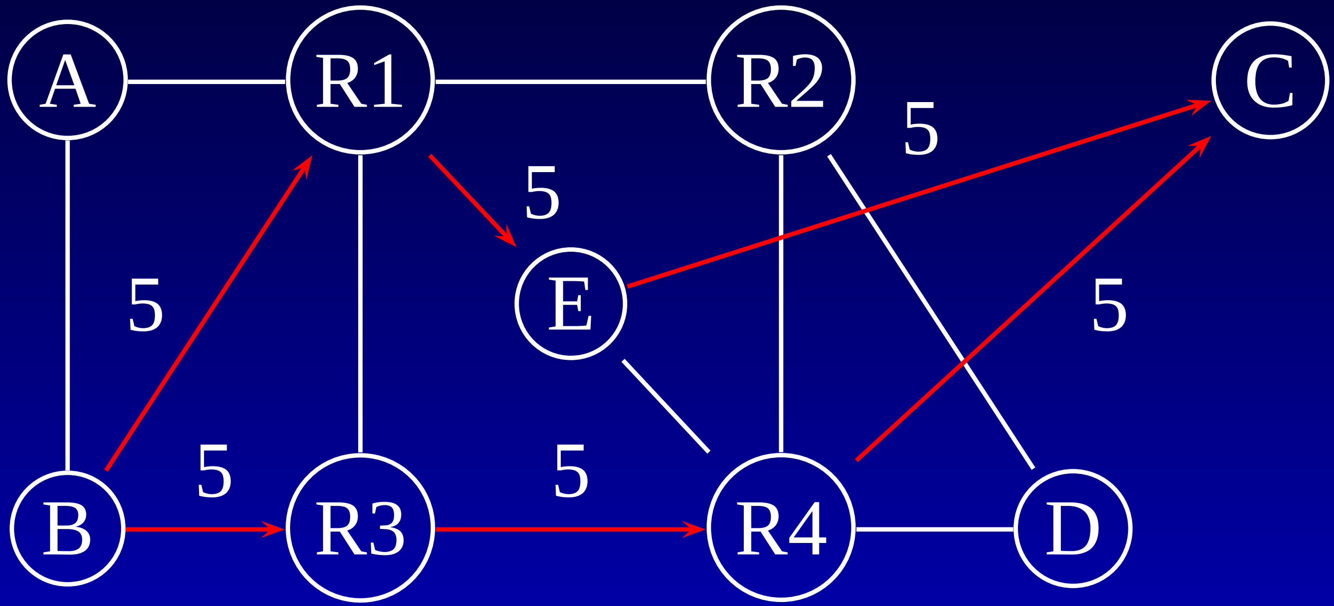
Routing

Demand AD 20



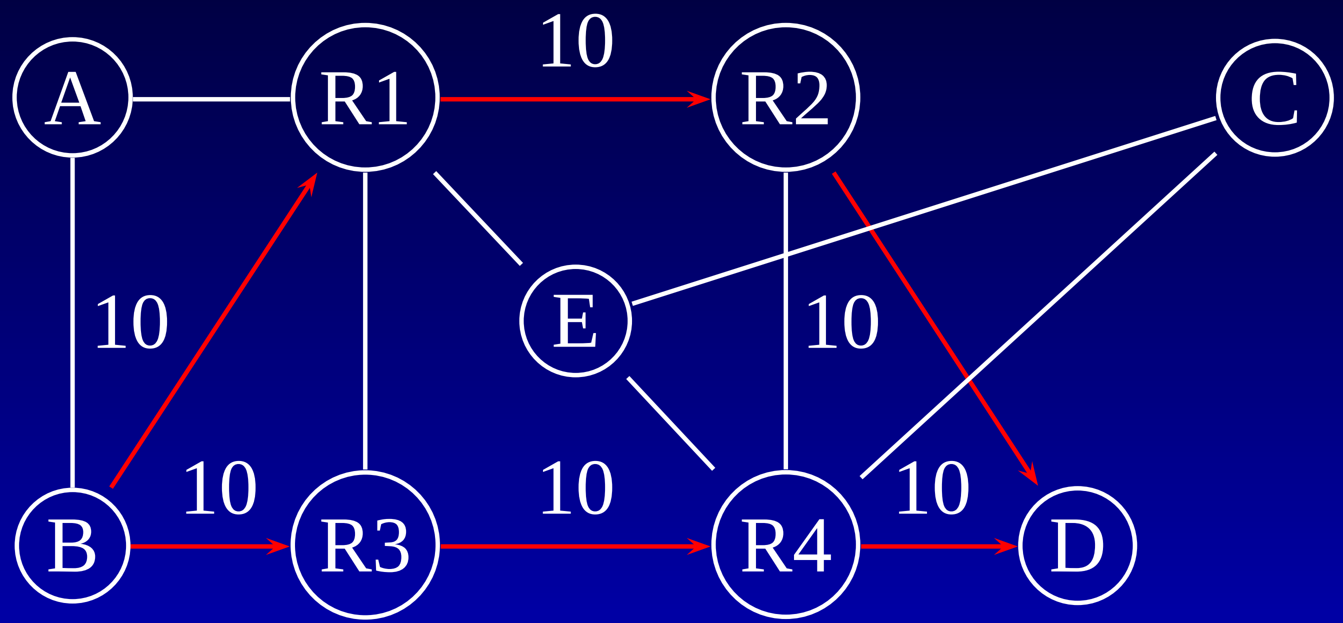
Routing

Demand BC 10



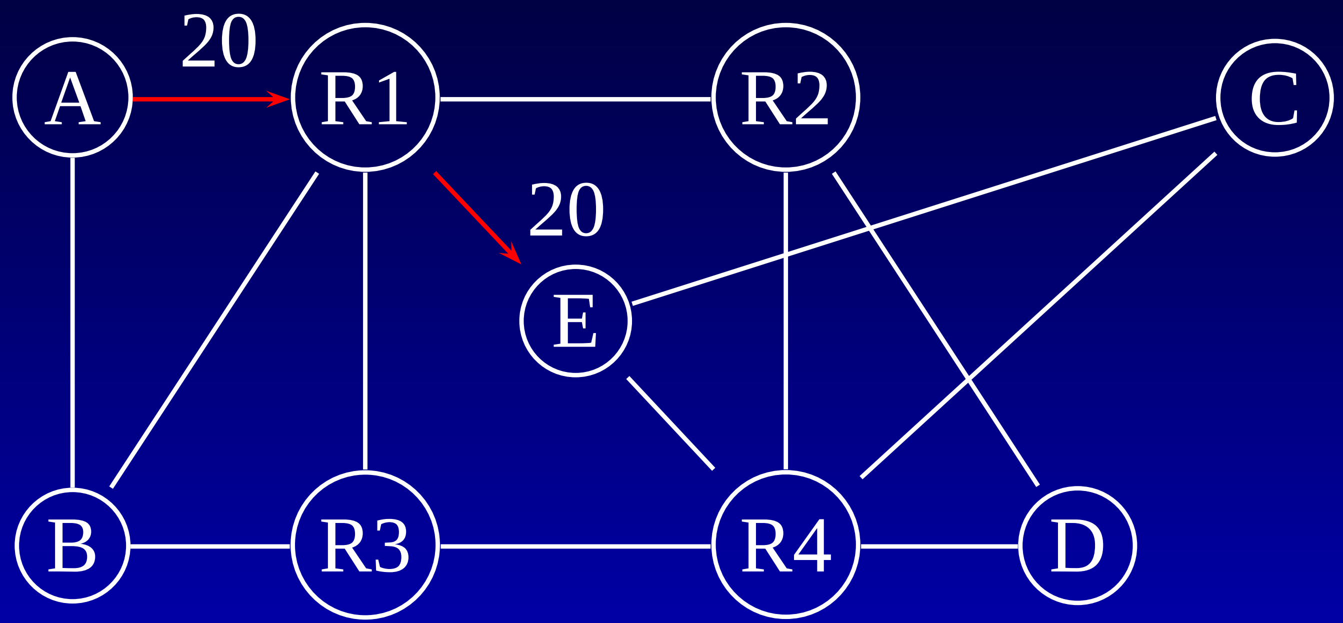
Routing

Demand BD 20



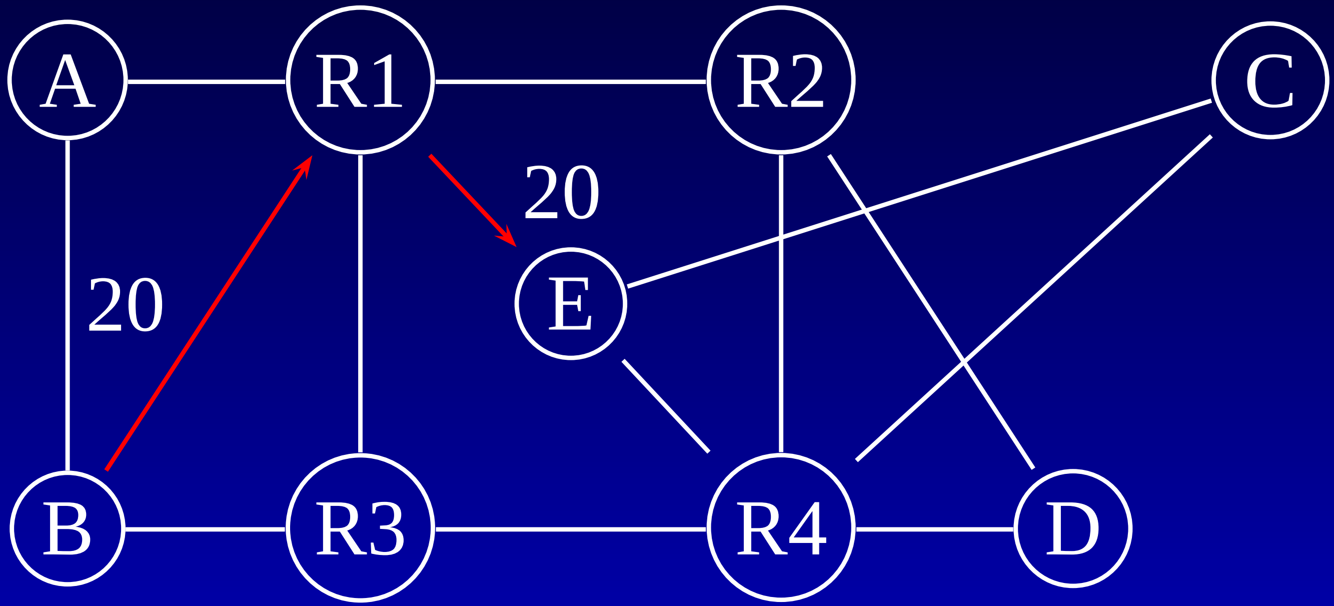
Routing

Demand AE 20



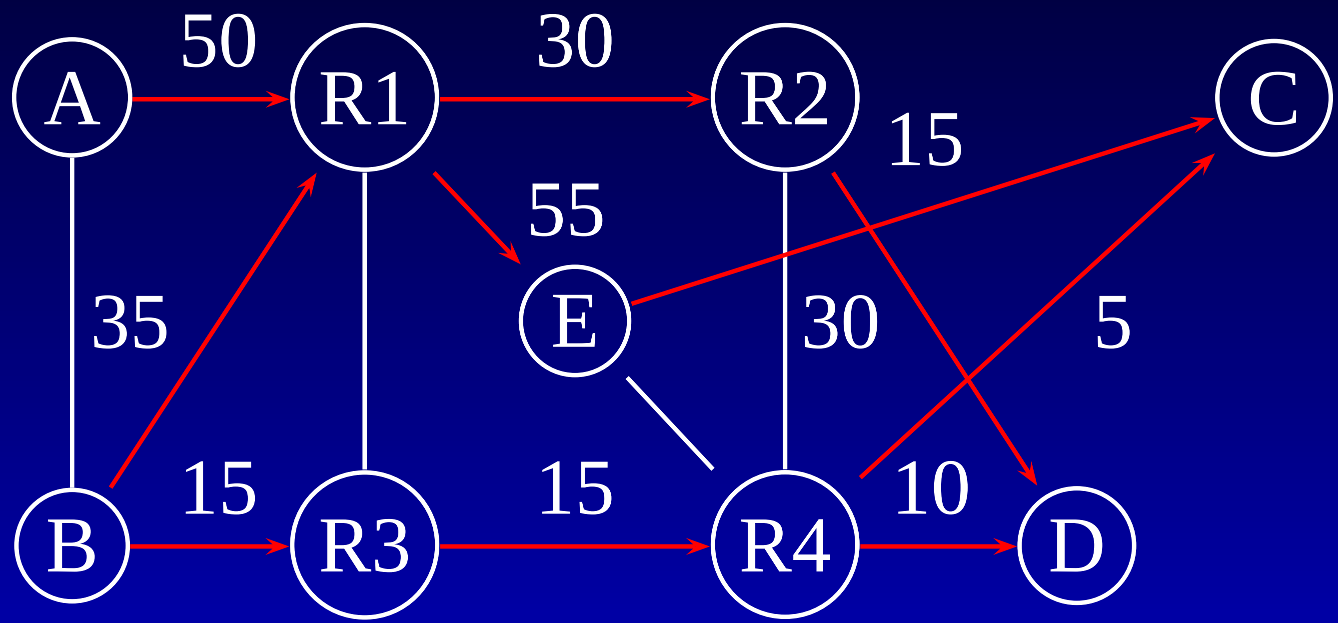
Routing

Demand BE 20



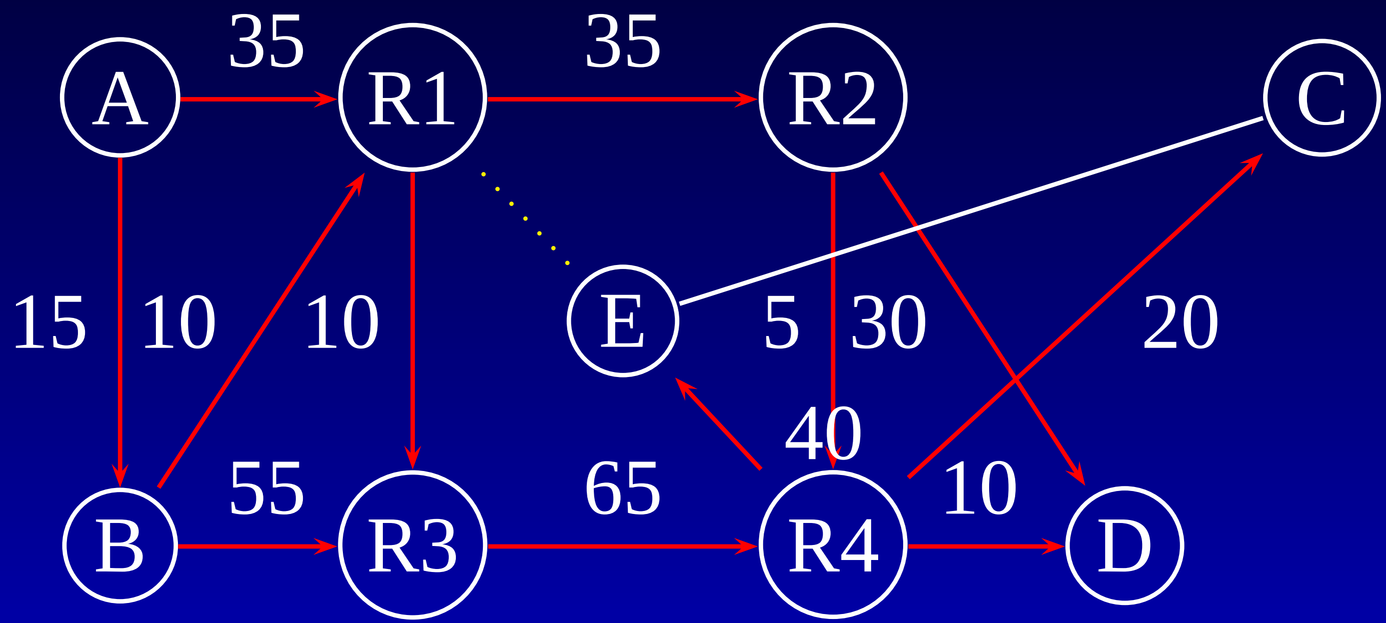
Resulting Network Load

Observed Traffic



Considering failure of R1-E

Observed Traffic



Can we do better?

- Choose single, explicit path for each demand

Can we do better?

- Choose single, explicit path for each demand
- Requires hardware support in routers (MPLS-TE)

Can we do better?

- Choose single, explicit path for each demand
- Requires hardware support in routers (MPLS-TE)
- Baseline: CSPF, greedy heuristic

Can we do better?

- Choose single, explicit path for each demand
- Requires hardware support in routers (MPLS-TE)
- Baseline: CSPF, greedy heuristic
- Why not just use (Integer) Multi-Commodity Flow Problem Solution?
  - Can not use arbitrary, fractional flows
  - MILP does not scale too well

Can we do better?

- Choose single, explicit path for each demand
- Requires hardware support in routers (MPLS-TE)
- Baseline: CSPF, greedy heuristic
- Why not just use (Integer) Multi-Commodity Flow Problem Solution?
  - Can not use arbitrary, fractional flows
  - MILP does not scale too well
- Modelling Alternatives
  - Link based Model
  - Path based Model
  - Node based Model

Model: Traffic Placement

$$\min_{\{X_{de}\}} \max_{e \in \mathbf{E}} \frac{1}{\texttt{cap}(e)} \sum_{d \in \mathbf{D}} \texttt{bw}(d) X_{de} \qquad \texttt{or} \qquad \min_{\{X_{de}\}} \sum_{e \in \mathbf{E}, d \in \mathbf{D}} \texttt{bw}(d) X_{de}$$

st.

$$\forall d \in \mathbf{D}, \forall n \in \mathbf{N} : \quad \sum_{e \in \mathbf{OUT}(n)} X_{de} - \sum_{e \in \mathbf{IN}(n)} X_{de} = \begin{cases} -1 & n = \texttt{dest}(d) \\ 1 & n = \texttt{orig}(d) \\ 0 & \text{otherwise} \end{cases}$$

$$\forall e \in \mathbf{E} : \quad \sum_{d \in \mathbf{D}} \texttt{bw}(d) X_{de} \leq \texttt{cap}(e)$$

$$\forall d \in \mathbf{D} : \quad \sum_{e \in \mathbf{E}} \texttt{del}(e) X_{de} \leq \texttt{req}(d)$$

$$X_{de} \in \{0, 1\}$$

Solution Methods

- Lagrangian Relaxation - path decomposition [Ouaja&Richards2003]
  - Dualize capacity constraints
  - Starting with CSPF initial solution
  - Finite domain solver for path constraints
  - Added capacity constraints from st-cuts
  - At each step solve shortest path problems

Solution Methods

- Lagrangian Relaxation - path decomposition [Ouaja&Richards2003]
  - Dualize capacity constraints
  - Starting with CSPF initial solution
  - Finite domain solver for path constraints
  - Added capacity constraints from st-cuts
  - At each step solve shortest path problems
- Lagrangian Relaxation - knapsack decomposition [Ouaja&Richards2005]
  - Dualize path constraints
  - At each step solve knapsack problems
  - Reduced cost based filtering

Solution Methods (cont'd)

- Probe Backtracking [Liatsos et al 2003]

Solution Methods (cont'd)

- Probe Backtracking [Liatsos et al 2003]
- Start with (infeasible) CSPF heuristic

Solution Methods (cont'd)

- Probe Backtracking [Liatsos et al 2003]
- Start with (infeasible) CSPF heuristic
- Consider capacity violation
  - Resolve by forcing one demand off/on link
  - Find new path respecting path and added constraints with ILP

Solution Methods (cont'd)

- Probe Backtracking [Liatsos et al 2003]
- Start with (infeasible) CSPF heuristic
- Consider capacity violation
  - Resolve by forcing one demand off/on link
  - Find new path respecting path and added constraints with ILP
- Repeat until no more violations, feasible solution

Solution Methods (cont'd)

- Probe Backtracking [Liatsos et al 2003]
- Start with (infeasible) CSPF heuristic
- Consider capacity violation
  - Resolve by forcing one demand off/on link
  - Find new path respecting path and added constraints with ILP
- Repeat until no more violations, feasible solution
- Optimality proof when exhausted search space
  - Search space often very small

Path Based Model

Demand Acceptance Variant

$$\max_{\{Z_d, Y_{id}\}} \quad \sum_{d \in \mathbf{D}} \texttt{val}(d) Z_d$$

st.

$$\forall d \in \mathbf{D} : \quad \sum_{1 \leq i \leq \texttt{path}(d)} Y_{id} = Z_d$$

$$\forall e \in \mathbf{E} : \quad \sum_{d \in \mathbf{D}} \texttt{bw}(d) \sum_{1 \leq i \leq \texttt{path}(d)} h^e_{id} Y_{id} \leq \texttt{cap}(e)$$

$$Z_d \in \{0, 1\}$$

$$Y_{id} \in \{0, 1\}$$

Solution Methods

- Blocking Islands [Frei&Faltings1999]
  - Feasible solution only
  - CSP with variables ranging over paths for demands
  - No explicit domain representation
  - Possible to perform forward checking by updating blocking island structure

Solution Methods

- Blocking Islands [Frei&Faltings1999]
  - Feasible solution only
  - CSP with variables ranging over paths for demands
  - No explicit domain representation
  - Possible to perform forward checking by updating blocking island structure
- Column Generation [Chabrier2003]

Solution Methods (cont'd)

- Local Search/FD Hybrid [Lever2004]

Solution Methods (cont'd)

- Local Search/FD Hybrid [Lever2004]
- Start with (feasible) CSPF heuristic

Solution Methods (cont'd)

- Local Search/FD Hybrid [Lever2004]
- Start with (feasible) CSPF heuristic
- Add more demands one by one
  - Use repair to solve capacity violations

Solution Methods (cont'd)

- Local Search/FD Hybrid [Lever2004]
- Start with (feasible) CSPF heuristic
- Add more demands one by one
  - Use repair to solve capacity violations
- Use FD model to check necessary conditions
  - Determine bottlenecks by st-cuts
  - Force paths on/off links

Surviving Element Failures

- What happens if element on selected path fails?

Surviving Element Failures

- What happens if element on selected path fails?
- Choose second path which is link (element) disjoint

Surviving Element Failures

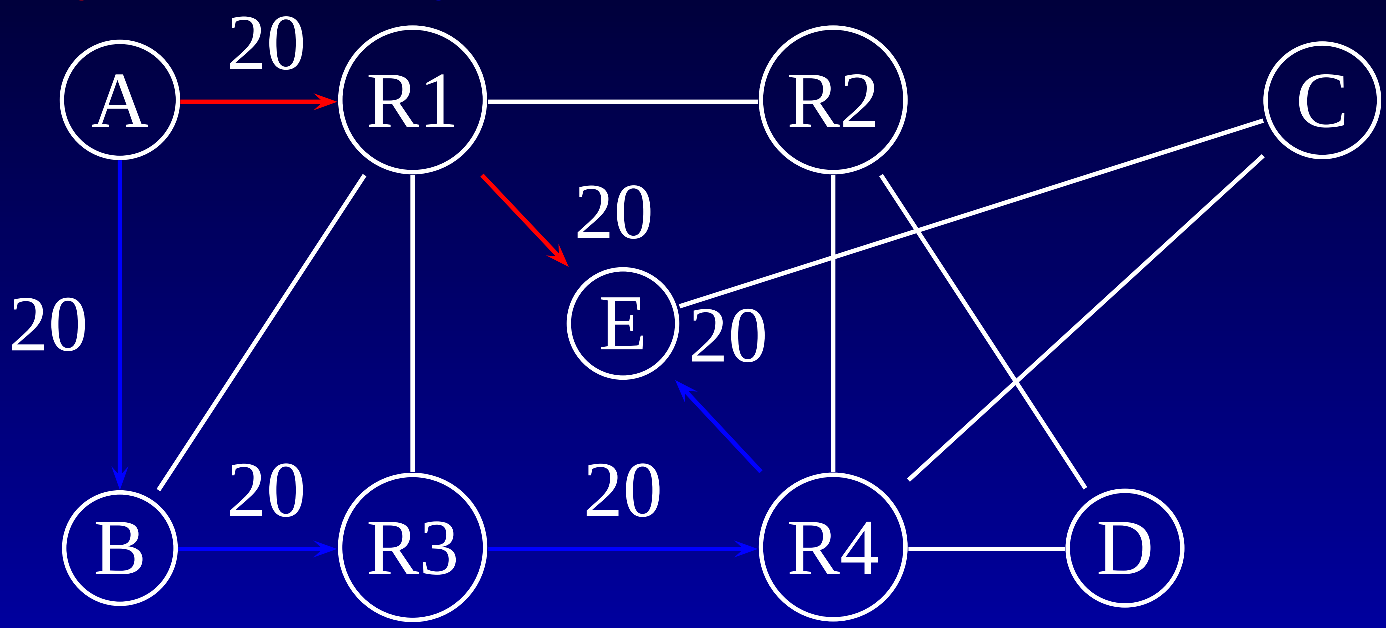
- What happens if element on selected path fails?
- Choose second path which is link (element) disjoint
- State bandwidth constraints for each considered failure case

Surviving Element Failures

- What happens if element on selected path fails?
- Choose second path which is link (element) disjoint
- State bandwidth constraints for each considered failure case
- Problem: Very large number of capacity constraints

Example

Primary/Secondary path for demand AE



Failed Element	No Failure	A-R1	R1-E	All Others
Capacity for Path	Primary	Secondary	Secondary	Primary

Model: Secondary Paths

$$\max_{\{Z_d, X_{de}, W_{de}\}} \quad \sum_{d \in \mathbf{D}} \mathtt{val}(d) Z_d$$

$$\forall d \in \mathbf{D}, \forall n \in \mathbf{N} : \quad \sum_{e \in \mathbf{OUT}(n)} X_{de} - \sum_{e \in \mathbf{IN}(n)} X_{de} = \begin{cases} -Z_d & n = \mathtt{dest}(d) \\ Z_d & n = \mathtt{orig}(d) \\ 0 & \text{otherwise} \end{cases}$$

$$\forall e \in \mathbf{E} : \quad \sum_{d \in \mathbf{D}} \mathtt{bw}(d) * X_{de} \leq \mathtt{cap}(e)$$

$$\forall d \in \mathbf{D}, \forall n \in \mathbf{N} : \quad \sum_{e \in \mathbf{OUT}(n)} W_{de} - \sum_{e \in \mathbf{IN}(n)} W_{de} = \begin{cases} -Z_d & n = \mathtt{dest}(d) \\ Z_d & n = \mathtt{orig}(d) \\ 0 & \text{otherwise} \end{cases}$$

$$\forall e \in \mathbf{E}, \forall e' \in \mathbf{E} \setminus e : \quad \sum_{d \in \mathbf{D}} \mathtt{bw}(d) * (X_{de} - X_{de'} * X_{de} + X_{de'} * W_{de}) \leq \mathtt{cap}(e)$$

$$\forall d \in \mathbf{D}, \forall e \in \mathbf{E} : \quad X_{de} + W_{de} \leq 1$$

$$Z_d \in \{0, 1\}$$

$$X_{de} \in \{0, 1\}$$

$$W_{de} \in \{0, 1\}$$

Solution Method

- Benders Decomposition [Xia&Simonis2005]

Solution Method

- Benders Decomposition [Xia&Simonis2005]
- Use MILP for standard demand acceptance problem

Solution Method

- Benders Decomposition [Xia&Simonis2005]
- Use MILP for standard demand acceptance problem
- Find two link disjoint paths for each demand

Solution Method

- Benders Decomposition [Xia&Simonis2005]
- Use MILP for standard demand acceptance problem
- Find two link disjoint paths for each demand
- Sub-problems consist of capacity constraints for failure cases

Solution Method

- Benders Decomposition [Xia&Simonis2005]
- Use MILP for standard demand acceptance problem
- Find two link disjoint paths for each demand
- Sub-problems consist of capacity constraints for failure cases
- Benders cuts are just no-good cuts for secondary violations

Planning Ahead

- Consider demands with fixed start and end times

Planning Ahead

- Consider demands with fixed start and end times
- Demands overlapping in time compete for bandwidth

Planning Ahead

- Consider demands with fixed start and end times
- Demands overlapping in time compete for bandwidth
- Demands arrive in batches, not always in temporal sequence

Planning Ahead

- Consider demands with fixed start and end times
- Demands overlapping in time compete for bandwidth
- Demands arrive in batches, not always in temporal sequence
- Problem called Bandwidth on Demand (BoD)

Model: BoD

$$\max_{\{Z_d, X_{de}\}} \quad \sum_{d \in \mathbf{D}} \mathtt{val}(d) Z_d$$

st.

$$\mathbf{T} = \{\mathtt{start}(d) | d \in \mathbf{D}\}$$

$$\forall d \in \mathbf{D}, \forall n \in \mathbf{N} : \quad \sum_{e \in \mathbf{OUT}(n)} X_{de} - \sum_{e \in \mathbf{IN}(n)} X_{de} = \begin{cases} -Z_d & n = \mathtt{dest}(d) \\ Z_d & n = \mathtt{orig}(d) \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t \in \mathbf{T}, \forall e \in \mathbf{E} : \quad \sum_{\substack{d \in \mathbf{D} \\ \mathtt{start}(d) \le t \\ t < \mathtt{end}(d)}} \mathtt{bw}(d) X_{de} \le \mathtt{cap}(e)$$

$$Z_d \in \{0, 1\}$$

$$X_{de} \in \{0, 1\}$$

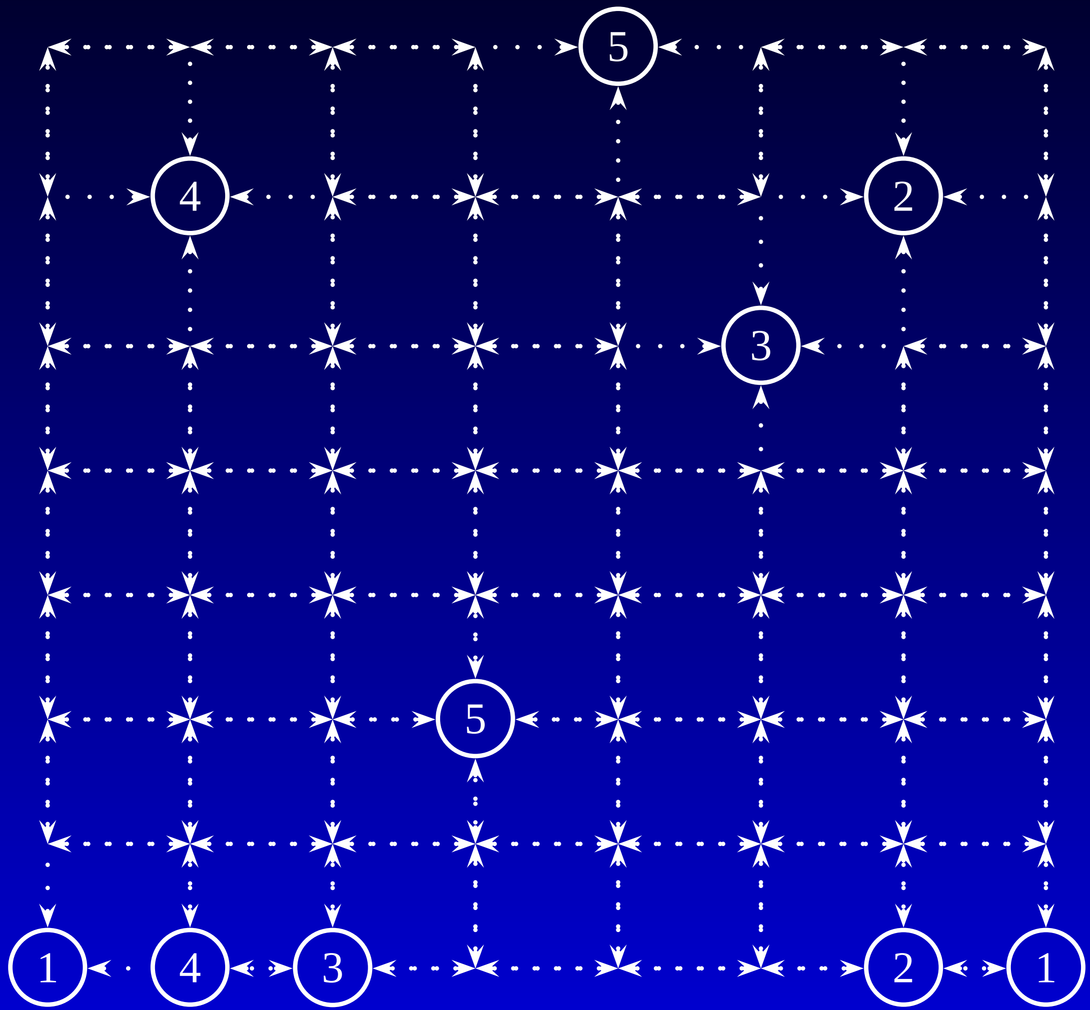
Solution Methods

- France Telecom for ATM network [Lauvergne et al 2002, Loudni et al 2003]

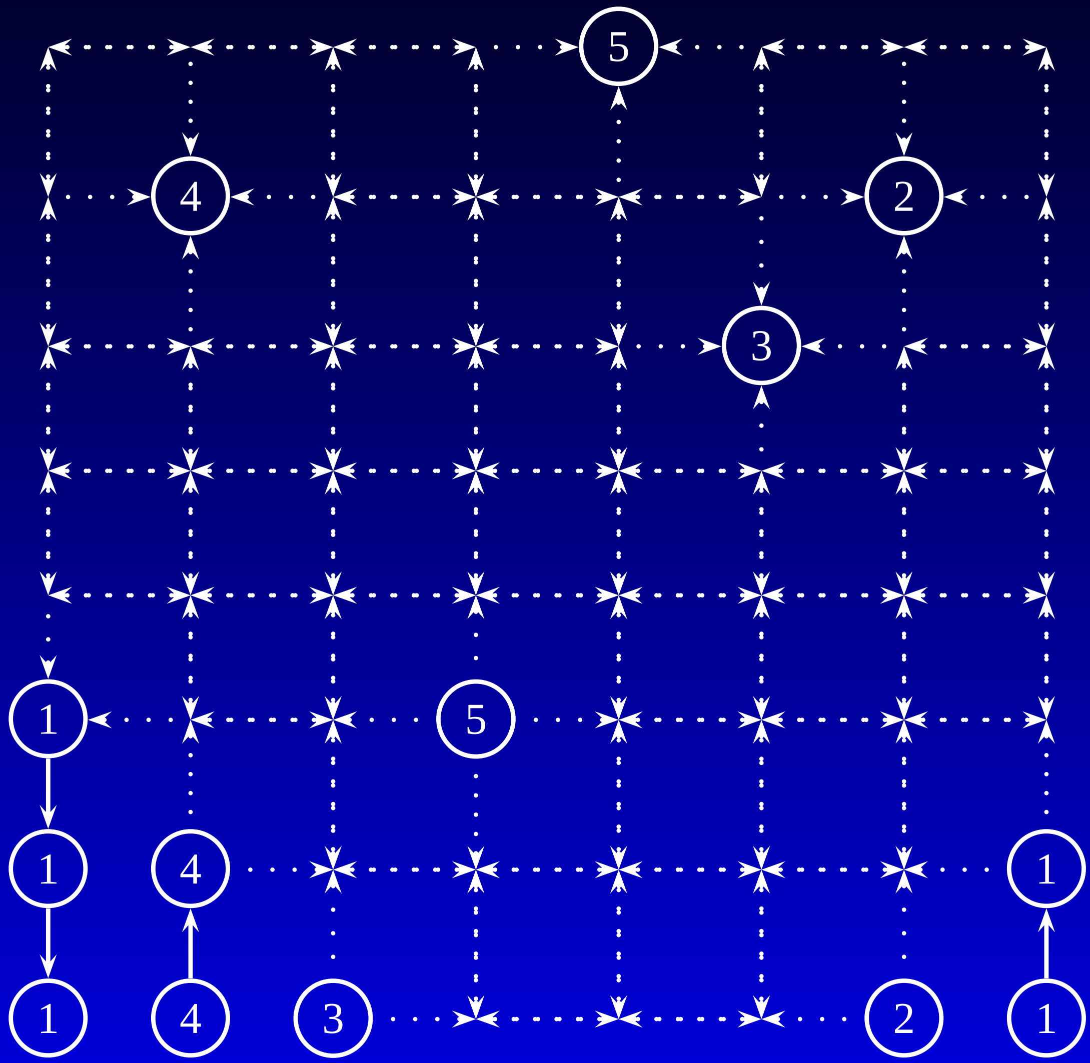
Solution Methods

- France Telecom for ATM network [Lauvergne et al 2002, Loudni et al 2003]
- Schlumberger Dexa.net (PTL, IC-Parc)

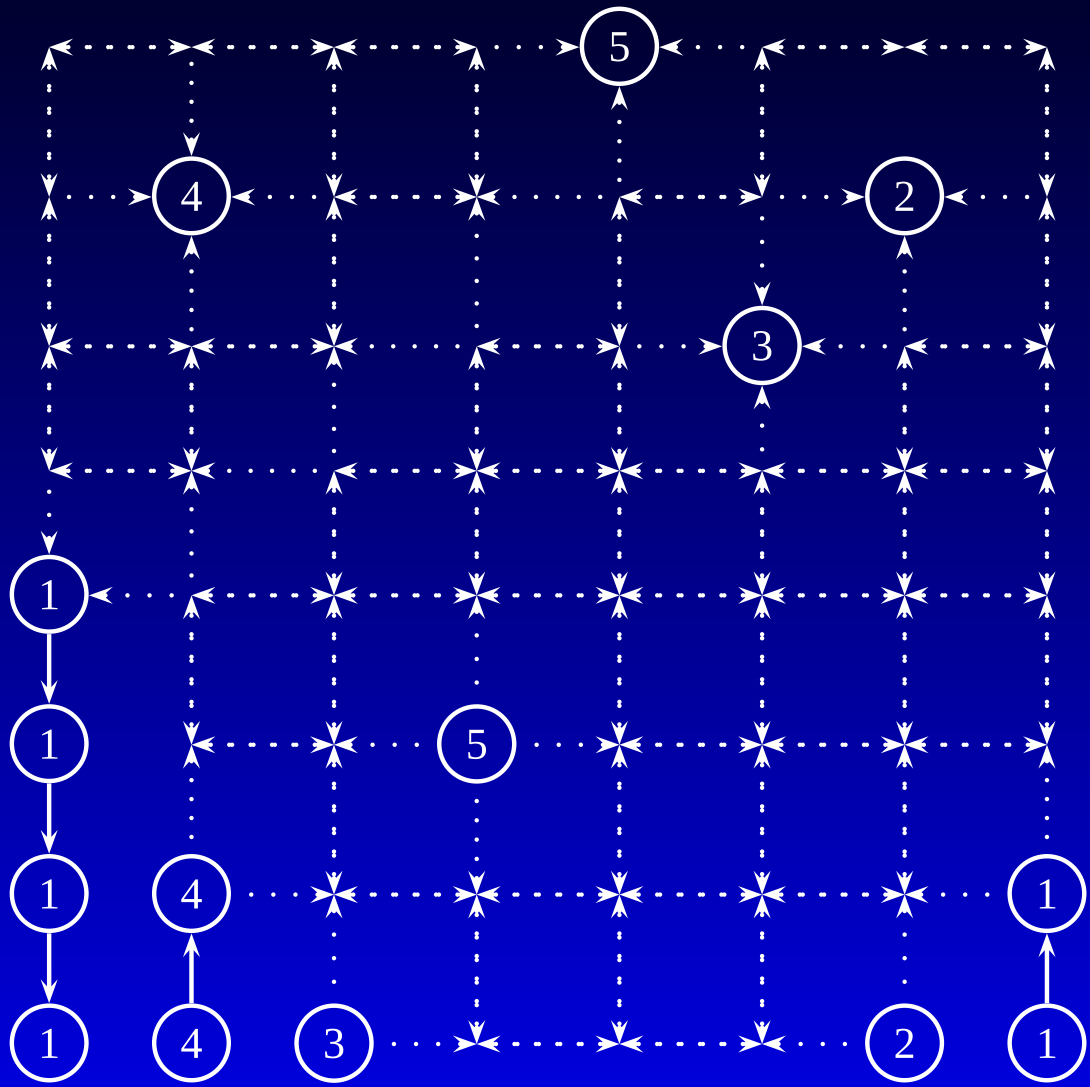
Diversion: Nikoli Numberlink



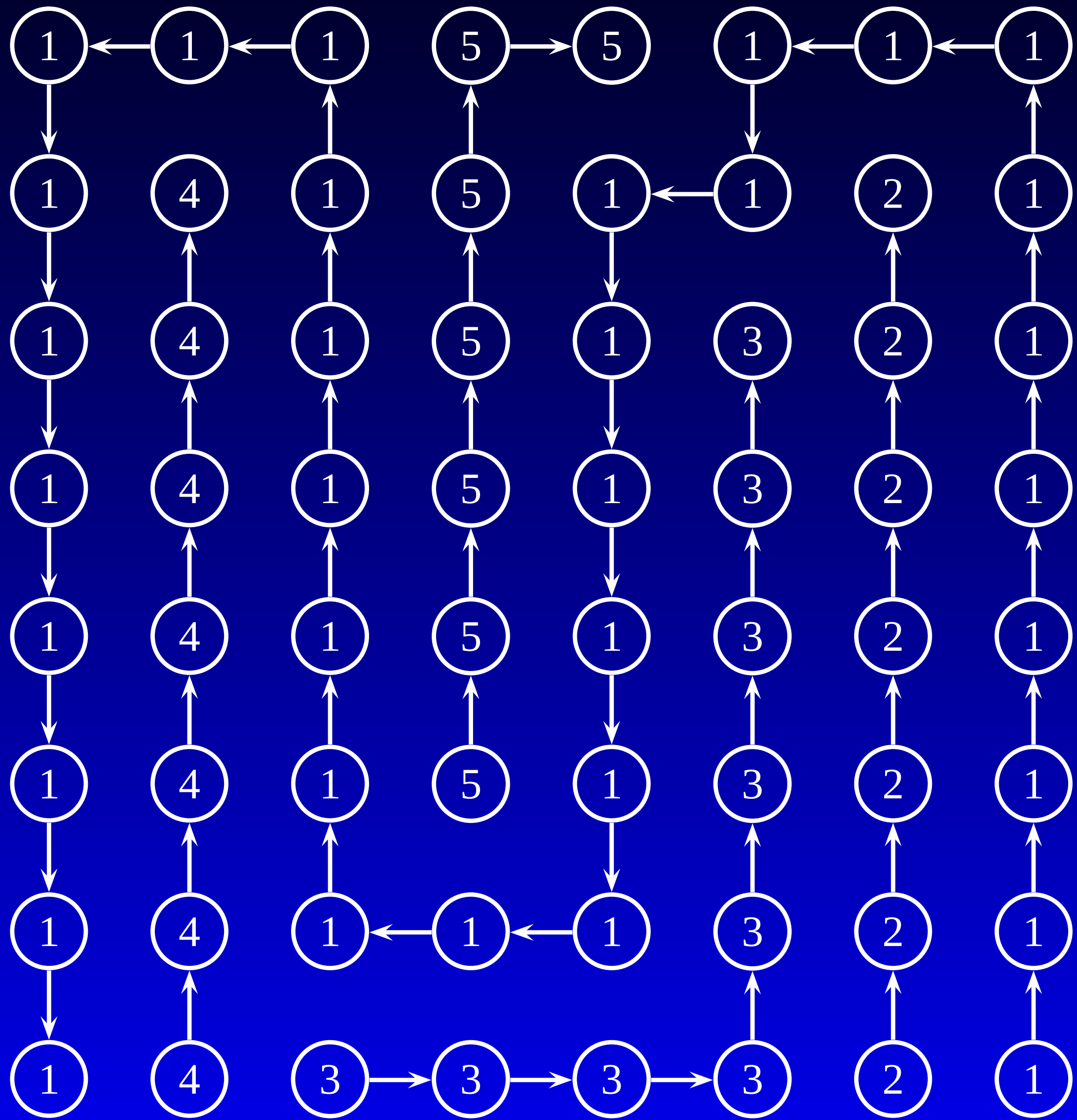
Diversion: Nikoli Numberlink



Diversion: Nikoli Numberlink



Diversion: Nikoli Numberlink



Commercial Break

- As of 21 Sep 2006, ECLiPSe is open source!
  - Released by current owner, Cisco Systems Inc
  - Supersedes previous academic licensing scheme

Commercial Break

- As of 21 Sep 2006, ECLiPSe is open source!
  - Released by current owner, Cisco Systems Inc
  - Supersedes previous academic licensing scheme
- Cisco-style MPL (Mozilla Public Licence)
  - Permission to use for any purpose
  - Applications can remain proprietary

Commercial Break

- As of 21 Sep 2006, ECLiPSe is open source!
  - Released by current owner, Cisco Systems Inc
  - Supersedes previous academic licensing scheme
- Cisco-style MPL (Mozilla Public Licence)
  - Permission to use for any purpose
  - Applications can remain proprietary
- Source available now, binaries coming
  - see: `www.sourceforge.net/projects/eclipse-clp`

Commercial Break

- As of 21 Sep 2006, ECLiPSe is open source!
  - Released by current owner, Cisco Systems Inc
  - Supersedes previous academic licensing scheme
- Cisco-style MPL (Mozilla Public Licence)
  - Permission to use for any purpose
  - Applications can remain proprietary
- Source available now, binaries coming
  - see: `www.sourceforge.net/projects/eclipse-clp`
- Foundation web site coming soon
  - see: `www.eclipse-clp.org`

How to get a Traffic Matrix?

- Many algorithms assume given traffic matrix

How to get a Traffic Matrix?

- Many algorithms assume given traffic matrix
- Traffic flow information is not collected in the routers

How to get a Traffic Matrix?

- Many algorithms assume given traffic matrix
- Traffic flow information is not collected in the routers
- Only link traffic is readily available

How to get a Traffic Matrix?

- Many algorithms assume given traffic matrix
- Traffic flow information is not collected in the routers
- Only link traffic is readily available
- Demand pattern changes over time, often quite dramatically

How to get a Traffic Matrix?

- Many algorithms assume given traffic matrix
- Traffic flow information is not collected in the routers
- Only link traffic is readily available
- Demand pattern changes over time, often quite dramatically
- Measuring traffic flows with probes is very costly

How to get a Traffic Matrix?

- Many algorithms assume given traffic matrix
- Traffic flow information is not collected in the routers
- Only link traffic is readily available
- Demand pattern changes over time, often quite dramatically
- Measuring traffic flows with probes is very costly

From a network consultant:

> *We have been working on extracting a TM for this network for 15 months, and we still don't have a clue if we've got it right.*

Idea

- We are back considering routed networks

Idea

- We are back considering routed networks
- Use the observed traffic to deduce traffic flows

Idea

- We are back considering routed networks
- Use the observed traffic to deduce traffic flows
- Network Tomography [Vardi1996]
    - All flows routed over a link cause the observed traffic
    - Must correct for observation errors
    - Highly dependent on accurate routing model

Idea

- We are back considering routed networks
- Use the observed traffic to deduce traffic flows
- Network Tomography [Vardi1996]
  - All flows routed over a link cause the observed traffic
  - Must correct for observation errors
  - Highly dependent on accurate routing model
- Gravity Model [Medina et al 2002]
  - Ignore core of network
  - Assume that flows are proportional to product of ingress/egress size

Idea

- We are back considering routed networks
- Use the observed traffic to deduce traffic flows
- Network Tomography [Vardi1996]
  - All flows routed over a link cause the observed traffic
  - Must correct for observation errors
  - Highly dependent on accurate routing model
- Gravity Model [Medina et al 2002]
  - Ignore core of network
  - Assume that flows are proportional to product of ingress/egress size
- Results are very hard to validate/falsify

Model: Traffic Flow Analysis

$$\forall i, j \in \mathbf{N}: \quad \min_{\{F_{ij}\}} / \max_{\{F_{ij}\}} \quad F_{ij}$$

st.

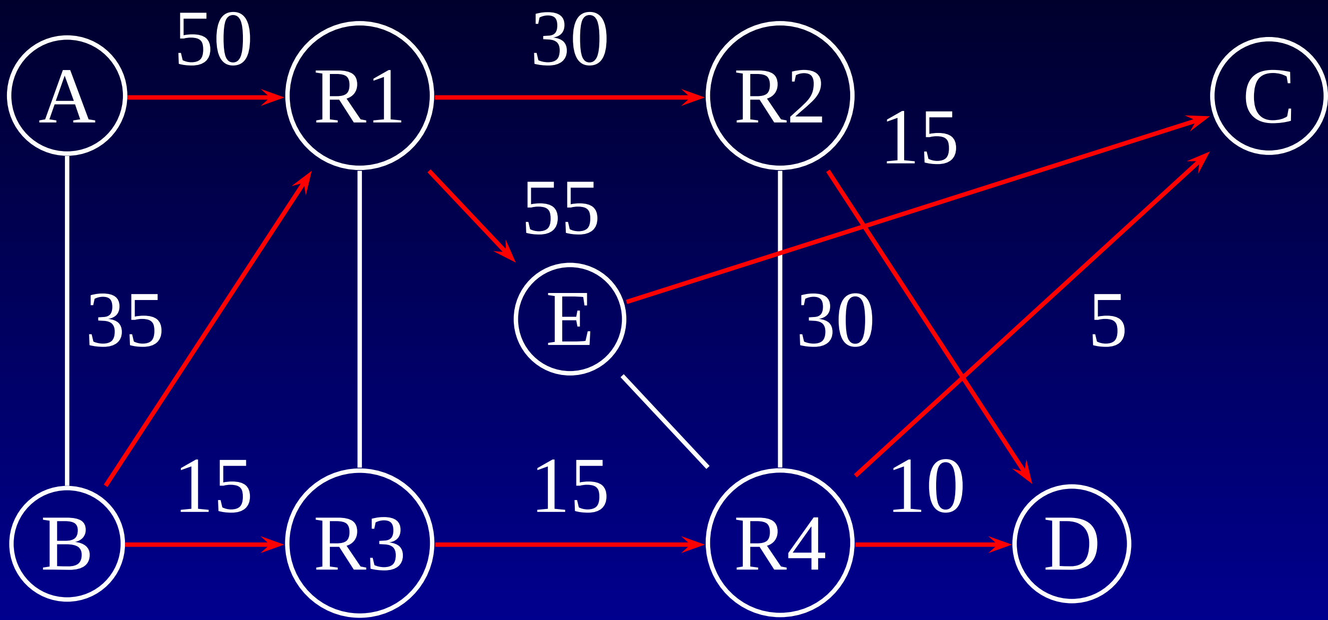
$$\forall e \in \mathbf{E}: \quad \sum_{i,j \in \mathbf{N}} r_{ij}^{e} F_{ij} = \mathtt{traf}(e)$$

$$\forall i \in \mathbf{N}: \quad \sum_{j \in \mathbf{N}} F_{ij} = \mathtt{ext}^{in}(i)$$

$$\forall j \in \mathbf{N}: \quad \sum_{i \in \mathbf{N}} F_{ij} = \mathtt{ext}^{out}(j)$$

$$F_{ij} \geq 0$$

Start with Link Traffic



Setup Model to Find Flows

```
[AC,AD,BC,BD,AE,BE] :: 0.0 .. 1.0Inf,
AC + AD + AE $= 50, % A R1
0.5*BC + 0.5*BD + BE $= 35, % B R1
0.5*BC + 0.5*BD $= 15, % B R3
AD + 0.5*BD $= 30, % R1 R2
AC + 0.5*BC + AE +BE $= 55, % R1 E
AD + 0.5*BD $= 30, % R2 D
0.5*BC + 0.5*BD $= 15, % R3 R4
AC + 0.5*BC $= 15, % E C
0.5*BC $= 5, % R4 C
0.5*BD $= 10, % R4 D
```

	C	D	E
A	10	20	20
B	10	20	20

TFA Result for Benchmarks

Network	Low/Simul (%)	High/Simul (%)	Obj	Time (sec)
dexa	0	2310.65	1190	11
as1221	0.09	8398.64	11556	1318
as1239	n/a	n/a	n/a	n/a
as1755	0.15	6255.31	7482	699
as3257	0.04	12260.03	25760	12389
as3967	0.1	5387.10	6162	500
as6461	0.28	8688.39	19740	8676

What now?

- Clearly, some particular solution can not be trusted

What now?

- Clearly, some particular solution can not be trusted
- Massively under constrained problem
  - $|N|^2$ variables
  - $|E| + 2|N|$ constraints
  - $2|N|^2$ queries

What now?

- Clearly, some particular solution can not be trusted
- Massively under constrained problem
  - $|N|^2$ variables
  - $|E| + 2|N|$ constraints
  - $2|N|^2$ queries
- Ill-conditioned even after error correction

What now?

- Clearly, some particular solution can not be trusted
- Massively under constrained problem
  - $|N|^2$ variables
  - $|E| + 2|N|$ constraints
  - $2|N|^2$ queries
- Ill-conditioned even after error correction
- Aggregation helps
  - We are usually not interested in individual flows
  - We want to use the TM to investigate something else

Resilience Analysis

- How much capacity is needed to survive all reasonable failures?

Resilience Analysis

- How much capacity is needed to survive all reasonable failures?
- Use normal state as starting point

Resilience Analysis

- How much capacity is needed to survive all reasonable failures?
- Use normal state as starting point
- Consider routing in each failure case

Resilience Analysis

- How much capacity is needed to survive all reasonable failures?
- Use normal state as starting point
- Consider routing in each failure case
- Aggregate flows in rerouted network

Resilience Analysis

- How much capacity is needed to survive all reasonable failures?
- Use normal state as starting point
- Consider routing in each failure case
- Aggregate flows in rerouted network
- Calculate bounds on traffic in failure case

Model: Resilience Analysis

$$\forall e \in \mathbf{E}: \quad \min_{\{F_{ij}\}} / \max_{\{F_{ij}\}} \quad \sum_{i,j \in \mathbf{N}} r_{ij}^{\bar{e}} F_{ij}$$

st.

$$\forall e \in \mathbf{E}: \quad \sum_{i,j \in \mathbf{N}} r_{ij}^{e} F_{ij} = \mathtt{traf}(e)$$

$$\forall i \in \mathbf{N}: \quad \sum_{j \in \mathbf{N}} F_{ij} = \mathtt{ext}^{in}(i)$$

$$\forall j \in \mathbf{N}: \quad \sum_{i \in \mathbf{N}} F_{ij} = \mathtt{ext}^{out}(j)$$

$$F_{ij} \geq 0$$

Resilience Analysis

Network	Low/Simul (%)	High/Simul (%)	Obj	Time (sec)	Cases
dexa	68.91	108.25	3503	57	59
as1221	85.75	102.60	14191	2869	153
as1239	*92.53*	*102.64*	*4499*	*44205*	*10*
as1755	92.82	105.39	8409	1815	161
as3257	93.69	103.15	31093	39934	328
as3967	91.60	108.79	9090	1635	141
as6461	96.51	103.44	24808	20840	374

Perspectives

- High polynomial complexity

Perspectives

- High polynomial complexity
- Possible to reduce number of queries
  - Small differences between failure cases
  - Many queries are identical or dominated

Perspectives

- High polynomial complexity
- Possible to reduce number of queries
  - Small differences between failure cases
  - Many queries are identical or dominated
- Possible to reduce size of problem dramatically

Perspectives

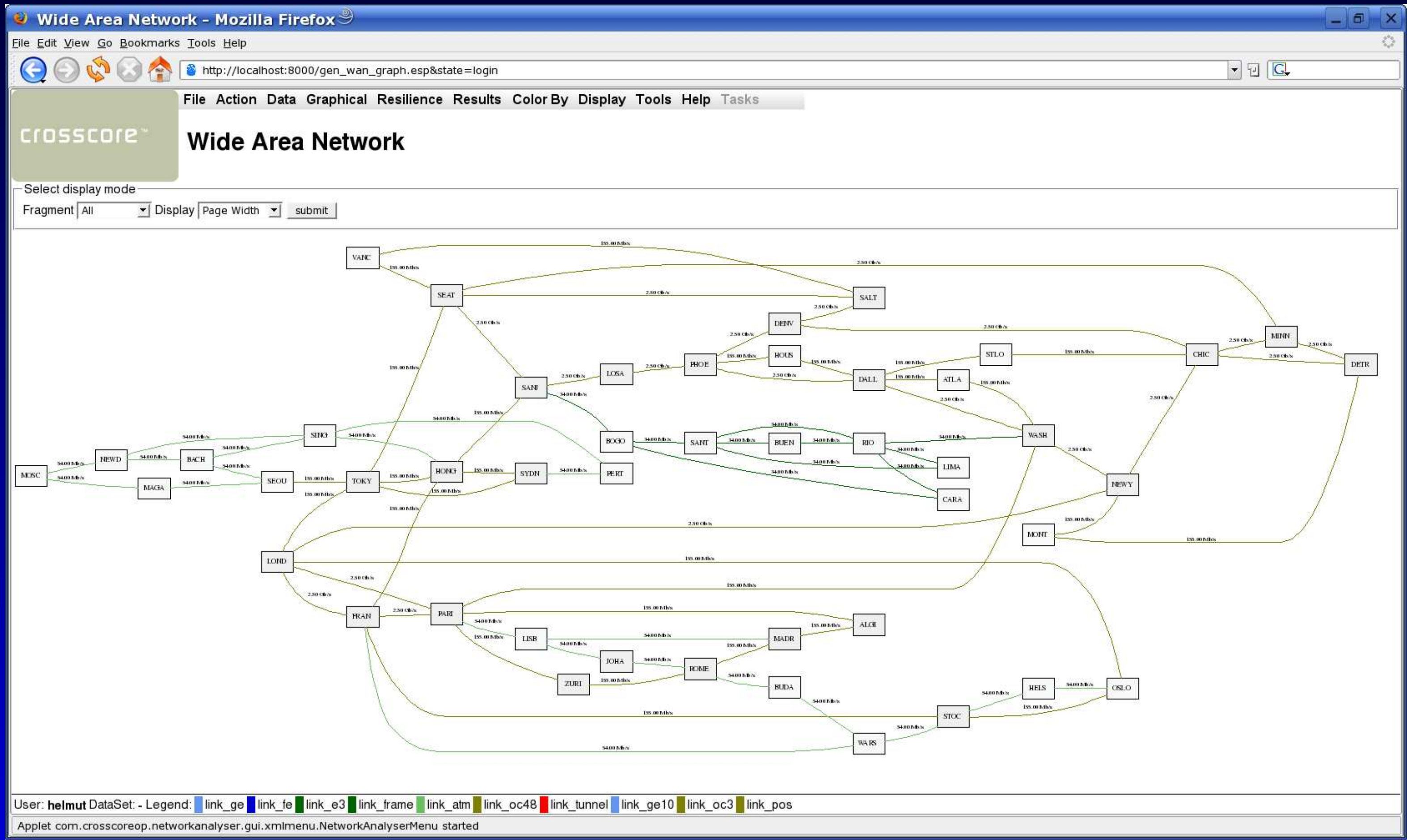
- High polynomial complexity
- Possible to reduce number of queries
  - Small differences between failure cases
  - Many queries are identical or dominated
- Possible to reduce size of problem dramatically
- Integrate multiple measurements in one model

Perspectives

- High polynomial complexity
- Possible to reduce number of queries
  - Small differences between failure cases
  - Many queries are identical or dominated
- Possible to reduce size of problem dramatically
- Integrate multiple measurements in one model
- Which other problems can we solve without explicit TM?

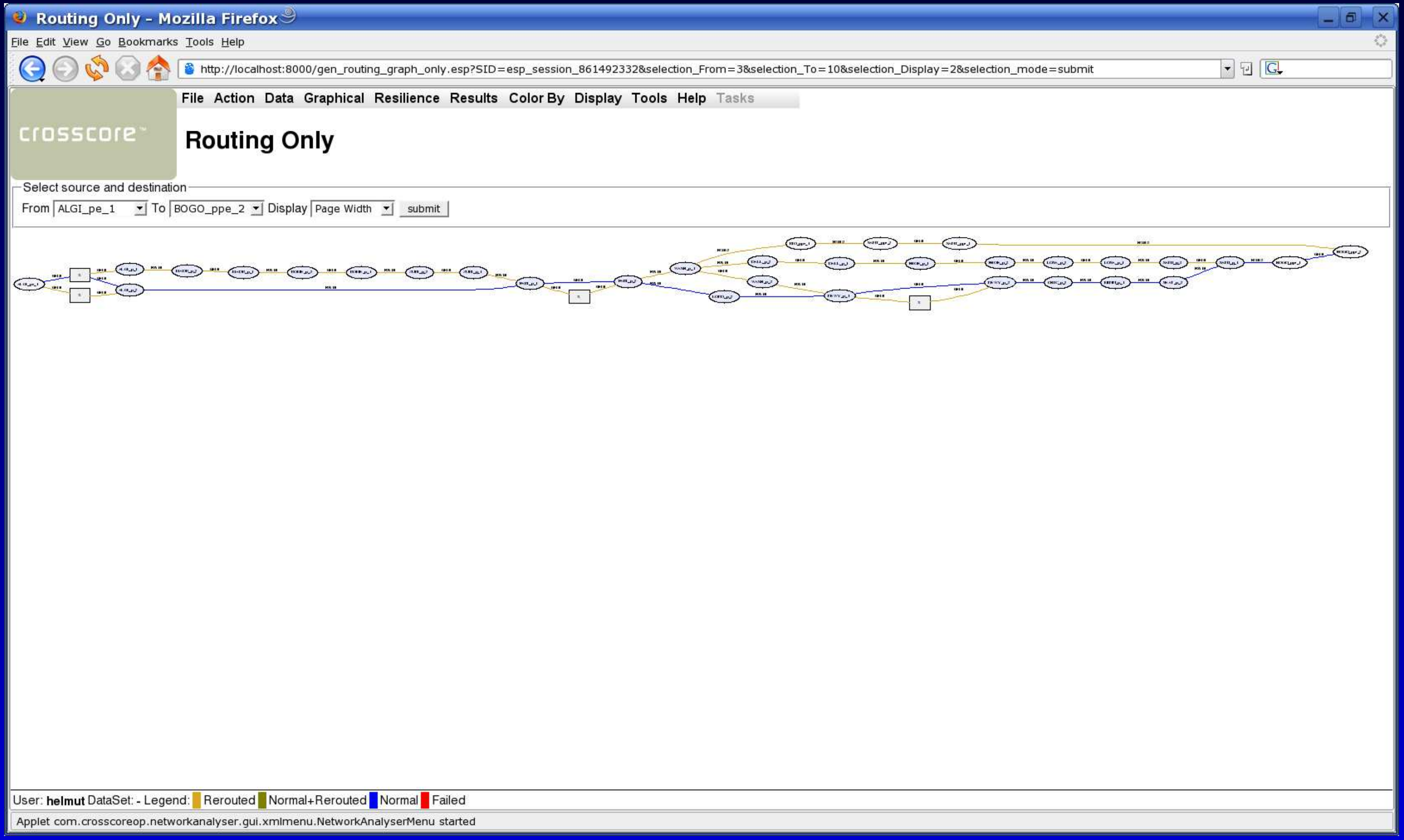
Commercial Break

Core:Insight: Capacity Planning Tool



Commercial Break

Core:Insight Routing Module

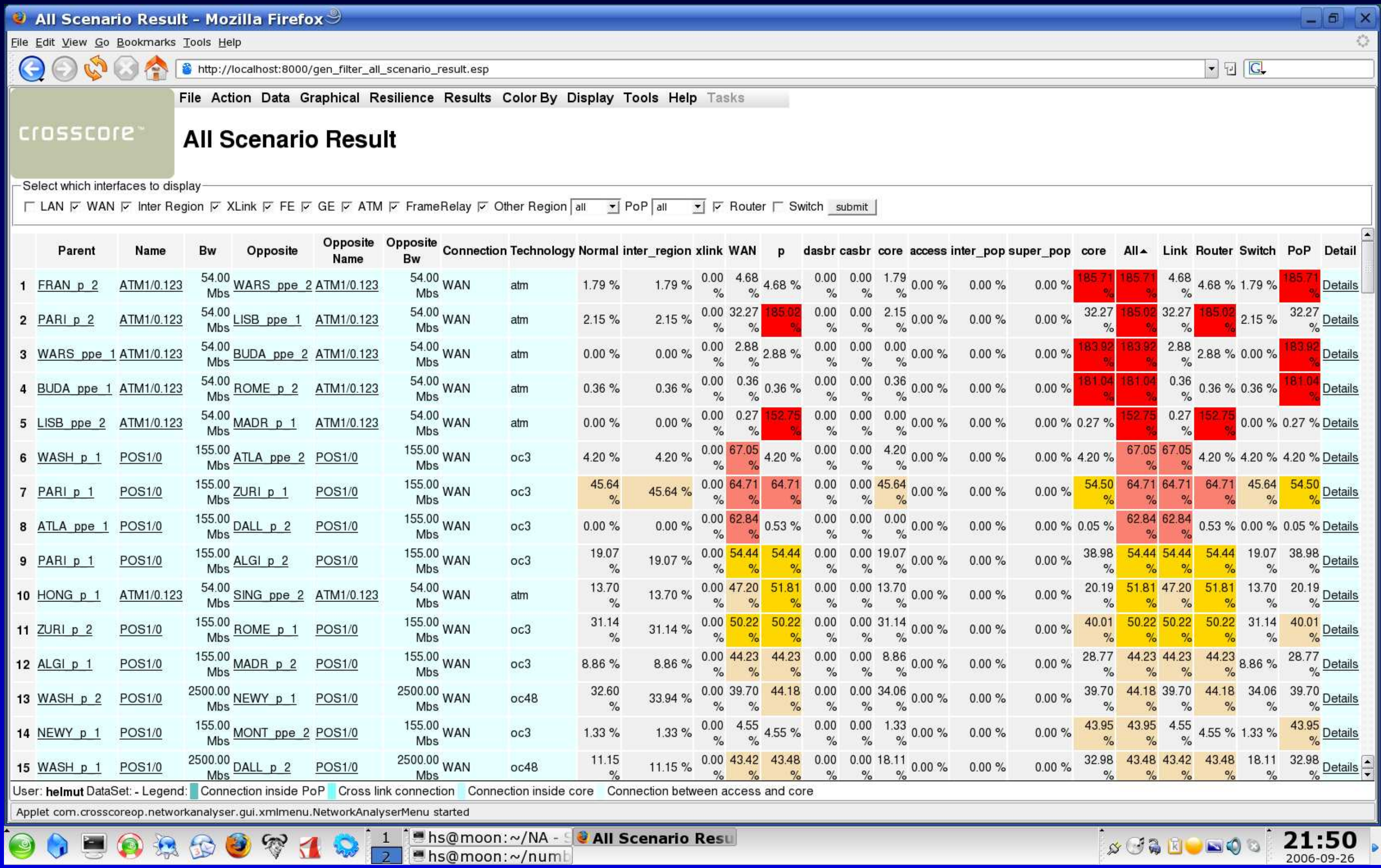


Commercial Break

Core:Insight Traffic Profile

Commercial Break

Core:Insight Capacity Information



	Parent	Name	Bw	Opposite	Opposite Name	Opposite Bw	Connection	Technology	Normal	inter_region	xlink	WAN	p	dasbr	casbr	core	access	inter_pop	super_pop	core	All▲	Link	Router	Switch	PoP	Detail
1	FRAN_p_2	ATM1/0.123	54.00 Mbs	WARS_ppe_2	ATM1/0.123	54.00 Mbs	WAN	atm	1.79 %	1.79 %	0.00 %	4.68 %	4.68 %	0.00 %	0.00 %	1.79 %	0.00 %	0.00 %	0.00 %	185.71 %	185.71 %	4.68 %	4.68 %	1.79 %	185.71 %	Details
2	PARI_p_2	ATM1/0.123	54.00 Mbs	LISB_ppe_1	ATM1/0.123	54.00 Mbs	WAN	atm	2.15 %	2.15 %	0.00 %	32.27 %	185.02 %	0.00 %	0.00 %	2.15 %	0.00 %	0.00 %	0.00 %	32.27 %	185.02 %	32.27 %	185.02 %	2.15 %	32.27 %	Details
3	WARS_ppe_1	ATM1/0.123	54.00 Mbs	BUDA_ppe_2	ATM1/0.123	54.00 Mbs	WAN	atm	0.00 %	0.00 %	0.00 %	2.88 %	2.88 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	183.92 %	183.92 %	2.88 %	2.88 %	0.00 %	183.92 %	Details
4	BUDA_ppe_1	ATM1/0.123	54.00 Mbs	ROME_p_2	ATM1/0.123	54.00 Mbs	WAN	atm	0.36 %	0.36 %	0.00 %	0.36 %	0.36 %	0.00 %	0.00 %	0.36 %	0.00 %	0.00 %	0.00 %	181.04 %	181.04 %	0.36 %	0.36 %	0.36 %	181.04 %	Details
5	LISB_ppe_2	ATM1/0.123	54.00 Mbs	MADR_p_1	ATM1/0.123	54.00 Mbs	WAN	atm	0.00 %	0.00 %	0.00 %	0.27 %	152.75 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.27 %	152.75 %	0.27 %	152.75 %	0.00 %	0.27 %	Details
6	WASH_p_1	POS1/0	155.00 Mbs	ATLA_ppe_2	POS1/0	155.00 Mbs	WAN	oc3	4.20 %	4.20 %	0.00 %	67.05 %	4.20 %	0.00 %	0.00 %	4.20 %	0.00 %	0.00 %	0.00 %	4.20 %	67.05 %	67.05 %	4.20 %	4.20 %	4.20 %	Details
7	PARI_p_1	POS1/0	155.00 Mbs	ZURI_p_1	POS1/0	155.00 Mbs	WAN	oc3	45.64 %	45.64 %	0.00 %	64.71 %	64.71 %	0.00 %	0.00 %	45.64 %	0.00 %	0.00 %	0.00 %	54.50 %	64.71 %	64.71 %	64.71 %	45.64 %	54.50 %	Details
8	ATLA_ppe_1	POS1/0	155.00 Mbs	DALL_p_2	POS1/0	155.00 Mbs	WAN	oc3	0.00 %	0.00 %	0.00 %	62.84 %	0.53 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.05 %	62.84 %	62.84 %	0.53 %	0.00 %	0.05 %	Details
9	PARI_p_1	POS1/0	155.00 Mbs	ALGI_p_2	POS1/0	155.00 Mbs	WAN	oc3	19.07 %	19.07 %	0.00 %	54.44 %	54.44 %	0.00 %	0.00 %	19.07 %	0.00 %	0.00 %	0.00 %	38.98 %	54.44 %	54.44 %	54.44 %	19.07 %	38.98 %	Details
10	HONG_p_1	ATM1/0.123	54.00 Mbs	SING_ppe_2	ATM1/0.123	54.00 Mbs	WAN	atm	13.70 %	13.70 %	0.00 %	47.20 %	51.81 %	0.00 %	0.00 %	13.70 %	0.00 %	0.00 %	0.00 %	20.19 %	51.81 %	47.20 %	51.81 %	13.70 %	20.19 %	Details
11	ZURI_p_2	POS1/0	155.00 Mbs	ROME_p_1	POS1/0	155.00 Mbs	WAN	oc3	31.14 %	31.14 %	0.00 %	50.22 %	50.22 %	0.00 %	0.00 %	31.14 %	0.00 %	0.00 %	0.00 %	40.01 %	50.22 %	50.22 %	50.22 %	31.14 %	40.01 %	Details
12	ALGI_p_1	POS1/0	155.00 Mbs	MADR_p_2	POS1/0	155.00 Mbs	WAN	oc3	8.86 %	8.86 %	0.00 %	44.23 %	44.23 %	0.00 %	0.00 %	8.86 %	0.00 %	0.00 %	0.00 %	28.77 %	44.23 %	44.23 %	44.23 %	8.86 %	28.77 %	Details
13	WASH_p_2	POS1/0	2500.00 Mbs	NEWY_p_1	POS1/0	2500.00 Mbs	WAN	oc48	32.60 %	33.94 %	0.00 %	39.70 %	44.18 %	0.00 %	0.00 %	34.06 %	0.00 %	0.00 %	0.00 %	39.70 %	44.18 %	39.70 %	44.18 %	34.06 %	39.70 %	Details
14	NEWY_p_1	POS1/0	155.00 Mbs	MONT_ppe_2	POS1/0	155.00 Mbs	WAN	oc3	1.33 %	1.33 %	0.00 %	4.55 %	4.55 %	0.00 %	0.00 %	1.33 %	0.00 %	0.00 %	0.00 %	43.95 %	43.95 %	4.55 %	4.55 %	1.33 %	43.95 %	Details
15	WASH_p_1	POS1/0	2500.00 Mbs	DALL_p_2	POS1/0	2500.00 Mbs	WAN	oc48	11.15 %	11.15 %	0.00 %	43.42 %	43.48 %	0.00 %	0.00 %	18.11 %	0.00 %	0.00 %	0.00 %	32.98 %	43.48 %	43.42 %	43.48 %	18.11 %	32.98 %	Details

Summary

- Problem: Operate cost-effective, high quality IP networks

Summary

- Problem: Operate cost-effective, high quality IP networks
- Different solutions which help improve quality/throughput

Summary

- Problem: Operate cost-effective, high quality IP networks
- Different solutions which help improve quality/throughput
- Solution methods are often hybrids with MIP/LP/Local search
  - Finite domain solver not contributing much
  - Special global constraints missing? [Chambazard&Bourreau2004]
  - Interaction of costs/capacity constraints?

Summary

- Problem: Operate cost-effective, high quality IP networks
- Different solutions which help improve quality/throughput
- Solution methods are often hybrids with MIP/LP/Local search
  - Finite domain solver not contributing much
  - Special global constraints missing? [Chambazard&Bourreau2004]
  - Interaction of costs/capacity constraints?
- Basis for CrossCore products