

Introduction – Chris Develder



- PhD, Ghent University, 2003
 - “Design and analysis of optical packet switching networks”
- Professor at **Ghent University** since Oct. 2007
 - *Research Interests*: dimensioning, modeling and optimizing **optical** (grid/cloud) networks; **smart grids**; multimedia and home networks; **information retrieval**
 - Visiting researcher at UC Davis, CA, USA, Jul-Oct. 2007 (optical grids)
 - Visiting researcher at Columbia Univ., NY, USA, 2013-14 (IR/IE)
- Industry Experience: **network planning/design** tools
 - OPNET Technologies (now part of Riverbed), 2004-05
- More info: <http://users.atlantis.ugent.be/cdvelder>

Dimensioning (optical) networks for cloud computing

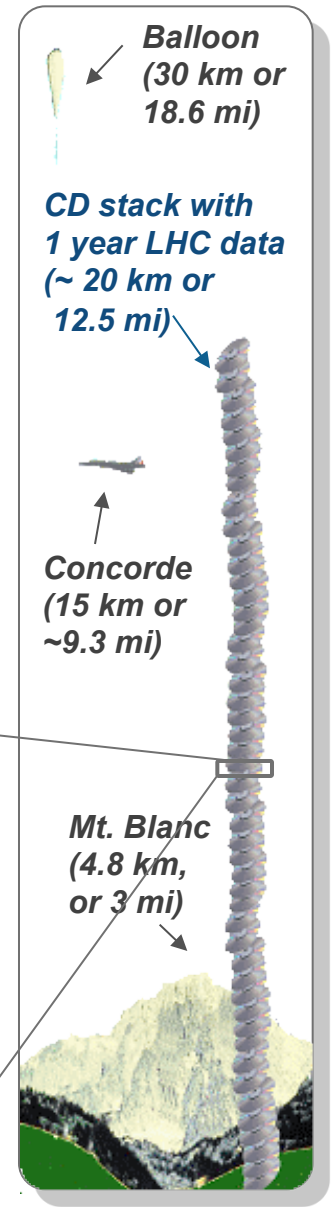
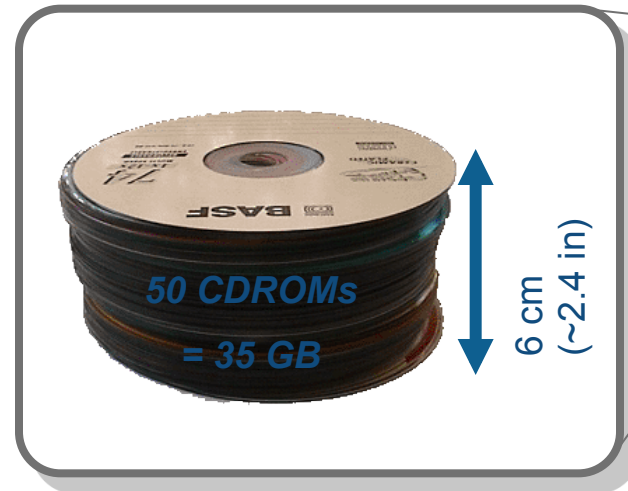
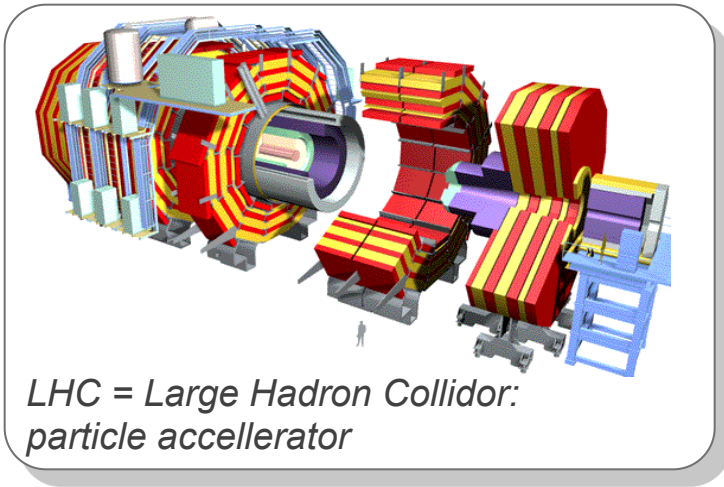
Chris Develder, *et al.*

Ghent University – iMinds
Dept. of Information Technology – IBCN

Background: "Optical grids"

■ eScience:

- By 2015 it is estimated that **particle physicists** will require exabytes (10^{18}) of storage and **petaflops** (10^{15}) per second of computation
- CERN's LHC Computing Grid (LGC), when fully operational generates **15 petabytes** annually (that's ~ 2 Gbit/s)



“Optical grids” for consumer services?

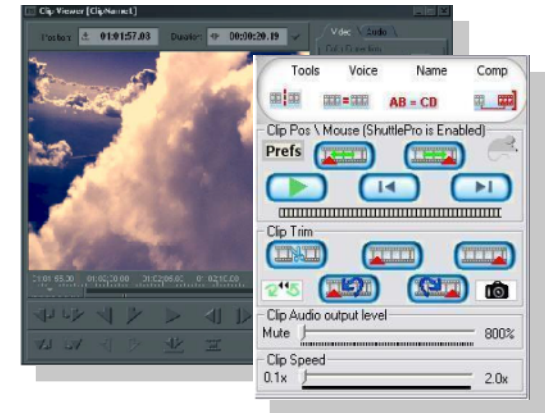
E.g., **video editing:**

2Mpx/frame for HDTV, suppose effect requires 10 flops/px/frame, then evaluating 10 options for 10s clip is **50 Gflops** (today’s high performance PC: <5 Gflops/s)



Online gaming:
e.g. Final Fantasy XI:
1.500.000 gamers

Virtual reality: rendering
of $3 \cdot 10^8$ polygons/s \rightarrow
 10^4 GFlops

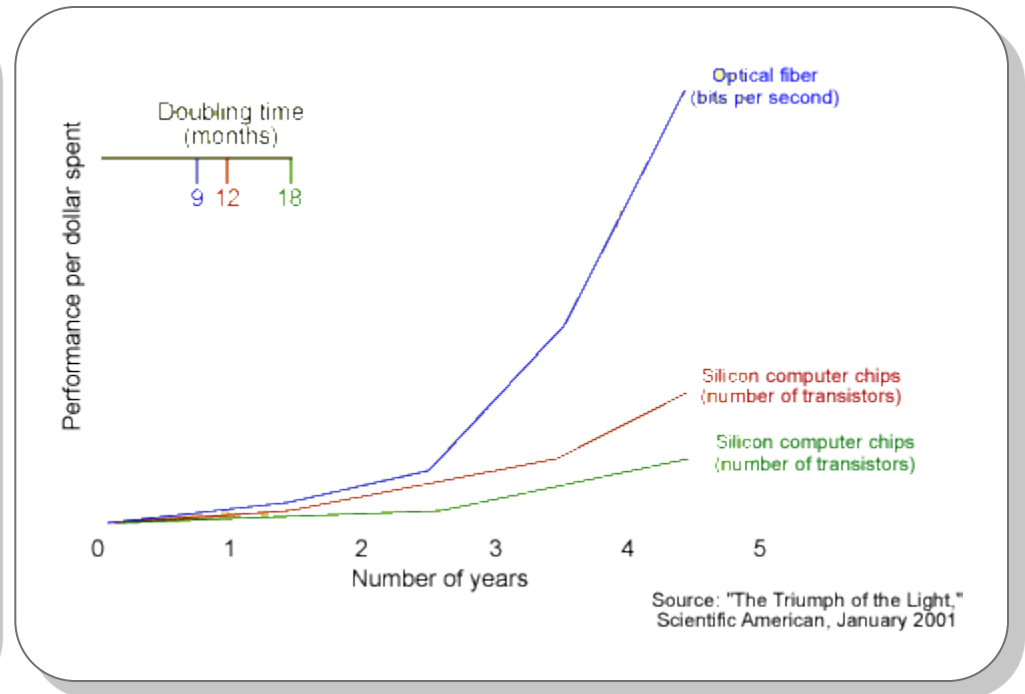
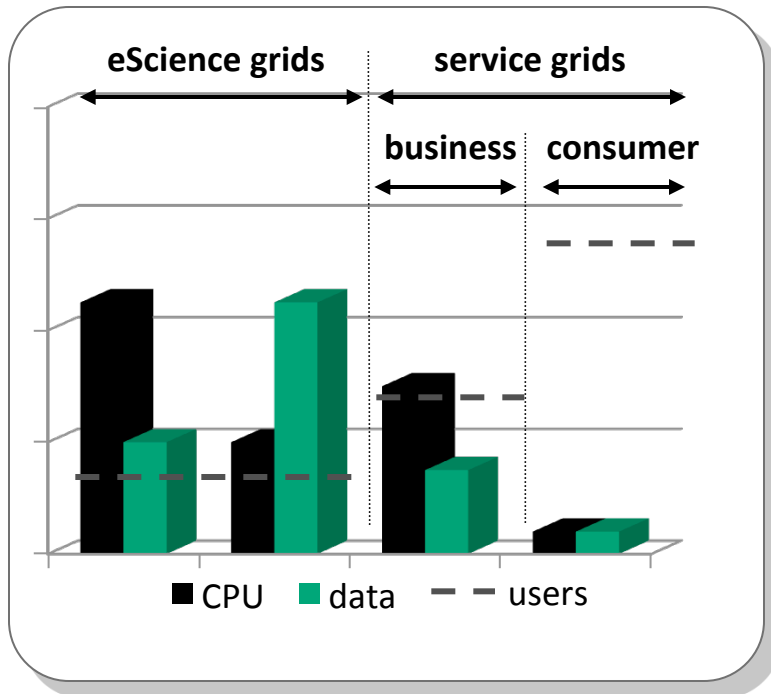


Multimedia editing

Why optics? (3)

■ Conclusion:

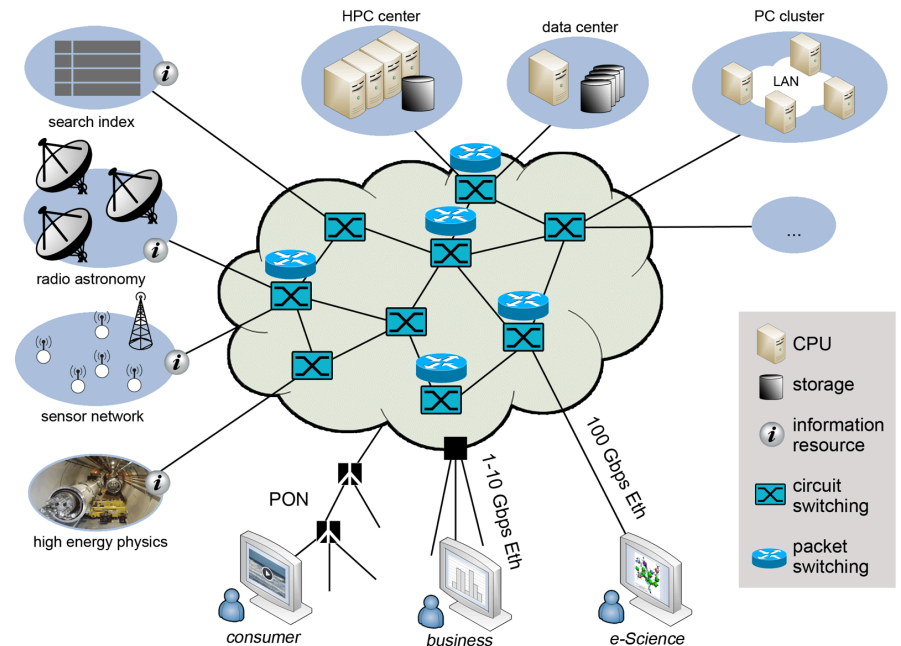
- Grid opportunities ranging from academia over corporate business to home users
- Optical data speeds \geq internal PC bus speeds
⇒ network speed no bottleneck



Today: towards optical grid / cloud computing

Optical networks crucial for increasingly demanding cloud services, e.g.,

- Computing:
 - High energy physics
 - Amazon EC2, Microsoft Azure
- Online storage:
 - Dropbox, Google Drive, etc.
- Collaboration tools:
 - MSOffice 365, Google Doc
- Video streaming:
 - Netflix, YouTube



C. Develder, et al., "Optical networks for grid and cloud computing applications", Proc. IEEE, Vol. 100, No. 5, May 2012, pp. 1149-1167.

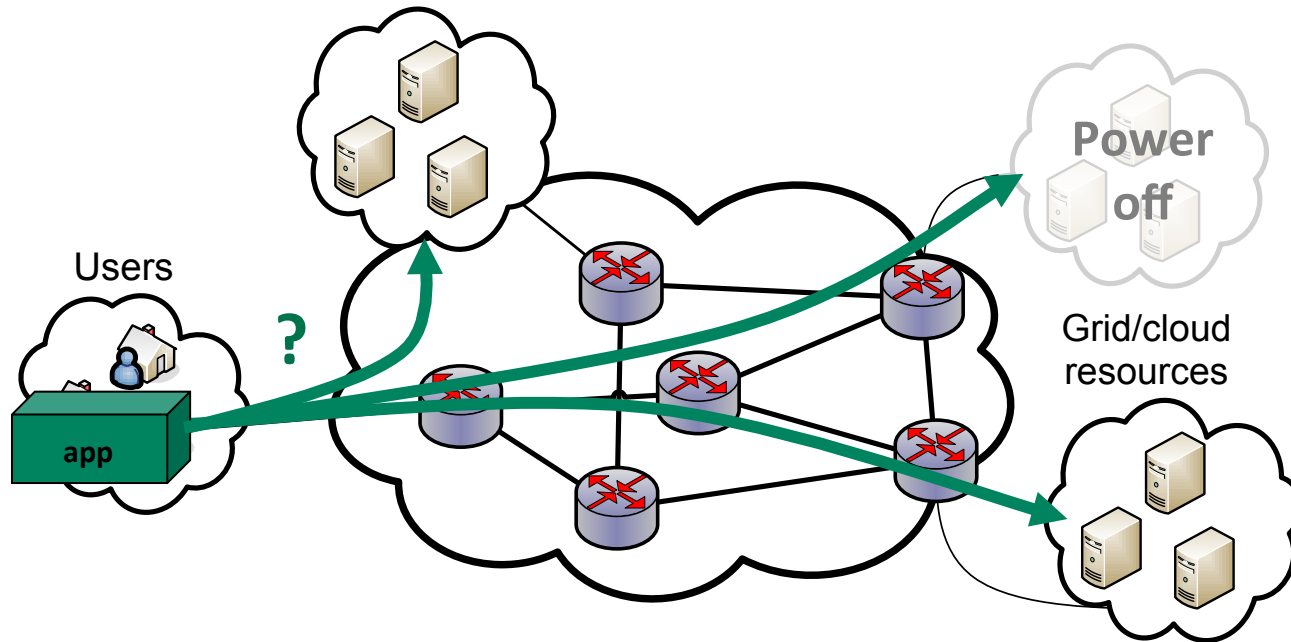
Outline

1. Introduction
2. Network dimensioning for clouds: What's different?
3. An iterative network + server dimensioning approach
4. Exploiting anycast for resilience purposes
5. The next step: accounting for inter-DC synchronization
6. Wrap-up

Dimensioning for clouds: What's different?

Anycast

- Users do (in general) **NOT** care where applications are served
 - E.g., virtual machines in IaaS can be instantiated anywhere
 - E.g., bag-of-tasks grid jobs can be run at any server

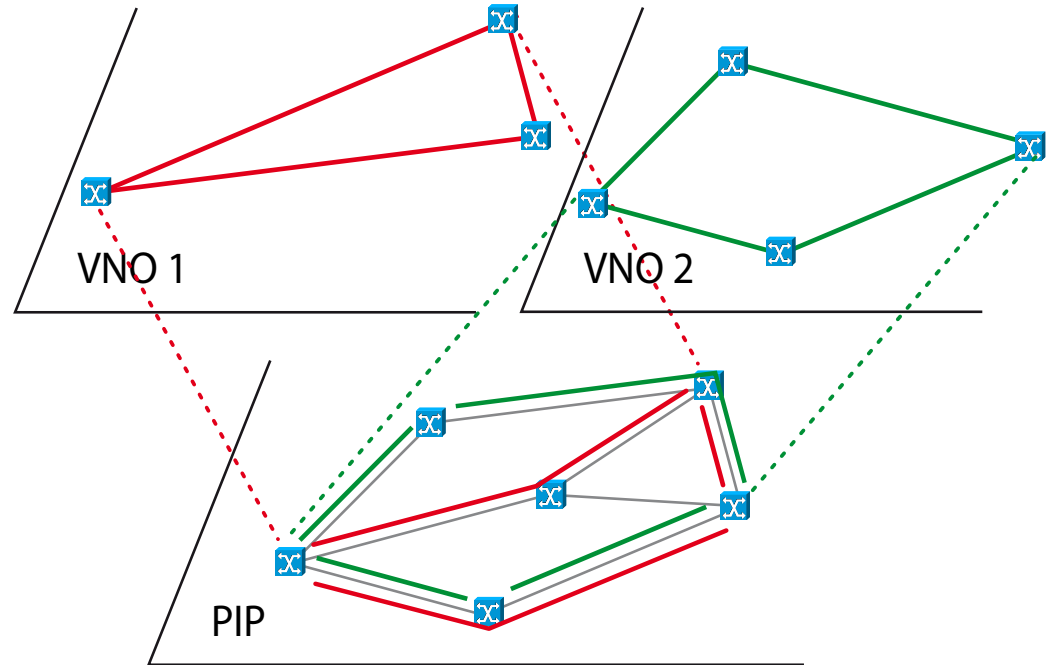


J. Buysse, K. Georgakilas, A. Tzanakaki, M. De Leenheer, B. Dhoedt and C. Develder, "Energy-efficient resource provisioning algorithms for optical clouds", IEEE/OSA J. Opt. Commun. Netw., Vol. 5, No. 3, Mar 2013, pp. 226-239. doi:10.1364/JOCN.5.000226.

Network virtualization

Physical network is logically partitioned in isolated virtual networks

- **Virtual Network Operators (VNO)** operate logically separated networks
- **Physical Infrastructure Providers (PIP)** have full control over infrastructure (fibers, OXCs)



J.A. García-Espín, et al., "Logical Infrastructure Composition Layer: the GEYSERS holistic approach for infrastructure virtualisation", in Proc. TERENA Networking Conference (TNC 2012), Reykjavík, Iceland, 21-24 May 2012.

An iterative network + server dimensioning approach and the impact of scheduling

C. Develder, B. Mukherjee, B. Dhoedt and P. Demeester, "*On dimensioning optical Grids and the impact of scheduling*", Photonic Netw. Commun., Vol. 17, No. 3, Jun. 2009, pp. 255-265. doi:10.1007/s11107-008-0160-z

Problem Statement

■ Given:

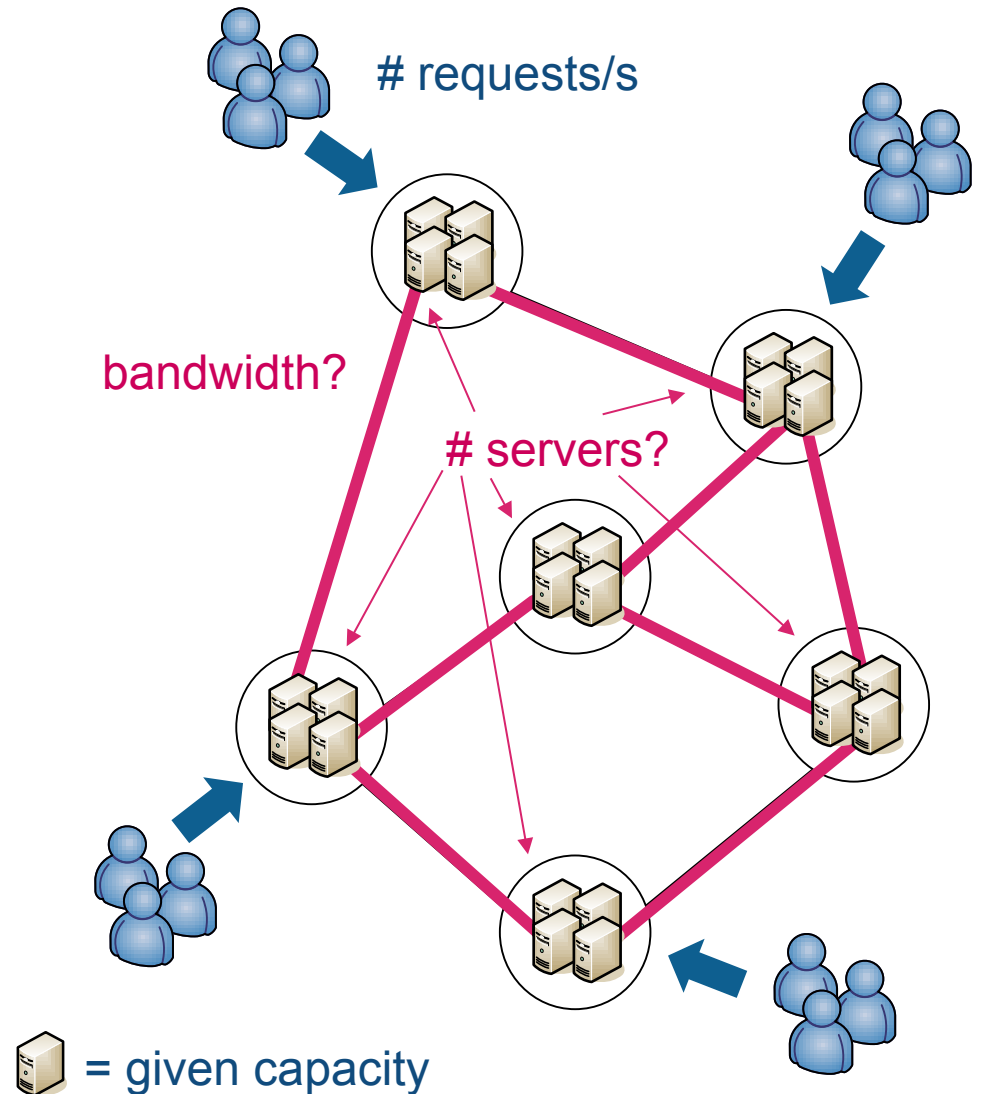
- Network topology
- Request arrival process
- Requested processing capacity
- Target maximum loss rate

■ Find

- Locations of servers,
- Amount of servers,
- Amount of link bandwidth

■ While

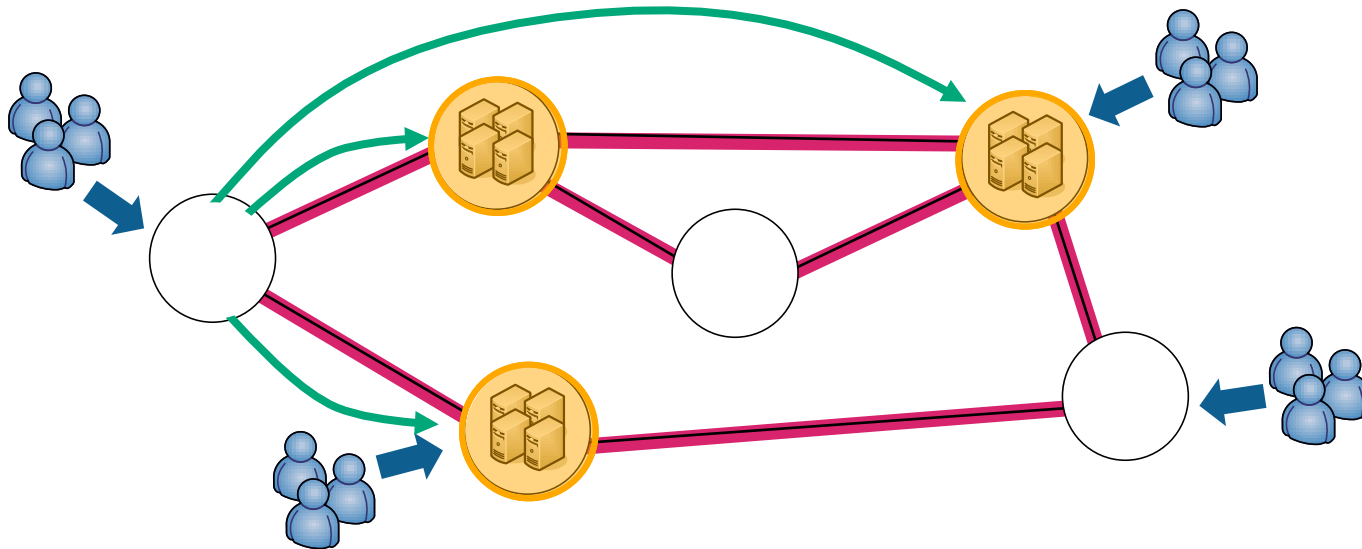
- Meeting max. loss
- Minimizing network capacity



Solution

■ Phased approach

- ① Determine K server locations (approx., ILP)
- ② Determine server capacity (analytical, ErlangB)
- ③ Determine inter-site bandwidths (simulation)
- ④ Dimension link bandwidths (= number of wavelengths)



Step 1: Server locations

- Given:

- Job arrivals at each site
- Each source S site sends all its requests to a single destination D
(simplifying assumption!)
- Shortest path routing is used

- Find:

- K server locations, minimizing total amount of used link bandwidth

 \approx K-means clustering problem

Heuristic: k-means clustering

■ Algorithm*:

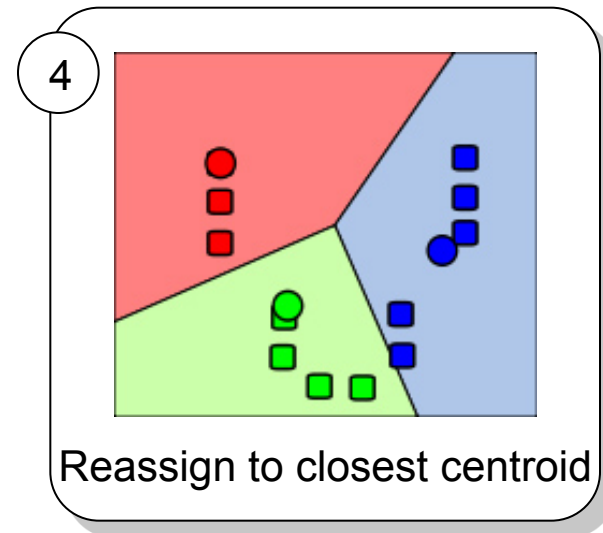
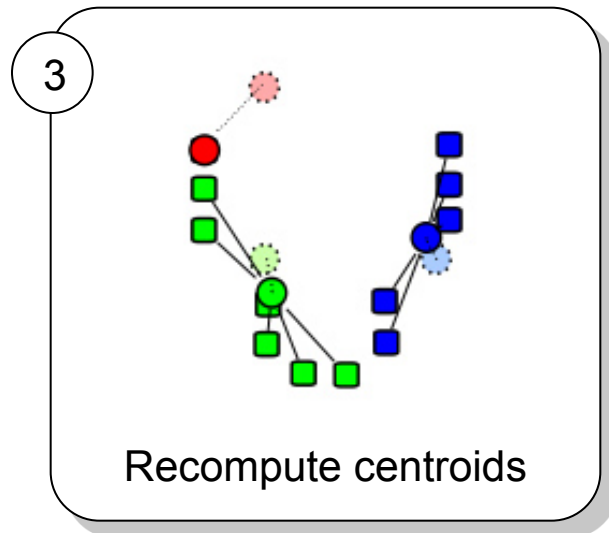
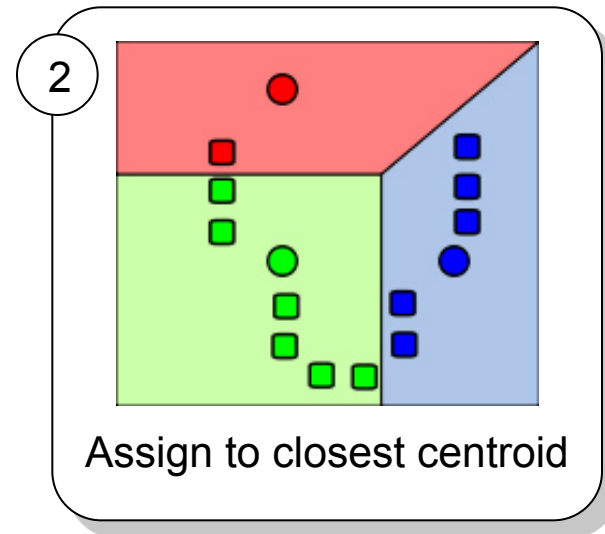
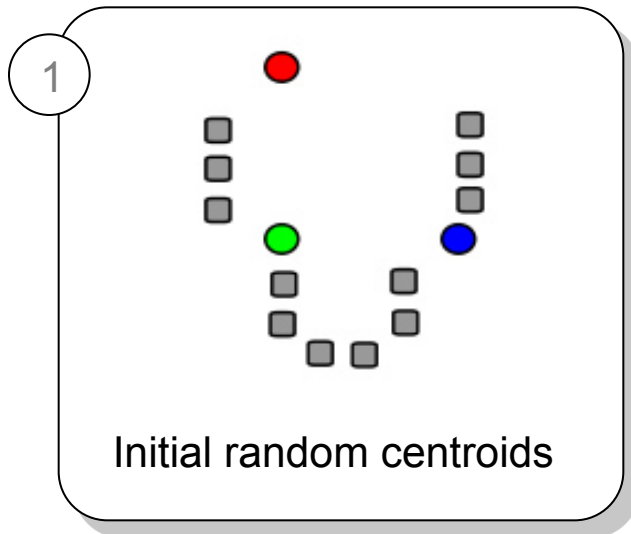
- (1) Choose K points, representing initial group centroids
- (2) Form clusters: assign each object to closest centroid
- (3) Recalculate K centroids positions within their cluster
- (4) Repeat Steps 2-3 until the centroids no longer move

■ Our problem:

- k-medoid (i.e., centers are actual data points)
- “closest” = shortest path distance

*: J. B. MacQueen (1967): “Some Methods for classification and Analysis of Multivariate Observations”, *Proc. 5th Berkeley Symp. on Mathematical Statistics and Probability*, Berkeley, University of California Press, 1:281-297

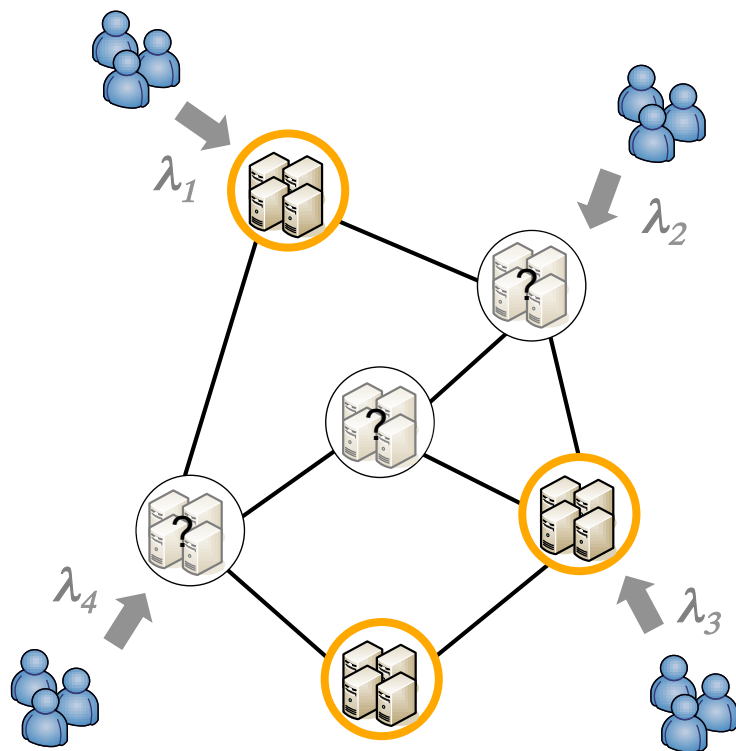
Heuristic: k-means clustering



Step 1: Finding the K “best” locations

Binary variables:

- $t_v = 1$ iff site v is server location
- $f_{vv'} = 1$ iff request from source v is directed to v'



Constants:

- $h_{vv'}$ = cost for sending 1 unit request from source v to server site v'
- Δ_v = number of unit requests from source v

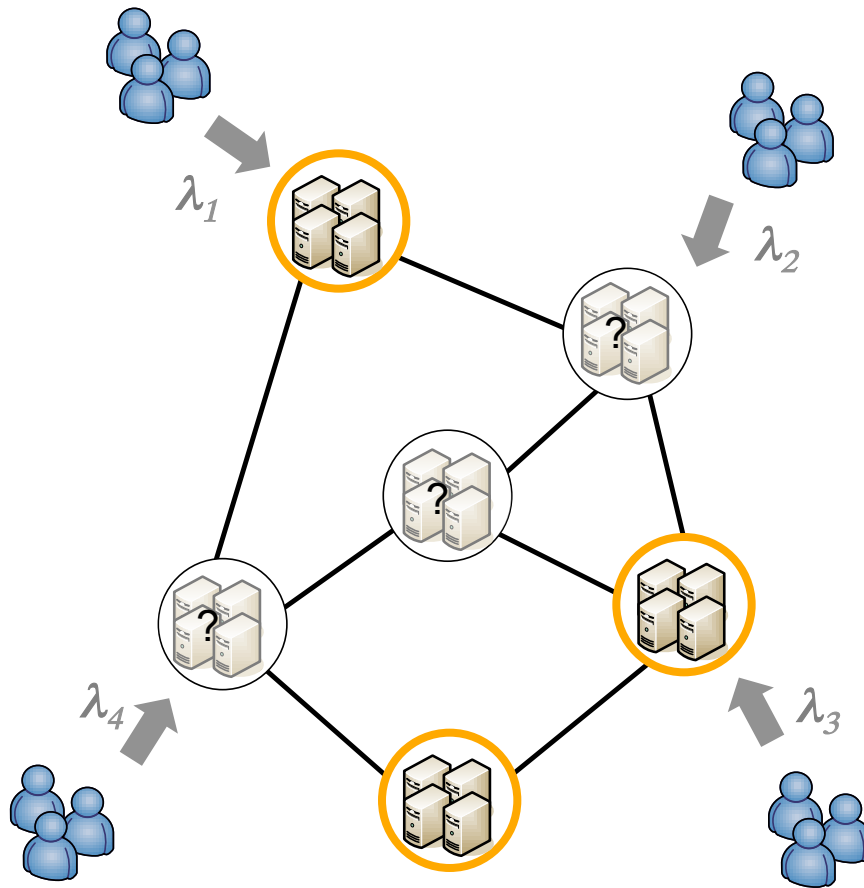
$$\min \sum_v \sum_{v'} \Delta_v \cdot h_{vv'} \cdot f_{vv'}$$

subject to

$$\begin{cases} \sum_v t_v = K \\ \sum_{v'} f_{vv'} = 1 \quad \forall v \\ f_{vv'} \leq t_{v'} \quad \forall v, v' \end{cases}$$

Step 1: Server locations

- Find K server locations:



$$\min \sum_i \sum_j \lambda_i \cdot H_{ij} \cdot S_{ij}$$

subject to

$$\begin{cases} \sum_j T_j = K \\ \sum_j S_{ij} = 1 \quad \forall i \\ S_{ij} \leq T_j \quad \forall i, j \end{cases}$$

Step 1: ILP formulation

- Binary variables:

- $T_j = 1$ iff site j is chosen as cluster center (i.e., server location)
- $S_{ij} = 1$ iff site i belongs to cluster with center j (i.e., sends traffic to center j)

- Constants:

- H_{ij} = distance from site i to site j
- λ_i = job arrival rate at site i

- Objective:

$$\min \sum_i \sum_j \lambda_i \cdot H_{ij} \cdot S_{ij}$$

- Constraints:

$$\sum_j T_j = K$$

K server sites

$$\sum_j S_{ij} = 1 \quad \forall i$$

Only send traffic
to 1 server site

$$S_{ij} \leq T_j \quad \forall i, j$$

Only send traffic
to centers

Step 2: Server capacities

- Find number of servers n :

- $$L = \text{ErlangB}(n, \lambda, \mu) = \frac{(\lambda/\mu)^n / n!}{\sum_{k=0}^n (\lambda/\mu)^k / k!}$$

- Distribution among server sites: 3 alternatives

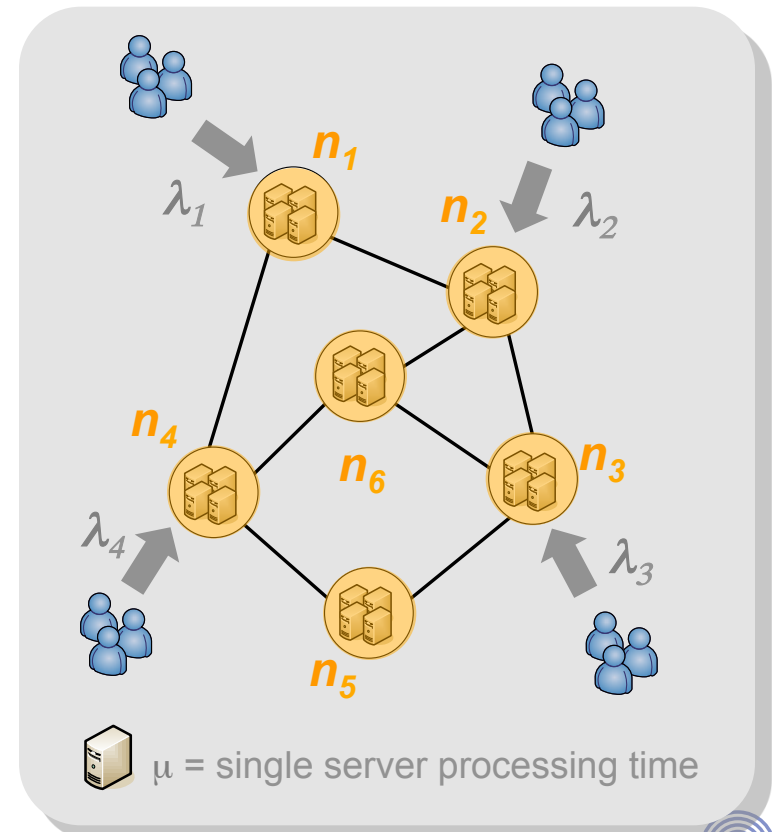
- unif**: uniformly distributed among all sites: $n_i = n/N$
- prop**: proportional to local arrival rate: $n_i = \lambda_i / (N \cdot \lambda)$
- lloss**: try to achieve the same (local) loss rate at each site: $n_i \sim n'_i$ with $L = \text{ErlangB}(n'_i, \lambda, \mu)$

L = target loss

λ = total arrival rate (all N sites)

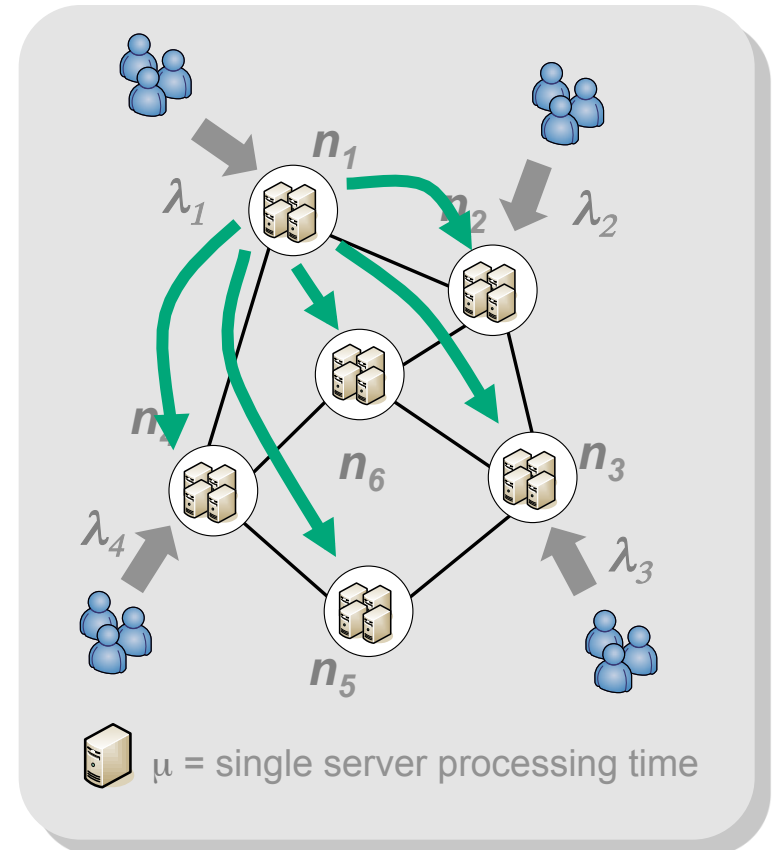
μ = single server processing time

n = total number of servers



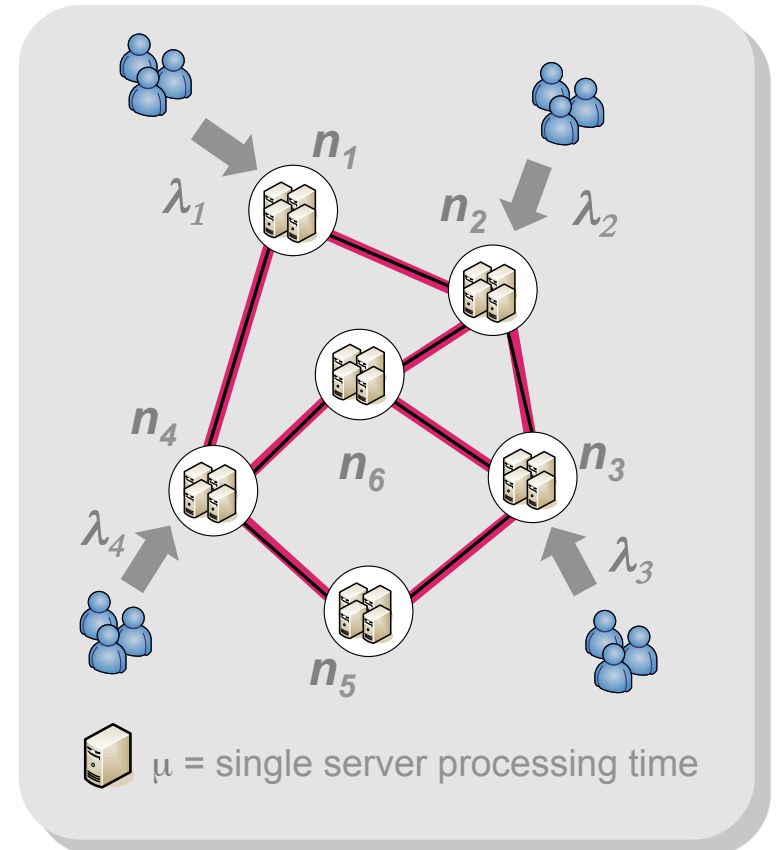
Step 3: Inter-site bandwidth

- Given:
 - Request arrivals
 - Site server capacities
- Find
 - Bandwidth exchanged between sites (i.e., amount of requests)
 - Scheduling alternatives: alwayst try local, if busy then
 - **SP**: shortest path (i.e., closest free server)
 - **rand**: randomly pick a free site
 - **mostfree**: choose site with most free servers



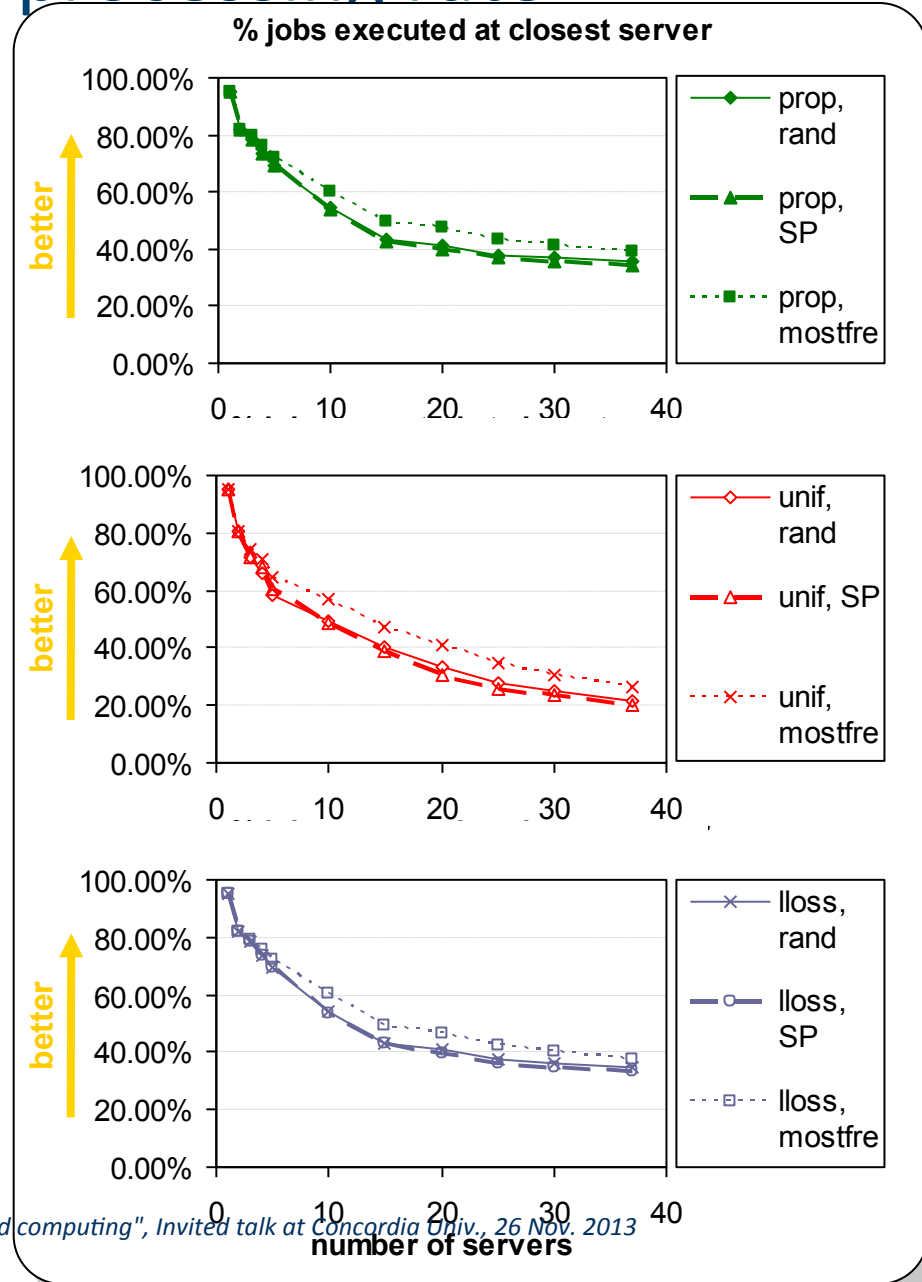
Step 4: Link dimensions

- Given:
 - Inter-site request arrivals (from step 2)
 - Shortest path routing (assumption)
- Find
 - Link bandwidth (amount of wavelengths)

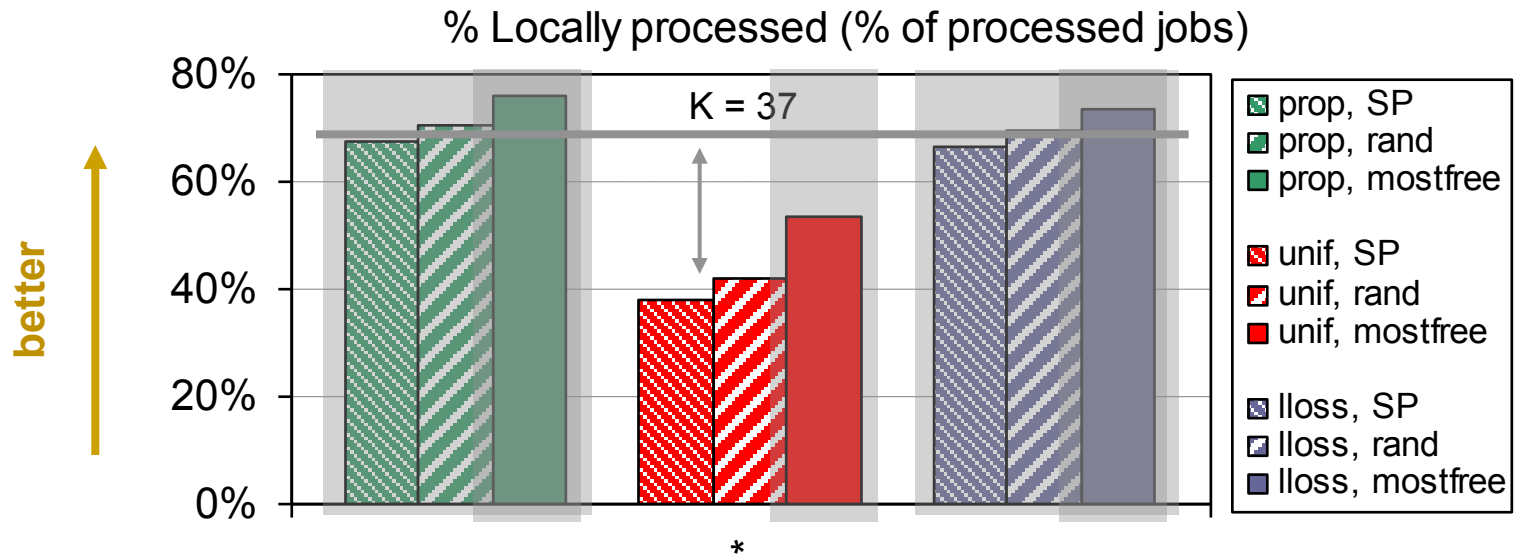


Case study results: 'Local' processing rate

- Influence of *# server sites*:
 - % processed at closest server decreases for increasing # servers
- Influence of *scheduling*:
 - mostfree scheduling achieves highest % at local site
- Influence of *server distribution*:
 - Non-uniform server distribution (**prop**, **lloss**) leads to significant increase in % at closest site



Case study results: 'Local' processing rate



- Server distribution:
 - **unif**: uniformly distributed
 - **prop**: ~ local arrival rate
 - **lloss**: ~ same (local) loss rate
- Scheduling: local first, if busy then...
 - ▨ **SP**: shortest path
 - ▨ **rand**: randomly pick a free site
 - ▨ **mostfree**: site with most free servers
- Conclusions:
 - **mostfree** achieves highest local processing
 - Intelligent server placement (prop, lloss) achieves higher local processing

Case study results: Link bandwidths

■ Influence of # server sites:

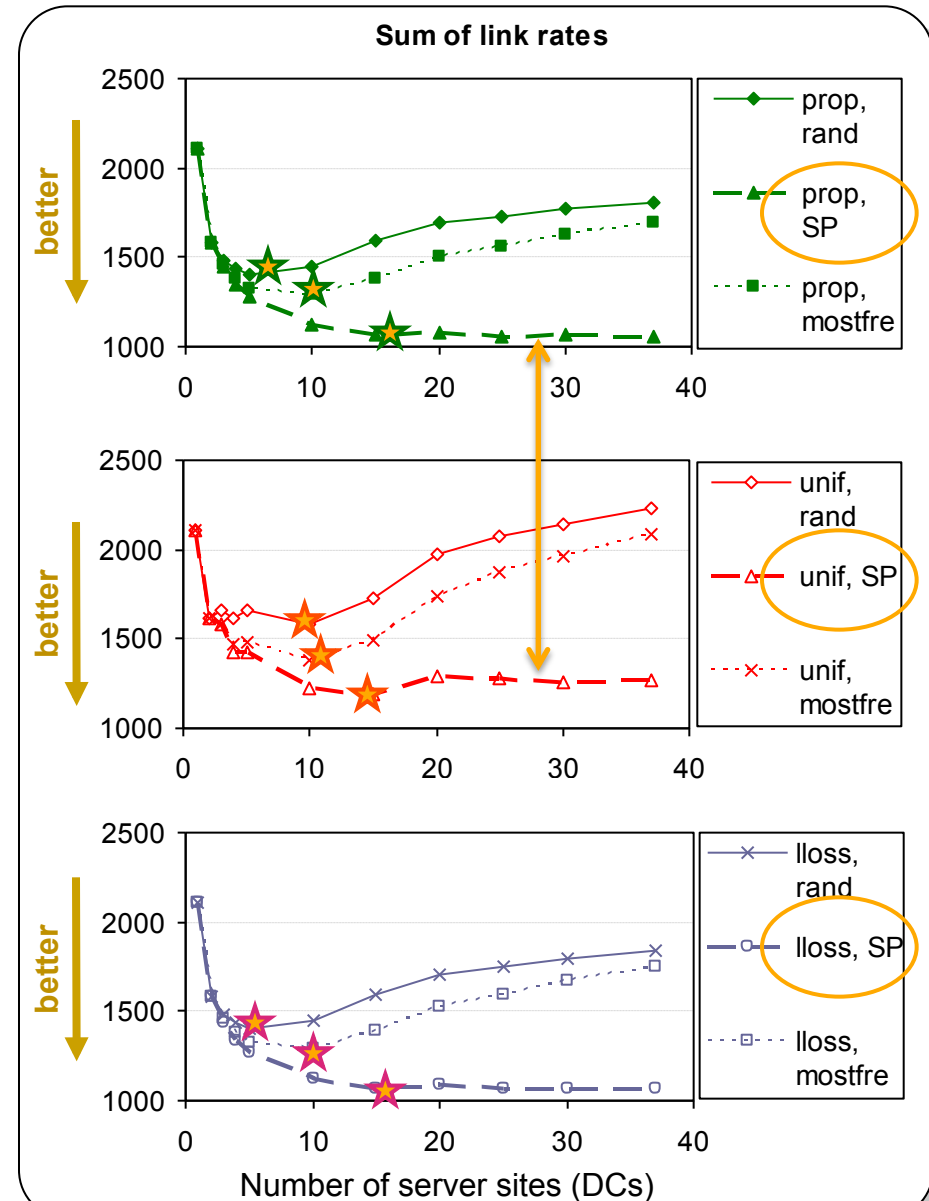
- There is an “optimal” value, depending on the scheduling algorithm & server distribution

■ Influence of scheduling:

- SP scheduling obviously leads to lowest total link bandwidth

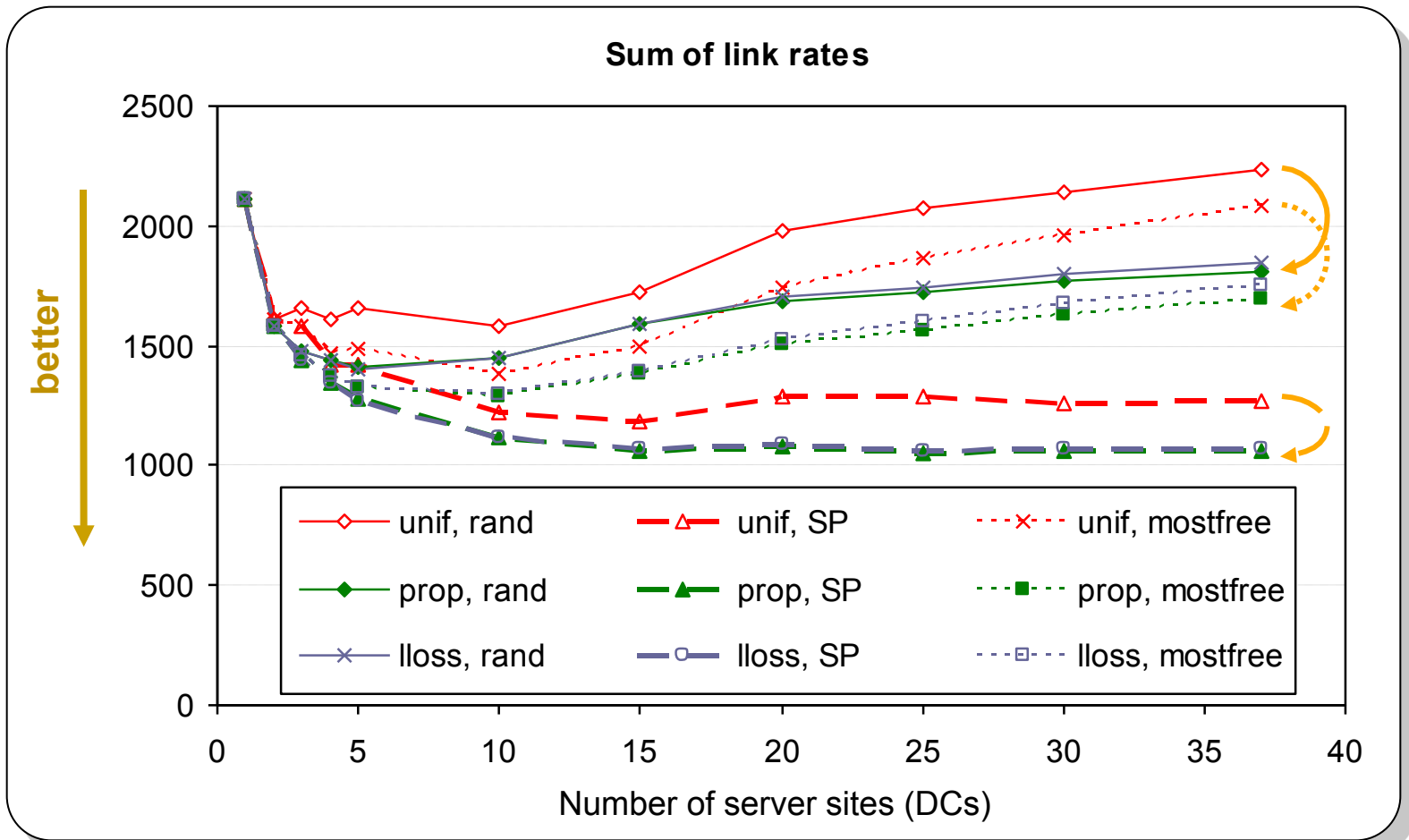
■ Influence of server distribution:

- Non-uniform server distribution (prop, lloss) leads to significant BW reduction



Case study results: Link bandwidths

- Influence of server distribution:
 - Non-uniform distribution leads to significant BW reduction



Conclusions wrt dimensioning

- Proposal of dimensioning approach
 - Sequential approach (first server locations & dimensions, then network)
 - Combination of analytics and simulation

- Comparison of site dimensioning and scheduling alternatives
 - Dimensioning: intelligent server placement allows higher local processing
 - Scheduling: maximizing “local” processing may come at link bandwidth price

Exploiting anycast for resilience

C. Develder, J. Buysse, B. Dhoedt and B. Jaumard, "*Joint dimensioning of server and network infrastructure for resilient optical grids/clouds*", IEEE/ACM Trans. Netw., Vol. PP, Oct. 2013, pp. 1-16.
doi:10.1109/TNET.2013.2283924

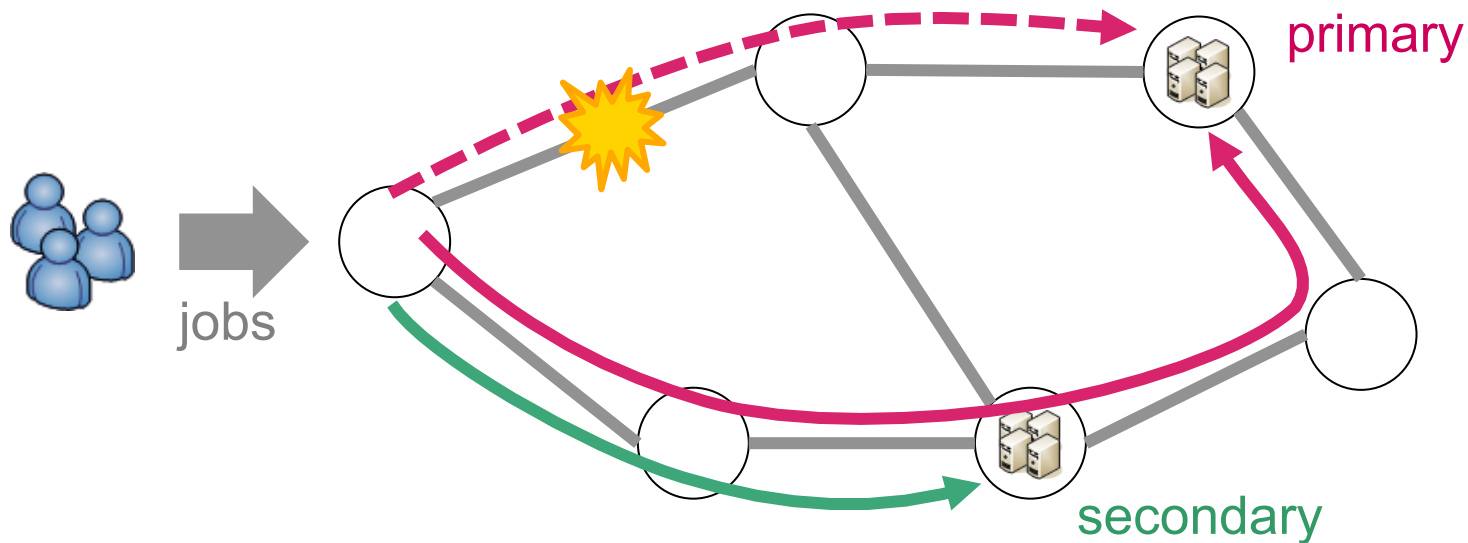
Exploiting relocation for resilience?

- **Goal:**

Achieve (network) dimensioning, for resiliency against failures

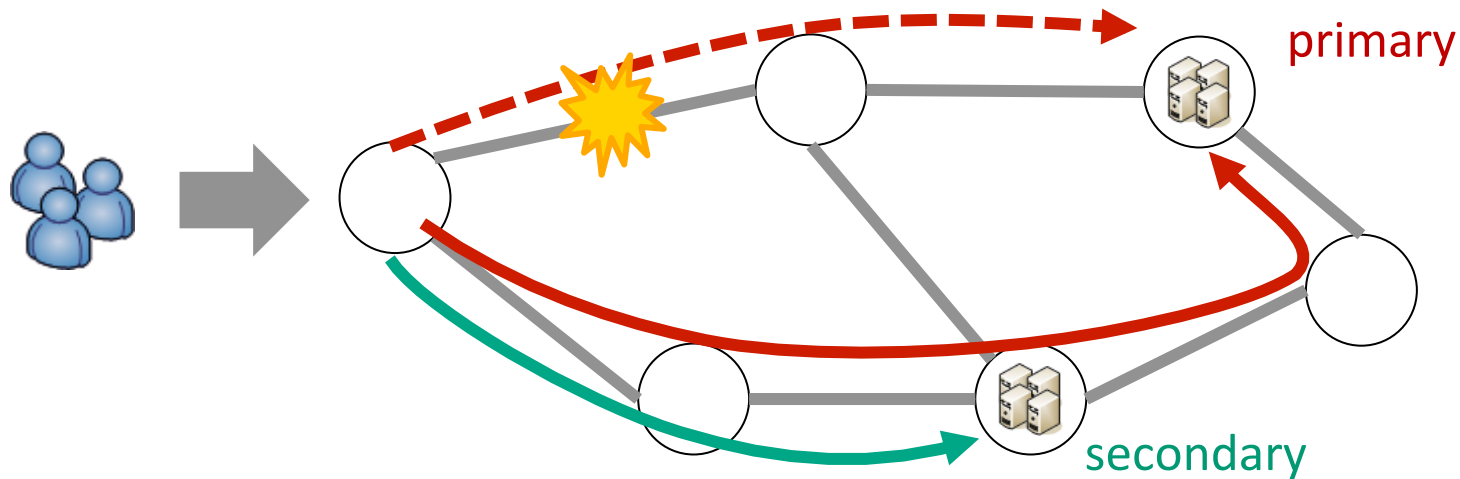
- **Idea:**

Exploit anycast principle: allow rerouting jobs to other destination



Exploiting relocation

- Dimension optical grid/cloud so that it is resilient against failures
- Exploit anycast principle: allow rerouting to other destinations



J. Buysse, M. De Leenheer, B. Dhoedt and C. Develder, "Providing resiliency for optical grids by exploiting relocation: A dimensioning study based on ILP", *Comput. Commun.*, Vol. 34, No. 12, Aug. 2011.

A. Shaikh, J. Buysse, B. Jaumard and C. Develder, "Anycast routing for survivable optical grids: scalable solution methods and the impact of relocation", *IEEE/OSA J. Opt. Commun. Netw.*, Vol. 3, No. 9, Sep. 2011.

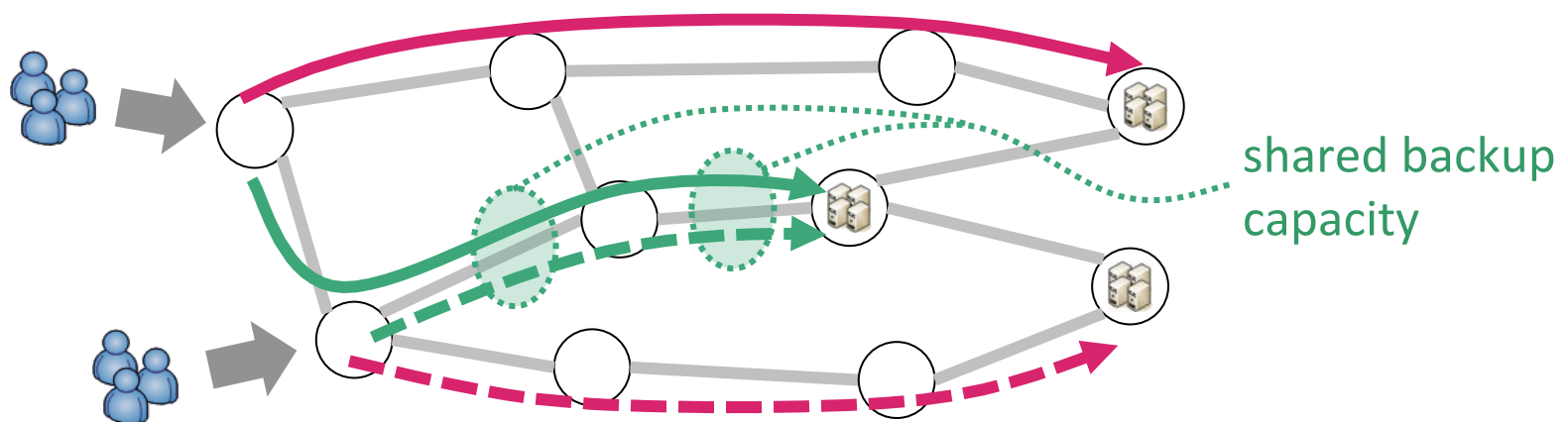
Problem statement

■ Given:

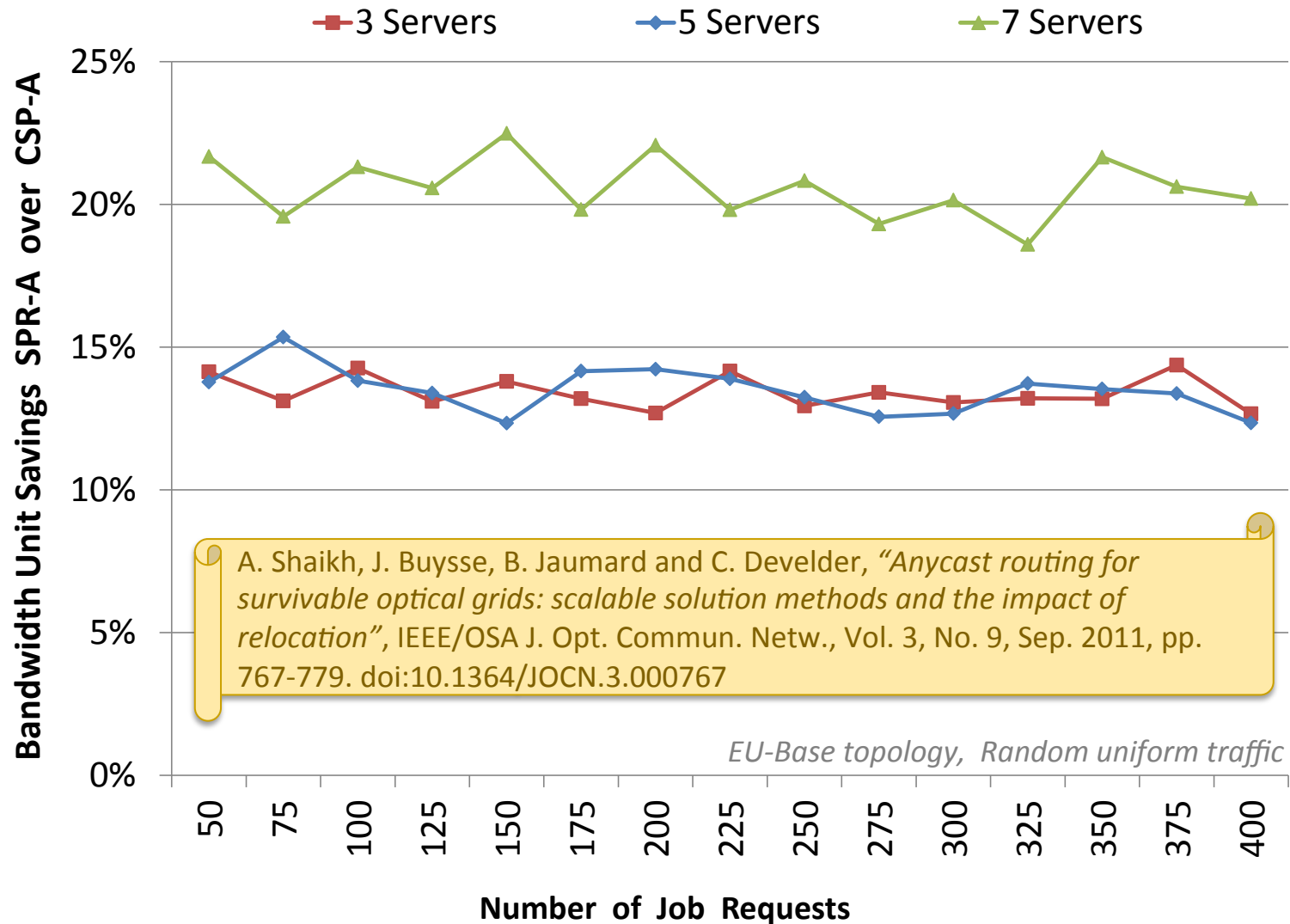
- Topology w/ K chosen server locations (i.e., data centers)
- Traffic “vector”: sources + volume of traffic

■ Find:

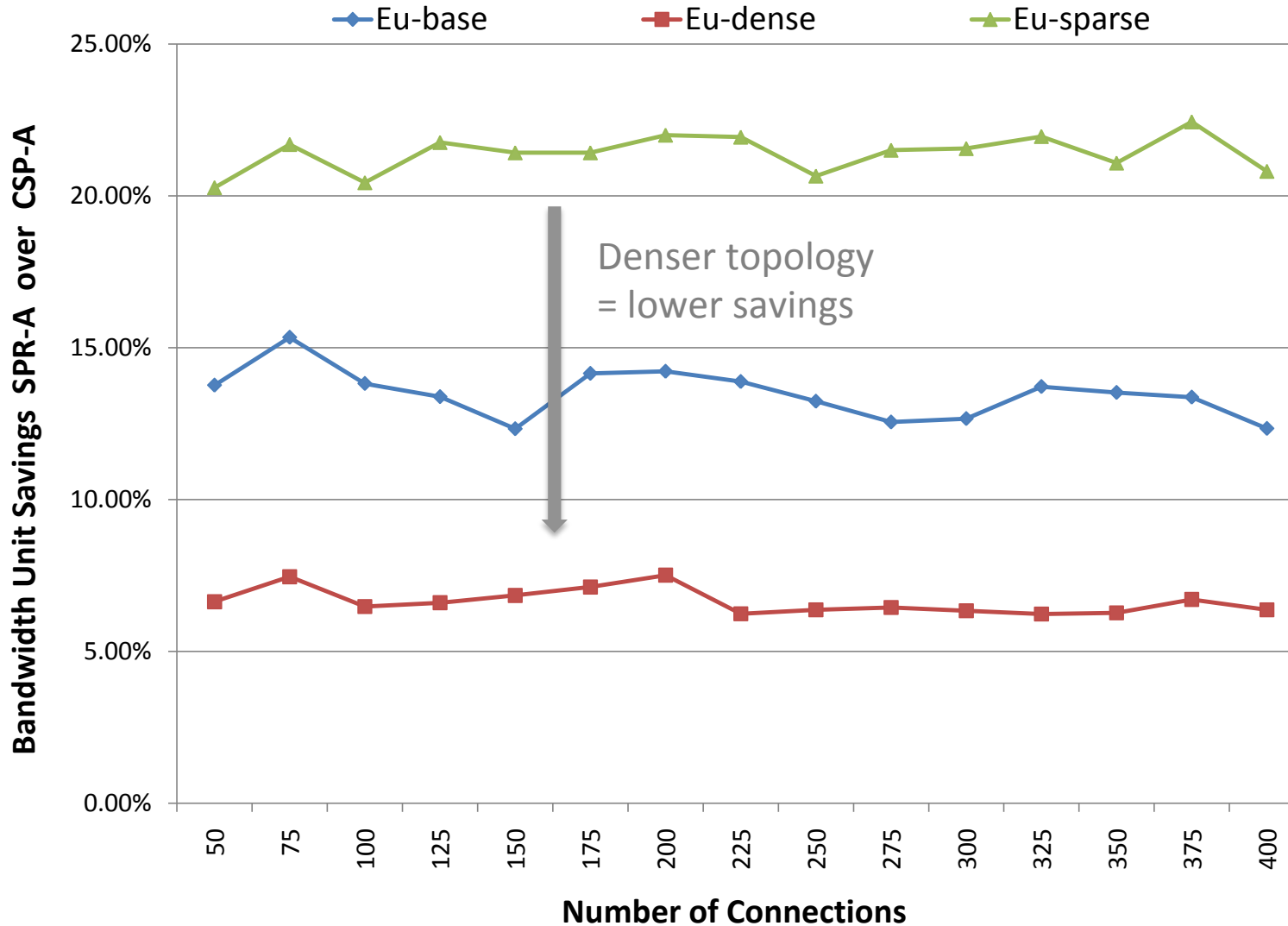
- Primary and secondary routes for each demand, while
- Minimizing resource dimensions (in terms of # wavelengths/link, # servers)
- For the case of shared protection (+ relocation) against single failures



Initial results on network savings



Initial results on network savings



Problem statement

Given

- Topology (sources, candidate data center locations, OXCs)
- Demand (for given sources)
- **Survivability** requirements (e.g., link and/or node failures)

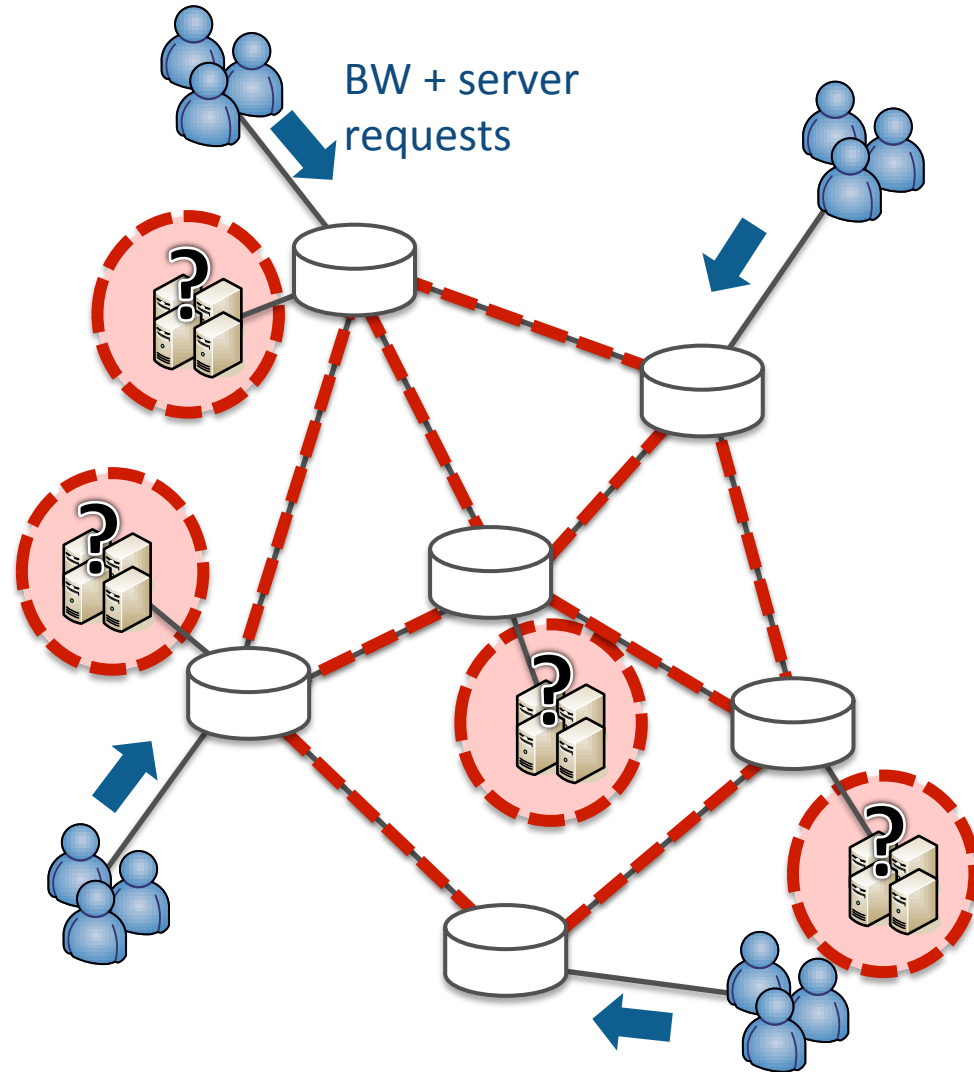
Find

Shared protection

- K locations (chosen from candidate data center locations)
- Destination sites and routes
- Network and server capacity

Such that

- Network and server resources are minimized

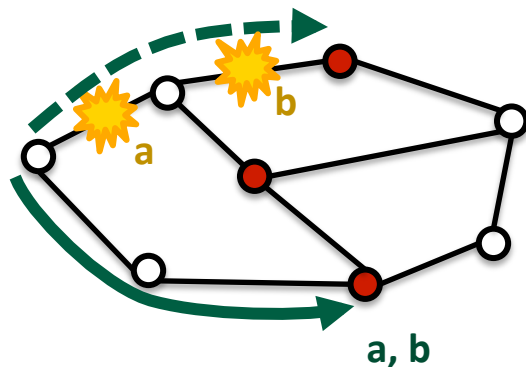


Solution approach

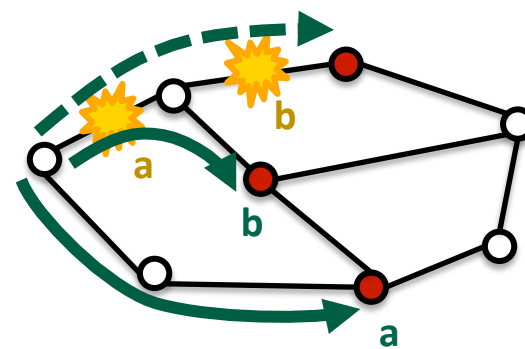
Step 1: Find the K best data center locations

Step 2: Find the primary/secondary destinations + paths towards them

Failure-Independent (FID)
rerouting
=> Column generation



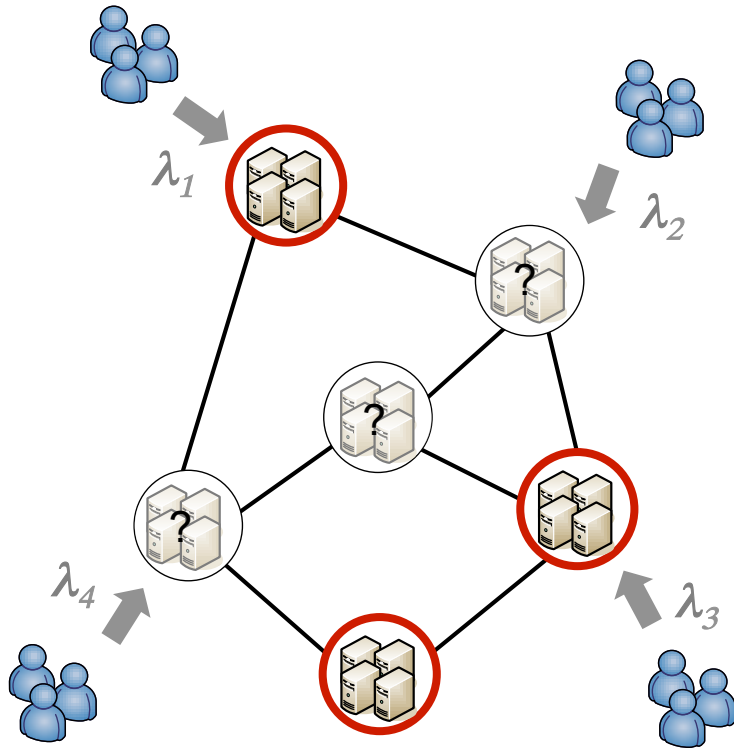
Failure-Dependent (FD)
rerouting
=> Single ILP



Step 1: Finding the K “best” locations

Binary variables:

- $t_v = 1$ iff site v is server location
- $f_{vv'v''} = 1$ iff request from source v is directed to primary v' , backup v''



Constants:

- $h_{vv'v''}$ = cost for sending 1 unit request from source v to server sites v', v''
- Δ_v = number of unit requests from source v

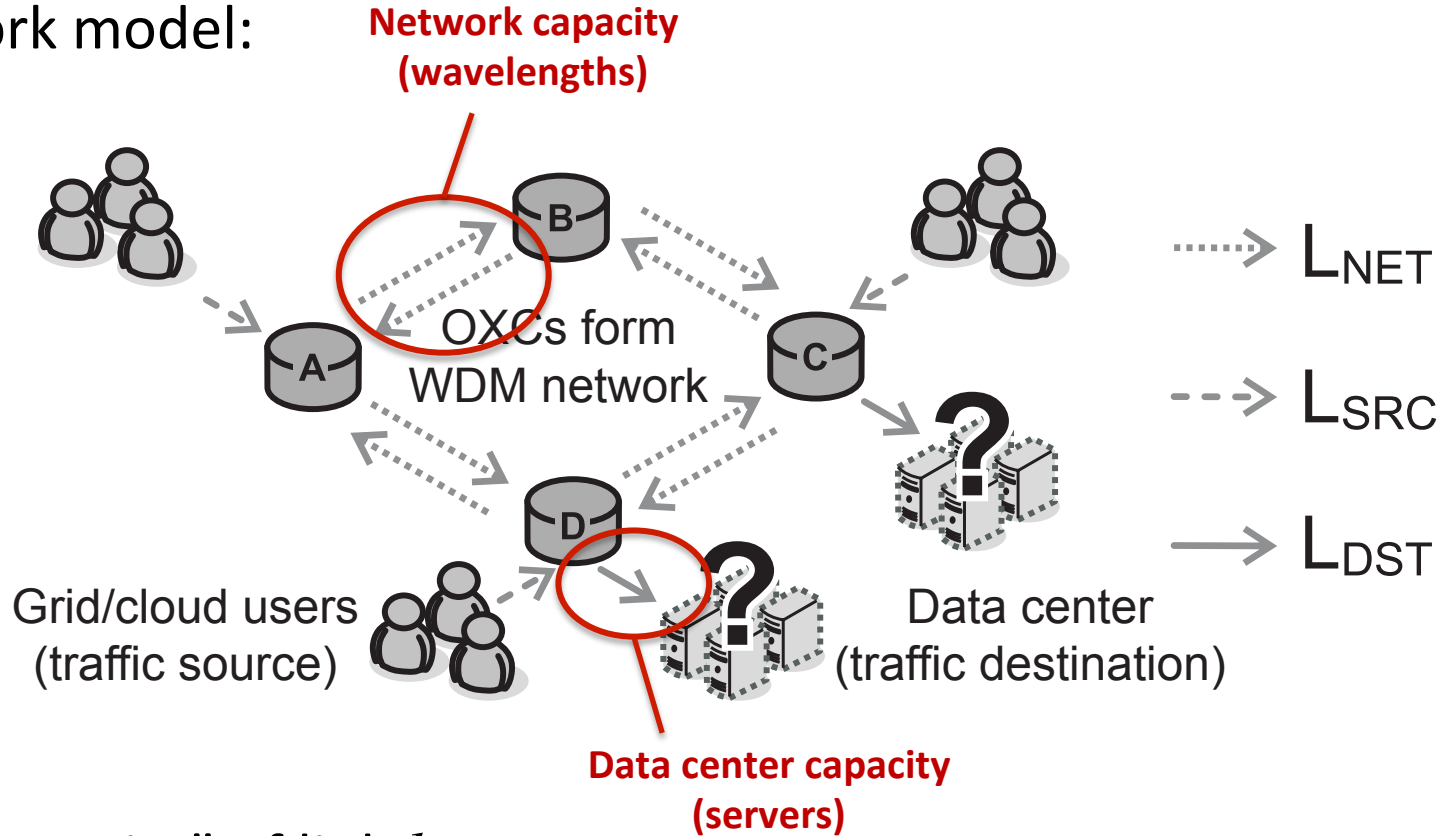
$$\min \sum_v \sum_{v'} \Delta_v \cdot h_{vv'v''} \cdot f_{vv'v''}$$

subject to

$$\left\{ \begin{array}{l} \sum_v t_v = K \\ \sum_{v'} \sum_{v''} f_{vv'v''} = 1 \quad \forall v \\ f_{vv'v''} \leq t_{v'} \quad \forall v, v', v'' \\ f_{vv'v''} \leq t_{v''} \quad \forall v, v', v'' \end{array} \right.$$

Step 2: Find destinations and routes towards them

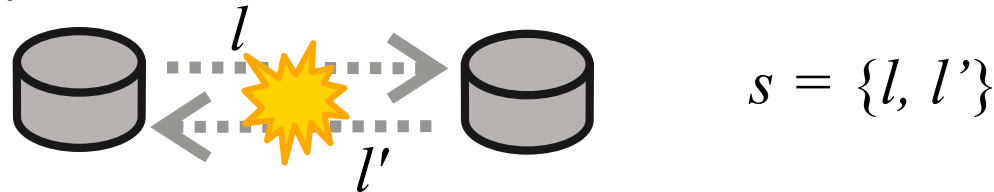
- Network model:



- $w_l =$ "capacity" of link l
- Capacity = wavelengths for NET links, servers for DST links!

Step 2: Find destinations and routes towards them

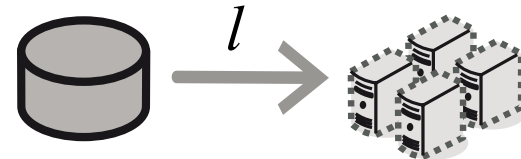
- Failure: modeled as SRLG = set of links s that simultaneously fail
- Single link failure:



- Single server failure: $1:N$ protection [= add 1 for each case single one out of N fails]

- No relocation:

- Let x = number of servers under working conditions

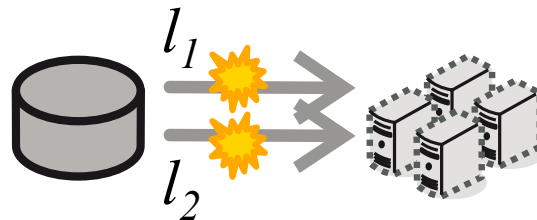


$$w_l \geq \rho_l \cdot x$$

$$\rho_l = 1 + 1/N$$

- Then we need $\lceil (1 + 1/N) \cdot x \rceil$ servers

- Relocation: consider $(1+N)$ parallel links, at most 1 fails



$$s_1 = \{l_1\}$$

$$s_2 = \{l_2\}$$

Step 2: Find destinations and routes towards them

- Failure dependent (FD) rerouting => Single ILP

- Variables:

- p_{vls} : number of unit demands with source v that cross link l under failure s
- w_l : capacity on link l

- Objective:

$$\min \left(\sum_{l \in L_{NET}} w_l + \alpha \sum_{l \in L_{DST}} w_l \right)$$

Network capacity (wavelengths)
Data center capacity (servers)

- Constraints:

- p_{vls} : flow constraints + don't use failing links when protecting against s
- w_l : count capacity

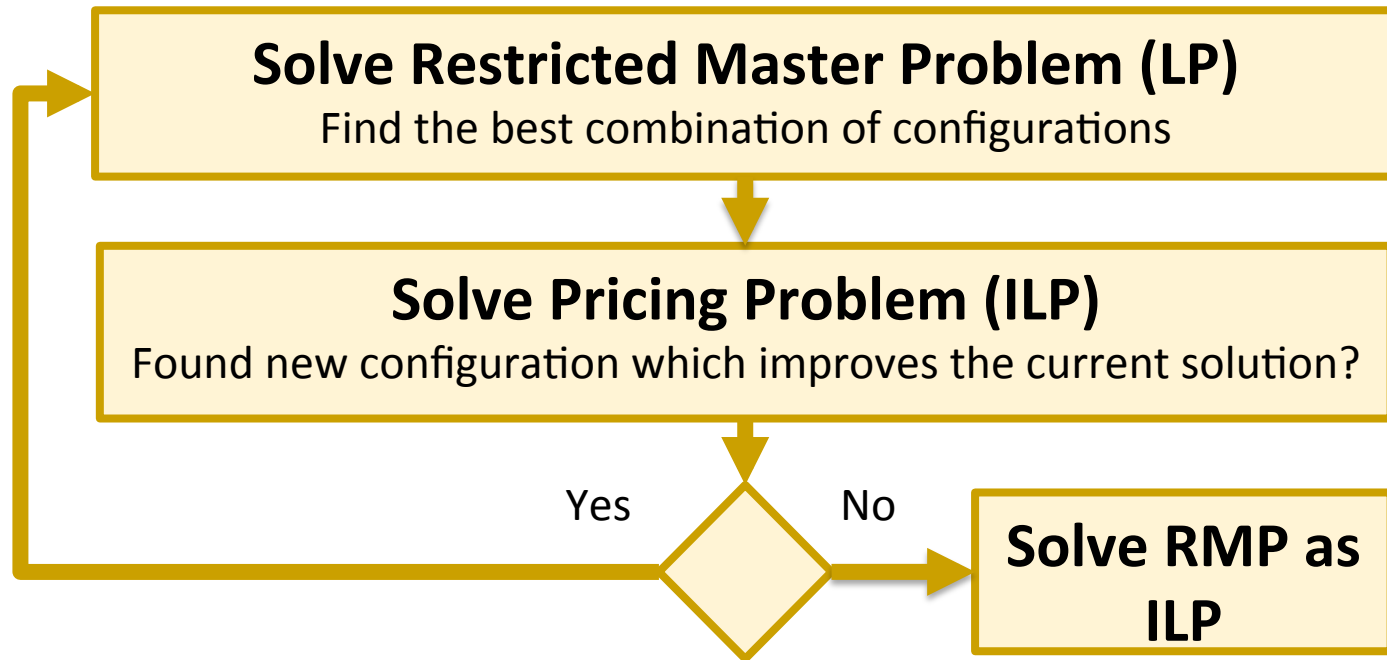
1 for network link

1+1/N for server link, in case of relocation

$$w_l \geq \rho_l \cdot \sum_{v \in V_{SRC}} p_{vls} \quad \forall s \in S$$

Step 2: Find destinations and routes towards them

- **Failure-independent (FID) rerouting** => Column generation:
 - Assume: given “configurations” = combination of working and backup paths
 - Restricted Master Problem (RMP) finds best combination of configurations
 - Pricing Problem (PP) finds new configuration that can reduce cost



Finding initial configurations for RMP of Step 2

- Configuration associated with a particular source

- Working path to primary destination
- Backup path to secondary destination

- Algorithm:

Suurballe's algorithm

- Find shortest 2 disjoint paths from s to candidate destination(s): configuration c

- Create new configurations that share backup links:

Sharing of backup wavelengths

- Remove working links of c
- For all configurations c' with primary disjoint from c : set backup link weights to 0
- Find backup path b'' as shortest path to candidate destination in reduced graph
- Create c'' : primary of c , and b'' as secondary

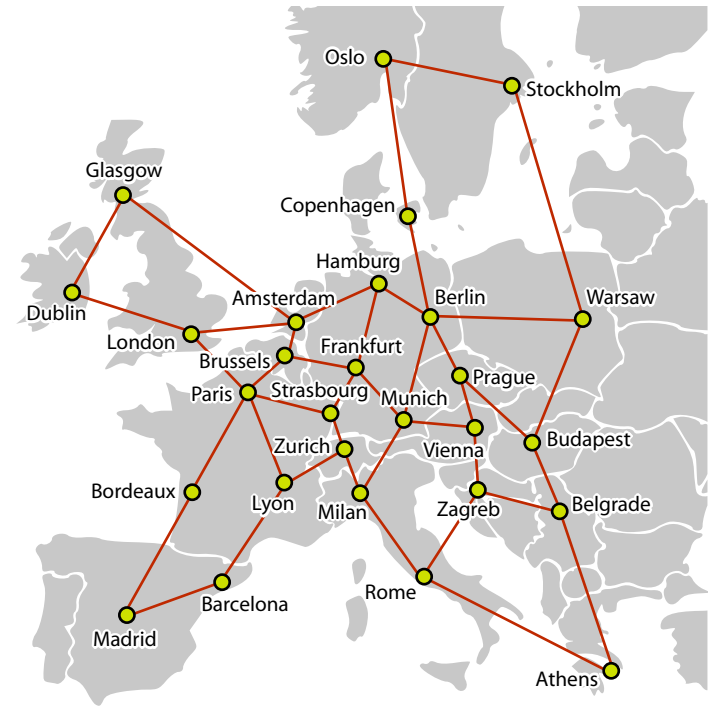
Case study set-up

■ Topology

- European network
- 28 nodes and 41 bidirectional links

■ Demand

- Randomly generated requests (10-350)
- 10 instances for each number of requests



■ Four scenarios:

No relocation

Exploiting relocation

Single link failures:

1L, NoReloc



1L, Reloc


Single failures of either link or server:


1LSN, NoReloc

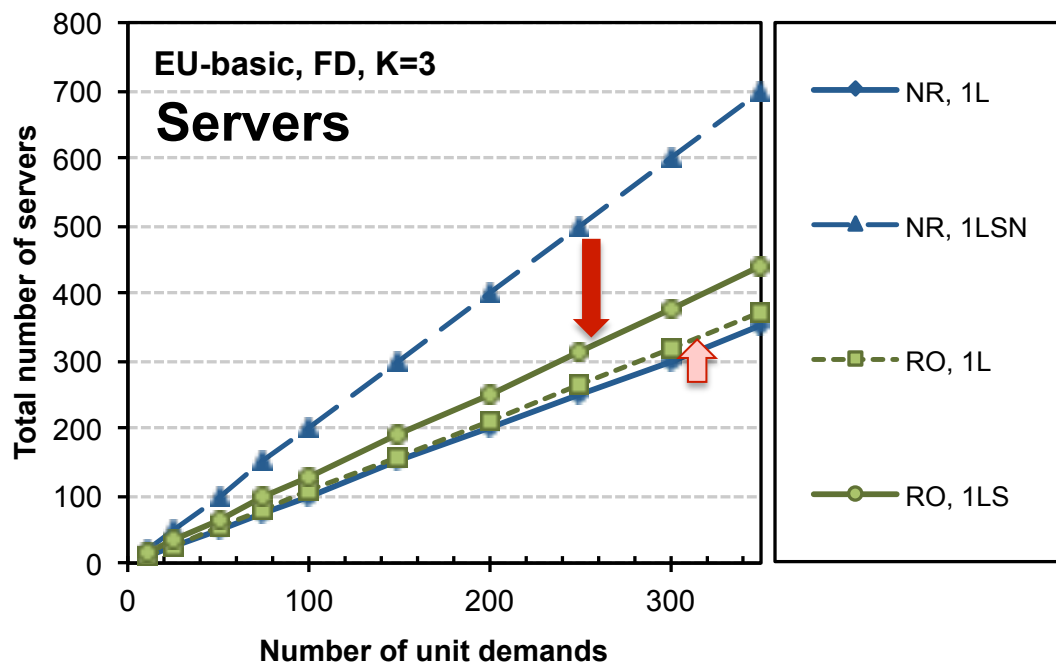
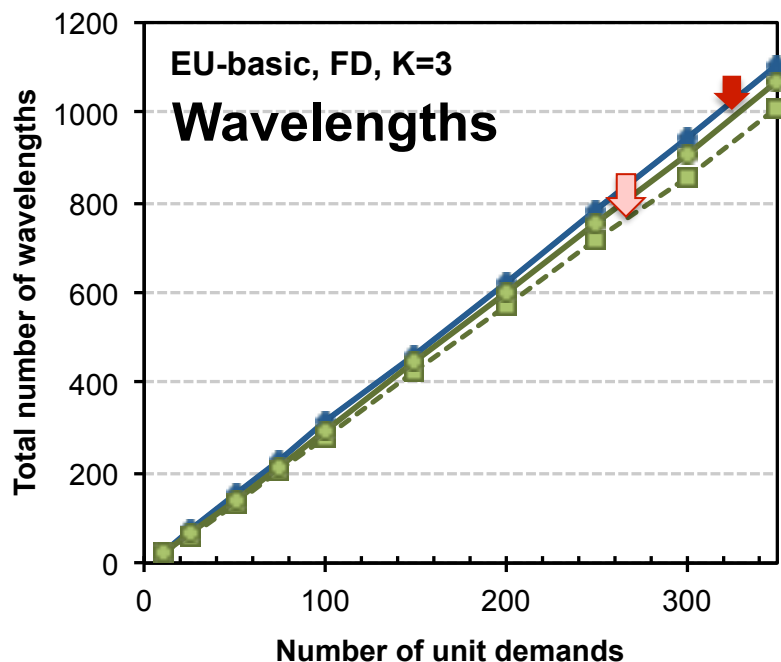


1LS, Reloc


The impact of relocation


- Single Link failures (1L): 
 - Reduction of backup wavelengths
 - Slight increase in server capacity

- Single link/server failure (1LS) 
 - Reduction of backup wavelengths
 - Fewer servers than 1:N server protection (N=1)

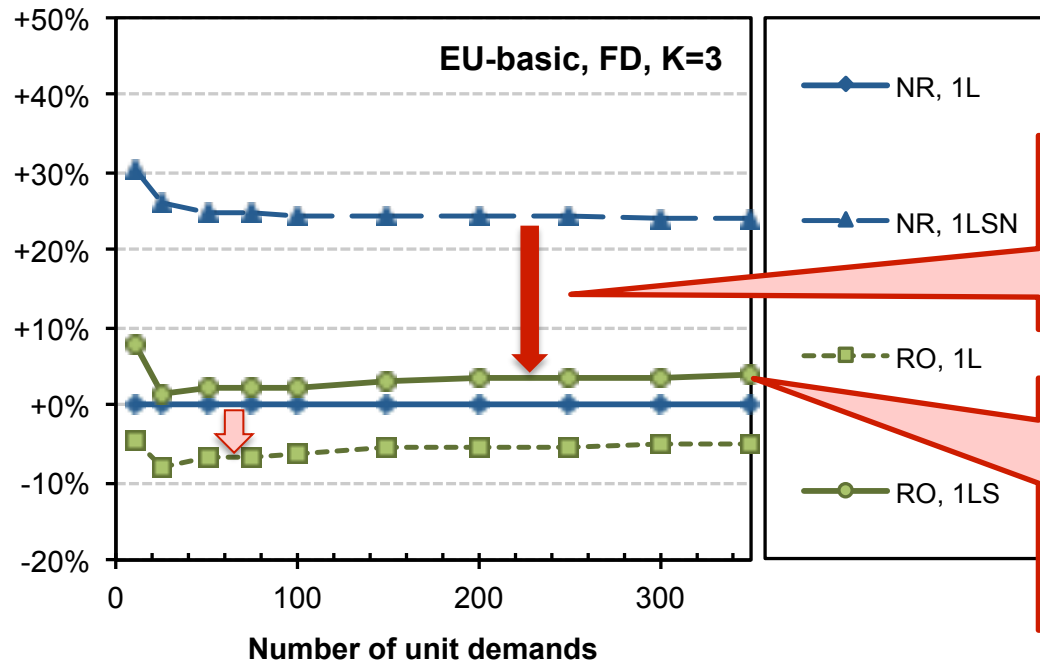


The impact of relocation

- Single Link failures (1L): 
 - Reduction of backup wavelengths
 - Slight increase in server capacity

- Single link/server failure (1LS) 
 - Reduction of backup wavelengths
 - Fewer servers than 1:N server protection (N=1)

Total cost relative to the NR, 1L case

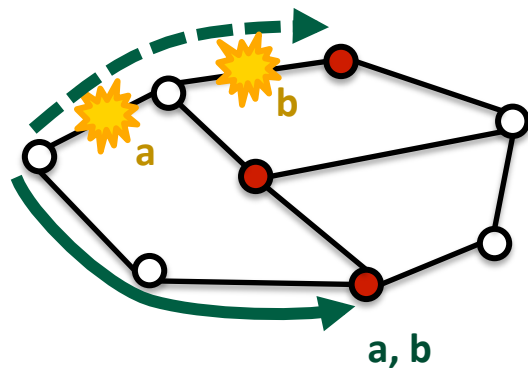


Mainly reduction of servers ($1+1/K$ vs $1+1/N$)

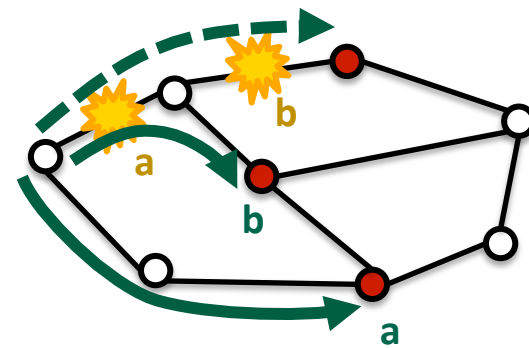
Protection against 1LS failures at almost same cost as 1L without relocation

Failure dependent rerouting? (FD vs FID)

Failure-Independent (FID)
rerouting

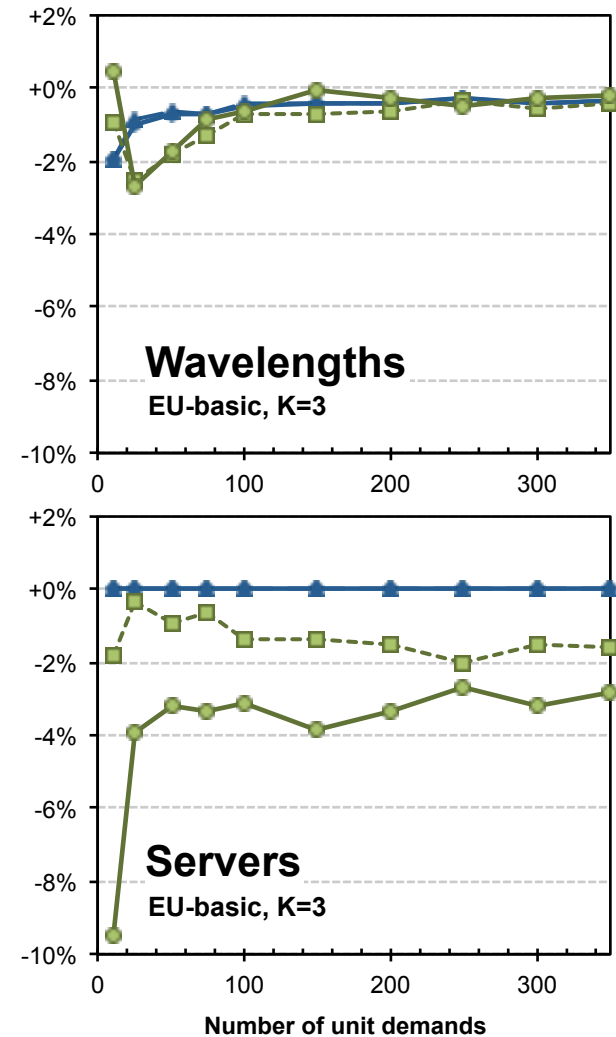
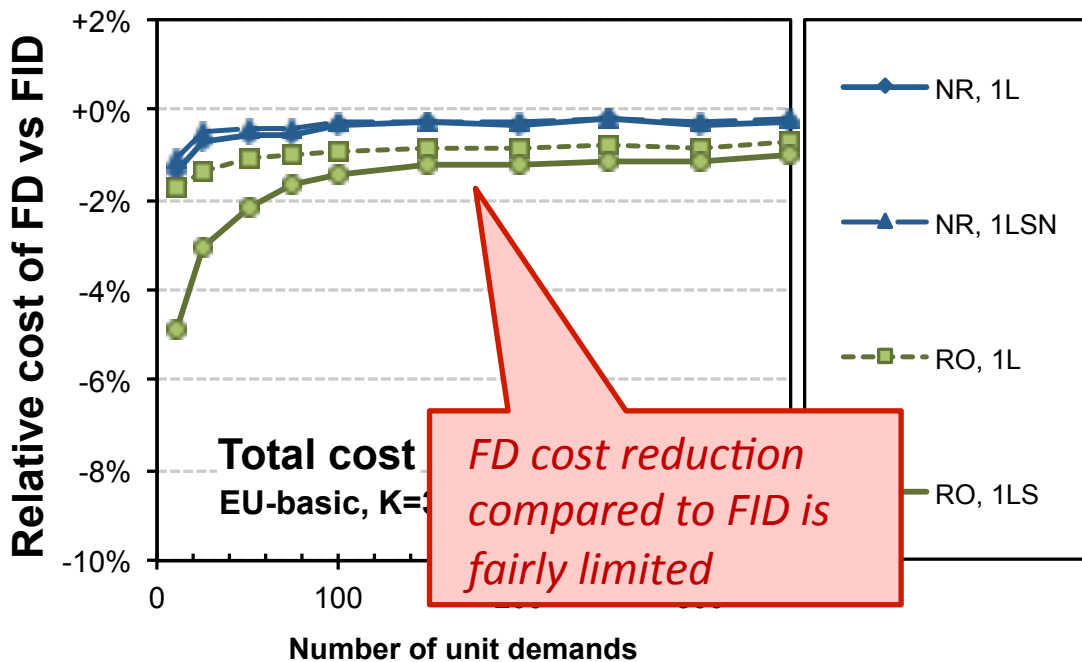


Failure-Dependent (FD)
rerouting



Failure dependent rerouting? (FD vs FID)

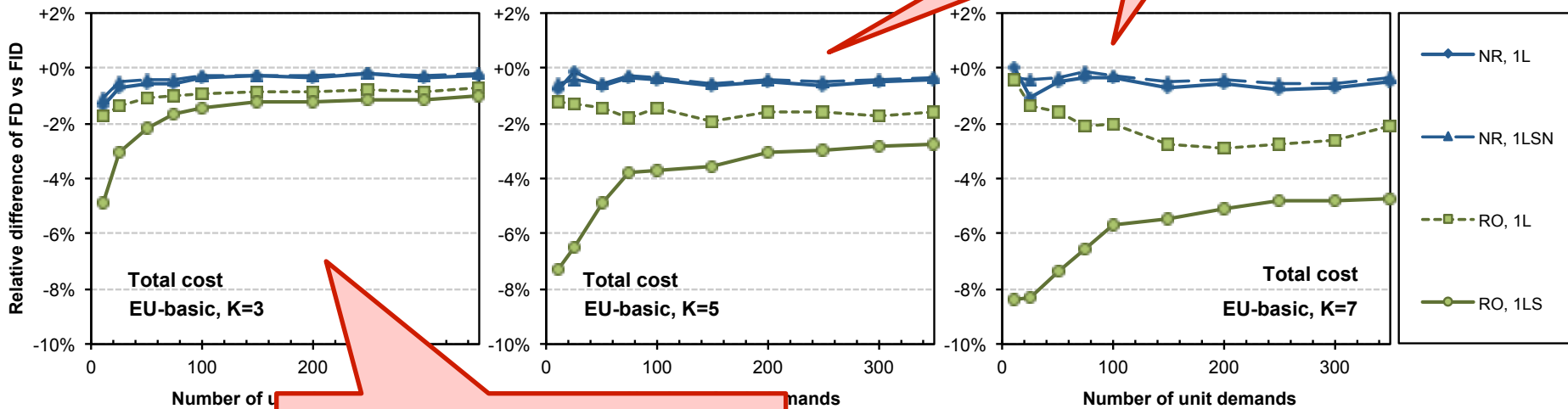
- FD is best, obviously
- Yet, difference is limited (few %)
 - at least for small K (= number of server sites)



Failure dependent rerouting? (FD vs FID)

- FD is best, obviously
- Yet, difference is limited (few %)
 - at least for small K (= number of server sites)

FD advantage increases for larger number of server sites!

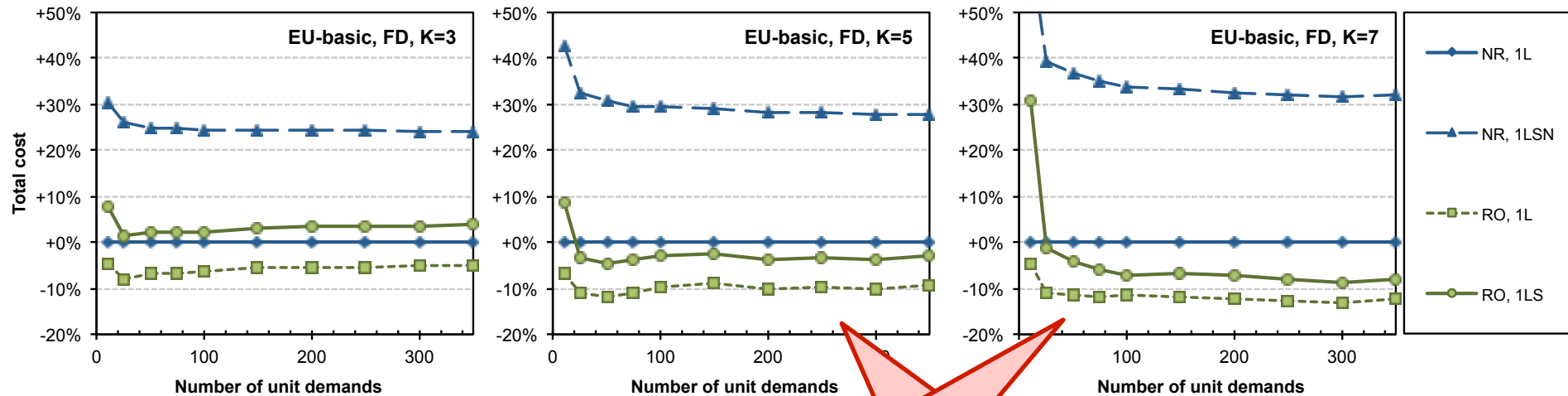


FD cost reduction larger for case of relocation (esp. 1LS)

Influence of K on benefit of relocation?

■ K ↗

- Wavelength reduction more pronounced
- Lower extra cost to provide single server failure protection



Relocation advantage increases for larger number of server sites!

Conclusions

- Dimensioning algorithm for resilient optical grids
 - Exploit relocation for resiliency
 - Compact ILP for finding K best locations
 - ILP (w/ column generation) for server & network dimensions
 - Generic model based on SRLG concept

- Case study on EU network topology [10-350 unit demands]
 - Relocation offers cost advantage of up to ca. 10% to protect against single network link failures
 - Total cost to protect against 1LS with relocation
≈ Cost to protect against 1L only, without relocation
 - Relocation advantage more substantial for larger number of server sites
 - Failure-dependent rerouting advantage if we use relocation (couple of %)

The next step: virtualization & accounting for server synchronization

M. Bui, B. Jaumard and C. Develder, *"Anycast end-to-end resilience for cloud services over virtual optical networks (Invited)"*, in Proc. 15th Int. Conf. Transparent Optical Netw. (ICTON 2013), Cartagena, Spain, 23-27 Jun. 2013. doi:10.1109/ICTON.2013.6603032

Resiliently provisioning virtual cloud networks

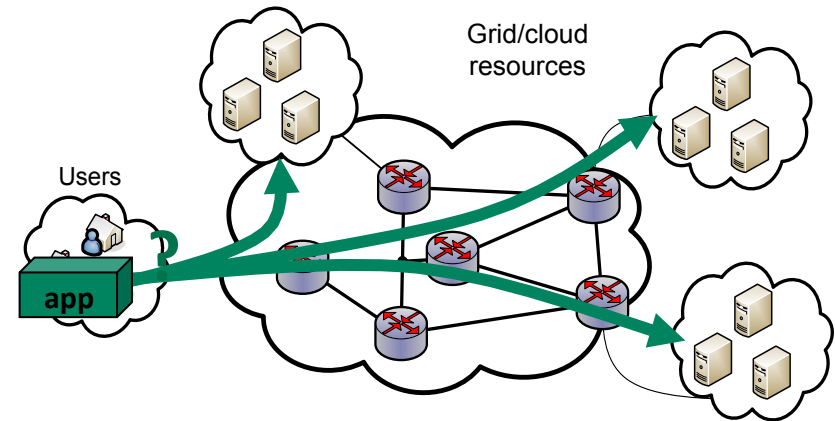
How to choose the virtual to physical mapping, such that

- Services remain available in case of network failures
- Bandwidth for providing services is minimal



Note:

- **Anycast**: requests coming from users can be served by any server
- Cloud services offered by VNO
- Cloud services run on top PIP

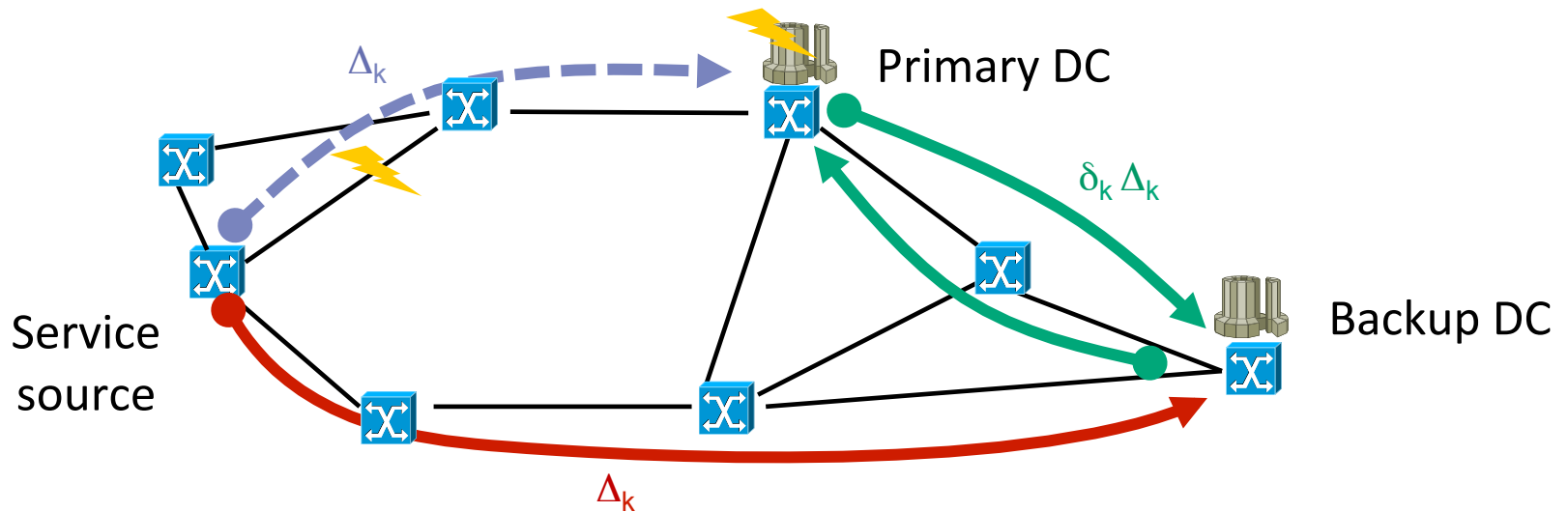


B. Jaumard, A. Shaikh and C. Develder, "Selecting the best locations for data centers in resilient optical grid/cloud dimensioning (Invited Paper)", in Proc. 14th Int. Conf. Transparent Optical Netw. (ICTON 2012), Coventry, UK, 2-5 Jul. 2012.

Protection scheme concept

■ Covered failures

- Network links \rightarrow disjoint **primary** and **backup** paths
- Data centers \rightarrow disjoint primary and backup server locations

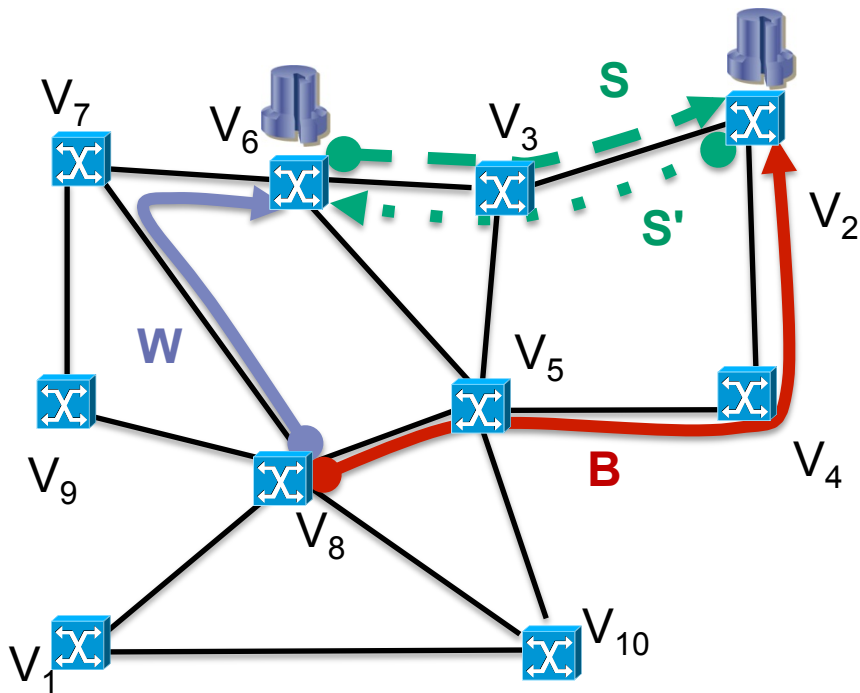


■ Note: **synchronization** between data centers for smooth fail-over switching!

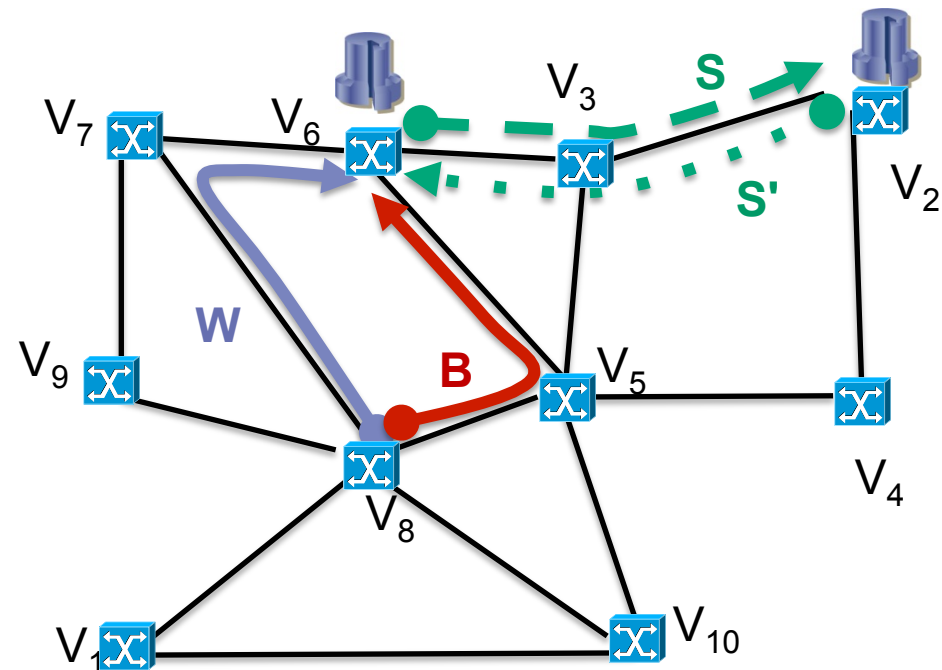
- We assume: sync needs fraction δ of service bandwidth Δ

Two proposed protection schemes

Scheme 1: VNO-resilience



Scheme 2: PIP-resilience



Related work

- Traditional dimensioning (no virtualisation, no resilience):
 - Develder *et al.* 2009: Anycast, flexibility in choosing data center
- Resilient dimensioning problem:
 - Shaikh *et al.* 2011, Develder *et al.* 2012: scalable method, no synchronization between working and backup DCs
- Routing cloud service requests and mapping a VNet to the physical infrastructure separately:
 - Lee *et al.* 2009, Yu *et al.* 2010: Survivable VNet embedding, but *assume VNet is given*
 - Jiang *et al.* 2012, Alicherry *et al.* 2012: Optimal server selection and routing of anycast services in the physical layer for intra- and inter-DC networks but *no resilient network design in the virtual layer*
- VNet planning problem:
 - Barla *et al.* 2012, Barla *et al.* 2013: using mixed integer linear programming, but *no synchronization between working and backup DCs*

Problem statement

Given

- Cloud network topology: $G = (V, L)$, with V = nodes, L = links
- Locations of the data centers, $V_D \subseteq V$
- Set of service requests, K
 - v_k : source of service
 - Δ_k : bandwidth requirement
 - Services originating from the same source are aggregated

Find

- Choice of primary and backup DC locations for each service
- Primary, backup **and synchronization** paths

Such that total used network bandwidth utilization is minimized

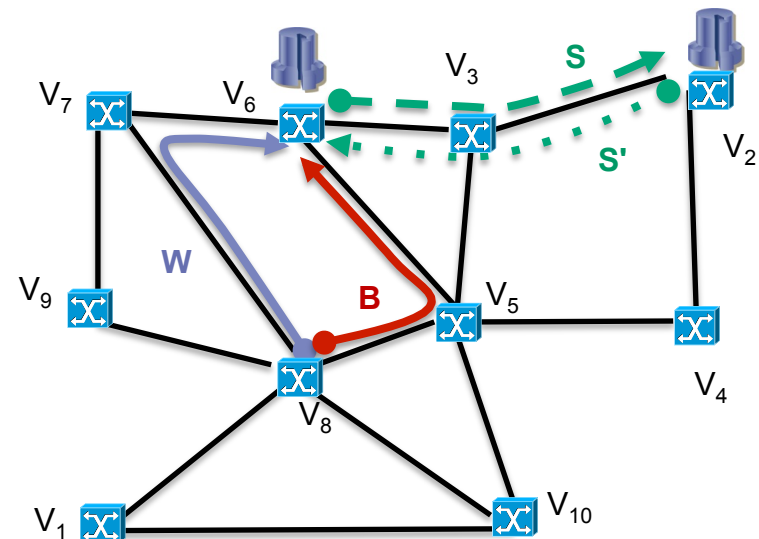
Solution: Column generation model

■ Column generation idea:

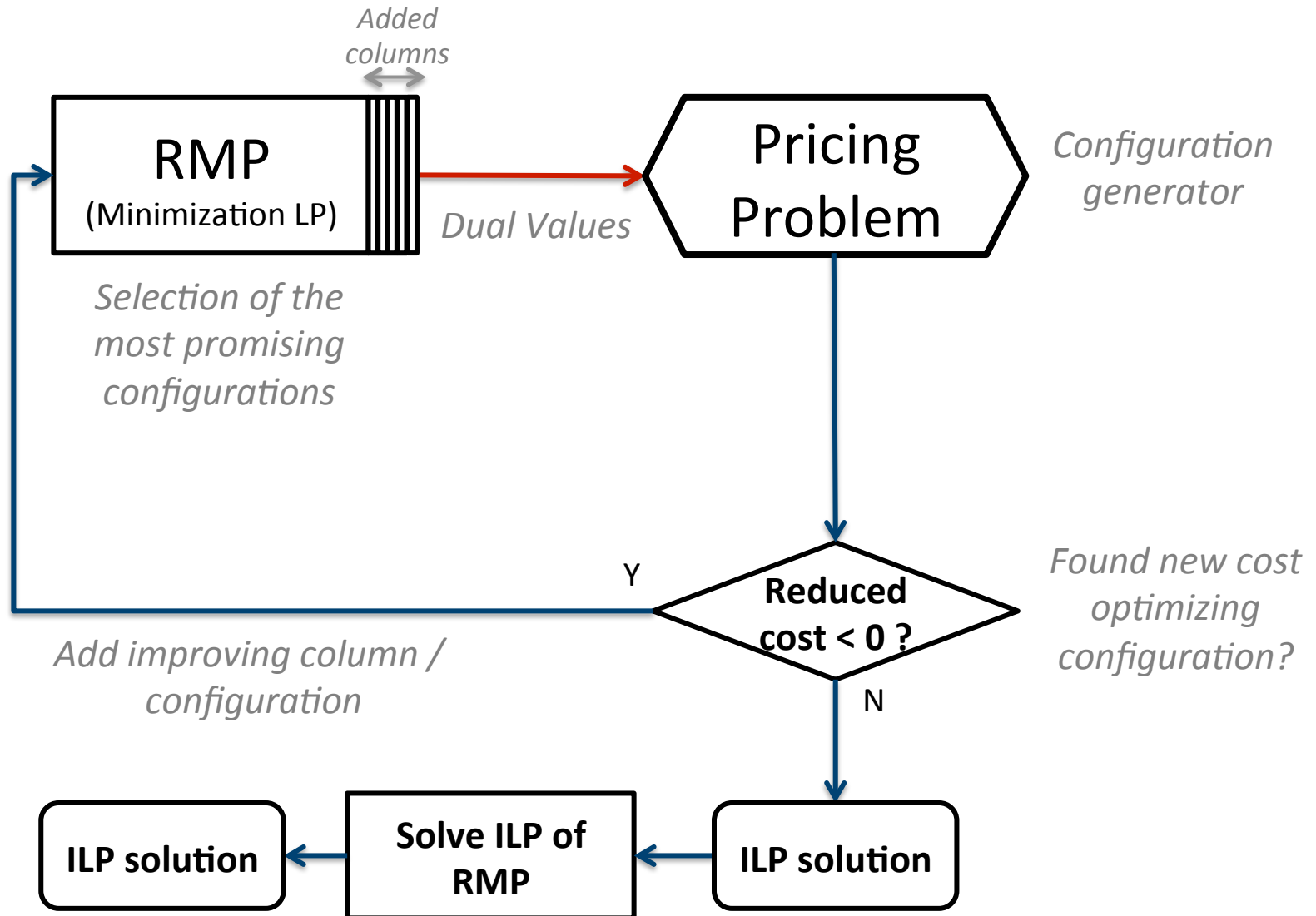
- Many different “configurations”
- Start from a restricted subset of such “configurations”
- Iteratively find additional configurations that reduce the cost:
 - (1) Restricted Master Problem (RMP)
 - (2) Pricing Problem (PP) to find new configs

■ A configuration =

- **Working** path
- **Backup** path
- 2 **sync** paths, one in each direction, between the primary & backup DCs
- Set of services protected by the set of 4 paths



Column generation solution algorithm



Master problem

- F – set of failure sets. Each $\mathcal{F} \in F$ contains links that fail at the same time.
- C – set of configurations c , associated with a source node v :
 - p^W : working path from v to primary datacenter DC^W
 - $p_\ell^W = 1$ if link ℓ used by the working path
 - p^B : backup path from v to primary datacenter DC^W (model 1) or backup datacenter DC^B (model 2)
 - $p_\ell^B = 1$ if link ℓ is used by the backup path
 - p^S : synchronization path from DC^W to DC^B
 - $p^{S'}$: synchronization path from DC^B to DC^W
- $\alpha_k = 1$ if service k is routed and protected by configuration c
- b_ℓ^W working synchronization bandwidth requirement on $\ell \in p^S$
- b_ℓ^S primary-to-backup synchronization bandwidth requirement on $\ell \in p^{S'}$
- β_ℓ^W is the working bandwidth and β_ℓ^B is the backup bandwidth on link ℓ

Restricted Master Problem (RMP)

Objective:

$$\min \sum_{l \in L} (\beta_l^W + \beta_l^B)$$

Constraints:

working backup

$$\beta_l^W + \beta_l^B \leq W_l \quad \ell \in L \quad (2)$$

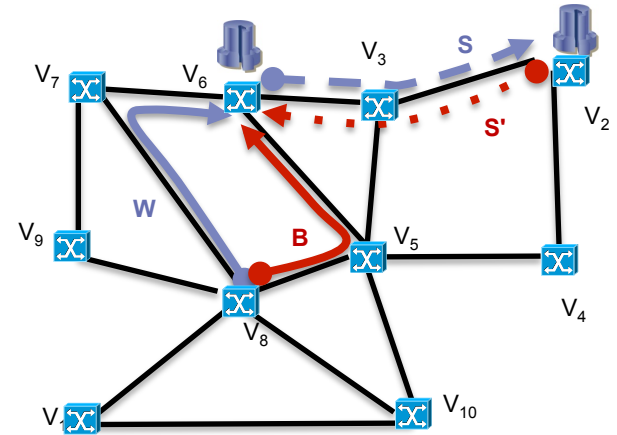
$$\beta_l^W = \sum_{c \in C} (b_l^{W,c} + b_l^{S,c}) z_c \quad \ell \in L \quad (3)$$

$$\sum_{c \in C} \alpha_k^c z_c \geq 1 \quad k \in K \quad (4)$$

$$\sum_{v \in V} \sum_{c \in C_v} \left(b_l^{W,c} p_{\ell'}^{B,c} + \sum_{k \in K_v} \Delta_k \alpha_k^c p_{\ell'}^{W,c} \delta_k p_{\ell'}^{S',c} \right) z_c \leq \beta_{\ell'}^B \quad \ell \in F, F \in \mathcal{F}, \ell' \in L \setminus F \quad (5)$$

$$z_c \in \{0, 1\} \quad c \in C \quad (6)$$

$$\beta_l^W \in \mathbb{R} \quad \ell \in L \quad (7)$$



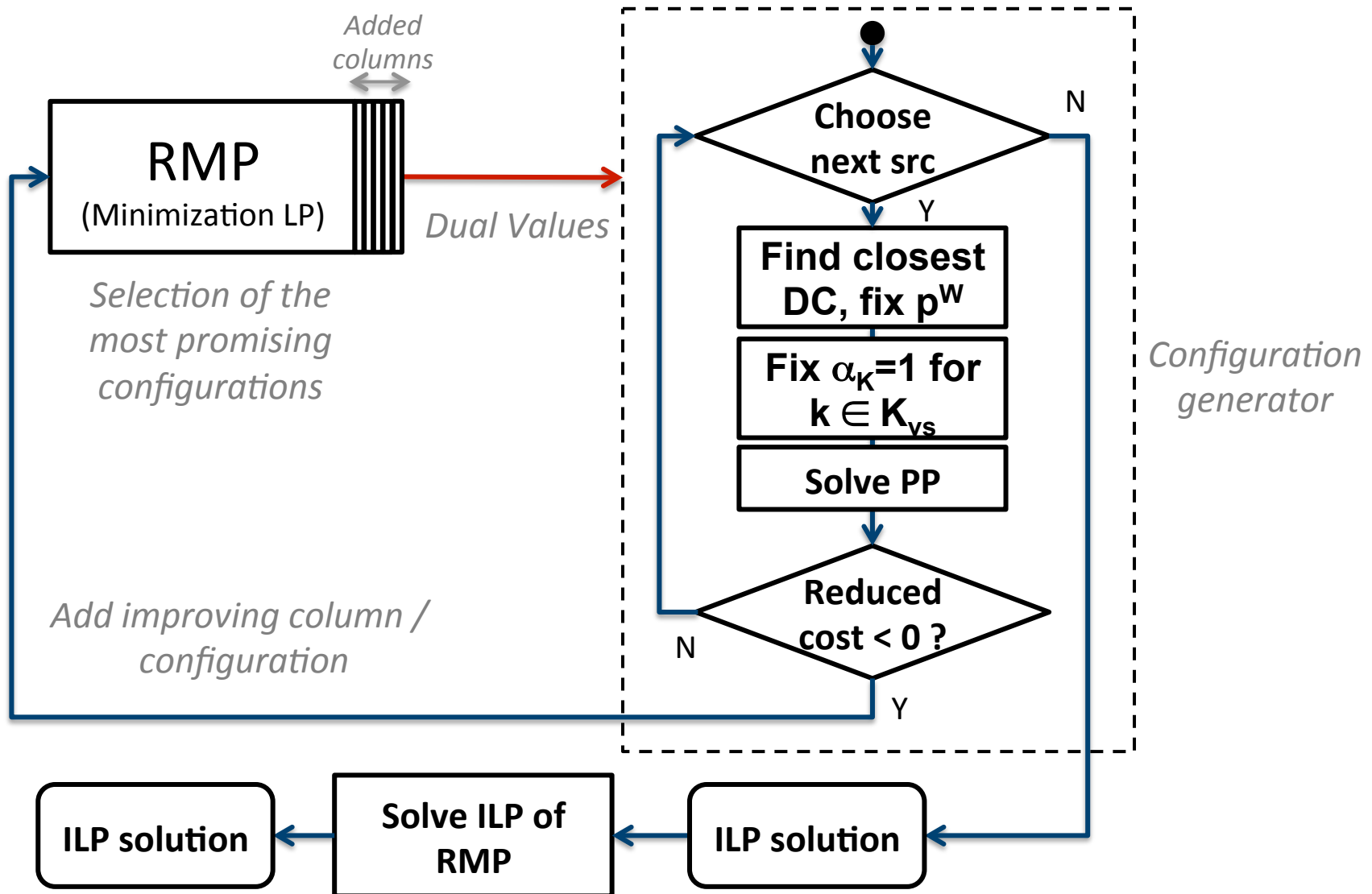
Pricing Problem (PP)

Minimize

$$\begin{aligned}
 \overline{\text{COST}} &= \sum_{\ell \in L} u_{\ell}^{(3)} (b_{\ell}^W + b_{\ell}^S) - \sum_{k \in K} u_k^{(4)} \alpha_k - \sum_{F \in \mathcal{F}} \sum_{\ell \in F} \sum_{\ell' \in L: \ell \neq \ell'} \sum_{k \in K_{vs}} u_{\ell \ell' F}^{(5)} \Delta_k \alpha_k p_{\ell}^W (p_{\ell'}^B + \delta_k p_{\ell'}^{S'}) \\
 &= \sum_{\ell \in L} u_{\ell}^{(3)} (b_{\ell}^W + b_{\ell}^S) - \sum_{k \in K} u_k^{(4)} \alpha_k - \sum_{F \in \mathcal{F}} \sum_{\ell \in F} \sum_{\ell' \in L: \ell \neq \ell'} u_{\ell \ell' F}^{(5)} b_{\ell}^W p_{\ell'}^B \\
 &\quad - \sum_{F \in \mathcal{F}} \sum_{\ell \in F} \sum_{\ell' \in L: \ell \neq \ell'} \sum_{k \in K_{vs}} u_{\ell \ell' F}^{(5)} \Delta_k \delta_k \alpha_k p_{\ell}^W p_{\ell'}^{S'}
 \end{aligned}$$

Cubic and square terms!

Column generation solution algorithm - Heuristic



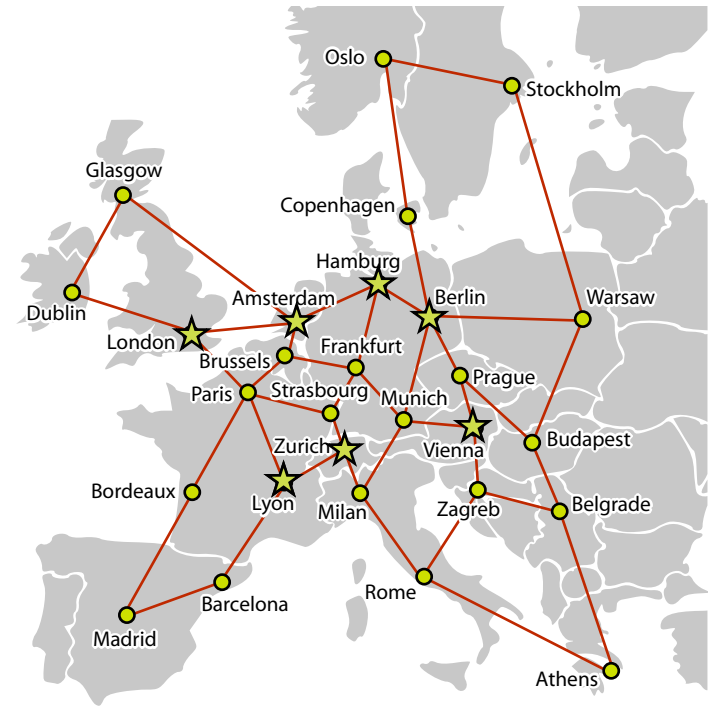
Case study

■ Topology:

- European network
- 28 nodes and 41 bidir links (= 82 directed)

■ Two choices of 4 data centers (DCs)

- *Scattered evenly:*
Lyon, Berlin, London, Vienna
- *Pairs of close data centers:*
Amsterdam, Hamburg, Lyon, Zurich

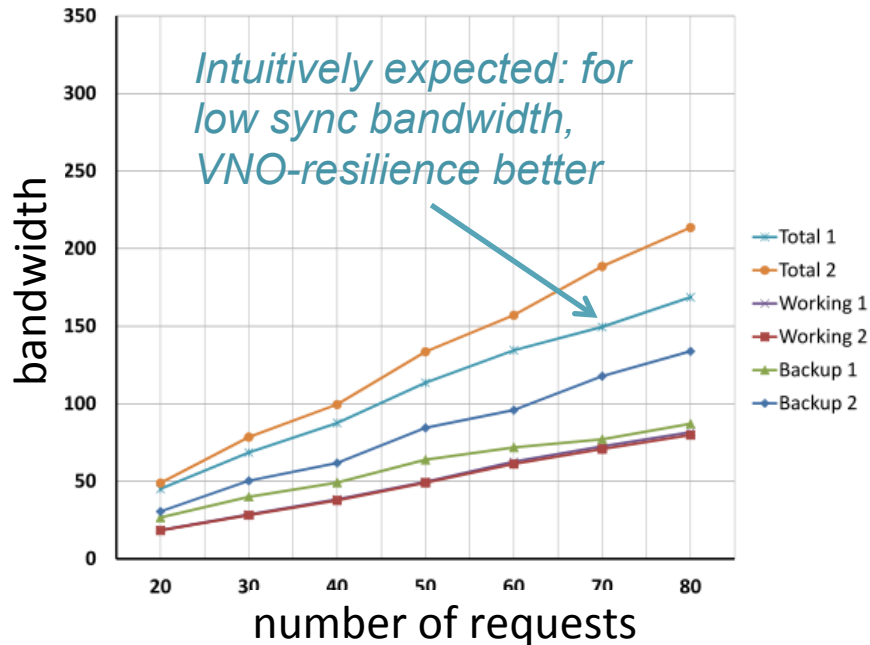


■ Synchronization bandwidth fraction: $\delta_k = 0.1$ or 0.9

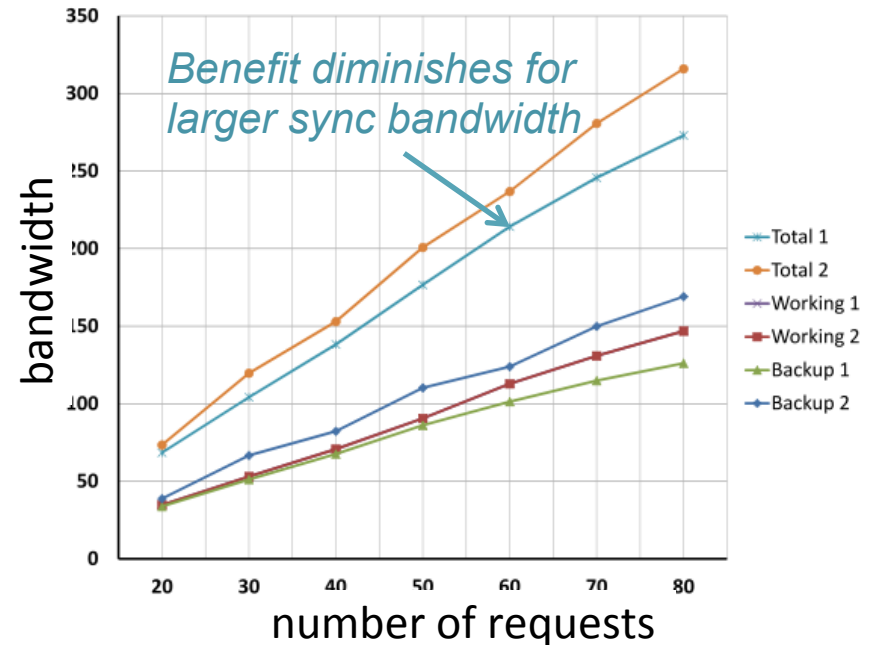
■ Requests generated randomly with bandwidth in $[0,1]$ wavelengths

Results: evenly distributed DCs

- DCs in Lyon, Berlin, London, and Vienna
- Model 1: **VNO-resilience**, Model 2: **PIP-resilience**



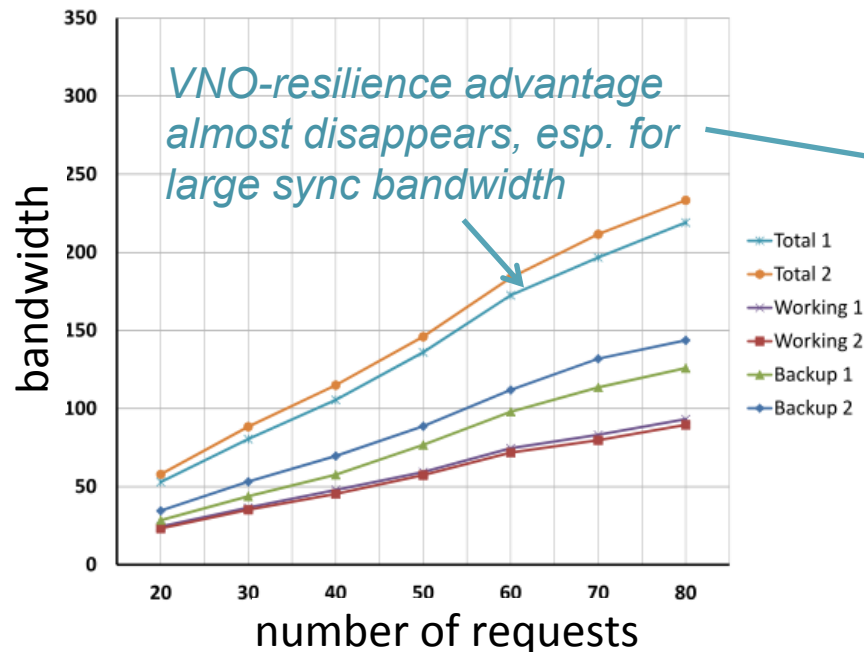
(a) $\delta_k = 0.1$



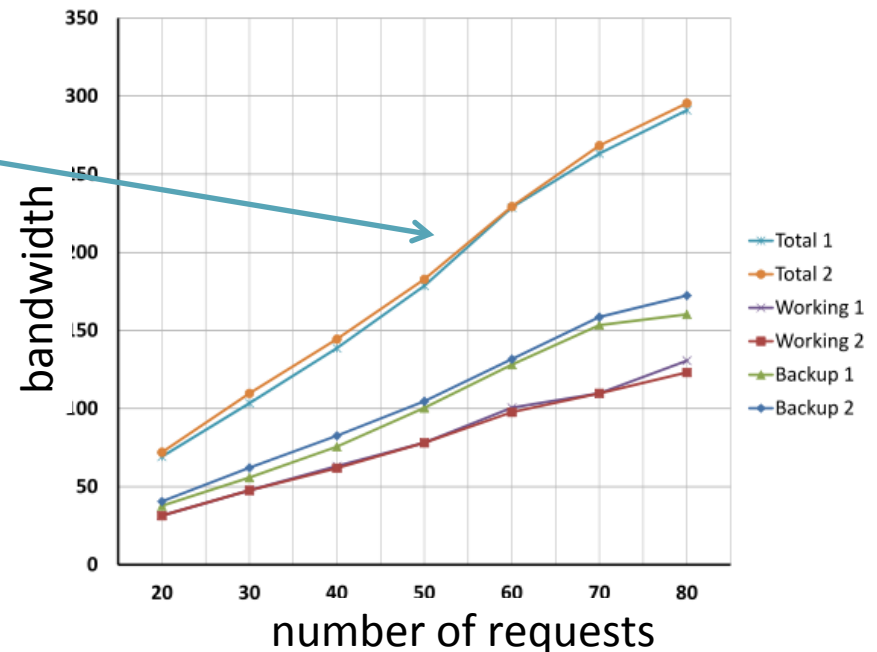
(b) $\delta_k = 0.9$

Results: close DC pairs

- DCs in Amsterdam, Hamburg, Lyon, Zurich
- Model 1: **VNO-resilience**, Model 2: **PIP-resilience**



(a) $\delta_k = 0.1$



(b) $\delta_k = 0.9$

Conclusions

- Scalable column-generation based method for resilient VNet planning
- Intuition: VNO-resilience has lower physical network requirements than PIP-resilience
- **But...** relative advantage of VNO-resilience may be limited
 - When accounting for synchronization bandwidth between DCs
 - If DCs occur in nearby locations
- Future work:
 - Optimization of choice of DC locations?
 - Incorporate DC capacity constraints (e.g., limit max load)

Wrap-up

Take-away points

- Characteristics of cloud computing:
 - Anycast: User does not greatly care of exact location of servers
 - Virtualization: Cloud service provider may want isolation
- Dimensioning cloud networks:
 - Network + DC: locations of data centers can be optimized
 - Shared protection: exploit anycast through relocation
 - Failure-dependent (FD) vs -independent (FID) routing: limited advantage of FD for small number of data center locations
 - Virtualization: VNO vs PIP resilience: VNO savings can be limited for certain data center location strategies

Thank you ... any questions?

?

Prof. Chris Develder

chris.develder@intec.ugent.be

Ghent University – iMinds

