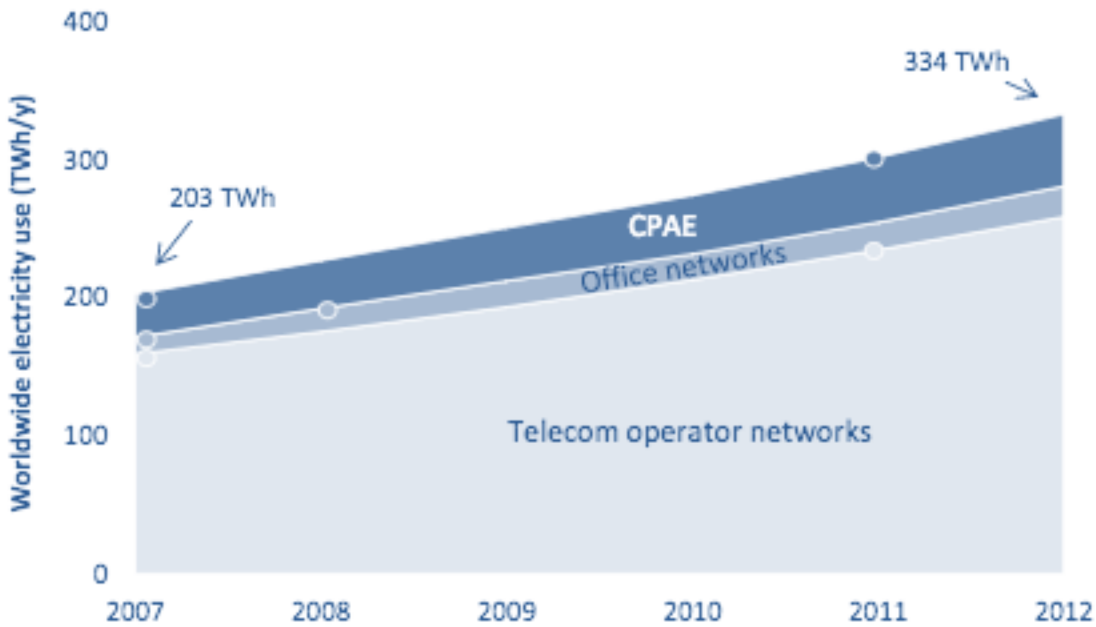
ENERGY EFFICIENT SOFTWARE DEFINED NETWORKS

Nicolas HUIN COATI and SigNet, I3S/Inria

Supervisors: Frédéric Giroire & Dino Lopez

28th September 2017

Energy consumption of Networks



- In 2012, communication networks consumed 330 TWh (4,6%)
- 10% yearly growth (worldwide: 3%)

[Van Heddeghem et al., ‘14]

Reducing Network's Power Consumption

- Device's power consumption is not proportional to its load
  - Improving devices' **power proportionality** [Nicollini et al, 12]
- Power off **base station** in mobile networks [Zhou et al, 09]
- Consolidation of **Virtual Machines** [Lin et al, 11]
- Energy Aware Routing (EAR)
  - Minimizing the number of **active network devices**:
    - ➢Our approach

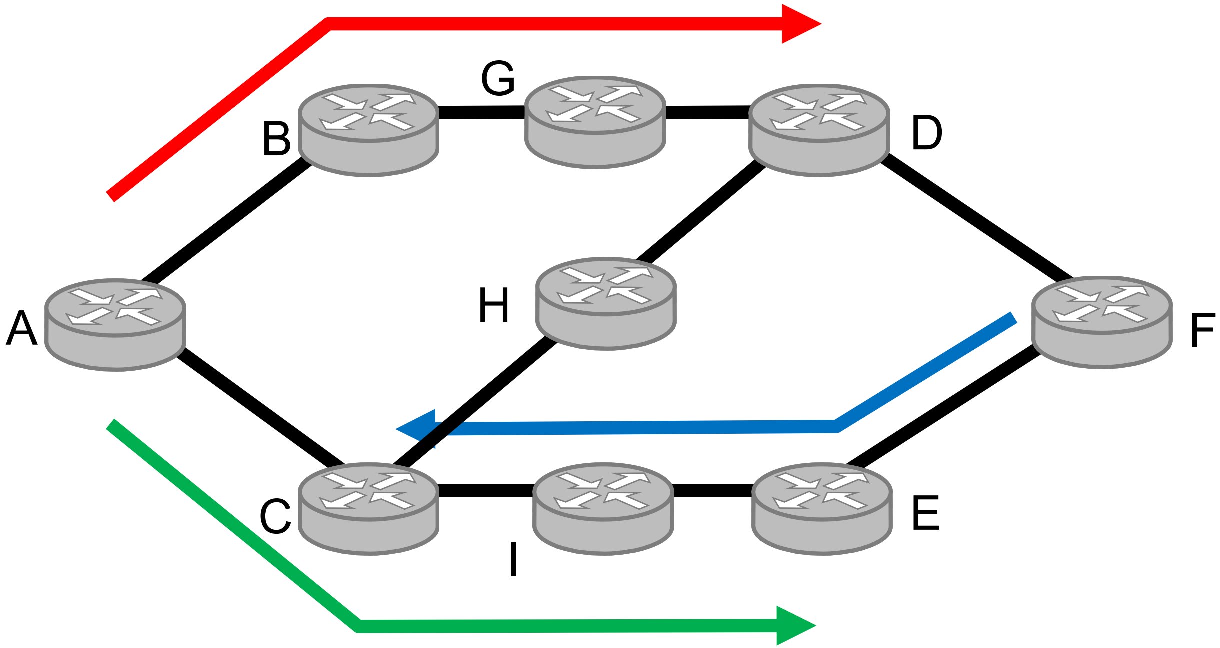
Energy Aware Routing (EAR)

Path between:

A et D

F et C

A et E



Routing request while minimizing the number of active devices (routers and/or links)

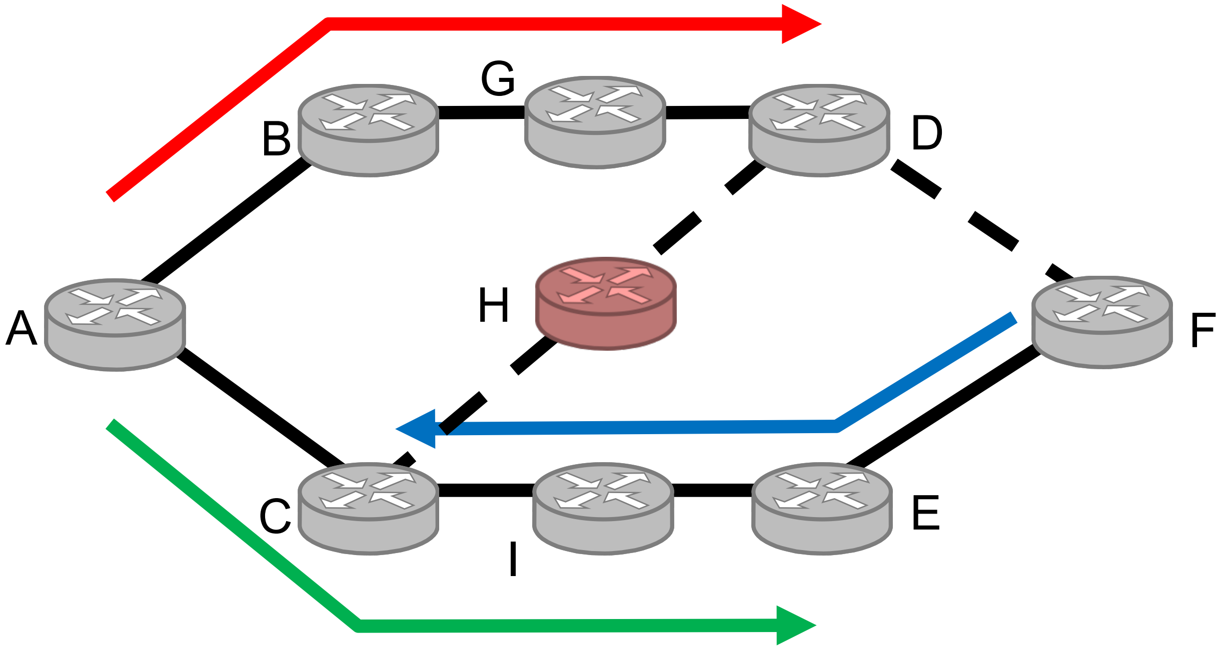
Energy Aware Routing (EAR)

Path between:

A et D

F et C

A et E



Shortest path routing

Routing request while minimizing the number of active devices (routers and/or links)

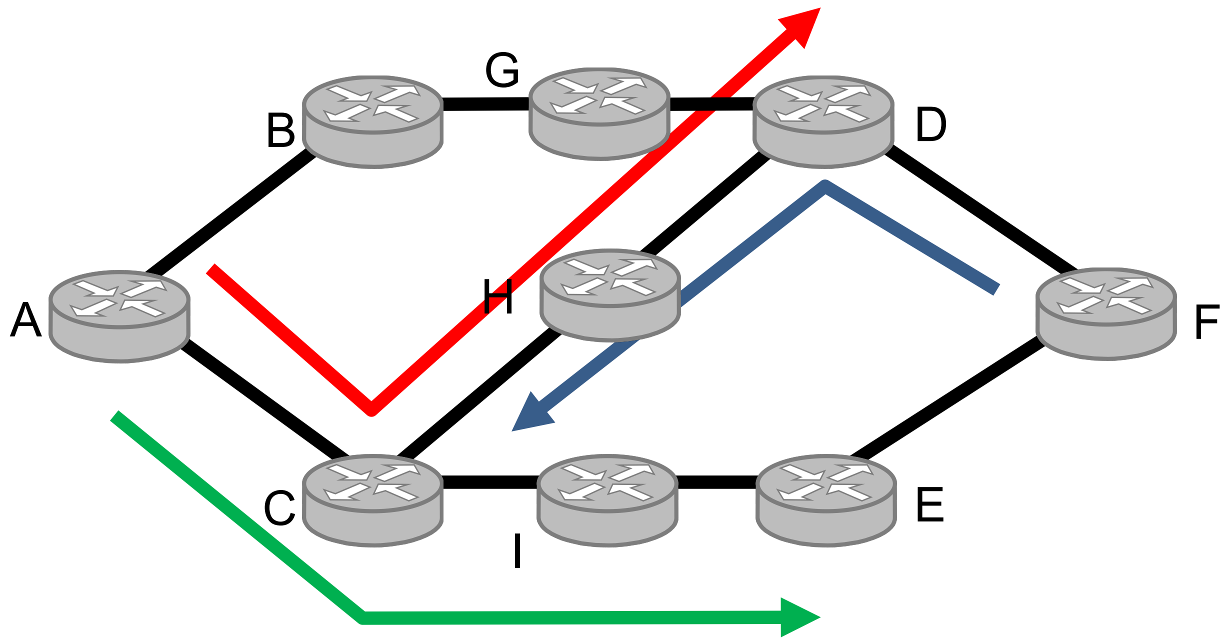
Energy Aware Routing (EAR)

Path between:

A et D

F et C

A et E



Routing request while minimizing the number of active devices (routers and/or links)

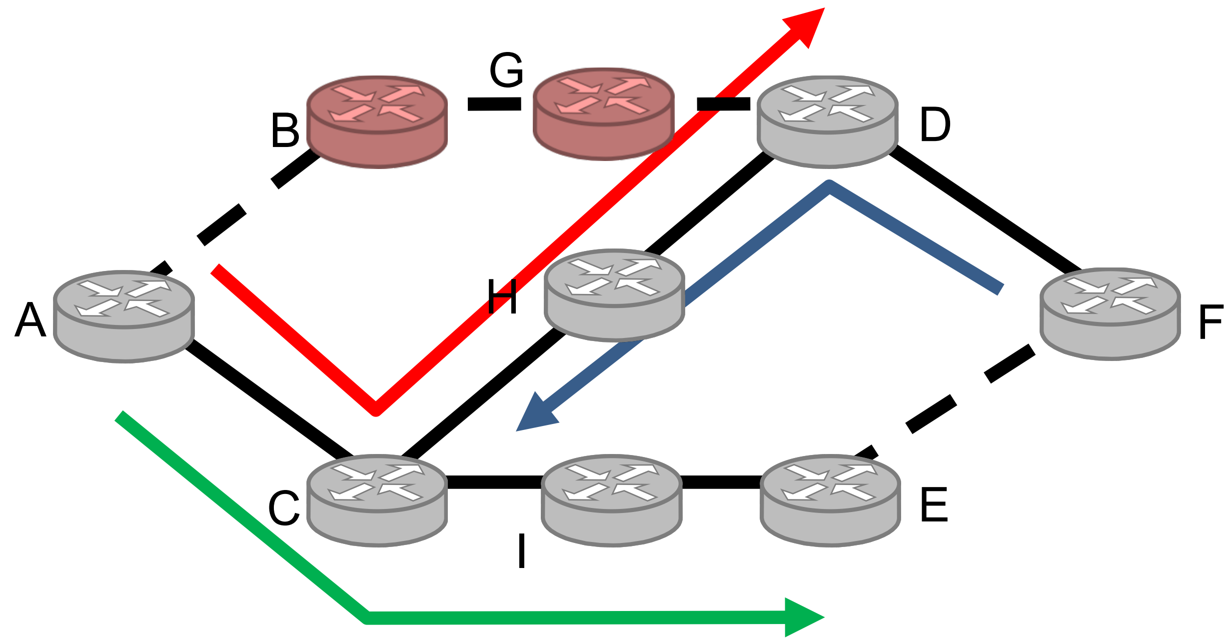
Energy Aware Routing (EAR)

Path between:

A et D

F et C

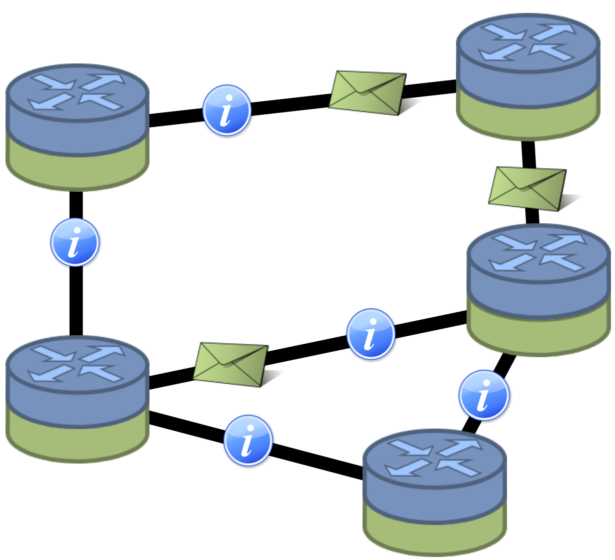
A et E

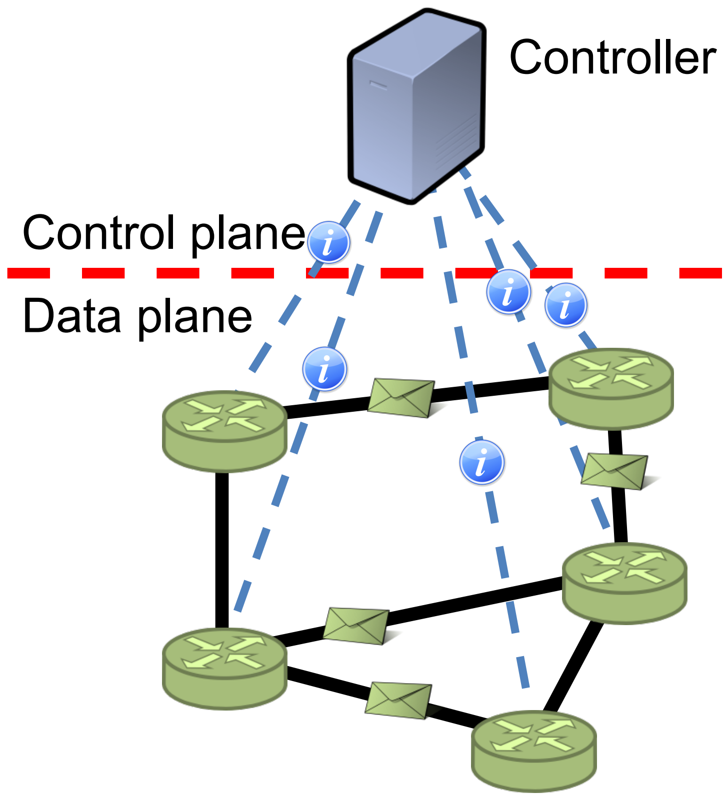


Energy Aware Routing

Routing request while minimizing the number of active devices (routers and/or links)

Legacy vs. Software Defined Networks (SDN)





Legacy network

- Distributed control
- Manual configuration

SDN network

- Centralized control
- Policies deployed by the controller

Network Function Virtualization (NFV)

Legacy networks implements *network functions* using expensive specific hardware called **middleboxes**.

- Limit adaptability to traffic (even with SDN)

The NFV initiative allows function to be run on general hardware using **Virtual Machines** (VMs).

- Enables greater flexibility (good for energy)

Goal of this thesis

Leveraging **SDN** and **NFV** for the deployment of **Energy Aware Routing**

Consider the new **constraints** of these paradigms

Tools

- Linear Programming
- Column Generation
- Greedy Heuristics
- SDN testbed (with SigNet team)

During my thesis

SDN

- Forwarding table constraints
  - The Compression Problem *(Chapter 3)*
  - EAR with Compression *(Chapter 4)*
  -  *MINNIE (Chapter 5)*
- Hybrid SDN networks: *SENAtoR (Chapter 6)*

Greedy
ILP

Testbed

NFV

- Service Function Chaining
  - Provisioning *(Chapter 7)*
  - Energy efficiency *(Chapter 8)*

Column Generation

P2P

- Structured overlay for live video streaming
  - Homogeneous *(Appendix 1)*
  - Heterogeneous *(Appendix 2)*

SOFTWARE DEFINED NETWORKS

Energy Aware Routing with Compression

« The first day there was OpenFlow »

The OpenFlow API was developed at Stanford [McKeown et al., 2008]

Matching Rule	Action
Dest. IP (as in legacy network) Src. IP Port … **40 fields** in OpenFlow 1.3	DROP FORWARD TO PORT ENCAPSULATE & FORWARD …

OpenFlow provides **per flow** routing (more complex)
Rules stored in **TCAM,** power hungry and with **limited size** (1 to 3k rules)

- Constraints on the number of forwarding rules

Related Work

- Reduce OpenFlow rule size [Banerjee et al., 14], [Kannan et al, 13]
  - Not standard
- Eviction of rules
  - Frequent contact with the controller
- Spread the rules on the network (« One Big Switch » abstraction) [Nguyen et al., '15]
  - Not practical for forwarding rules
- Our contribution: **Aggregation rules**

The Compression Problem

Priority

Flow	Output port
(0, 4)	Port-4
(0, 5)	Port-5
(0, 6)	Port-5
(1, 4)	Port-6
(1, 5)	Port-4
(1, 6)	Port-6
(2, 4)	Port-4
(2, 5)	Port-5
(2, 6)	Port-6

Flow	Output port
(1, 5)	Port-4
(2, 6)	Port-6
(1, *)	Port-6
(*, 4)	Port-4
(*, *)	Port-5

Reduce the size of forwarding table using wildcard and default rules while maintaining the same routing

(NP-Hard) [Giroire et al., ‘15]

The Compression Problem

Priority

Flow	Output port
(0, 4)	Port-4
(0, 5)	Port-5
(0, 6)	Port-5
(1, 4)	Port-6
(1, 5)	Port-4
(1, 6)	Port-6
(2, 4)	Port-4
(2, 5)	Port-5
(2, 6)	Port-6

→

Flow	Output port
(1, 5)	Port-4
(2, 6)	Port-6
(1, *)	Port-6
(*, 4)	Port-4
(*, *)	Port-5

Be careful about the order of the rules
(1, *) then (*, 4) != (*, 4) then (1, *)

Energy Aware Routing with Compression Problem (EARC)

Goal

Minimize the total energy consumption of the network

Input

- Network G=(V, A)
- Set of requests D, between s_i and t_i and bandwidth d_i
- Link capacities C_{uv}
- Forwarding table capacities C_u

Output

- Path for every request
- Respect node and link capacities

Contributions

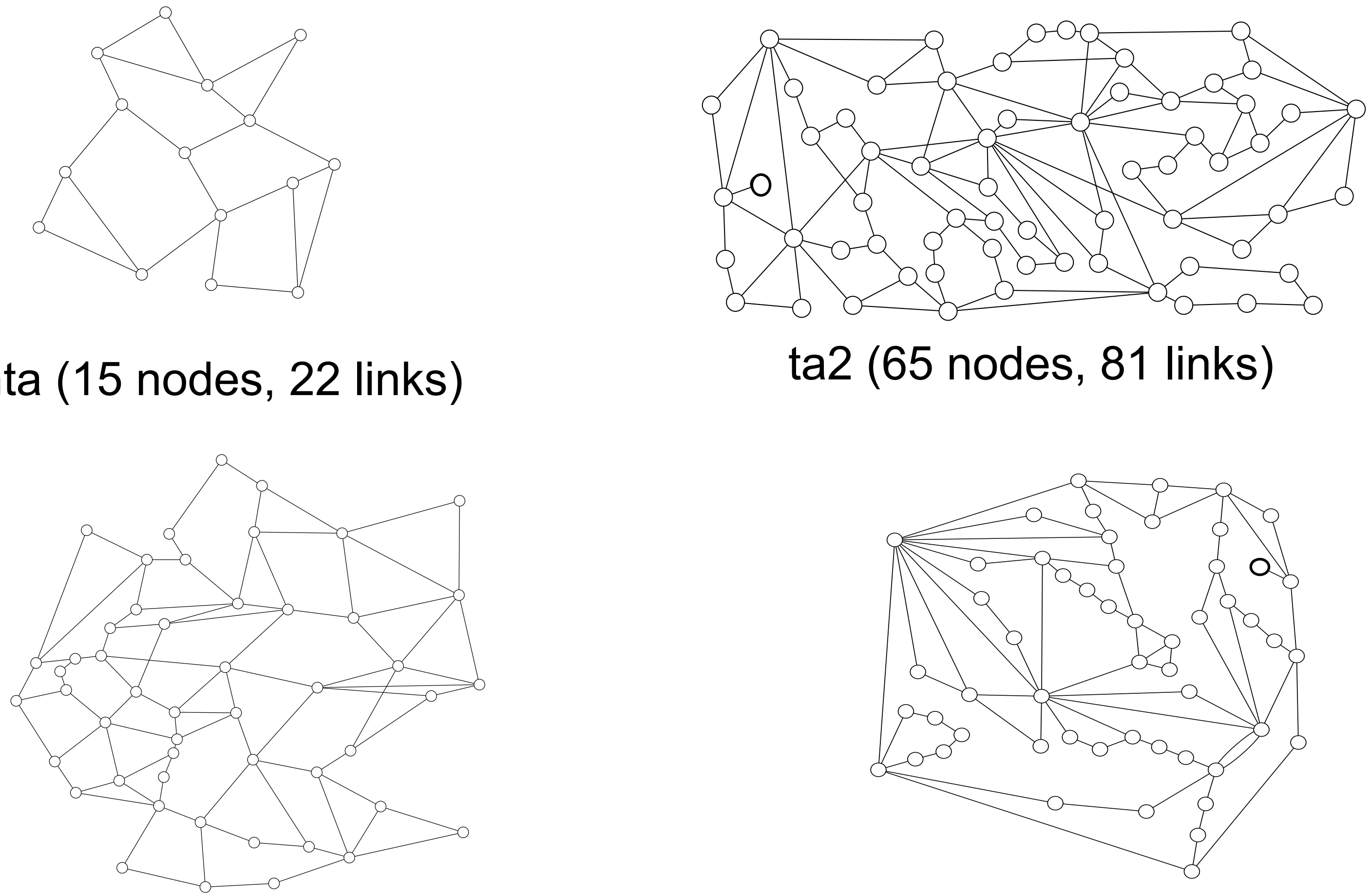
Havet, **H,** Moulierac, Phan
AlgoTel'16

- ILP formulations
  - default rule only
  - default rule and wildcard rules

- Heuristic
  - Energy saving module
    - Shutdown links
  - Routing module
    - Find a weighted shortest path according to **table and link usage**
  - Compression module
    - Reduce table at max capacity using wildcard rules (multiple solutions)

SNDlib topologies

http://sndlib.zib.de



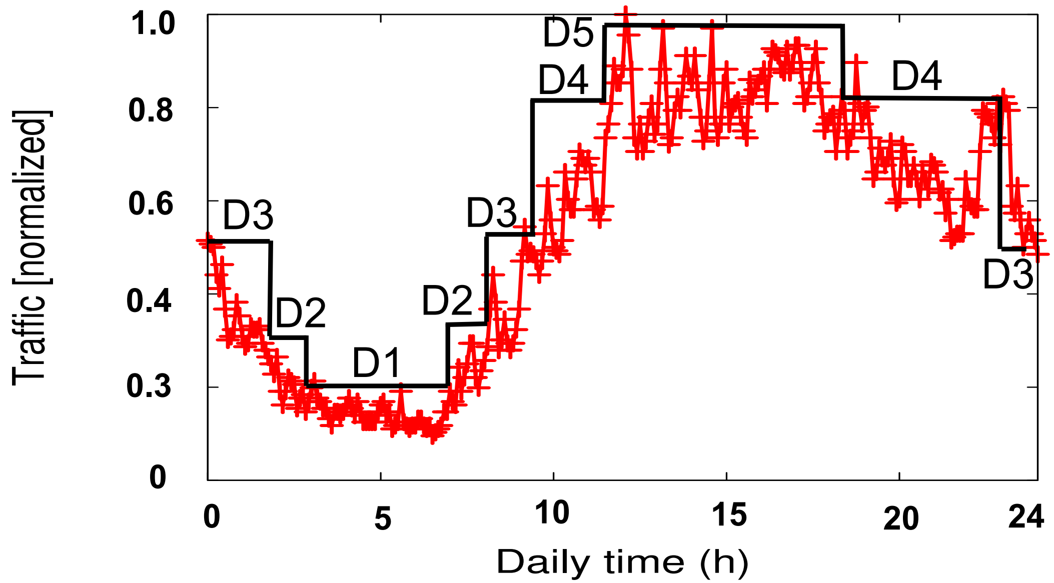
atlanta (15 nodes, 22 links)

ta2 (65 nodes, 81 links)

germany50 (50 nodes, 44 links)

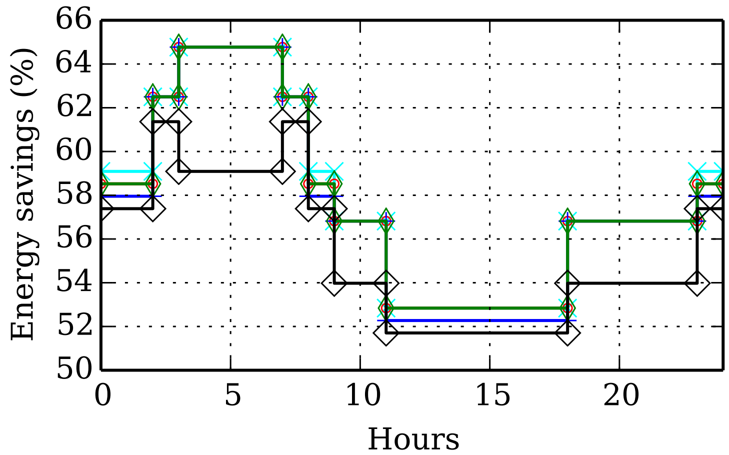
zib54 (54 nodes, 108 links)

Traffic estimation

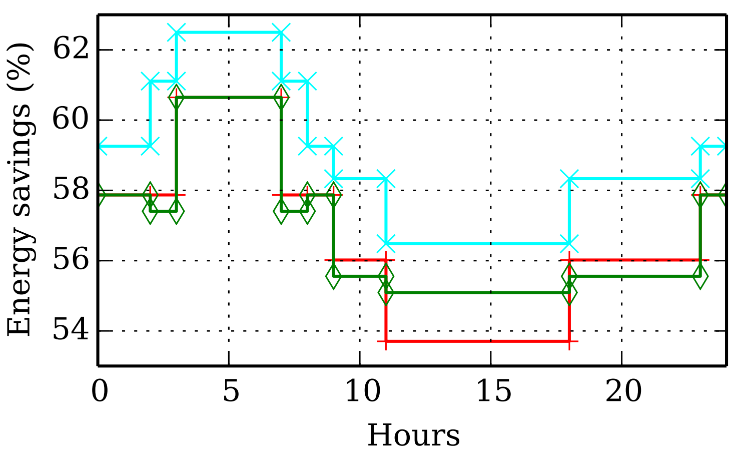


- ISP traffic follows predictable patterns
- Small granularity of period creates instability
- Only a few configurations are sufficient [Araujo et al. ,2016]

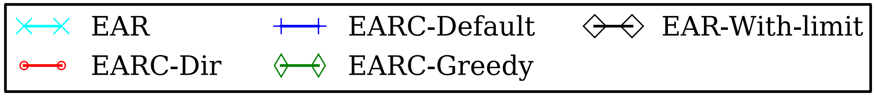
Energy savings during the day



germany50 (50 nodes, 44 links)



ta2 (65 nodes, 81 links)

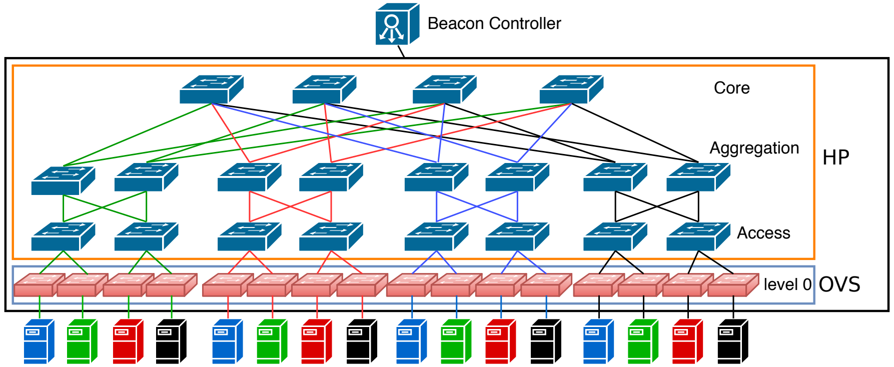


- **Not always possible** to route w/o aggregation rules
- Aggregation rules enable energy savings close to **classical EAR**

SDN IN PRACTICE

MINNIE

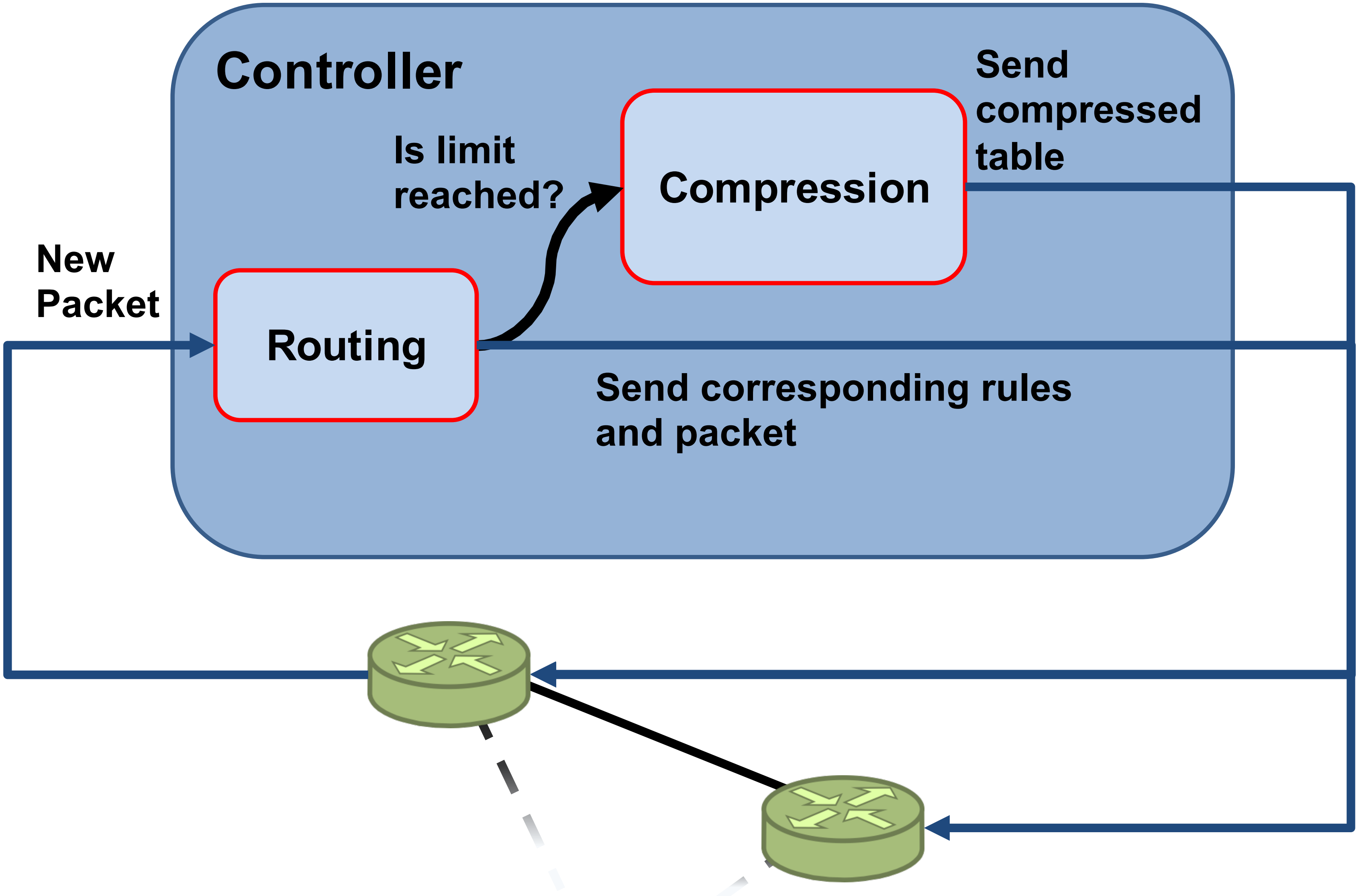
MINNIE: Compressing in data centers



Rifai, **H**, Caillouet, Giroire, Moulierac , Lopez, Urvoy-Keller
GLOBECOM '15, AlgoTel '16, Computer Network

- Collaboration with the SigNet team
- HP SDN capable switch
- Impact of compression on packet's delay and losses

MINNIE

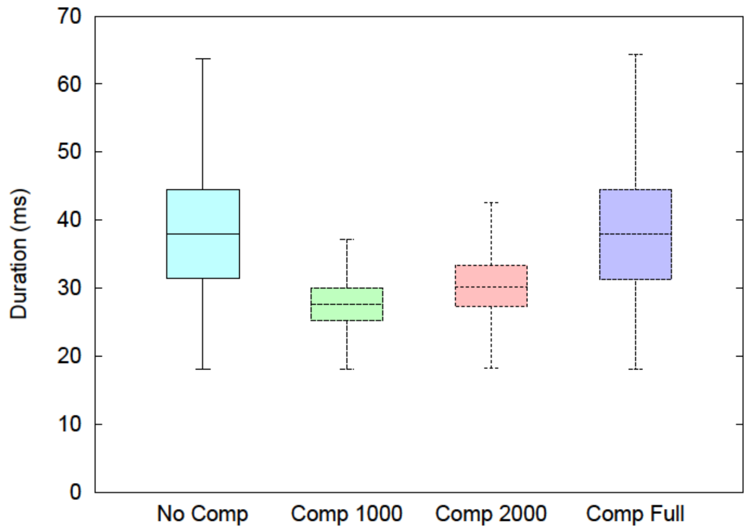

Controller
Send compressed table
Is limit reached?
Compression
New Packet
Routing
Send corresponding rules and packet

Results: Ratio, losses & # compressions

	Compression				
	None	at 500	at 1000	at 2000	when full
Average compression ratio	-	83.21%	82.19%	81.55%	81.44%
Packet losses (%)	6.25×10^{-6}	0.003	5.65×10^{-4}	2.83×10^{-5}	3.7×10^{-4}
# compressions	-	16 594	95	28	20

- Average compression ratio >80% (**at least** 77%)
- Compression has **no significant impact** on losses
  - Except when the threshold is too low

Results: Delay



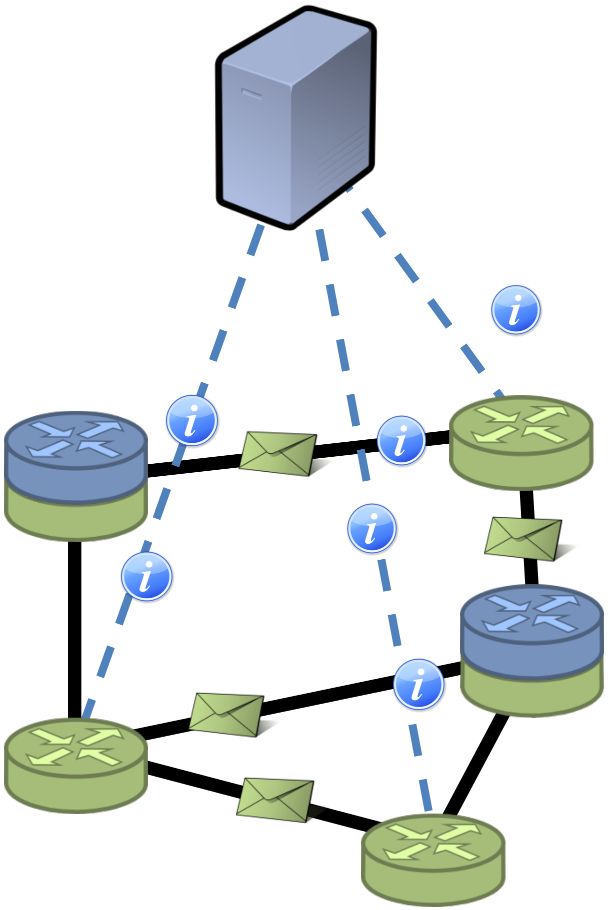
Delay

- Compression adds **no delay** (if we forget the « 500 » threshold)
  - ➢ Delayed compression
- Compression **reduces** the first packet delay
  - ➢ Avoid installing rule if corresponding wildcard rule exists

SDN IN PRACTICE

EAR in hybrid networks

SDN & Legacy Interaction



- All solutions and framework consider full SDN networks
- Progressive migration from legacy to SDN
- Cohabitation of SDN & legacy devices and protocols (e.g., OSPF)

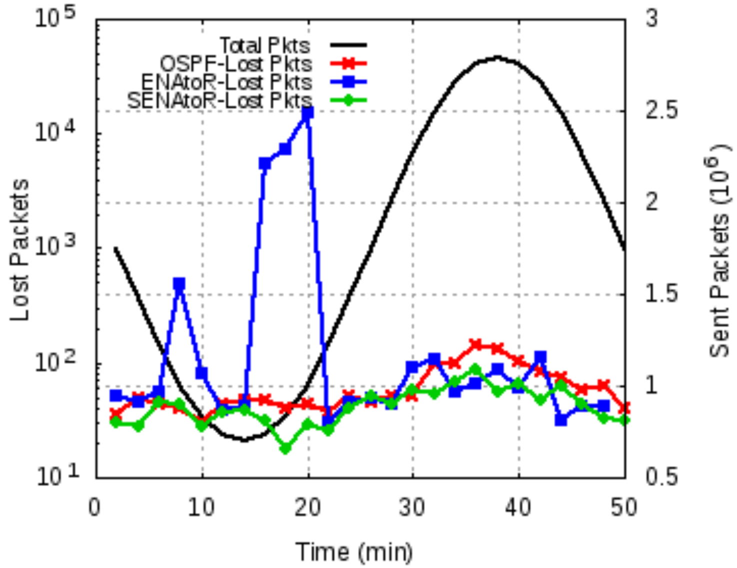
For Energy Aware Routing:
SDN devices shutdown
➢ failure for legacy

Contributions

H, Rifai, Giroire, Lopez, Urvoy-Keller, Moulierac
GLOBECOM '17

- Bring Energy Aware Routing closer to reality
- *Smooth ENergy Aware Routing* (SENAtoR)
  - Smooth link extinction
  - Backup tunnels for link shutdown
  - Traffic spike mitigation (link failure or flash crowd)
  - Heuristic for EAR with SDN and backup tunnel placement

Results: Packet losses



- Same order of packet losses than legacy network
- **Smooth extinction** helps to mitigate packet losses

NETWORK FUNCTION VIRTUALIZATION

« À à à la queleuleu » ♪

NFV & Energy Efficiency

Network functions implemented on **specific hardware (middlebox)**

- Hard to move and, thus, adapt to traffic

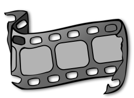
With **virtualization**, functions can be executed on Virtual Machines (VM)

- Enables greater flexibility (good for energy)

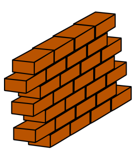
Scenario	Router	Network Function
Baseline	Legacy	Middlebox
Hardware	SDN	Middlebox
NFV	SDN	NFV

Service Function Chains (SFC)

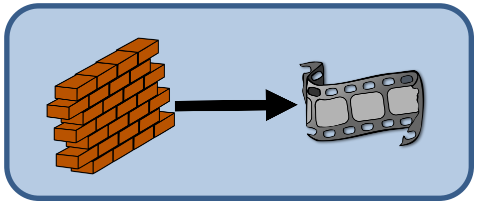
Service Chain: **ordered** chain of network functions to apply to flows on the network


Video optimization

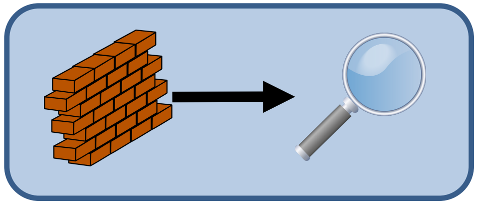

Deep packet inspection


Firewall

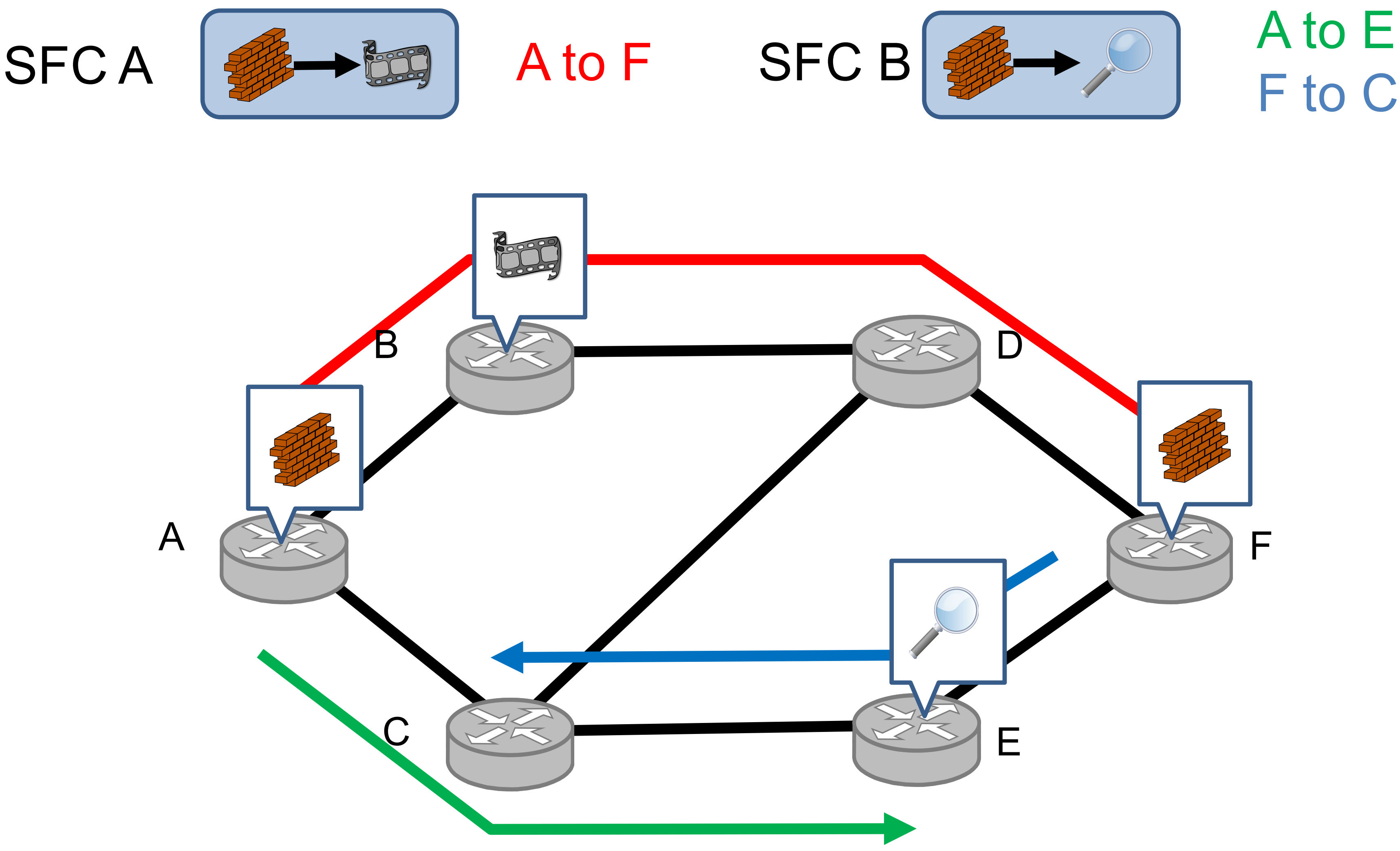
SFC A



SFC B



Example of Service Function Chains



Energy Efficient SFC Placement

Goal

Minimize the total energy consumption of the network

Input

- Network G=(V, A)
- Set of requests D
  - between s_i and t_i , bandwidth d_i and **chain** $\mathbf{c_i = (f_0, f_1, \ldots, f_k)}$
- Link capacities C_{uv}
- Node capacities C_u (e.g., number of available CPU cores, memory)

Output

Path and function placement for every request

Respect node and link capacities

Related Work

- SFC Placement
  - Heuristics with no performance guarantee
  - Partial and exact mathematical formulations
    - Solve placement and routing independently. [Martini et al., 2015] [Riggio et al., 2015]
    - Small network or small number of requests. [Gupta et al., 2015] [Savi et al, 2015]
- Energy & Virtualization
  - Some works on NFV, not on SFC

Contributions

H, Jaumard, Giroire *ICC 2017*
H, Tomassilli, Giroire ,Jaumard *INOC 2017*

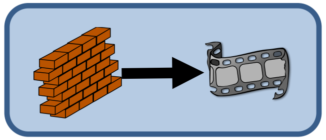
- Minimize:
  - Bandwidth and study impact of number of NFV nodes (near optimal)
  - Energy consumption of links and nodes

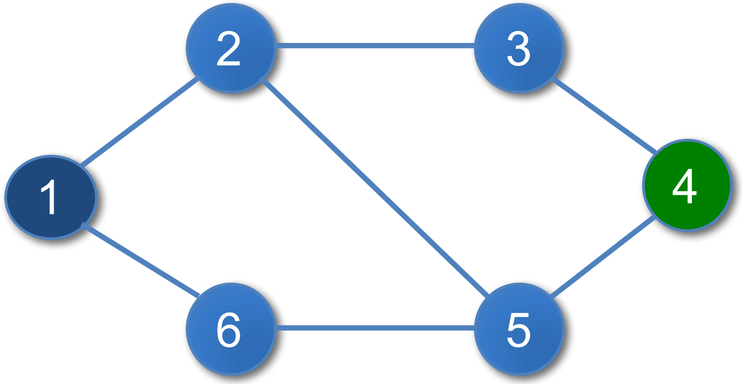
- Find solutions for all-to-all traffic (10k requests) on networks up to 50 nodes.
  - **Layered graph**
  - **Column generation model**
  - **Improving integrality gap with cuts**

- Function replicas limit
- GreenChains: ILP-Based heuristics

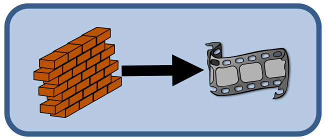
Layered Graph

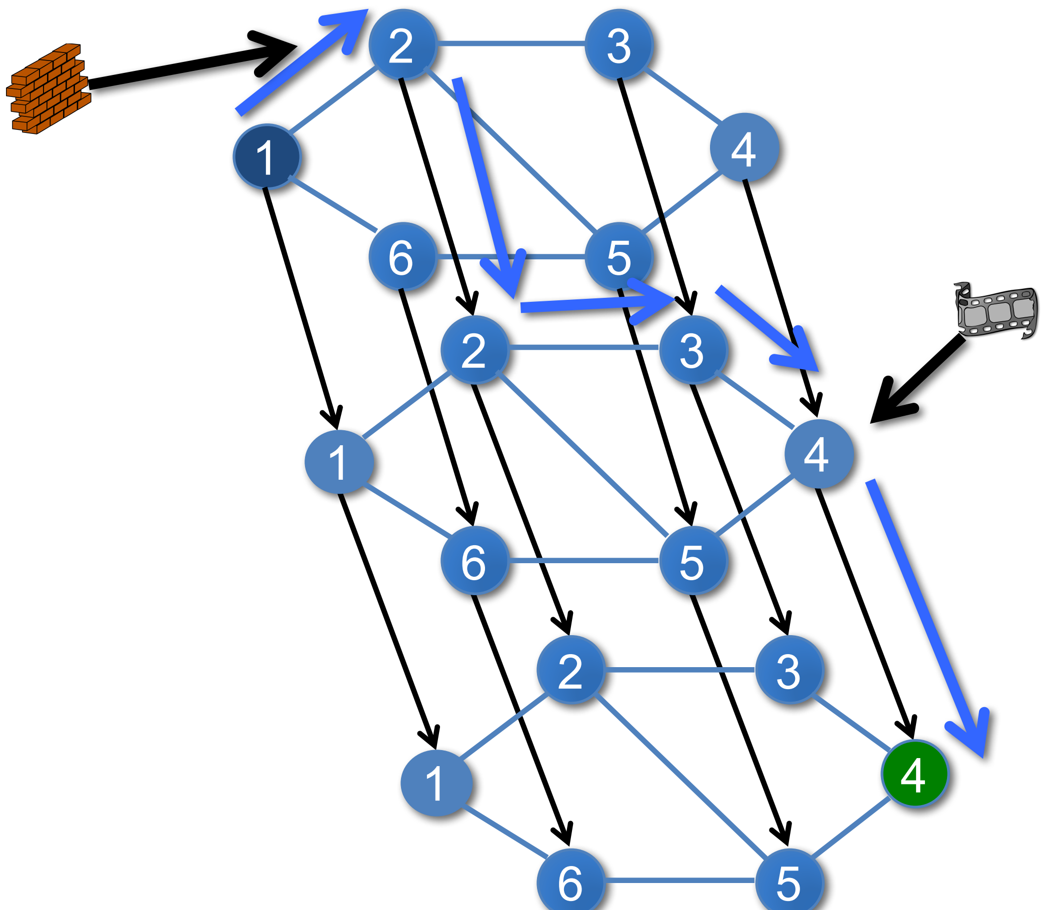
Propose an alternate way to find **Service Path** (path & placement of function)

Request between 1 and 4 for SFC 



Layered Graph

Request between 1 and 4 for service 



- # layers = # functions + 1
- Link **between layers** gives the **placement**
- Link **inside layers** gives the **routing**
- Path from first to last layer

ILP Formulation (CG-simple)

$$\min \underbrace{\sum_{(u,v)\in A} P^{\text{IDLE}}_{uv} x_{uv}}_{\text{link switch on energy}} + \underbrace{\sum_{(u,v)\in A} \sum_{p\in P^c_{sd}} \delta^p_{uv} \left(\sum_{d=(u_s,u_d,c)\in D} \frac{D^c_{sd}}{C^{\text{LINK}}_{\ell}} \mathrm{P}^{\max}_{uv} \right) y^p_d}_{\text{link bandwidth energy}} + \underbrace{\sum_{u\in V} \mathrm{P}_u \mathrm{K}_u}_{\text{node resource energy}}$$

One path per demand: $$\sum_{p\in P^c_{sd}} y^p_d = 1 \qquad (u_s,u_d)\in \mathcal{SD}, c\in C_{sd}$$

Link capacity: $$\sum_{d=(u_s,u_d,c)\in D} \sum_{p\in P^c_{sd}} D^c_{sd}\, \delta^p_{uv} y^p_d \le x_{uv} C^{\text{LINK}}_{uv} \qquad (u,v)\in A$$

Node capacity: $$\sum_{d\in D} \sum_{p\in P^c_{sd}} D^c_{sd} \left(\sum_{i=1}^{n_c} \Delta_{f_i} a^p_{uf_i} \right) y^p_d \le \mathrm{K}_u \le C^{\text{NODE}}_u \qquad u\in V$$

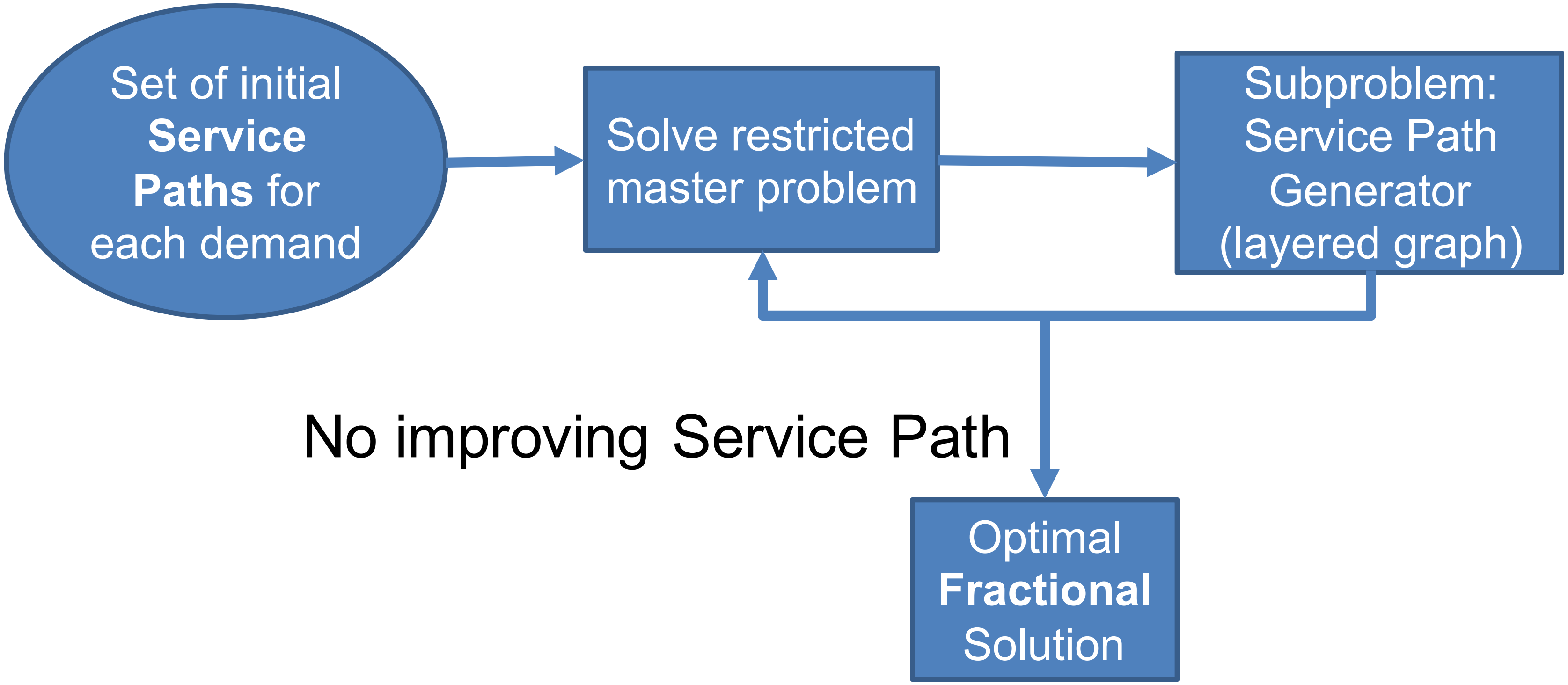
Variables for

- *Link State:* *ON or OFF*
- *Number of Active Cores per Node*
- *Service Path*: potential route for a request (path & placement)

Column generation on the *Service Path* variables

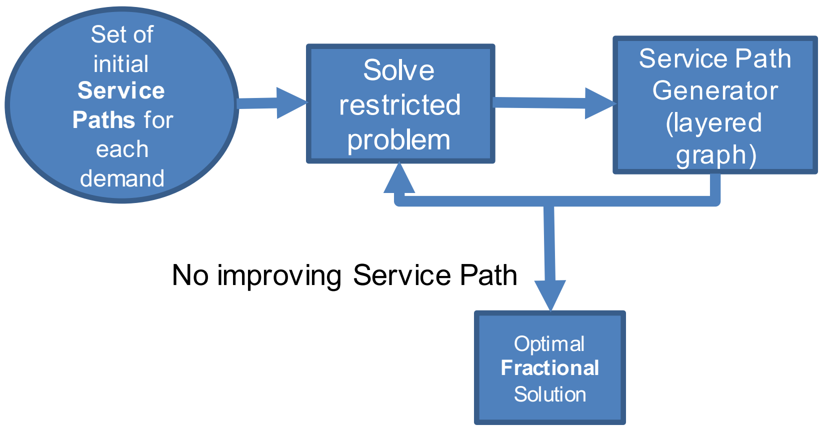
Generation of *Service Path* variables

Column generation works on Linear Program



Generation of *Service Path* variables

Column generation works on Linear Program



1. Transform LP to ILP
2. Solve **ILP**

LP optimal value gives **lower bound**

Integrality gap (ratio LP-ILP) gives quality of ILP solution

Improving the gap: CG-cuts

$$\sum_{d=(u_s,u_d,c)\in D} \sum_{p\in P^c_{sd}} D^c_{sd}\, \delta^p_{uv}\, y^p_d \le \boxed{x_{uv}}\, C^{\text{LINK}}_{uv}$$

Creates big gap

All to all traffic implies:

- At least one link active per node $\quad \sum_{v\in N^+(u)} x_{uv} \ge 1 \qquad u \in V$
- Both arcs share the same state so minimum network is a tree $\quad \sum_{(u,v)\in A} x_{uv} \ge n-1$

Improving the gap: CG-cut+

$$\sum_{d=(u_s,u_d,c)\in D} \sum_{p\in P^c_{sd}} D^c_{sd}\,\delta^p_{uv}\,y^p_d \leq x_{uv} C^{\text{LINK}}_{uv}$$

Creates big gap

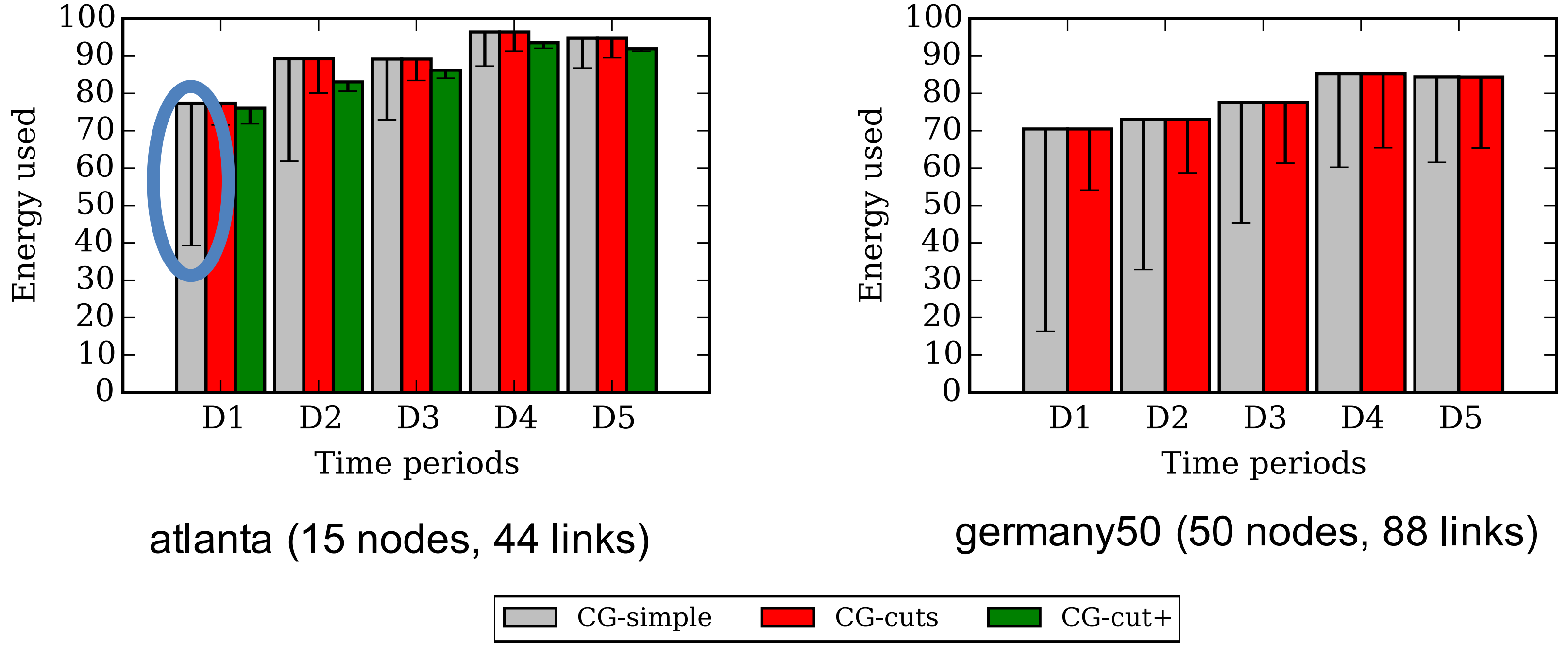
For each demand, the sum of its paths is equal

$$\sum_{p\in P^c_{sd}} y^p_d = 1 \implies \sum_{p\in P^c_{sd}} \gamma^p_{uv} y^p_d \leq 1$$

Path p uses link (u, v)

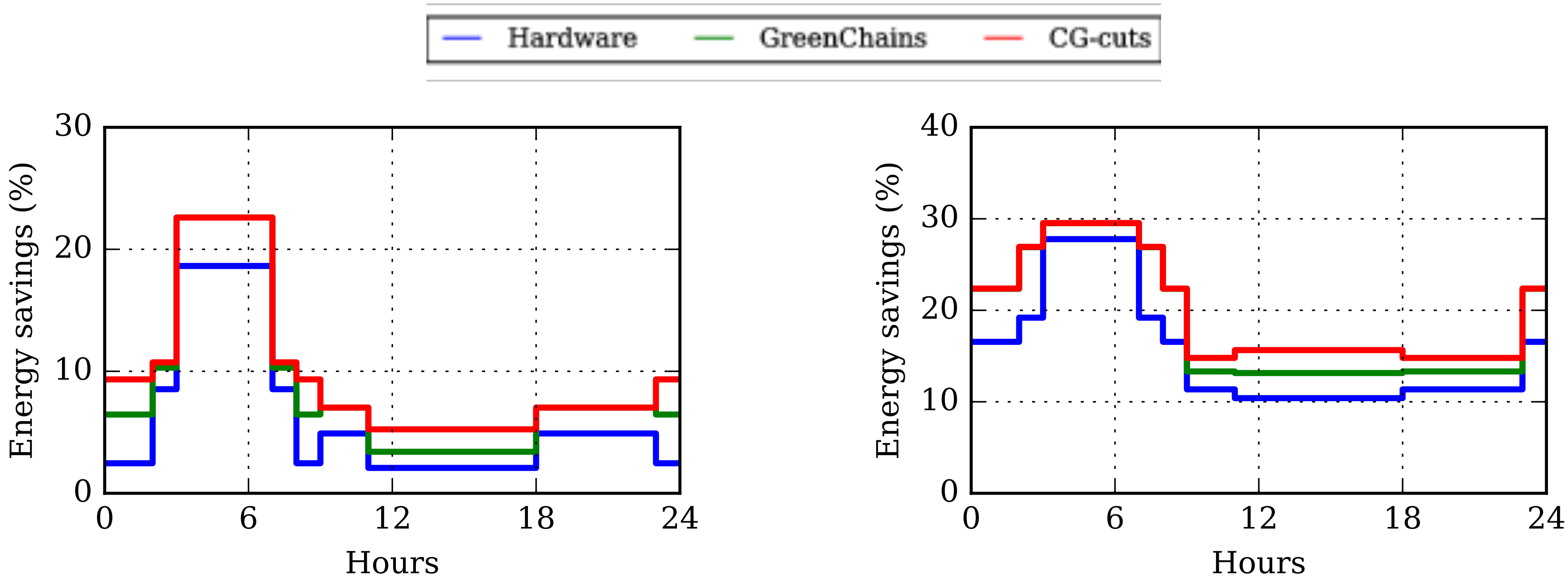
$$x_{uv} \geq \sum_{p\in P^c_{sd}} \gamma^p_{uv} y^p_d \qquad \forall (u,v)\in A, d\in D$$

Results: Integrality gap



- Both sets of cuts **improve** the **integrality** gap
- CG-cuts+ improve solution but **not scalable**

Results: Energy savings



atlanta (15 nodes, 44 links)

germany50 (50 nodes, 88 links)

- Hardware (SDN+ middlebox) only provides 18 to 51% energy savings
- NFV (SDN+NFV) provides an extra 4 to 12%

In this thesis

- SDN
  - EARC
    - Compression provides close savings to classic EAR
    - MINNIE: no noticeable impact on performance
  - Hybrid networks
    - SENAtoR: Backup tunnels + Smooth extinction of links
    - EAR with no losses
- NFV & SFC
  - First scalable mathematical formulation
  - NFV helps to reduce energy consumption

Perspectives

- QoS/QoE, resiliency/reliability
  - currently working on SFC w/ protection
- Multiple controllers (placement, activation)
- SFC extensions
  - Partial order
  - Affinity
- PostDoc in Concordia, Montreal, Canada with Brigitte Jaumard

Thank you for your attention

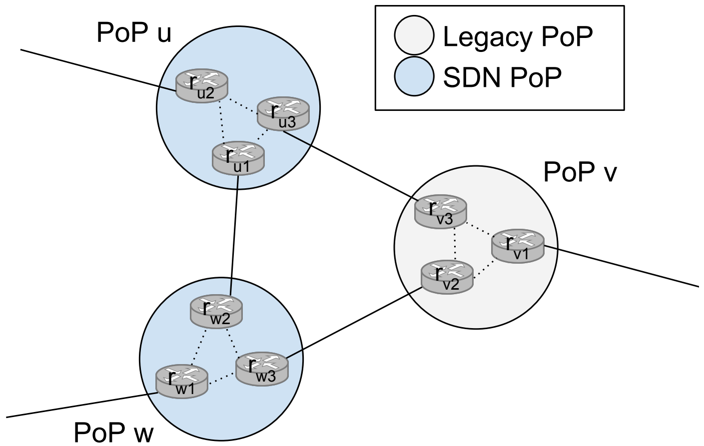
Result: Ratio, losses & # compressions

	Compression				
	None	at 500	at 1000	at 2000	when full
Average compression ratio	-	83.21%	82.19%	81.55%	81.44%
Packet losses (%)	6.25×10^{-6}	0.003	5.65×10^{-4}	2.83×10^{-5}	3.7×10^{-4}
# compressions	-	16 594	95	28	20

- Average compression ratio >80% (at least 76%)
- Compression has no significant impact on losses
  - Except when the threshold is too low

Hybrid Energy Aware Routing (hEAR)

- Network G=(V, A)
- Set of requests D, between s_i and t_i with bandwidth d_i
- Link capacities
- Forwarding table capacities
- **SDN budget**
- **OSPF next hops**
- **Set of backup tunnels**



Satisfy all requests (find a path) *and* minimize energy consumption
while respecting link capacities using backup tunnels and *k* SDN nodes

ILP Formulation (CG-simple)

$$\min \underbrace{\sum_{(u,v)\in A} P^{\text{IDLE}}_{uv} x_{uv}}_{\text{link switch on energy}} + \underbrace{\sum_{(u,v)\in A} \sum_{p \in P^c_{sd}} \delta^p_{uv} \left(\sum_{d=(u_s,u_d,c)\in D} \frac{D^c_{sd}}{C^{\text{LINK}}_{\ell}} \mathrm{P}^{\max}_{uv} \right) y^p_d}_{\text{link bandwidth energy}} + \underbrace{\sum_{u \in V} \mathrm{P}_u \, \mathrm{K}_u}_{\text{node resource energy}}$$

One path per demand: $\sum_{p \in P^c_{sd}} y^p_d = 1 \qquad (u_s, u_d) \in \mathcal{SD}, c \in C_{sd}$

Link capacity: $\sum_{d=(u_s,u_d,c)\in D} \sum_{p \in P^c_{sd}} D^c_{sd} \, \delta^p_{uv} \, y^p_d \le x_{uv} \, C^{\text{LINK}}_{uv} \qquad (u,v) \in A$

Node capacity: $\sum_{d \in D} \sum_{p \in P^c_{sd}} D^c_{sd} \left(\sum_{i=1}^{n_c} \Delta_{f_i} a^p_{u f_i} \right) y^p_d \le \mathrm{K}_u \le C^{\text{NODE}}_u \qquad u \in V$

Variables for

- *Link State: ON or OFF*
- *Number of Active Cores per Node*
- *Service Path*: potential route for a request (path & placement)

ILP Formulation (CG-simple)

$$\min \underbrace{\sum_{(u,v)\in A} P^{\text{IDLE}}_{uv} x_{uv}}_{\substack{\text{link switch}\\ \text{on energy}}} + \underbrace{\sum_{(u,v)\in A} \sum_{p \in P^c_{sd}} \delta^p_{uv} \left(\sum_{d=(u_s,u_d,c)\in D} \frac{D^c_{sd}}{C^{\text{LINK}}_{\ell}} \mathrm{P}^{\max}_{uv} \right) y^p_d}_{\text{link bandwidth energy}} + \underbrace{\sum_{u\in V} \mathrm{P}_u \mathrm{K}_u}_{\text{node resource energy}}$$

One path per demand: $\sum_{p\in P^c_{sd}} y^p_d = 1 \qquad (u_s,u_d)\in \mathcal{SD}, c \in C_{sd}$

Link capacity: $\sum_{d=(u_s,u_d,c)\in D} \sum_{p\in P^c_{sd}} D^c_{sd}\, \delta^p_{uv}\, y^p_d \le x_{uv}\, C^{\text{LINK}}_{uv} \qquad (u,v)\in A$

Node capacity: $\sum_{d\in D} \sum_{p\in P^c_{sd}} D^c_{sd} \left(\sum_{i=1}^{n_c} \Delta_{f_i} a^p_{uf_i} \right) y^p_d \le \mathrm{K}_u \le C^{\text{NODE}}_u \qquad u\in V$

Variables for

- *Link State: ON or OFF*
- *Number of Active Cores per Node*
- *Service Path*: potential route for a request (path & placement)

ILP Formulation (CG-simple)

$$\min \underbrace{\sum_{(u,v)\in A} P^{\text{IDLE}}_{uv} x_{uv}}_{\text{link switch on energy}} + \underbrace{\sum_{(u,v)\in A} \sum_{p\in P^c_{sd}} \delta^p_{uv} \left(\sum_{d=(u_s,u_d,c)\in D} \frac{D^c_{sd}}{C^{\text{LINK}}_{\ell}} \mathrm{P}^{\max}_{uv} \right) y^p_d}_{\text{link bandwidth energy}} + \underbrace{\sum_{u\in V} \mathrm{P}_u \mathrm{K}_u}_{\text{node resource energy}}$$

One path per demand: $$\sum_{p\in P^c_{sd}} y^p_d = 1 \qquad (u_s,u_d)\in \mathcal{SD}, c\in C_{sd}$$

Link capacity: $$\sum_{d=(u_s,u_d,c)\in D} \sum_{p\in P^c_{sd}} D^c_{sd}\, \delta^p_{uv}\, y^p_d \le x_{uv}\, C^{\text{LINK}}_{uv} \qquad (u,v)\in A$$

Node capacity: $$\sum_{d\in D} \sum_{p\in P^c_{sd}} D^c_{sd} \left(\sum_{i=1}^{n_c} \Delta_{f_i} a^p_{uf_i} \right) y^p_d \le \mathrm{K}_u \le C^{\text{NODE}}_u \qquad u\in V$$

Variables for

- *Link State: ON or OFF*
- *Number of Active Cores per Node*
- *Service Path*: potential route for a request (path & placement)

ILP Formulation (CG-simple)

$$\min \underbrace{\sum_{(u,v)\in A} P_{uv}^{\text{IDLE}} x_{uv}}_{\text{link switch on energy}} + \underbrace{\sum_{(u,v)\in A} \sum_{p\in P_{sd}^c} \delta_{uv}^p \left(\sum_{d=(u_s,u_d,c)\in D} \frac{D_{sd}^c}{C_\ell^{\text{LINK}}} \mathrm{P}_{uv}^{\max} \right) y_d^p}_{\text{link bandwidth energy}} + \underbrace{\sum_{u\in V} \mathrm{P}_u \, \mathrm{K}_u}_{\text{node resource energy}}$$

One path per demand: $$\sum_{p\in P_{sd}^c} y_d^p = 1 \qquad (u_s,u_d)\in \mathcal{SD}, c\in C_{sd}$$

Link capacity: $$\sum_{d=(u_s,u_d,c)\in D} \sum_{p\in P_{sd}^c} D_{sd}^c \, \delta_{uv}^p \, y_d^p \le x_{uv} \, C_{uv}^{\text{LINK}} \qquad (u,v)\in A$$

Node capacity: $$\sum_{d\in D} \sum_{p\in P_{sd}^c} D_{sd}^c \left(\sum_{i=1}^{n_c} \Delta_{f_i} a_{uf_i}^p \right) y_d^p \le \mathrm{K}_u \le C_u^{\text{NODE}} \qquad u\in V$$

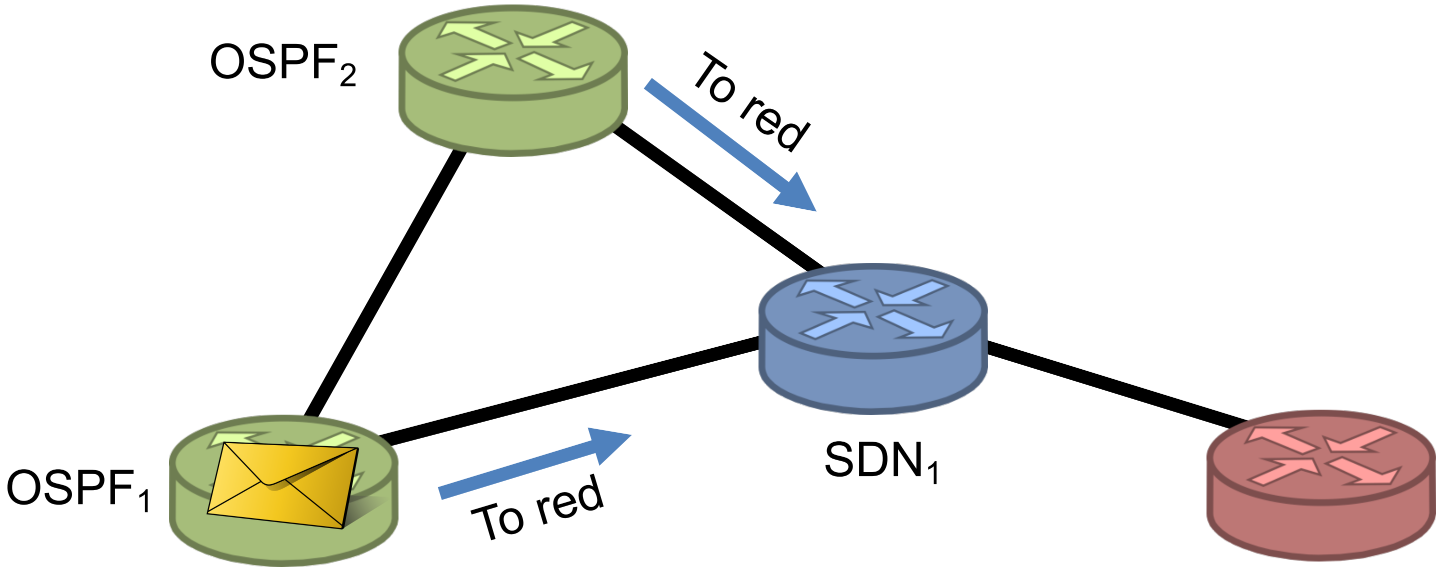
Variables for

- *Link State: ON or OFF*
- *Number of Active Cores per Node*
- *Service Path*: potential route for a request (path & placement)

Column generation on the *Service Path* variables

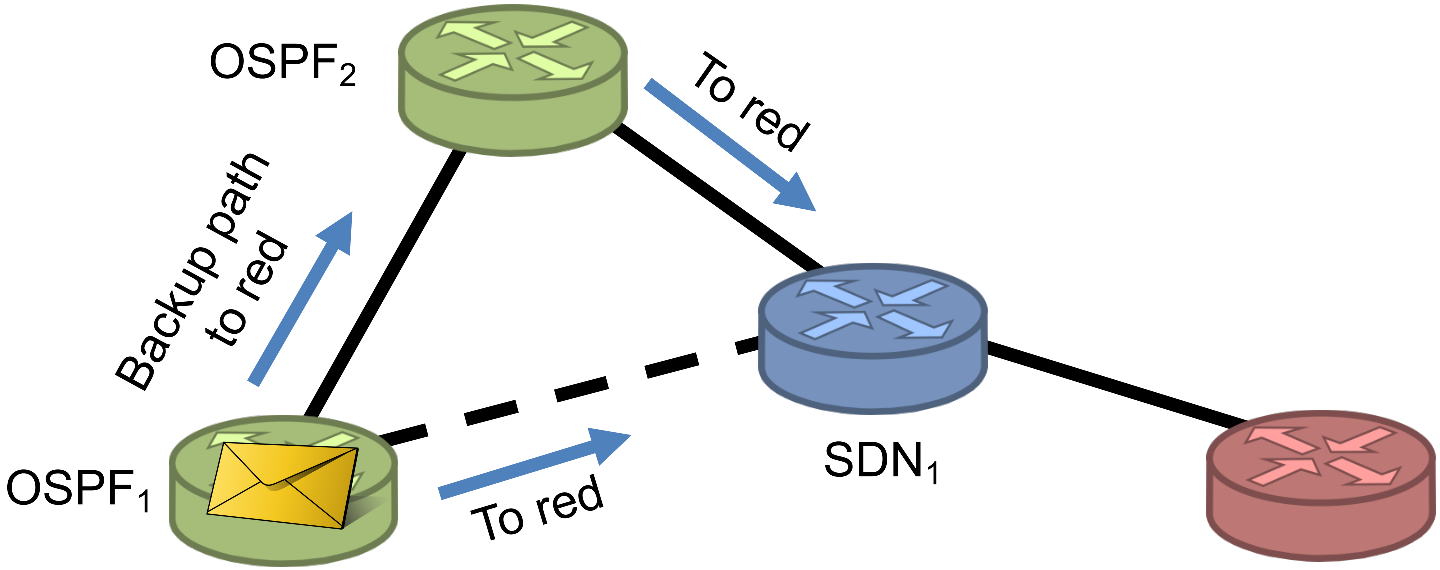
Use the tunnel if the road is closed

Use backup tunnels provided by legacy routers to redirect traffic [citation needed]



Use the tunnel if the road is closed

Use backup tunnels provided by legacy routers to redirect traffic [citation needed]



Stop saying hello to you

- OSPF uses *HELLO* packets, at regular intervals, to notify neighbors of their existence
  - 3 missing *HELLO* leads to a failure detection.
  - All data packets thus can be lost during this interval

- Before shutdown, an SDN switch stops sending HELLO packets but still listens for data packets
  - No packets are lost

Contributions

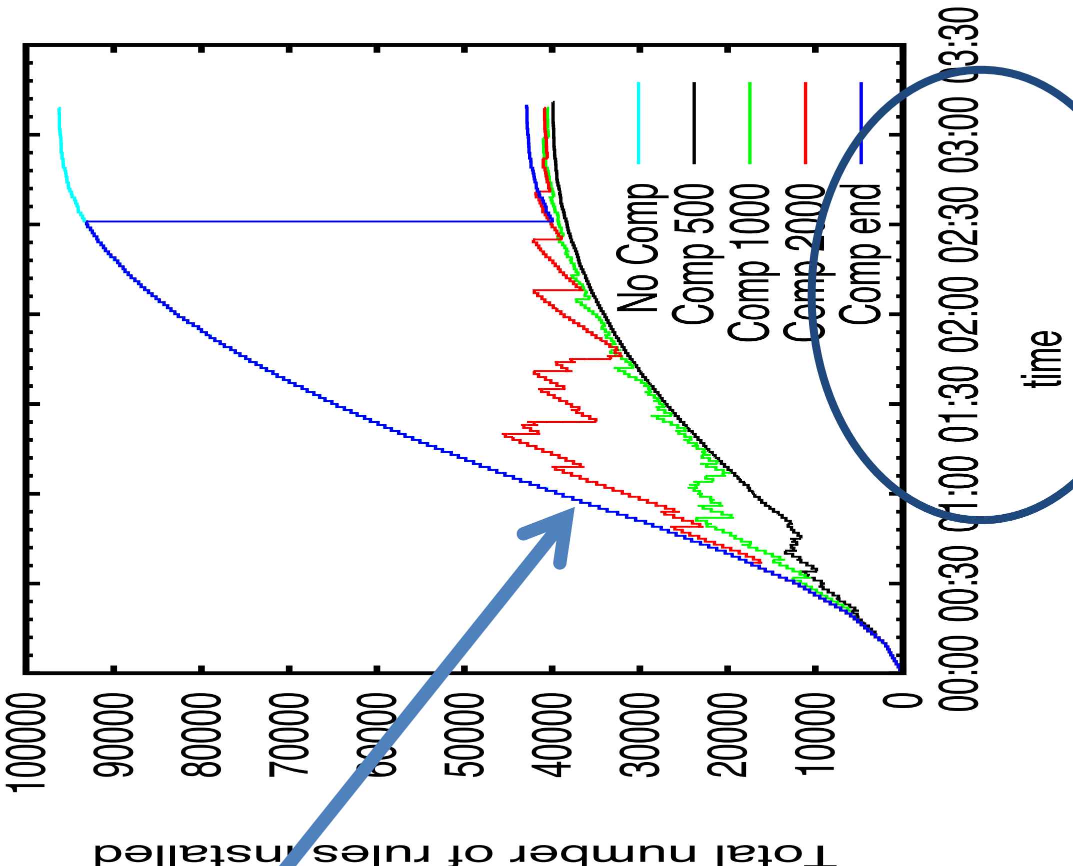
- Study the number of servers that can be deployed with limited number of rules
- Simulations on various data center topologies (fat tree, VL2, DCell, BCube)
- Experiments on a **HP SDN-capable switch** (65536 software rules, 3500 hardware rules)

Simulation: MINNIE & 1000 servers topologies

Topology	servers #	switches #	links #	Avg ports #	# flow per switch		Rule w/ comp #		Average Comp. Ratio	Computation time in average (ms)	
					Max	Average	Max	Average		Paths	Comp.
Group 1											
$k = 4$ Fat-Tree (64)	1024	20	1056	54.4	454 244	216 268	999	446	∼ 99.60	0.17	13
$k = 8$ Fat-Tree (8)	1024	80	1280	19.2	649 044	61 030	999	323	∼ 99.61	0.21	7
$k = 16$ Fat-Tree (1)	1024	320	3072	16	630 998	15 897	999	303	∼ 98.42	0.30	5
VL2(16, 16, 14)	896	88	384	16	261 266	42 906	1000	673	∼ 97.90	0.15	4
VL2(8, 8, 64)	1024	28	612	∼ 41.1	423 752	161 499	1000	799	∼ 99.45	0.19	11
VL2(16, 16, 16)	1024	88	1152	∼ 17.5	276 575	56 040	1000	648	∼ 98.39	0.18	4
Group 2											
DCell(32, 1)	1056	33	1584	∼ 2.91	63 787	4893	1000	113	∼ 97.23	0.09	2
DCell(5, 2)	930	186	1860	∼ 3.33	11 995	5716	994	642	∼ 87.84	0.19	2
BCube(32, 1)	1024	64	2048	∼ 3.77	37 738	3734	999	329	∼ 86.04	0.19	2
BCube(10, 2)	1000	300	3000	∼ 4.62	10 683	4153	998	653	∼ 80.85	0.25	2
BCube(6, 3)	1296	864	5184	4.8	7852	5184	991	831	∼ 83.18	0.49	4

- Around 1 million flows on each topologies
- With only 1000 rules
- Compression ratio between 80 and 99%

Experiment: Number of rules over time



80% compression ratio

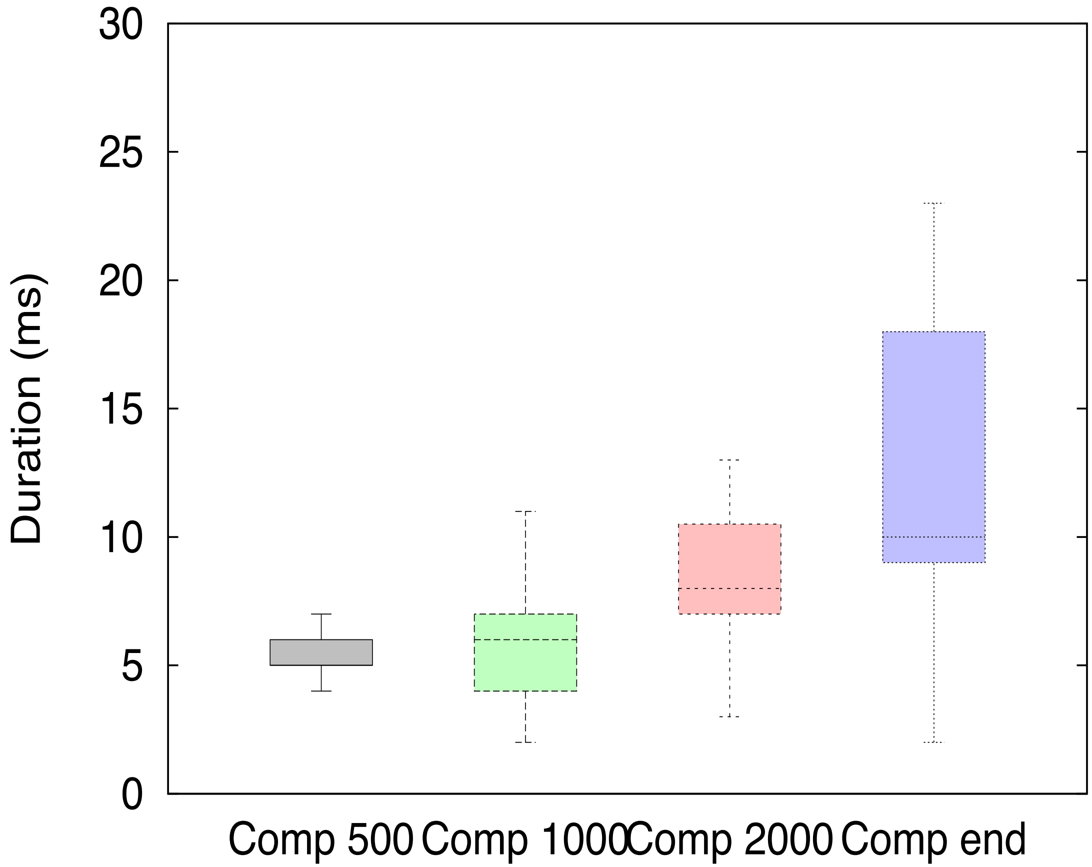
Compression event

Experiment: Delay

Delay :

- **increases** over time without compression
- **stays constant** when compressing at 1000
- goes **haywire** when compression at 500

Experiment: Compression Duration



Compression + table modification

Energy model

$$\min \underbrace{\sum_{(u,v)\in A} P_{uv}^{\mathrm{IDLE}} x_{uv}}_{\substack{\text{link switch}\\ \text{on energy}}} + \underbrace{\sum_{(u,v)\in A} \sum_{p\in P_{sd}^{c}} \delta_{uv}^{p} \left(\sum_{d=(u_s,u_d,c)\in D} \frac{D_{sd}^{c}}{C_{\ell}^{\mathrm{LINK}}} \mathrm{P}_{uv}^{\mathrm{MAX}} y_d^p \right)}_{\text{link bandwidth energy}}$$

$$+ \underbrace{\sum_{u\in V} \mathrm{P}_u \, \mathrm{K}_u}_{\text{node resource energy}}$$

- Hybrid model for links
- Node consumption linear w.r.t. the number of cores

Experiment: Packet losses

	Compression threshold				
	None	500	1000	2000	When full
# of compressions	0	16 594	95	28	20
% packet loss	6.25×10^{-6}	0.003	5.65×10^{-4}	2.83×10^{-5}	3.7×10^{-4}

No significant packet losses except for 500

Direction-Based Algorithm

Compress using source aggregation, destination aggregation or default rule

$\Rightarrow$ Take the best table

1. For each source (resp. destination), get the most occuring ports
   $\Rightarrow$ Gives the default port of the source
2. Get the most occuring port in the most occuring ports
   $\Rightarrow$ Gives the default port
3. Add the default rules and wildcard rules with lowest priority
4. Add the original rules that don’t match any aggregation rules

Direction-Based Algorithm

Compress using source aggregation, destination aggregation or default rule

$\Rightarrow$ Take the best table

Flow	Output port
(0, 4)	Port-4
(0, 5)	Port-5
(0, 6)	Port-5
(1, 4)	Port-6
(1, 5)	Port-4
(1, 6)	Port-6
(2, 4)	Port-4
(2, 5)	Port-5
(2, 6)	Port-6

→

	4	5	6
0	4	5	5
1	6	4	6
2	4	5	6

Direction-Based Algorithm

Compress using **source aggregation**, destination aggregation or default rule

⇒ Take the best table

- Get the most occuring port for each source

	4	5	6
0	4	5	5
1	6	4	6
2	4	5	6

$P_0 = \{5\}$

Direction-Based Algorithm

Compress using **source aggregation**, destination aggregation or default rule

$\Rightarrow$ Take the best table

- Get the most occuring port for each source

	4	5	6	
0	4	5	5	$P_0 = \{5\}$
1	6	4	6	$P_1 = \{6\}$
2	4	5	6	$P_2 = \{4, 5, 6\}$

Direction-Based Algorithm

Compress using **source aggregation**, destination aggregation or default rule

$\Rightarrow$ Take the best table

- Get the most occuring port in the set of most occuring ports (default rule)

	4	5	6
0	4	5	5
1	6	4	6
2	4	5	6

$D = \{5\}$

$P_0 = \{5\}$

$P_1 = \{6\}$

$P_2 = \{4, 5, 6\}$

Direction-Based Algorithm

Compress using **source aggregation**, destination aggregation or default rule

$\Rightarrow$ Take the best table

- Build the table

$D = \{5\}$

- Add with lowest priority (*, *, 5)

$P_0 = \{5\}$

- No rule (overlap with default)

$P_1 = \{6\}$

- Add (1, *, 6)

$P_2 = \{4, \mathbf{5}, 6\}$

- No rule (overlap with default)

Forwarding table :

(1, *, 6)
(*, *, 5)

Direction-Based Algorithm

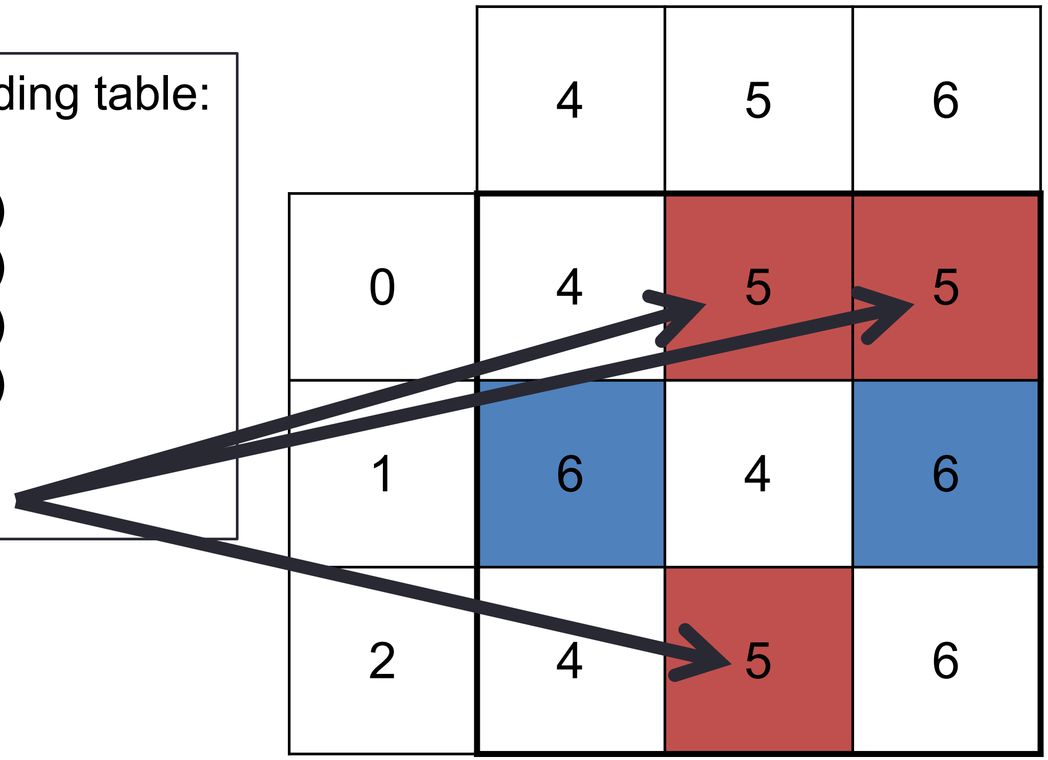
Compress using **source aggregation**, destination aggregation or default rule

⇒ Take the best table

- Build the table

Forwarding table:

(0, 4, 4)
(1, 4, 4)
(2, 4, 4)
(2, 6, 6)
(1, *, 6)
(*, *, 5)



Direction-Based Algorithm

Compress using **source aggregation**, destination aggregation or default rule

$\Rightarrow$ Take the best table

- Build the table

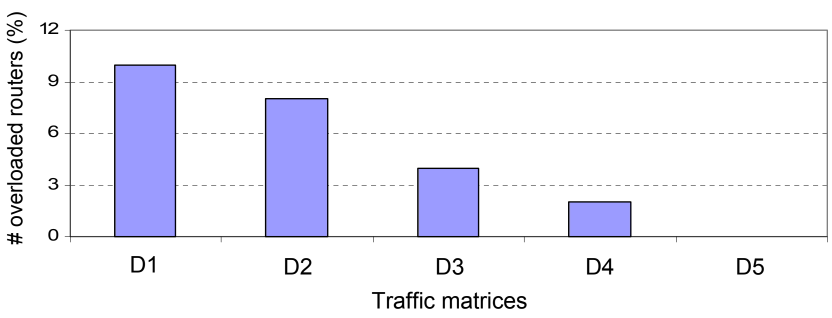
Forwarding table:

(0, 4, 4)
(1, 4, 4)
(2, 4, 4)
(2, 6, 6)
(1, *, 6)
(*, *, 5)

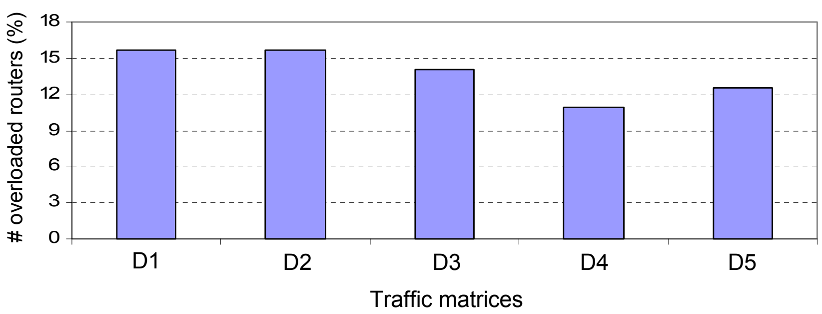
	4	5	6
0	4	5	5
1	6	4	6
2	4	5	6

More rules for less energy

- Shutting down links increases shortest paths
  - Increase in number of required rules



germany50 (50 nodes, 88 links)



ta2 (65 nodes, 81 links)

Direction-Based Algorithm

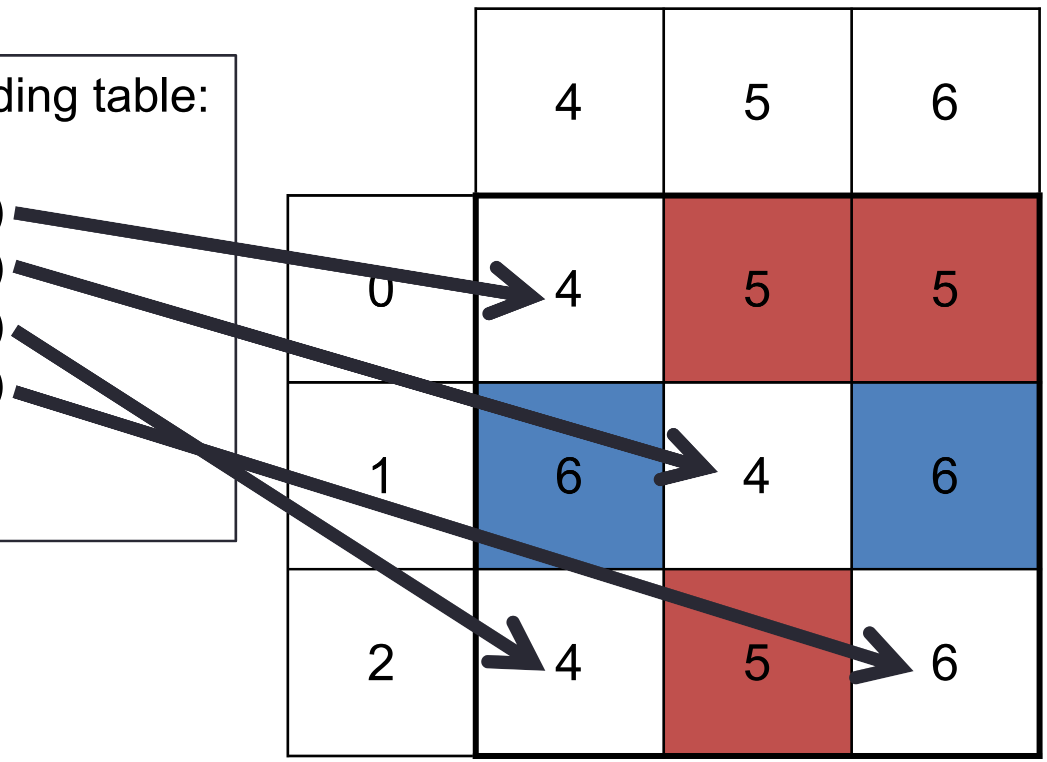
Compress using **source aggregation**, destination aggregation or default rule

⇒ Take the best table

- Build the table

Forwarding table:

(0, 4, 4)
(1, 4, 4)
(2, 4, 4)
(2, 6, 6)
(1, *, 6)
(*, *, 5)



Direction-Based Algorithm

Compress using source aggregation, destination aggregation or default rule

$\Rightarrow$ Take the best table

Flow	Output port
$(0, 4)$	Port-4
$(1, 5)$	Port-4
$(2, 4)$	Port-4
$(2, 6)$	Port-6
$(1, *)$	Port-6
$(*, *)$	Port-5

Source

Flow	Output port
$(1, 4)$	Port-6
$(1, 5)$	Port-4
$(0, 6)$	Port-5
$(*, 4)$	Port-4
$(*, 5)$	Port-5
$(*, *)$	Port-6

Destination

Flow	Output port
$(0, 5)$	Port-5
$(0, 6)$	Port-5
$(1, 4)$	Port-6
$(1, 6)$	Port-6
$(2, 5)$	Port-5
$(2, 6)$	Port-6
$(*, *)$	Port-4

Default

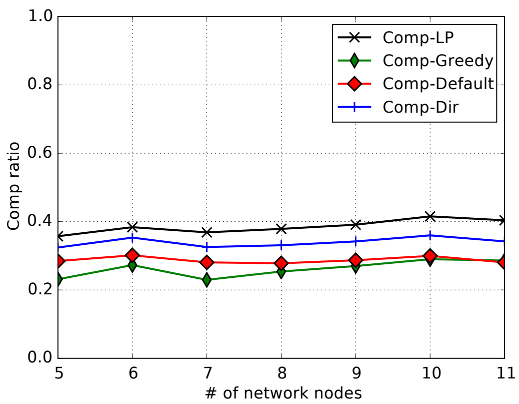
Other solutions

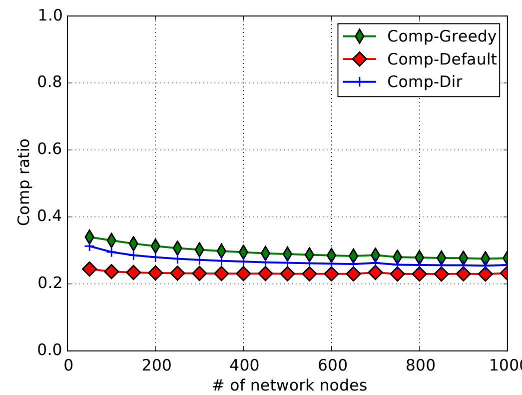
- Integer Linear Programing formulation
  - Not scalable
- Greedy algorithm
  - Each time, select the source or destination that can be compressed the best
- Just the default port
  - The third table of Direction-Based

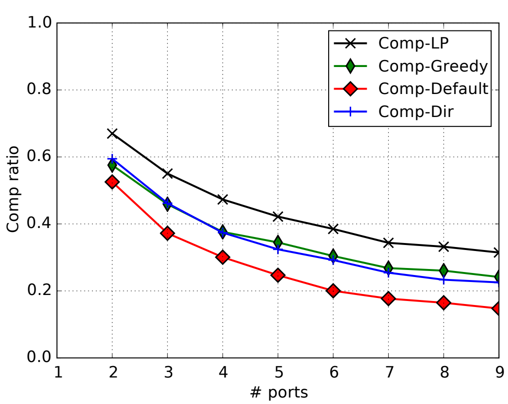
Data sets

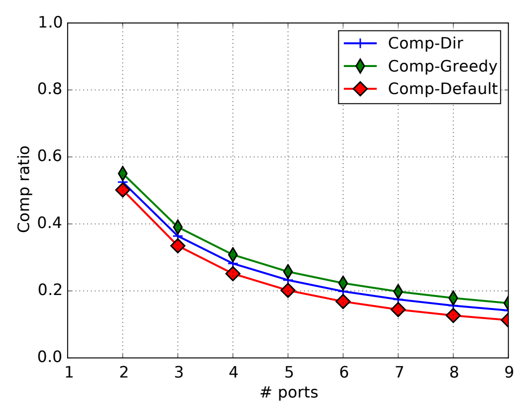
- Random tables
  - Density, number of sources/destinations, number of ports
- Network tables
  - SNDlib instances (atlanta, germany50, zib54, ta2)

Compression Ratio: Random tables



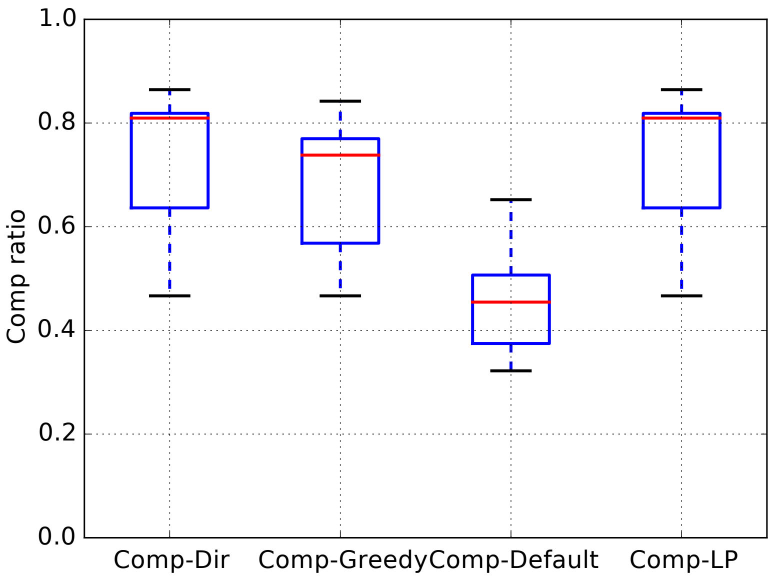




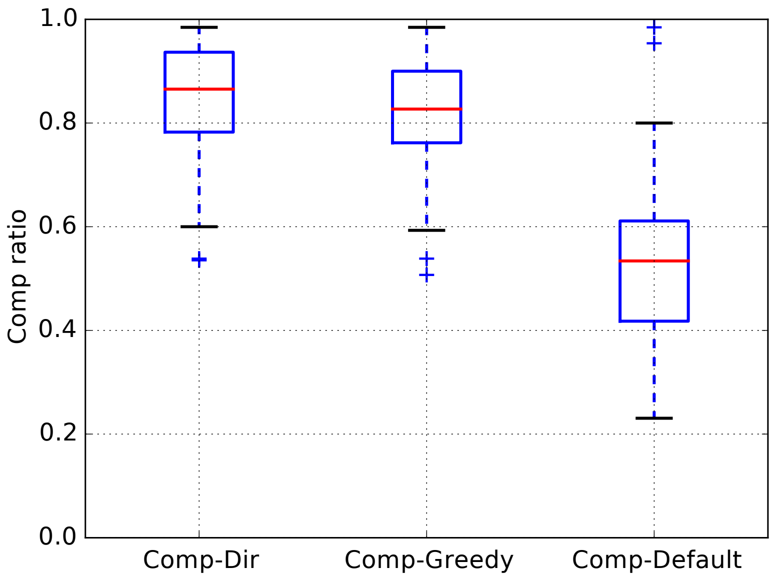


Greedy and Direction-Based have similar results

Compression Ratio: Network tables



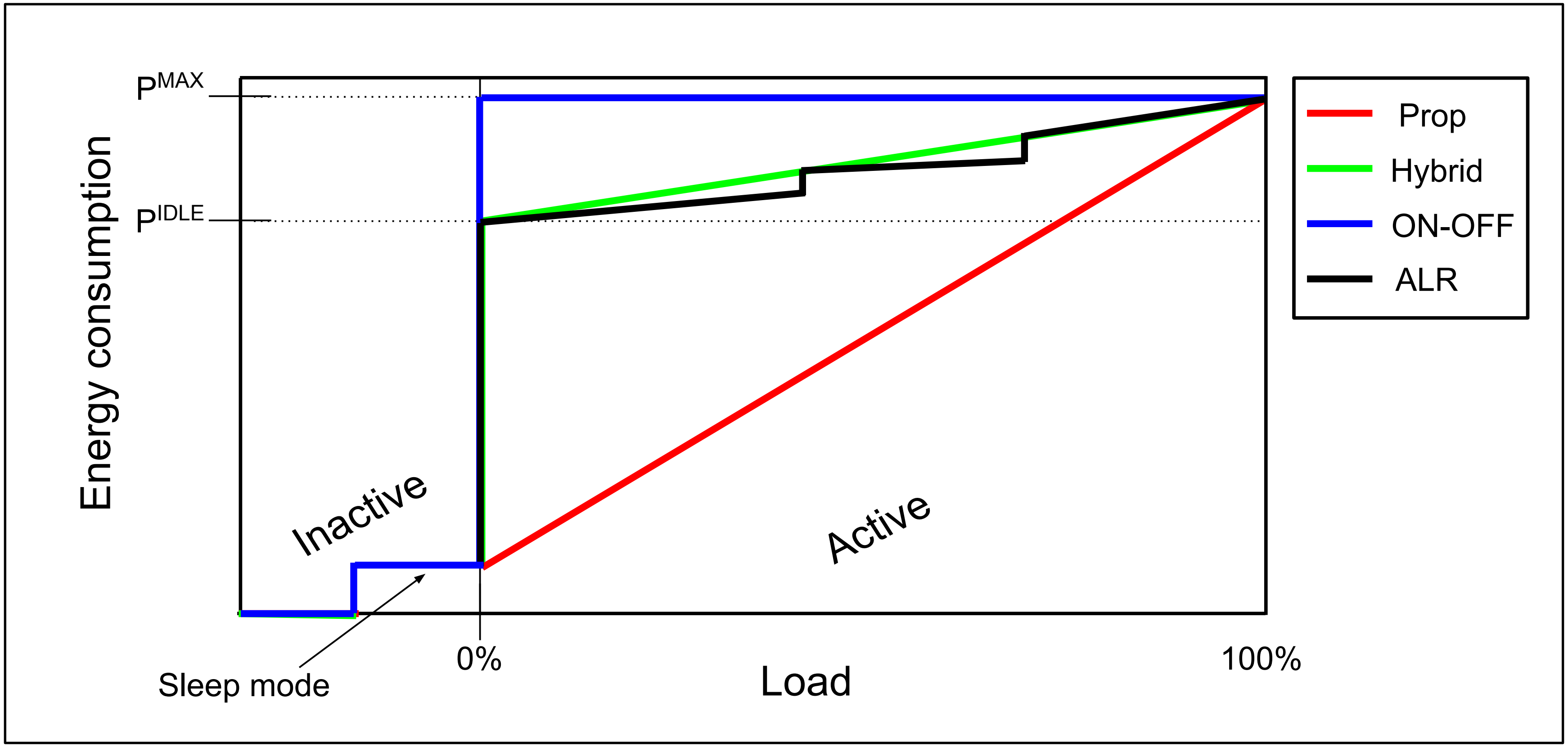
atlanta (15 nodes, 44 links)



ta2 (81 nodes, 162 links)

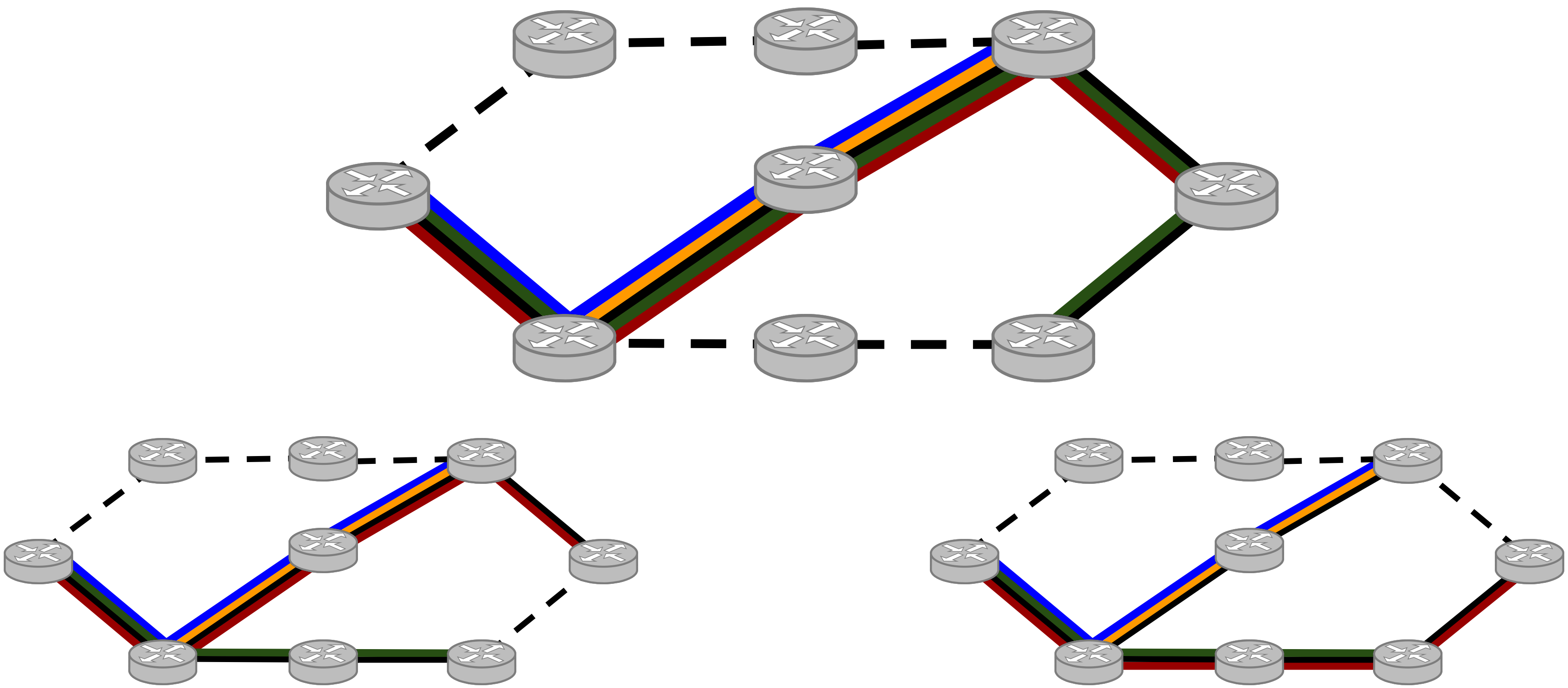
Direction-Based behaves better on network tables

Energy Proportionality



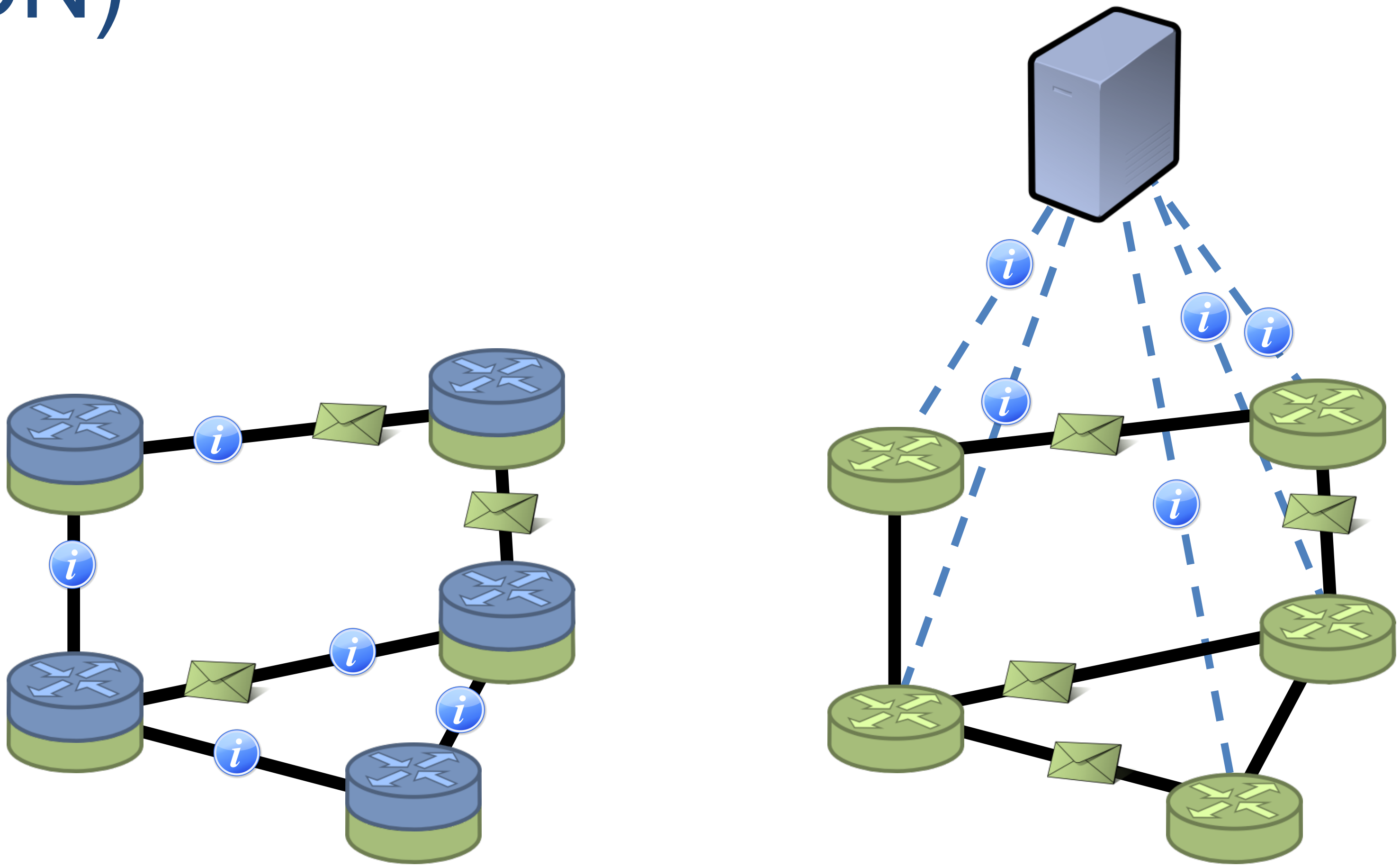
Network devices are not energy proportional [Chabarek et al., 2008]

Energy Aware Routing (EAR)

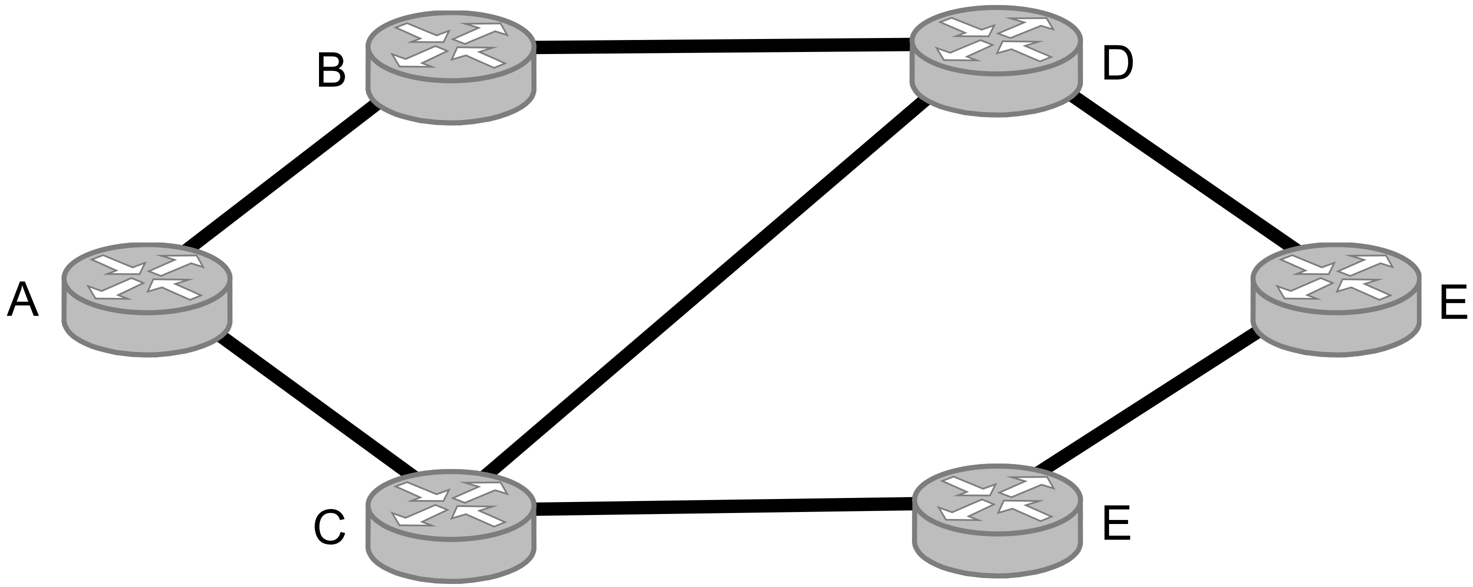


Satisfy the requests on the network with a subset of active devices

Legacy vs. Software Defined Networks (SDN)

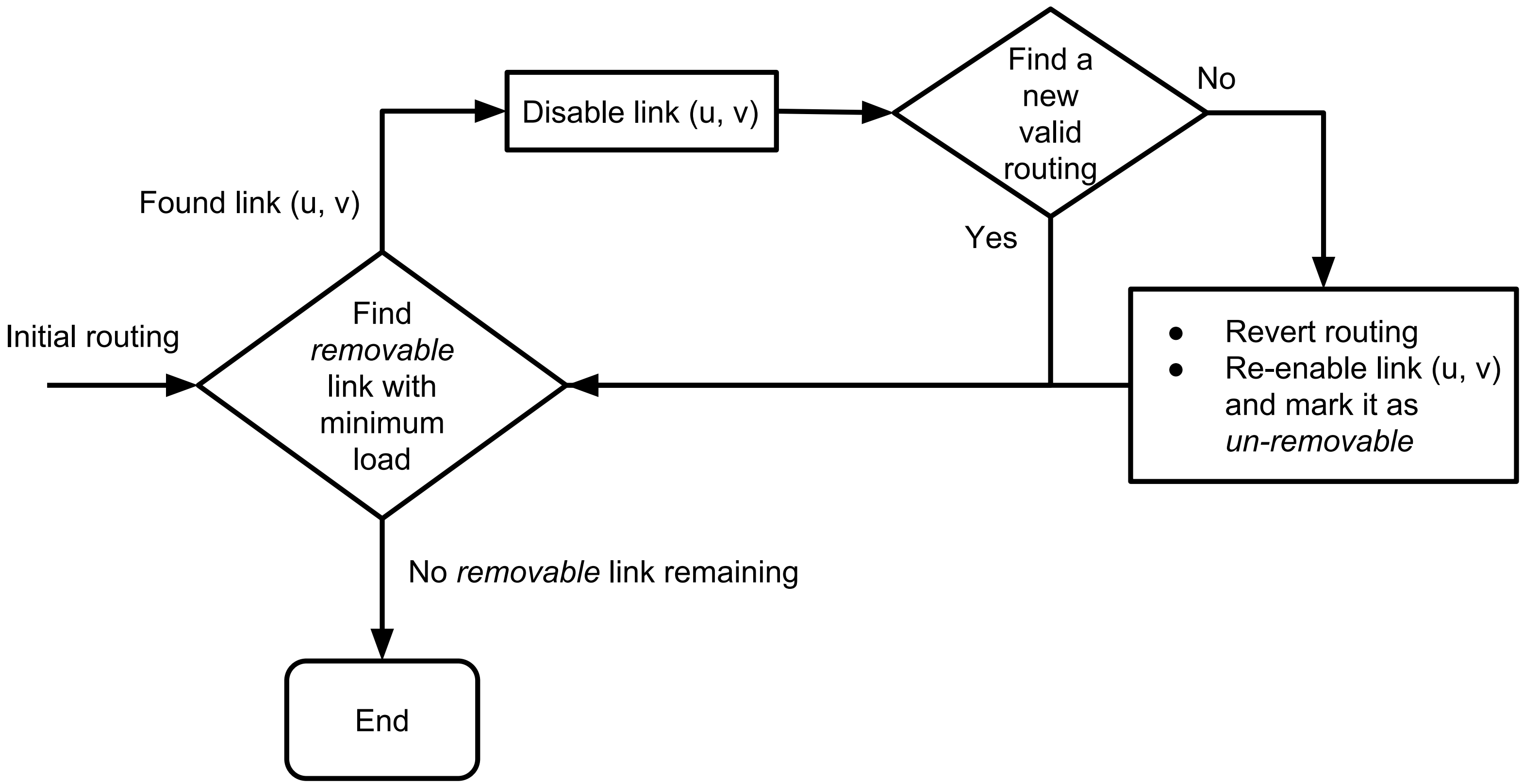


Energy Aware Routing (EAR)



Satisfy the requests on the network with a subset of active devices

Heuristic: Energy saving module



Heuristic: Routing module

Weighted shortest path on residual graph

Assignment of paths according to table and link usage

Compress tables when full

$$w_{uv} = \alpha \times w_{uv}^{r} + \beta \times w_{uv}^{l}$$

Table usage weight (0 if corresponding wildcard)

Link usage weight

Heuristic: Compression module

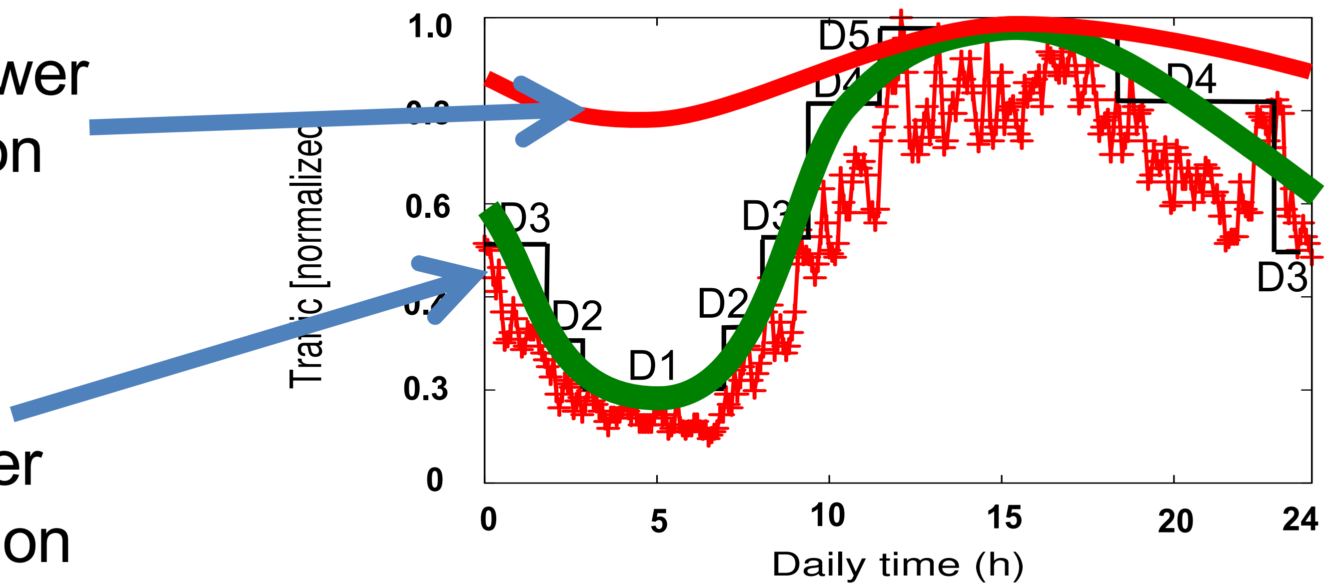
- default port only OR wildcard + default

- Propose several solutions to the compression problem
  - ILP formulations
  - **3-approximation algorithm**
  - Greedy heuristic
  - Default port

Energy Efficiency of Networks

Current power consumption

Ideal power consumption



Power Model Optimization

$$\min \sum_{(u,v)\in A} \left(P_{uv}^{\mathrm{IDLE}} x_{uv} + P_{uv}^{\mathrm{LOAD}} \frac{f_{uv}}{C_{uv}} \right)$$

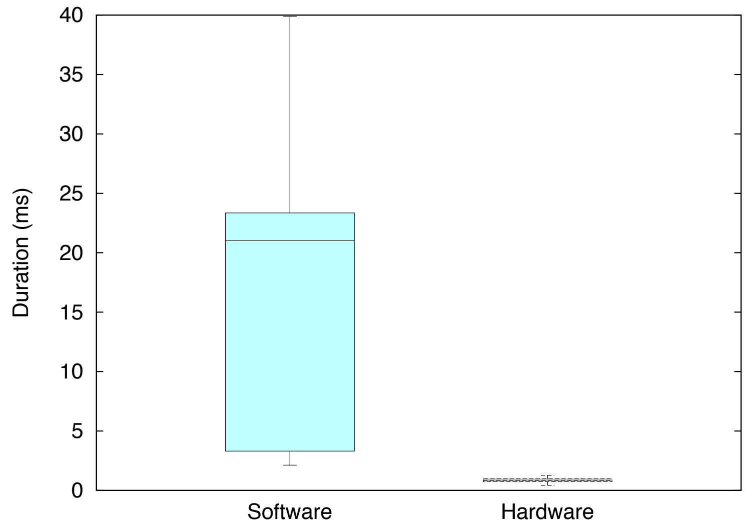
State of the link

Fraction of bandwidth used

Power used when idle

Additional power

Results: Hardware vs. Software



Performances of software forwarding table are way behind TCAM

Contributions

- Propose several solutions to the compression problem
  - ILP formulations, 3-approximation algorithm, greedy heuristic
- Study EAR with Compression
  - Heuristic with **joint** routing and compression
  - Compare *EARC* and classic *EAR*
- Validate on a HP SDN-capable switch (w/o energy)
  - Study end-to-end delay, packet losses, controller charge
  - Compare hardware and software rules

Results: Spike &failure mitigation