ENERGY EFFICIENT SOFTWARE DEFINED NETWORKS

Nicolas HUIN[1]
COATI, I3S/INRIA

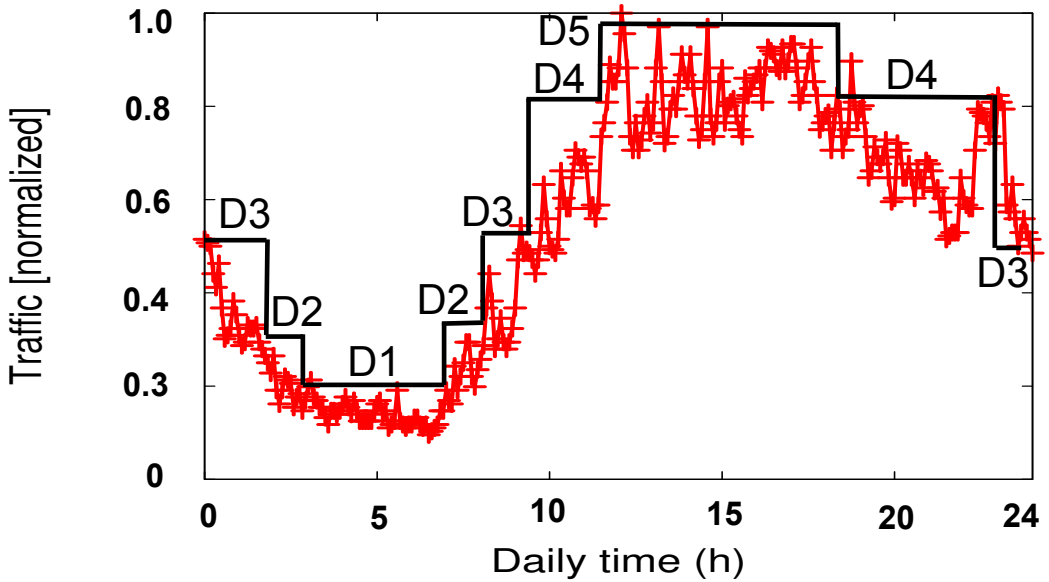
Green Days @ Sophia

26 Juin 2017

[1] En collaboration avec Christelle Caillouet, Frédéric Giroire, Frédéric Havet, Brigitte Jaumard, Dino Lopez, Joanna Moulierac, Myriana Rifai, Guillaume Urvoy-Keller

Energy efficiency of networks

- Yearly growth of **10%** for computer networks (**3%** global) [Van Heddeghem et al., '14]
- Over-provisioned to handle peak traffic

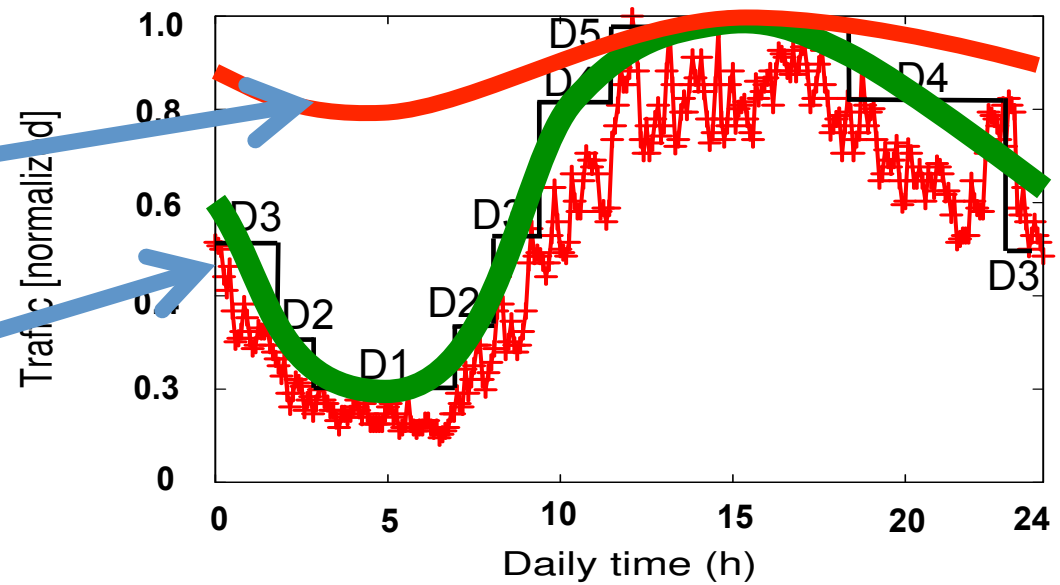


Energy efficiency of networks

- Yearly growth of **10%** for computer networks (**3%** global) [Van Heddeghem et al., '14]
- Over-provisioned to handle peak traffic

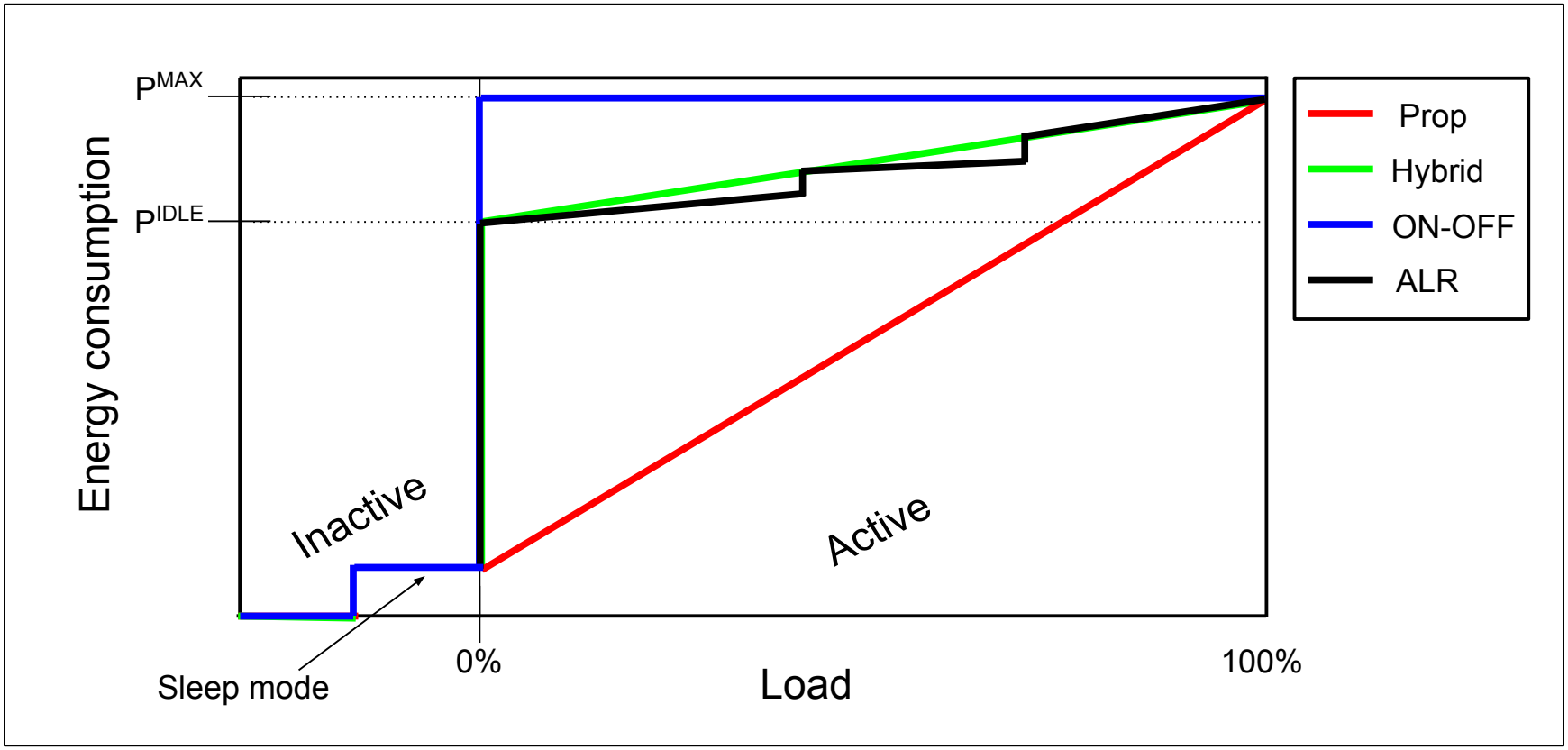
Current power consumption

Ideal power consumption



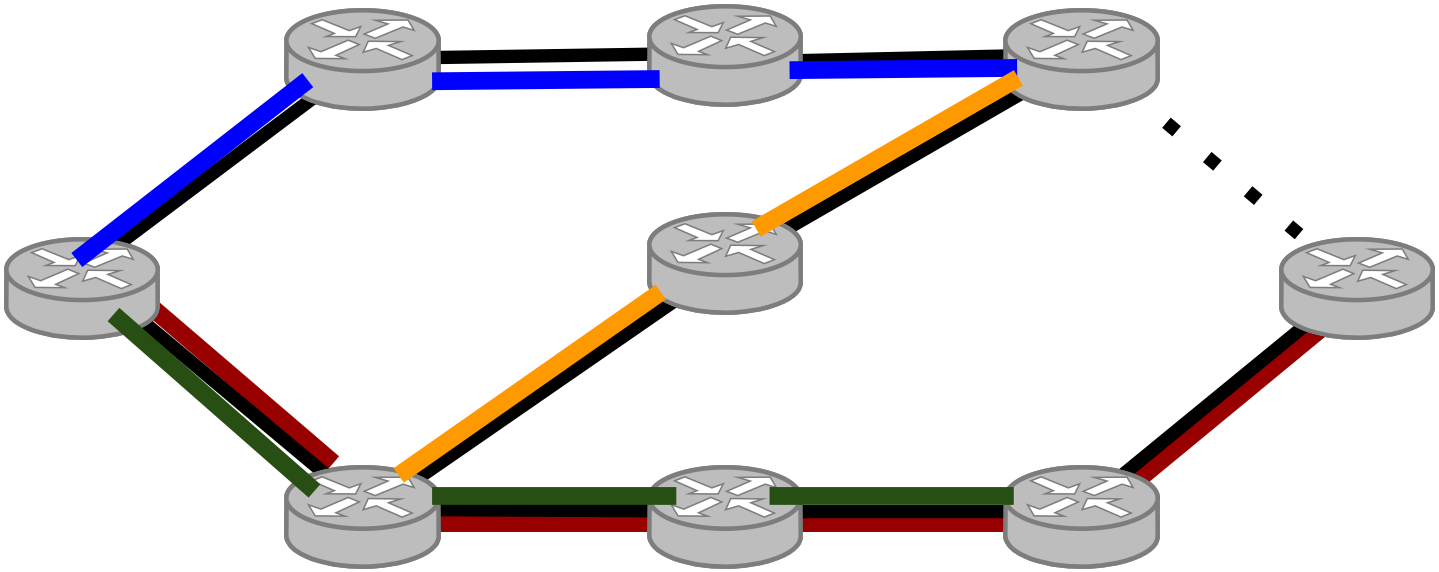
Energy Proportionality

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



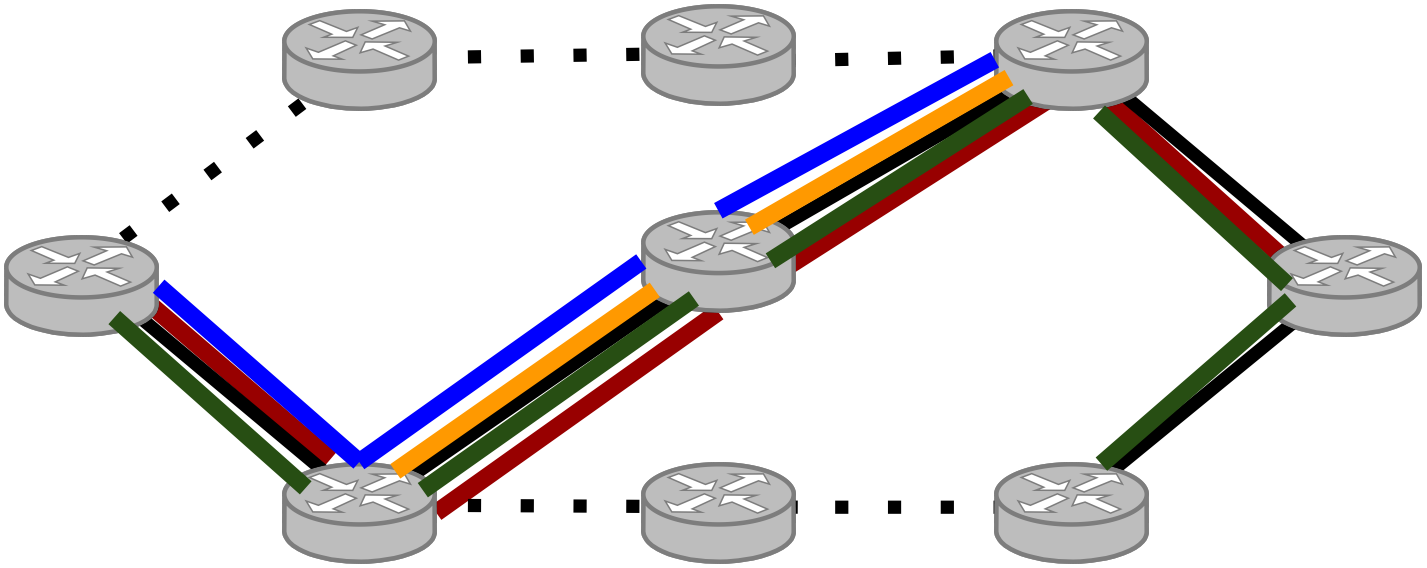
Energy Aware Routing (EAR)

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

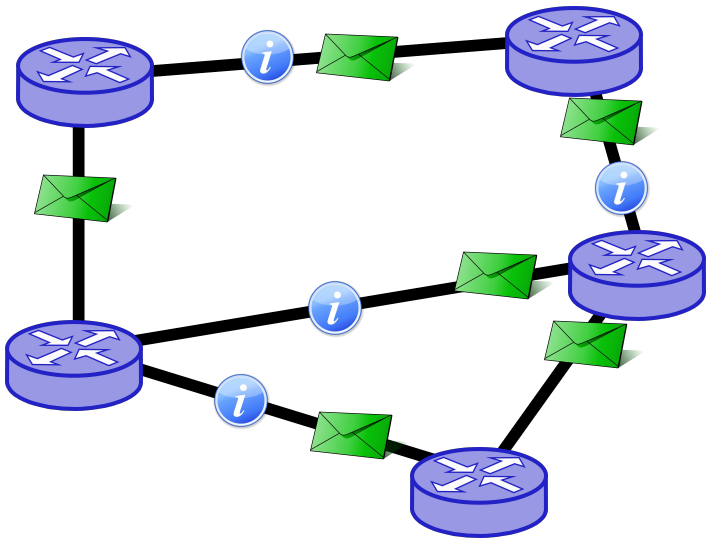


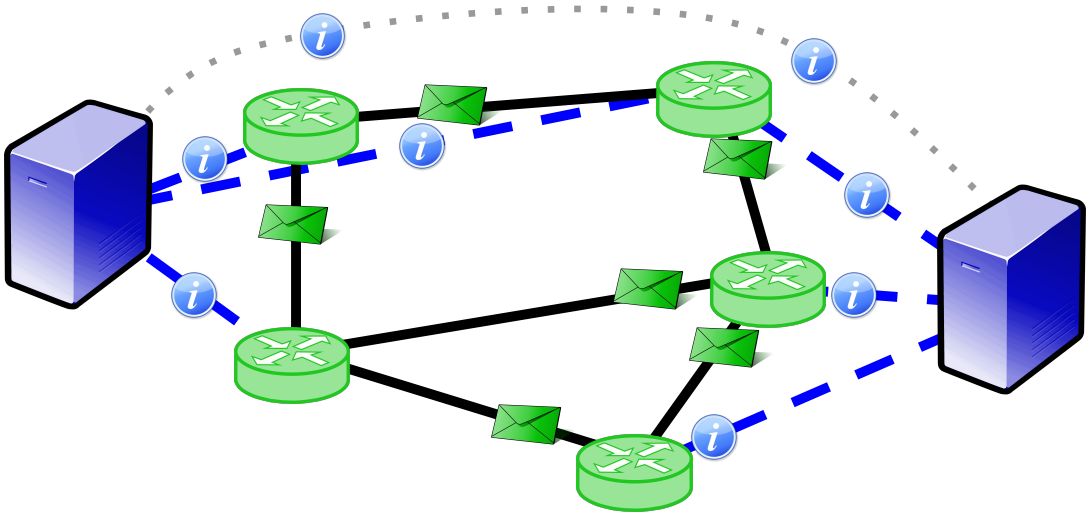
Energy Aware Routing (EAR)

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



Legacy vs. SDN





- Distributed control
- Manual configuration
- Closed systems

- Centralized control
- Apps running on the controller
- OpenFlow API

Enables network-wide green policies

During my thesis

- **Forwarding table constraints**
  - **The Compression Problem**
  - **EAR with Compression**
  - **MINNIE**
- Hybrid networks (presented by Myriana Rifai)
- Service Function Chaining
  - Provisioning
  - Energy efficiency

The first day there was OpenFlow

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

OpenFlow provides **per flow** routing -> More complex rules
Rules stored in *TCAM*, power hungry and with **limited size** (around 1k)

=> 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
- Use of aggregation rules
  - Our contribution

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 (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, *)

Contributions

- Propose several solutions to the compression problem
  - ILP, **3-approximation algorithm**, greedy heuristic
- Study EAR with Compression
  - Propose an efficient heuristic with **joint** routing and compression
  - Compare EARC with EAR without limits
- Validate on a HP SDN-capable switch (no energy)
  - Study end-to-end delay, packet losses
  - Compare hardware and software rules

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

⇒ 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

⇒ 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

$P_0 = \{5\}$

$P_1 = \{6\}$

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

Direction-Based Algorithm

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

⇒ Take the best table

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

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

$\boxed{D = \{5\}}$

$P_0 = \{5\}$

$P_1 = \{6\}$

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

Direction-Based Algorithm

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

⇒ 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)

	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

- 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

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)

	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

$\Rightarrow$ 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

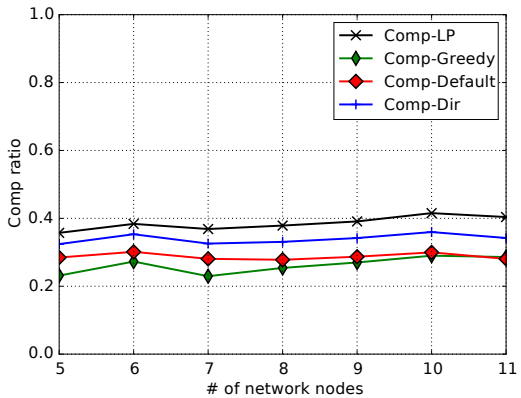
Other solutions

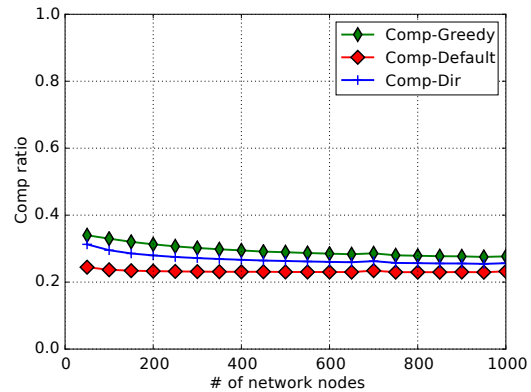
- Integer Linear Programing formulation
  - ⇒ Not scalable
- Greedy algorithm
  - ⇒ Each time, select the source or destination that can be compressed the best
- Just the default port
  - ⇒ The third table of Direction-Based

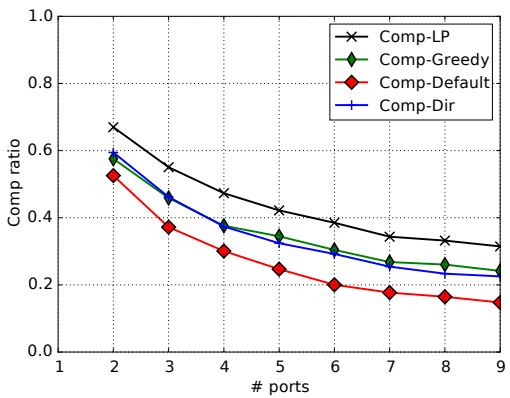
Data sets

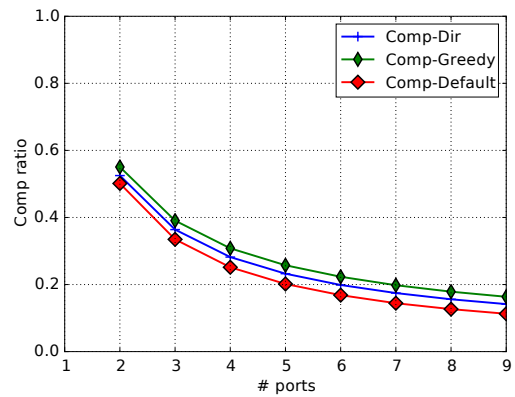
- Random tables
  - Density, number of sources/destinations, number of ports
- Network tables
  - SNDlib instances (atlanta, germany50, zib54, ta2)

Compression Ratio: Random tables



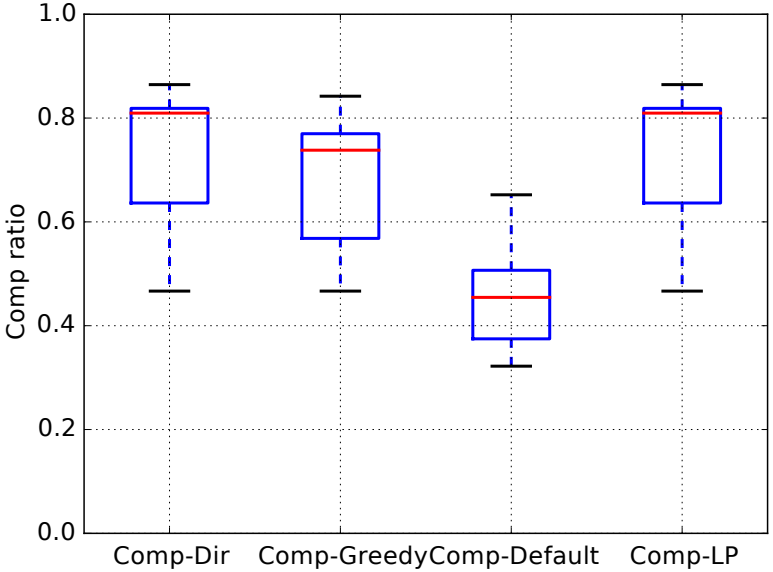




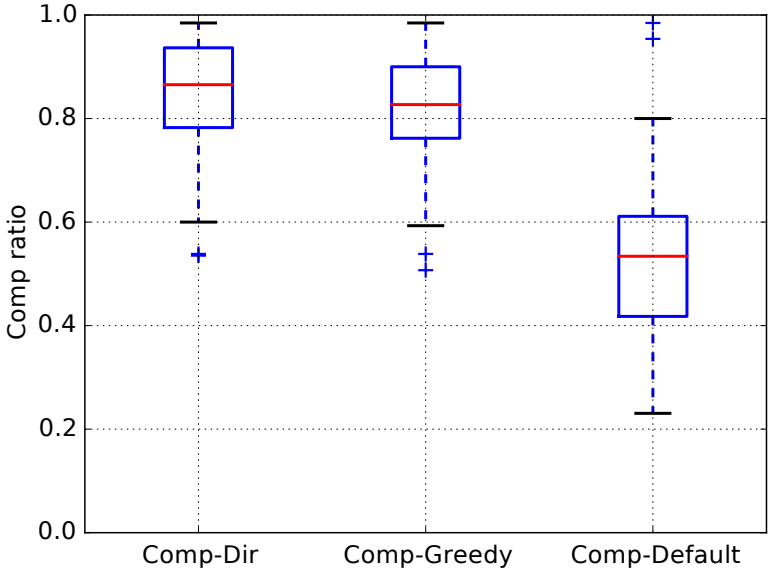


Greedy and Direction-Based have similar results

Compression Ratio: Network tables



atlanta (15 nodes, 44 links)



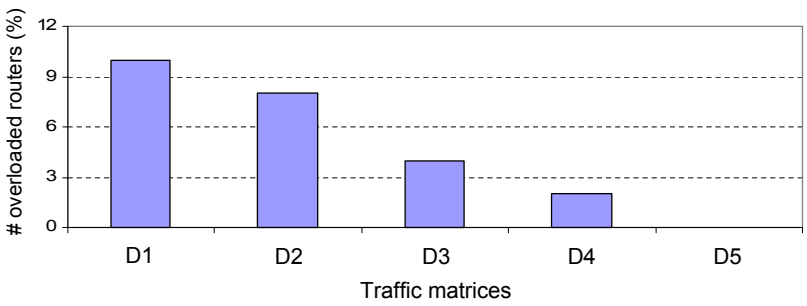
ta2 (81 nodes, 162 links)

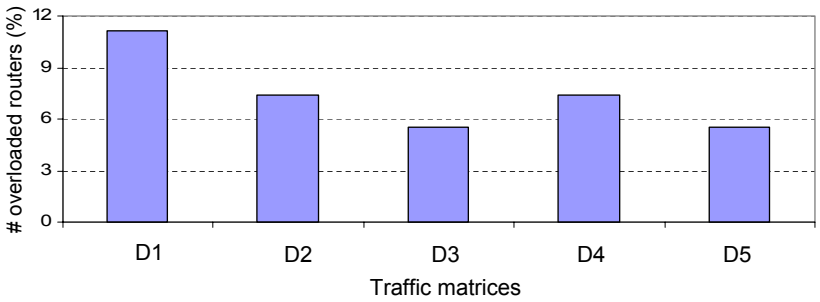
Direction-Based behaves better on network tables

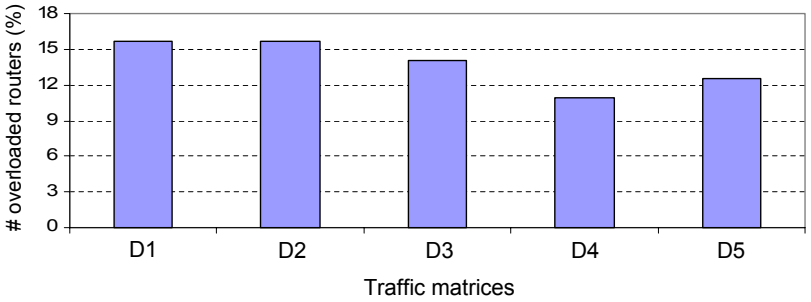
ENERGY AWARE ROUTING WITH COMPRESSION

Energy Aware Routing with Compression Problem (EARC)

- Shutting down links increases shortest paths
  - Increases the number of rules required







Joint problem of routing requests and compressing forwarding tables

Power Model

State of the link

Fraction of bandwidth used

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

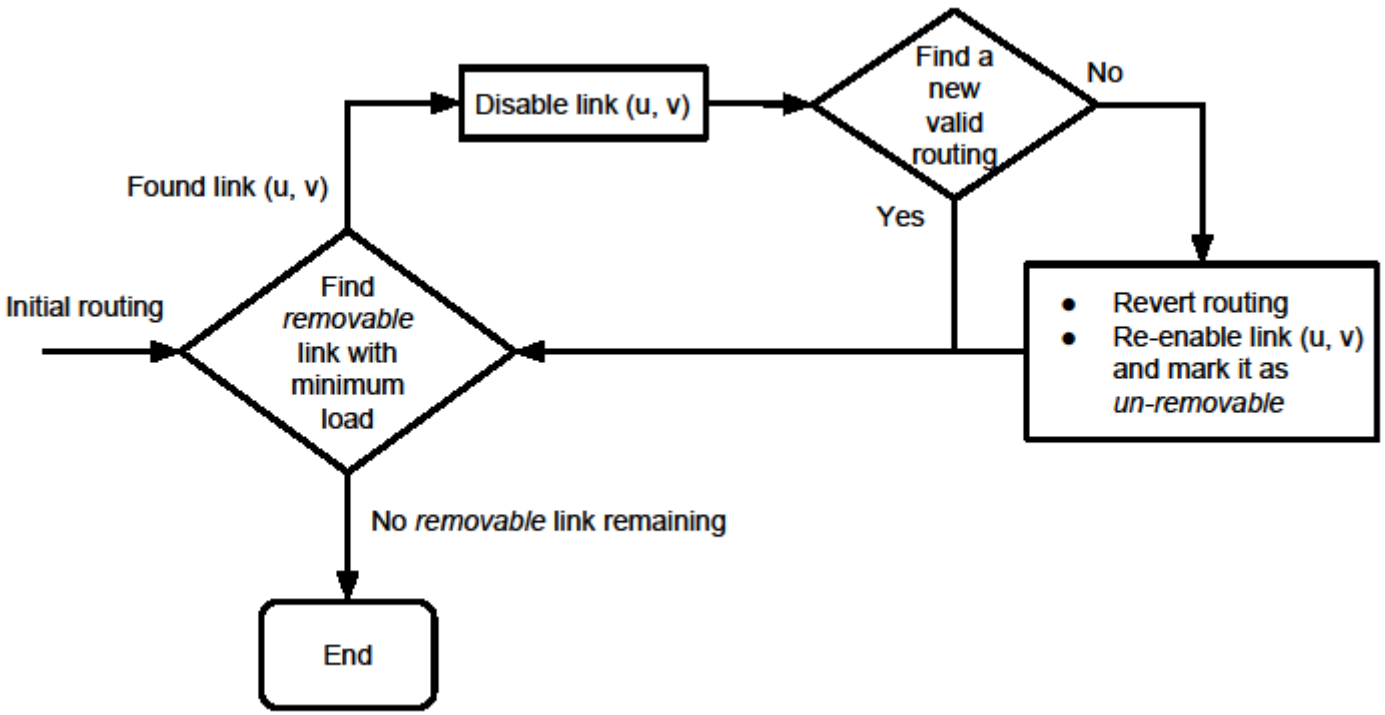
Power used when idle

Additional power

Contributions

- ILP formulation
  - default rule only
  - default rule and wildcard rules
- Heuristic
  - Energy saving module
  - Routing module
  - Compression module (previously mentioned)
- Simulations on SNDlib topologies and traffic traces

Heuristic: Energy saving module



Heuristic: Routing Module

Weighted shortest path on residual graph

Assignment of paths according to table and link usage

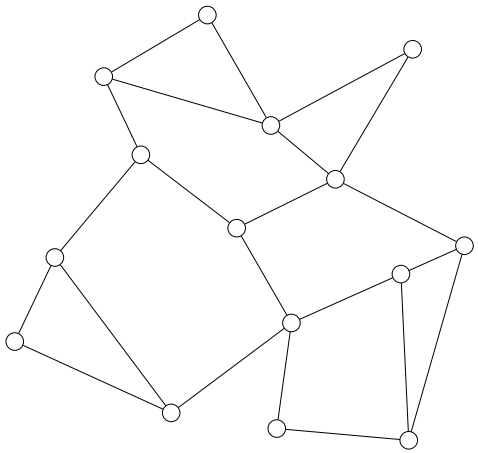
Compress tables when full

$$w_{uv} = \alpha \times w^r_{uv} + \beta \times w^l_{uv}$$

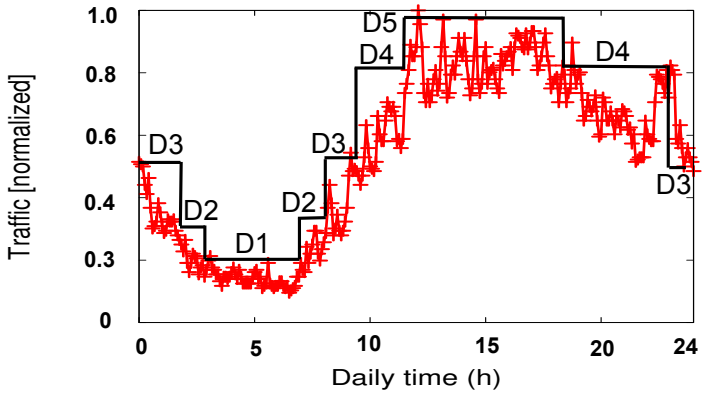
Table usage weight (0 if corresponding wildcard)

Link usage weight

Results: ILP vs Heuristic

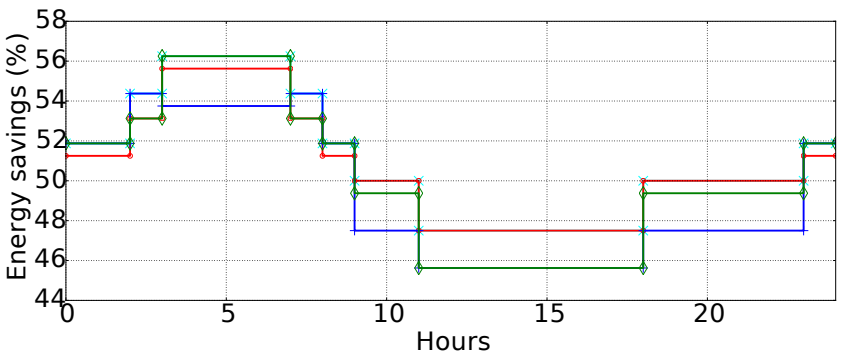


The ILPs run on the atlanta network (15 nodes and 44 links)

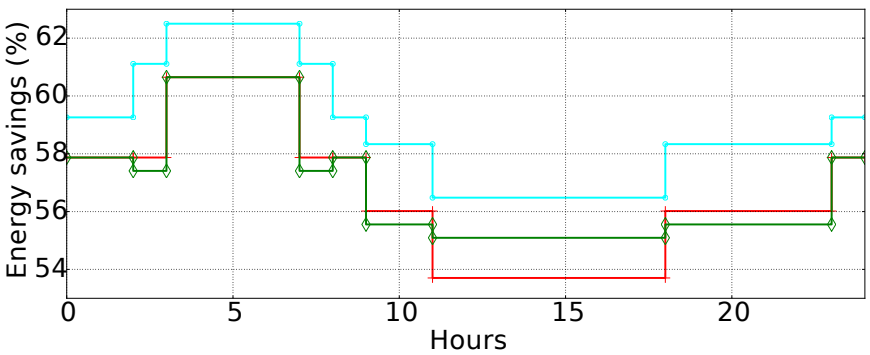


Rule capacity	EARC-LP-Default		EARC-LP-Multi		EARC-H-Direction		EARC-H-LP	
	savings	time	savings	time	savings	time	savings	time
100	52.27	641 940	52.27	694 302	40.91	~ 14	40.91	3381
750	52.27	33 830	52.27	486 759	40.91	~ 14	40.91	3311
2000	52.27	23 640	52.27	487 386	40.91	~ 14	40.91	3300

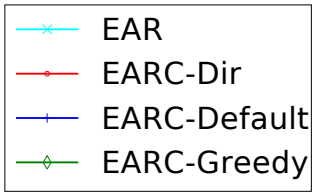
Results: Energy Savings



zib54 (54 nodes, 216 links)



ta2 (81 nodes, 162 links)



- No feasible solutions without compression for some networks
- Energy close to classic EAR using compression

Conclusion & Future Works

- Study the constraints for the deployment of green policies in SDN networks
- In this talk, TCAM constraints on forwarding tables
- Using wildcard rules, we can provide almost the same savings as without limits
- Shown on an SDN testbed, that we can use compression without noticeable impact on the network performances
- QoS/QoE & protection ?