CSE 550: Systems for all

Au 2021

Ratul Mahajan

What is SDN?

Not quite “software-defined”

- Network control planes were always software defined

Not quite “centralized control”

Separation of control and data planes

- Enables centralization but not centralization is not pre-requisite

Why might you want to separate control and data plane?

Arbitrary control over how packets are forwarded

Complex requirements can be hard to specify in terms of distributed, local rules

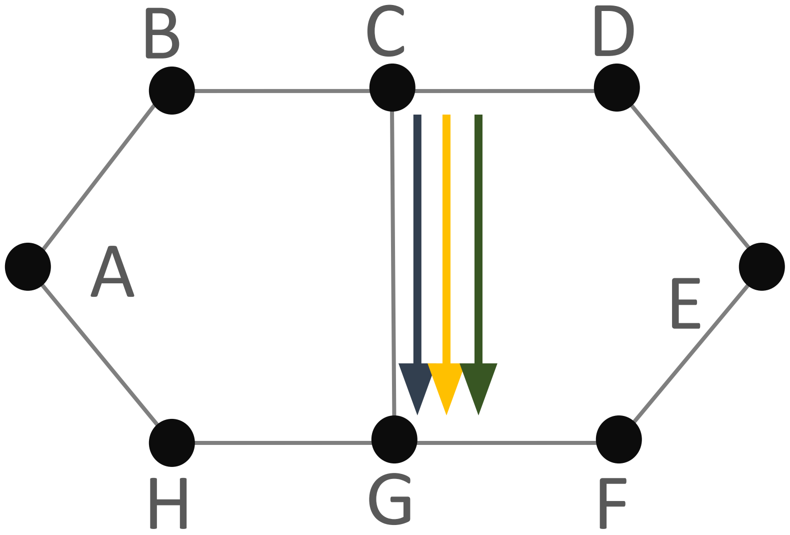
- Suppose you wanted all paths in the network to be of length 10

Traffic engineering case study

Traffic engineering journey

- SPF with load-based cost
- SPF with static cost
- CSPF (used in MPLS)
- SDN

Limitations of static-cost SPF



CSPF

Each ingress router measure traffic that it is sending to other routers

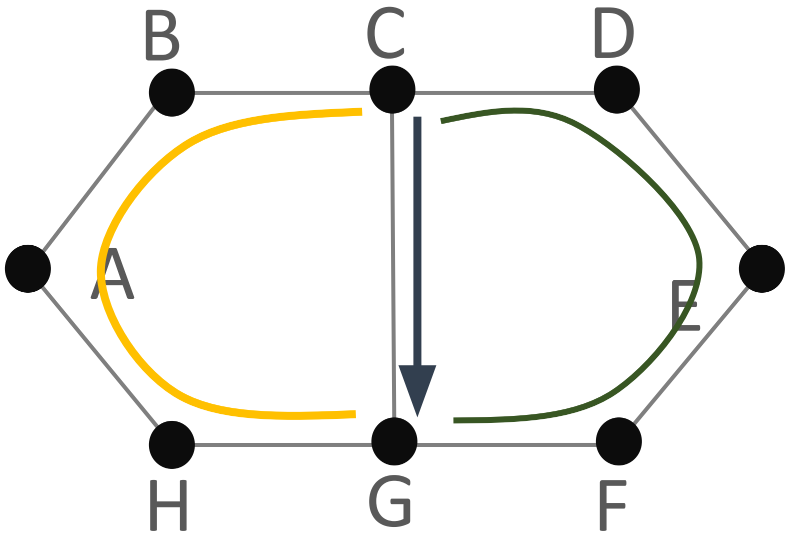
Ingress router finds paths that can accommodate its traffic

- Shortest path that meets the capacity constraint (CSPF)

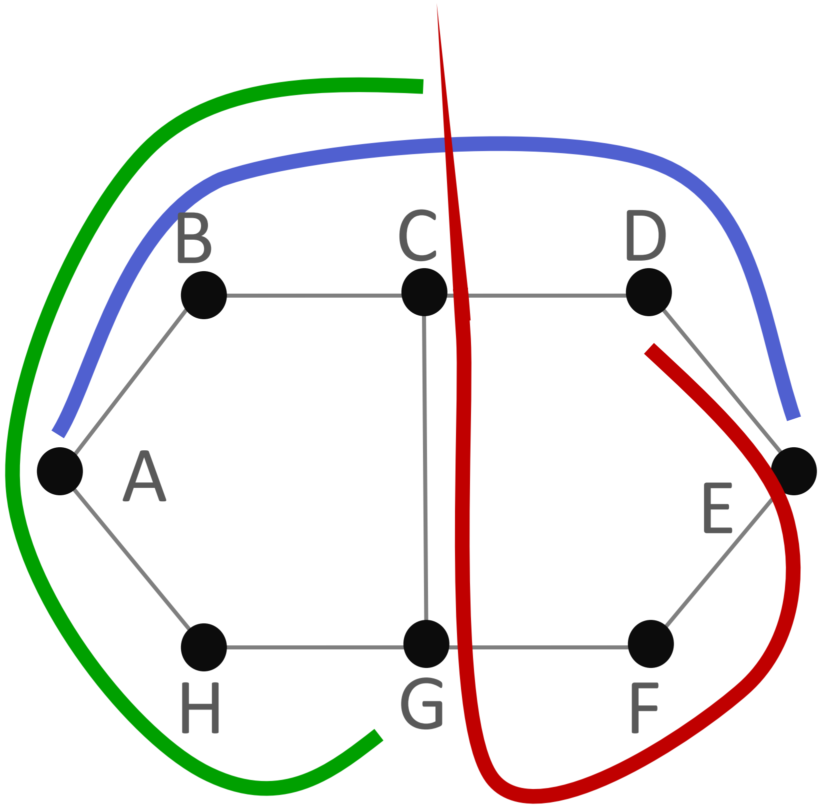
Ingress router asks other routers if they can use the path

- Necessary because all ingress routers are operating independently

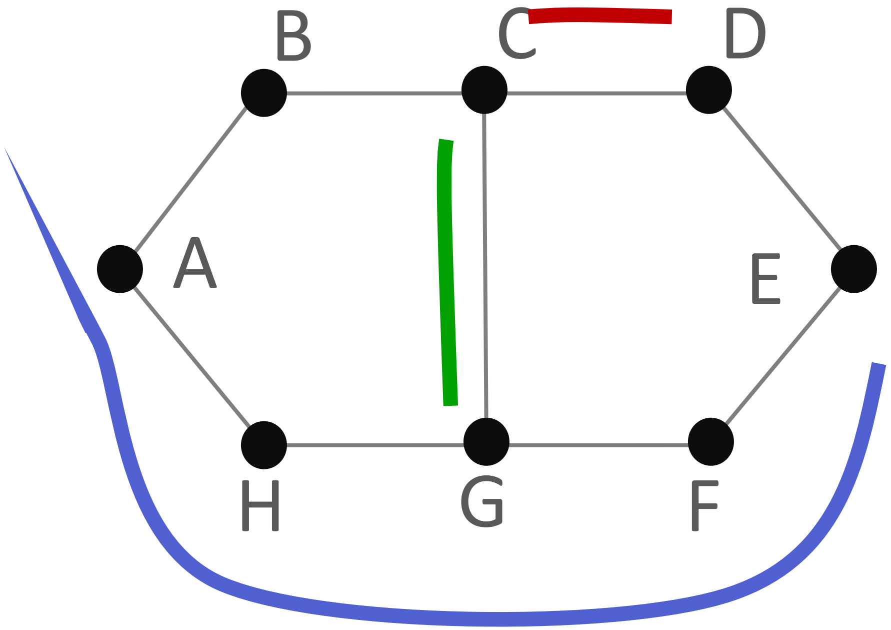
Same example with CSPF



But CSPF has issues too



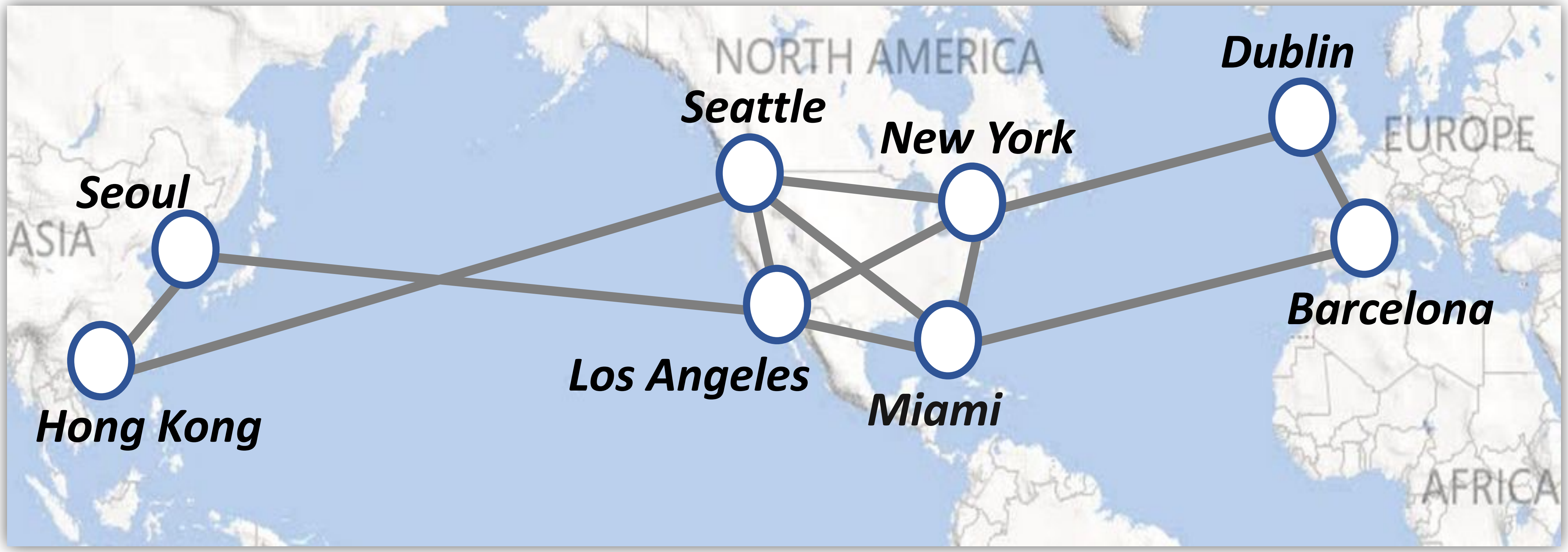
Local, greedy allocation
(Distributed CSPF)



Globally optimal allocation
(Centralized)

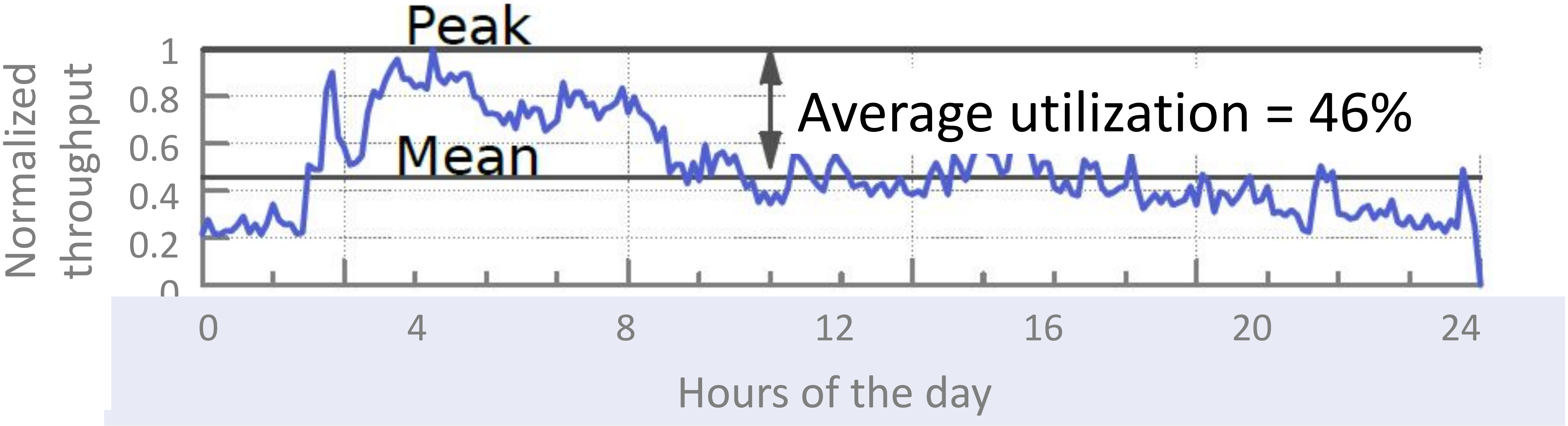
SWAN: SDN based TE

Inter-DC WAN: A critical, expensive resource



But it was being used highly inefficiently

Inefficiency of the inter-DC WAN



Normalized traffic on a busy link between data centers

Root cause: Service-level allocations

Operators configure individual services with maximum sending rate

	S1	S2	S3	…..
SEA □ NYC (80)	10	15	5	….
SEA □ CHI (100)	20	20	10	….
……				

Inefficient: The combined maximum is uncommon

Unreliable: Load can exceed capacity when failures occur

Slow to change: Must change all allocations to add services or network links

Centralized control can increase efficiency

Service 1

- Priority: Bg
- Weight: 1

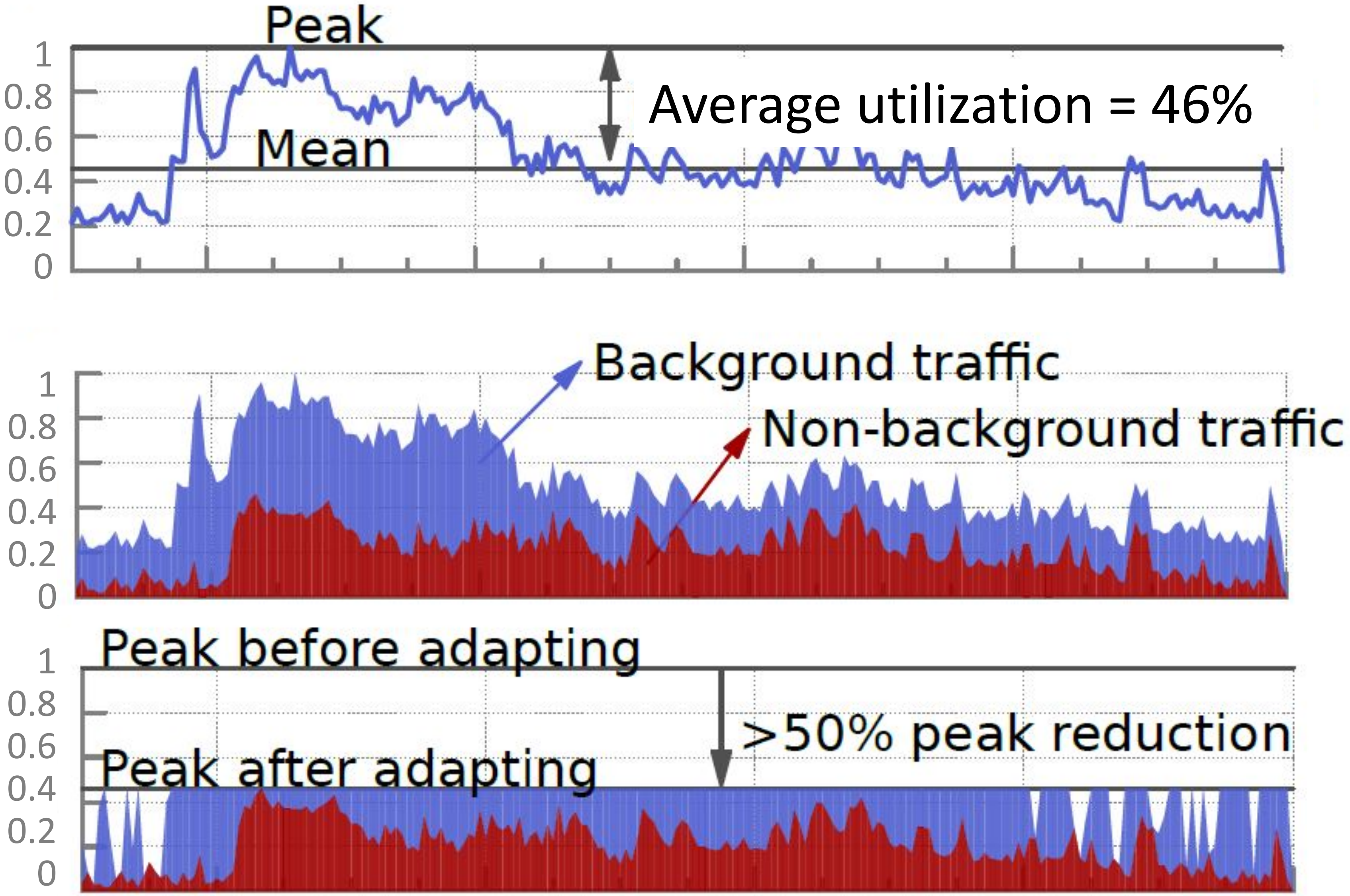
Service 2

- Priority: Bg
- Weight: 2

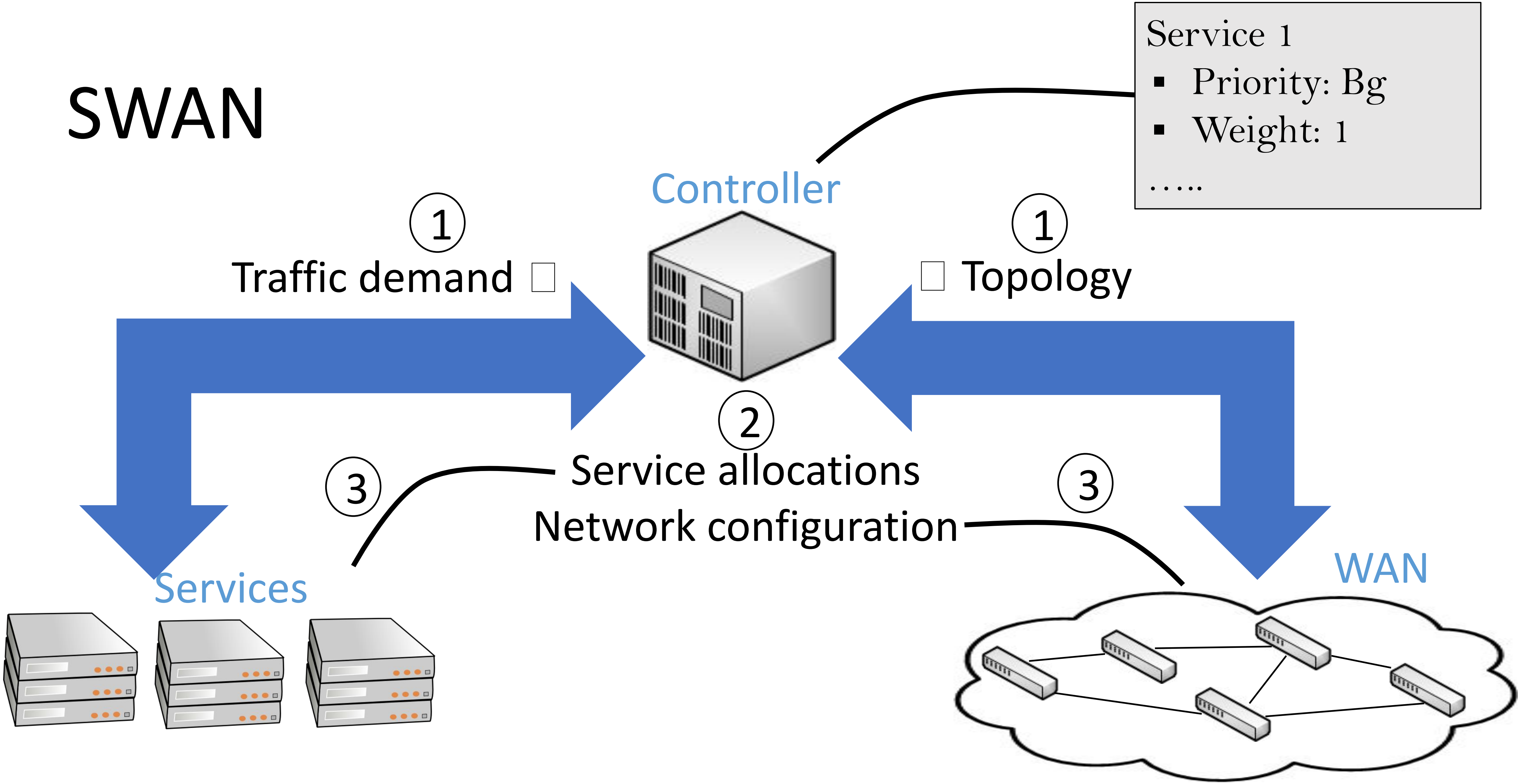
Service 3

- Priority: Non-bg
- Weight: 1

……



SWAN



Service 1
- Priority: Bg
- Weight: 1

…..

Controller

1 Traffic demand □

1 □ Topology

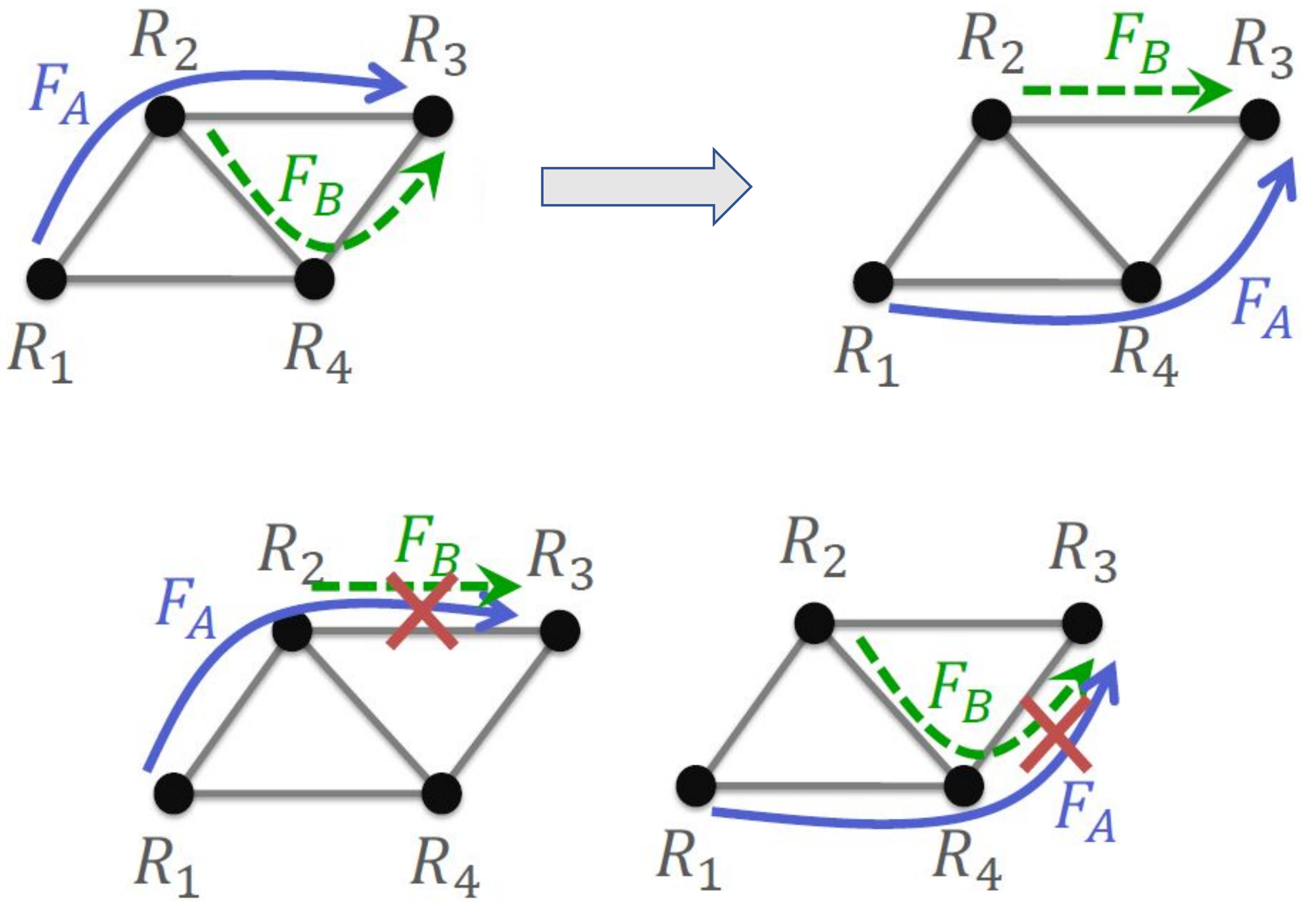
2 Service allocations

3 Network configuration 3

Challenge: Congestion during network updates

Link capacity: 10

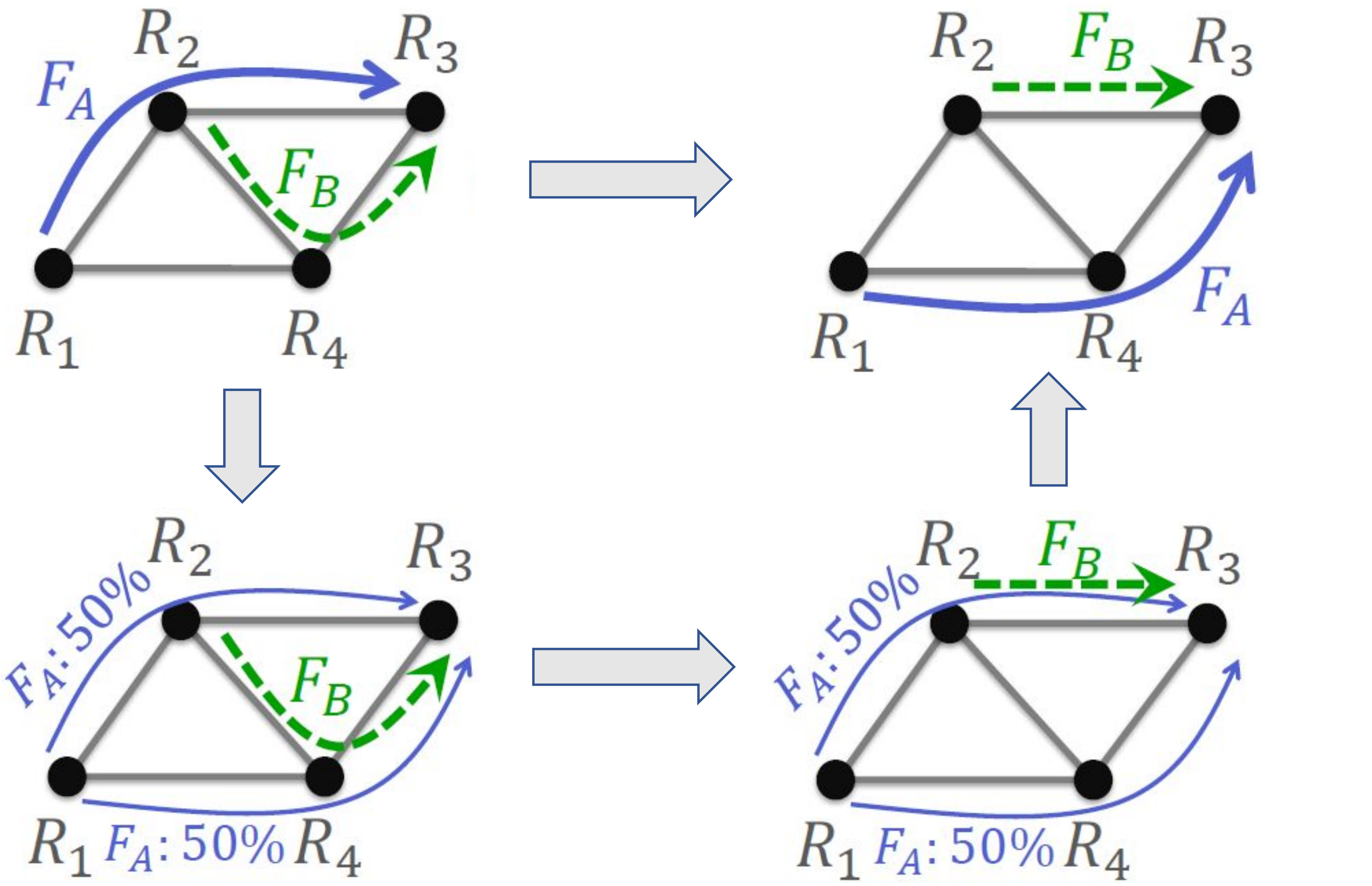
Flow size: 6.6



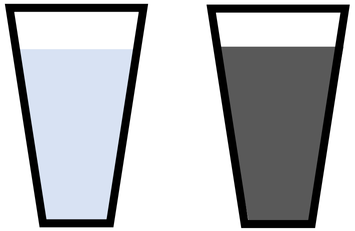
Solution: Congestion-free update plans

Link capacity: 10

Flow size: 6.6



Computing congestion-free update plans



- Leave scratch capacity s on each link
  - Guarantees a plan with at most $\left\lceil \frac{1}{s} \right\rceil - 1$ steps

Find a plan with minimum number of steps using an LP

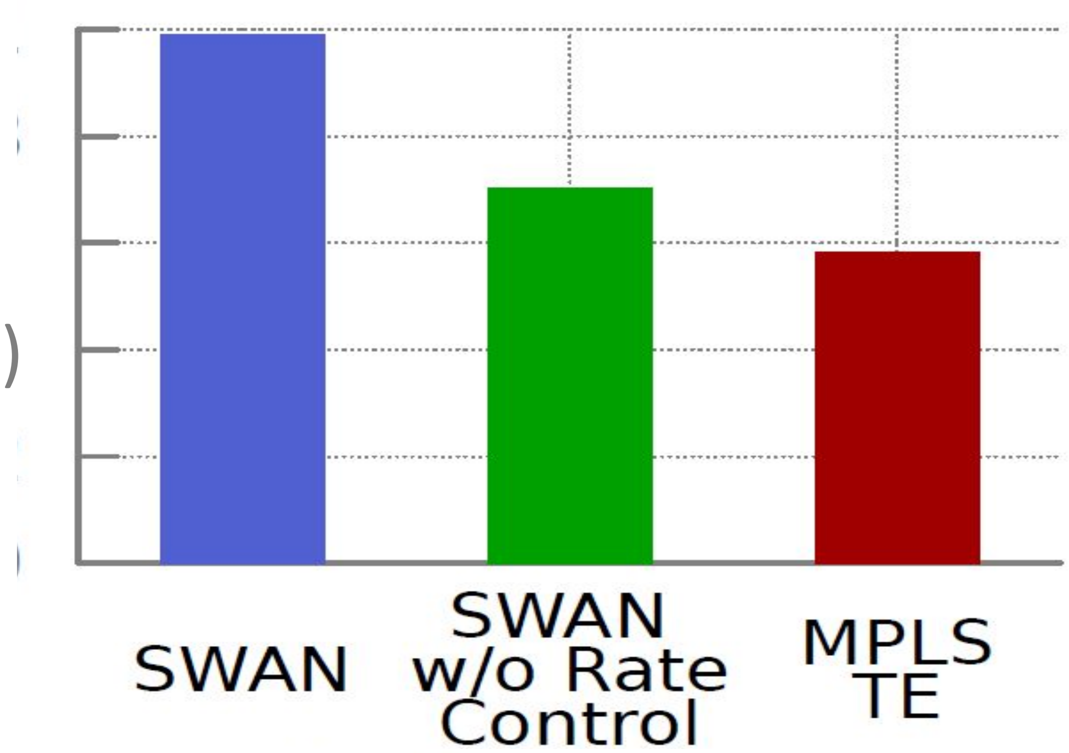
- Search for a feasible plan with 1, 2, …. max steps

Use scratch capacity for background traffic

- Bound its experienced congestion

Efficiency improvement with SWAN

Throughput
(relative to optimal)



Over to Innocent and Sirui