

Minimum Congestion Routing of Unsplittable Flows in Data-Center Networks

Miguel Ferreira

CMU, IST

Nirav Atre, Justine Sherry

CMU

Michael Dinitz

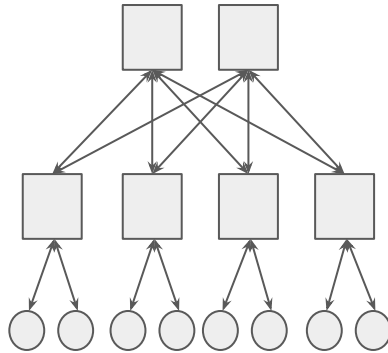
JHU

João Luís Sobrinho

IST

Most data-centers are modeled after Clos networks

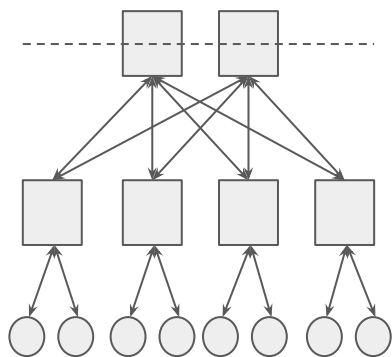
[Singh *et al.* 15, Greenberg *et al.* 15, Gangidi *et al.* 24, Qian *et al.* 24]



Folded Clos network

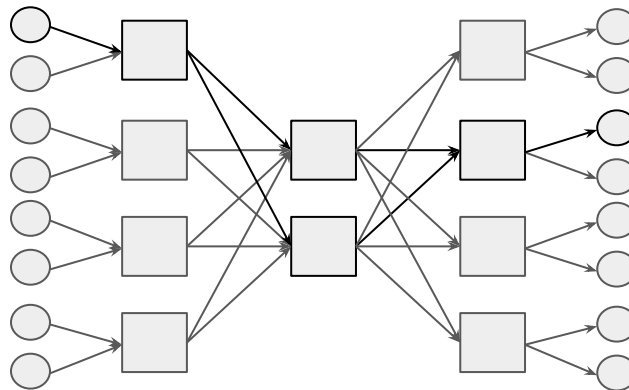
Most data-centers are modeled after Clos networks

[Singh *et al.* 15, Greenberg *et al.* 15, Gangidi *et al.* 24, Qian *et al.* 24]



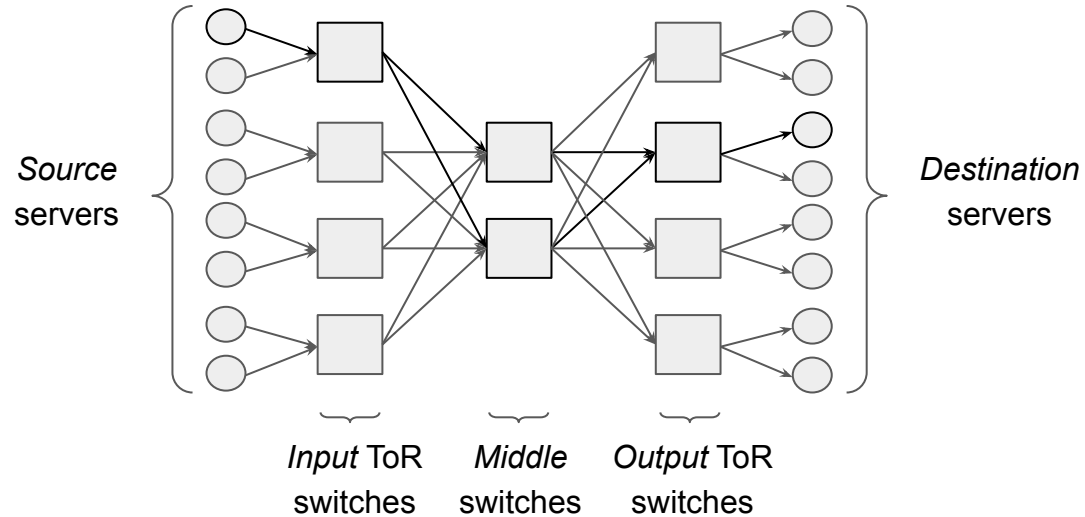
Folded Clos network

→
*Unfolded around
symmetry axis*

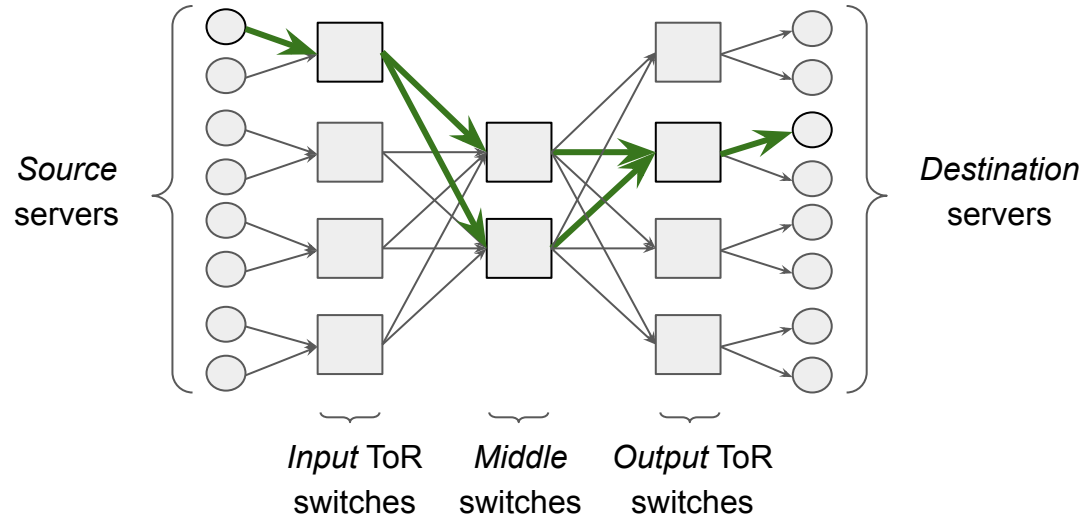


Unfolded Clos network

Characteristics of Clos networks [Clos 53]

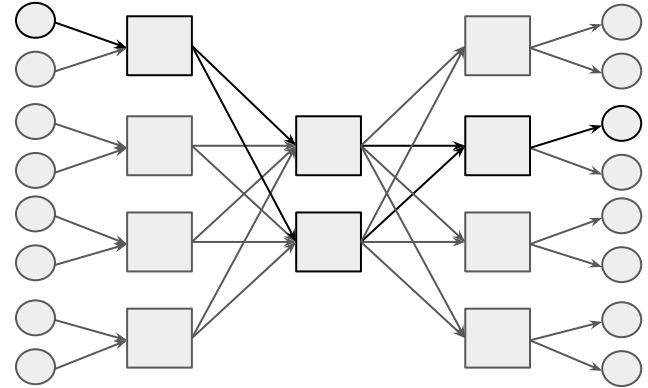


Characteristics of Clos networks [Clos 53]



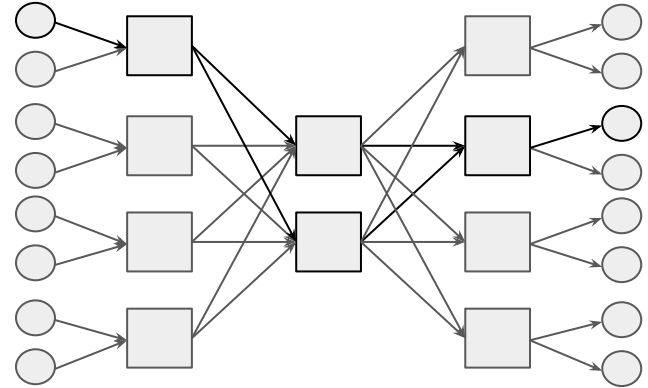
Characteristics of Clos networks [Clos 53]

- N = number of servers per ToR switch
= number of middle switches



Characteristics of Clos networks [Clos 53]

- N = number of servers per ToR switch
= number of middle switches
- Uniform link capacities



Data-center wish to closely satisfy flow demands ...

[Ballani *et al.* 11, Lee *et al.* 14]

- Guaranteeing predictable bandwidth is desirable for most applications

Data-center wish to closely satisfy flow demands ...

[Ballani *et al.* 11, Lee *et al.* 14]

- Guaranteeing predictable bandwidth is desirable for most applications

... therefore, flow routing in data-center seeks to minimize congestion [Alizadeh *et al.* 14, Singla *et al.* 14, Namyar *et al.* 21]

Characteristics of the minimum congestion routing problem

- **Input:** Set of flows; flow maps to source-destination pair and demand
 - Aggregate demand on any external link (between server and ToR) ≤ 1

Characteristics of the minimum congestion routing problem

- **Input:** Set of flows; flow maps to source-destination pair and demand
 - Aggregate demand on any external link (between server and ToR) ≤ 1

- **Solution:** Routing for the flows, that is, assignment from flows to middle switches

Characteristics of the minimum congestion routing problem

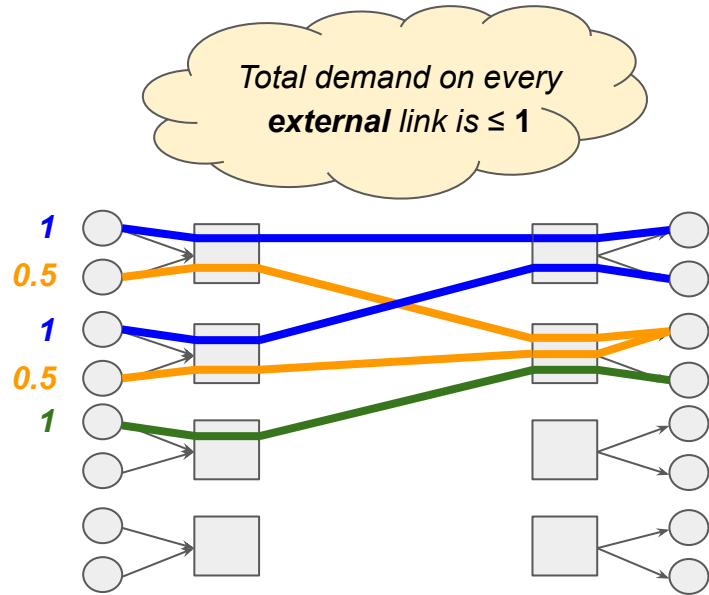
- **Input:** Set of flows; flow maps to source-destination pair and demand
 - Aggregate demand on any external link (between server and ToR) ≤ 1
- **Solution:** Routing for the flows, that is, assignment from flows to middle switches
- **Objective:** Minimize *congestion*, that is, maximum aggregate demand routed on any internal link (between ToR and middle switch)

Characteristics of the minimum congestion routing problem

- **Input:** Set of flows; flow maps to source-destination pair and demand
 - Aggregate demand on any external link (between server and ToR) ≤ 1
- **Solution:** Routing for the flows, that is, assignment from flows to middle switches
- **Objective:** Minimize *congestion*, that is, maximum aggregate demand routed on any internal link (between ToR and middle switch)

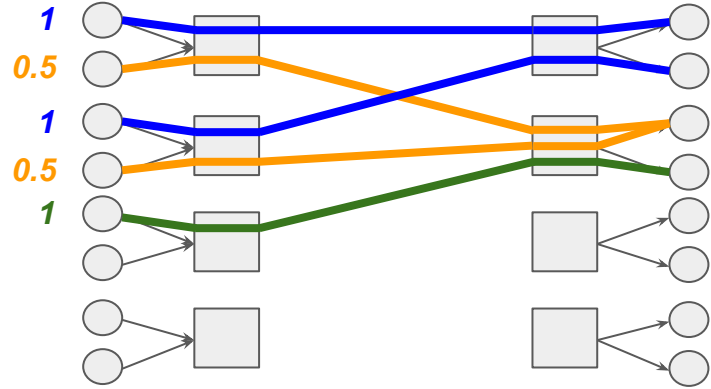
Every flow satisfies its demand if and only if routing has congestion ≤ 1

Characteristics of the minimum congestion routing problem

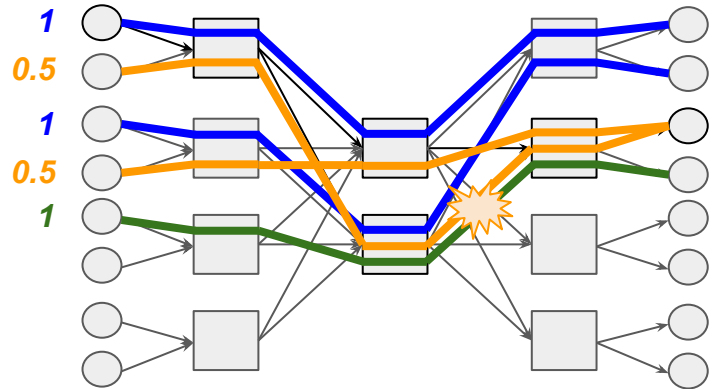


Characteristics of the minimum congestion routing problem

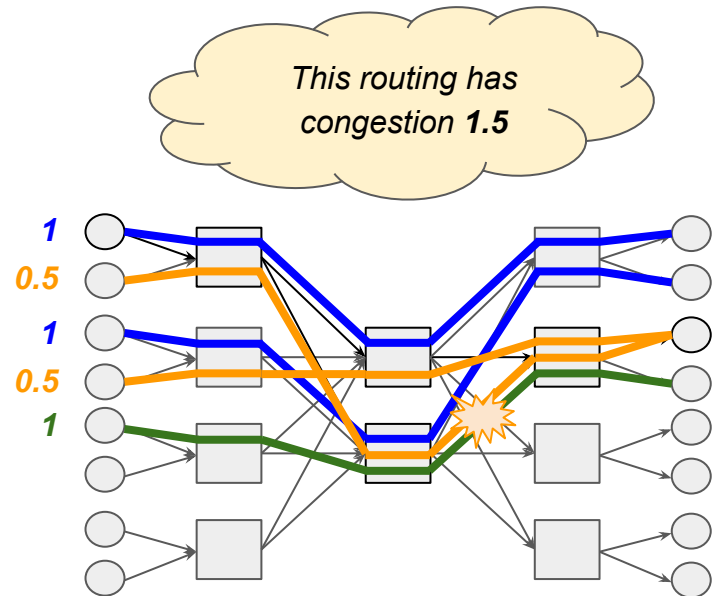
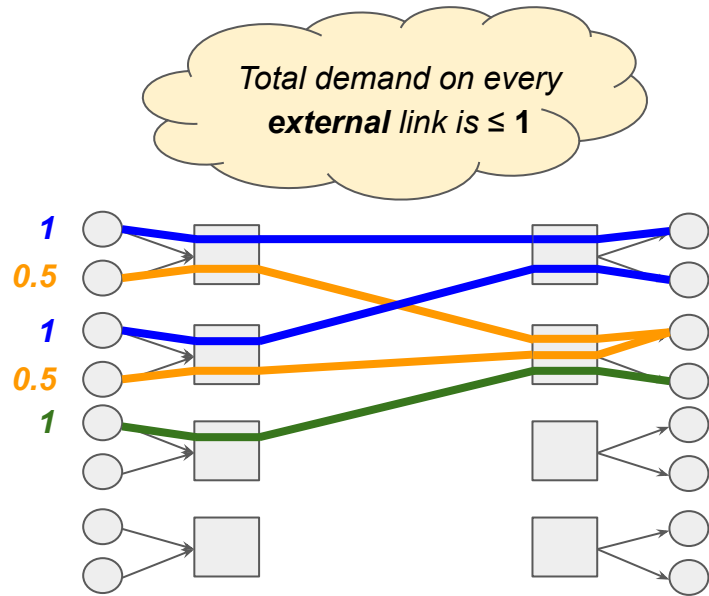
Total demand on every external link is ≤ 1



There is *internal* link with total demand 1.5



Characteristics of the minimum congestion routing problem



With arbitrary flow splitting, minimizing congestion is easy

[Chiesa *et al.* 17]

- **Theorem:** For all sets of flows, uniformly splitting each flow demand over all source-destination paths yields a minimum congestion routing with congestion ≤ 1

With arbitrary flow splitting, minimizing congestion is easy

[Chiesa *et al.* 17]

- **Theorem:** For all sets of flows, uniformly splitting each flow demand over all source-destination paths yields a minimum congestion routing with congestion ≤ 1

With splittable flows, every flow satisfies its demand

However, flows splitting has significantly deployability challenges

[Qureshi *et al.* 22, Gangidi *et al.* 24, Qian *et al.* 24]

- While there have been multiple proposals for implement splittable flows...
 - MPTCP [Raiciu *et al.* 2011], packet-spraying [Dixit *et al.* 2013]

However, flows splitting has significantly deployability challenges

[Qureshi *et al.* 22, Gangidi *et al.* 24, Qian *et al.* 24]

- While there have been multiple proposals for implement splittable flows...
 - MPTCP [Raiciu *et al.* 2011], packet-spraying [Dixit *et al.* 2013]

- ... none of them has been adopted in practice
 - Require deep changes to current protocols, and are unverified in large scale

This talk

This talk

- **Question #1:** Is the congestion of a minimum congestion routing of unsplittable flows ≤ 1 for all sets of flows? If not, how close to 1 is it?

This talk

- **Question #1:** Is the congestion of a minimum congestion routing of unsplittable flows ≤ 1 for all sets of flows? If not, how close to 1 is it?

- **Question #2:** Is a minimum congestion routing of unsplittable flows computable in polynomial-time? If not, how well can it be approximated?

Prior results on minimum congestion routing

Prior results on minimum congestion routing

[Raghavan and Tompson 87, Chakrabarti *et al.* 07]

- In general networks, there are approximation algorithms that ensure worst-case congestion and approximation factor **poly-logarithmic** in size N of the network



Prior results on minimum congestion routing

[Al-Fares et al. 08]

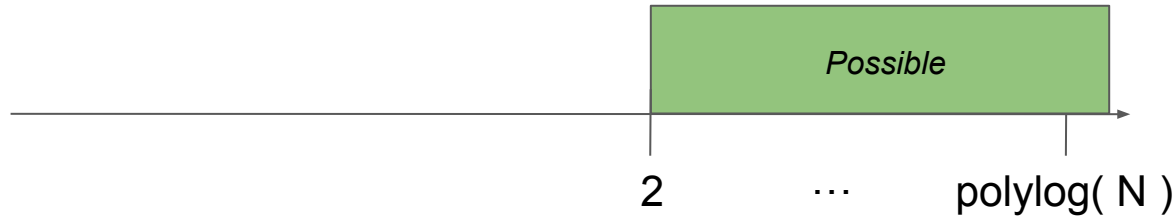
- In Clos networks, random ECMP ensures worst-case congestion and approximation factor **poly-logarithmic** in number N of the middle switches



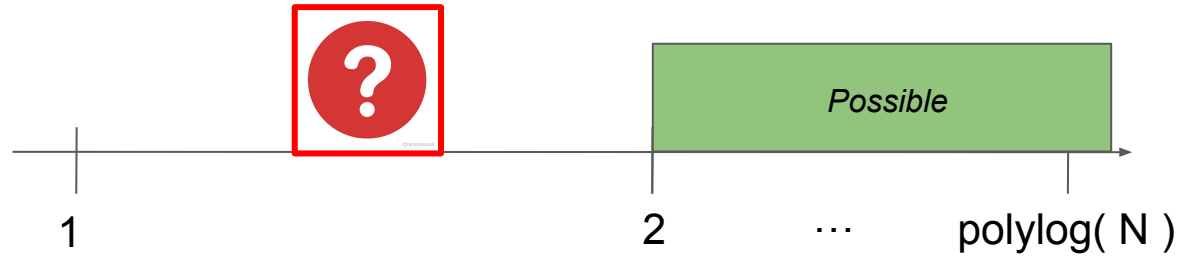
Prior results on minimum congestion routing

[Melen and Turner 89, Al-Fares et al. 10]

- In Clos networks, state-of-the-art heuristics ensure worst-case congestion and approximation factor of **2**

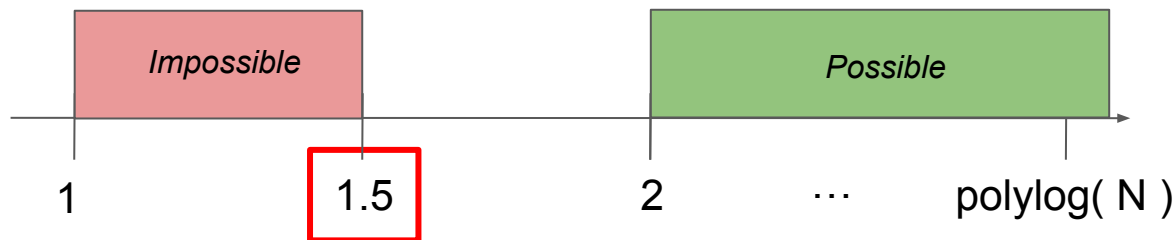


What we know: worst-case congestion and approximation factor is between 1 and 2



What we *show*: worst-case congestion and approximation factor is between 1.5 and ...

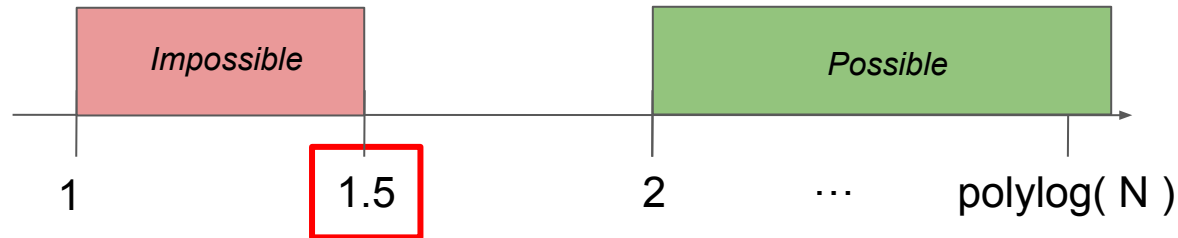
- **Result #1:** (#1.1) Minimum congestion is ≥ 1.5 . (#1.2) Furthermore, it is NP-hard to approximate minimum by factor < 1.5 .



What we *show*: worst-case congestion and approximation factor is between 1.5 and ...

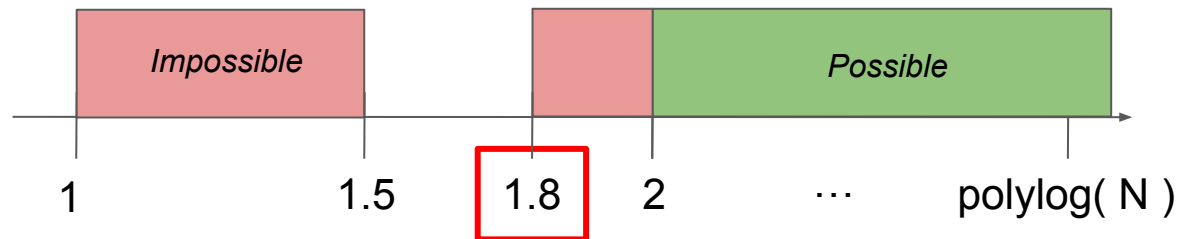
- **Result #1:** (#1.1) Minimum congestion is ≥ 1.5 . (#1.2) Furthermore, it is NP-hard to approximate minimum by factor < 1.5 .

- **Implication:** Special structure of Clos networks cannot avoid some flows obtaining $\leq 2/3$ of their demands



... 1.8

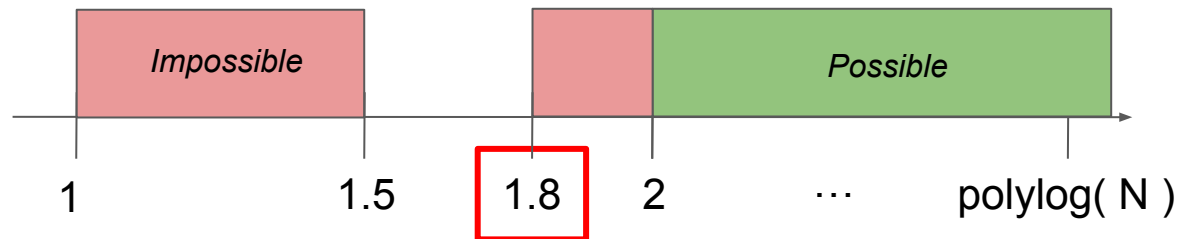
- **Result #2:** There is a polynomial-time algorithm that ensures congestion ≤ 1.8 for all sets of flows, and approximates minimum congestion by a factor ≤ 1.8



... 1.8

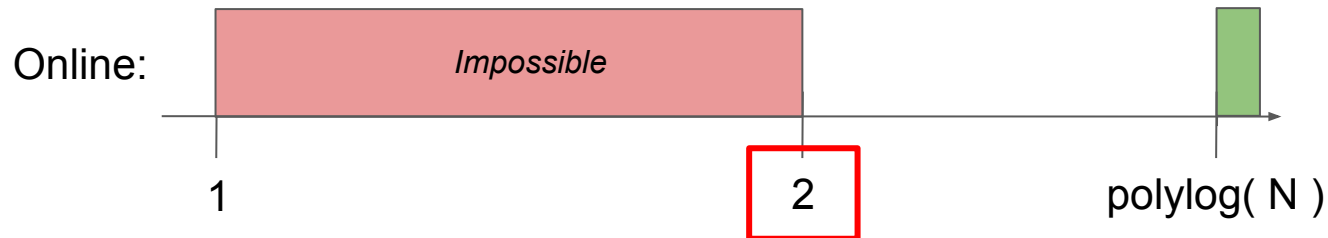
- **Result #2:** There is a polynomial-time algorithm that ensures congestion ≤ 1.8 for all sets of flows, and approximates minimum congestion by a factor ≤ 1.8

- **Implication:** Known heuristics are not optimal



What we show: in the online setting, worst-case congestion and approximation factor is at least 2

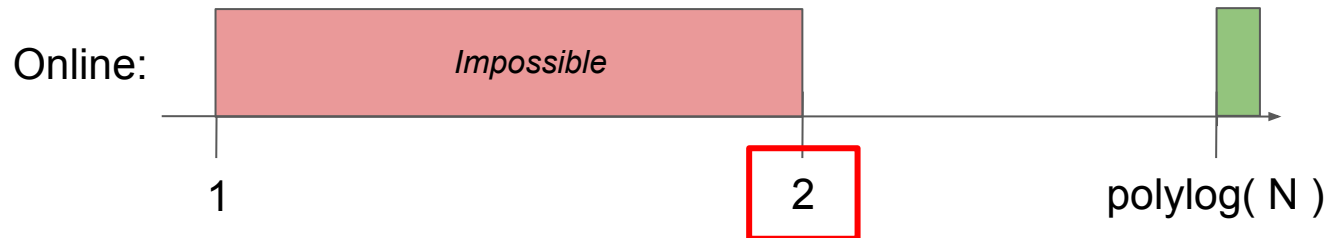
- **Result #3:** No online algorithm (even randomized) approximates minimum congestion by a factor < 2



What we show: in the online setting, worst-case congestion and approximation factor is at least 2

- **Result #3:** No online algorithm (even randomized) approximates minimum congestion by a factor < 2

- **Implication:** There is a strict separation between online and offline settings



Formal statement for limits to congestion and approximation

Formal statement for limits to congestion and approximation

- **Lemma [Hwang 83]:** For all sets of flows with demand 1, there is a routing with congestion 1, and such a routing can be found in polynomial-time

Formal statement for limits to congestion and approximation

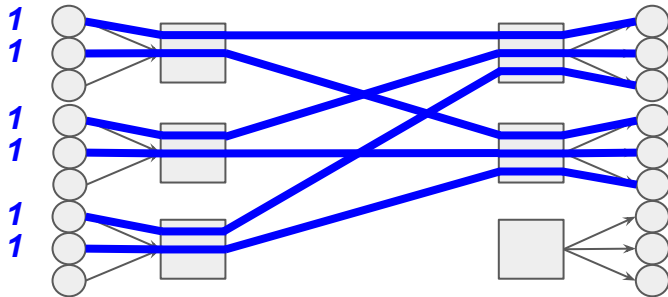
- **Lemma [Hwang 83]:** For all sets of flows with demand 1, there is a routing with congestion 1, and such a routing can be found in polynomial-time
- **Theorem #1.1:** There is a set of flows with demands 1 or 0.5 such that the minimum congestion is 1.5

Formal statement for limits to congestion and approximation

- **Lemma [Hwang 83]:** For all sets of flows with demand 1, there is a routing with congestion 1, and such a routing can be found in polynomial-time
- **Theorem #1.1:** There is a set of flows with demands 1 or 0.5 such that the minimum congestion is 1.5
- **Theorem #1.2:** For a set of flows with demands 1 or 0.5, deciding if minimum congestion is ≤ 1 or 1.5 is NP-complete

Key idea for limits to congestion and approximation

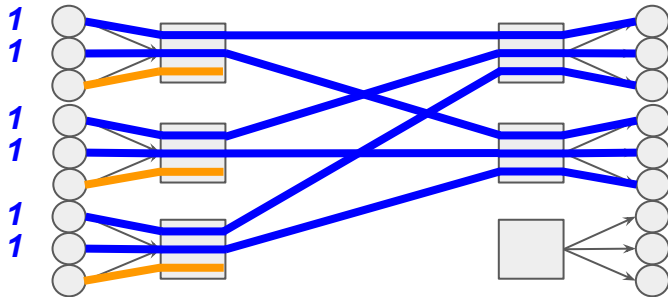
- **Lemma:** For every routing of the tunnel gadget with congestion 1, there is a different middle switch at each input switch to which no flow is assigned



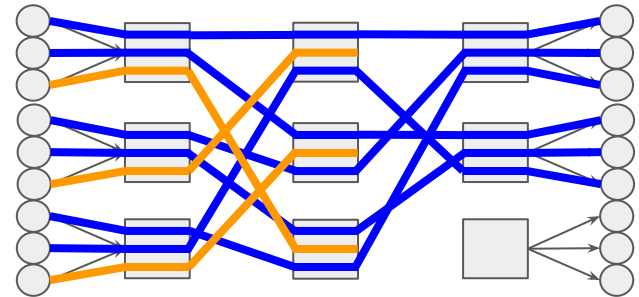
Funnel gadget

Key idea for limits to congestion and approximation

- **Lemma:** For every routing of the tunnel gadget with congestion 1, there is a different middle switch at each input switch to which no flow is assigned



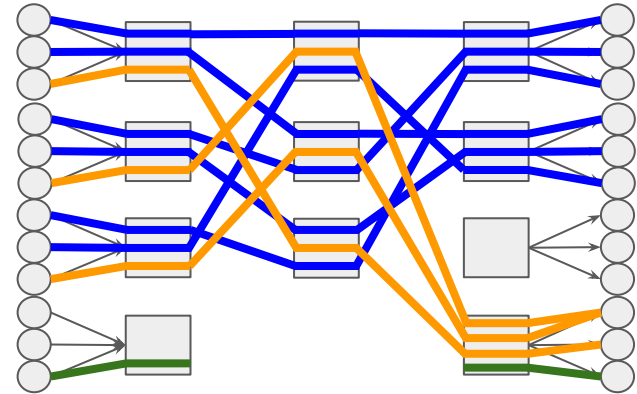
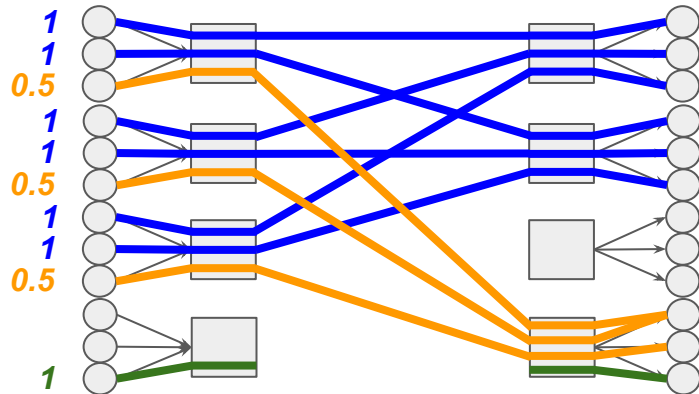
Funnel gadget



Routing with congestion 1

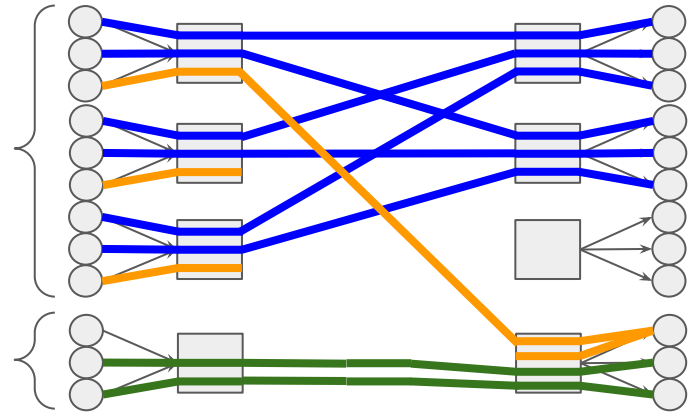
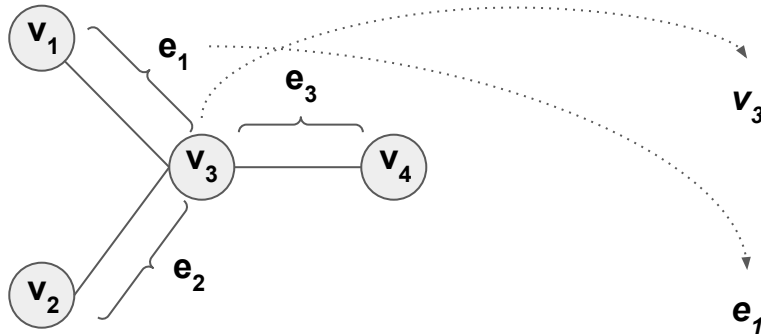
Proof hint for Theorem #1.1: Add demand 0.5 flows to funnel gadget such that all middle switches blocked

- **Theorem #1.1:** There is a set of flows with demands 1 or 0.5 such that the minimum congestion is ≥ 1.5



Proof hint for Theorem #1.2: Establish a reduction from the 3-edge coloring problem

- **Theorem #1.2:** For a set of flows with demands 1 or 0.5, deciding if minimum congestion is ≤ 1 or **1.5** is NP-complete



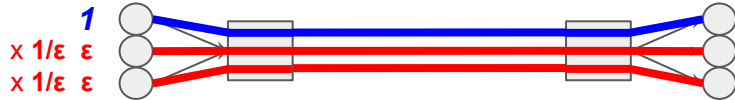
Towards a 1.8-approximation algorithm that ensures congestion 1.8

- Linear program corresponding to multi-commodity flow relaxation is not helpful

Towards a 1.8-approximation algorithm that ensures congestion 1.8

- Linear program corresponding to multi-commodity flow relaxation is not helpful
- Prior work suggests two combinatorial algorithms to build upon

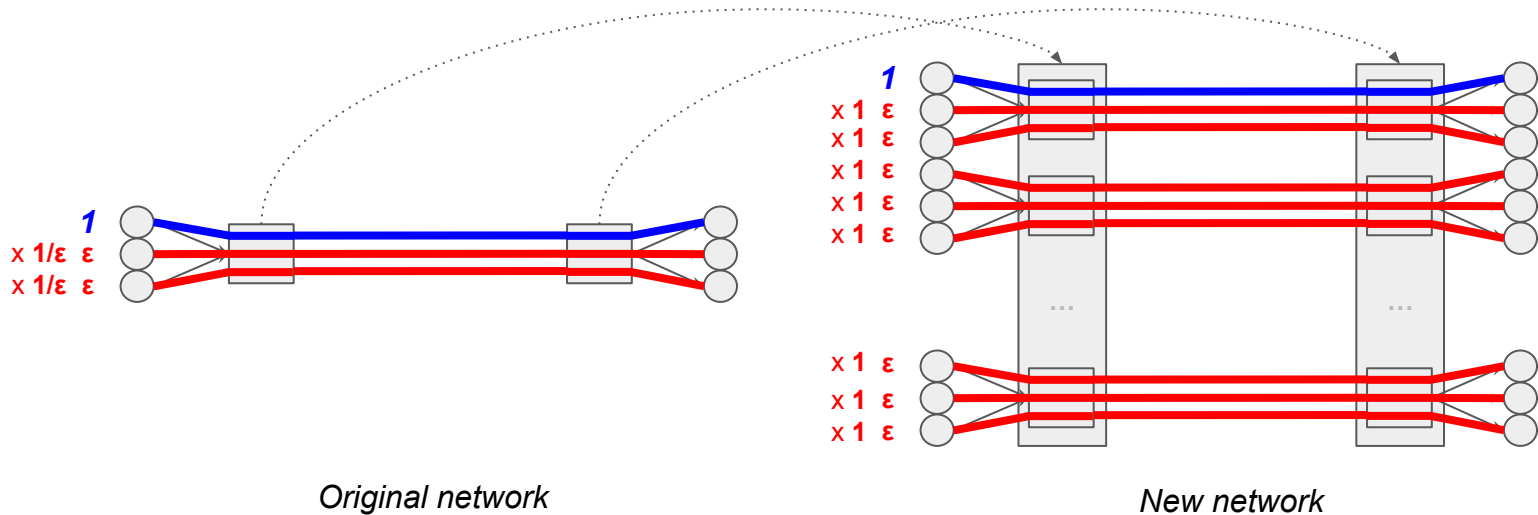
Description of the Melen-Turner algorithm [Melen and Turner 89]



Original network

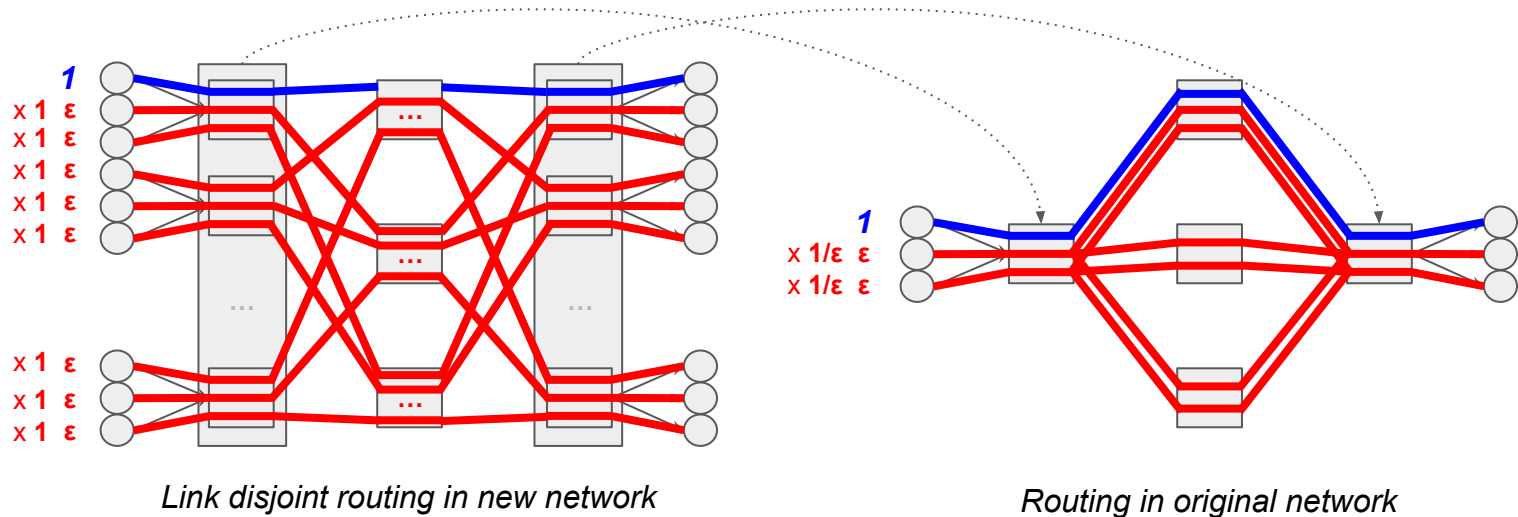
Description of the Melen-Turner algorithm [Melen and Turner 89]

1. Build new network from original one; in new network, there are multiple copies of each ToR, with $\leq N$ flows per copy in decreasing of demands



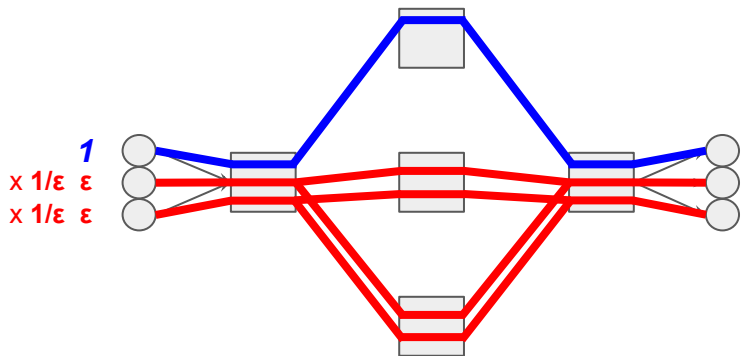
Description of the Melen-Turner algorithm [Melen and Turner 89]

2. Find link-disjoint routing in new network

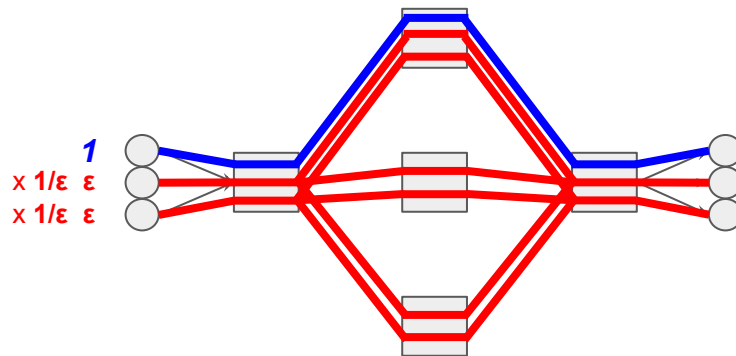


Melen-Turner algorithm has worst-case congestion and approximation factor no better than 2

- **Lemma:** Melen-Turner algorithm is a **2**-approximation algorithm, and returns a routing with congestion ≤ 2 for all sets of flows; **these bounds are tight**
- **Tight example:**



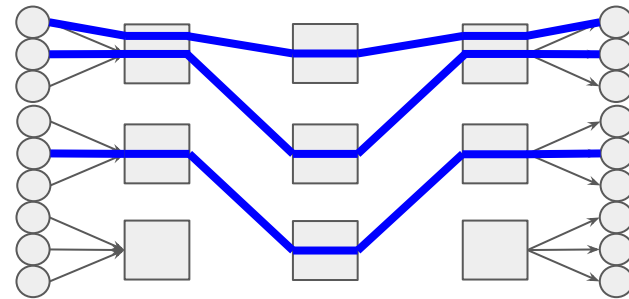
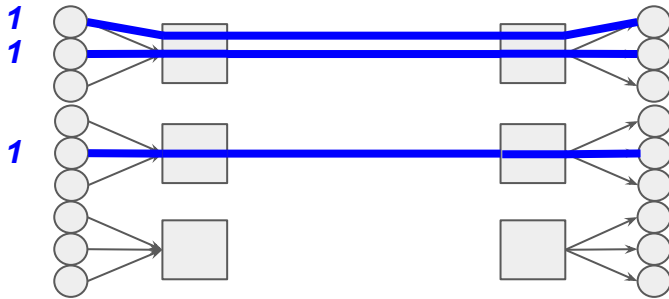
Congestion 1 routing



Congestion 2 routing returned by algorithm

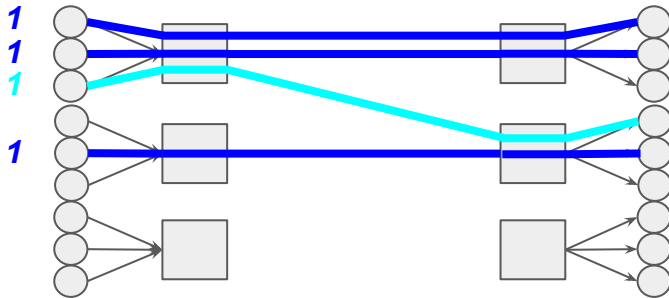
Description of the Sorted-Greedy algorithm [Al-Fares *et al.* 08]

1. Assign flows in decreasing order of demands to minimum congestion paths

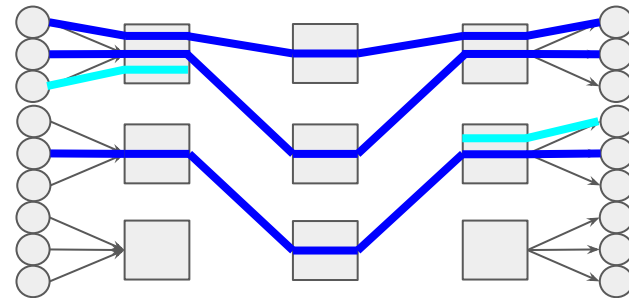


Melen-Turner algorithm has worst-case congestion and approximation factor no better than 2

- **Lemma:** Sorted-greedy algorithm is a 2-approximation algorithm, and returns a routing with congestion ≤ 2 for all sets of flows; **these bounds are tight**
- **Tight example:**



Congestion 1 routing



Congestion 2 routing returned by algorithm

Key idea for the routing algorithm

- Interpolate between the Melen-Turner and Sorted-Greedy algorithm,

Key idea for the routing algorithm

- Interpolate between the Melen-Turner and Sorted-Greedy algorithm,
- Two-phase algorithm bridged by threshold C :

Key idea for the routing algorithm

- Interpolate between the Melen-Turner and Sorted-Greedy algorithm,
- Two-phase algorithm bridged by threshold C :
 - **Phase 1:** Route a subset of the flows via **Melen-Turner algorithm** with congestion $\leq C$

Key idea for the routing algorithm

- Interpolate between the Melen-Turner and Sorted-Greedy algorithm,
- Two-phase algorithm bridged by threshold C :
 - **Phase 1:** Route a subset of the flows via **Melen-Turner algorithm** with congestion $\leq C$
 - **Phase 2:** Route remaining flows via **Sorted-Greedy algorithm** without increasing congestion $> C$

Key idea for the routing algorithm

- Interpolate between the Melen-Turner and Sorted-Greedy algorithm,
- Two-phase algorithm bridged by threshold C :
 - **Phase 1:** Route a subset of the flows via **Melen-Turner algorithm** with congestion $\leq C$
 - **Phase 2:** Route remaining flows via **Sorted-Greedy algorithm** without increasing congestion $> C$

- **Theorem #2:** If $C = 1.8$, then algorithm returns a routing with congestion ≤ 1.8 for all sets of flows

Key idea for the routing algorithm

- Interpolate between the Melen-Turner and Sorted-Greedy algorithm,
- Two-phase algorithm bridged by threshold C :
 - **Phase 1:** Route a subset of the flows via **Melen-Turner algorithm** with congestion $\leq C$
 - **Phase 2:** Route remaining flows via **Sorted-Greedy algorithm** without increasing congestion $> C$

- **Theorem #2:** If $C = 1.8$, then algorithm returns a routing with congestion ≤ 1.8 for all sets of flows; **however, it is not a 1.8-approximation algorithm**

A 1.8-approximation algorithm that guarantees congestion 1.8

- Two-phase algorithm bridged by threshold C and lower bound L on OPT
 - **Phase 1:** Route a subset of the flows including all with demand $> \frac{1}{3} \times L$ via Melen-Turner algorithm with congestion $\leq C \times L$
 - **Phase 2:** Route remaining flows via Sorted-Greedy algorithm without increasing congestion $> C \times OPT$
- **Theorem #2:** If $C = 1.8$, then algorithm is a 1.8-approximation algorithm, and returns a routing with congestion ≤ 1.8 for all sets of flows

Towards congestion-free data-center networks?

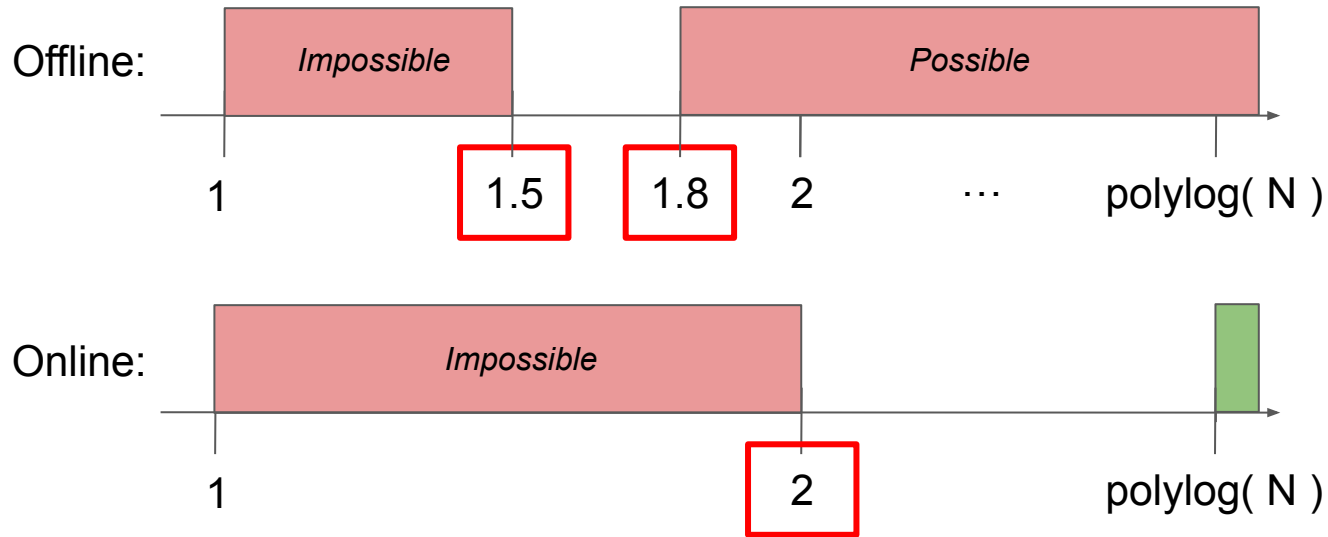
Towards congestion-free data-center networks?

- **Question #1** (Virtual machine placement): What if we jointly **place virtual machines** in servers and route the flows between them?

Towards congestion-free data-center networks?

- **Question #1** (Virtual machine placement): What if we jointly **place virtual machines** in servers and route the flows between them?
- **Question #2** (Multi-path routing): Does jointly route flows and **divide their demands** over a *constant* number of paths guarantee a congestion-free network?

Conclusion



Miguel Ferreira, maferrei@andrew.cmu.edu
CMU, IST