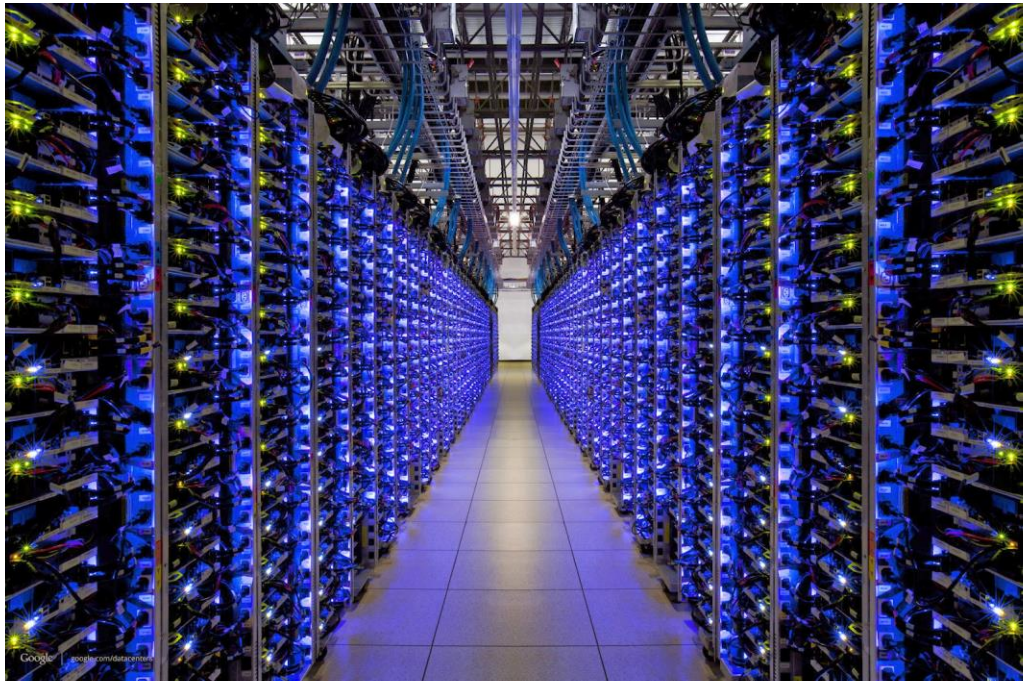
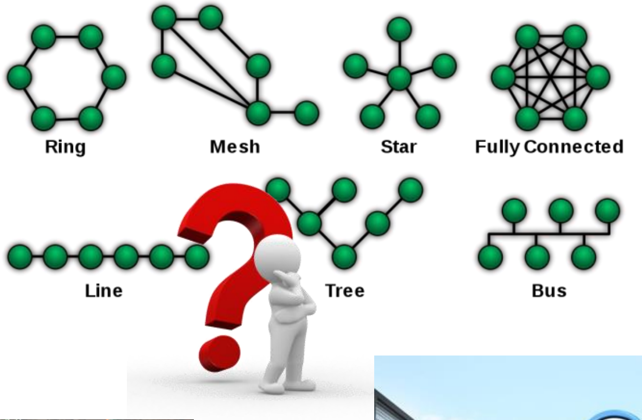
universität wien

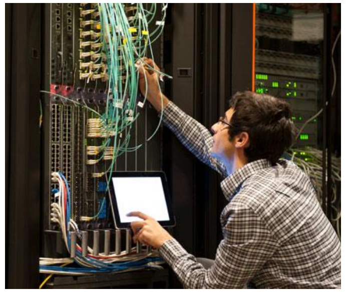
# Load-Optimization in Reconfigurable Networks: Algorithms and Complexity of Flow Routing

Wenkai Dai, Klaus-T. Foerster, David Fuchssteiner,  Stefan Schmid (CT Group, University of  Vienna)

# Motivation: Interconnecting Top of Rack in Datacenter

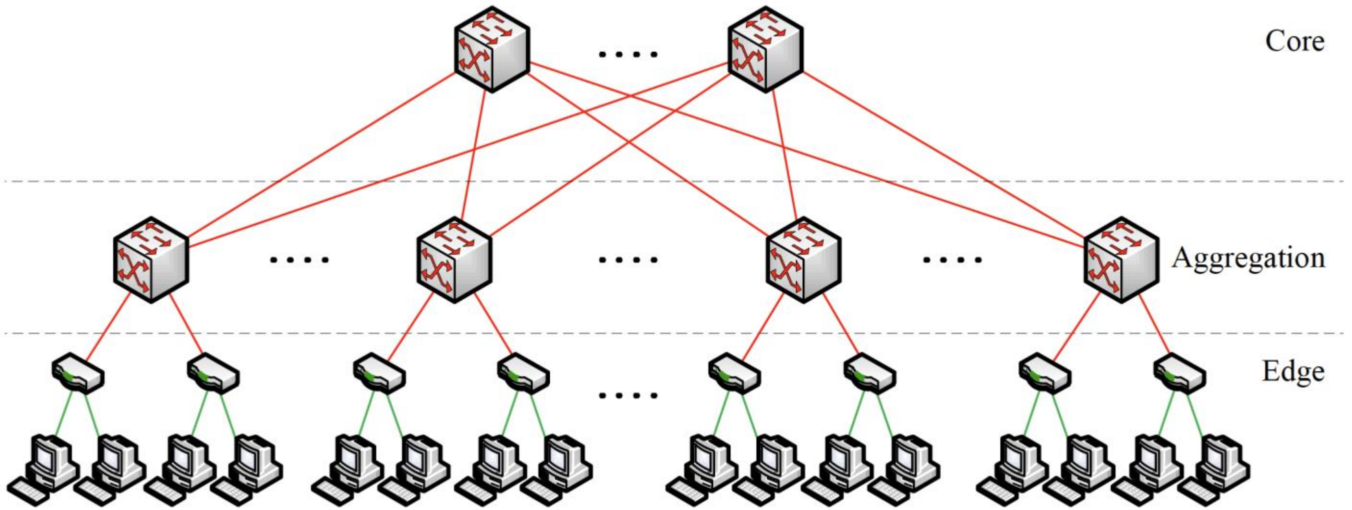


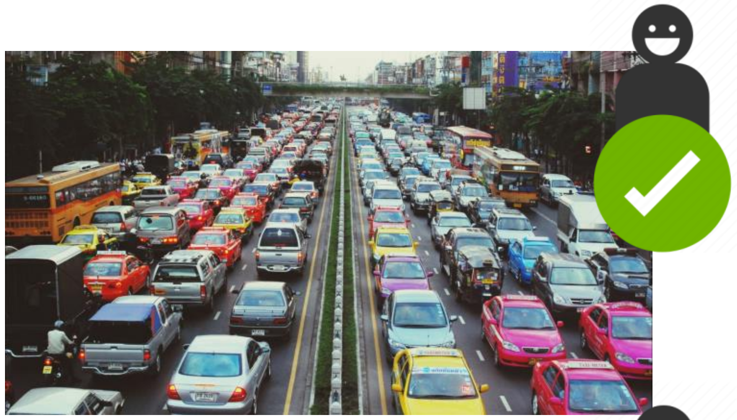


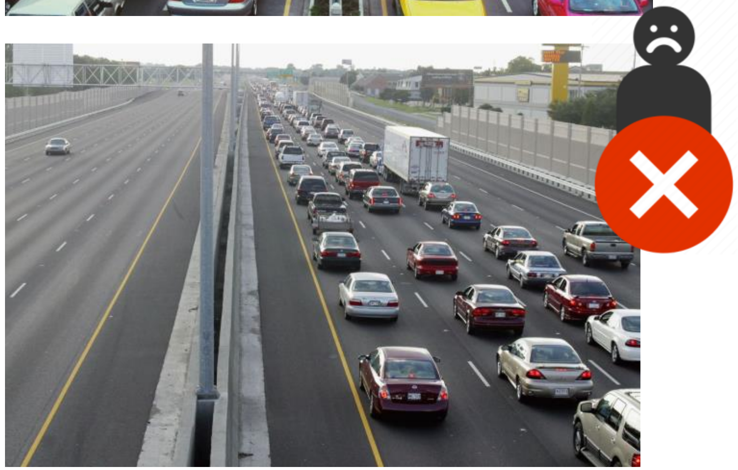



# Fat-Tree (Clos) Topology for Data Centers

- Fat-Tree is good for all-to-all traffic



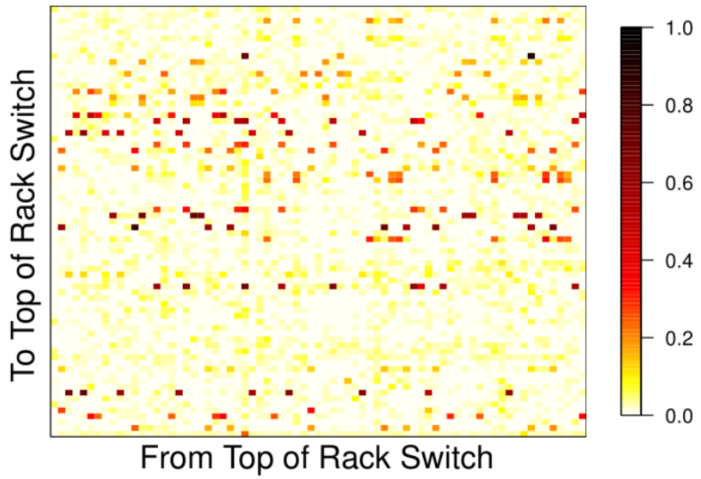





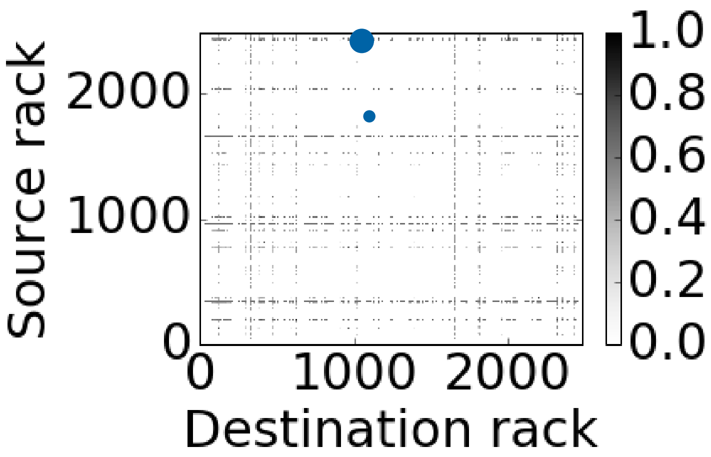
# Data Center Traffic ≠ Uniform

- However, DCN traffic is often ***not*** all-to-all

“Data reveal that 46-99% of the rack pairs exchange no traffic at all”



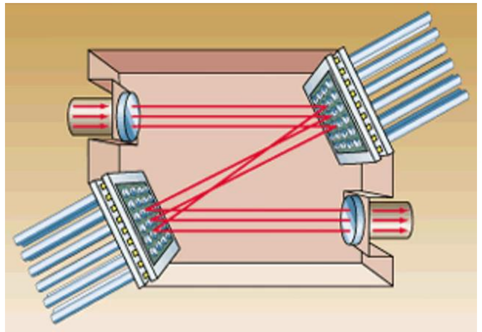

Traffic demands (normalized) between ToR switches. Halperin et al., SIGCOMM’11




Heatmap of rack to rack traffic. Color intensity is log-scale and normalized. Ghobadi et al., SIGCOMM’16

# Circuit Switches vs Packet Switches

1. Circuit Switches: usually *optical*
   - Fast (high bandwidth)
   - Connection between ports can be adjusted dynamically



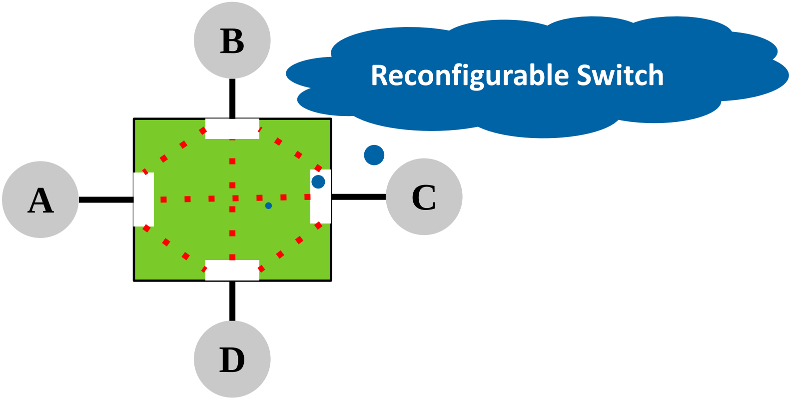
https://www.laserfocusworld.com/optics/article/16556781/many-approaches-taken-for-alloptical-switching (Hecht, 2001)

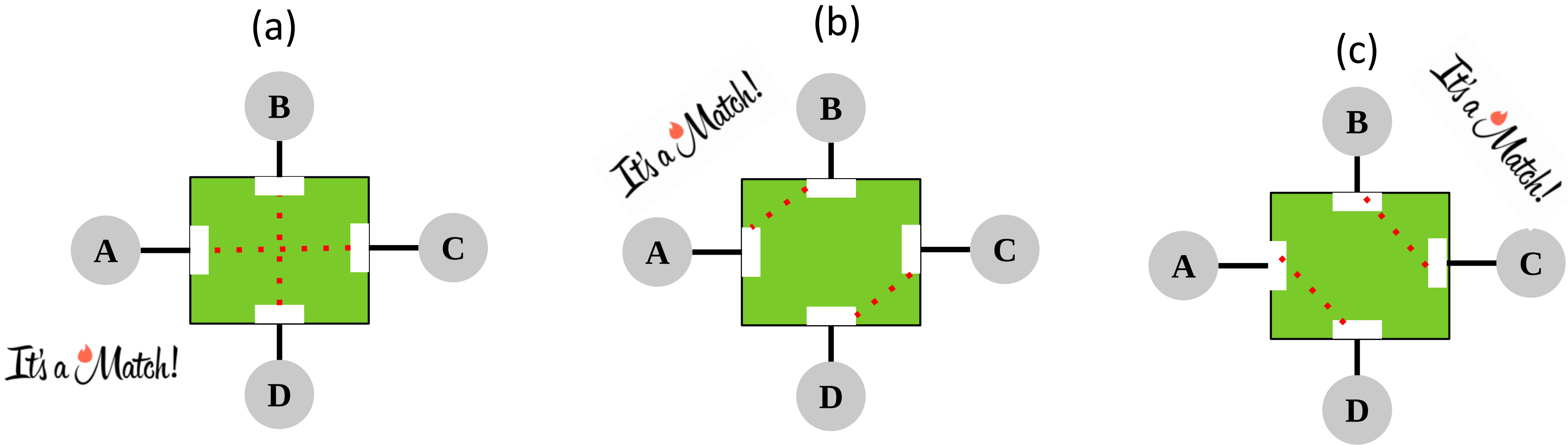
2. Packet Switches: usually electronic
   - Low bandwidth
   - The connections of links are fixed after deployment

# Understand Circuit Switches Physical layer: It‘s a Match(ing)!

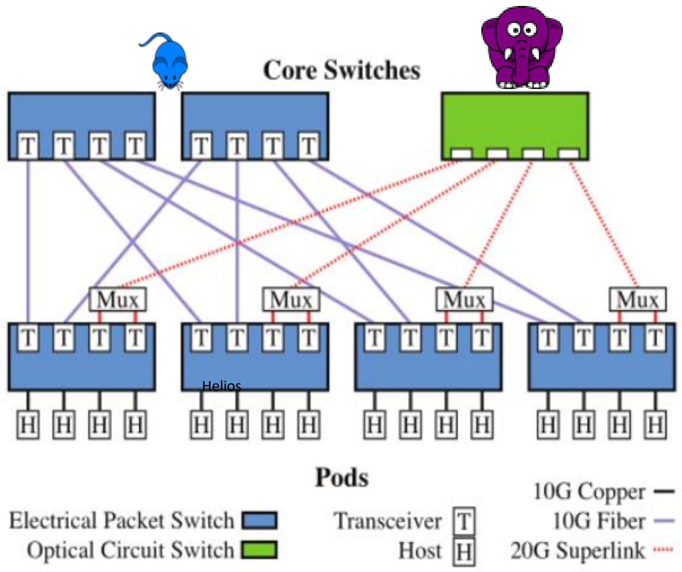
- Idea: implement “physical” connections
  - Difference: Not all-to-all switch
    - E.g. just 1 connection per node

- A matching is selected to connect nodes



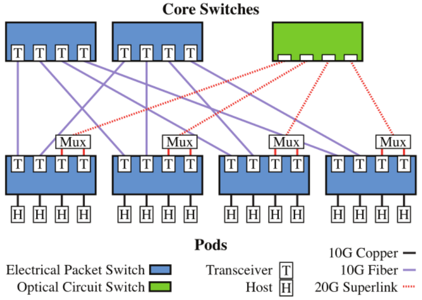
# Hybrid Architecture for Datacenter



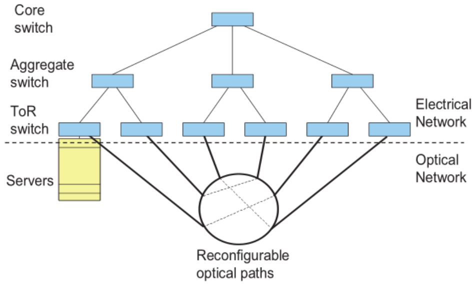

Helios, Farrington et al., SIGCOMM ‘10

- Adjust the topology **dynamically** for variant demands:
  - Elephant (big) flows → Circuit Switches
  - Mice (small) flows → Packet Switches

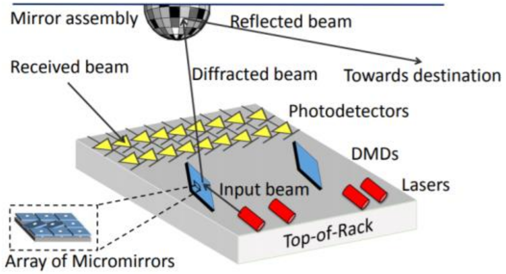
# Reconfigurable Data Center Networks (DCNs)



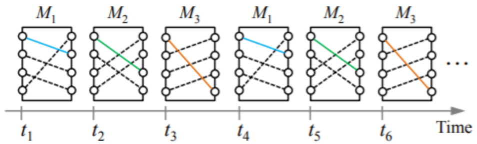

*Helios* (core)
Farrington *et al.*, SIGCOMM '10



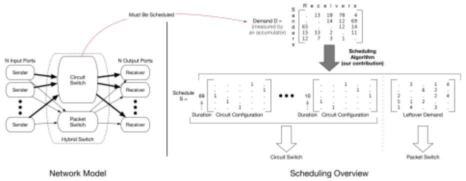

*c-Through* (*HyPaC* architecture)
Wang *et al.*, SIGCOMM '10



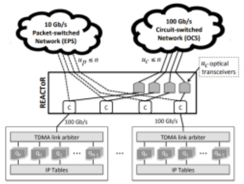

*ProjecToR* interconnect
Ghobadi *et al.*, SIGCOMM '16



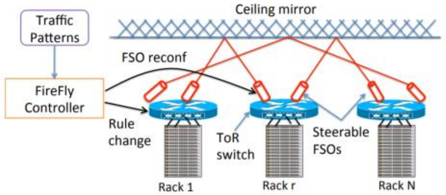

*Rotornet* (rotor switches)
Mellette *et al.*, SIGCOMM '17




*Solstice* (architecture & scheduling)
Liu *et al.*, CoNEXT '15




*REACToR*
Liu *et al.*, NSDI '15

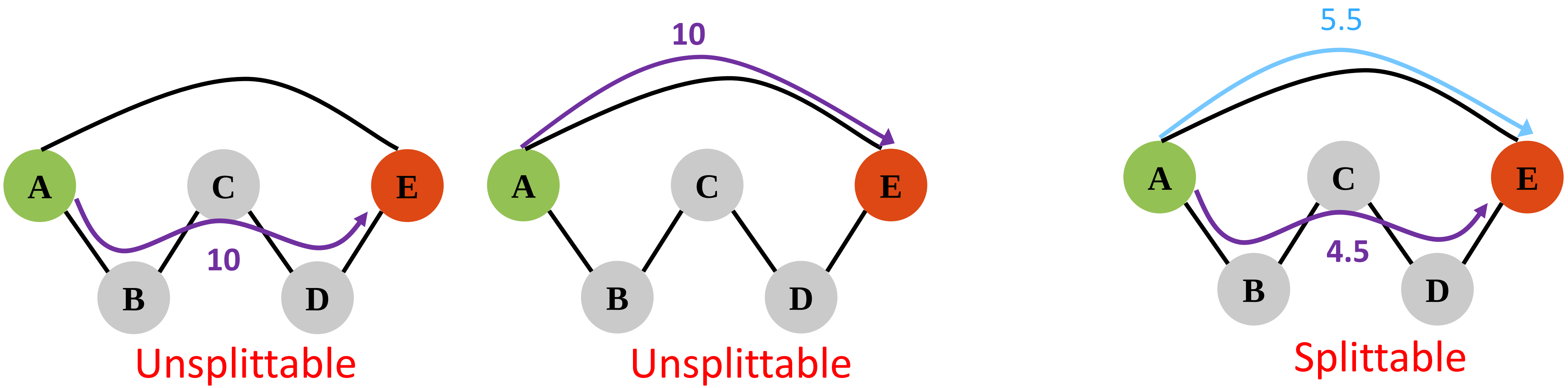



*FireFly*
Hamedazimi *et al.*, *SIGCOMM* '14

*… and many more …*

# Routing Models: Unsplittable vs Splittable

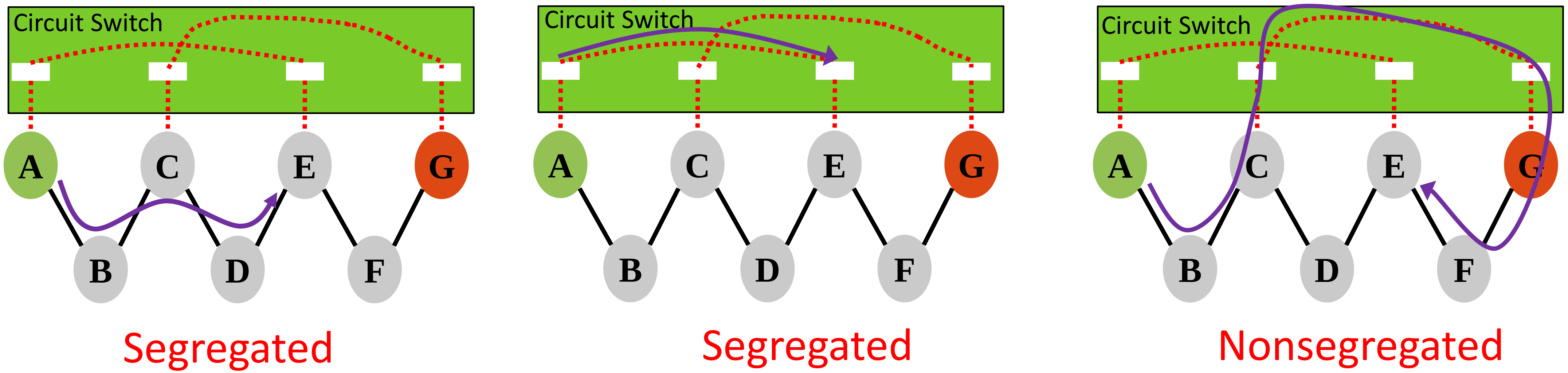
- For each demand, e.g., **A→E: 10**



# Routing Models: Segregated vs Nonsegregated

- In a reconfigurable datacenter, for each demand:

E.g., demand: A→E



# Four Routing Models in Reconfigurable Networks

| Routing Models | Segregation Model | Nonsegregation Model |
| --- | --- | --- |
| **Splittable Model** | SS | SN |
| **Unsplittable Model** | US | UN |

# Load-Optimization Reconfiguration Problem (Our Problem)

- Given: A routing model $\tau \in \{SS, SN, US, UN\}$

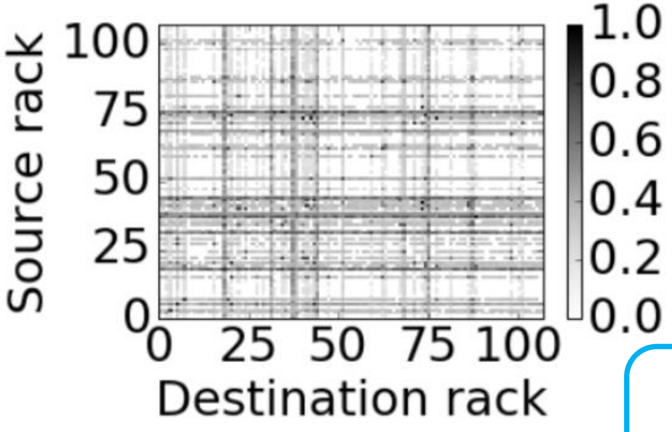
| Routing Models | Segregation Model | Nonsegregation Model |
|---|---|---|
| Splittable Flow | SS | SN |
| Unsplittable Flow | US | UN |

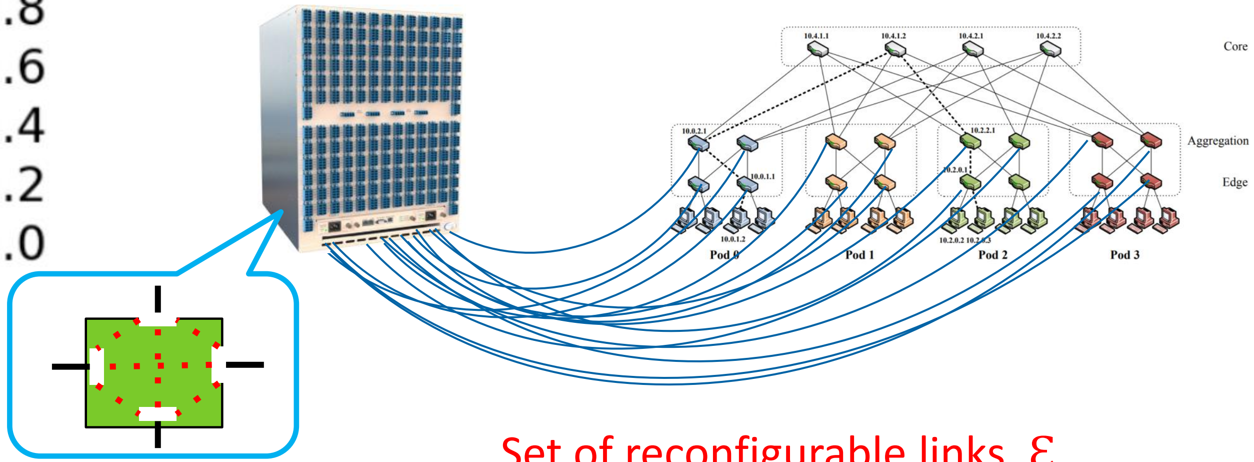
Circuit Switches

Static Network $\mathrm{N} = (V, E, C)$

From: calient.net

From: Al-Fares et al. 2008




Demands Matrix $D$

Set of reconfigurable links $\mathcal{E}$

# Load-Optimization Reconfiguration Problem (Our Problem)

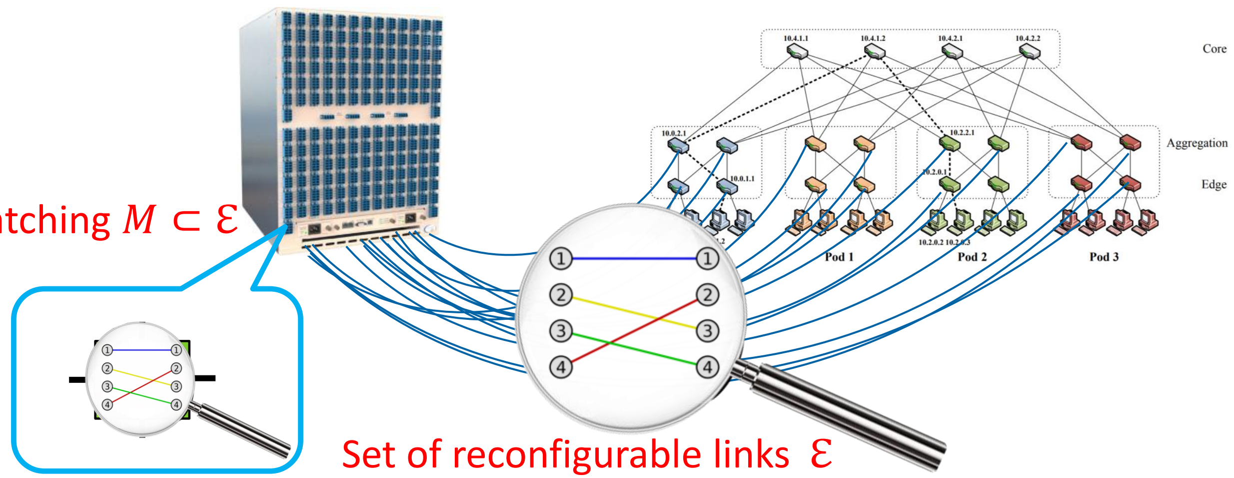
- Compute: a matching from reconfigurable links; and optimal routing schemes for demands

Circuit Switches

Static Network $\mathrm{N} = (V, E, C)$

From: calient.net

From: Al-Fares et al. 2008

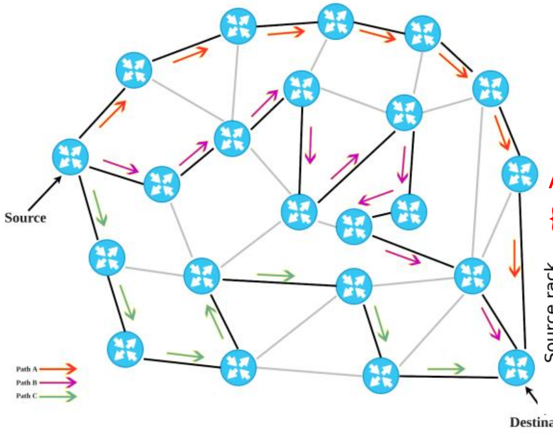



Matching $M \subset \mathcal{E}$

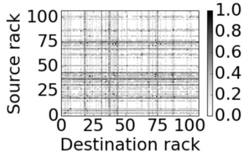
Set of reconfigurable links $\mathcal{E}$

+

From: cisco.com




A routing model $\tau \in \{SS, SN, US, UN\}$

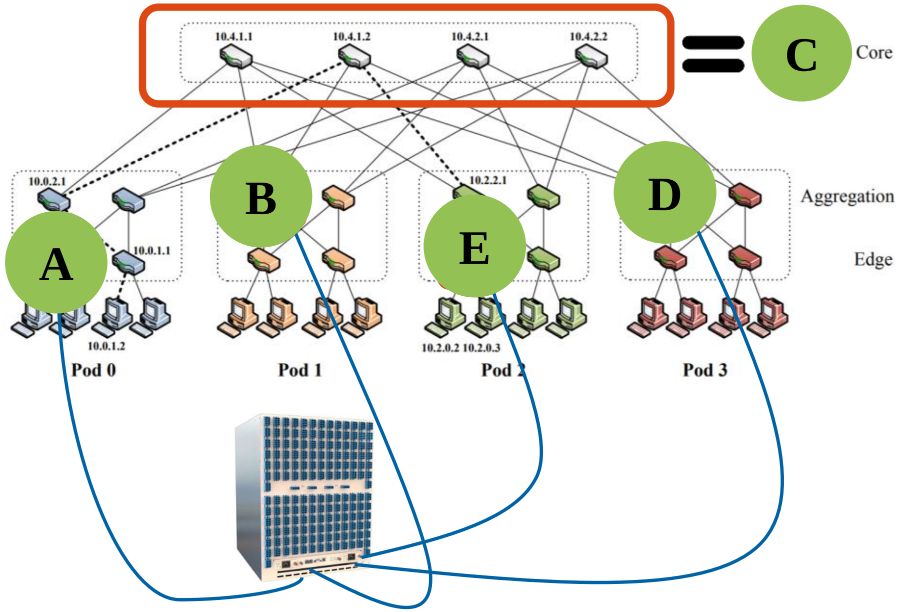


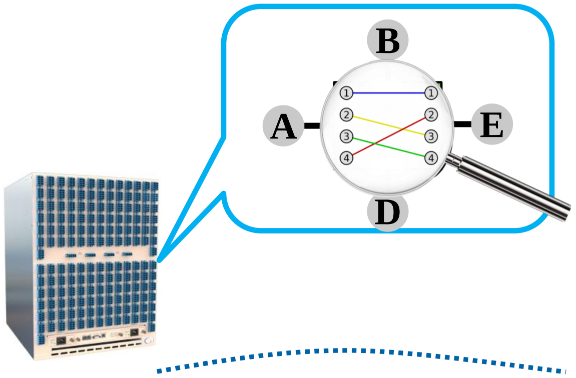

Optimal routing schemes for demands in the hybrid network $(V, E \cup M, C)$

- Objective: minimize the maximum link load in the hybrid network $(V, E \cup M, C)$

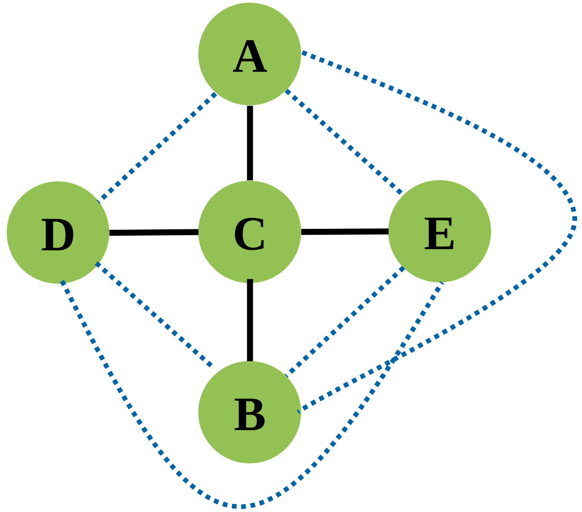
# An Example For Load-Optimization Reconfiguration Problem

From: Al-Fares et al. 2008



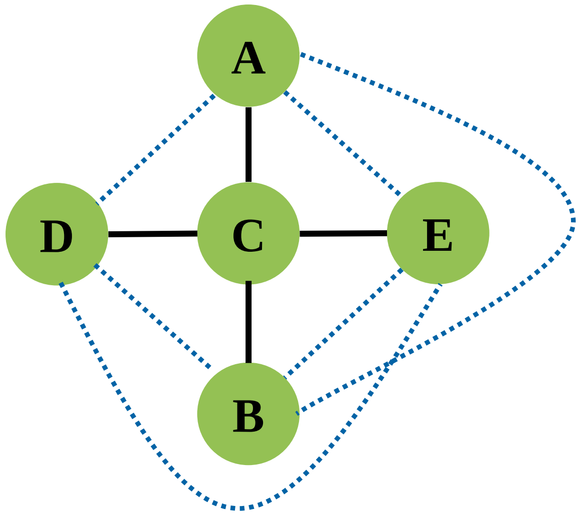

Reconfigurable links




Reconfigurable network

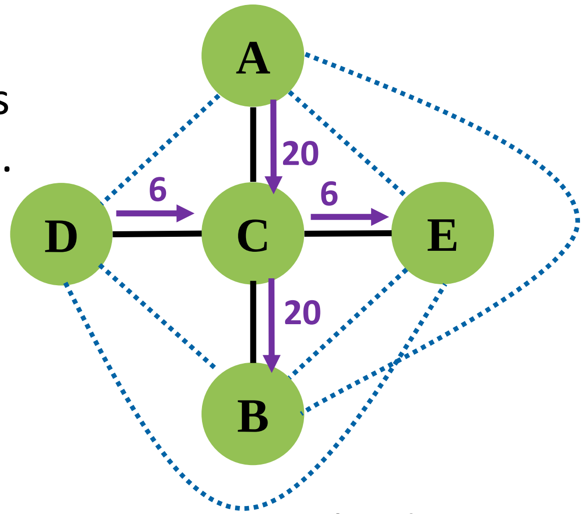
# Example: Loads Depend on Reconfigurations

- Consider demands D: **A→B: 8, A→C: 6, C→B: 6, D→B: 6, A→E: 6**
- Goal: determine a matching in reconfigurable links to minimize the maximum load




Reconfigurable network

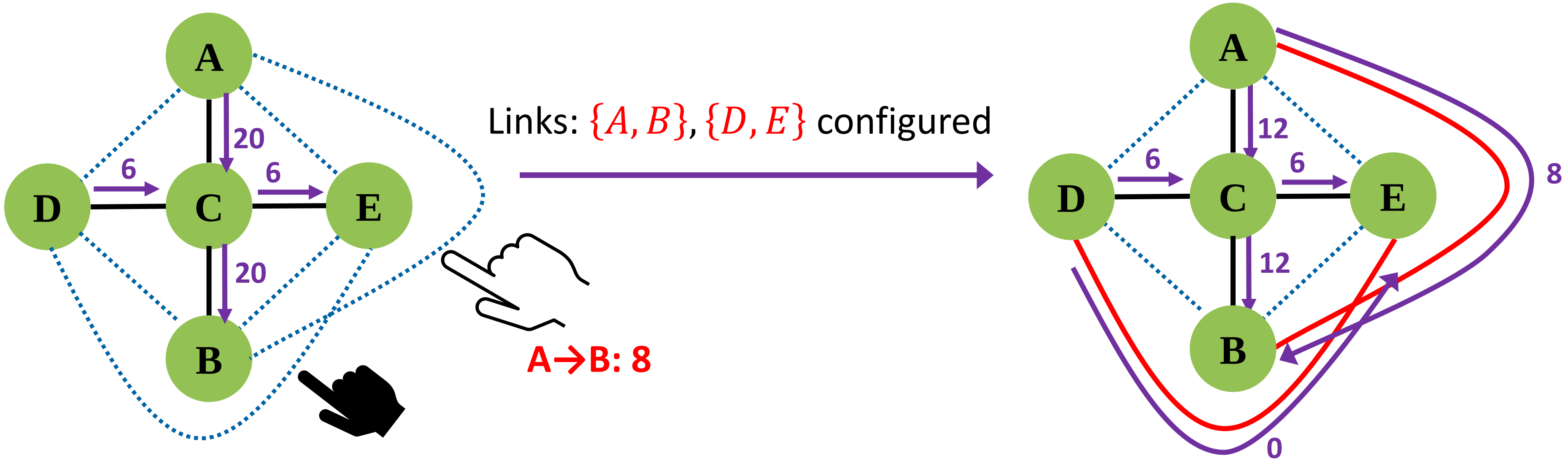
Compute flows for demands without reconfigurable links.




Maximum load 20

# Example: Determine Matching by Greedy

- *Demands D*: **A→B: 8, A→C: 6, C→B: 6, D→B: 6, A→E: 6**
  - Greedy chooses $\{A, B\}$ to serve **A→B**, then the matching is $\{A, B\}$ and $\{D, E\}$

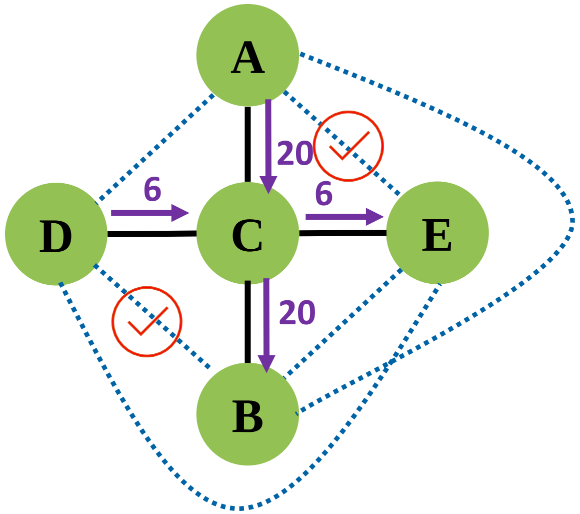



Maximum load 20

Greedy -> maximum load 12

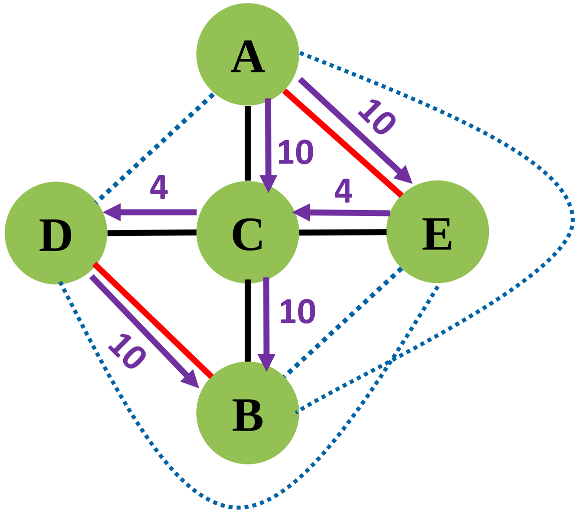
# Example: Optimal Matching

- *Demands D*: **A→B: 8, A→C: 6, C→B: 6, D→B: 6, A→E: 6**
  - The optimal matching is $\{D, B\}$ and $\{A, E\}$



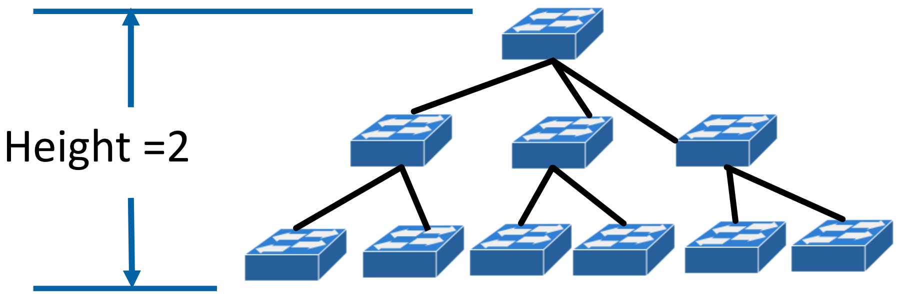

Maximum load 20

Links: $\{A, E\}, \{D, B\}$ configured




Optimal -> maximum load 10

# Complexity for Simple Trees

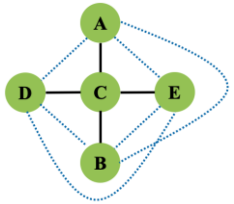



- If the given static network is a tree with a height >=2, then

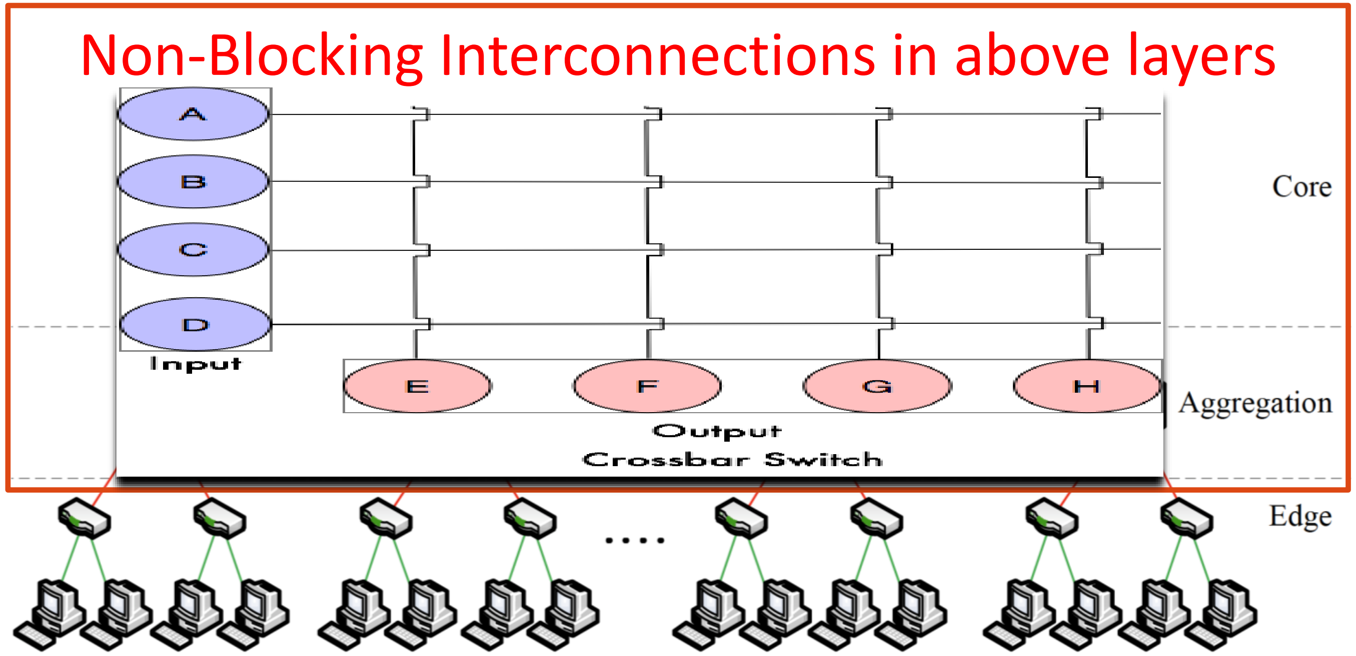
| Time Complexity | Segregation Model | Nonsegregation Model |
|---|---|---|
| Splittable Model | SS is strongly NP-hard | SN is strongly NP-hard |
| Unsplittable Model | US is strongly NP-hard | UN is strongly NP-hard |

  - Reduction from 3-Partition problem

- Especially, UN model is weakly NP-hard for star networks
  - Reduction from 2-Partition problem
  - Not hard anymore for small demands

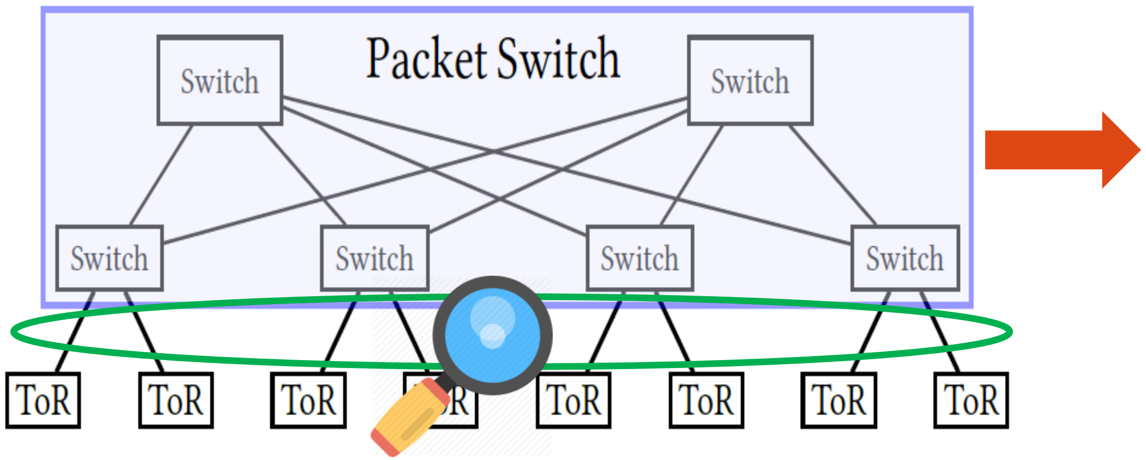


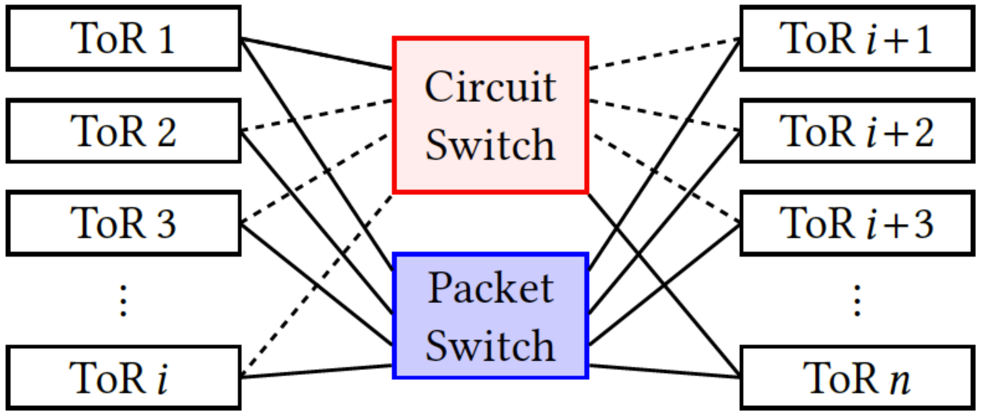
# Non-Blocking Interconnects, e.g., Clos, Fat-Tree etc.



# Simplified Problem defined by Non-Blocking Interconnections

Above layers abstracted as a packet switch.

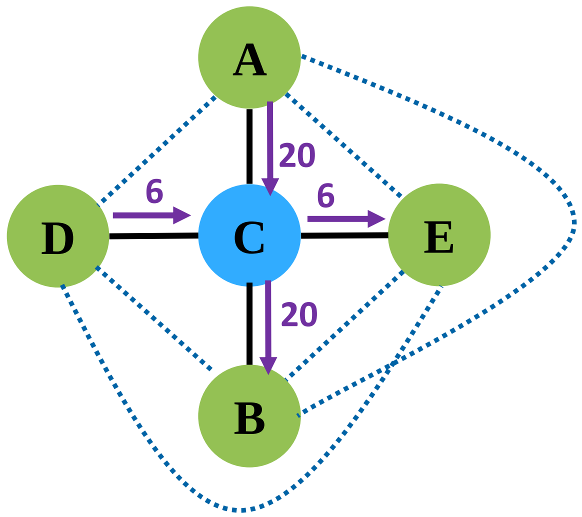



(Mohammad Alizadeh et al. 2016).

# Optimal Algorithms for Simplified Problem (Notations)

- Consider a decision problem
- Assume the optimized maximum load: $\theta$
- Let $S$ be the set of possible values for $\theta$
- $S$ contains the load for each static link before reconfiguration
- Next, we show how to compute the set $S$

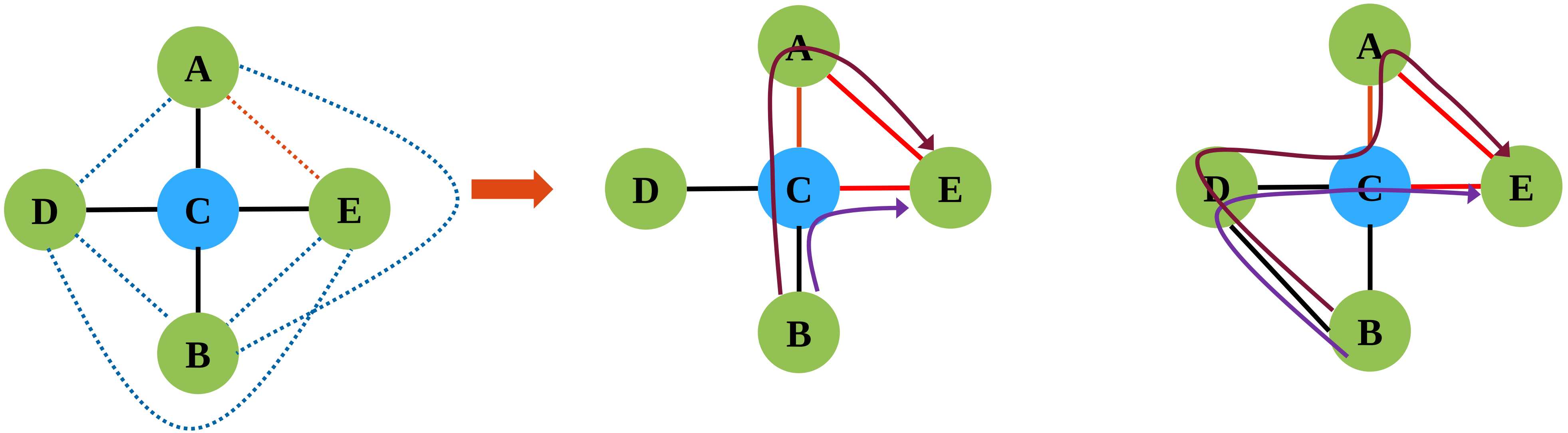



The set $S$

## Useful Observations

- If a reconfigurable link is selected, it defines triangle.
- E.g., the triangle $\{A, E, C\}$

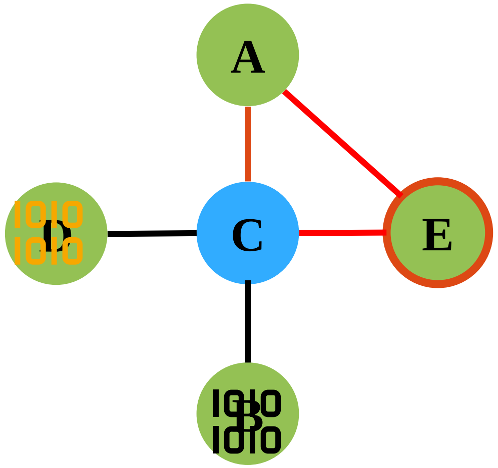
E.g., demand : **B→E**

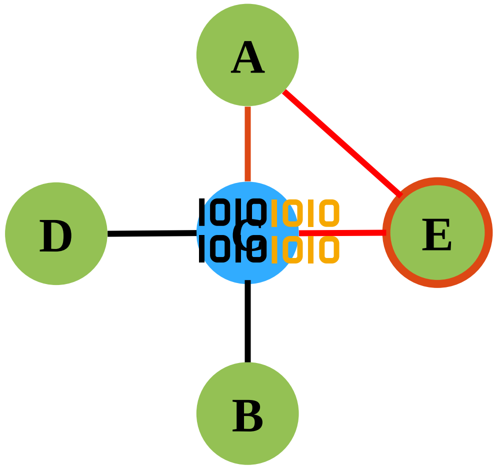


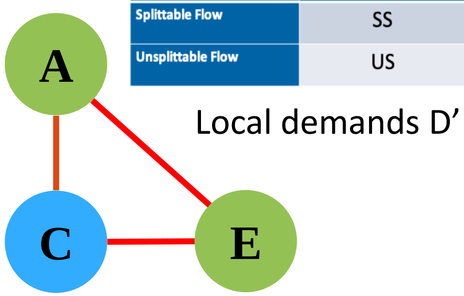
# Local Optimization For Each Triangle

- For each reconfigurable link $\{X, Y\}$, in the triangle $\{X, Y, C\}$:
  - Compute local demands, and find optimal load for the local demands

| Routing Models | Segregation Model | Nonsegregation Model |
|---|---|---|
| Splittable Flow | SS | SN |
| Unsplittable Flow | US | UN |



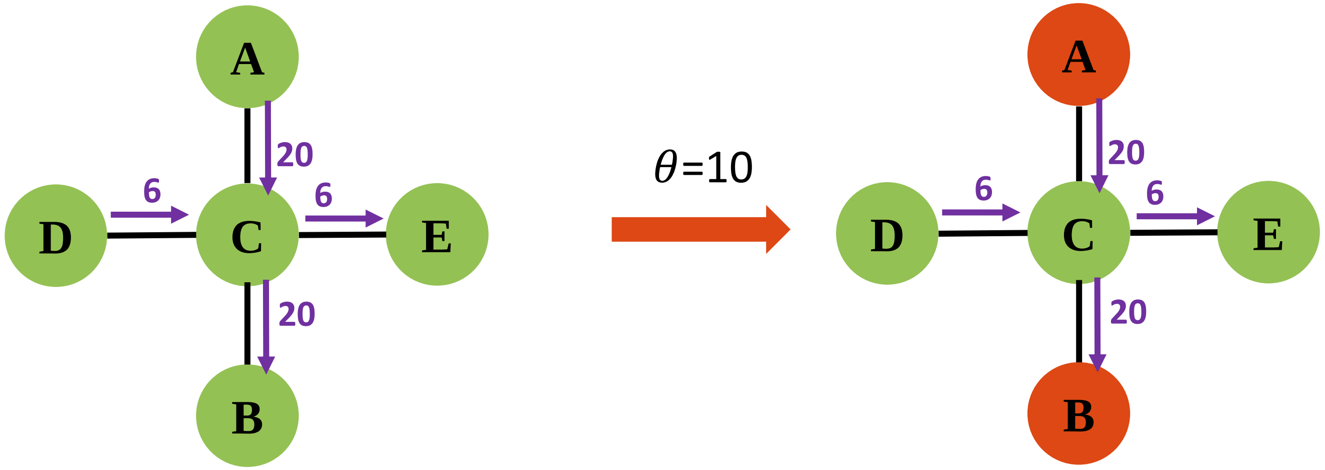




- Find optimal routing in $O(1)$
- Let the maximum load be $\Delta_i$
- Put $\Delta_i$ into the set $S$

Local demands :D'(**C→E**)=D(**B→E**)+D(**D→E**)
D'(**E→C**)=D(E→**B**)+D(**E→D**)

## Optimal Algorithm: Mark Target Nodes

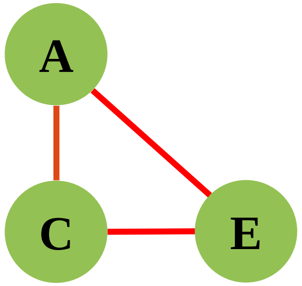
- Binary search in the set $S$ to find the actual $\theta$ (optimized maximum load) within $O(\log|V|)$
- For a specific $\theta$:
  - Mark each node "target" ($V^r \subseteq V$ ) if its link load is larger than $\theta$ before reconfiguration



# Optimal Algorithm: Compute Useful Reconfigurable Links

- For a specific $\theta$:
  - Define a set $\mathcal{E}'$: useful reconfigurable links, where $\mathcal{E}' \subseteq \mathcal{E}$
  - For each triangle, if its maximum load $\Delta_i \leq \theta$, put its reconfigurable link $\mathcal{E}'$

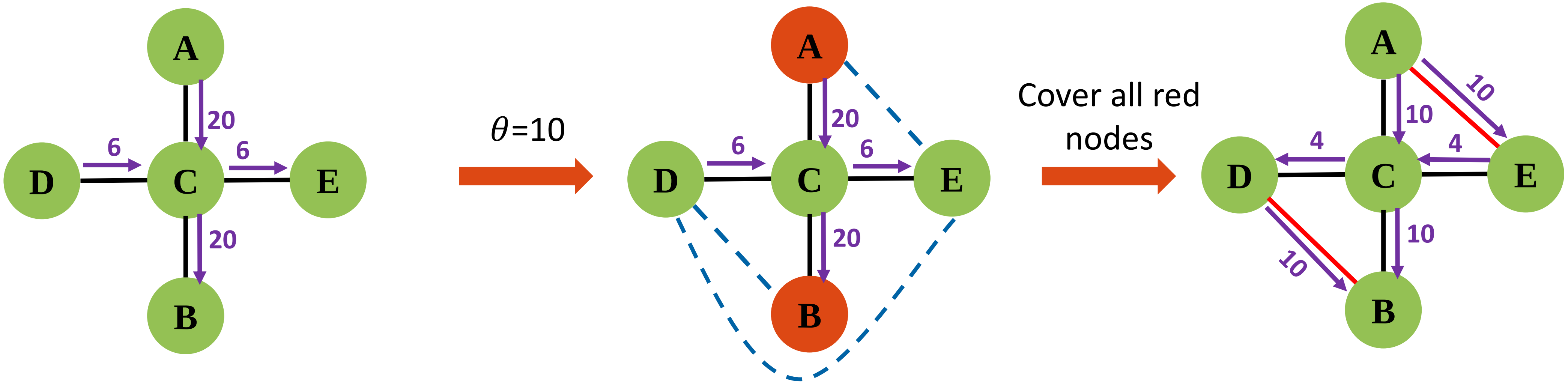
| Routing Models | Segregation Model | Nonsegregation Model |
|---|---|---|
| Splittable Flow | SS | SN |
| Unsplittable Flow | US | UN |




- Find optimal routing in $O(1)$
- Let the maximum load be $\Delta_i$
- If $\Delta_i \leq \theta$, put $\{A, E\}$ in the set $\mathcal{E}'$

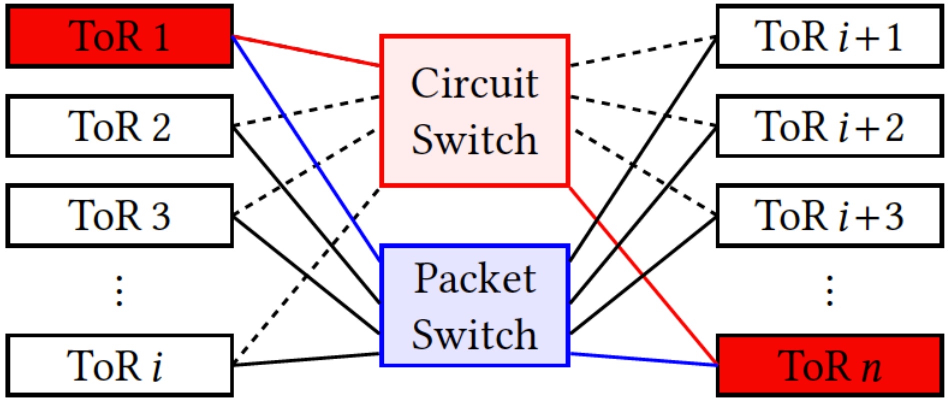
# Optimal Algorithm: Red-Target Matching and Binary Search

- For each specific $\theta$: ($V^r$ and $\mathcal{E}'$ computed )
  - Obtain a new graph $\mathrm{G}' = (V, \mathcal{E}')$
  - Find a matching $M$ in $\mathrm{G}'$ to cover all target nodes $V^r$ (by maximum weight matching)
- Total run-time cost: $O(\log|V| * T)$, and $T$ is the run-time of maximum weight matching



# Theoretical Analysis of Performance

- Lower bound: the maximum load decreased by 50% by adding reconfigurable links
- Why: at most two paths between any two nodes
- Our optimal algorithm achieves the lower bound
- Maximum matching works badly:
  - For some cases, maximum matching can only decrease the maximum load by an arbitrarily small value $\varepsilon$

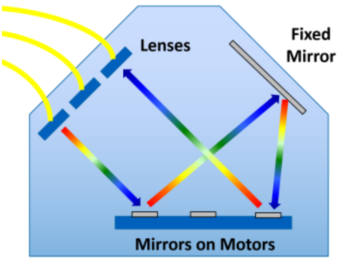


universität wien

**performance 2x, similar run time**

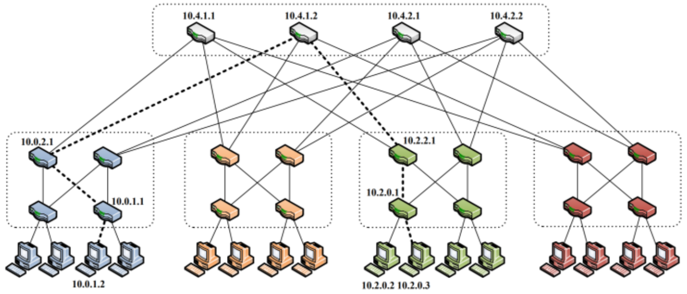
# Evaluation: Minimize Maximum Link Load

- Traces from facebook

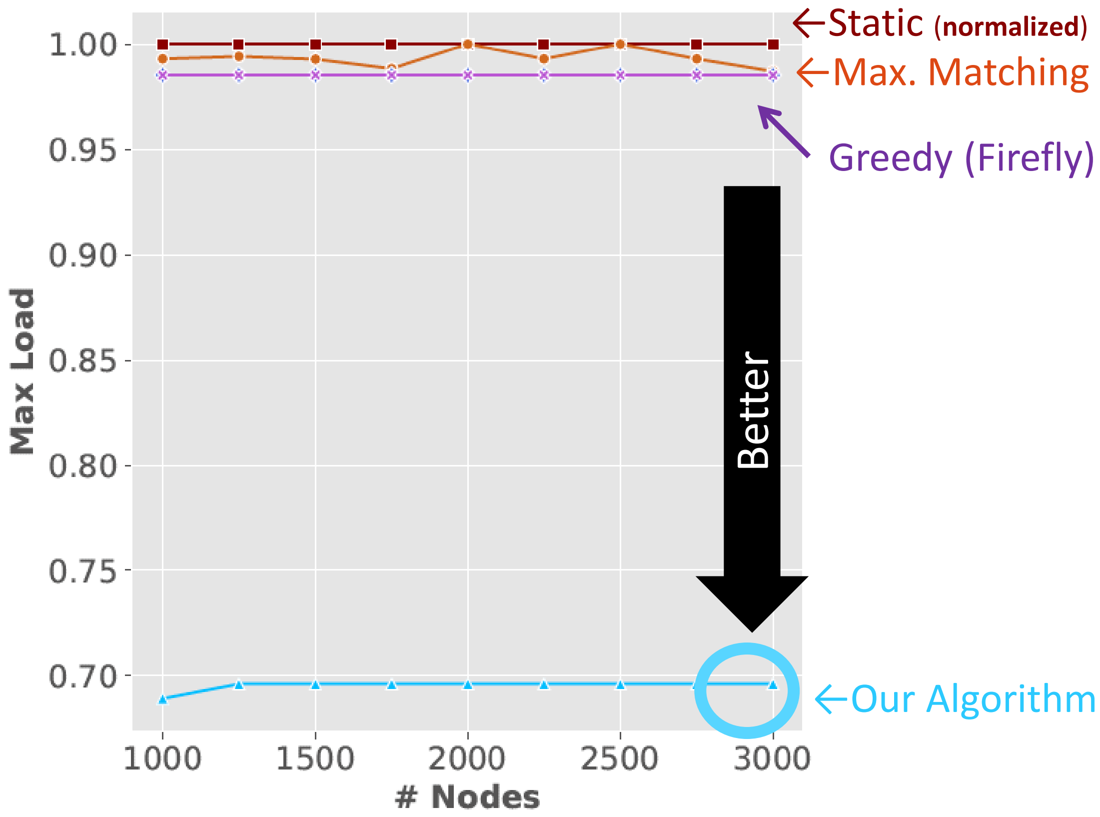


Optical Circuit Switch

+



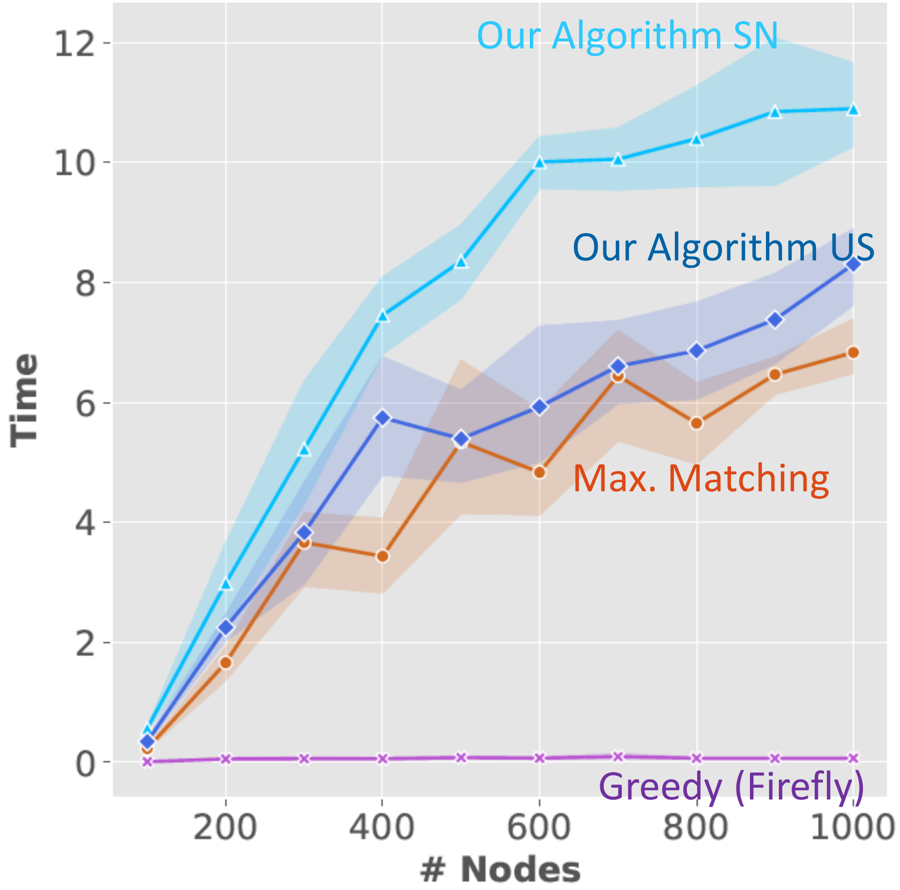

**Topology**



# Evaluation: Comparing Time Costs

- Theoretical Running Time:
  - Greedy: $O(|V|)$
  - Maximum Matching (Blossom Alg.): $O(|E||V|^2)$
  - Our Algorithm: $O(\log|V| * |E||V|^2)$

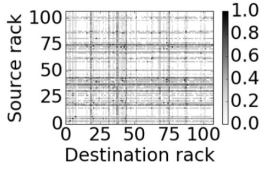
- The experiments match our theoretical analysis



# Thank you! ☺

# Load-Optimization in Reconfigurable Networks: Algorithms and Complexity of Flow Routing

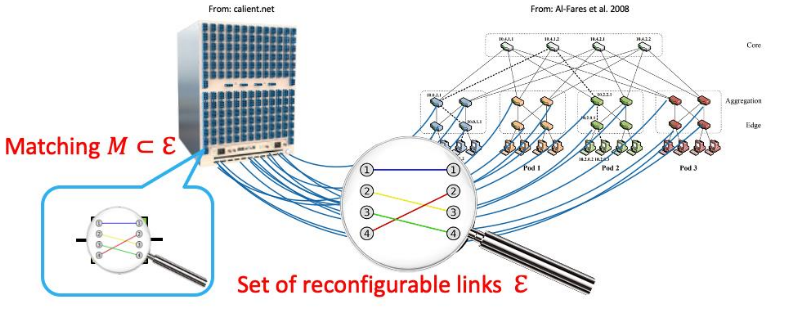
Wenkai Dai, Klaus-T. Foerster, David Fuchssteiner, Stefan Schmid (CT Group, University of Vienna)

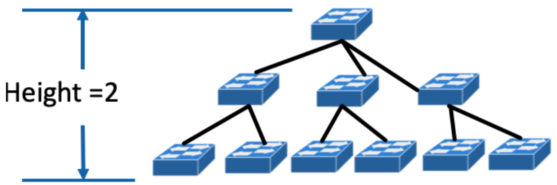



| Routing Models | Segregation Model | Nonsegregation Model |
|---|---|---|
| Splittable Flow | SS | SN |
| Unsplittable Flow | US | UN |

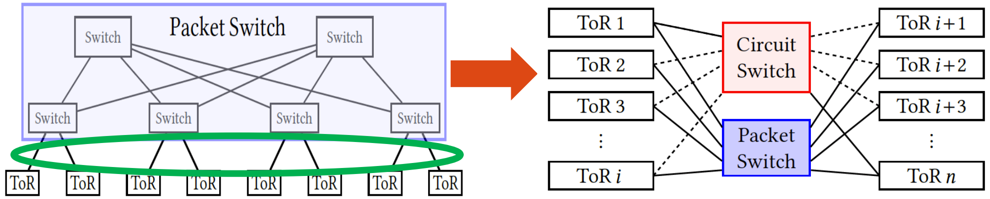
Circuit Switches

Static Network $N = (V, E, C)$

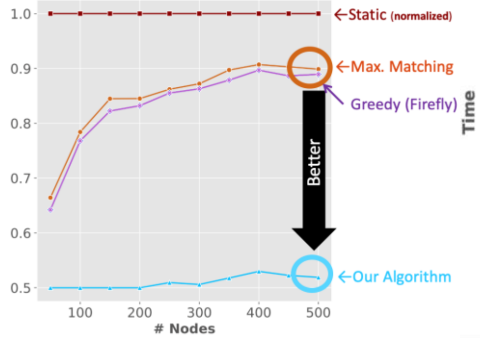


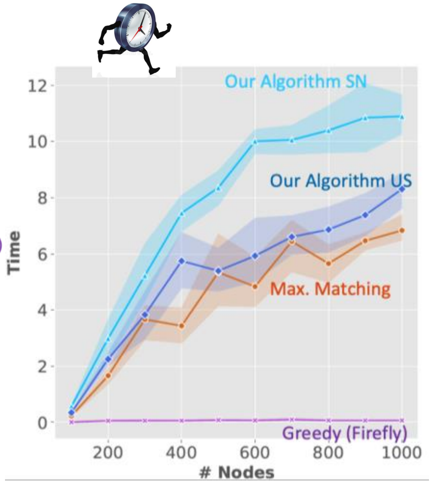




| Time Complexity | Segregation Model | Nonsegregation Model |
|---|---|---|
| Splittable Model | SS is strongly NP-hard | SN is strongly NP-hard |
| Unsplittable Model | US is strongly NP-hard | UN is strongly NP-hard |




**performance 2x, similar run time**




- Theoretical Running Time:
  - Greedy: $O(|V|)$;
  - Maximum Matching (Blossom Alg.): $O(|E||V|^2)$;
  - Our Algorithm: $O(\log|V| * |E||V|^2)$ .