University of Wrocław

PhD Thesis

Algorithmic aspects of contemporary networks

Maciej Pacut

supervised by
Dr hab. Marcin Bieńkowski

Thesis overview

Algorithmic aspects of contemporary networks

- Part 1: Utilizing redundant storage in MapReduce applications (ICNP '15 + TCS '17)
- **Part 2: Assigning tasks to machines in data centers** (DISC '17 + SIAM J. Disc. Math '20)
- Part 3: Caching forwarding tables in routers (ACM SPAA '17)

Assigning tasks to machines in data centers

The data center in CERN, European Organization for Nuclear Research

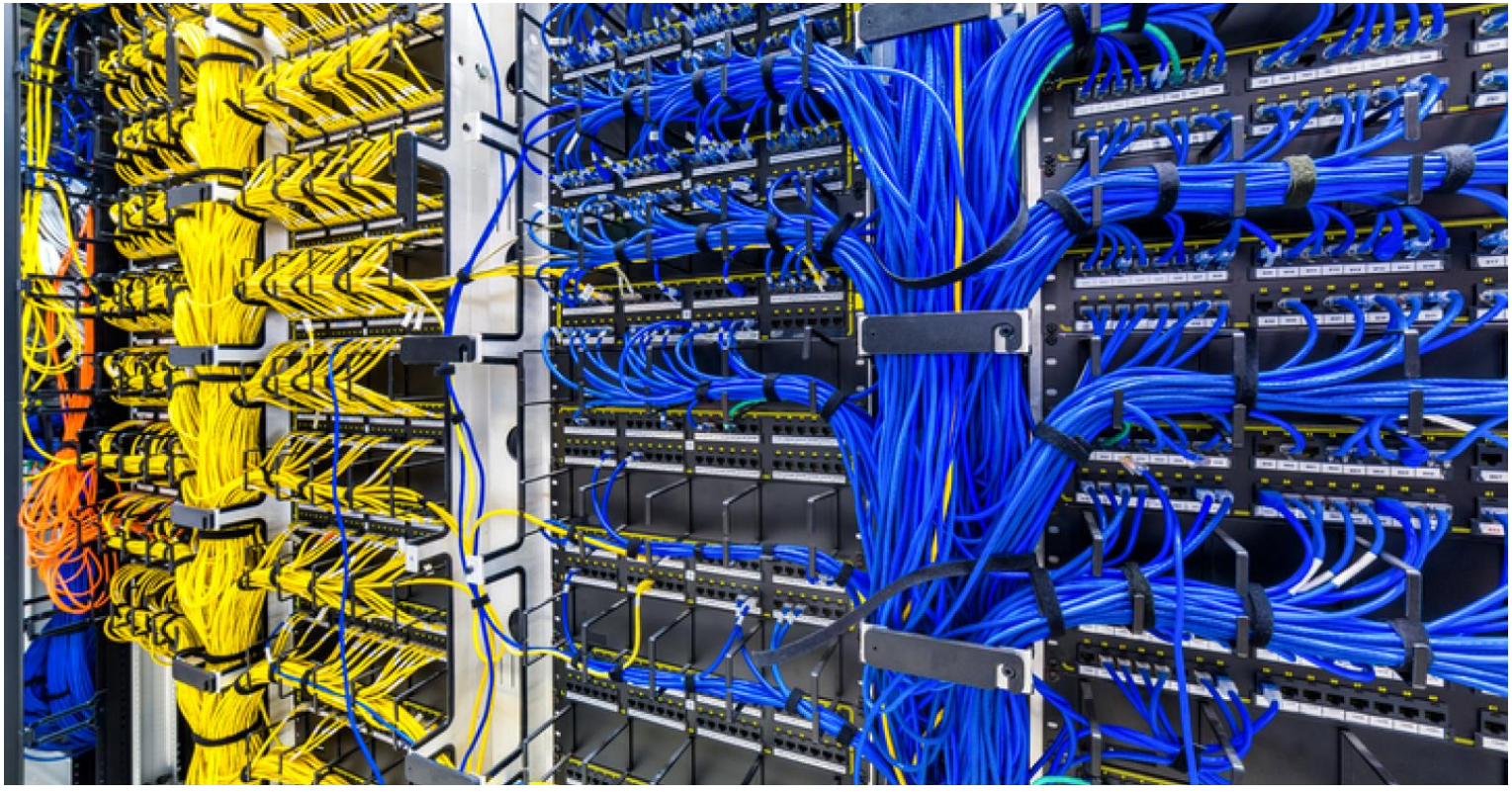
Data center architecture

Data center contains multiple physical machines.

IBM Blue Gene Supercomputer

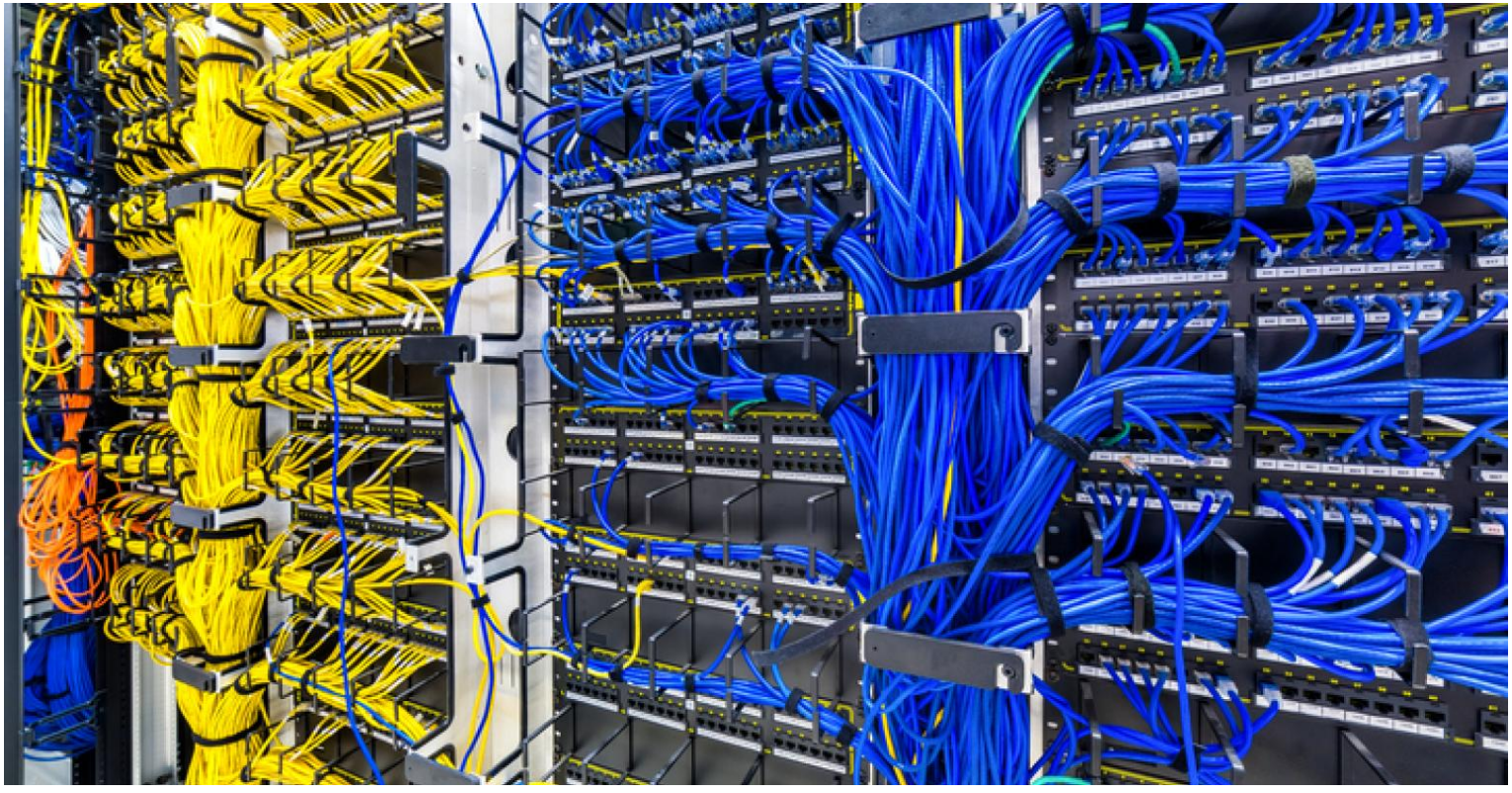
Data center architecture

Physical machines are connected by interconnecting network.



Data center architecture

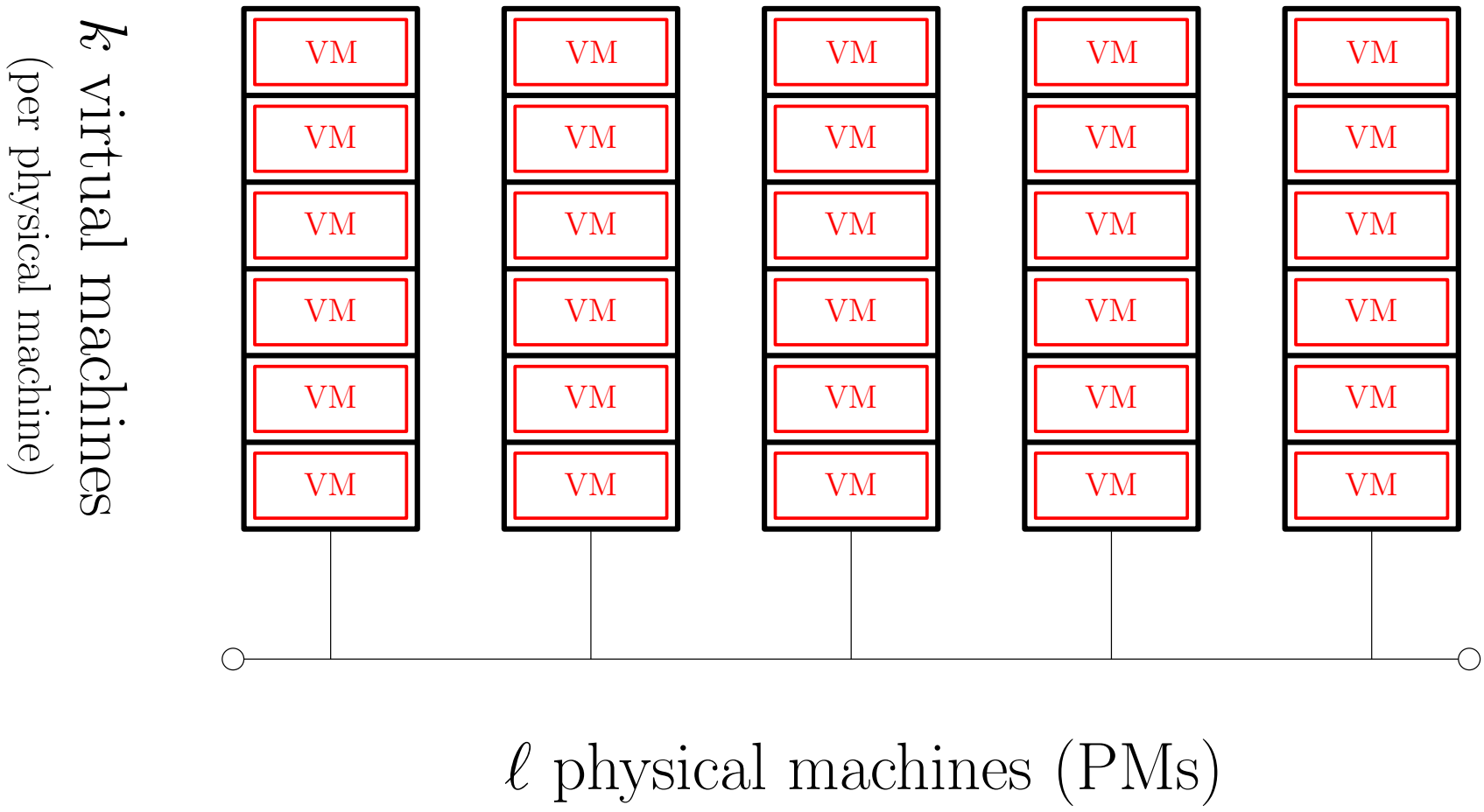
Physical machines are connected by interconnecting network.



Interconnecting network = performance bottleneck

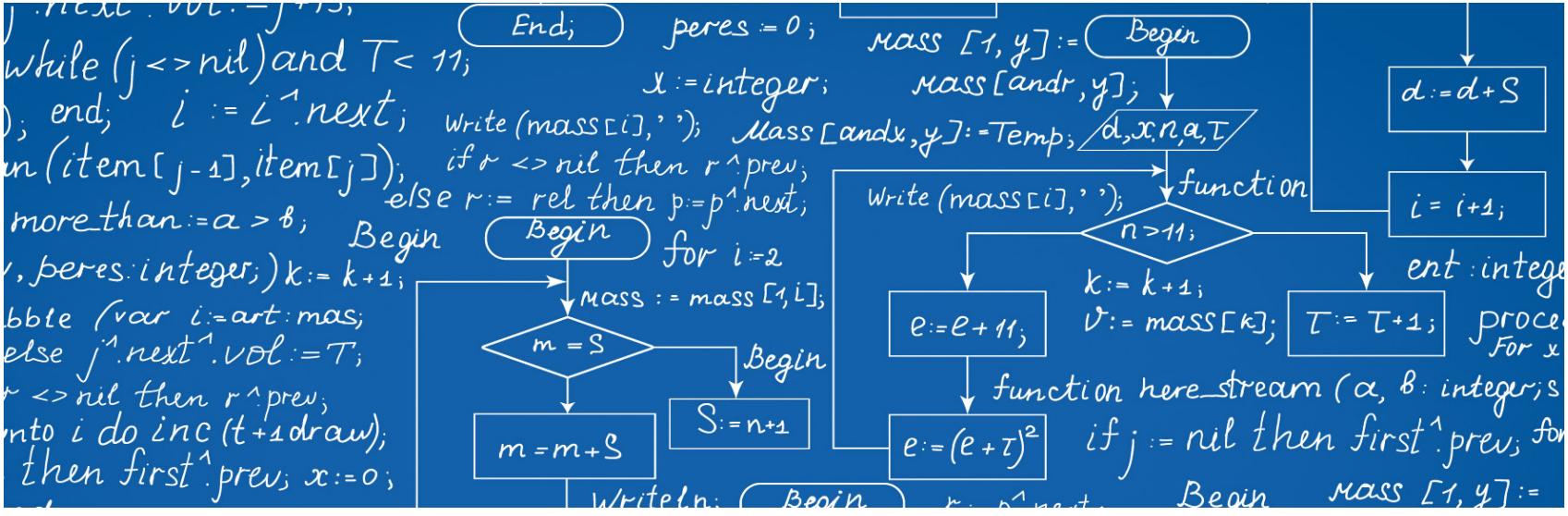
Data center architecture

Abstract view of physical machines and interconnecting network



Computational tasks in data centers

Client: I have *a* computational task to perform.

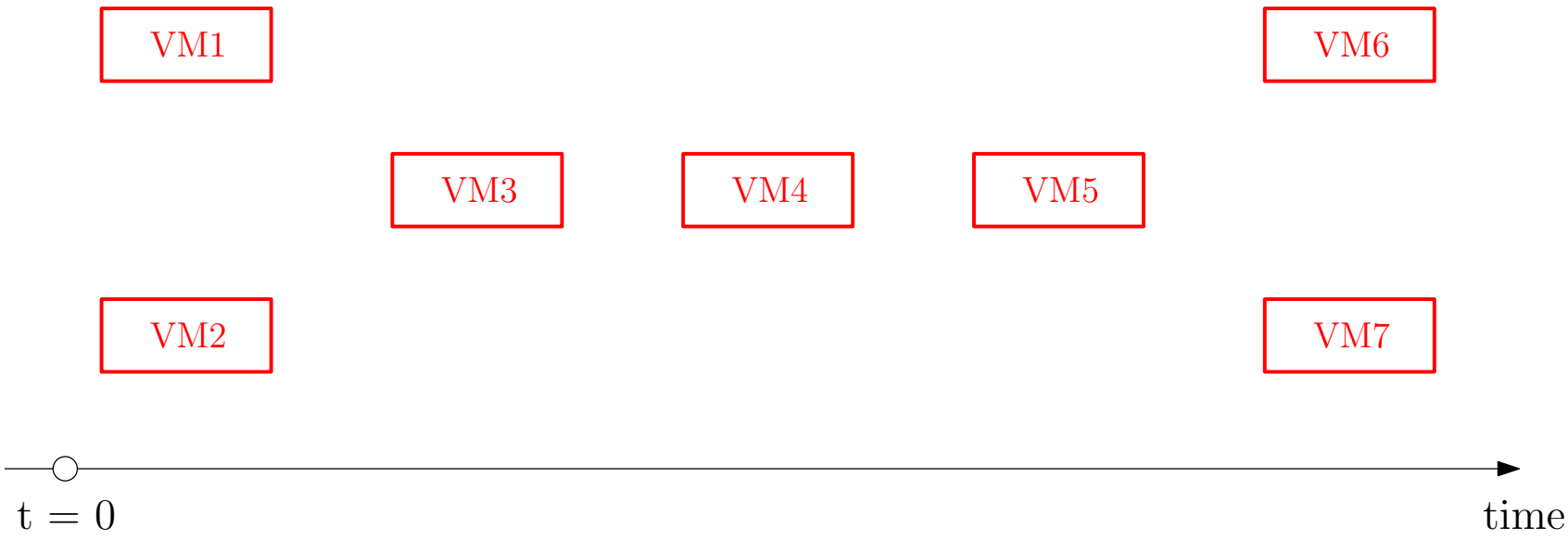


To finish in time, I need to run this task on 7 virtual machines!

VM1	VM2	VM3	VM4	VM5	VM6	VM7

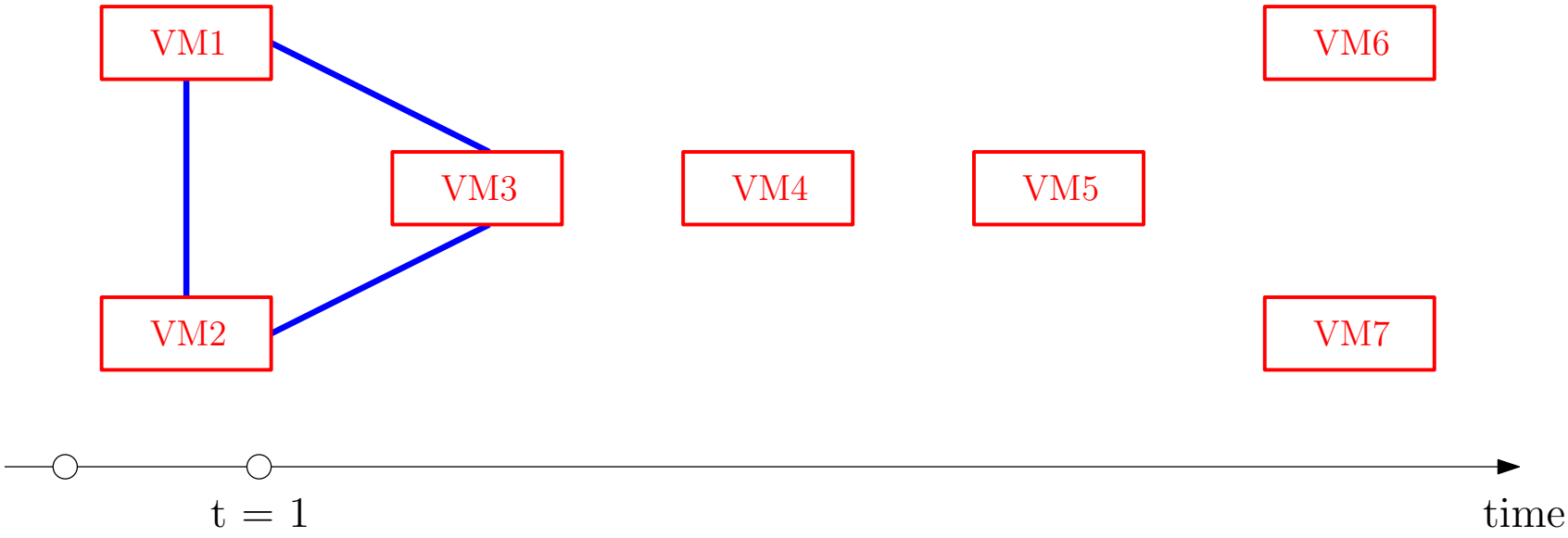
Computational tasks in data centers

During runtime VMs communicate with each other.



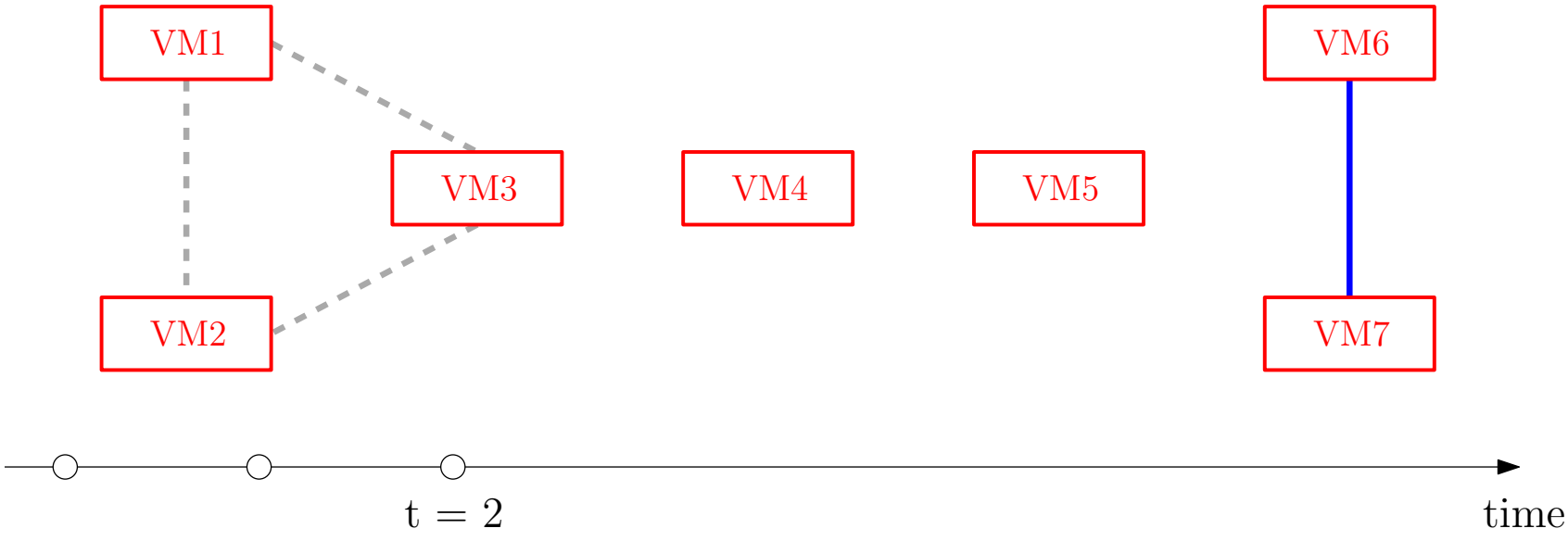
Computational tasks in data centers

During runtime VMs communicate with each other.



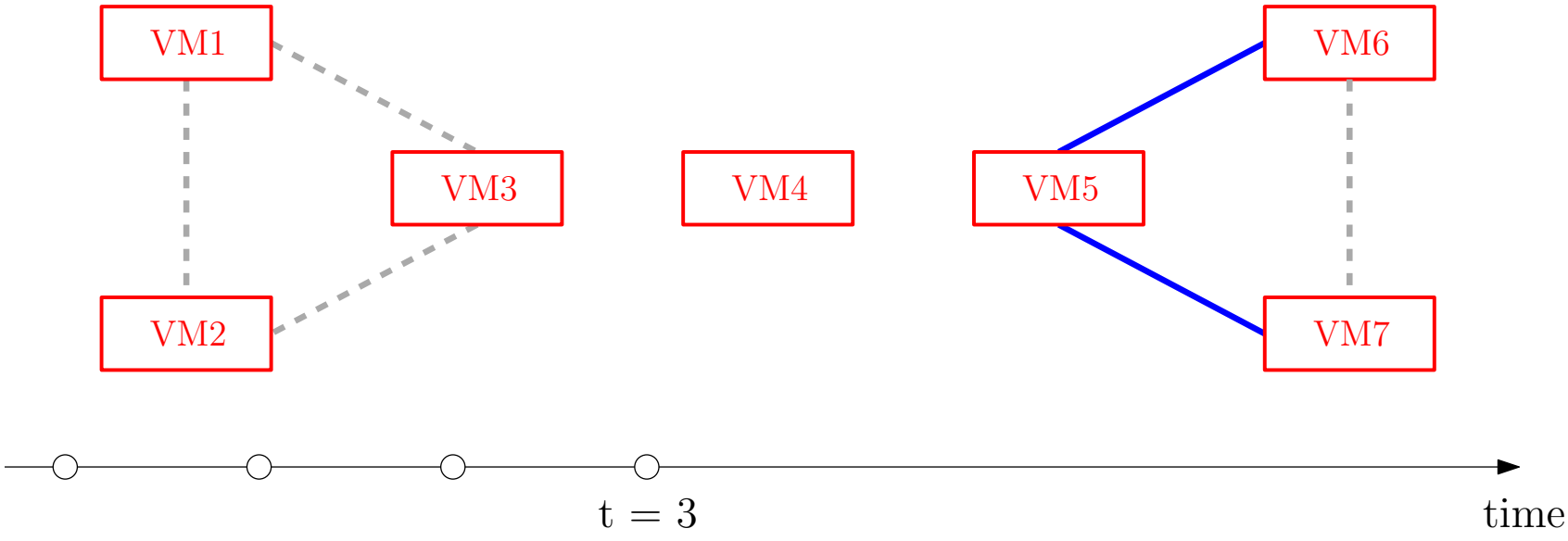
Computational tasks in data centers

During runtime VMs communicate with each other.



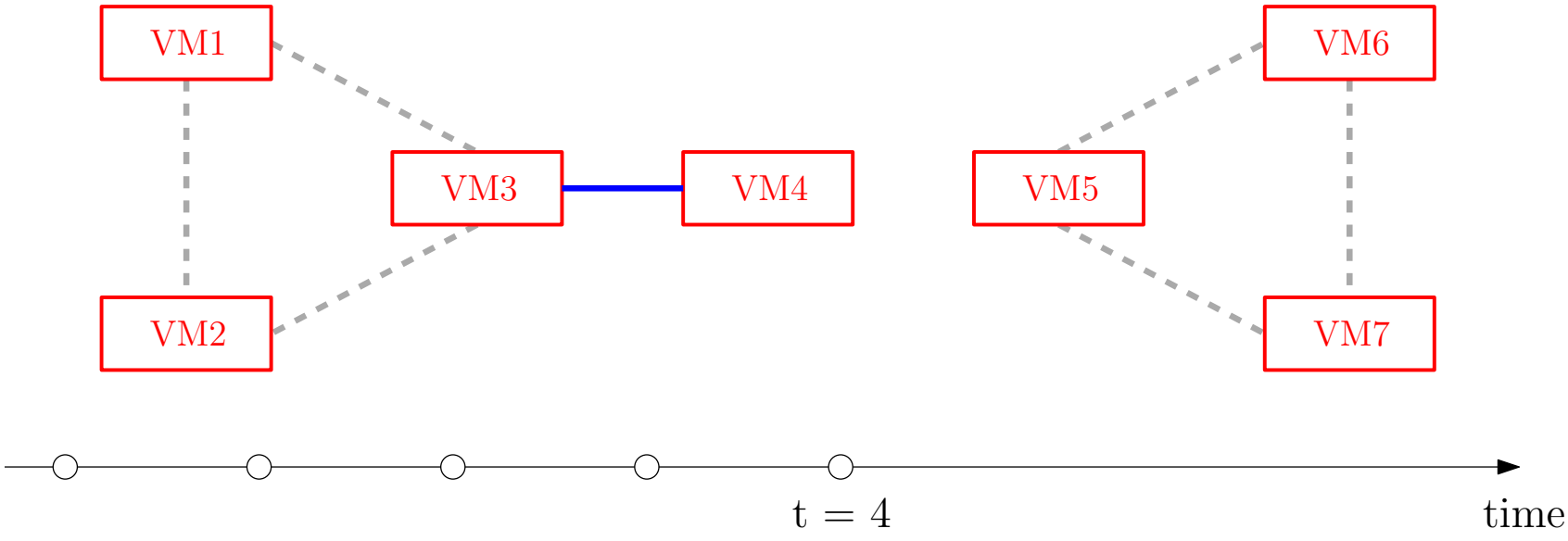
Computational tasks in data centers

During runtime VMs communicate with each other.



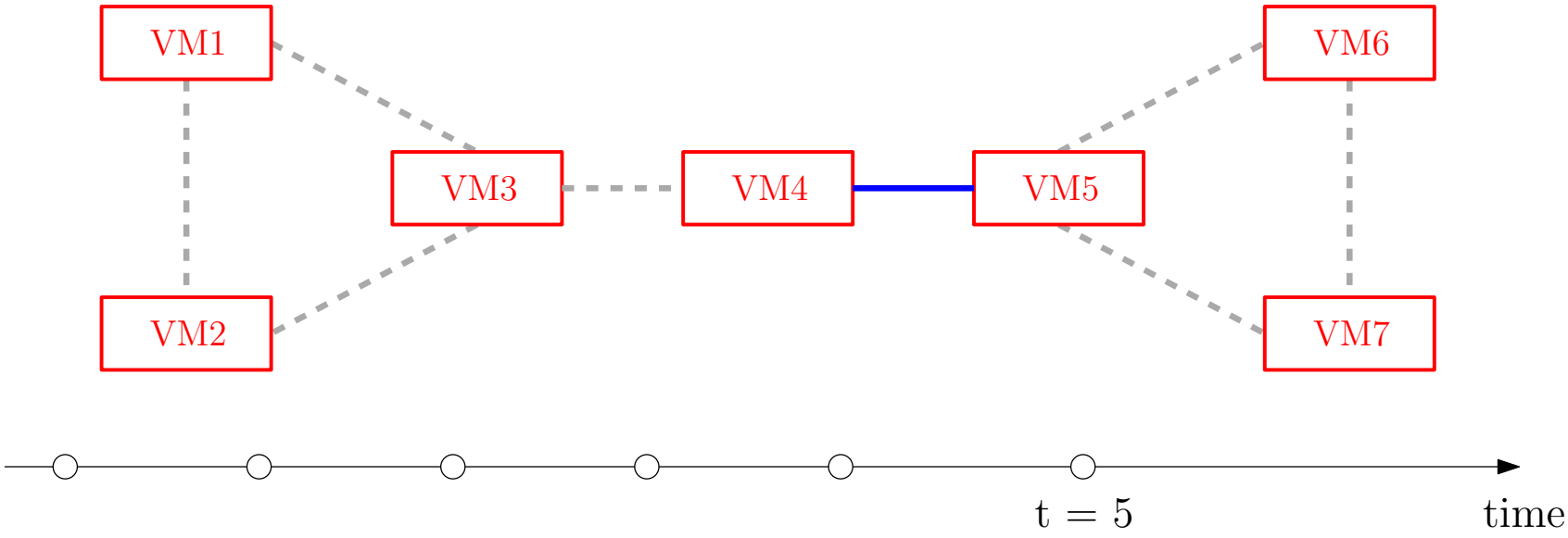
Computational tasks in data centers

During runtime VMs communicate with each other.

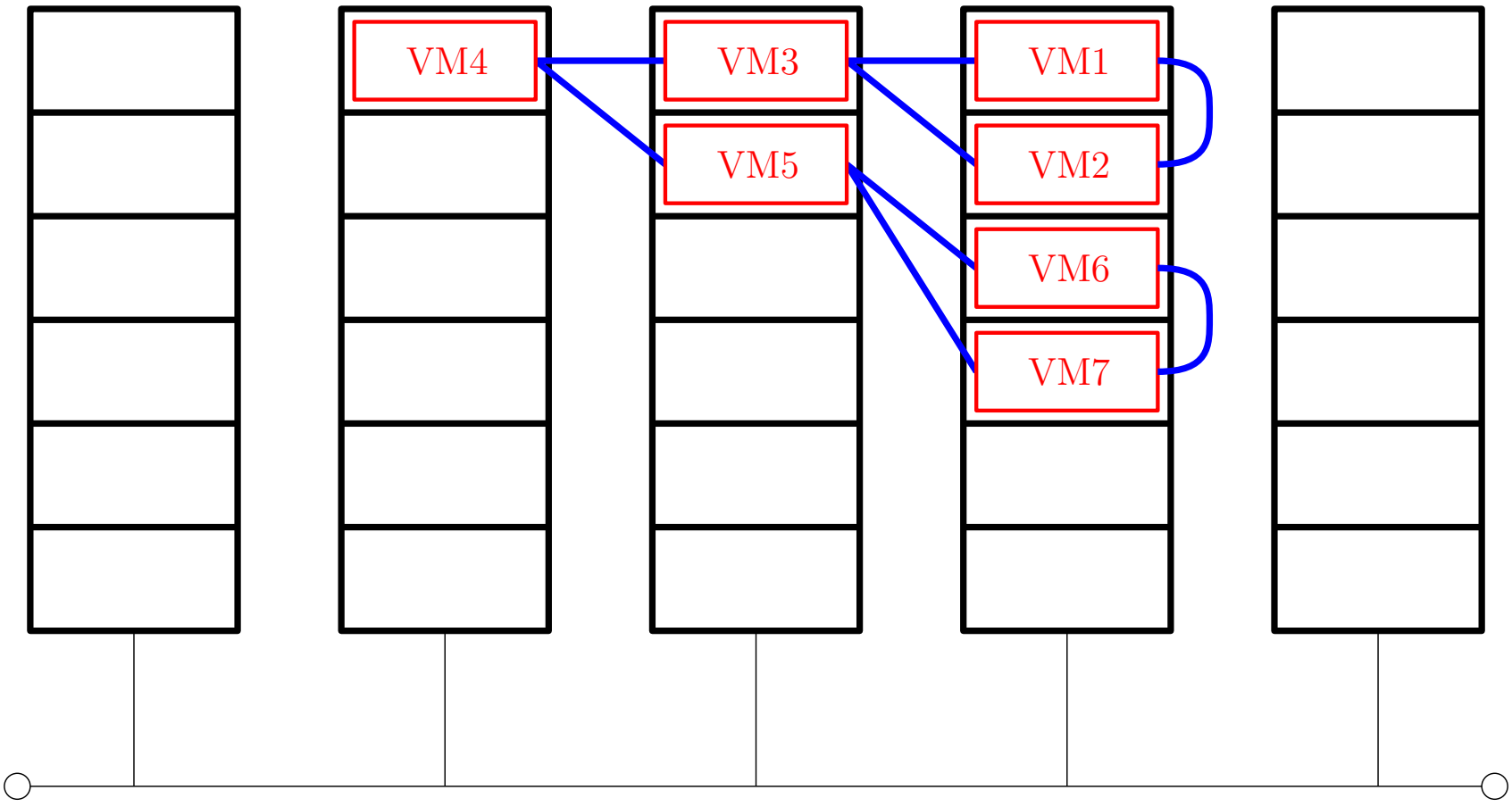


Computational tasks in data centers

During runtime VMs communicate with each other.



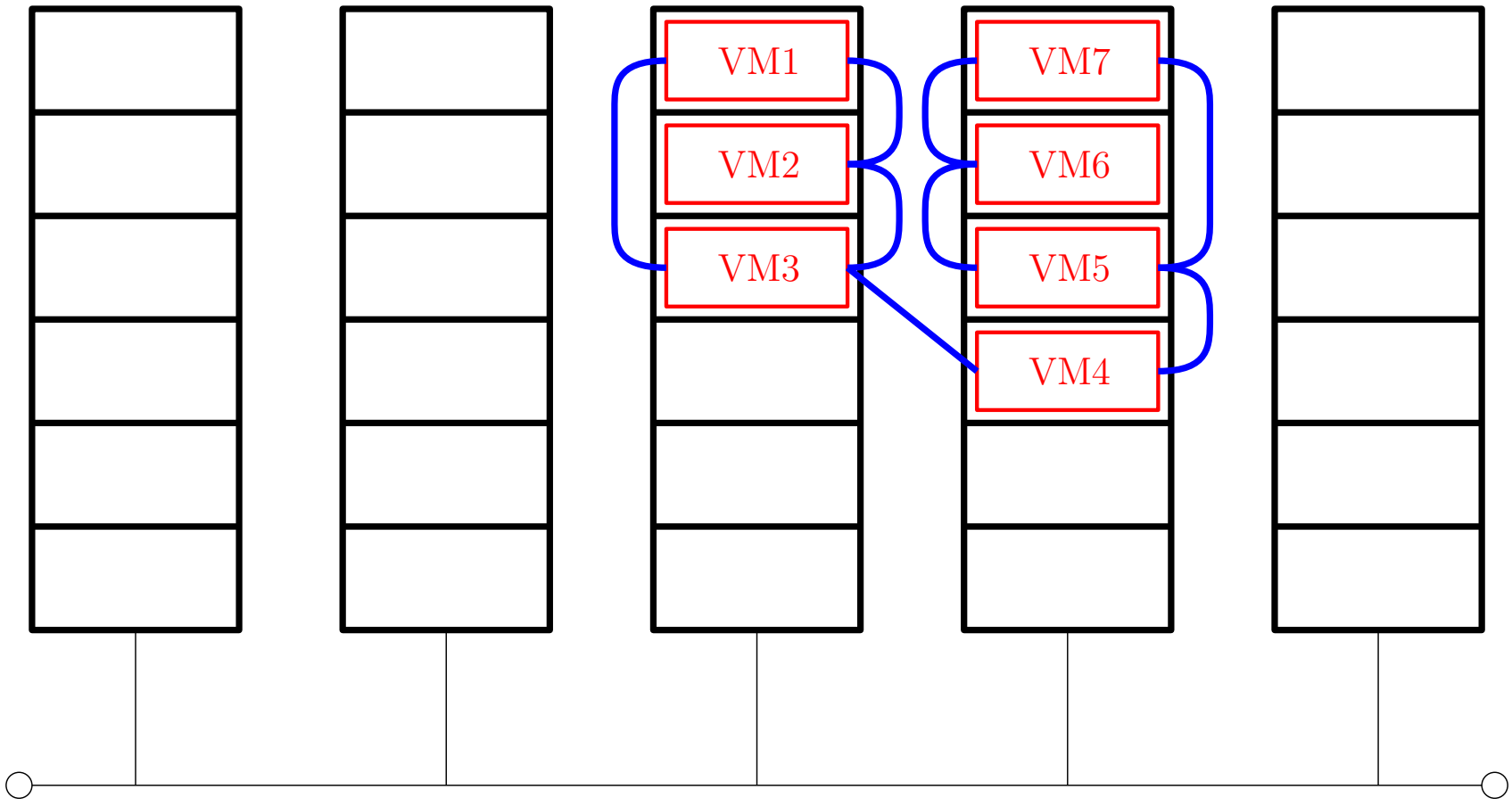
Network-efficient machine placement



Inefficient placement: total communication cost = 6

(internal communication is free)

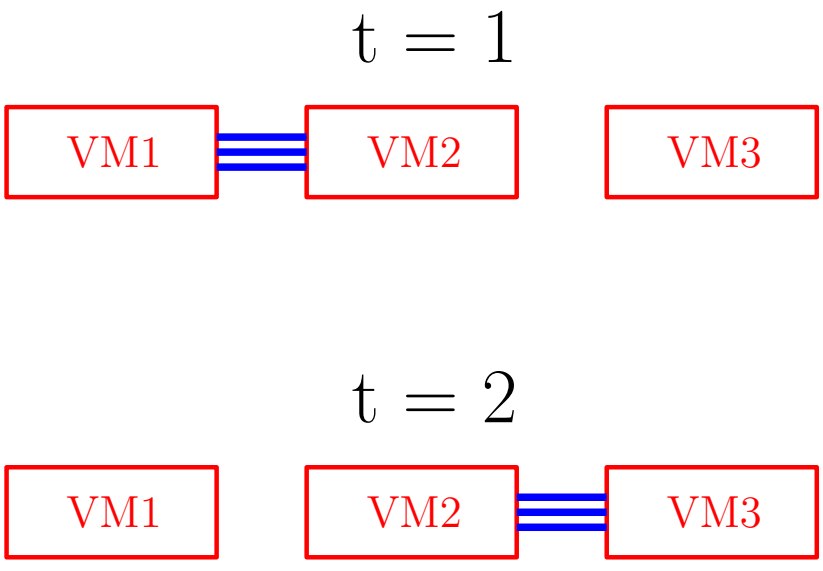
Network-efficient machine placement

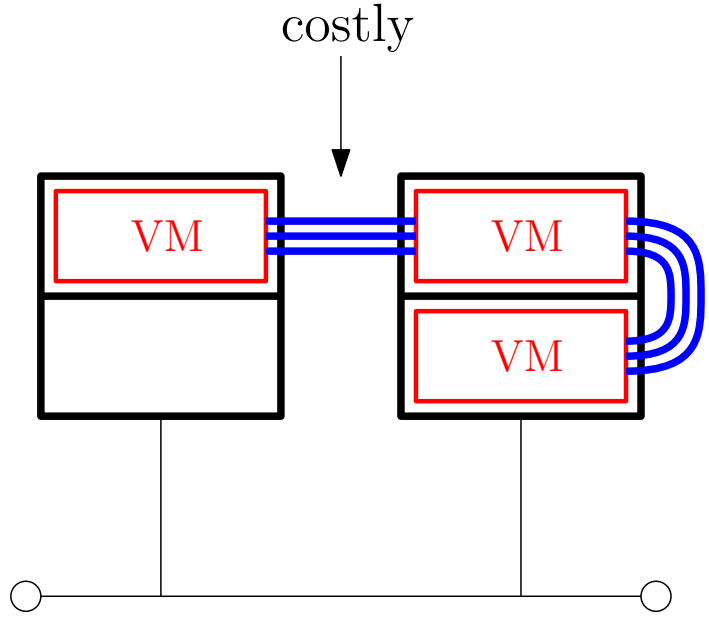


More efficient placement: total communication cost = 1

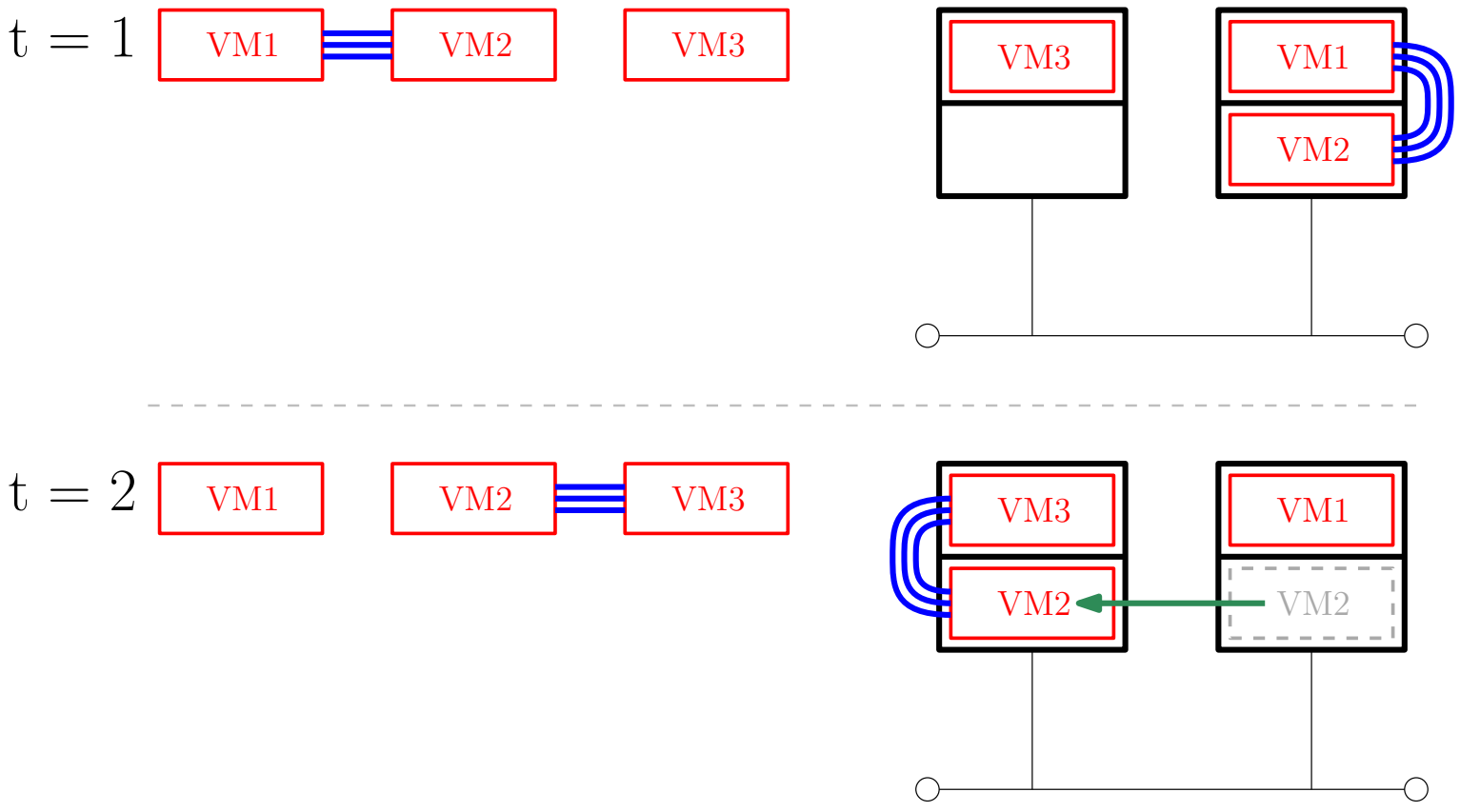
(internal communication is free)

Sometimes static placement is inherently inefficient





Virtual machine (VM) migration



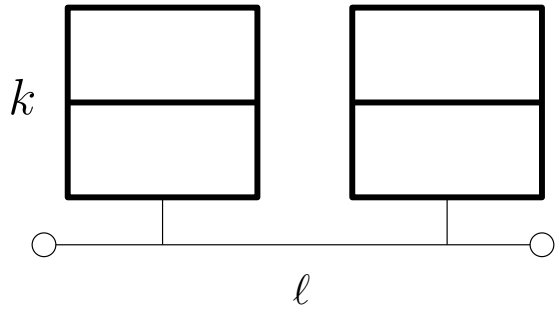
Fixed cost α for migrating a VM to another physical machine.

(migration is supported by major virtualization providers, inc. Xen, Hyper-V, VMware)

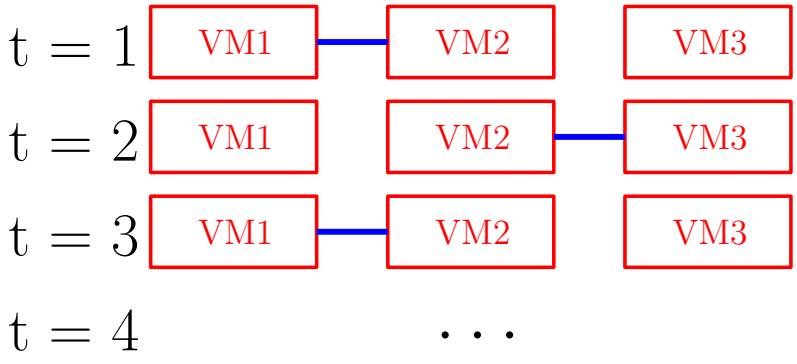
Balanced Re-partition Problem

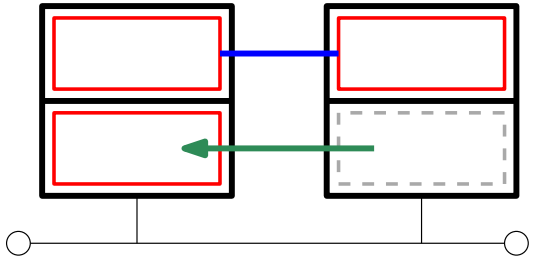
Input

Data center architecture:
ℓ physical machines with capacity k



Communication pattern:





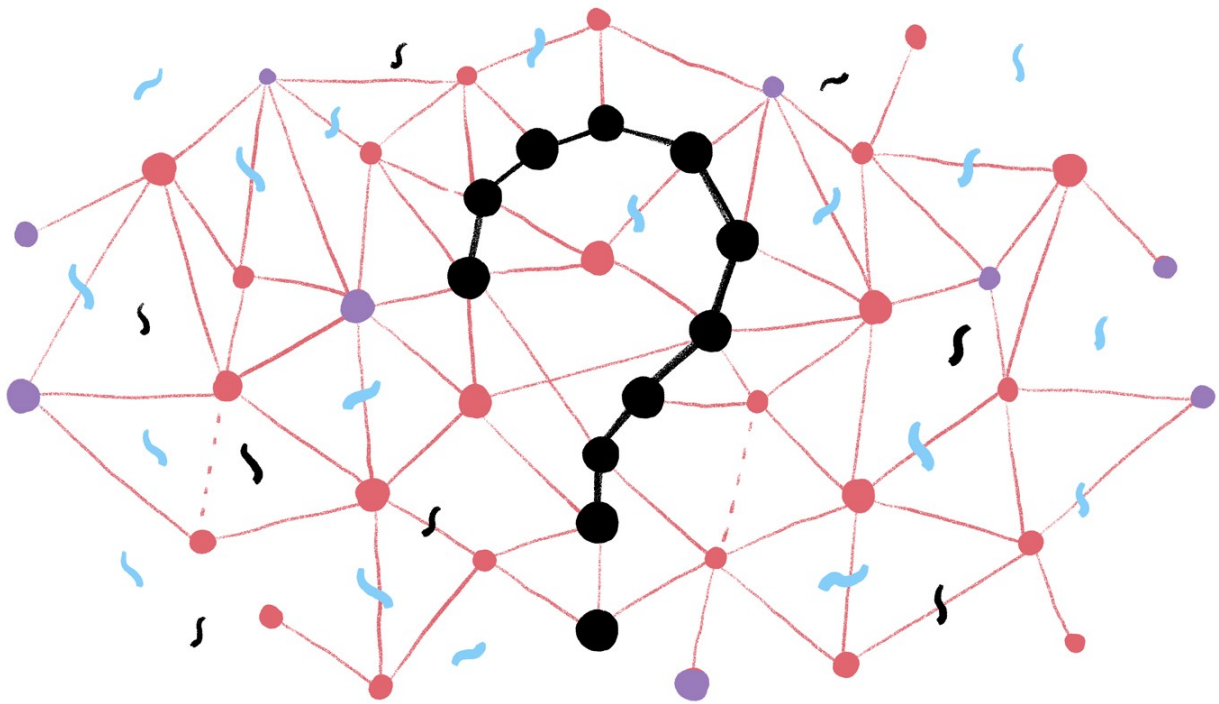
Costs:

External communication costs 1

Migration costs α

Objective: compute the migration schedule that minimizes the total cost of communication and migration

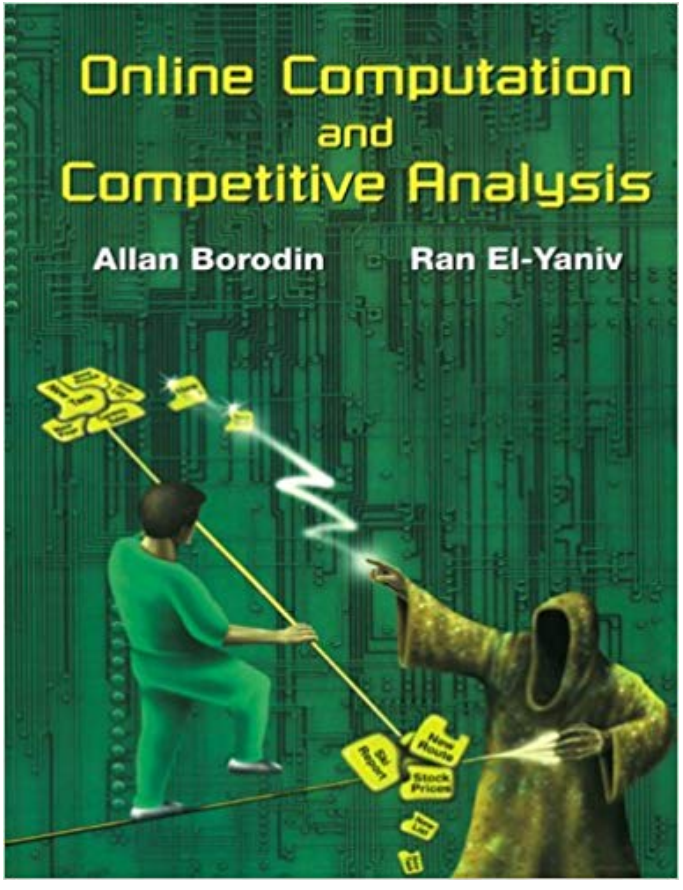
Problem: the communication pattern is unknown!



Communication requests appear on the fly.

Online algorithms and competitive analysis

- Input revealed piece by piece
- Irrevocable decisions
- Comparison to offline optimal algorithm



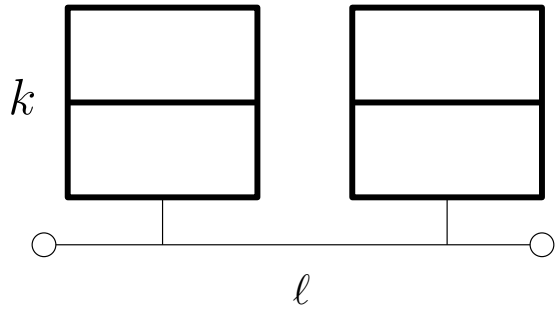
The competitive ratio:

$$ALG \leq c \cdot OPT$$

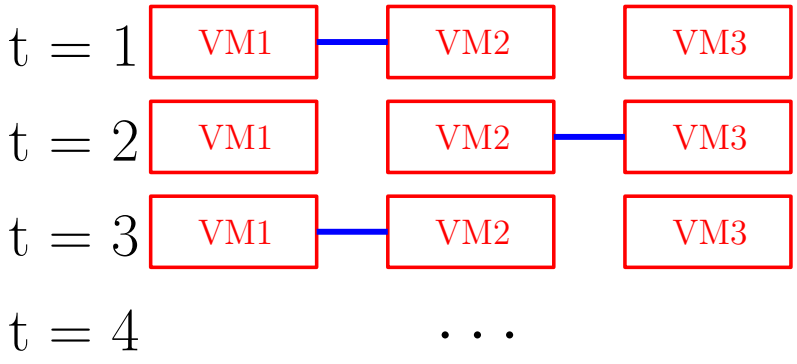
Offline Balanced Re-partition Problem

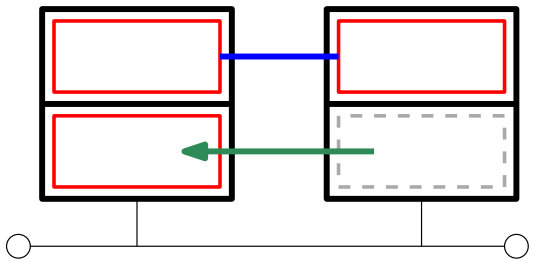
Input

Data center architecture:
ℓ physical machines with capacity k



Communication pattern:





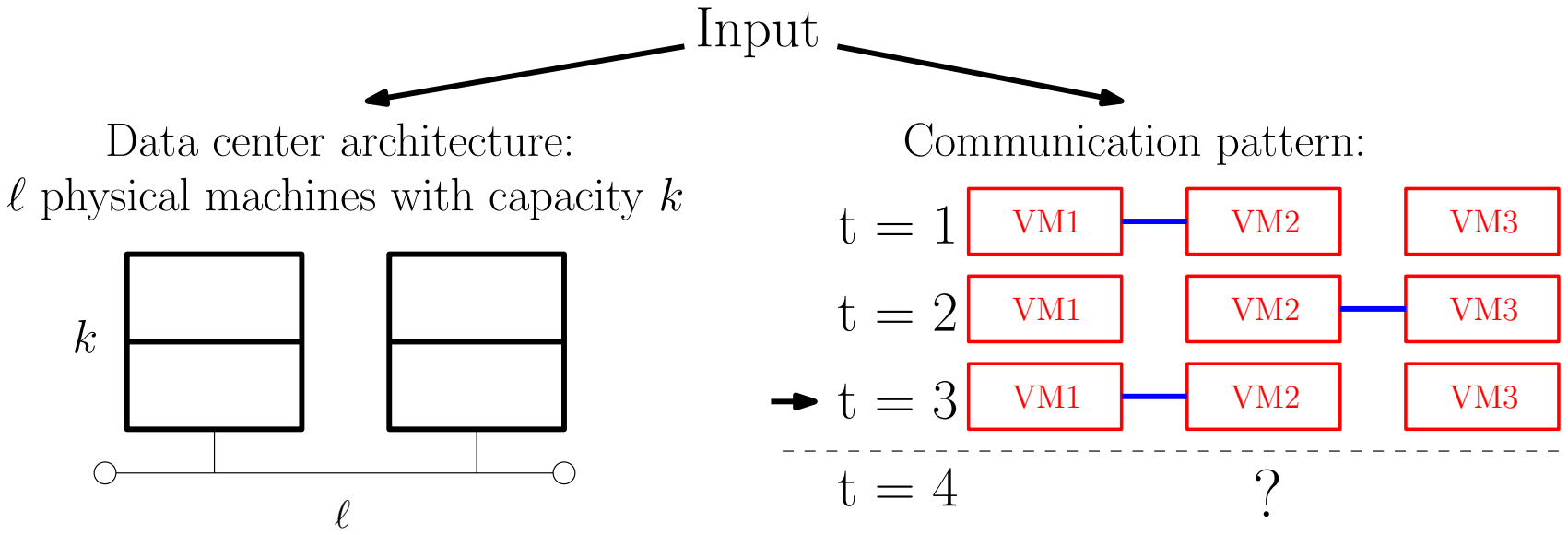
Costs:

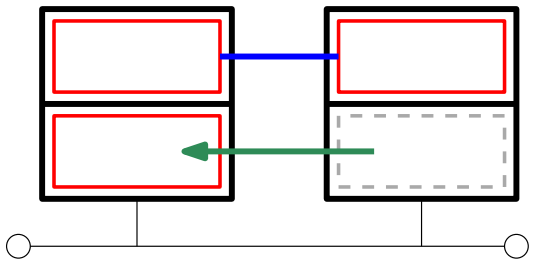
External communication costs 1

Migration costs α

Objective: compute the migration schedule that minimizes the total cost of communication and migration

Online Balanced Re-partition Problem





Costs:

External communication costs 1

Migration costs α

Objective: compute the migration schedule that minimizes the total cost of communication and migration

Results (DISC '16 and SIAM J. Disc. Math '20)

Theorem

No deterministic algorithm obtains competitive ratio better than k. (where k is the physical machine capacity)

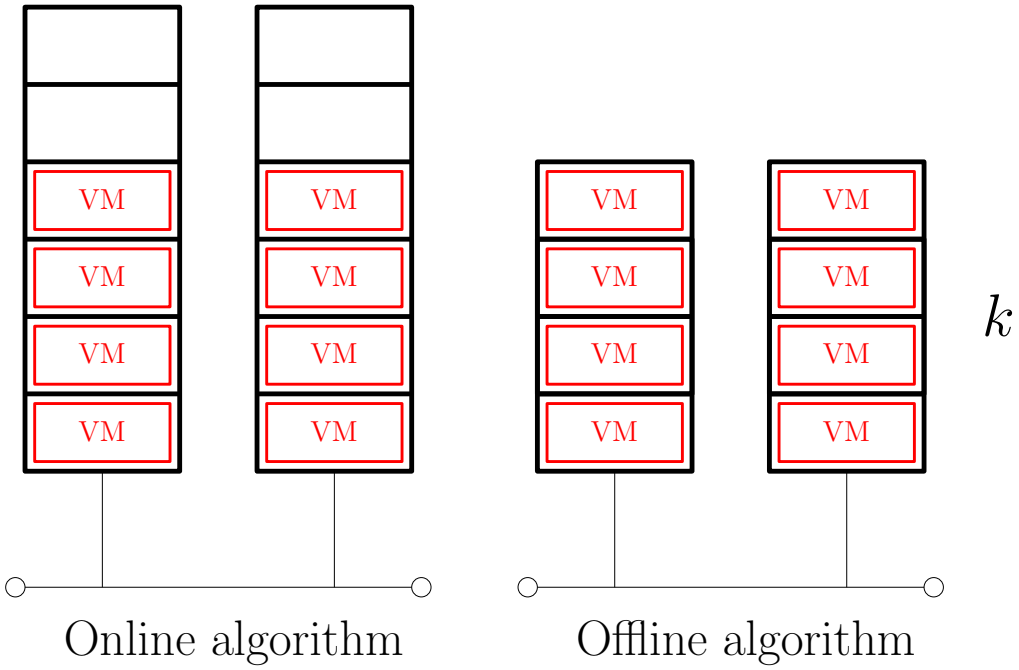
Reduction from the caching problem.

Caching and resource augmentation

- Caching is k-competitive for cache size $= k$
  (tight result of Sleator and Tarjan '85)
- Caching gets substantially easier when we have more resources than the offline algorithm

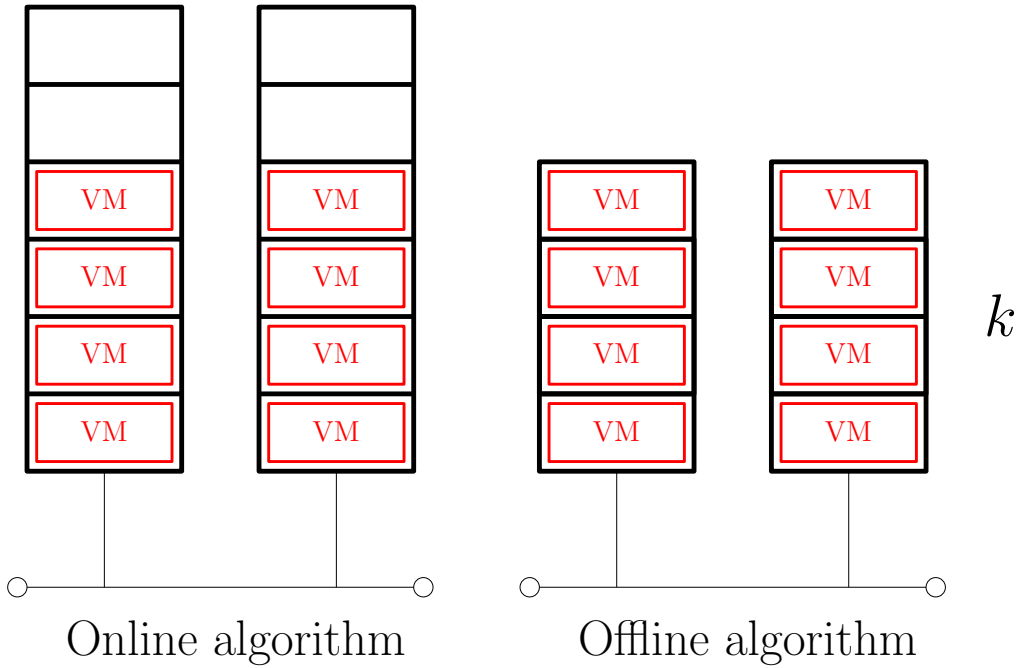
  OPT cache size $= k$,
  ALG cache size $= k_{ALG} \geq k$,
  then ALG is $\frac{k_{ALG}}{k_{ALG}-k+1}$-competitive.

Resource augmentation in Online Balanced Re-partition



(number of physical machines is the same)

Resource augmentation in Online Balanced Re-partition



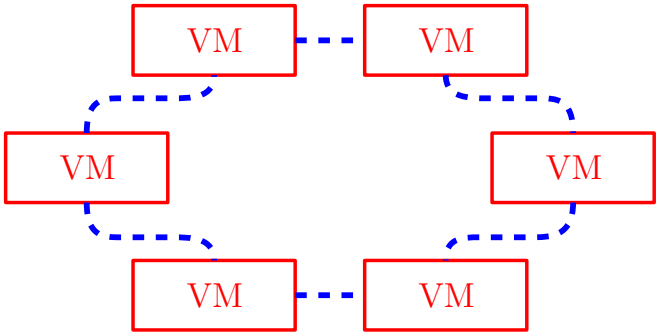
(number of physical machines is the same)

Theorem

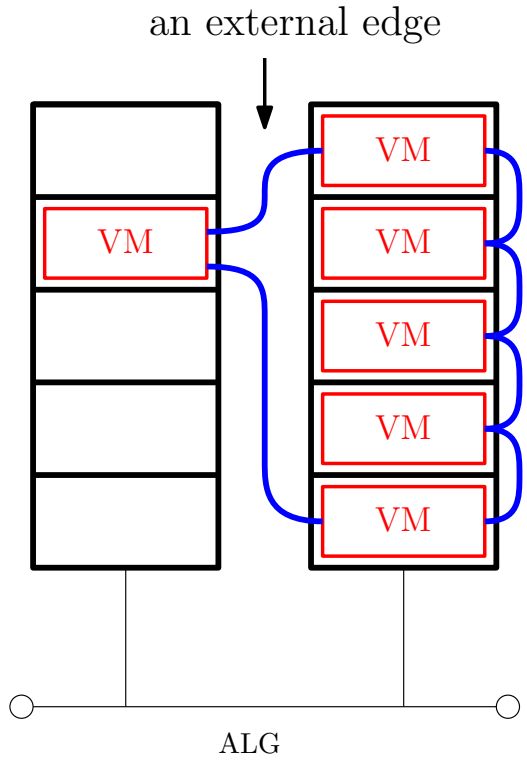
No resource-augmented deterministic algorithm obtains competitive ratio better than k.
(unless every virtual machine fits into one cluster; then trivial)

Lower bound of k with resource augmentation

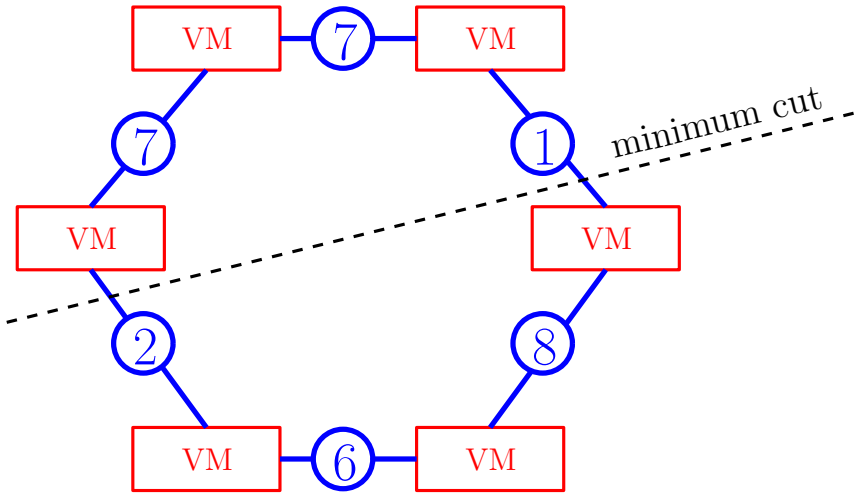
Fix a cycle.
Input: requesting *external* edge from the cycle

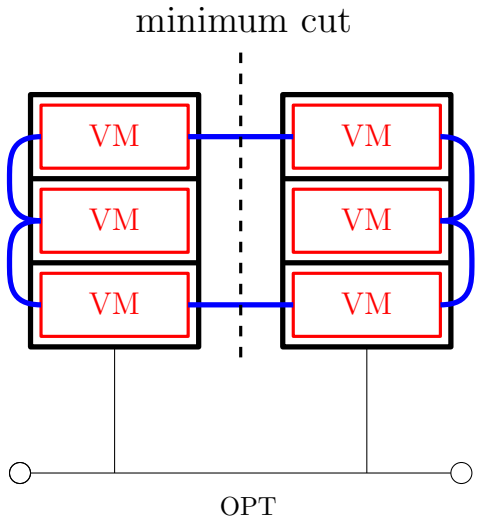


ALG pays for each such request!



Lower bound of k with resource augmentation

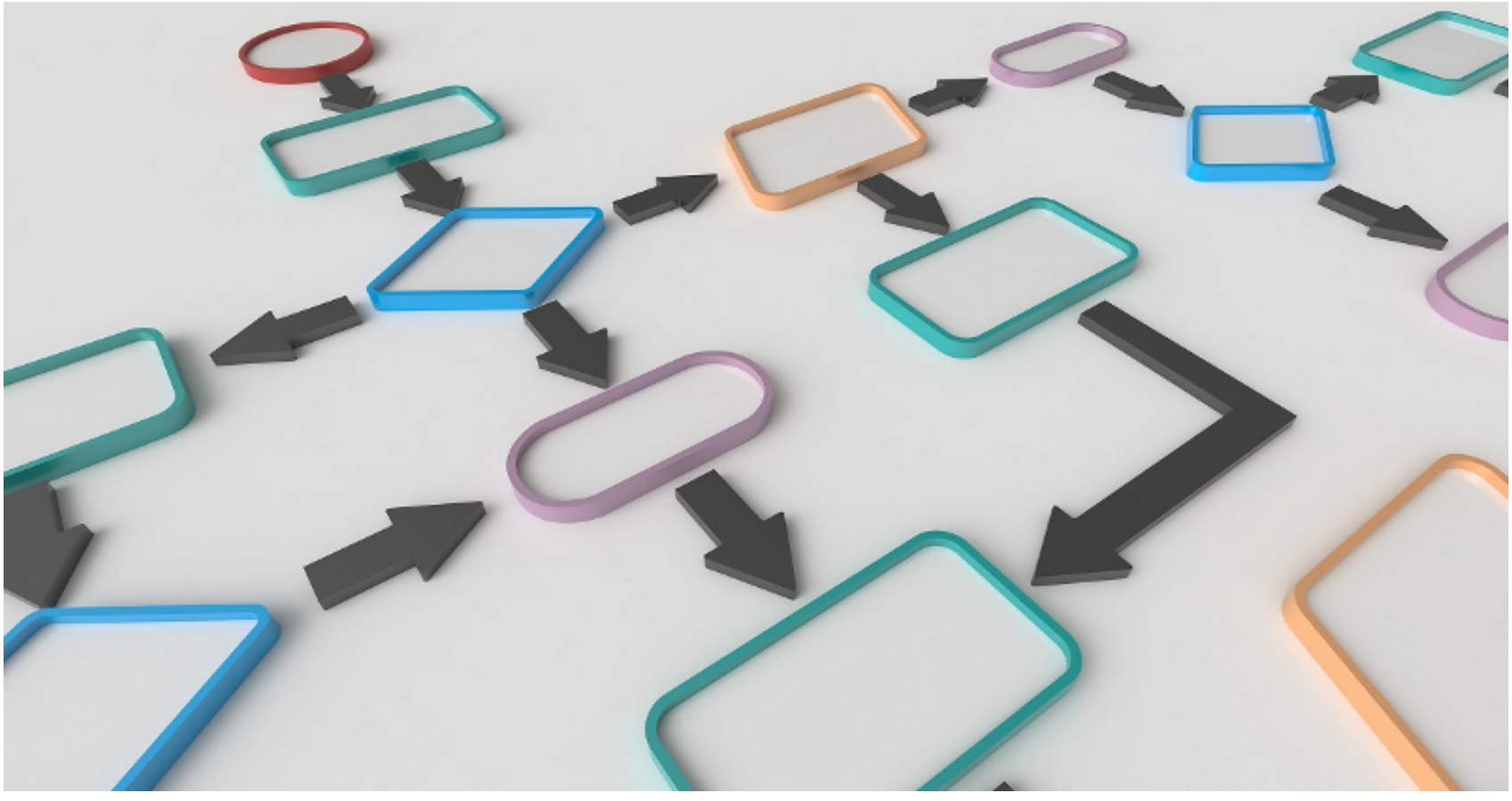




$\mathrm{ALG} = 1 + 8 + 6 + 2 + 7 + 7 = 31$

$\mathrm{OPT} = 2 + 1 \leq \frac{1}{k} \cdot \mathrm{ALG}$

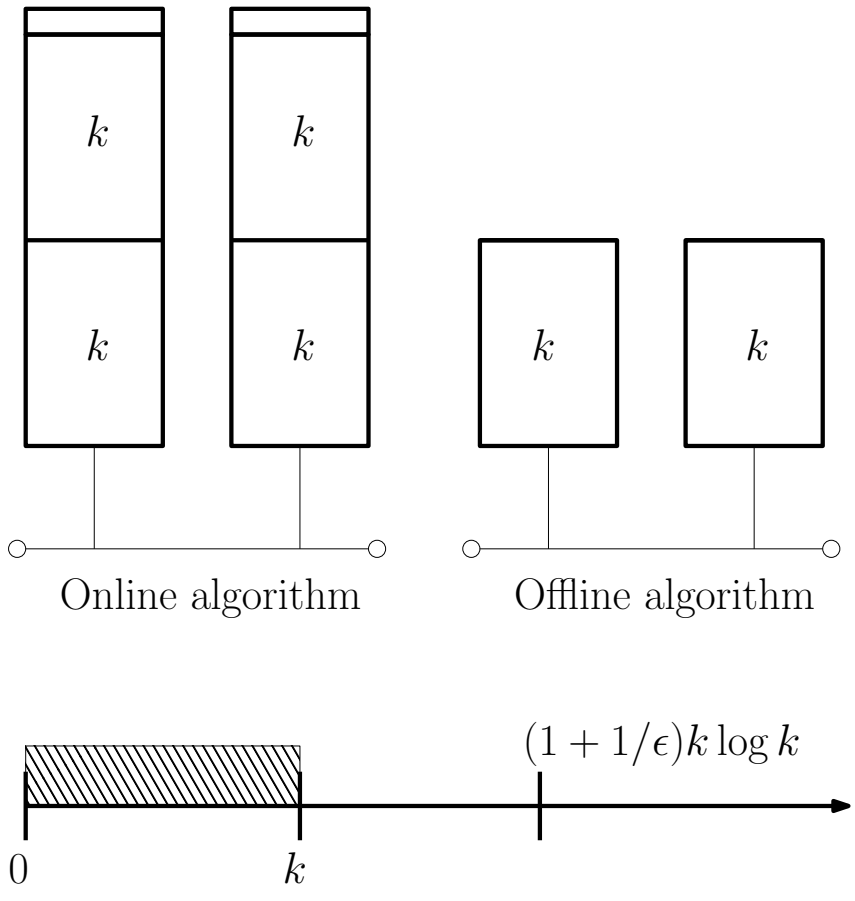
Algorithm for the scenario with resource augmentation



Algorithm for the scenario with resource augmentation

Theorem

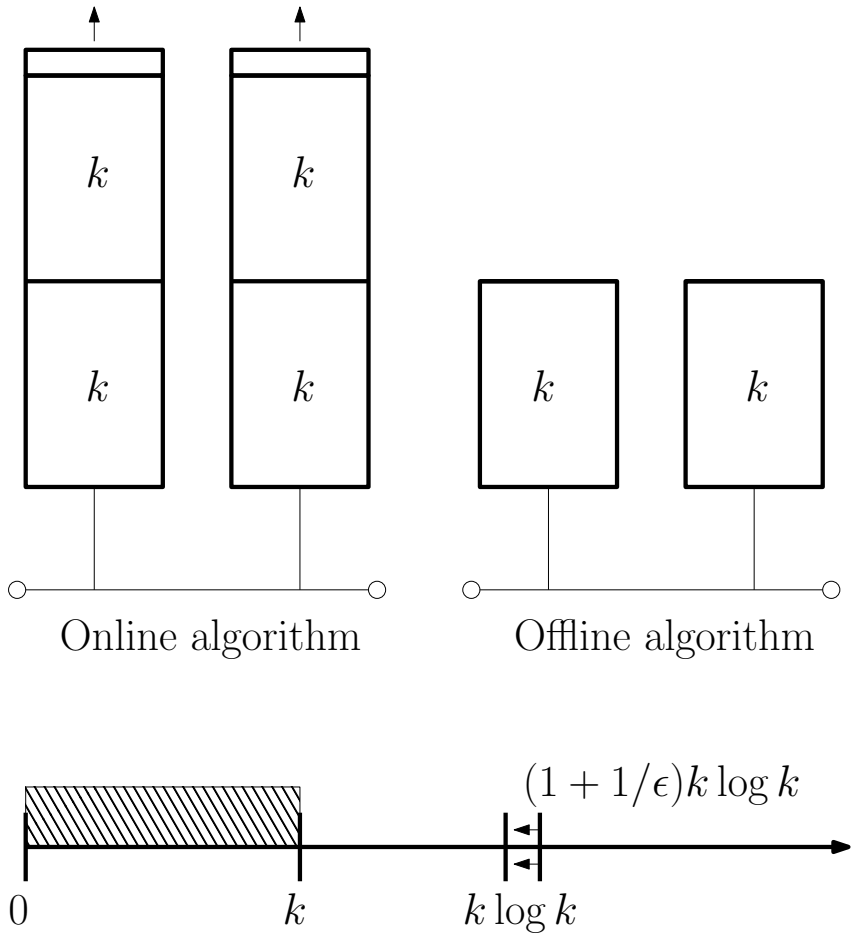
For the scenario with resource augmentation factor of $2 + \epsilon$ there exists a $O((1 + 1/\epsilon) \cdot k \log k)$-competitive algorithm.



Algorithm for the scenario with resource augmentation

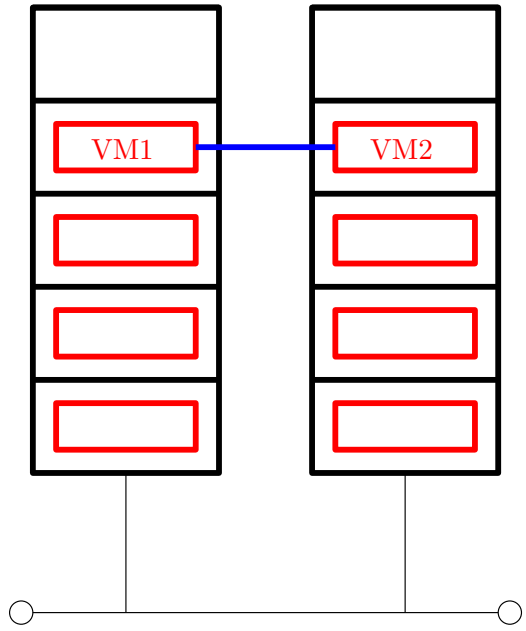
Theorem

For the scenario with resource augmentation factor of $2 + \epsilon$ there exists a $O((1 + 1/\epsilon) \cdot k \log k)$-competitive algorithm.



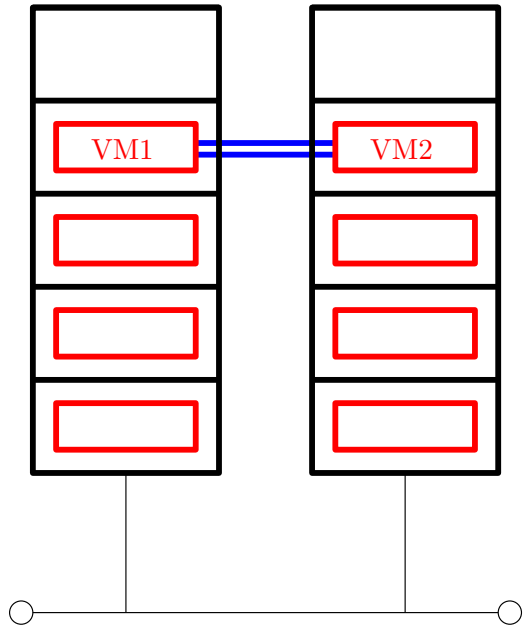
Our algorithm for 2 physical machines

$\text{cost} = 1$



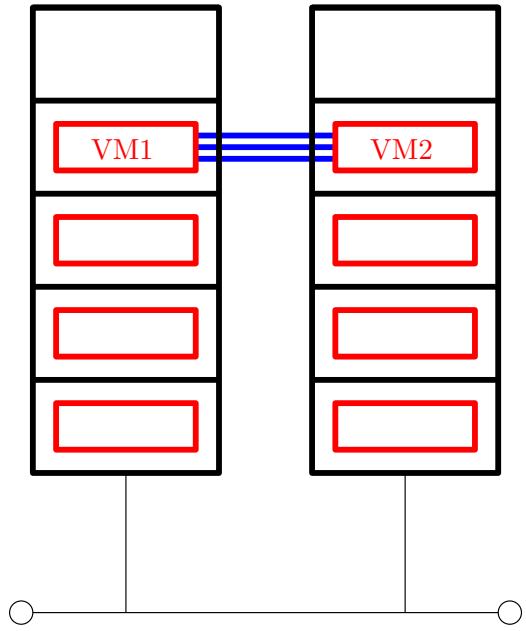
Our algorithm for 2 physical machines

$\text{cost} = 2$



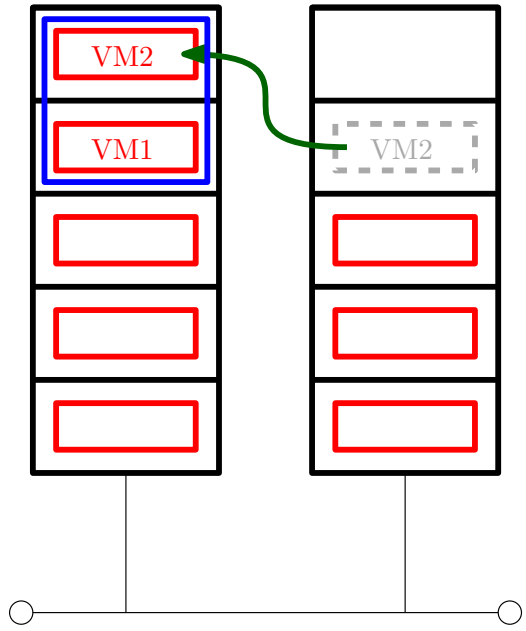
Our algorithm for 2 physical machines

$$\text{cost} = 2 \cdot \alpha$$



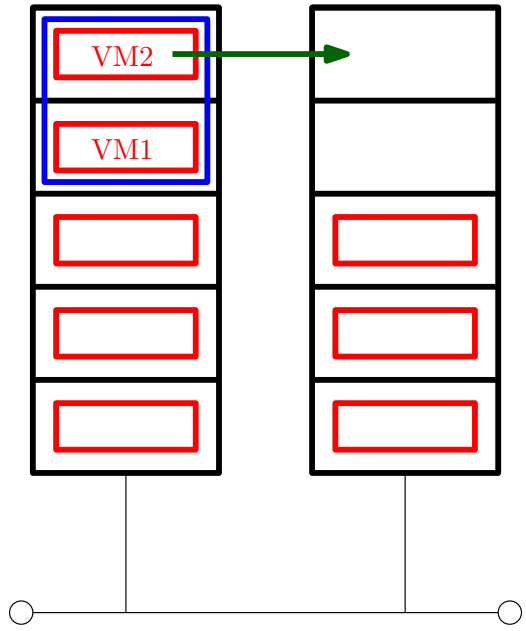
Our algorithm for 2 physical machines

Invariant: components stored always in a single PM



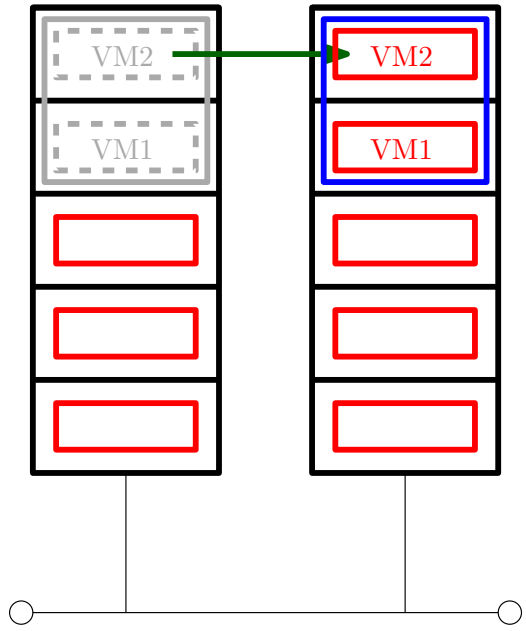
Our algorithm for 2 physical machines

Invariant: components stored always in a single PM

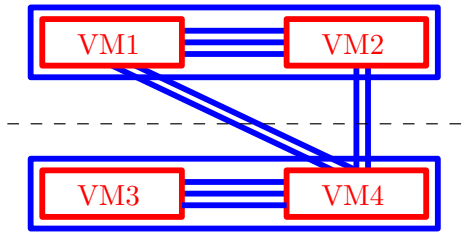


Our algorithm for 2 physical machines

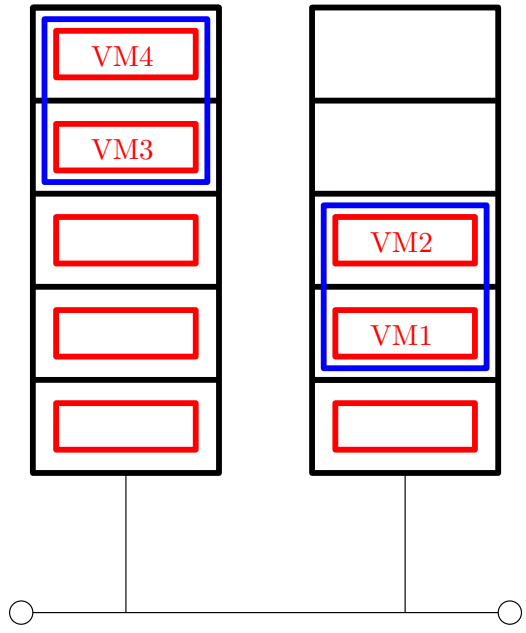
Invariant: components stored always in a single PM



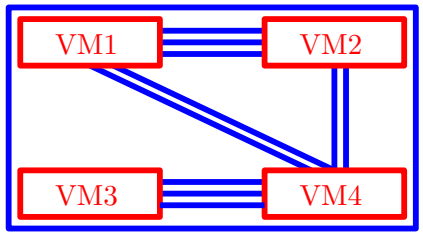
Our algorithm for 2 physical machines

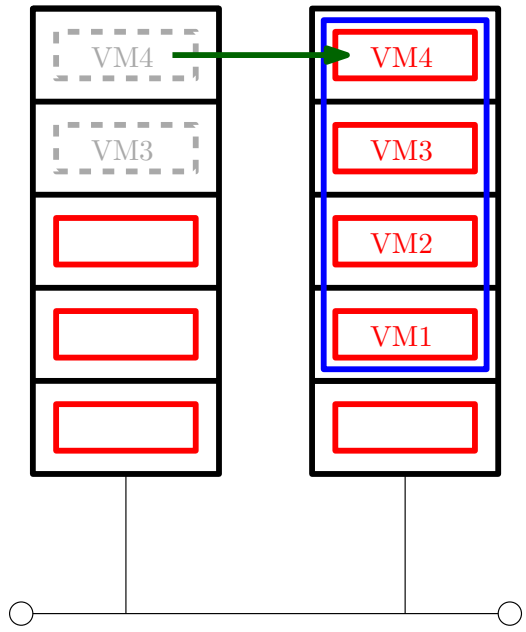


$\text{cost} = 4 \cdot \alpha$

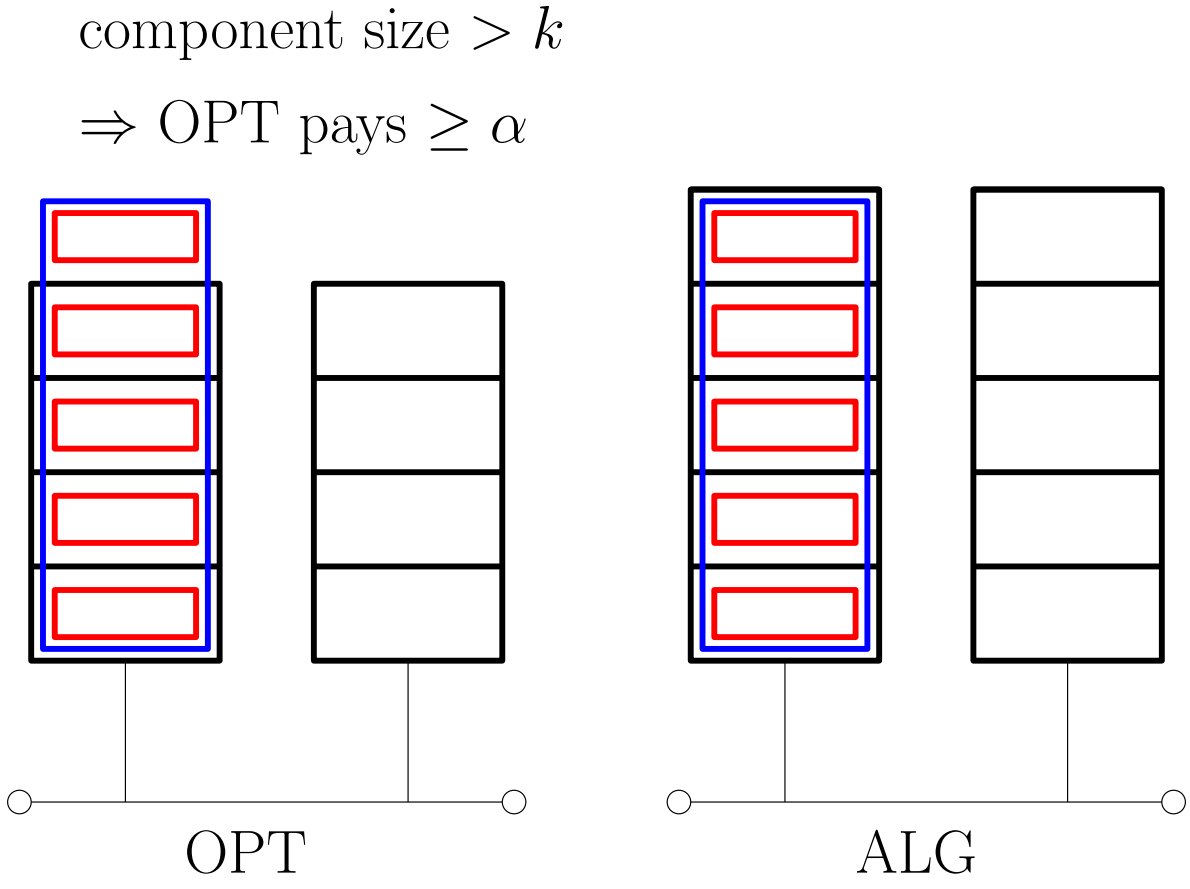


Our algorithm for 2 physical machines





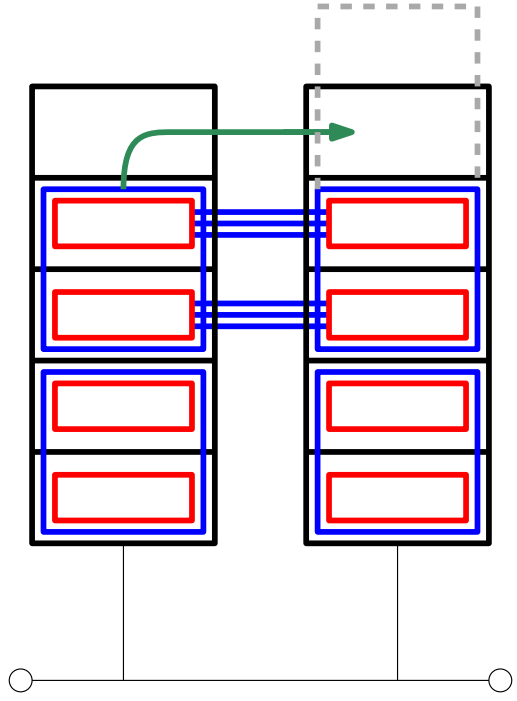
Our algorithm for 2 physical machines



When merging we migrate smaller component to larger
$\Rightarrow$ total cost of ALG $\leq k \log k \cdot \alpha$

Our algorithm for 2 physical machines

Sometimes merging components not immediately possible



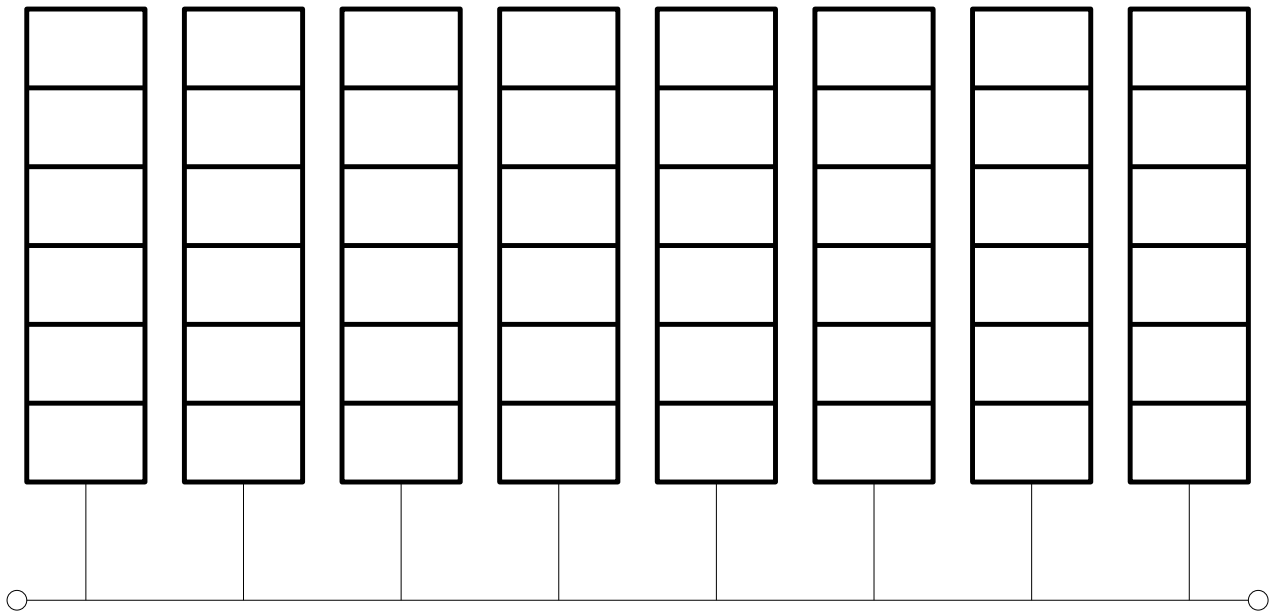
Then perform a global re-balancing

- rare because of resource augmentation
- introduces factor of $(1 + 1/\epsilon)$

Actual analysis more complicated

For general number of physical machines ℓ:

- epochs per component
- more complicated computation of OPT's cost
- re-balancing is more tricky



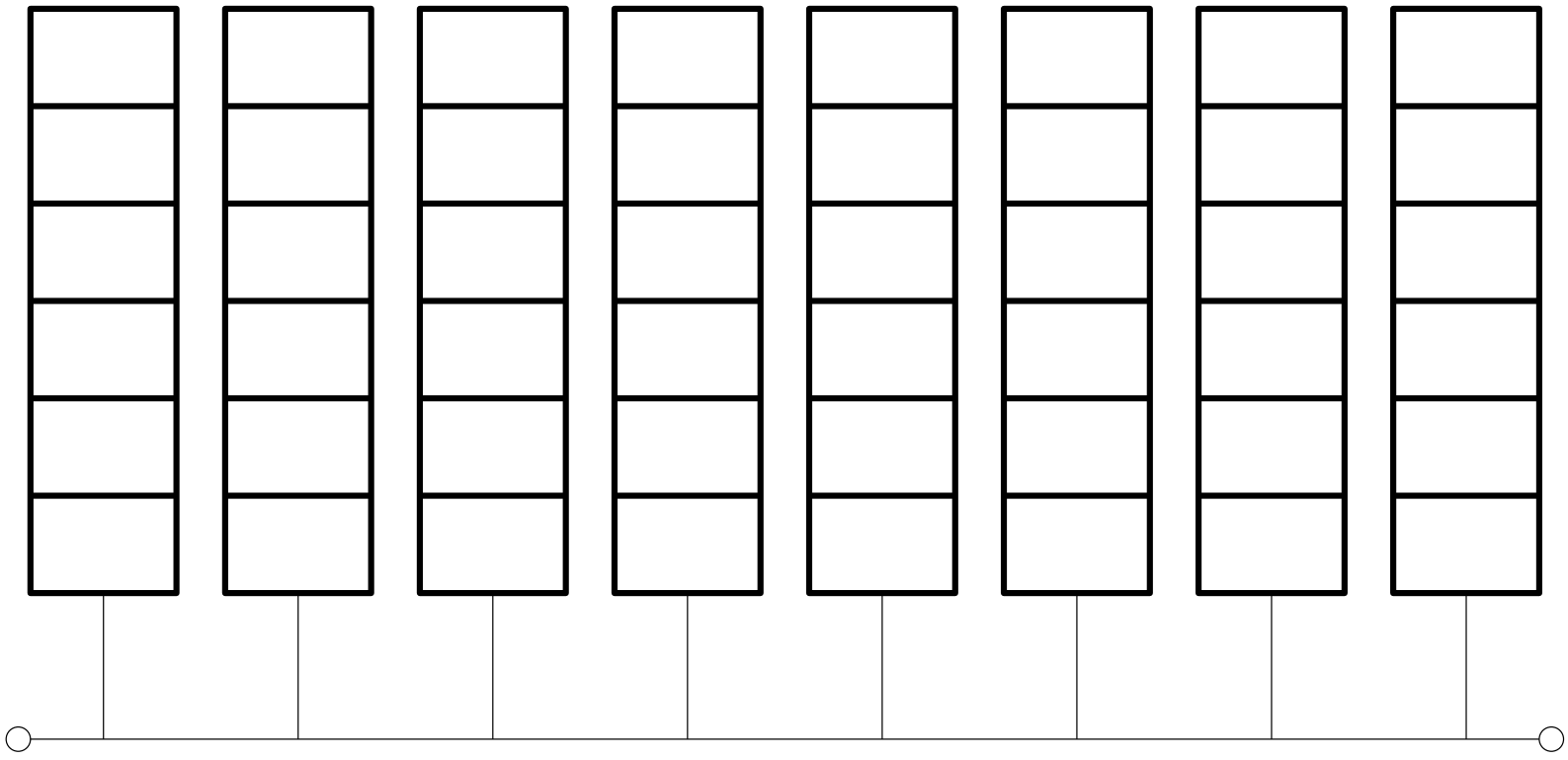
$\ell > 2$

Thesis overview

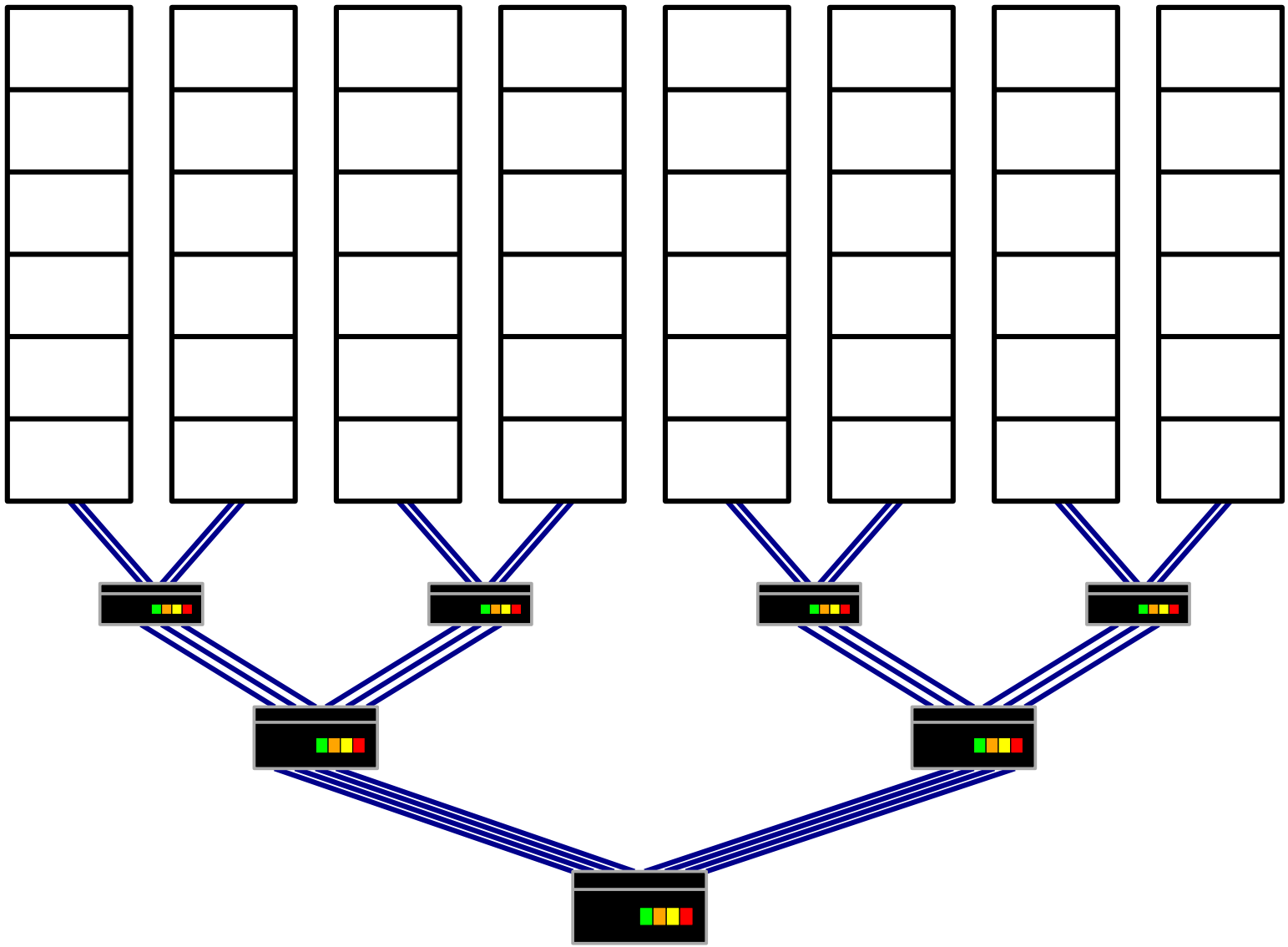
Algorithmic aspects of contemporary networks

- **Part 1: Utilizing redundant storage in MapReduce applications**
  (ICNP '15 + TCS '17)
- Part 2: Assigning tasks to machines in data centers
  (DISC '17 + SIAM J. Disc. Math '20)
- Part 3: Caching forwarding tables in routers
  (ACM SPAA '17)

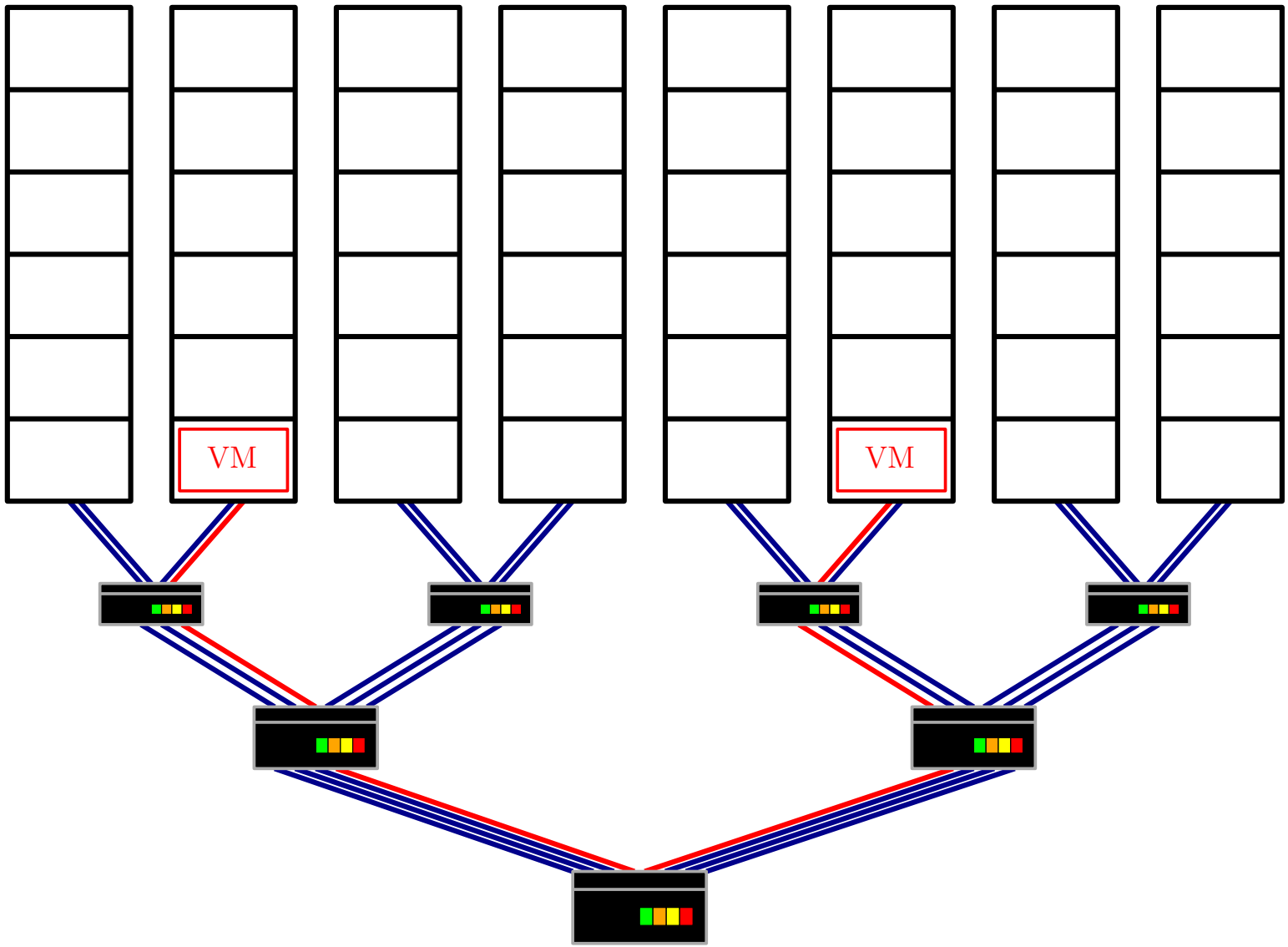
Static placement of virtual machines



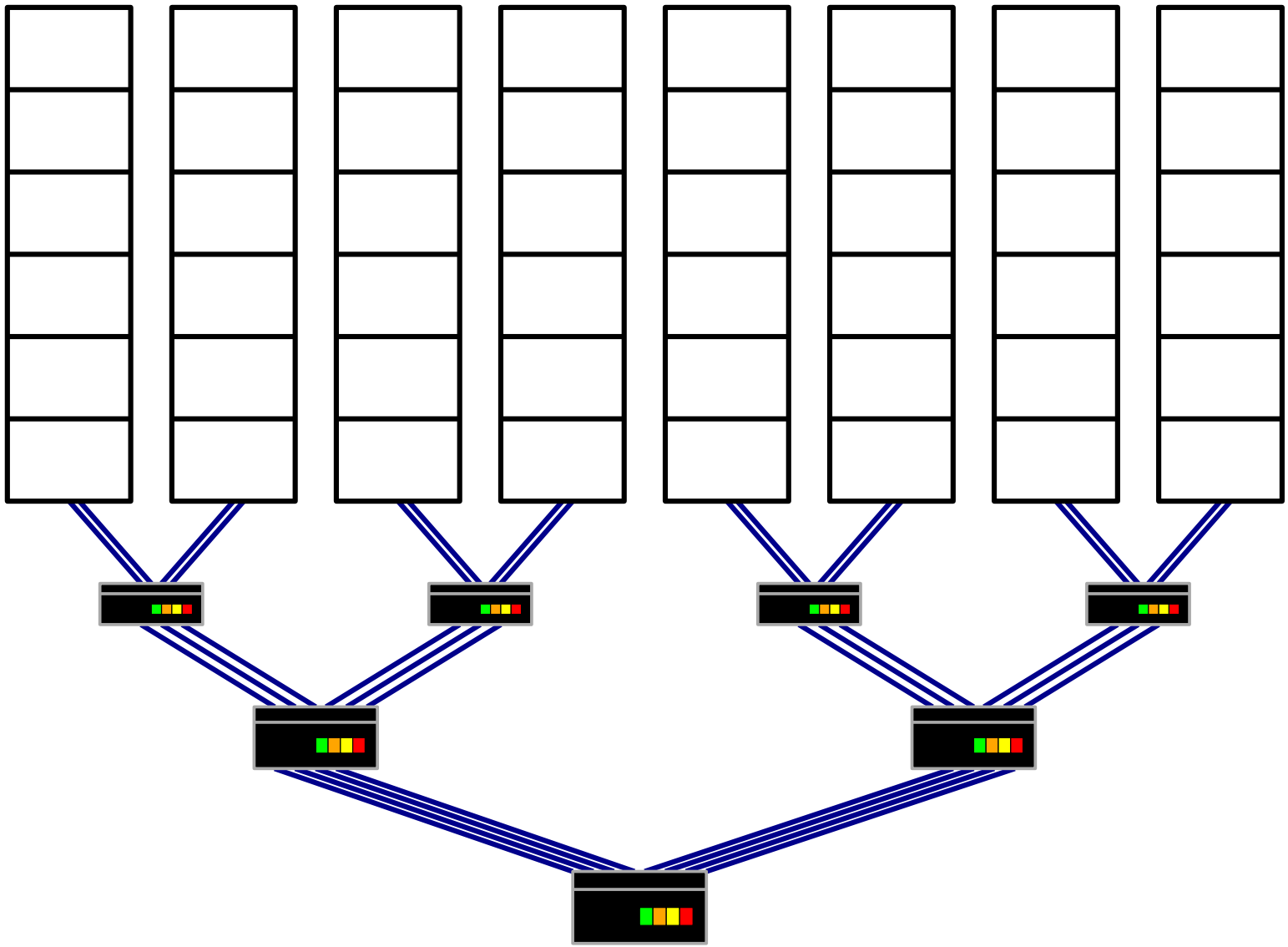
Static placement of virtual machines



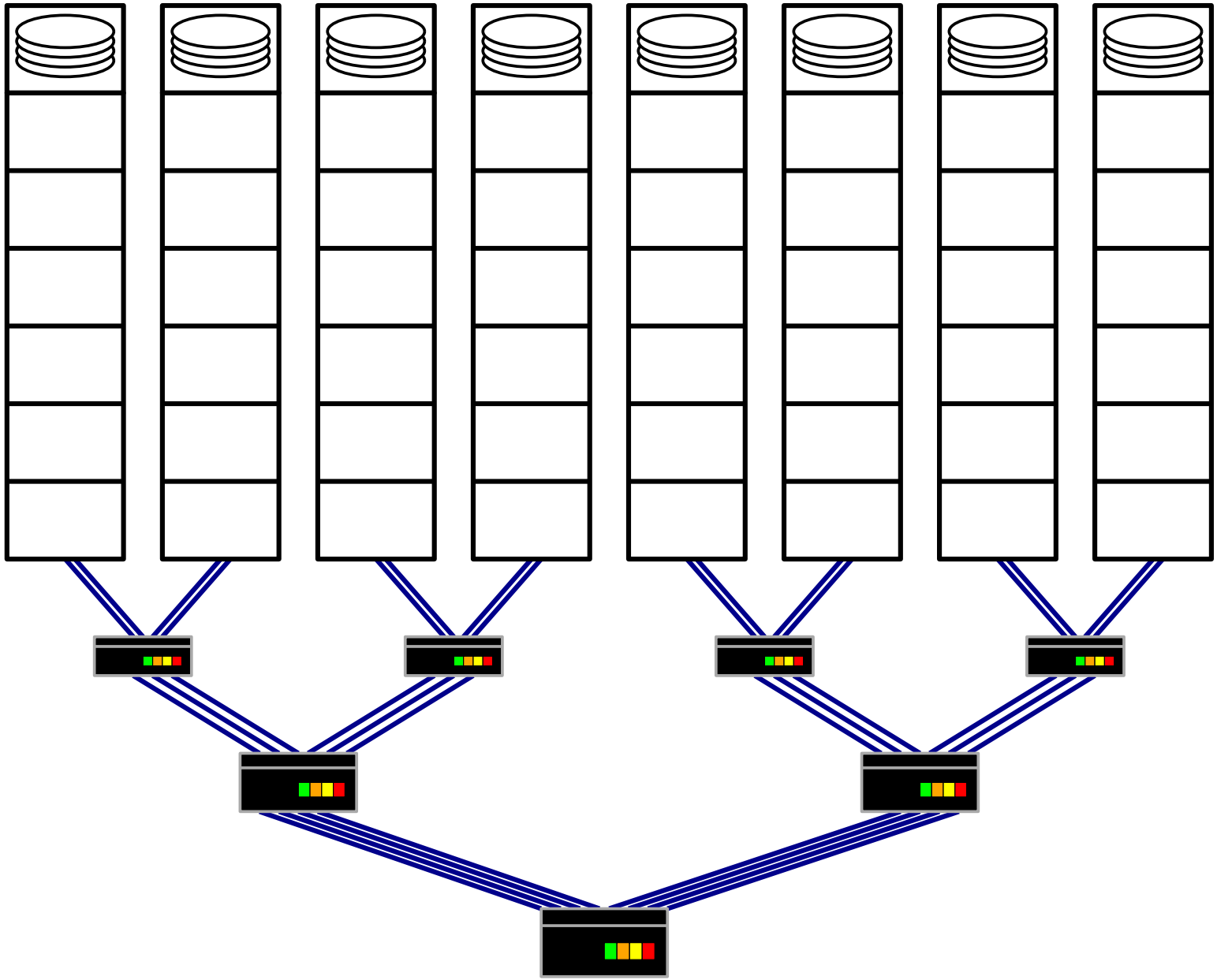
Static placement of virtual machines



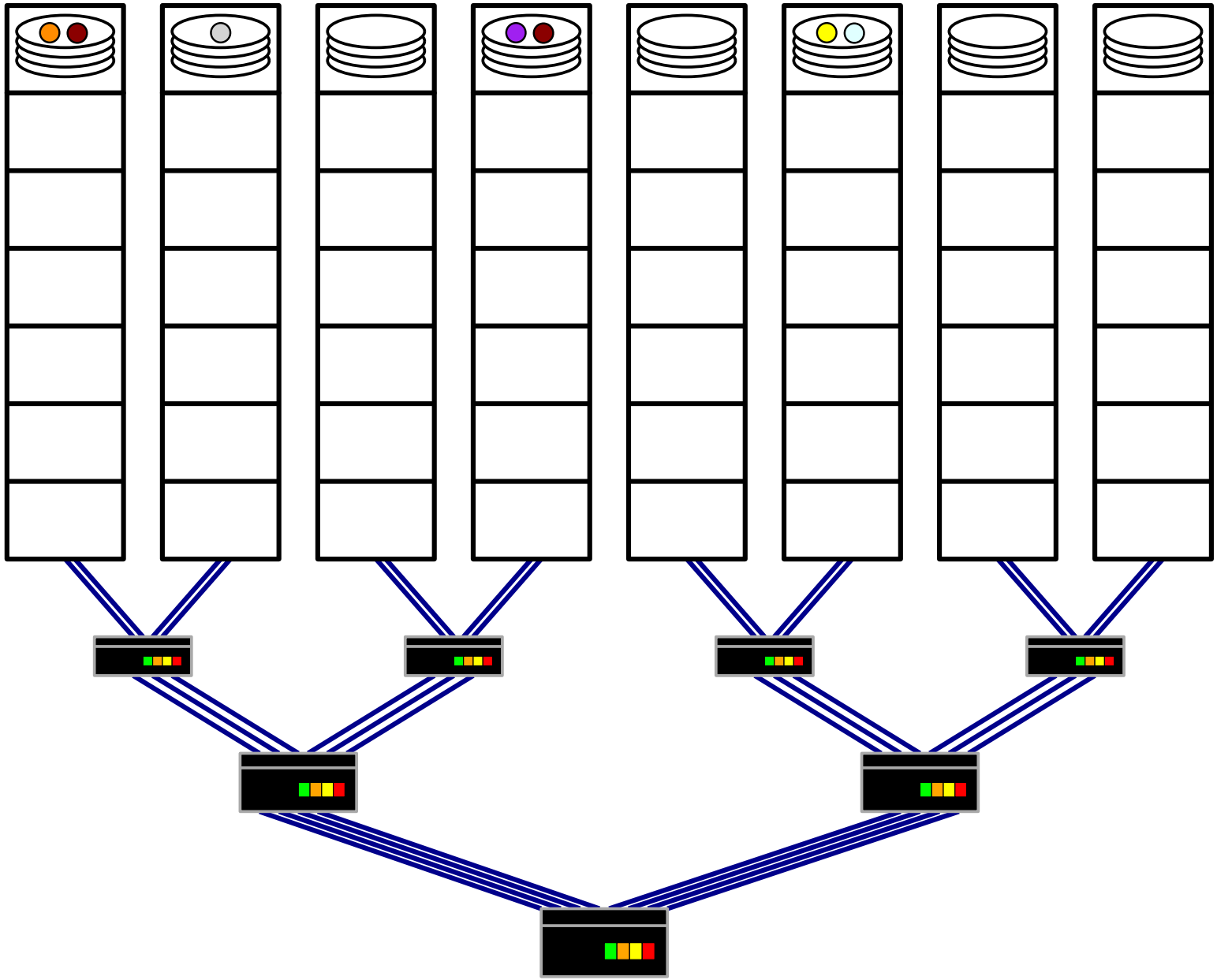
Static placement of virtual machines



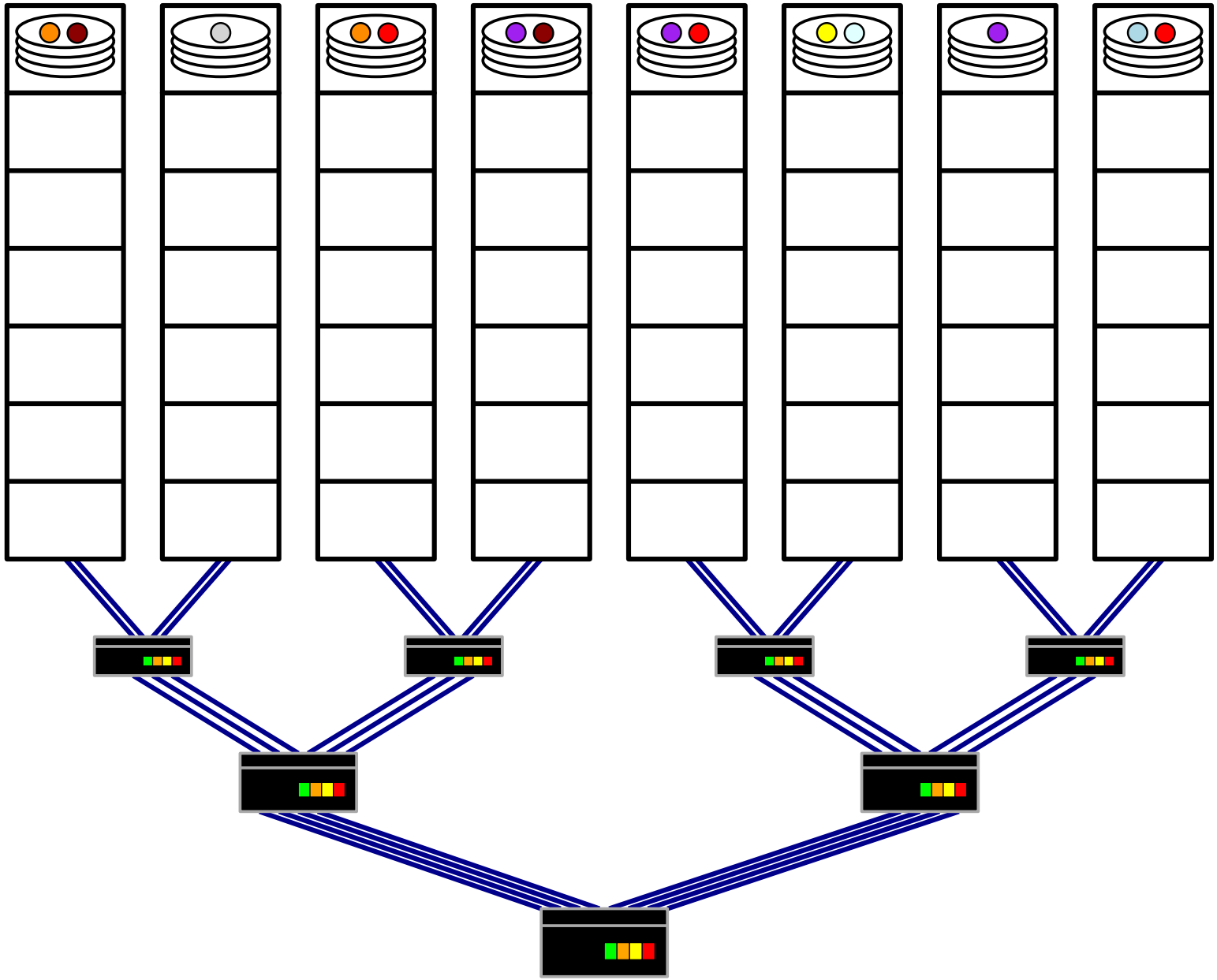
Static placement of virtual machines



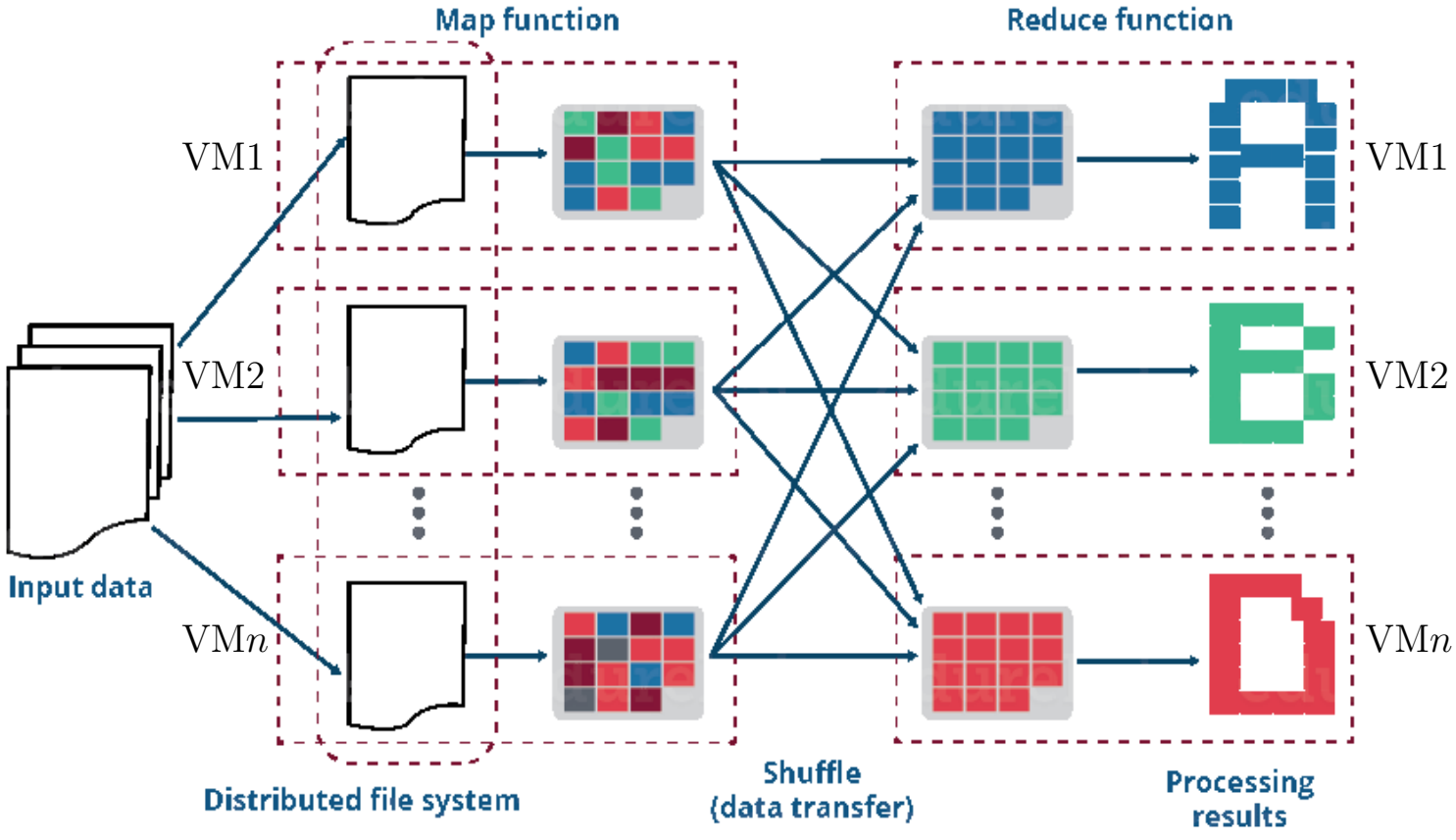
Static placement of virtual machines



Static placement of virtual machines

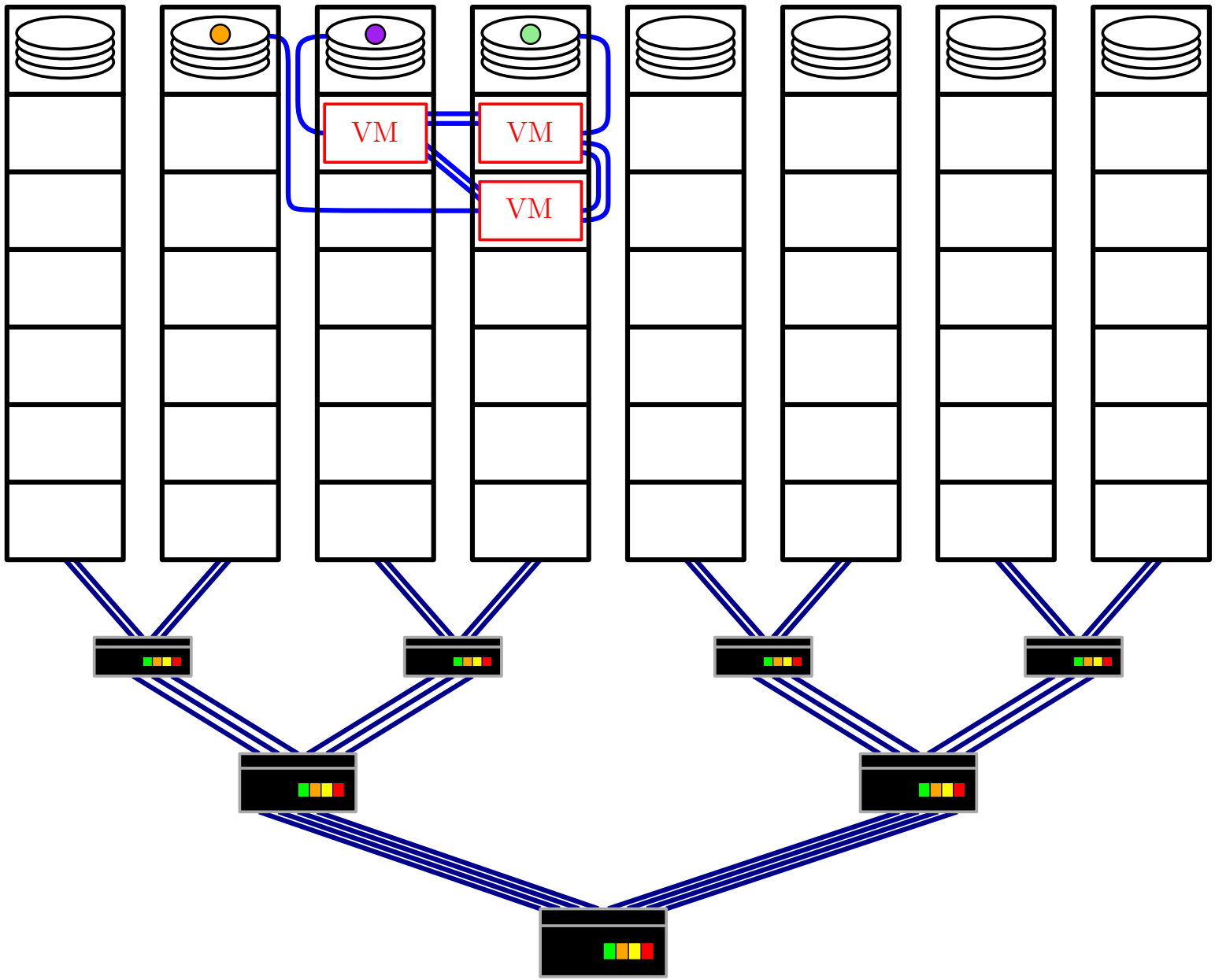


Applications in MapReduce large-scale data processing

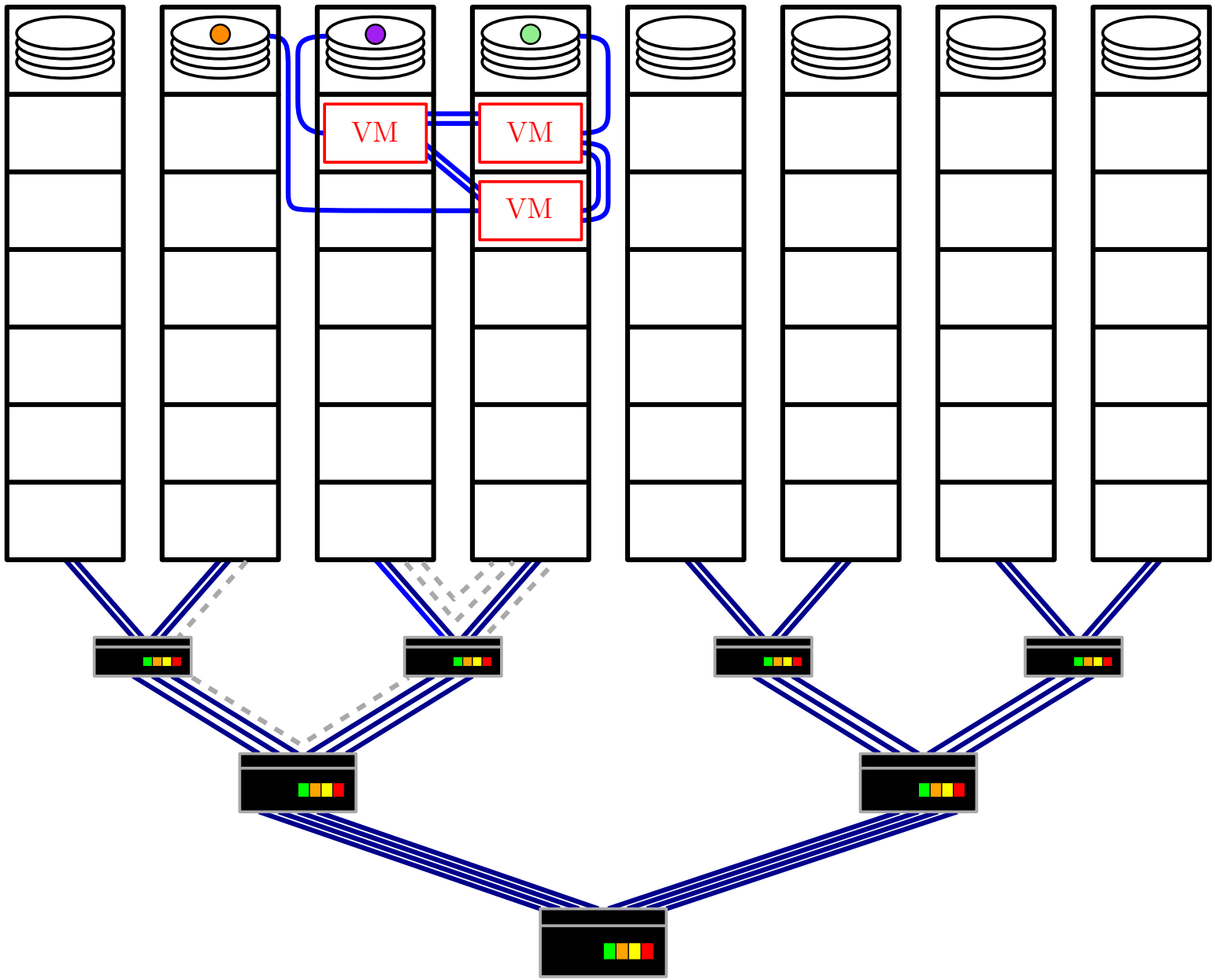


We assume that each pair of VMs communicate.

Static placement of virtual machines



Static placement of virtual machines



Variety of results

We consider multiple variants of the model:

- with/without data replication
- single VM accesses one/more data chunks
- limited/unlimited bandwidth
- ...

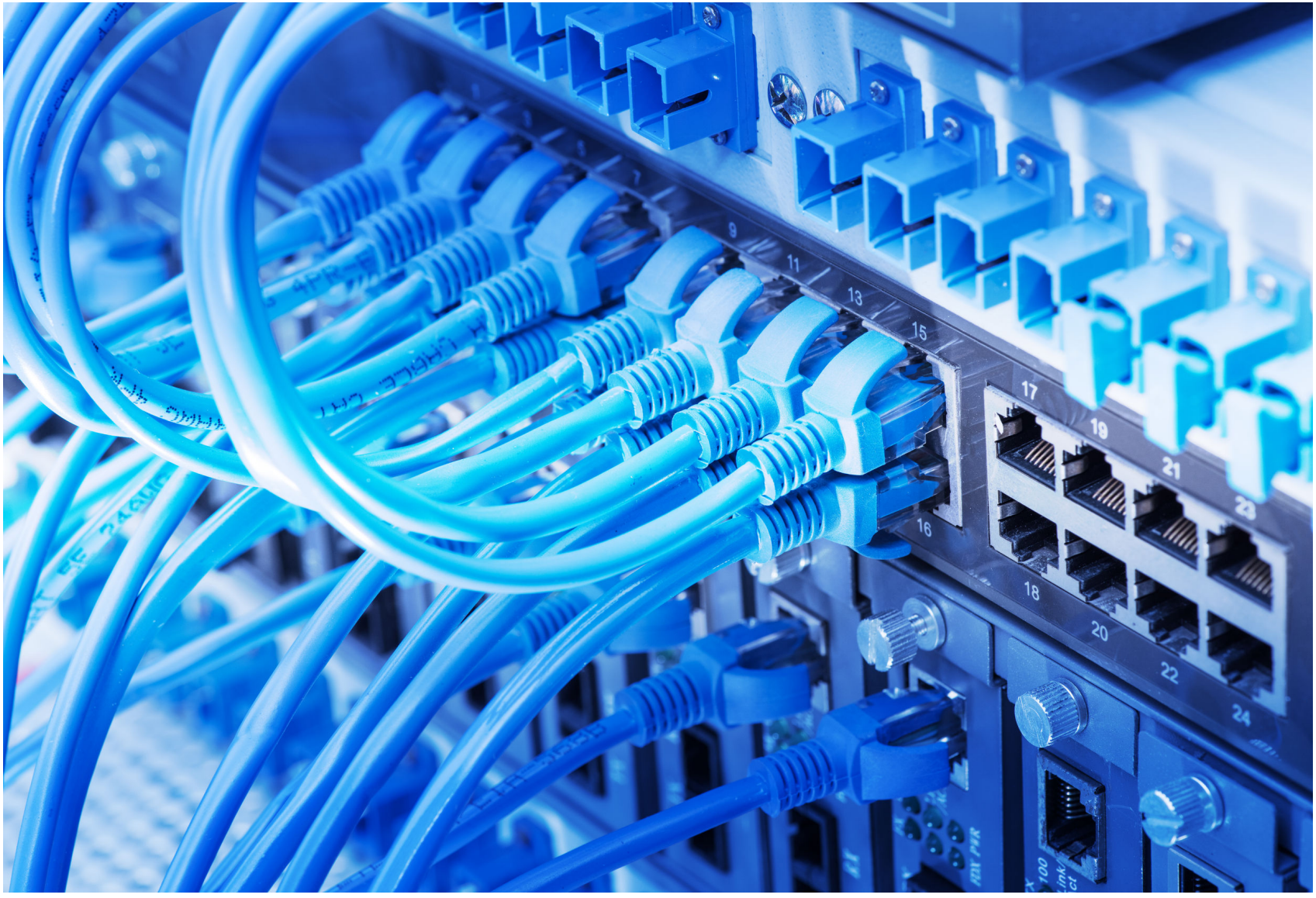
For all variants we classify: P / NP-complete.

Thesis overview

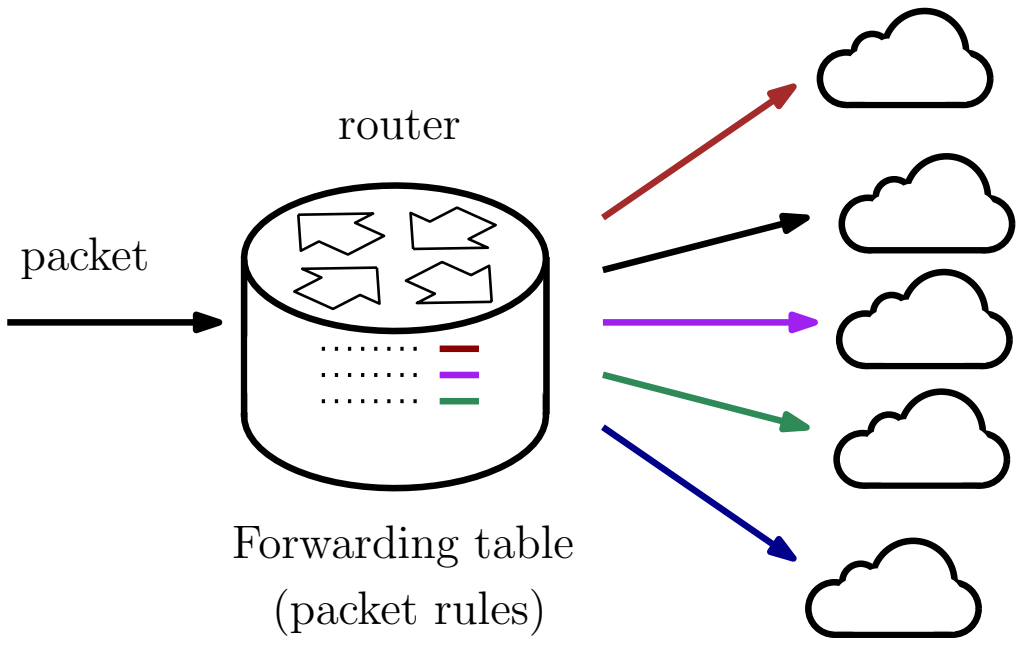
Algorithmic aspects of contemporary networks

- Part 1: Utilizing redundant storage in MapReduce applications (ICNP '15 + TCS '17)
- Part 2: Assigning tasks to machines in data centers (DISC '17 + SIAM J. Disc. Math '20)
- **Part 3: Caching forwarding tables in routers** (ACM SPAA '17)

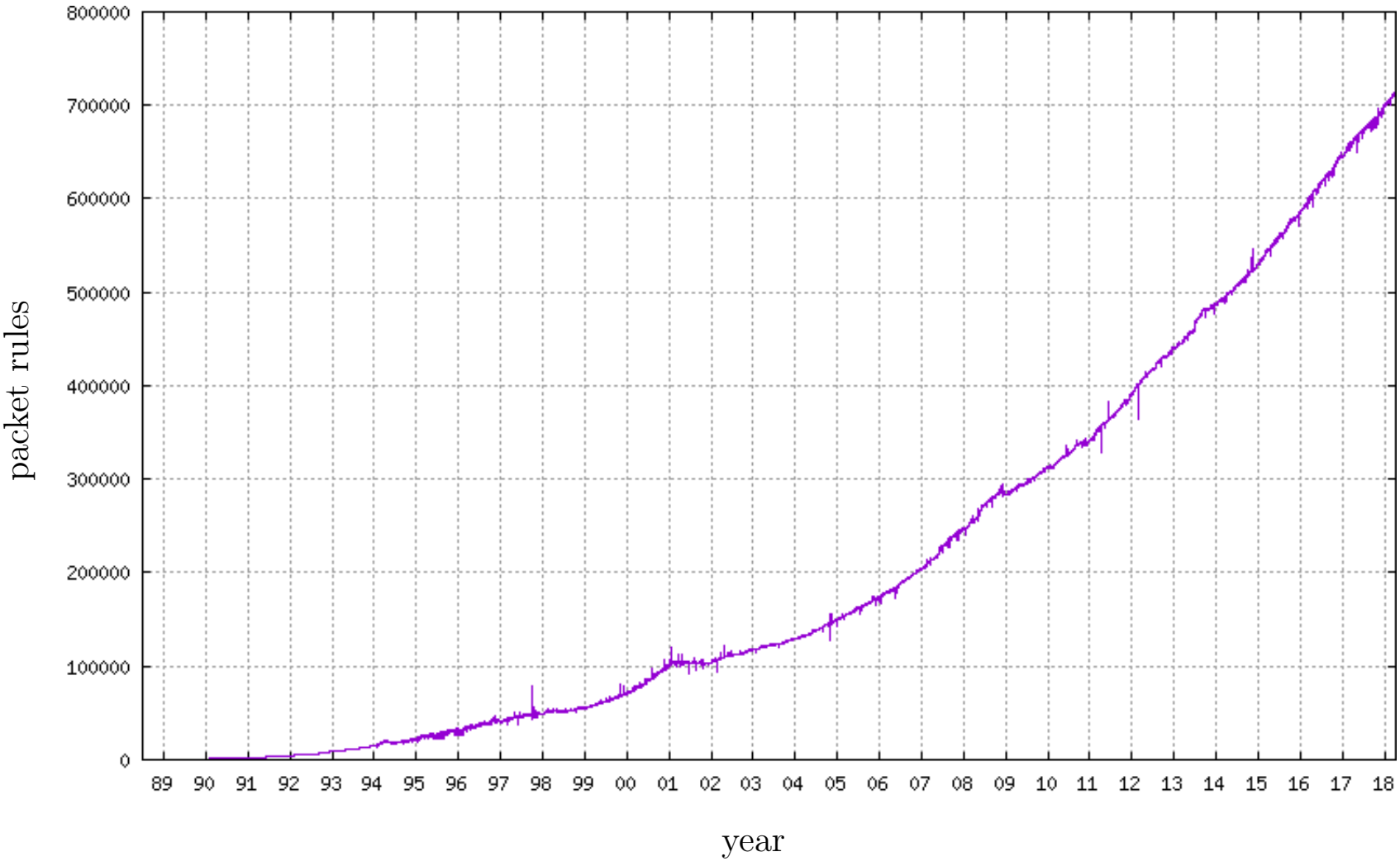
Memory management in routers



Packet forwarding



Growth of forwarding tables



FIB size at AS65000. Report from http://bgp.potaroo.net/as1221

Prolonging lifetime of routers

It is technically feasible to

- store a subset of rules in the fast memory of the router
- store the remainder of rules in external storage

Prolonging lifetime of routers

It is technically feasible to

- store a subset of rules in the fast memory of the router
- store the remainder of rules in external storage

$$\Downarrow$$

Caching?

Prolonging lifetime of routers

It is technically feasible to

- store a subset of rules in the fast memory of the router
- store the remainder of rules in external storage

$$\Downarrow$$

Caching?

Yes. But with additional dependencies.

Prolonging lifetime of routers

It is technically feasible to

- store a subset of rules in the fast memory of the router
- store the remainder of rules in external storage

$$\Downarrow$$

Caching?

Yes. But with additional dependencies.

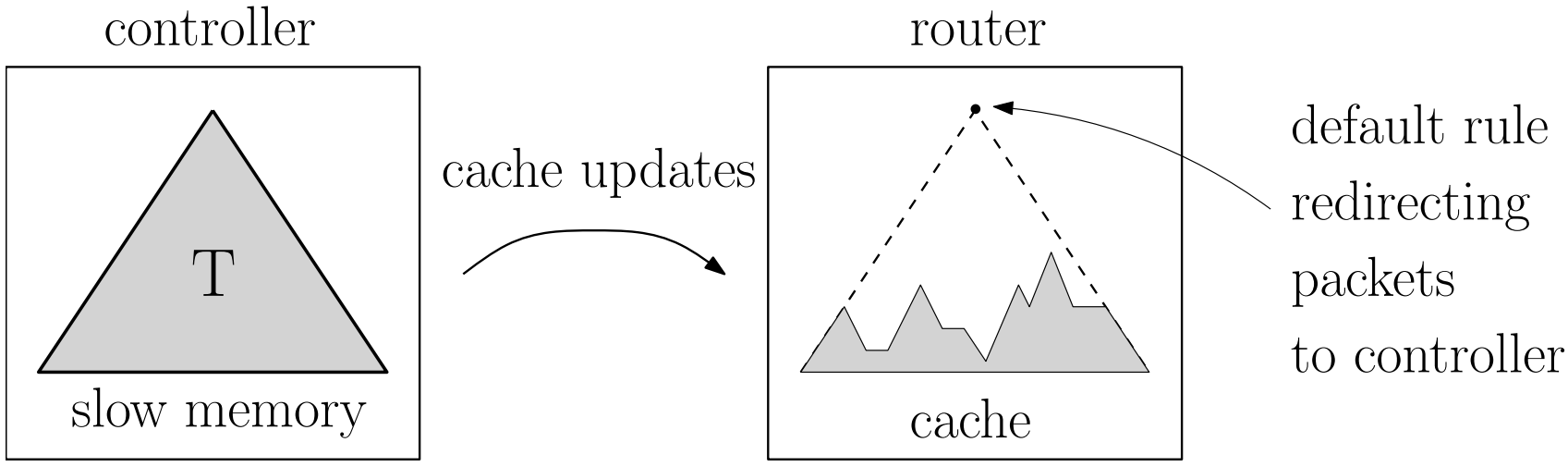
Results (ACM SPAA '18):
Online algorithm with almost optimal competitive ratio.

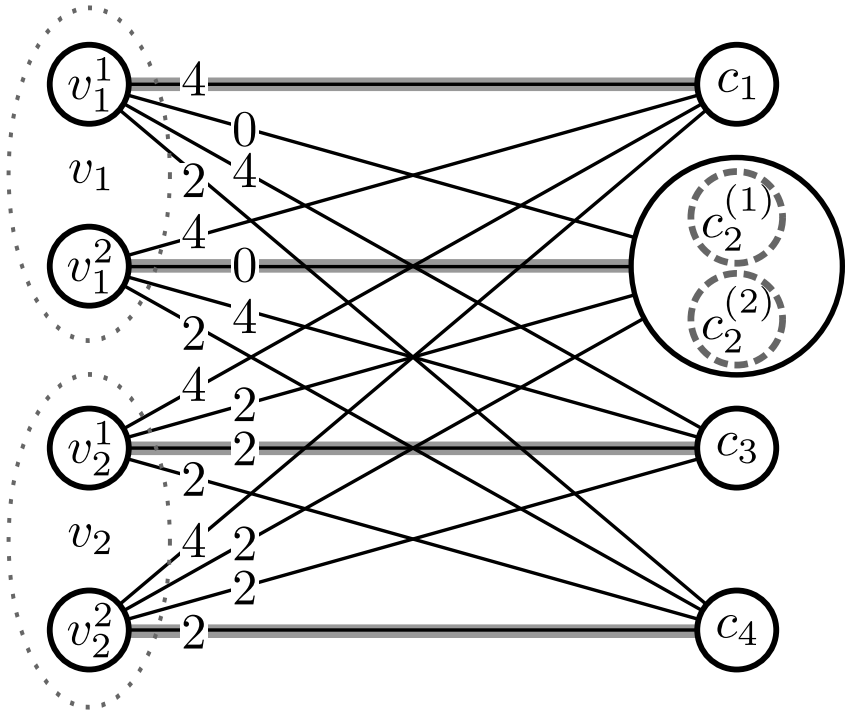
List of papers

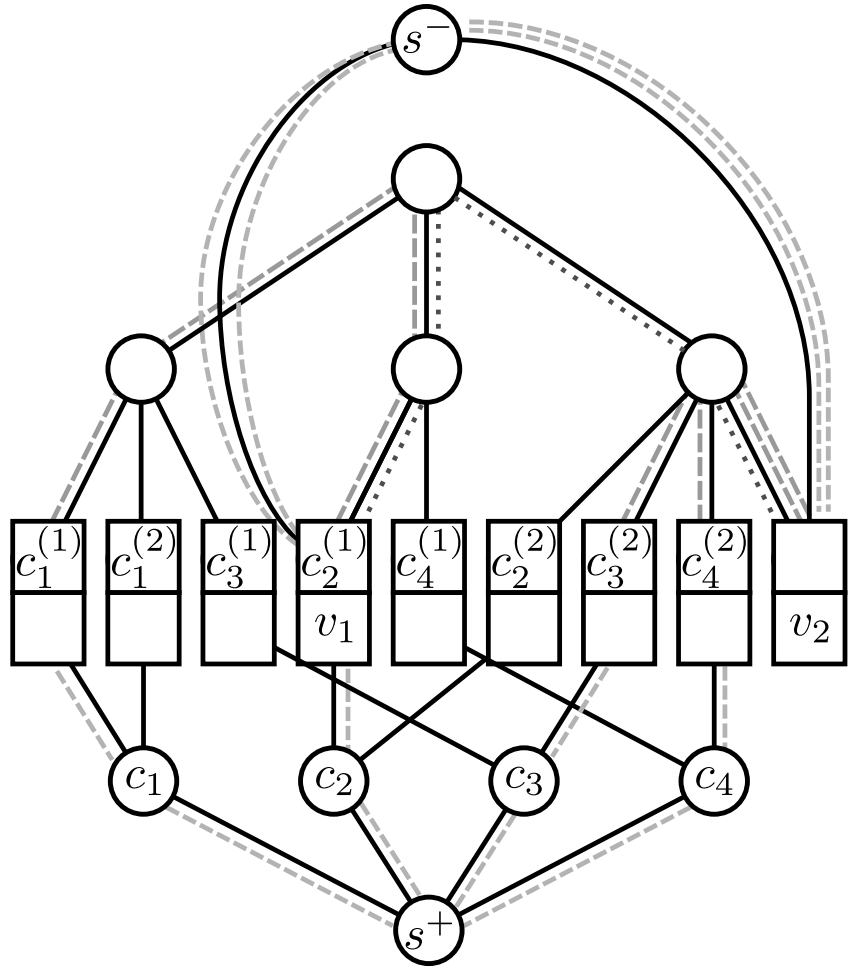
1. Carlo Fuerst, Maciej Pacut, Paolo Costa, Stefan Schmid: *Understanding the Complexity of Replica Aware Virtual Cluster Embeddings*, ICNP 2015
2. Chen Avin, Andreas Loukas, Maciej Pacut, Stefan Schmid: *Online Balanced Repartitioning*, DISC 2016
3. Marcin Bienkowski, Jan Marcinkowski, Maciej Pacut, Stefan Schmid, Aleksandra Spyra: *Online Tree Caching*, SPAA 2017
4. Carlo Fuerst, Maciej Pacut, Stefan Schmid: *Data locality and replica aware virtual cluster embeddings*, Journal of Theoretical Computer Science 697
5. Klaus-Tycho Foerster, Maciej Pacut, Stefan Schmid: *On the Complexity of Non-Segregated Routing in Reconfigurable Data Center Architectures*, Computer Communication Review 49
6. Chen Avin, Andreas Loukas, Maciej Pacut, Stefan Schmid: *Dynamic Balanced Partitioning*, SIAM Journal on Discrete Mathematics (SIDMA)

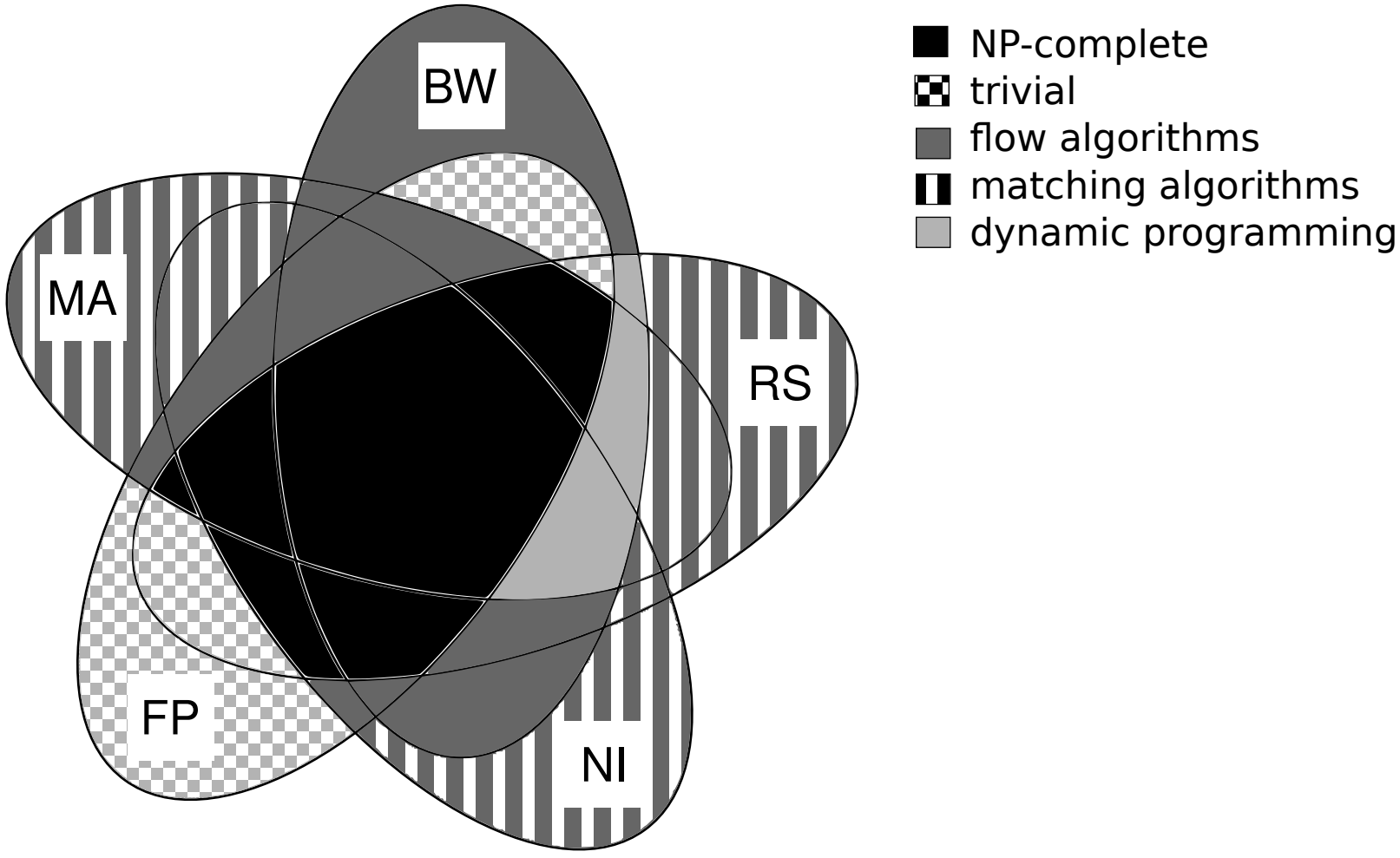
This thesis was supported by NCN grant Preludium 2016/23/N/ST6/03412.

Thanks for your attention!


controller
T
slow memory
cache updates
router
cache
default rule
redirecting
packets
to controller


v_1^1
v_1^2
v_2^1
v_2^2
v_1
v_2
4
0
4
2
4
0
4
2
4
2
2
2
4
2
2
2
c_1
$c_2^{(1)}$
$c_2^{(2)}$
c_3
c_4


s^-
$c_1^{(1)}$
$c_1^{(2)}$
$c_3^{(1)}$
$c_2^{(1)}$
$c_4^{(1)}$
$c_2^{(2)}$
$c_3^{(2)}$
$c_4^{(2)}$
v_1
v_2
c_1
c_2
c_3
c_4
s^+


BW
MA
RS
FP
NI
NP-complete
trivial
flow algorithms
matching algorithms
dynamic programming

Static placement of virtual machines

Communication pattern is known

=

embedding problem of communication pattern into interconnecting network

Communication pattern is known

+

placement of virtual machines is known

=

VPN problem

Interesting open problems in online algorithm

- k-server
  - deterministic k-server hypothesis: LB: k, UB: 2k-1
  - randomized k-server hypothesis: is there *logk* LB for any metric?
  - LB for uniform: logk, UB general: $\log^6 k$, using tight result for HST $\log^2 k$
- MTS
  - is there $\Omega(\log n)$ for general metric?
  - ALG: $\log n$ on HST $\rightarrow \log^2 n$ for general
- Set Cover: LB: $\frac{\log n \log m}{\log\log n + \log\log m}$ vs UB: $\log n \log m$