NSDI'13

Scalable Rule Management for Data Centers

Masoud Moshref, Minlan Yu,

Abhishek Sharma, Ramesh Govindan

4/3/2013

USC University of Southern California

NSL

NEC

Introduction: Definitions

Datacenters use rules to implement **management policies**

- Access control
- Rate limiting
- Traffic measurement
- Traffic engineering

Introduction: Definitions

Datacenters use **rules** to implement management policies

An **action** on a set of ranges on **flow fields**

Examples:

- Deny
- Accept
- Enqueue

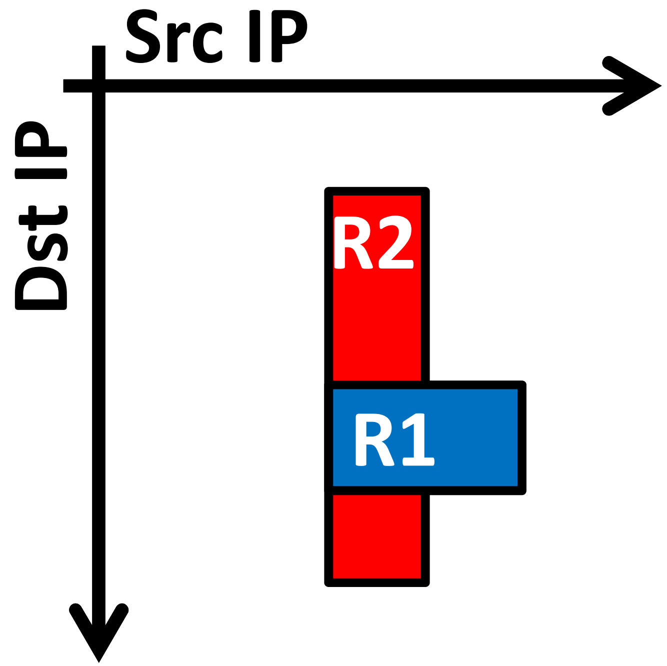
Flow fields examples:

- Src IP / Dst IP
- Protocol
- Src Port / Dst Port

Introduction: Definitions

Datacenters use **rules** to implement management policies

An **action** on a set of ranges on **flow fields**



R1: Accept

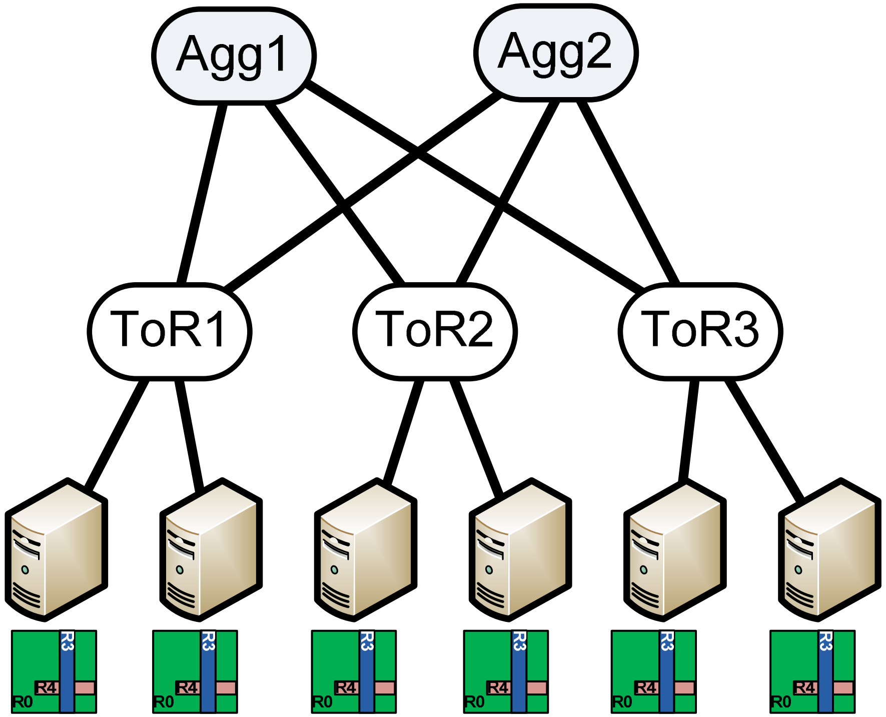
- SrcIP: 12.0.0.0/7
- DstIP: 10.0.0.0/8

R2: Deny

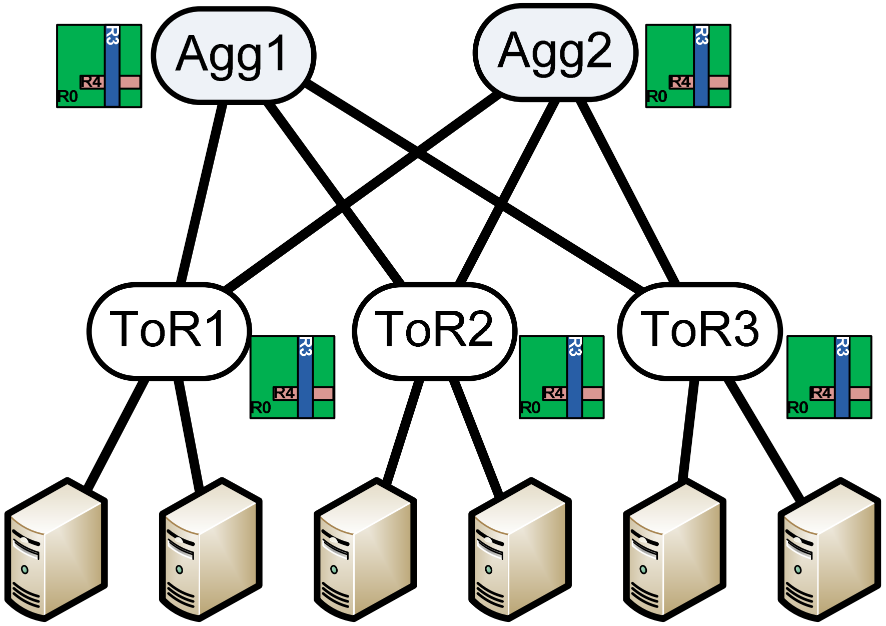
- SrcIP: 12.0.0.0/8
- DstIP: 8.0.0.0/6

Current practice

Rules are saved on **predefined fixed** machines



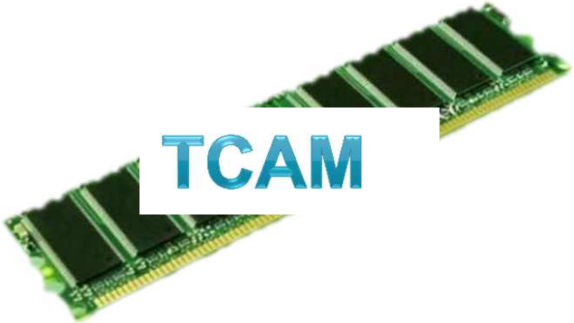
On hypervisors



On switches

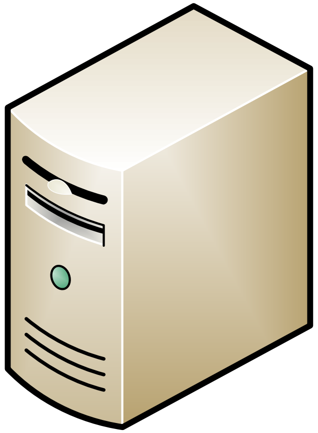
Machines have limited resources

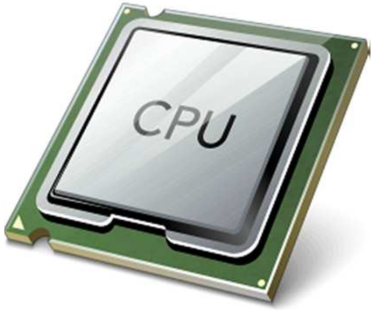
Top-of-Rack switch



Network Interface Card

Software switches on servers





Future datacenters will have many fine-grained rules

VLAN per server

- Traffic management (NetLord, Spain)

1M rules

Per flow decision

- Flow measurement for traffic engineering (MicroTE, Hedera)

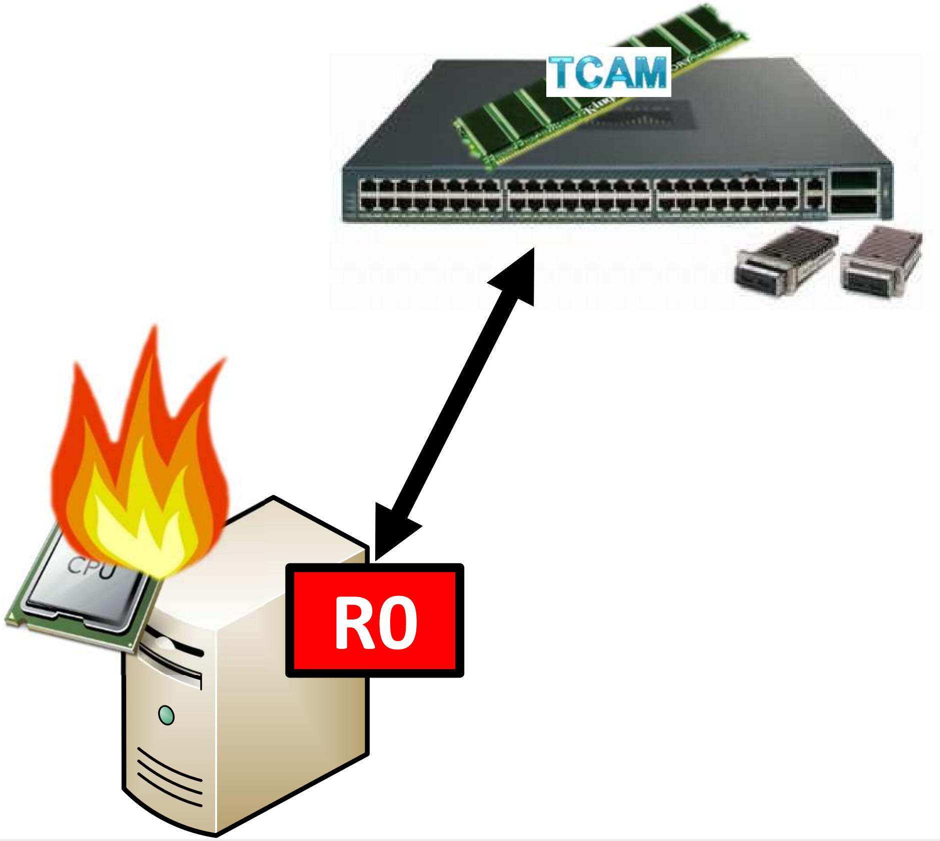
10M – 100M rules

Regulating VM pair communication

- Access control (CloudPolice)
- Bandwidth allocation (Seawall)

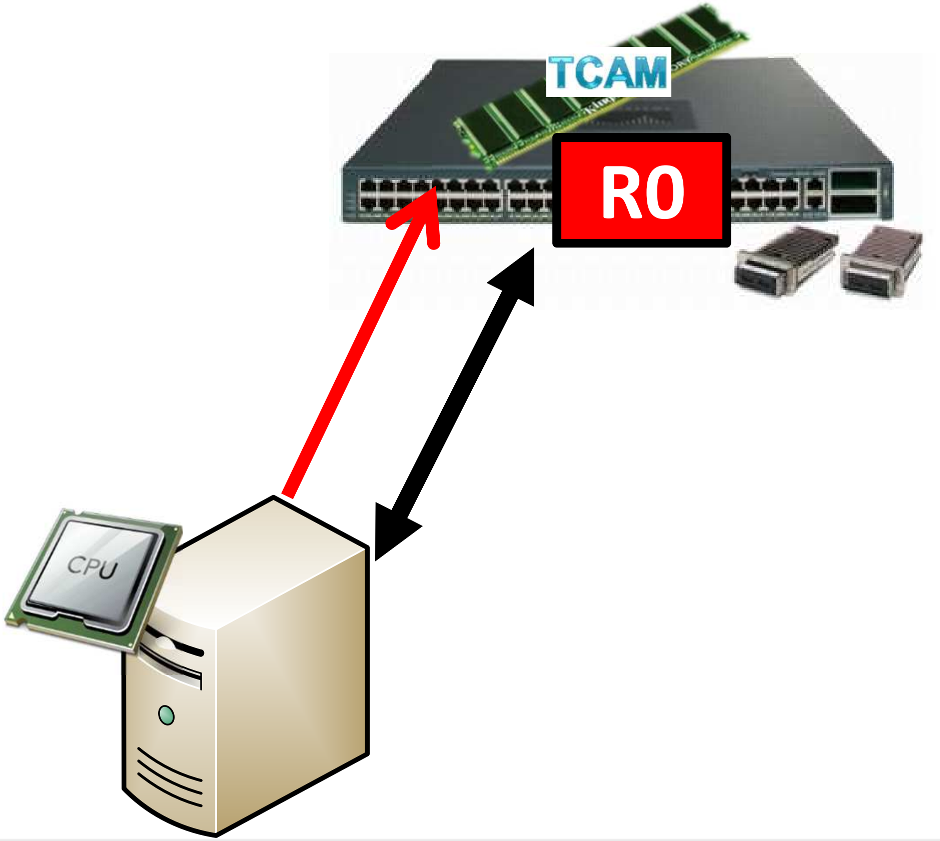
1B – 20B rules

Rule location trade-off (resource vs. bandwidth usage)



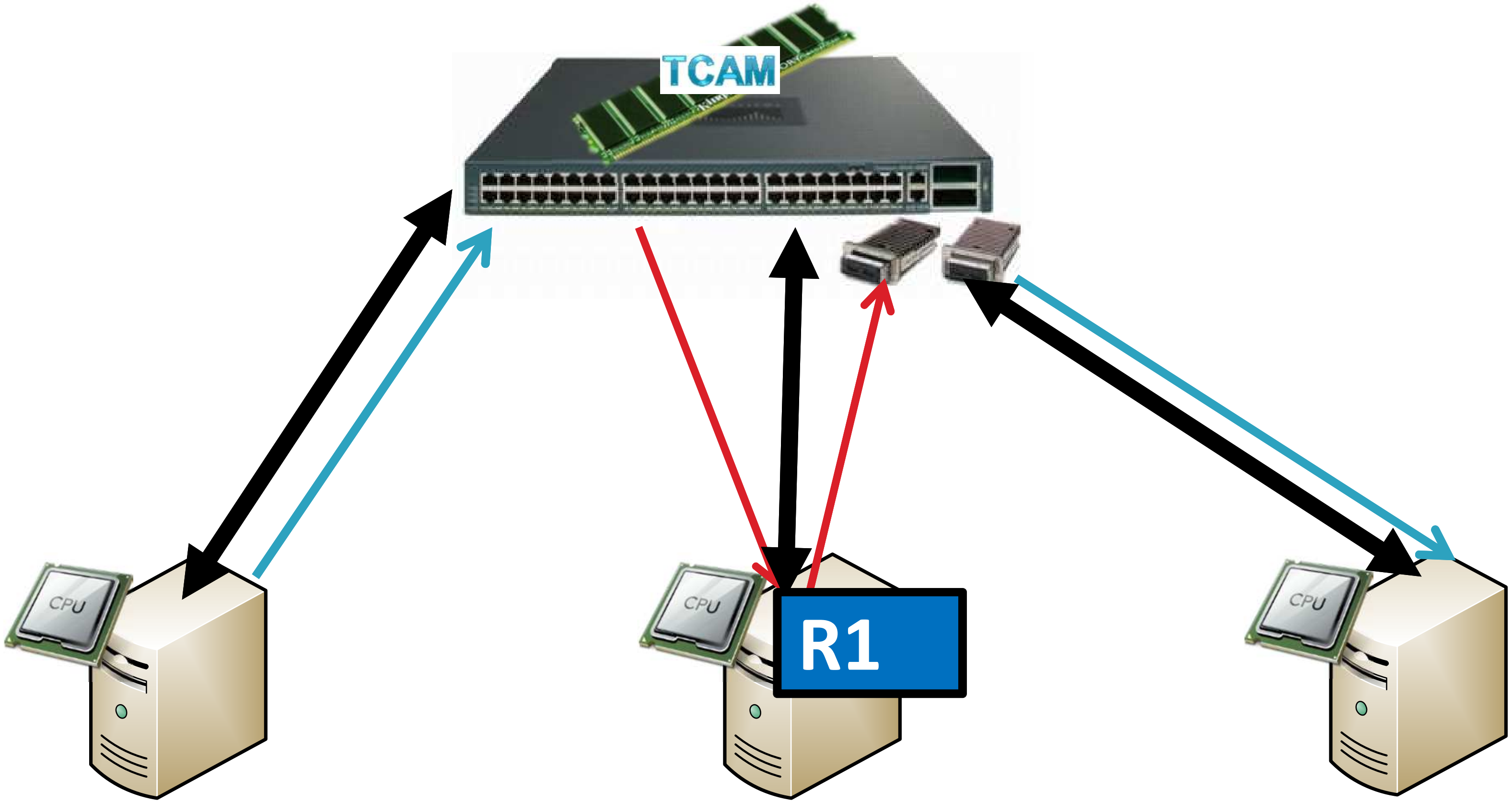
Storing rules at hypervisor incurs CPU overhead

Rule location trade-off (resource vs. bandwidth usage)

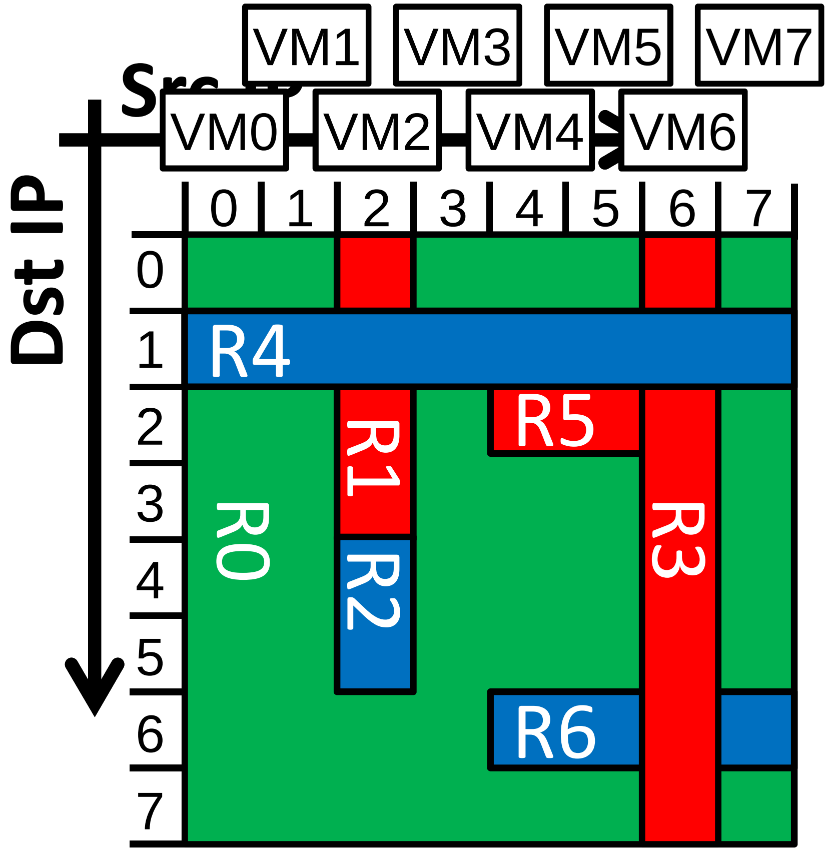


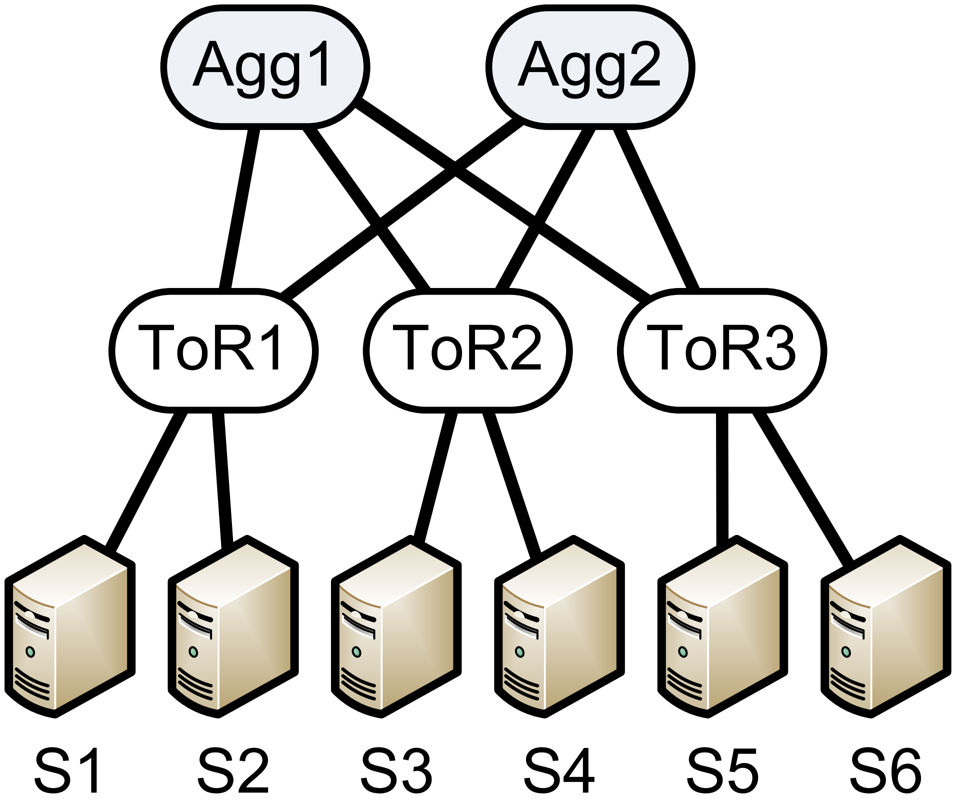
Move the rule to ToR switch and forward traffic

Rule location trade-off: Offload to servers



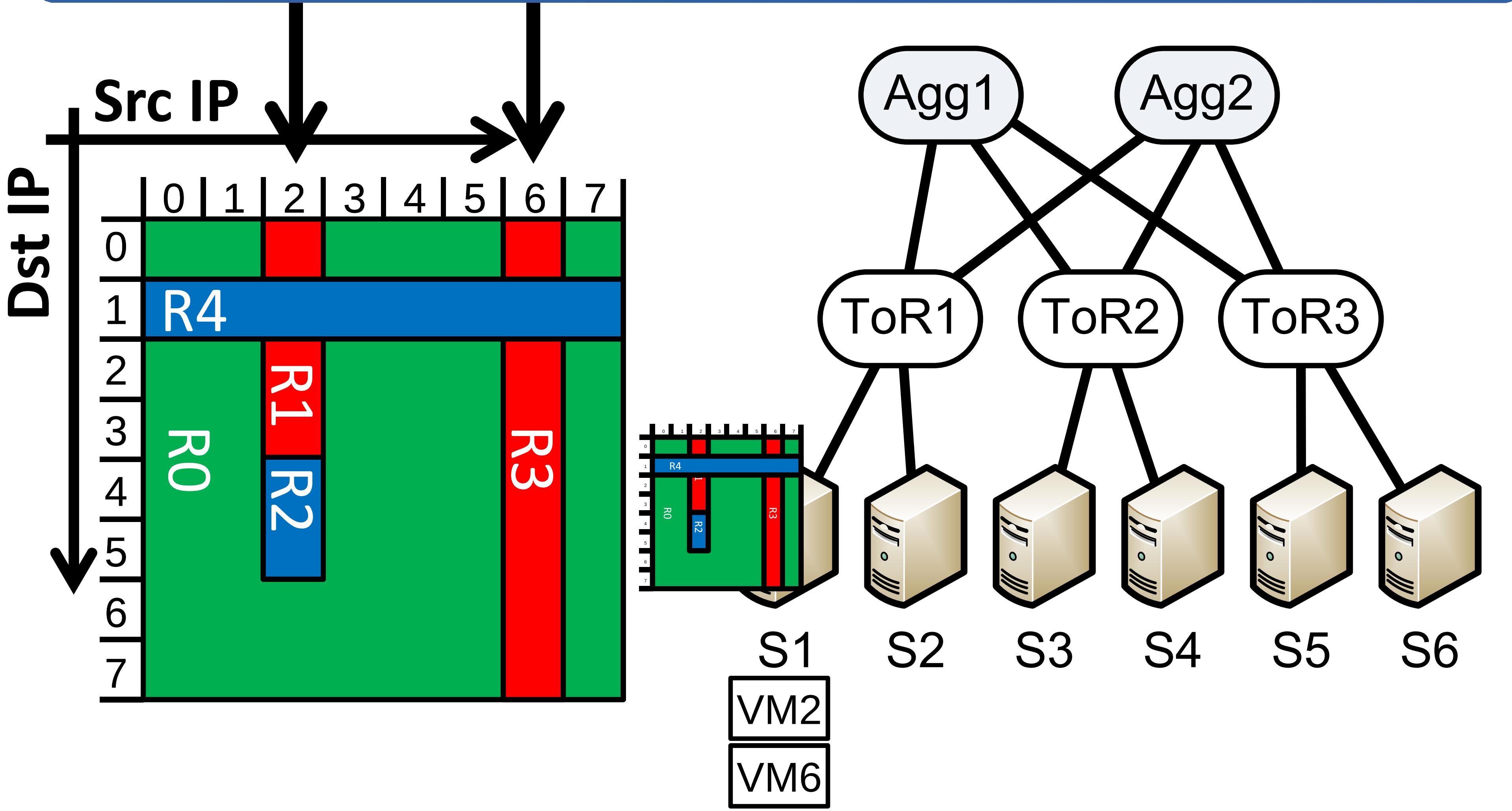
Challenges: Concrete example





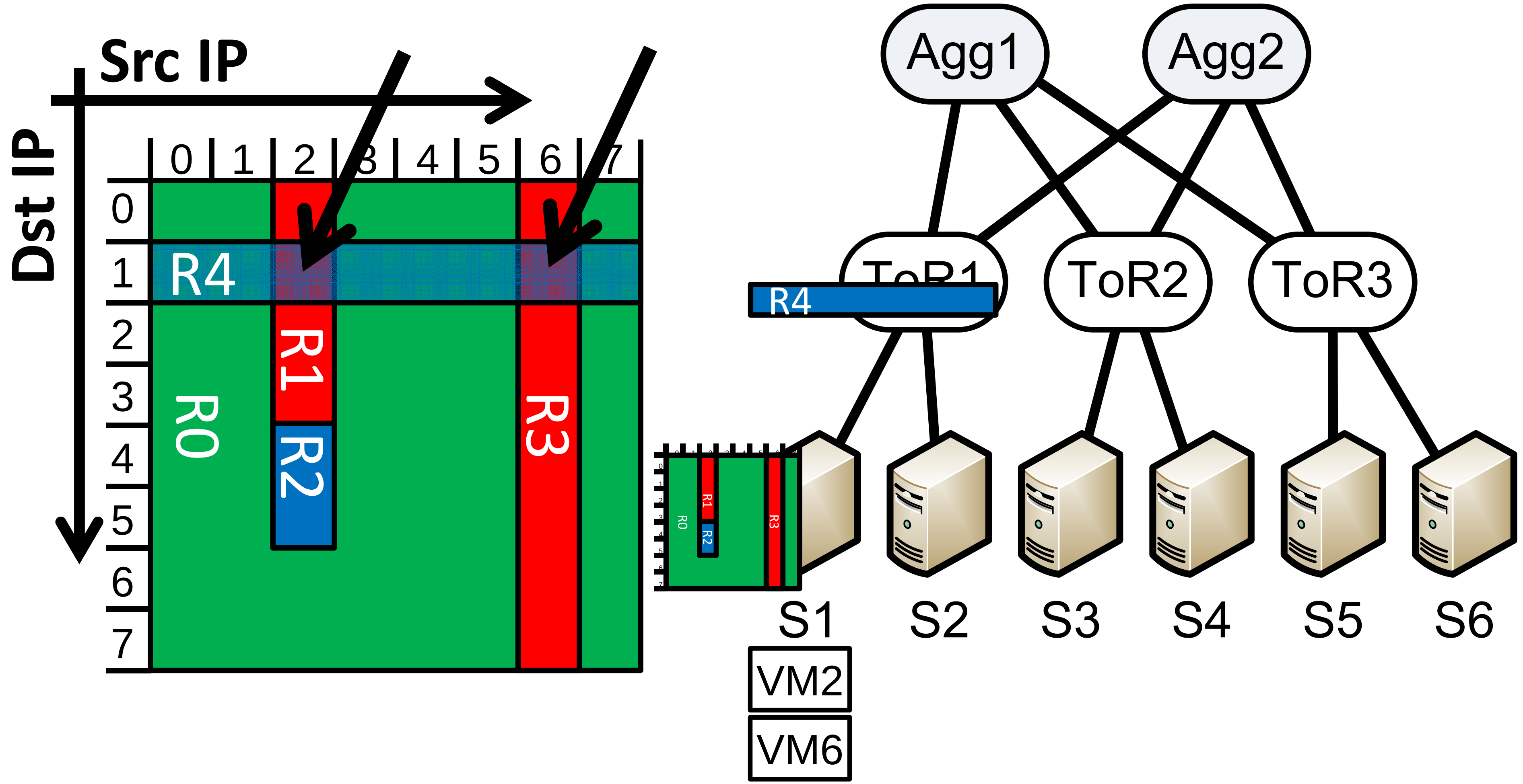
Challenges: Overlapping rules

Source Placement: Saving rules on the source machine means minimum overhead



Challenges: Overlapping rules

If Source Placement is not feasible



Challenges

Preserve the semantics of overlapping rules

Respect resource constraints

Heterogeneous devices

Minimize traffic overhead

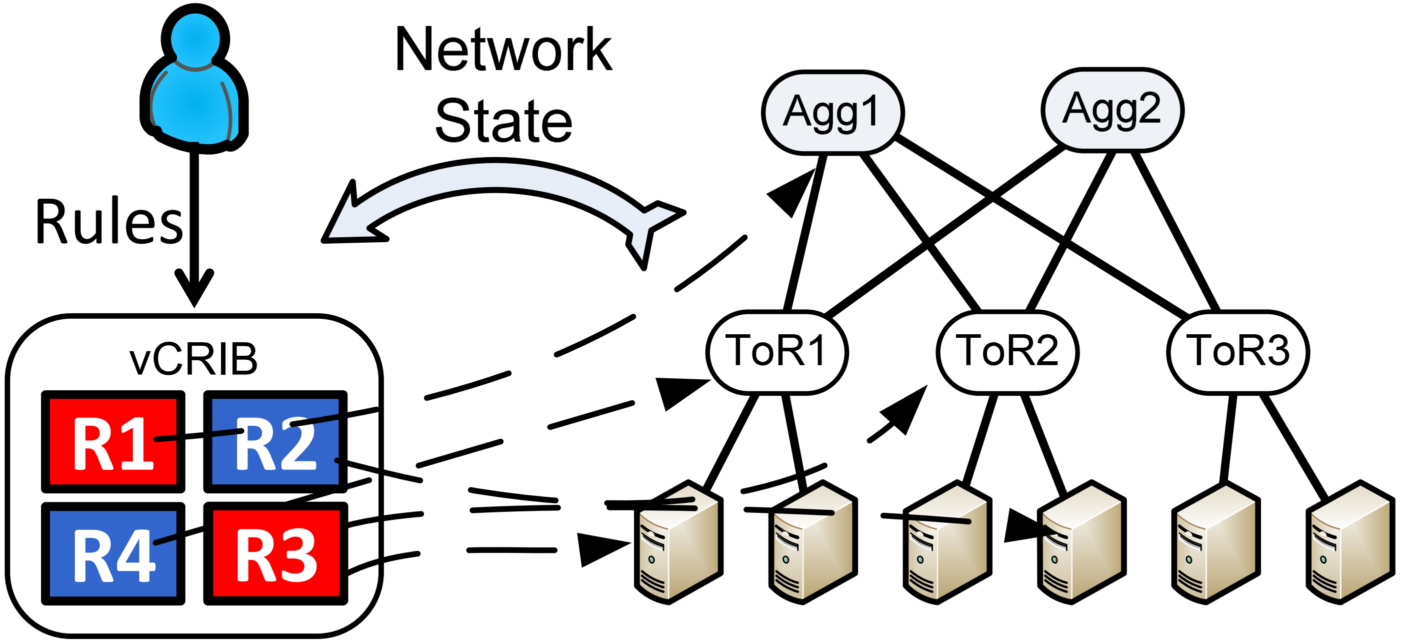
Handle Dynamics

Traffic changes
Rule changes
VM Migration

Contribution: vCRIB, a Virtual Cloud Rule Information Base

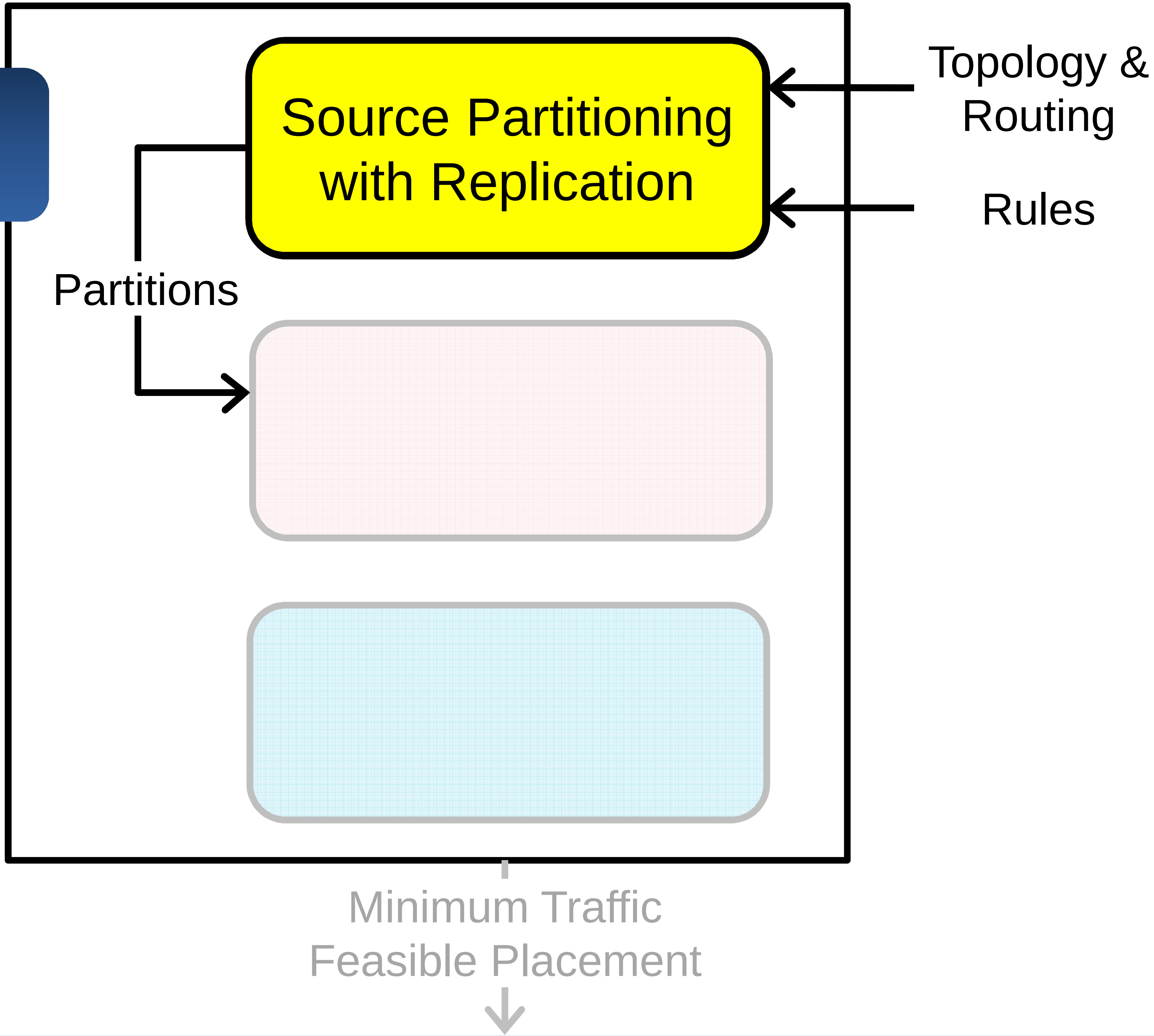
Proactive rule placement abstraction layer

Optimize traffic given resource constraints & changes

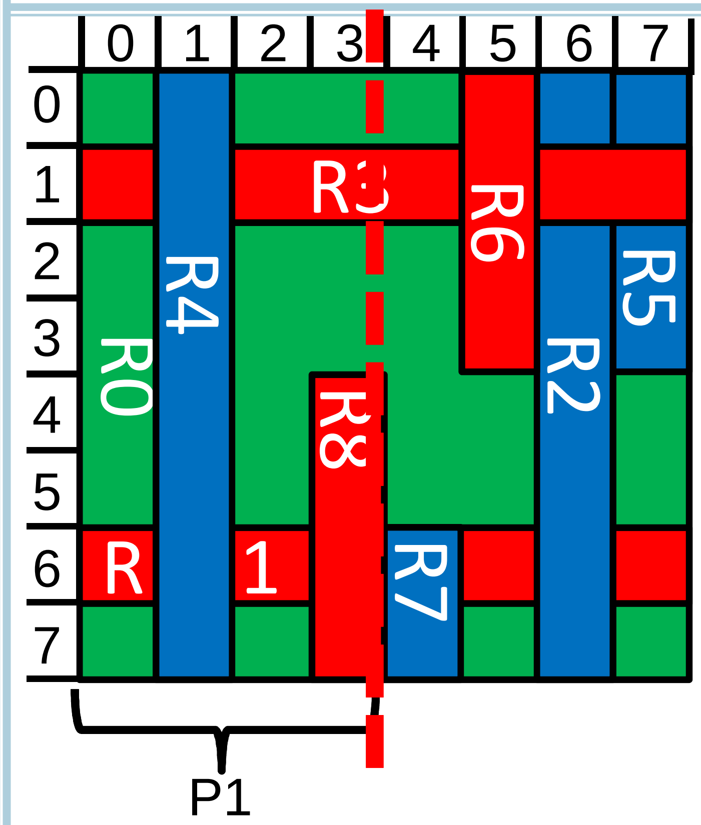


vCRIB design

Overlapping Rules

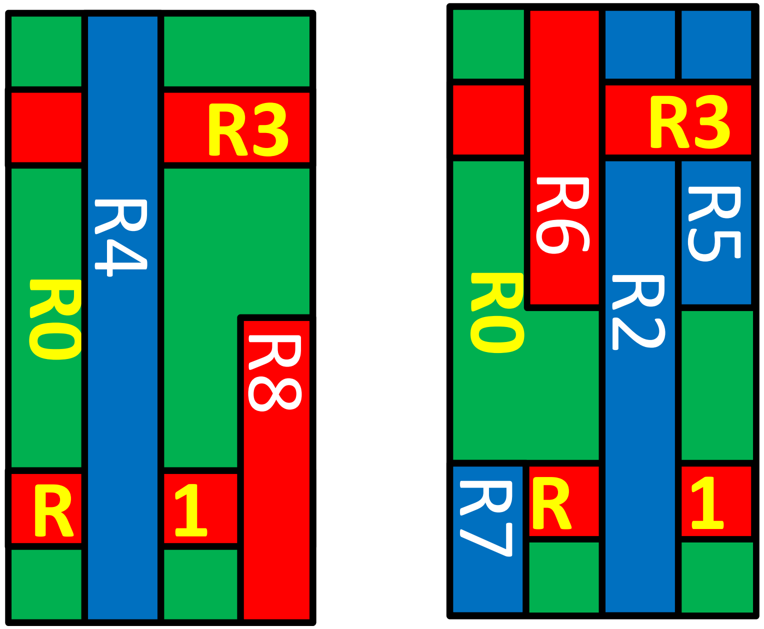

Source Partitioning with Replication
Topology & Routing
Rules
Partitions
Minimum Traffic Feasible Placement

Partitioning with cutting



Smaller partitions have more flexibility

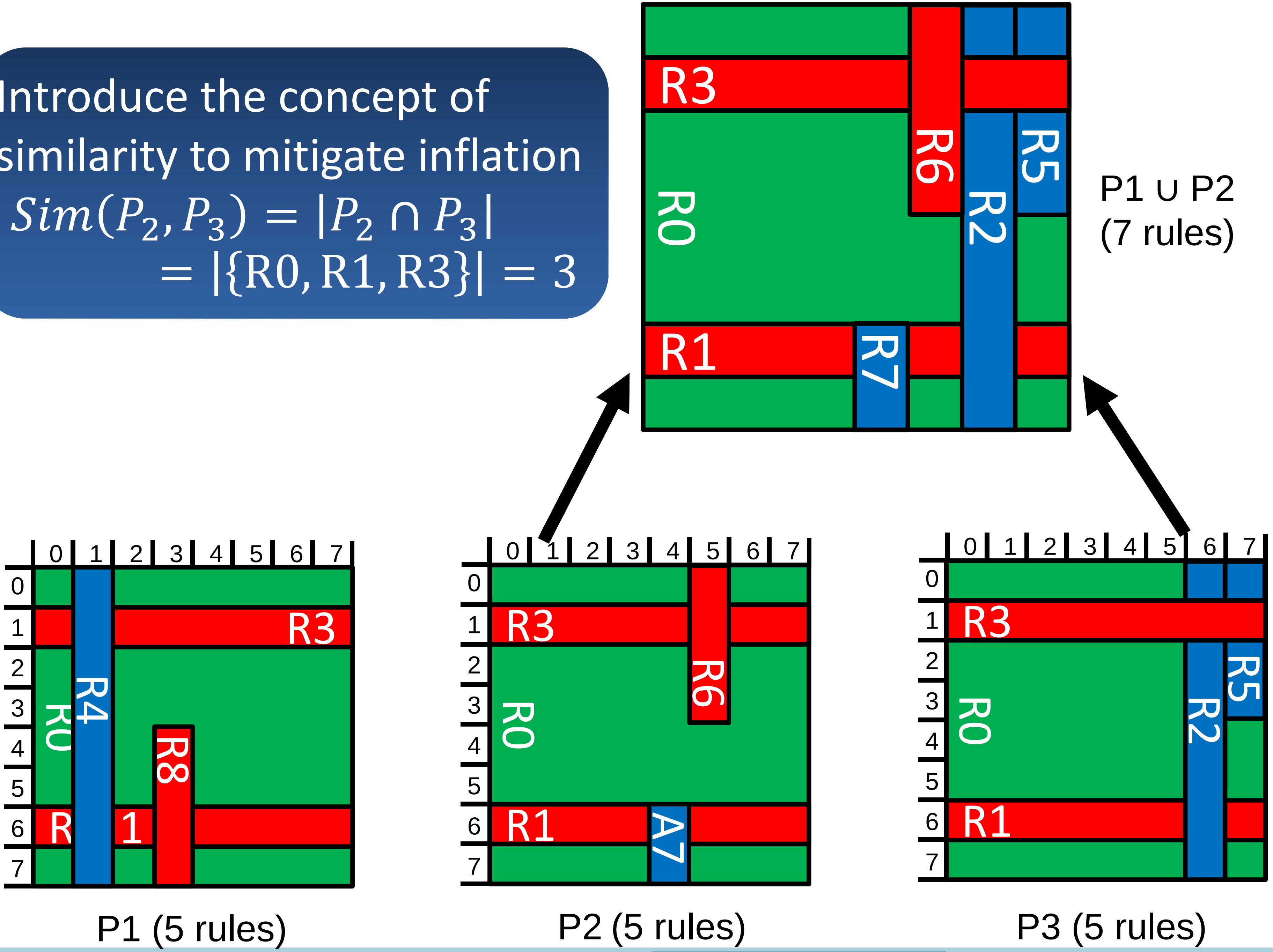
Cutting causes rule inflation



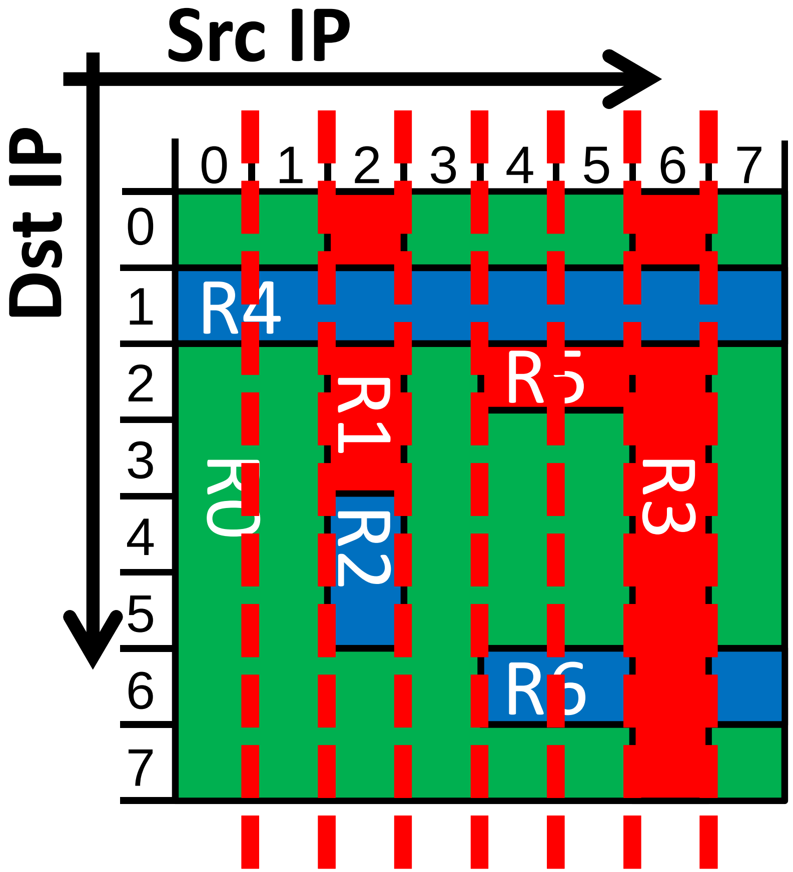
Partitioning with replication

Introduce the concept of similarity to mitigate inflation

$$Sim(P_2, P_3) = |P_2 \cap P_3| \\ = |\{\mathrm{R0}, \mathrm{R1}, \mathrm{R3}\}| = 3$$

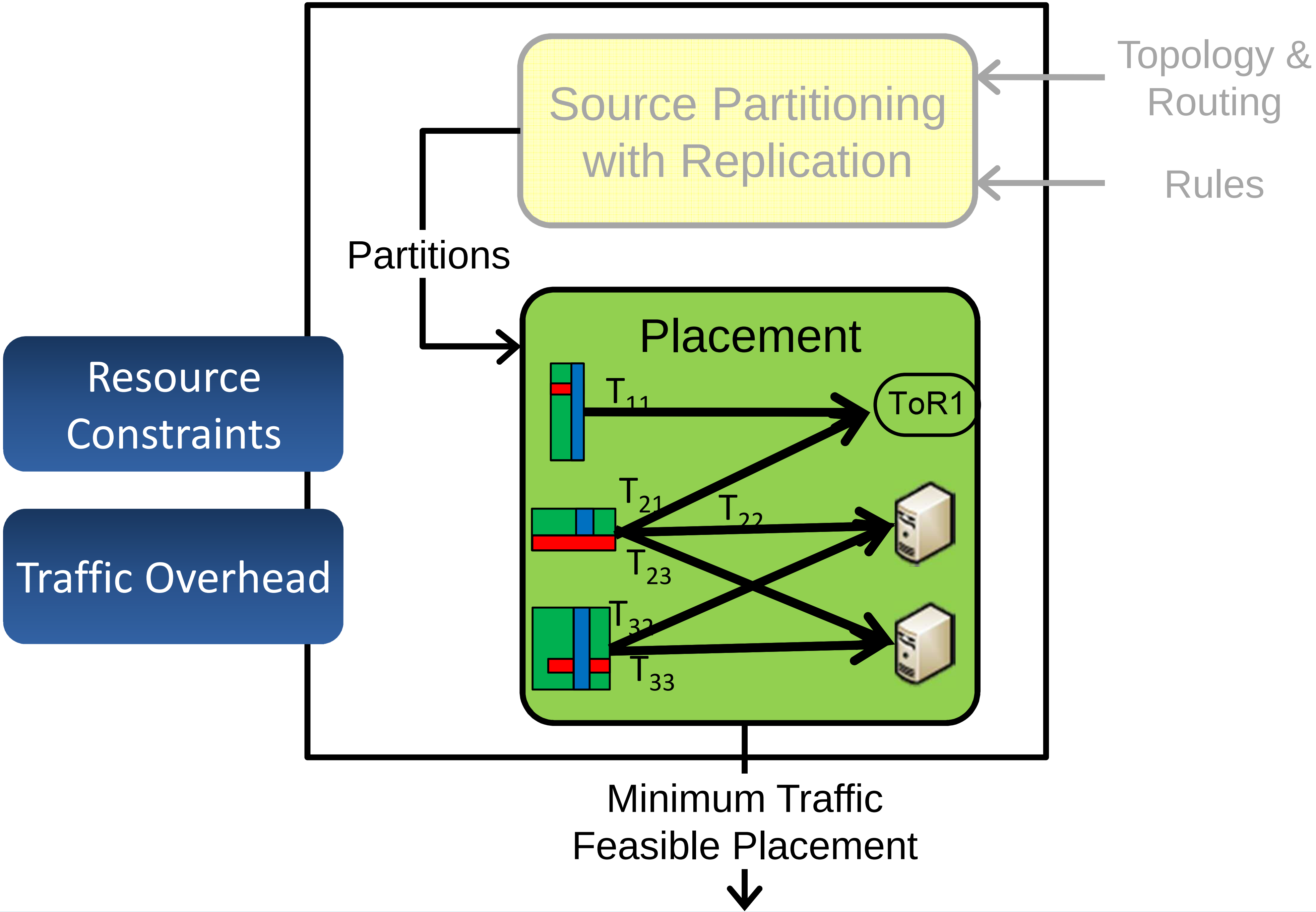


Per-source partitions



- Limited resource for forwarding
- No need for replication to approximate source-placement
- Closer partitions are more similar

vCRIB design: Placement



Resource Constraints

Traffic Overhead

Source Partitioning with Replication

Topology & Routing

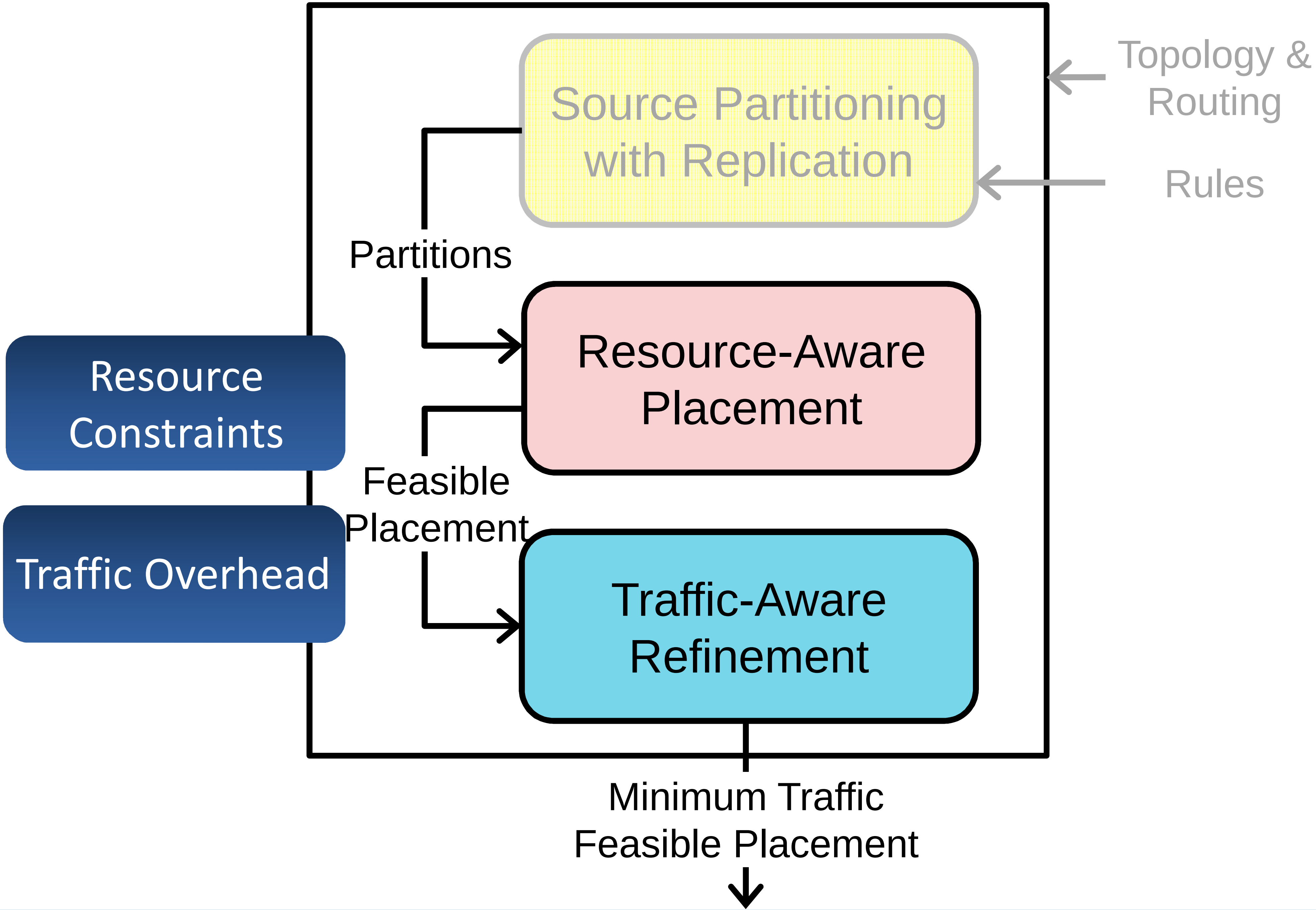
Rules

Partitions

Placement

Minimum Traffic Feasible Placement

vCRIB design: Placement



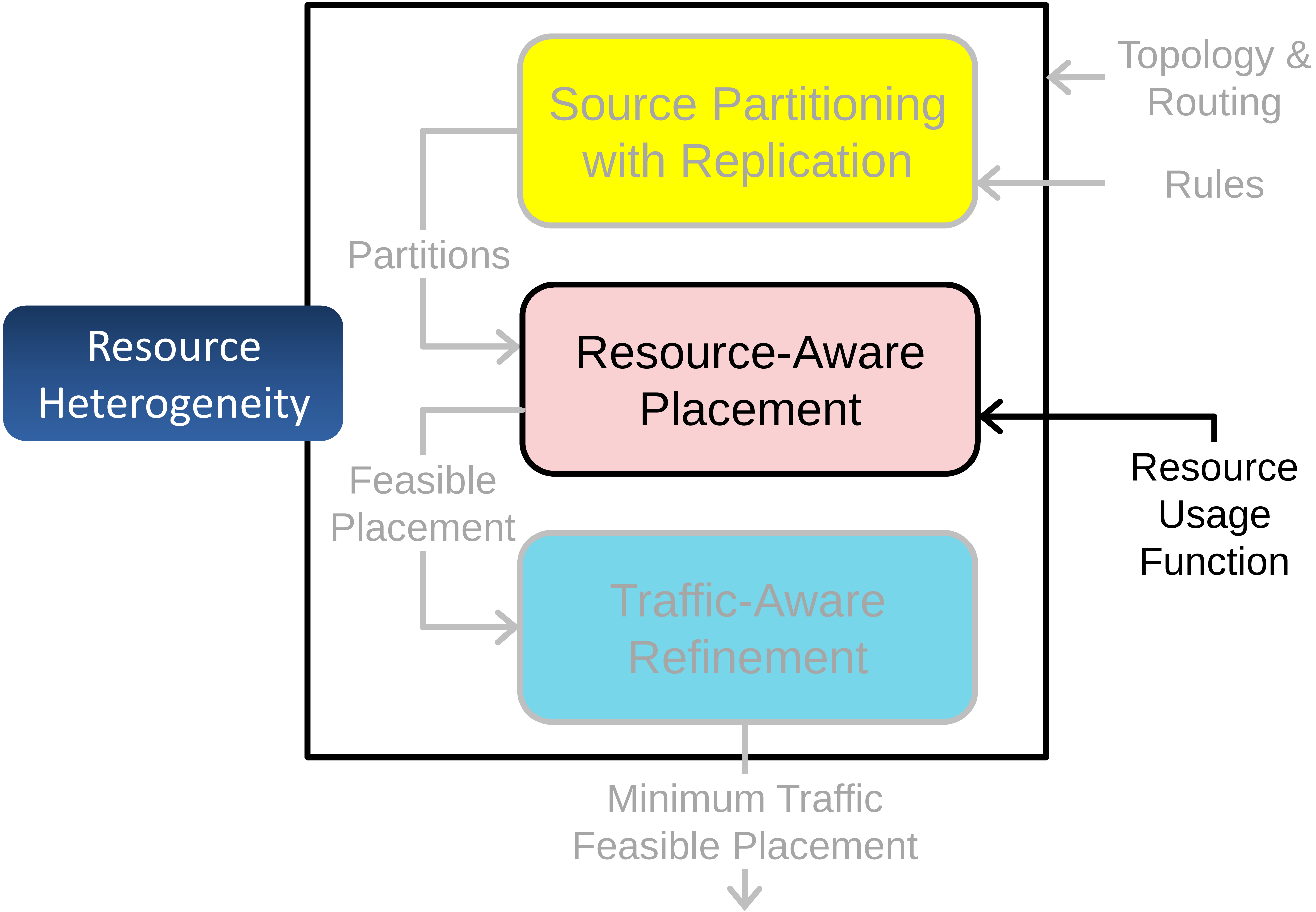
FFDS (First Fit Decreasing Similarity)

1. Put a random partition on an empty device
2. Add the most similar partitions to the initial partition until the device is full

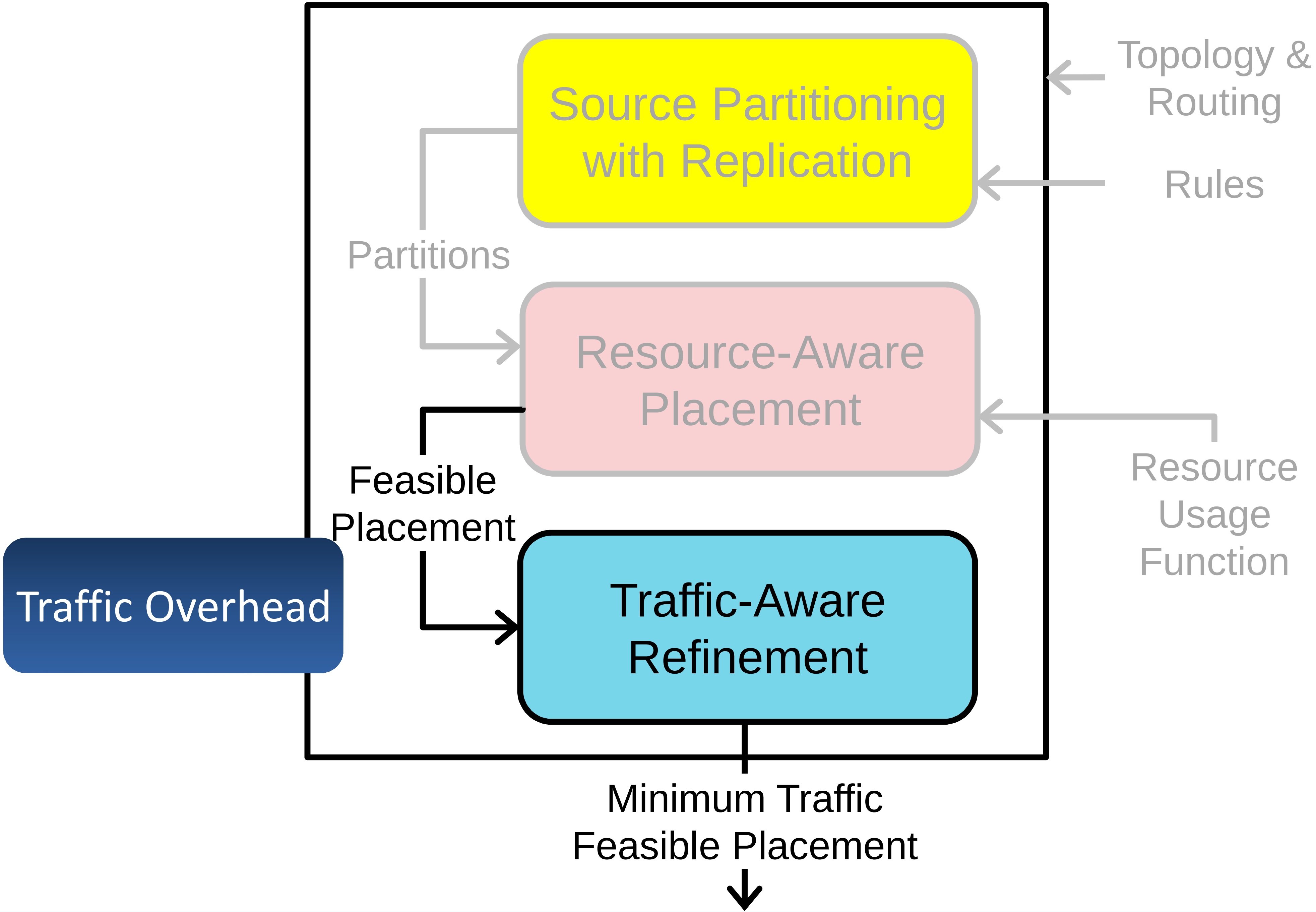
Find the lower bound for optimal solution for rules

Prove the algorithm is a 2-approximation of the lower bound

vCRIB design: Heterogeneous resources

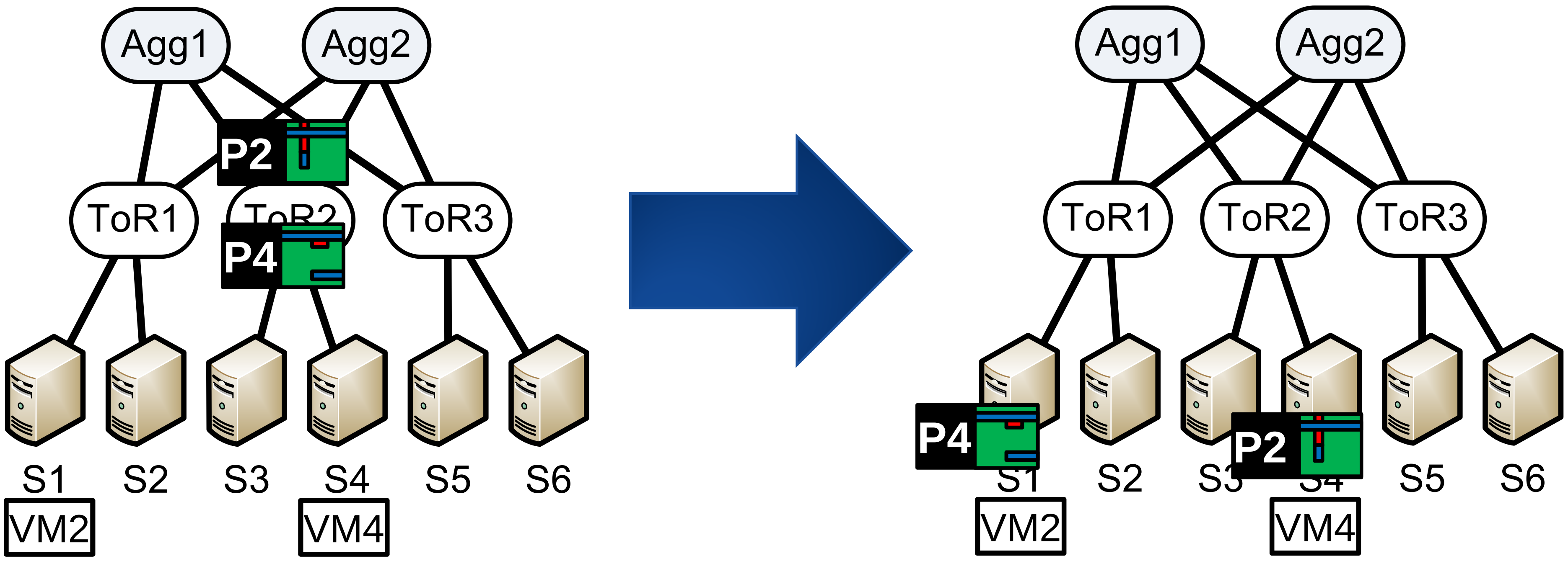


vCRIB design: Traffic-Aware Refinement



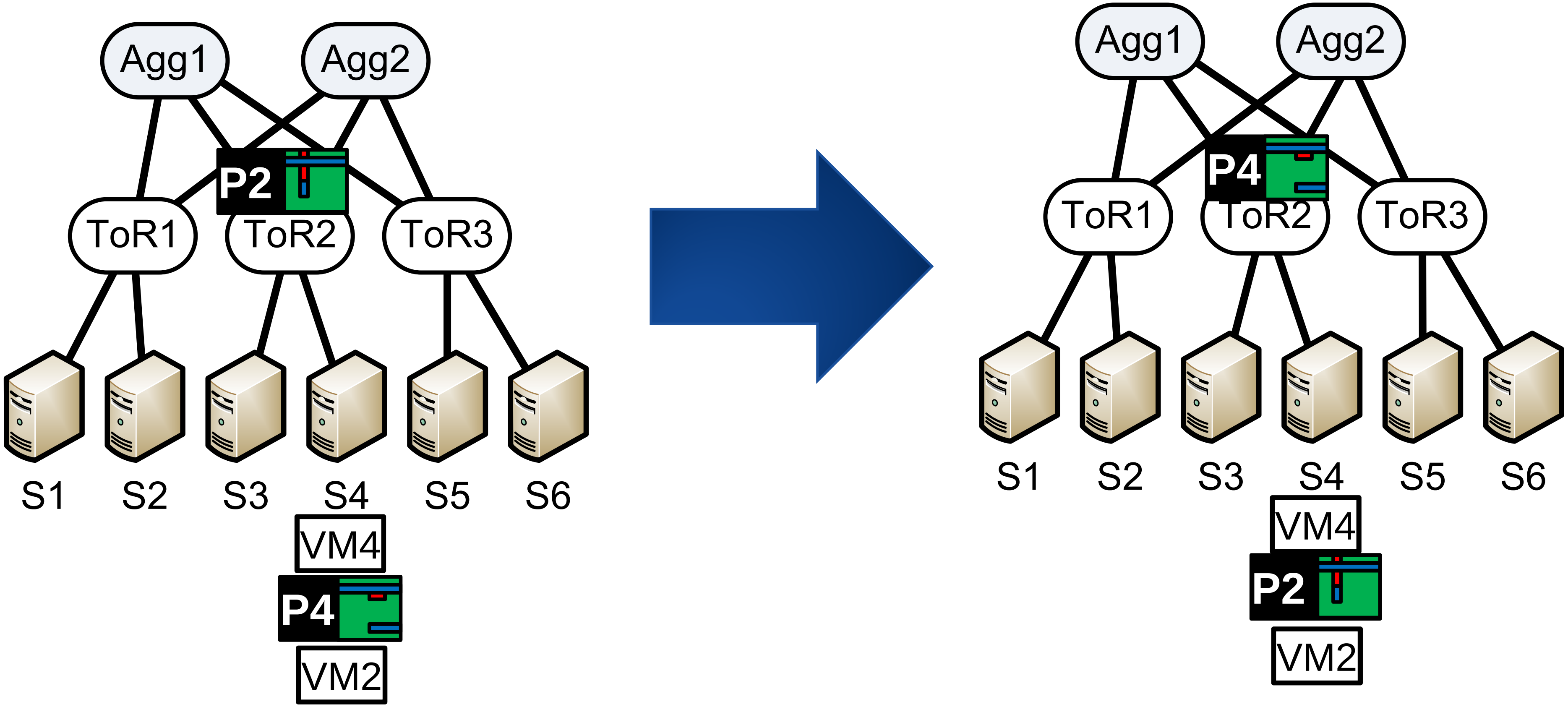
Traffic-aware refinement

- Overhead greedy approach
  1. Pick maximum overhead partition
  2. Put it where minimizes the overhead and maintains feasibility

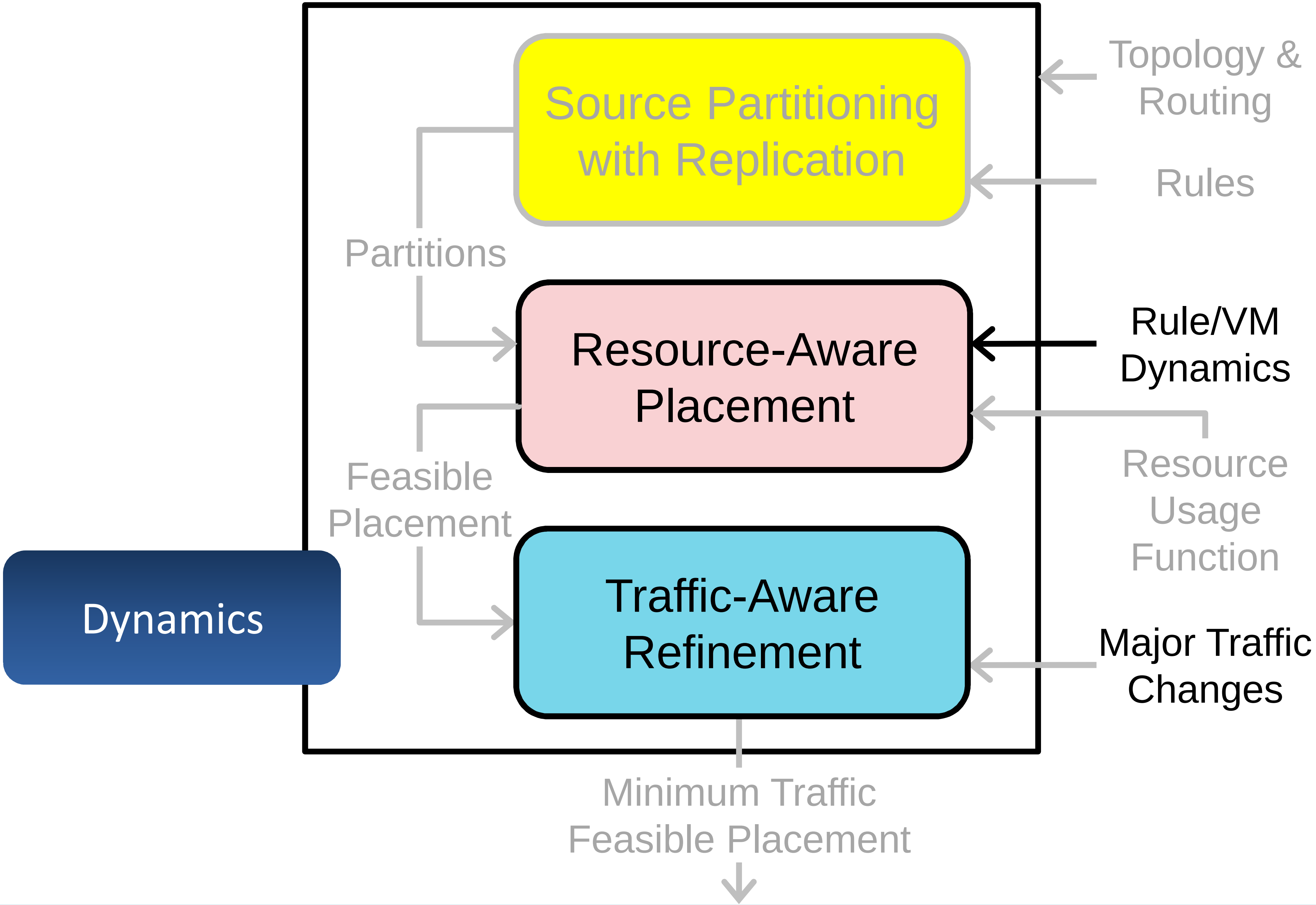


Traffic-aware refinement

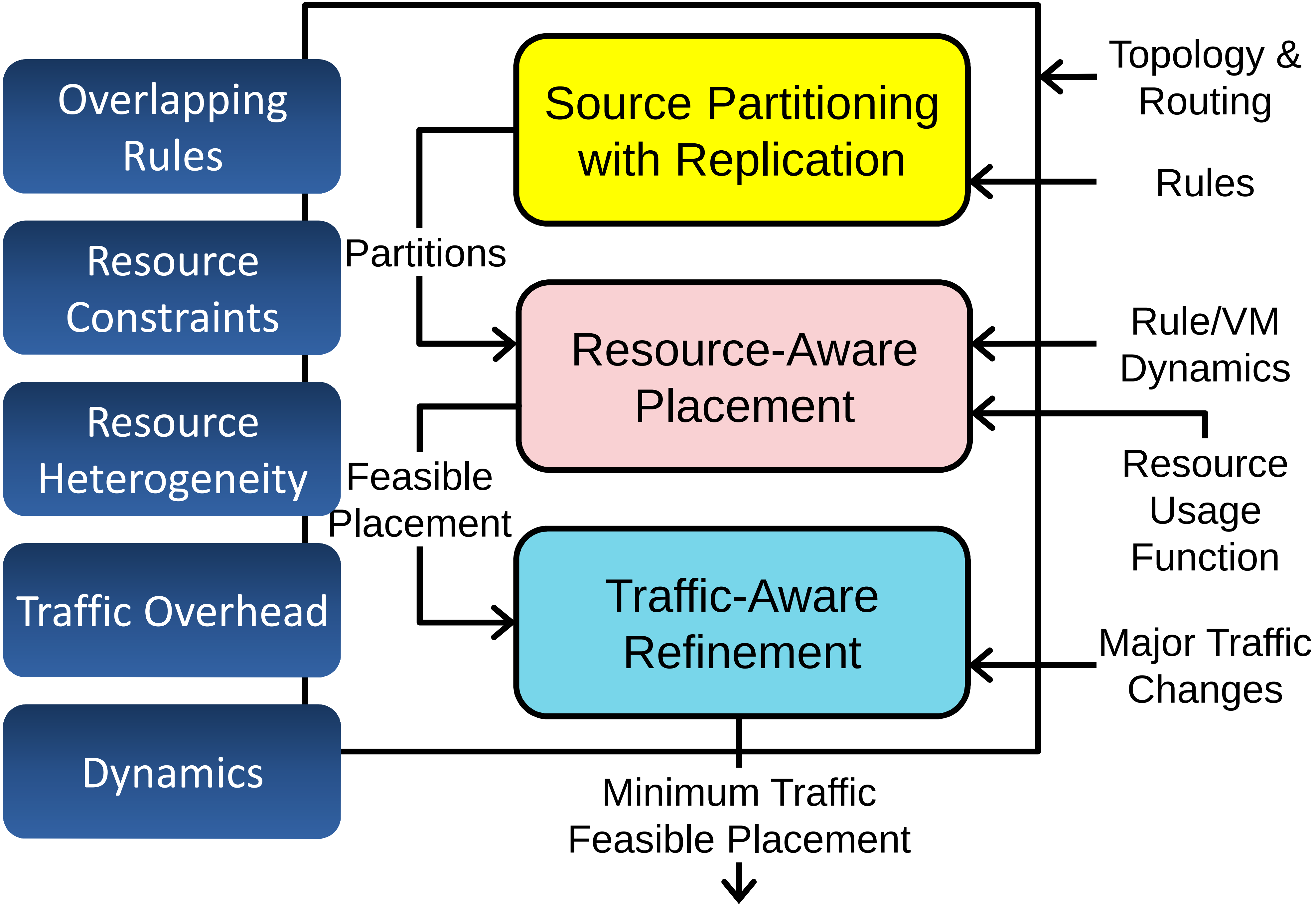
- Overhead greedy approach
  1. Pick maximum overhead partition
  2. Put it where minimizes the overhead and maintains feasibility
     - Problem: Local minima
- **Our approach: Benefit greedy**



vCRIB design: Dynamics



vCRIB design

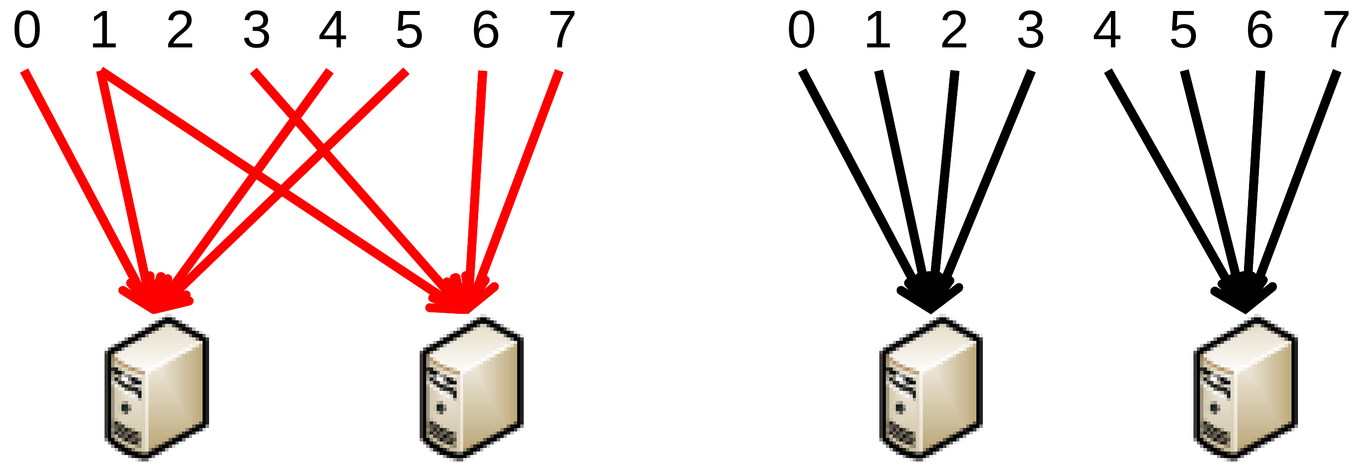


Evaluation

- Comparing vCRIB vs. Source-Placement
- Parameter sensitivity analysis
  - Rules in partitions
  - Traffic locality
  - VMs per server
  - Different memory sizes
- Where is the traffic overhead added?
- Traffic-aware refinement for online scenarios
- Heterogeneous resource constraints
- Switch-only scenarios

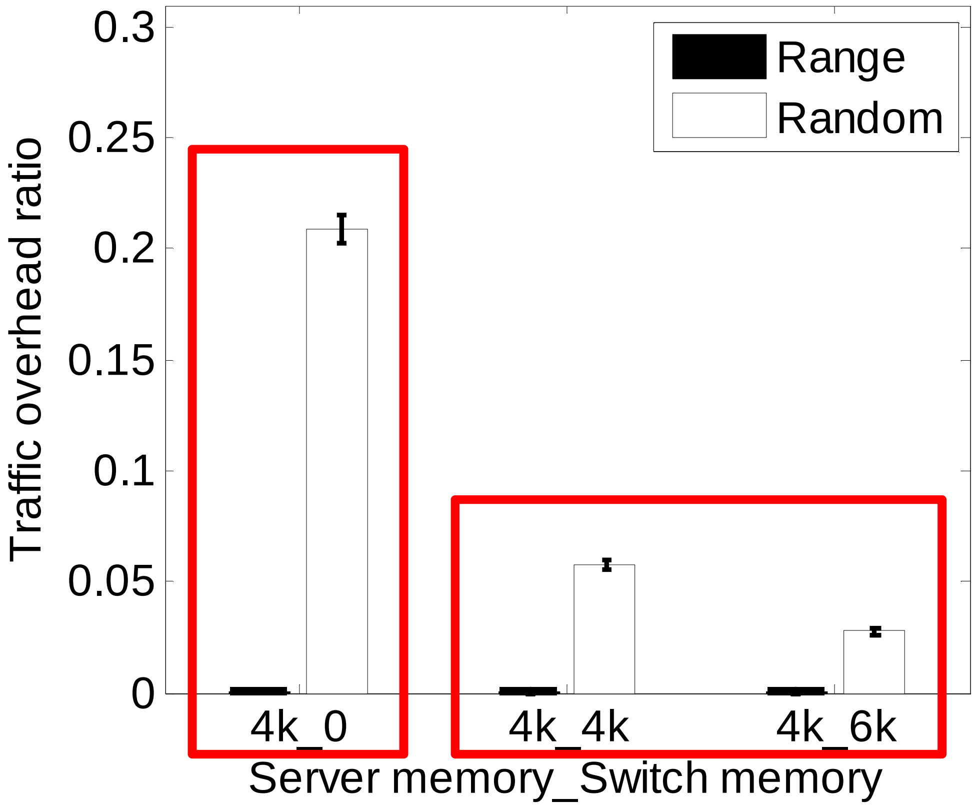
Simulation setup

- 1k servers with 20 VMs per server in a Fat-tree network
- 200k rules generated by ClassBench and random action
- IPs are assigned in two ways:
  - Random
  - Range



- Flows
  - Size follows long-tail distribution
  - Local traffic matrix (0.5 same rack, 0.3 same pod, 0.2 interpod)

Comparing vCRIB vs. Source-Placement



Maximum Load is 5K
Capacity is 4K

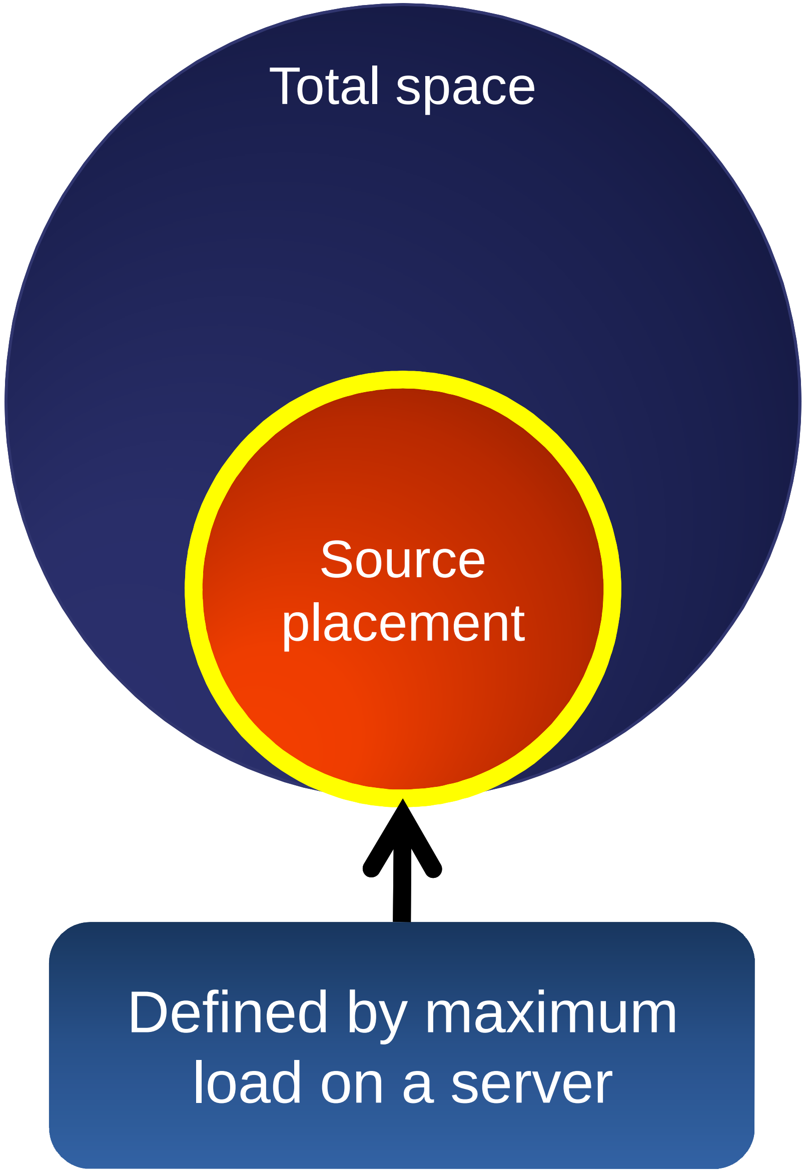
Random: Average load is 4.2K

vCRIB finds low traffic feasible solution

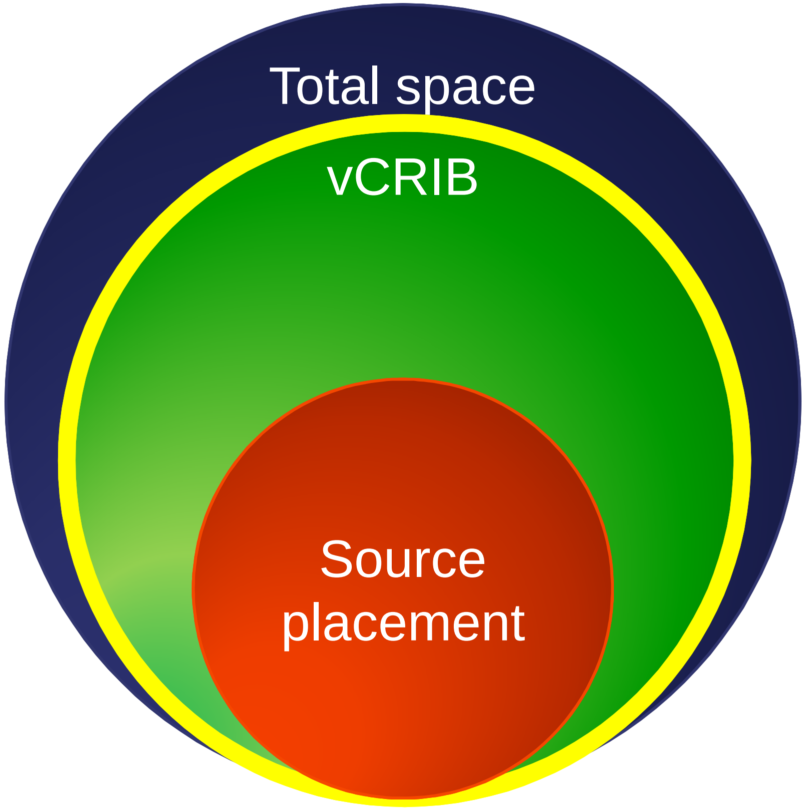
Range is better as similar partitions are from the same source

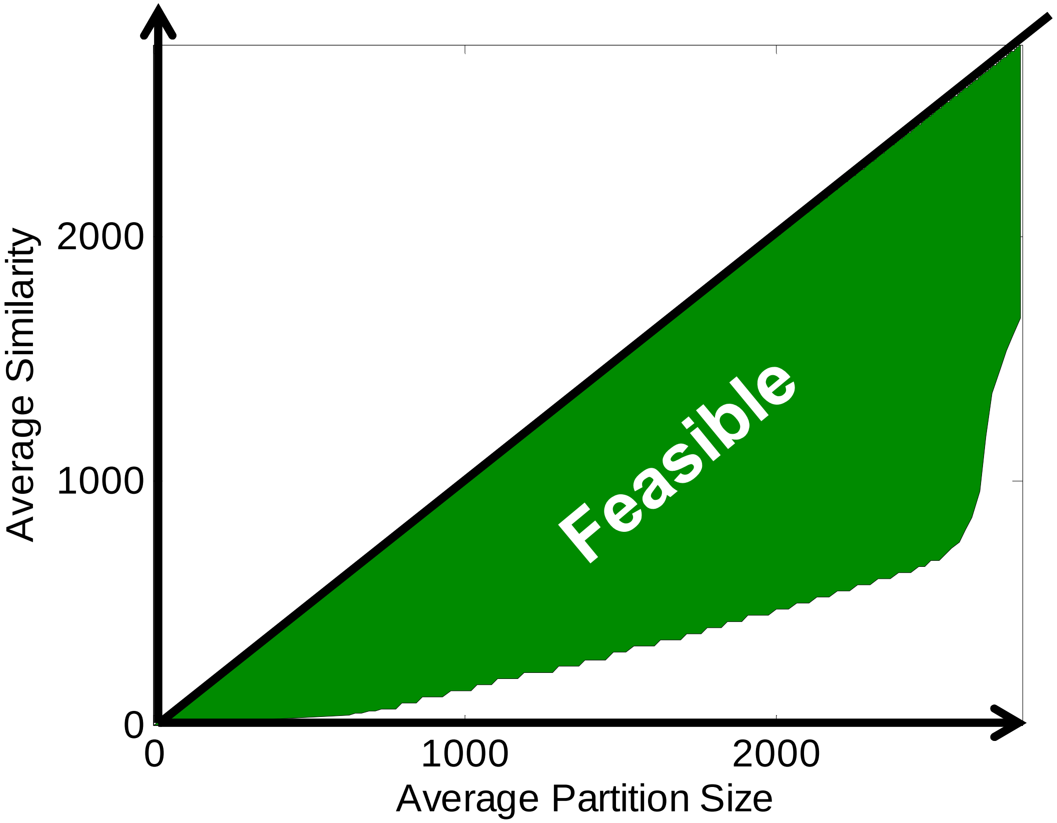
Adding more resources helps vCRIB reduce traffic overhead

Parameter sensitivity analysis: Rules in partitions

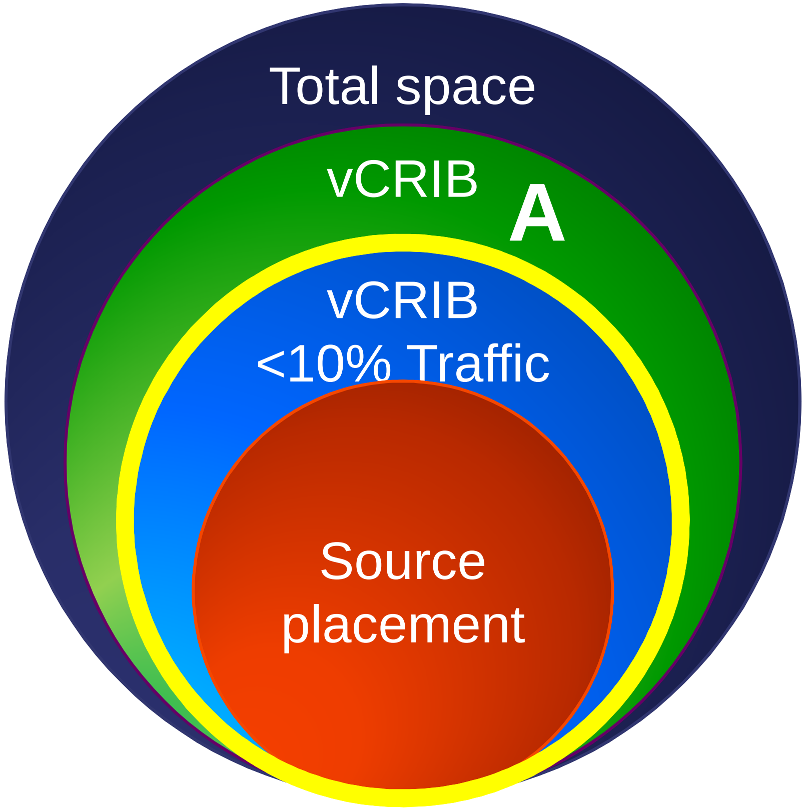


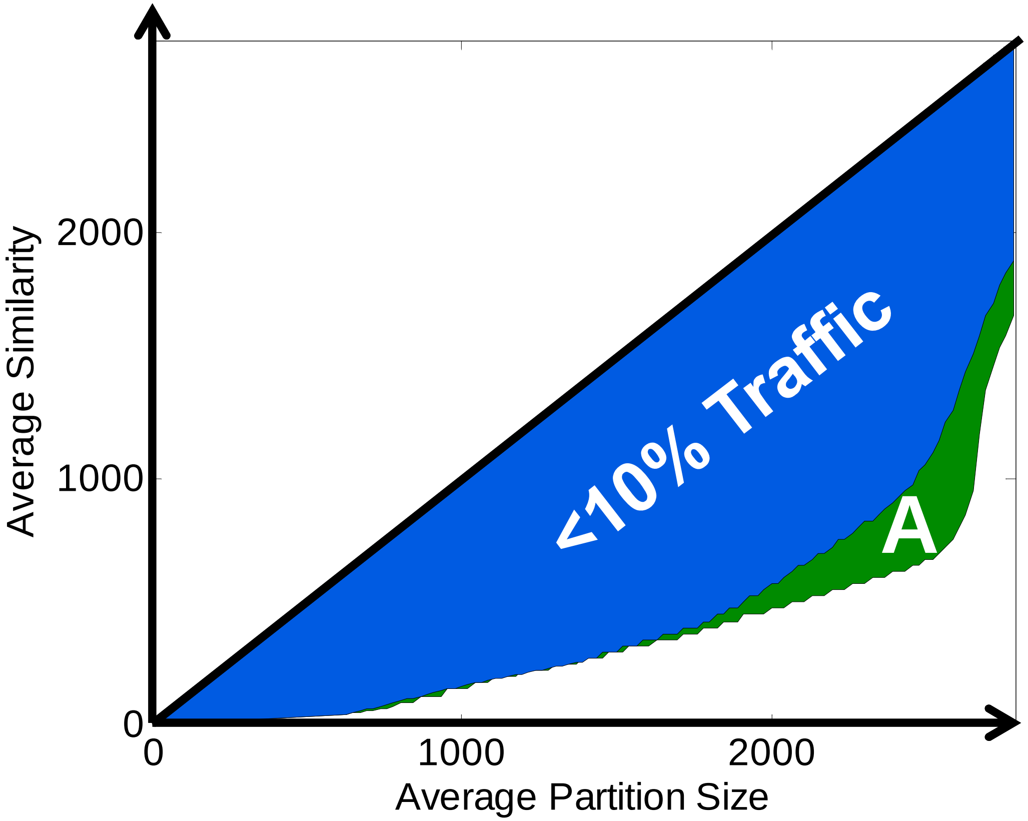
Parameter sensitivity analysis: Rules in partitions





Parameter sensitivity analysis: Rules in partitions





Lower traffic overhead for smaller partitions and more similar ones

Conclusion and future work

Conclusion

vCRIB allows operators and users to specify rules, and manages their placement in a way that respects resource constraints and minimizes traffic overhead.

Future work

- Support reactive placement by adding the controller in the loop
- Break a partition for large number of rules per VM
- Test for other rulesets

NSDI'13

Scalable Rule Management for Data Centers

Masoud Moshref, Minlan Yu,

Abhishek Sharma, Ramesh Govindan

4/3/2013

USC University of Southern California

NSL

NEC