

Data center resource management for in-network processing

Marco Micera

Politecnico di Torino, Technische Universität Darmstadt

Introduction

“Modern Internet services, such as search, social networking and e-commerce, critically depend on high-performance key-value stores. Rendering even a single web page often requires hundreds or even thousands of storage accesses.”

NetChain [6] authors

“As the number of compute elements grows, and the need to expose and utilize higher levels of parallelism grows, it is essential to [...] focus on developing architectures that lend themselves better to providing extreme-scale simulation capabilities.”

SHArP [3] authors

In-Network Processing (INP)

- INP refers to the technique of **offloading some parts of the computation to network devices** (e.g., programmable switches, network accelerators, middleboxes, etc.), hence reducing the load on servers
- Advantages:
 1. Serve network requests on the fly with low latency
 2. Reduce data center traffic and mitigate network congestion
 3. Save energy by running servers in a low-power mode
- Few solutions out there already: Daiet [10], SHArP [3], NetChain [6], IncBricks [8]

Thesis goals

Problem statement

For the time being, it seems that there is still no Resource Manager (RM) that takes into account the presence of a network having a data plane that supports (partially or completely) INP

Goals

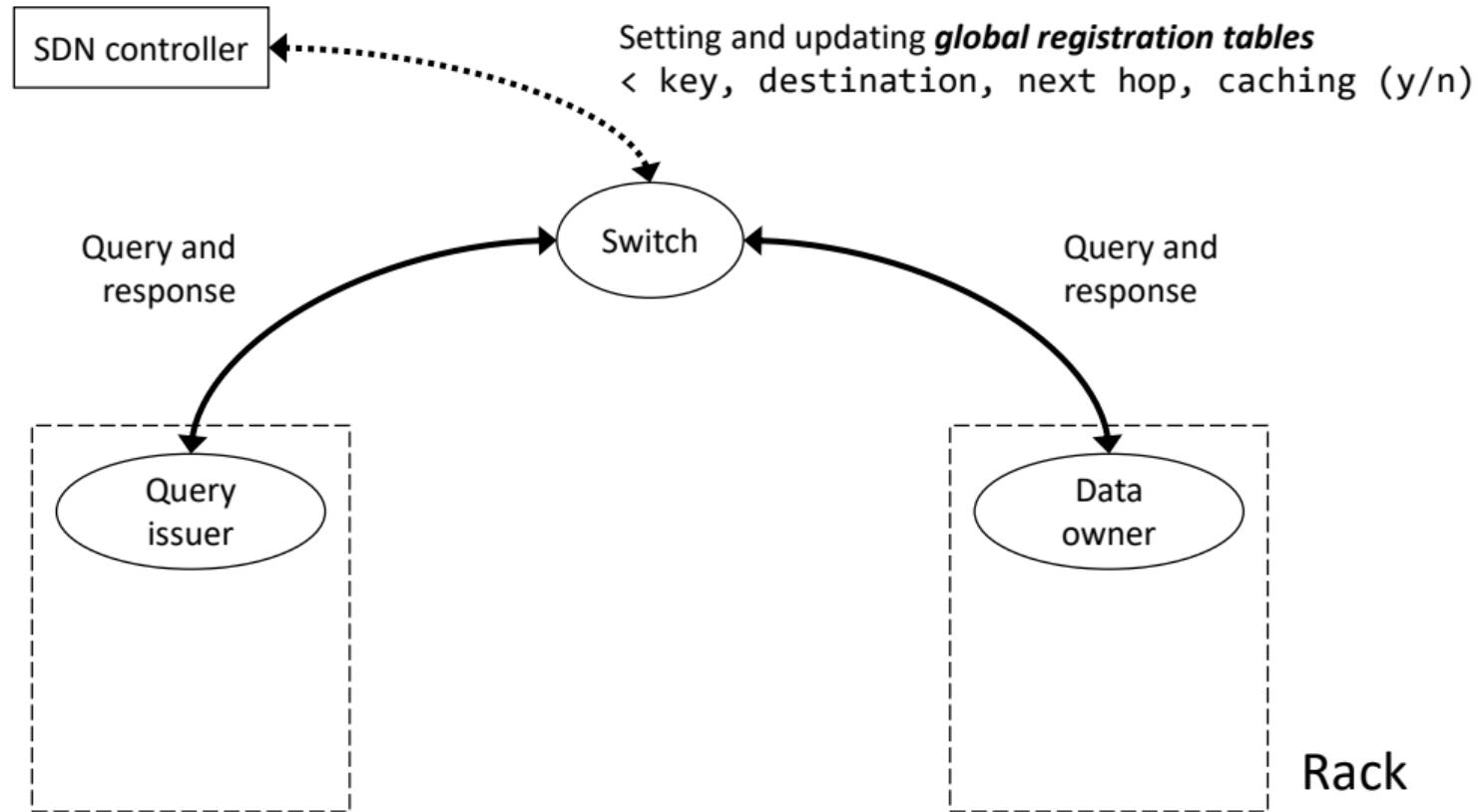
1. Model and evaluate an API through which applications can ask for INP resources
2. Discuss the importance of a scheduler which can reject INP requests and propose their server-only equivalent when needed (e.g., high switch utilization)

Analysis

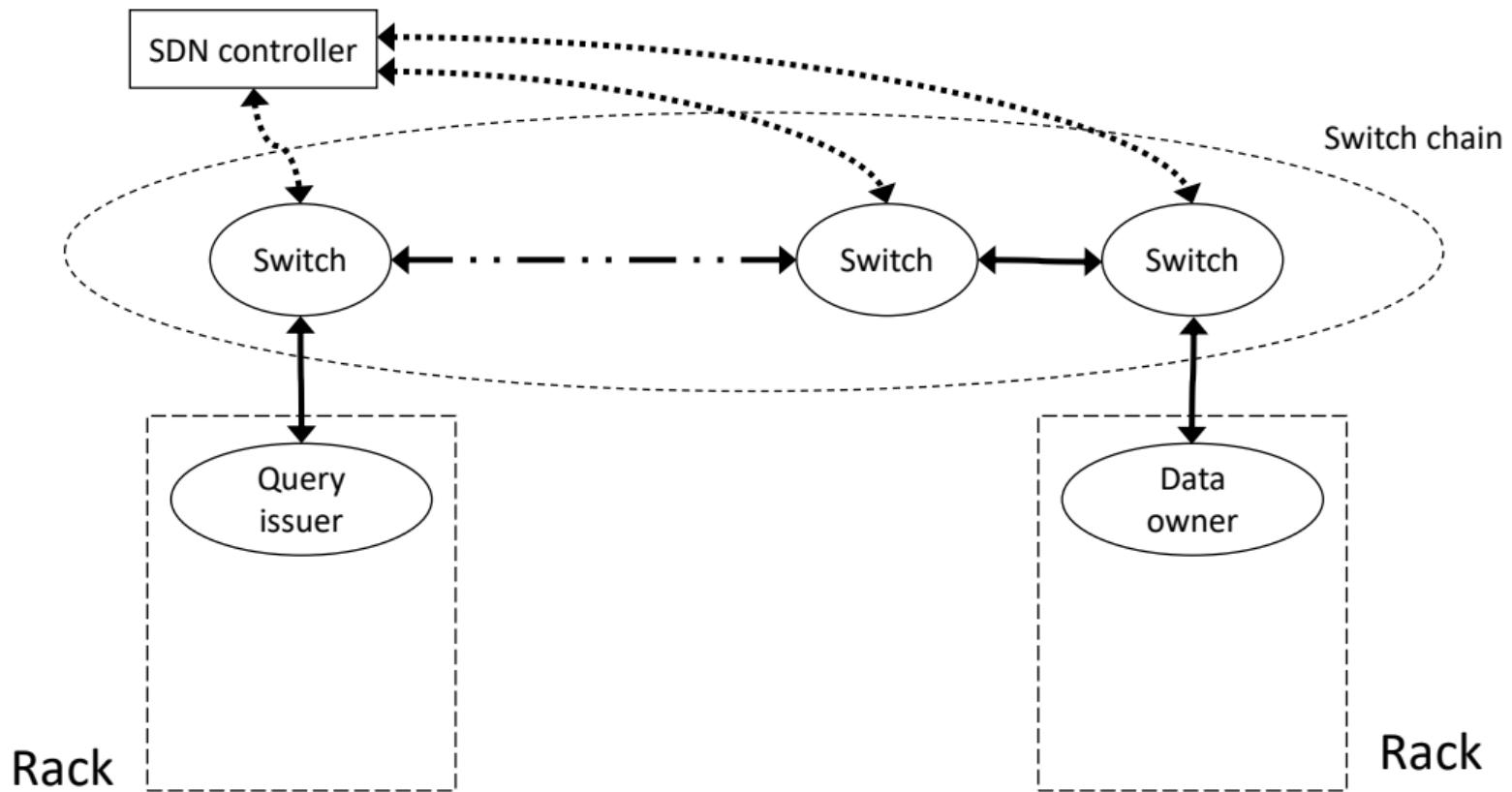
Currently existing In-Network Processing (INP) solutions

1. IncBricks (In-network caching system)
2. NetChain (Coordination services)
3. Daiet (In-network aggregation)
4. SHArP (Aggregation protocol)

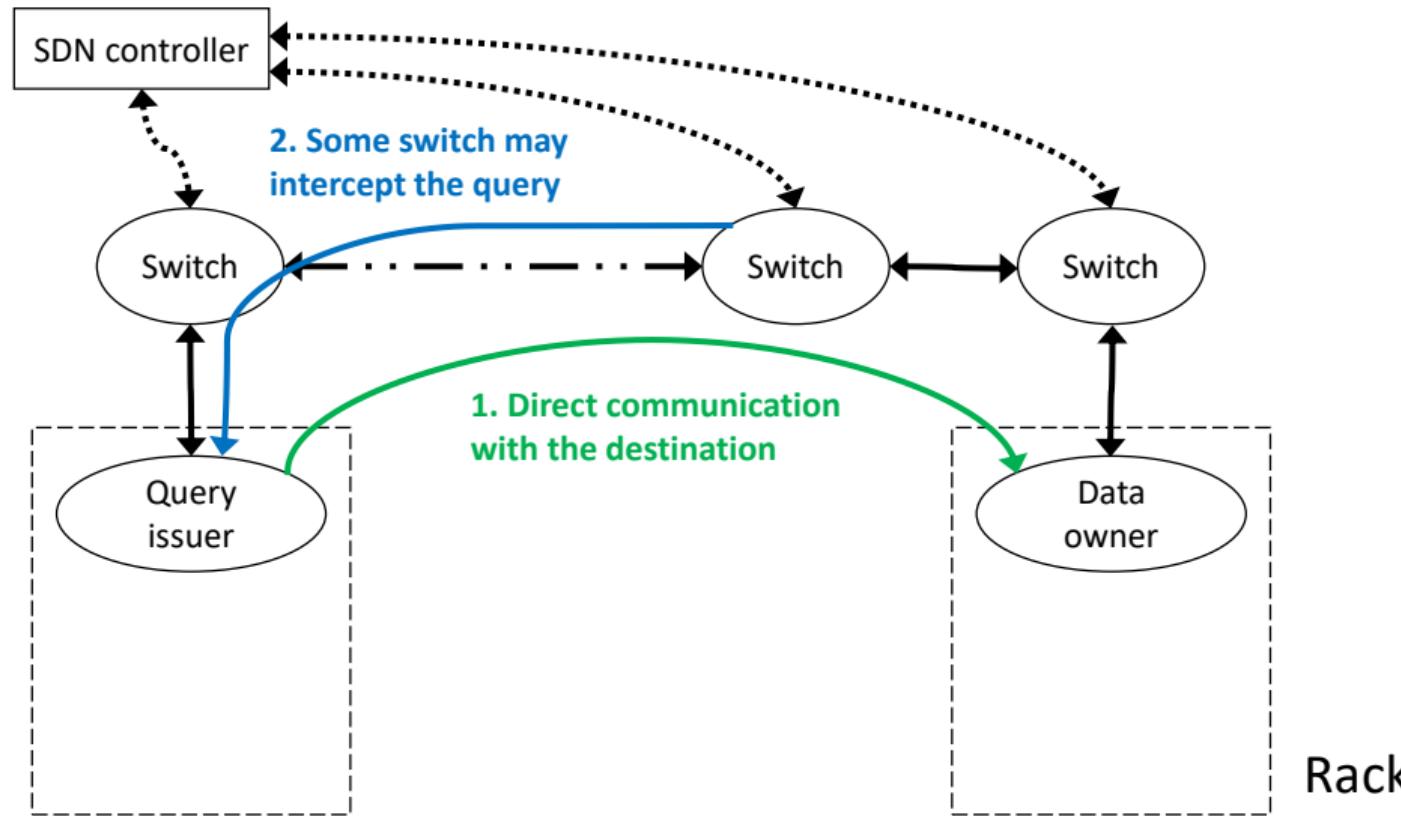
In-network caching system: IncBricks



In-network caching system: IncBricks



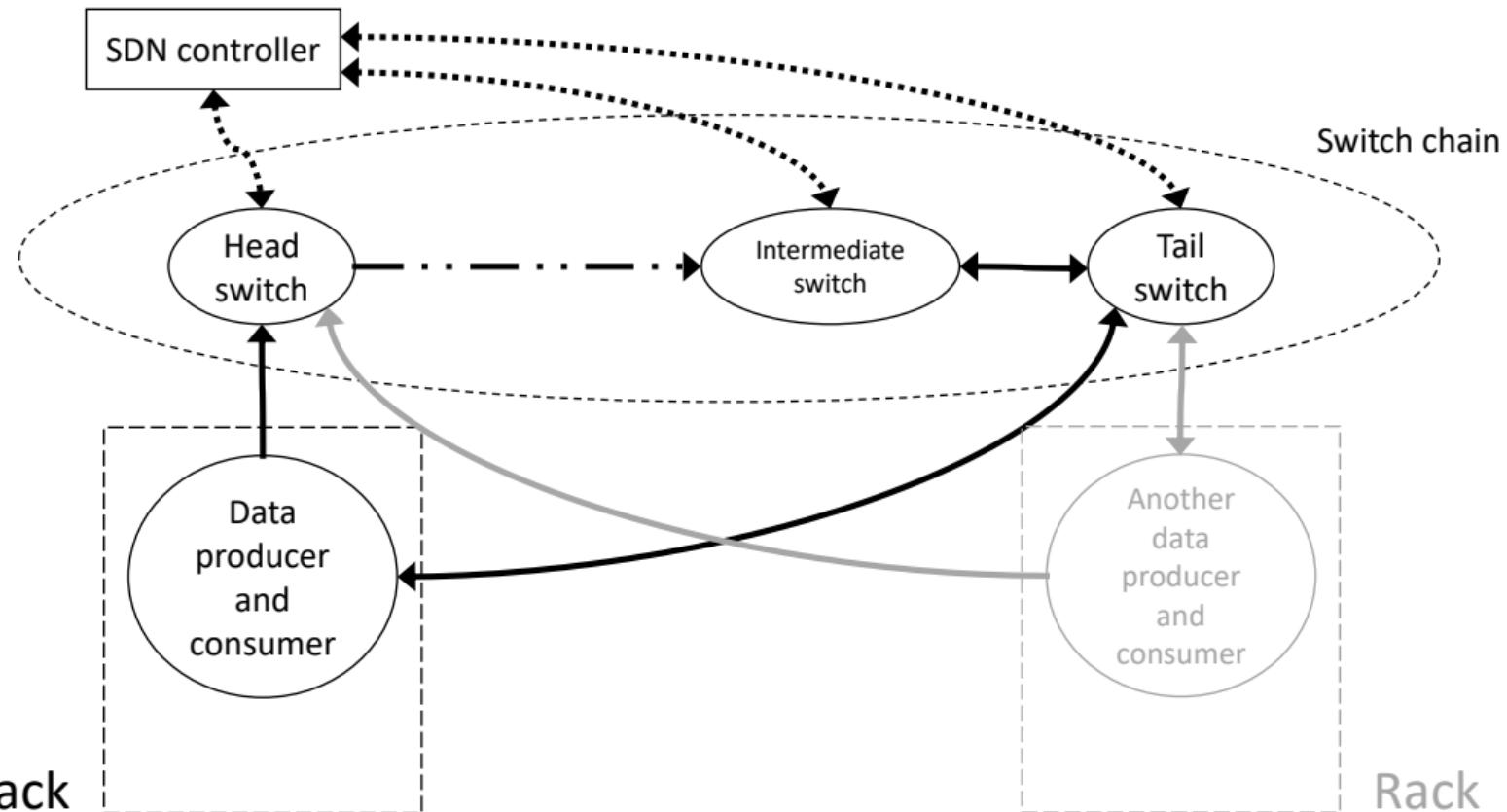
In-network caching system: IncBricks



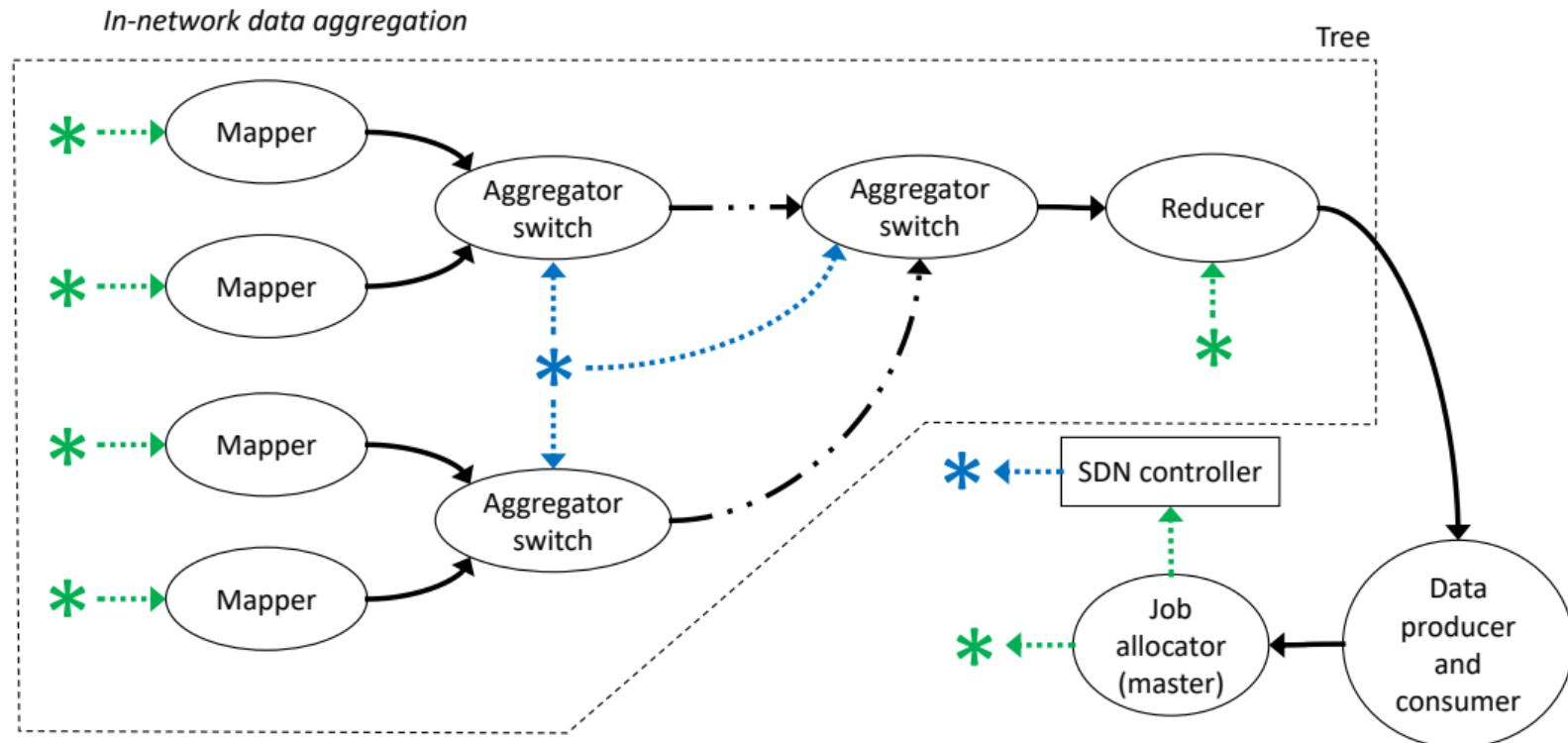
Rack

Rack

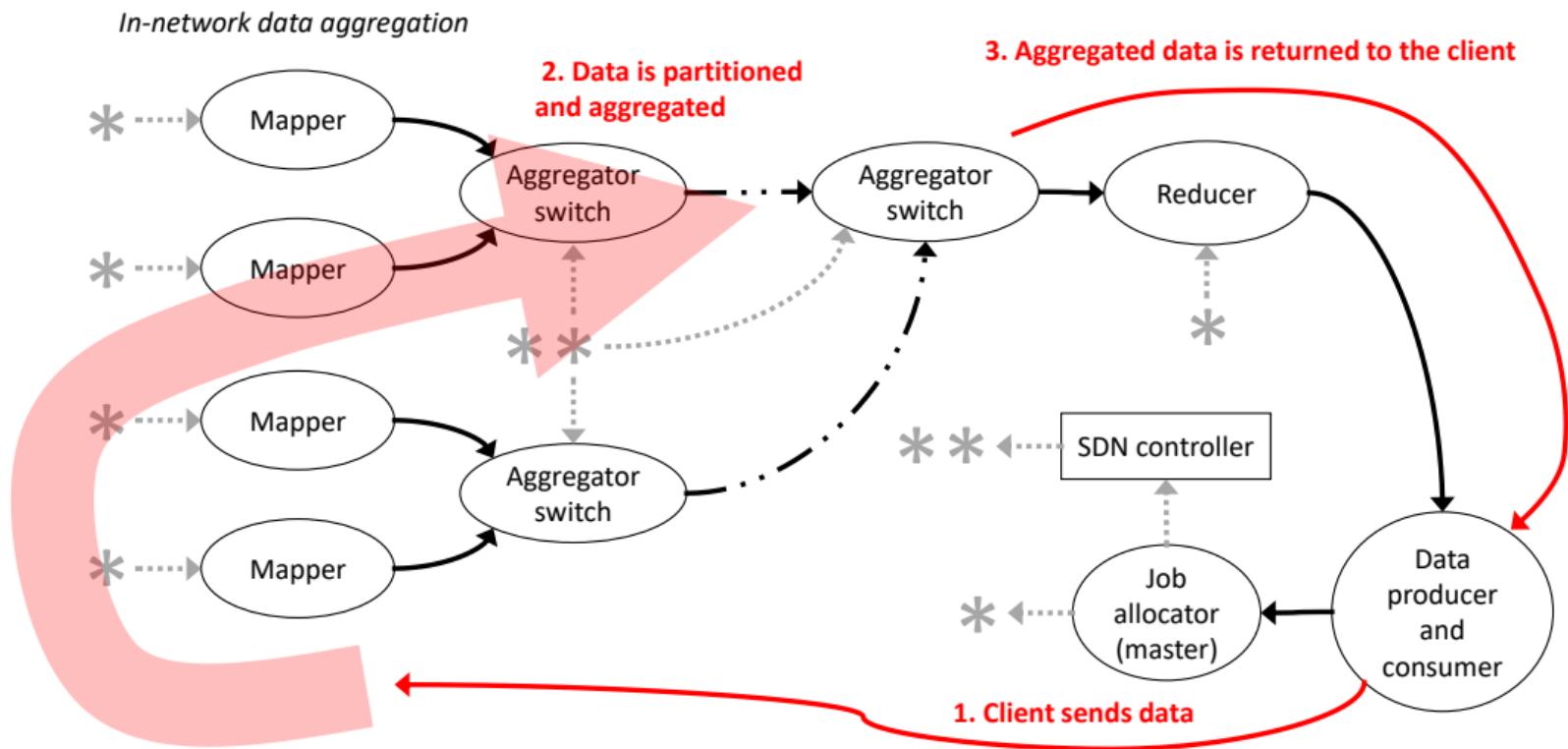
Coordination services: NetChain



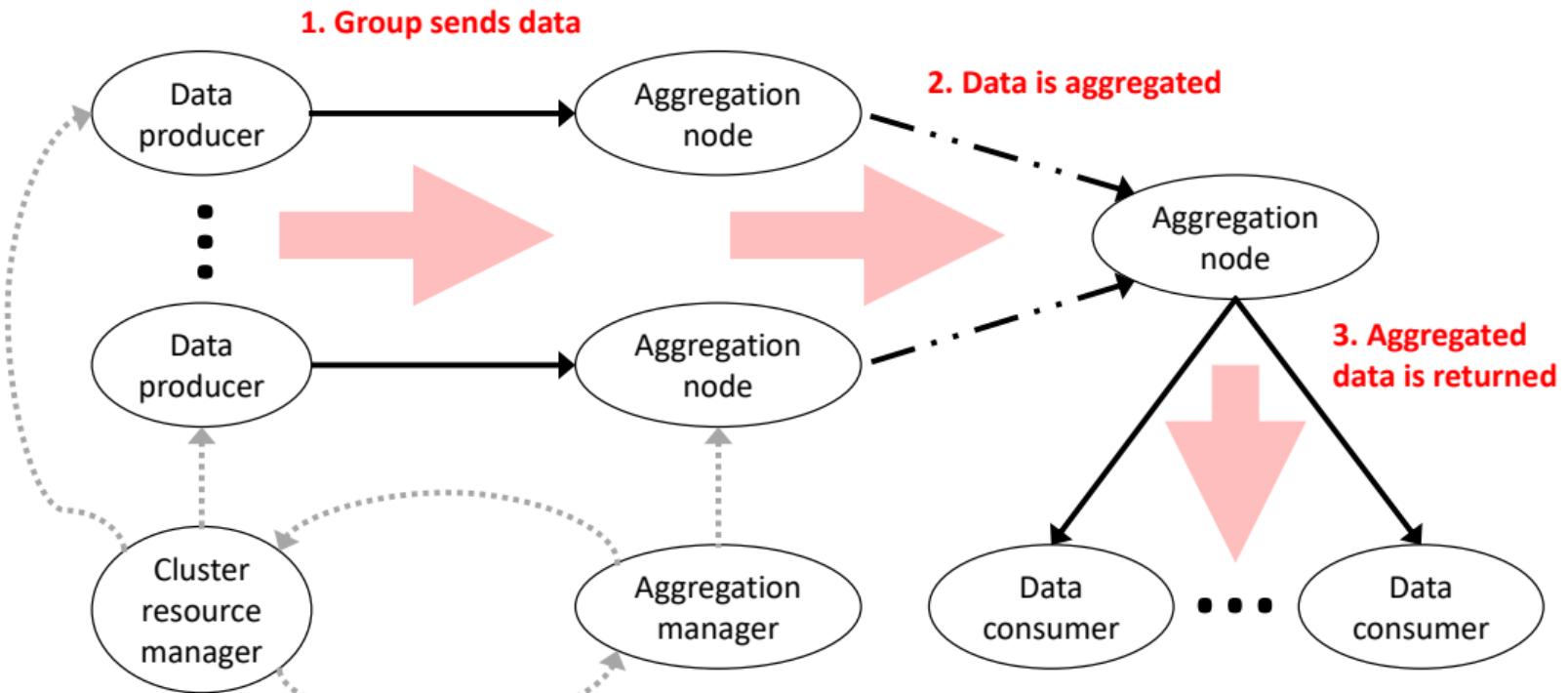
In-network aggregation: Daiet



In-network aggregation: Daiet

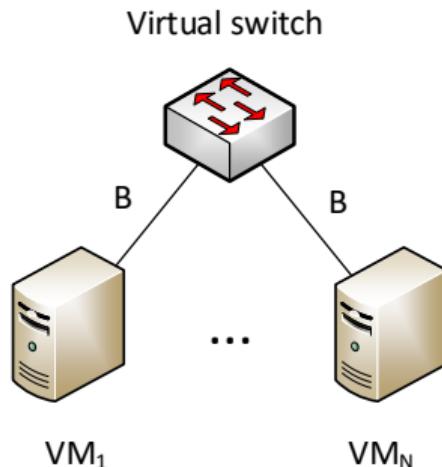


Aggregation protocol: SHArP



Resource models (3.4)

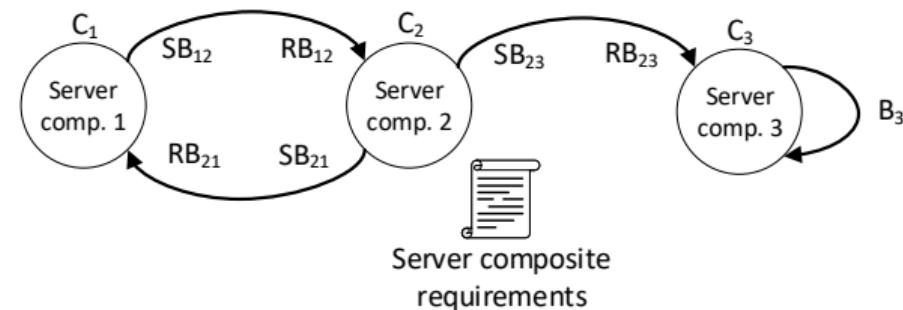
1. Virtual Cluster (VC)



2. Virtual Oversubscribed Cluster (VOC)

- N VCs connected to a single root virtual switch

3. Tenant Application Graph (TAG)



4. Fine-grained resource requests

- List of server-only resource demands

5. High-level goals

- E.g., job completion time (Bazaar [5])

Level of network awareness in RMs

- VMs proximity-aware
 - Spreading VMs across different failure domains (e.g., racks, power domains, etc.)
 - Omega [11], Apache™ YARN [12], Mesos [4], etc.
- Bandwidth-aware
 - Allowing tenants to specify bandwidth demands
 - "Virtual network" models (i.e., VCs, VOCs and TAGs)
 - CloudMirror [7], Oktopus [1], Kraken [2], Proteus [13], etc.
- Network resources-aware
 - At the time of writing, there seemed to be only one embedding solution¹ considering switch resources
 - The scheduler places server and switch resources in separate rounds

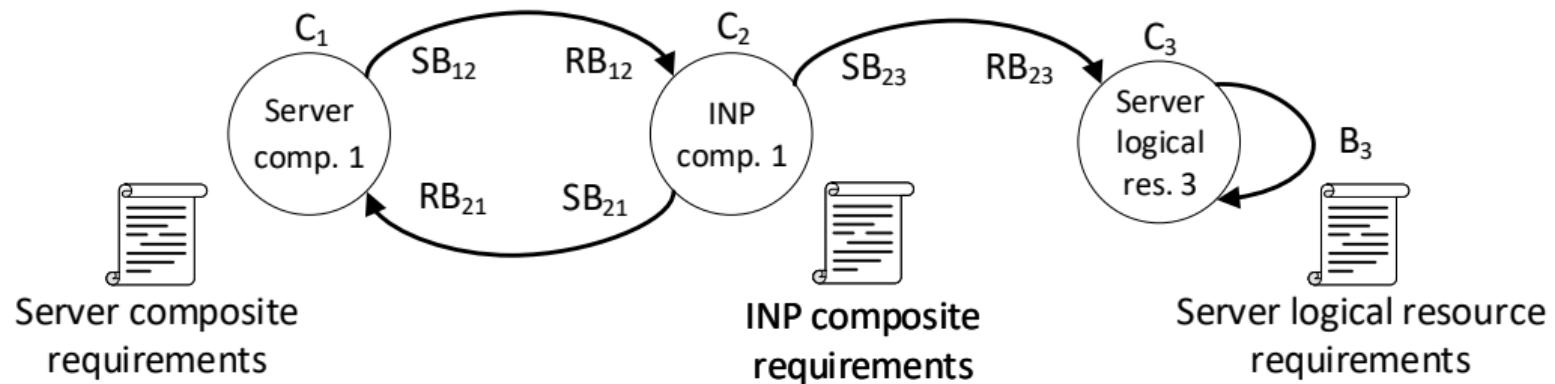
¹ Rabbani, Md Golam, et al. "On tackling virtual data center embedding problem." *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. IEEE, 2013. [9]

Design

Composites

- A composite is a template that describes high-level logical components
 - It can be of two types:
 - Server (e.g., “web server”, “database”, ...)
 - INP (e.g., “IncBricks caching system”, “NetChain locking system”, ...)
 - It can be made out of
 - Other Composites
 - A composite loop must not be valid, since it would be impossible to place
 - Logical resources

The extended-Tenant Application Graph (eTag)

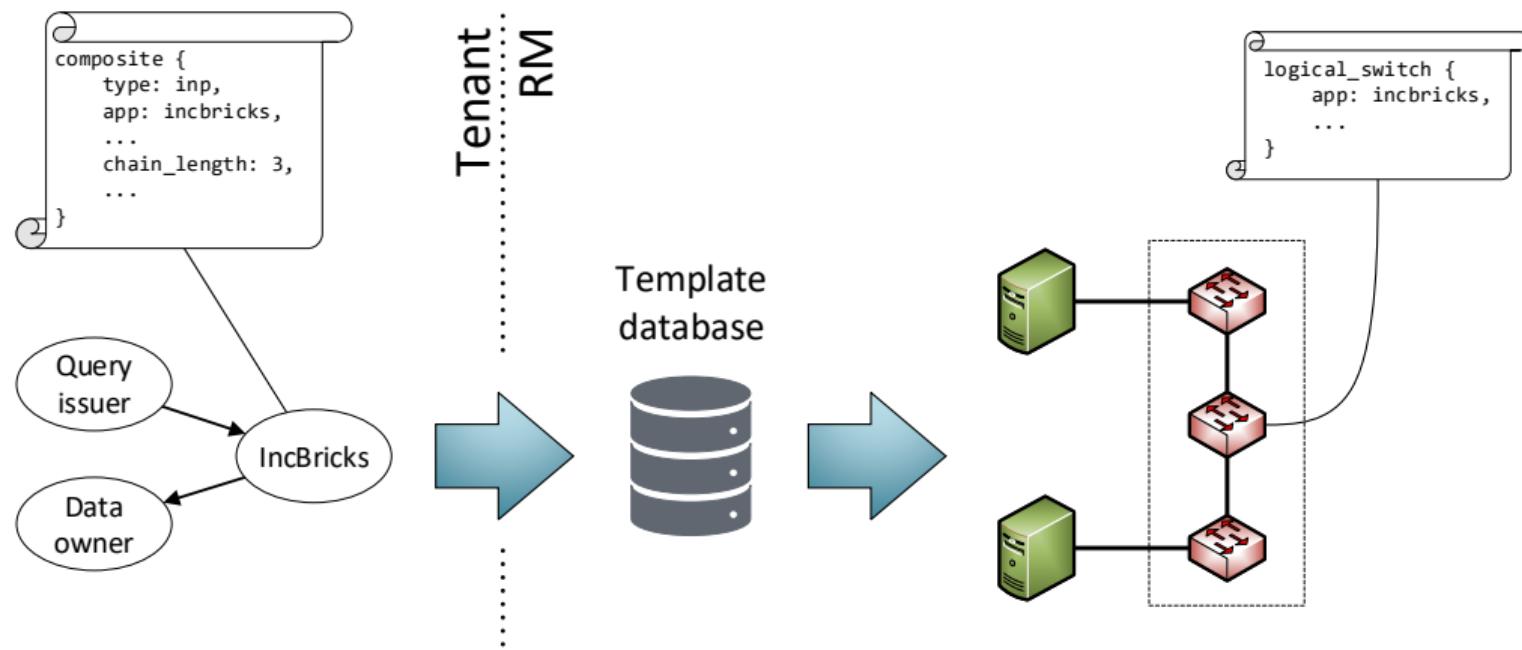


Generic groups

- In-network **storage**
 - Switches must
 - dedicate part of their local memory to store a distributed map
 - form a chain
 - IncBricks [8], NetChain [6]
- In-network **data aggregation**
 - Switches must
 - form a tree whose root is connected to data consumers and whose leaves are connected to data producers
 - dedicate part of their local memory to store a key-value map
 - be able to perform basic operations on data, such as writing and hashing
 - wait for all its children to send aggregated data
 - Daiet [10], SHArP [3]

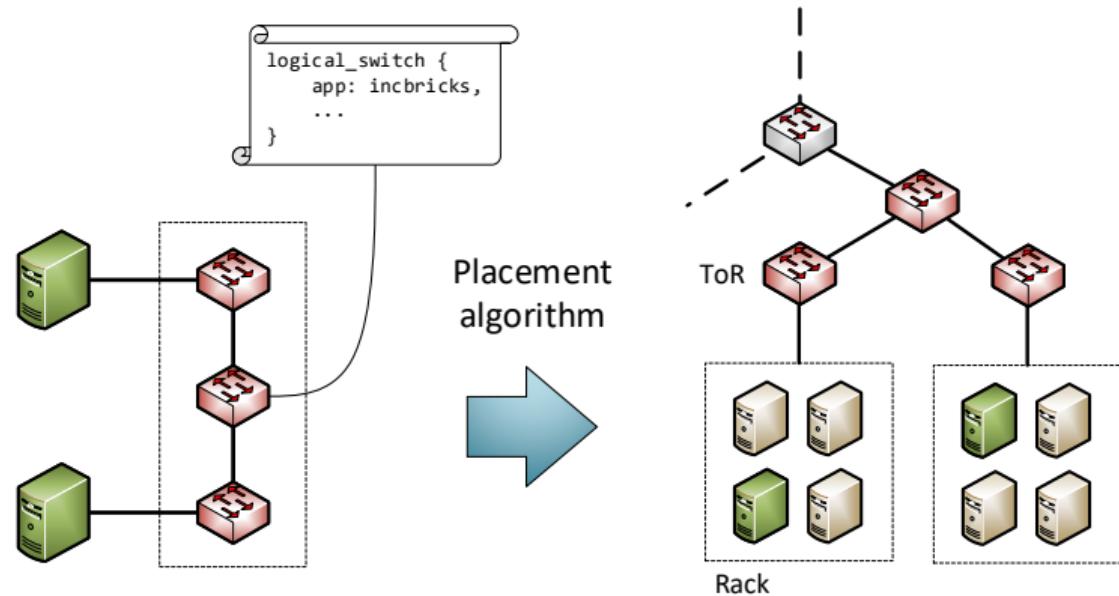
Mapping composites to logical resources

- The *template database* maps composites (or generic groups) to their equivalent made out of just logical resources

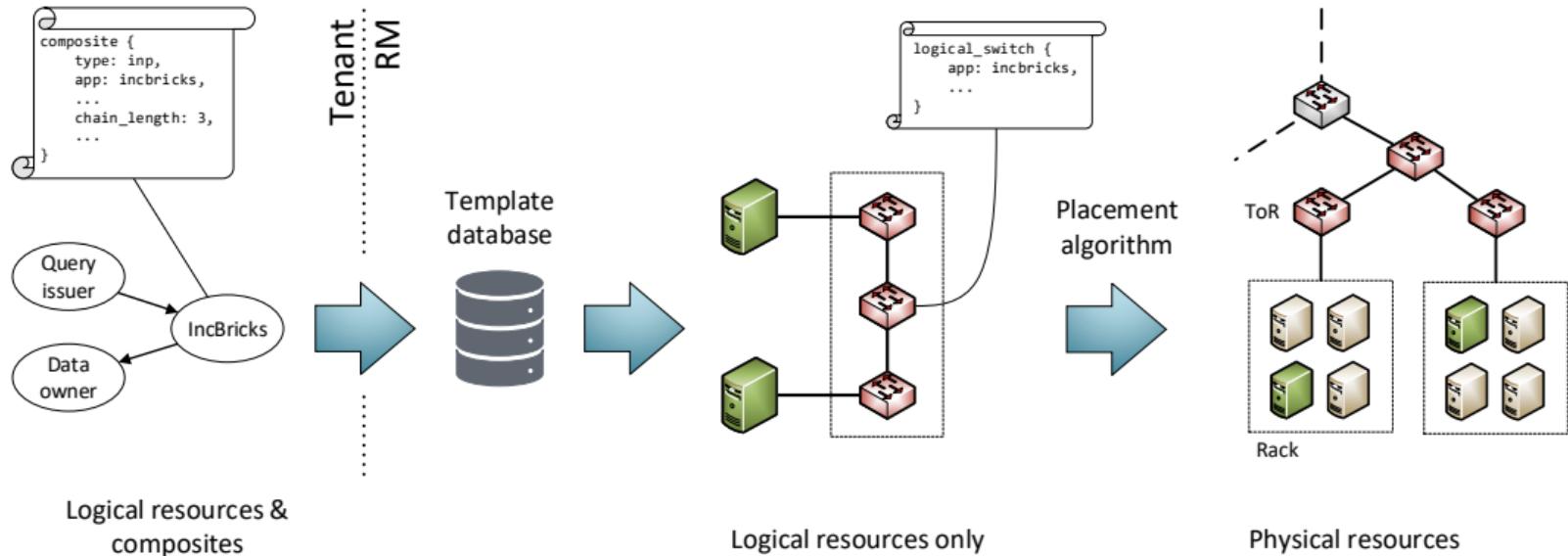


Placing logical resources

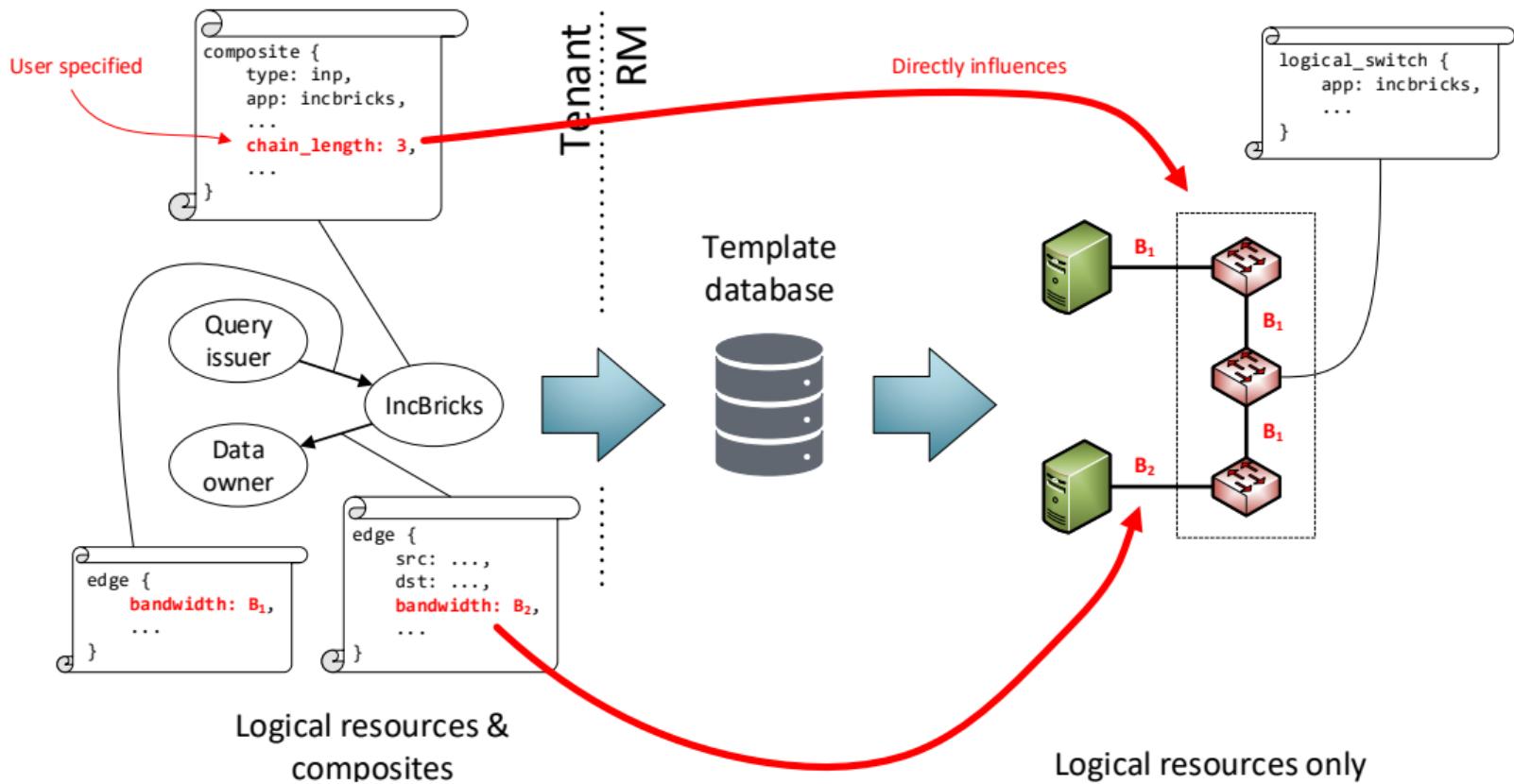
- The placement algorithm places logical resource onto physical ones
- Three types of physical resources:
 - Server
 - Switch
 - Link



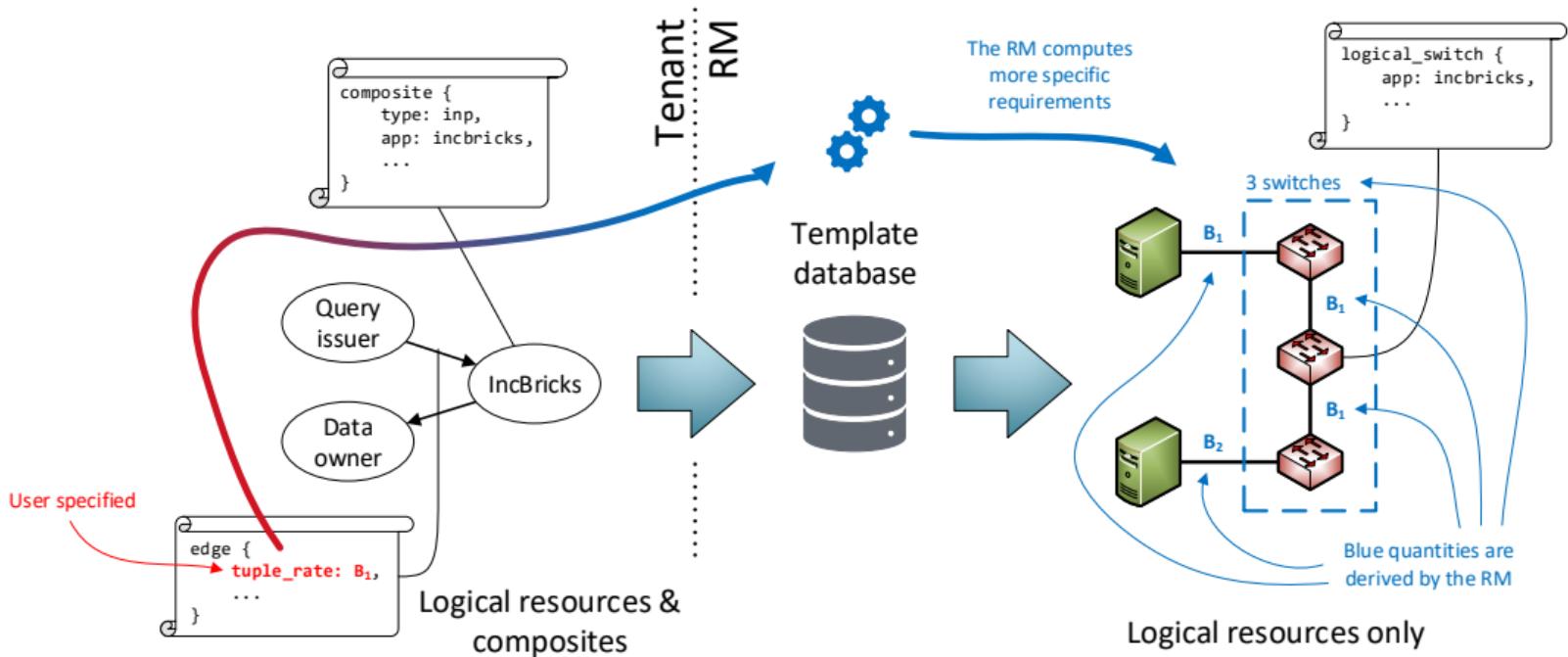
The whole picture



1st approach: passive template mapping



2nd approach: active template mapping



Evaluation

Simulation 1/2

- Simulator built from the ground up
 - Inspired by Omega's [11] *lightweight simulator*²
 - Supports multiple resource dimensions, switch resources, and composites
 - Switches have *properties* (e.g., supported INP solutions)
- Simulated data center physical architecture: fat-tree with 4 pods
- 3 days-long randomly-generated workload
 - Job properties (e.g., requirements, requests' interarrival time, etc.) are sampled from exponential distributions
- Simple greedy scheduler

²Available at github.com/google/cluster-scheduler-simulator

Simulation 2/2

- The template database contains two entries for the previously-mentioned generic groups
 - In-network storage (switch chain)
 - In-network data aggregation (switch tree)
- Server Tasks Cutback (STC): the reduction of server tasks once an INP solution is introduced

$$STC = \frac{\# \text{server tasks without INP}}{\# \text{server tasks with INP}} \quad (1)$$

- Sweep: percentage of requests including INP composites

Results 1/3

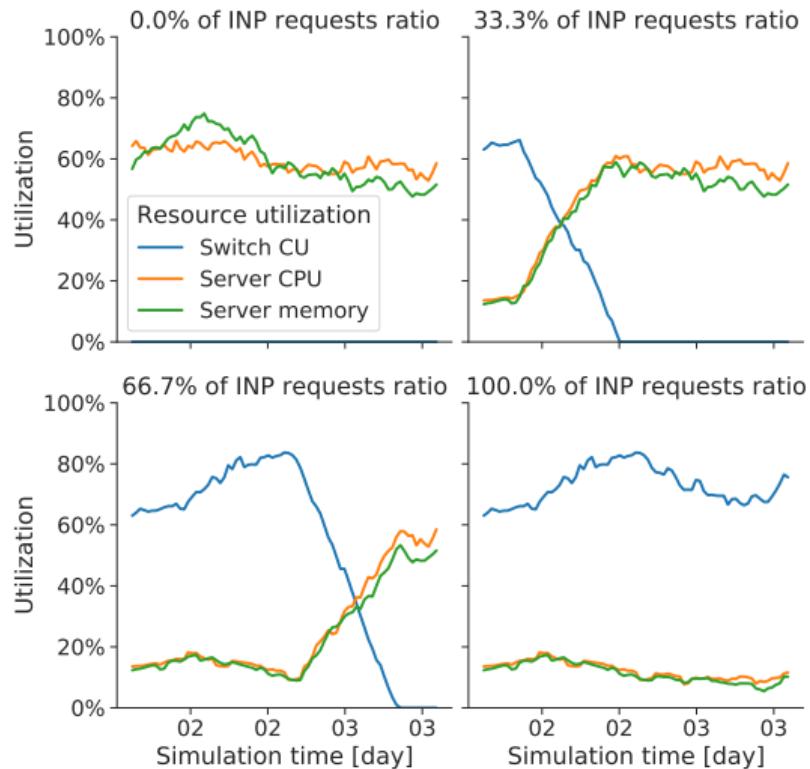


Figure 1: Physical resource utilization for different amounts of INP requests

Results 2/3

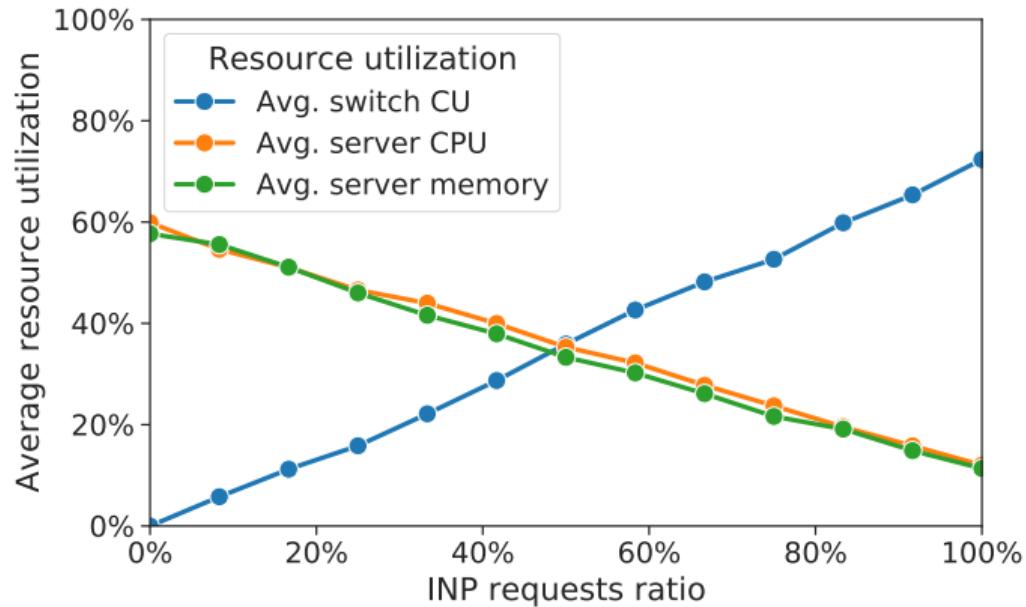


Figure 2: Average resource utilization as a function of the INP requests ratio

Results 3/3

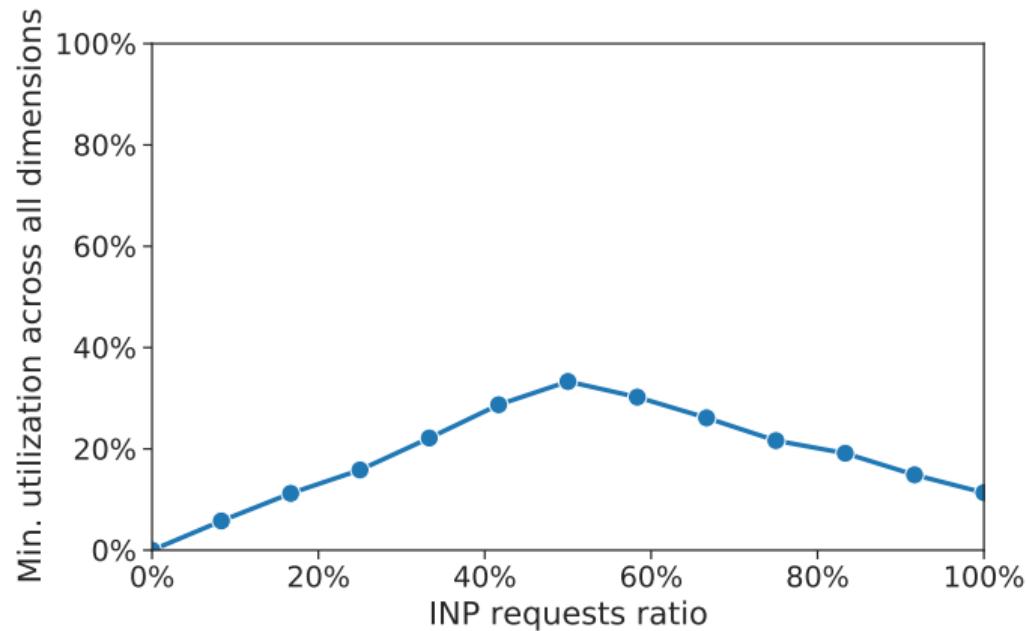


Figure 3: Minimum resource utilization across all dimensions

Conclusions

Questions?

Thank you

Bibliography i

-  H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron.
Towards predictable datacenter networks.
In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 242–253, New York, NY, USA, 2011. ACM.
-  C. Fuerst, S. Schmid, L. Suresh, and P. Costa.
Kraken: Online and elastic resource reservations for multi-tenant datacenters.
In *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, IEEE, pages 1–9. IEEE, 2016.

Bibliography ii

-  R. L. Graham, D. Bureddy, P. Lui, H. Rosenstock, G. Shainer, G. Bloch, D. Goldenerg, M. Dubman, S. Kotchubievsky, V. Koushnir, et al.
Scalable hierarchical aggregation protocol (sharp): a hardware architecture for efficient data reduction.
In *Proceedings of the First Workshop on Optimization of Communication in HPC*, pages 1–10. IEEE Press, 2016.
-  B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. H. Katz, S. Shenker, and I. Stoica.
Mesos: A platform for fine-grained resource sharing in the data center.
In *NSDI*, volume 11, pages 22–22, 2011.

-  V. Jalaparti, H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron.
Bridging the tenant-provider gap in cloud services.
In *Proceedings of the Third ACM Symposium on Cloud Computing*, SoCC '12, pages 10:1–10:14, New York, NY, USA, 2012. ACM.
-  X. Jin, X. Li, H. Zhang, N. Foster, J. Lee, R. Soulé, C. Kim, and I. Stoica.
Netchain: Scale-free sub-rtt coordination.
In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 35–49, Renton, WA, 2018. USENIX Association.

-  J. Lee, Y. Turner, M. Lee, L. Popa, S. Banerjee, J.-M. Kang, and P. Sharma.
Application-driven bandwidth guarantees in datacenters.
In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 467–478, New York, NY, USA, 2014. ACM.
-  M. Liu, L. Luo, J. Nelson, L. Ceze, A. Krishnamurthy, and K. Atreya.
Incbricks: Toward in-network computation with an in-network cache.
In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '17, pages 795–809, New York, NY, USA, 2017. ACM.

-  M. G. Rabbani, R. Esteves, M. Podlesny, G. Simon, L. Z. Granville, and R. Boutaba.

On tackling virtual data center embedding problem.

In *IM 2013: IFIP/IEEE International Symposium on Integrated Network Management*, 2013.

-  A. Sapiro, I. Abdelaziz, A. Aldilaijan, M. Canini, and P. Kalnis.

In-network computation is a dumb idea whose time has come.

In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, HotNets-XVI, pages 150–156, New York, NY, USA, 2017. ACM.

-  M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, and J. Wilkes.
Omega: flexible, scalable schedulers for large compute clusters.
In *SIGOPS European Conference on Computer Systems (EuroSys)*, pages 351–364, Prague, Czech Republic, 2013.
-  V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O’Malley, S. Radia, B. Reed, and E. Baldeschwieler.
Apache hadoop yarn: Yet another resource negotiator.
In *Proceedings of the 4th Annual Symposium on Cloud Computing, SOCC ’13*, pages 5:1–5:16, New York, NY, USA, 2013. ACM.

-  D. Xie, N. Ding, Y. C. Hu, and R. Kompella.
The only constant is change: incorporating time-varying network reservations in data centers.
ACM SIGCOMM Computer Communication Review, 42(4):199–210, 2012.