# Data center resource management for in-network processing

Marco Micera

March 31, 2020

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 [2] 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 [1] authors

# In-Network Processing (INP)

- INP refers to the technique of **offloading 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 [4], SHArP [1], NetChain [2], IncBricks [3]

# Thesis goals

**Problem statement**

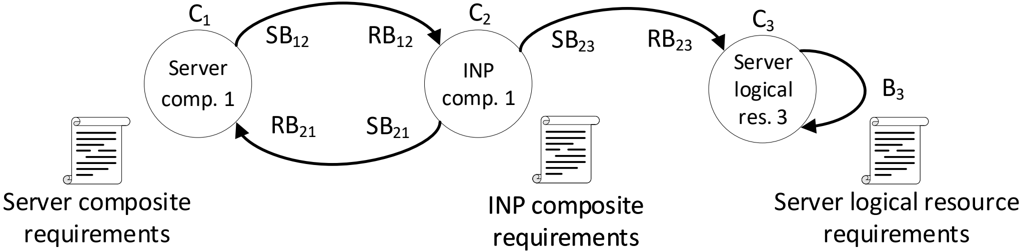
For the time being, it seems that there is still no Resource Manager (RM) that takes into account the presence of a network having a data plane that supports (partially or completely) INP

**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)

# Design

# The Extended-Tenant Application Graph (eTAG)



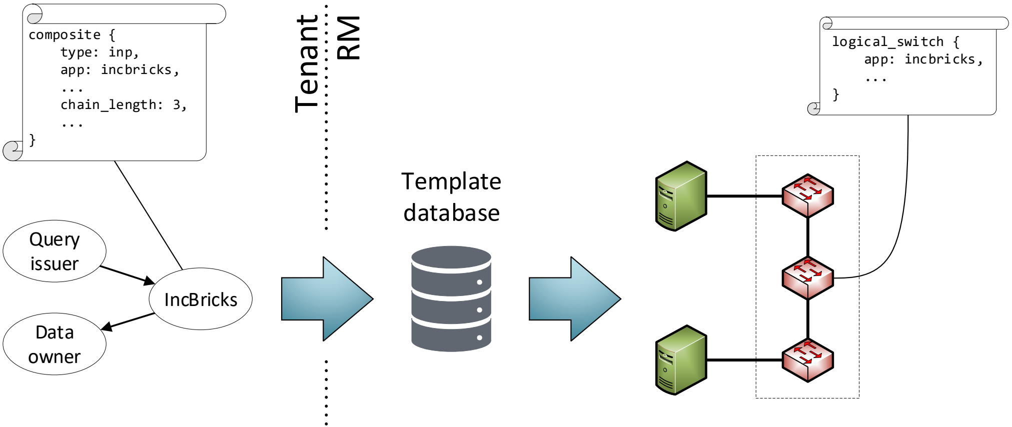

- A composite is a template that describes high-level logical components
  - It can be of two types:
    - Server (e.g., "web server", "database", ...)
    - INP (e.g., "IncBricks caching system", "NetChain locking system", ...)
  - It can be made out of
    - Other Composites
    - Logical resources

# Generic groups

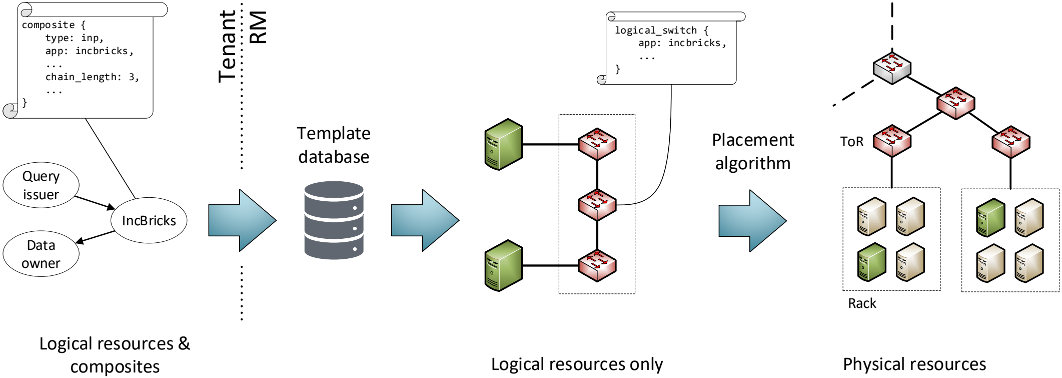
- In-network **storage**
  - Switches must
    - dedicate part of their local memory to store a distributed map
    - form a chain
  - IncBricks [3], NetChain [2]
- In-network **data aggregation**
  - Switches must
    - form a tree whose root is connected to data consumers and whose leaves are connected 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 [4], SHArP [1]

# Mapping composites to logical resources

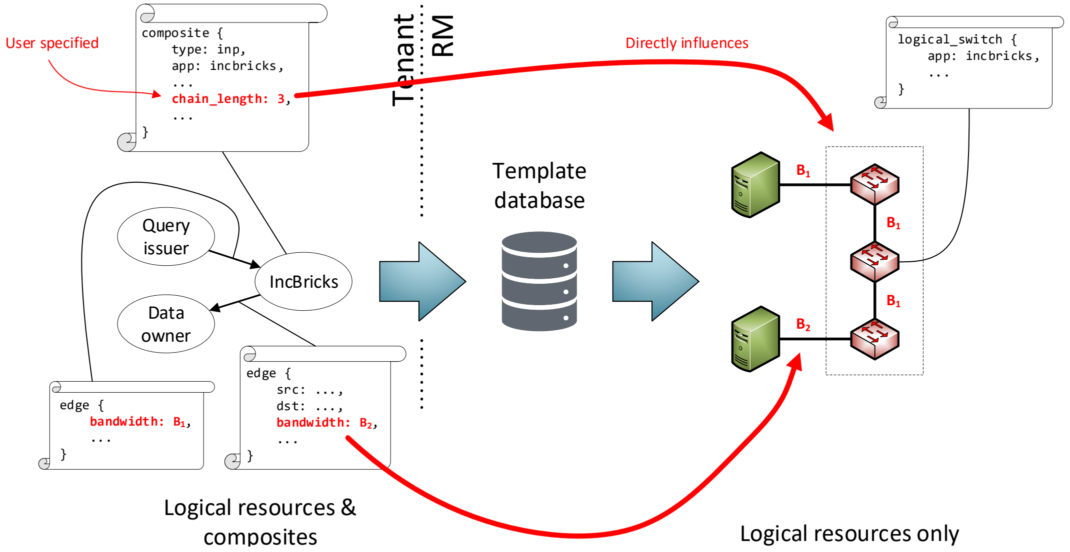
- The *template database* maps composites (or generic groups) to their equivalent made out of just logical resources



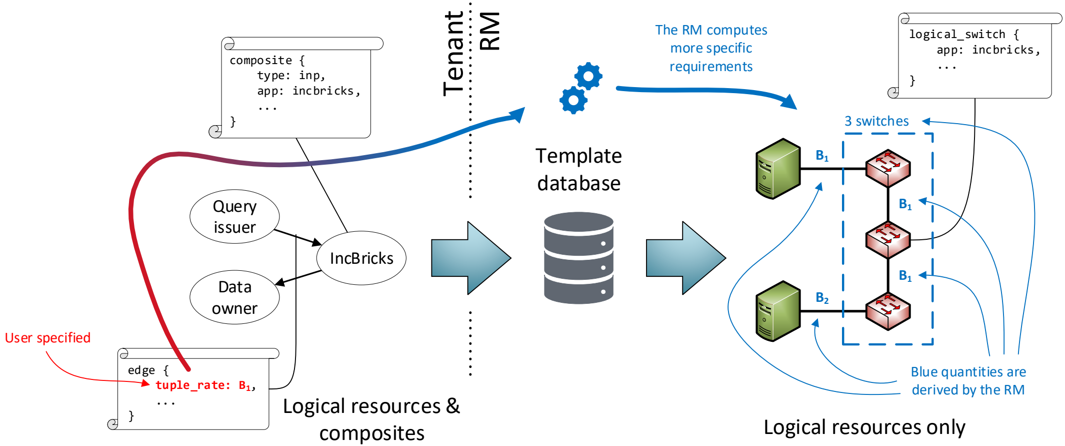
# The whole picture



# $1^{st}$ approach: passive template mapping



# $2^{nd}$ approach: active template mapping



# Evaluation

# Simulation 1/2

- Simulator built from the ground up
  - Inspired by Omega's [5] *lightweight simulator*[2]
  - Supports multiple resource dimensions, switch resources, and composites
- Simulated data center physical architecture: fat-tree with 4 pods
  - Switches have *properties* (e.g., list of supported INP solutions)
- 3 days-long randomly-generated workload
  - Job properties (e.g., requirements, requests' interarrival time, etc.) are sampled from exponential distributions
- Simple greedy scheduler

[2] Available 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

$$STC = \frac{\#server\ tasks\ without\ INP}{\#server\ tasks\ with\ INP} \tag{1}$$

- Sweep: percentage of requests including INP composites

# Results 1/3

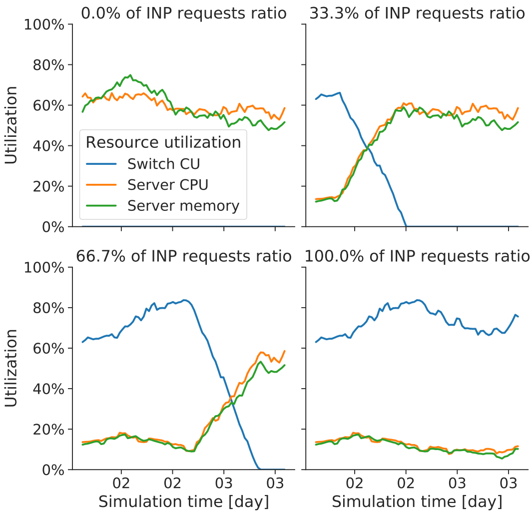



**Figure 1:** physical resource utilization for different amounts of INP requests

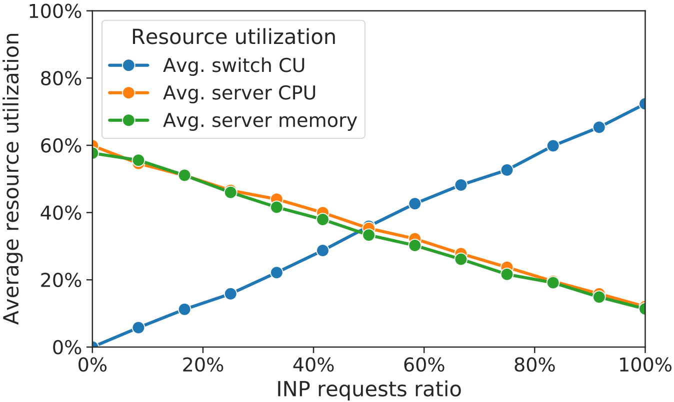



**Figure 2:** Average resource utilization as a function of the INP requests ratio

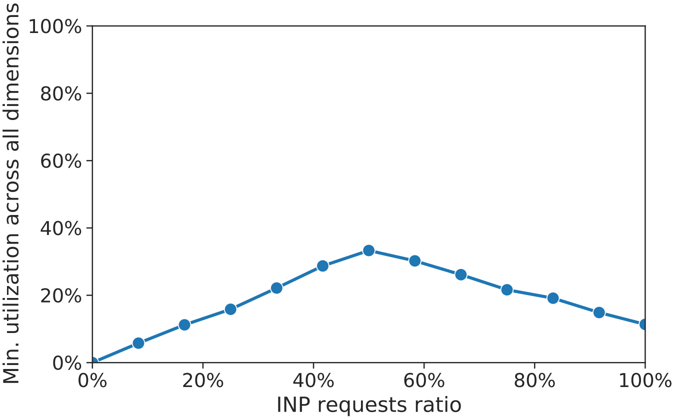



**Figure 3:** Minimum resource utilization across all dimensions

# Conclusions

# Conclusions

**Fully INP-aware RM features**

- Conjunct placement of server and switch resources
- INP alternatives

**Open problems**

- Accurately determine STC values for all INP solutions
- Determine the number of needed switch tasks for INP solutions
- Differentiate INP solutions based on their life cycle (e.g., short-term batch jobs vs. long-term services)

# Questions?

**Thank you**

# Bibliography i


- 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.

- 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.

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.

A. Sapio, 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.