A data science pipeline synchronization method for edge-fog-cloud continuum

Dante D. Sanchez-Gallegos, J. L. Gonzalez-Compean, **Jesus Carretero**, Heidy Marin-Castro

jcarrete@inf.uc3m.es

The 18th Workshop on Workflows in Support of Large-Scale Science (WORKS23)

Motivation

- Many **eScience** problems require very complex and **data intensive cooperation** among multidisciplinary actors.
- To cope with this, workflow managers usually create **dataflow processing schemes on the cloud or HPC centers.**

Common issues when centralizing the management of data

Data confidentiality

Vendor lock-in

Loss of control

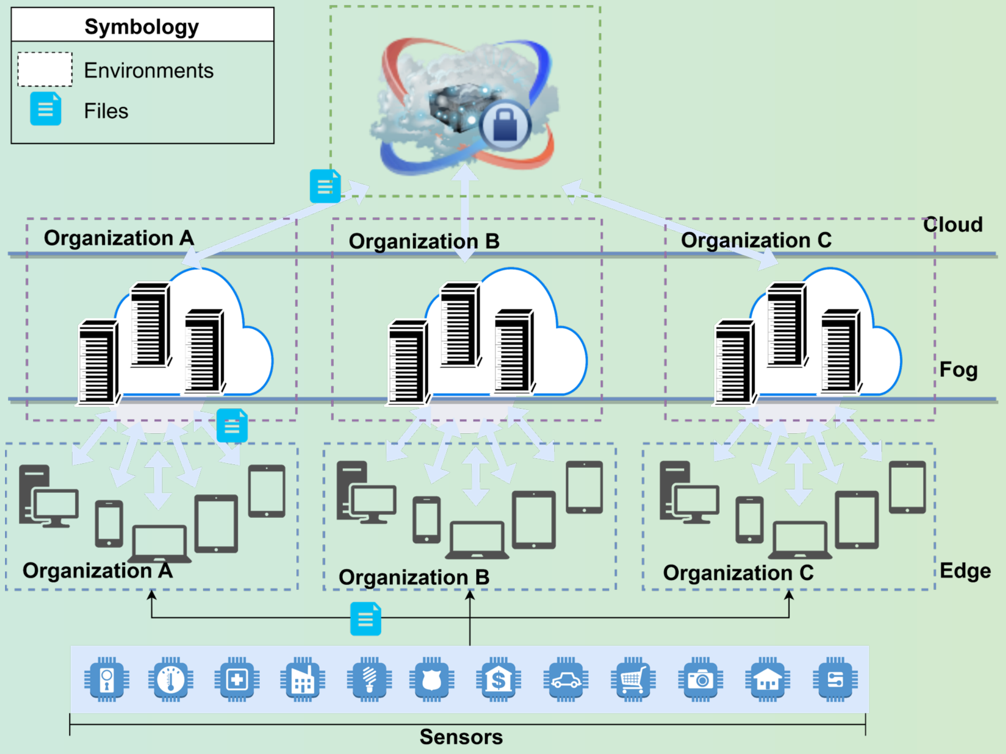
Data accessibility during outages

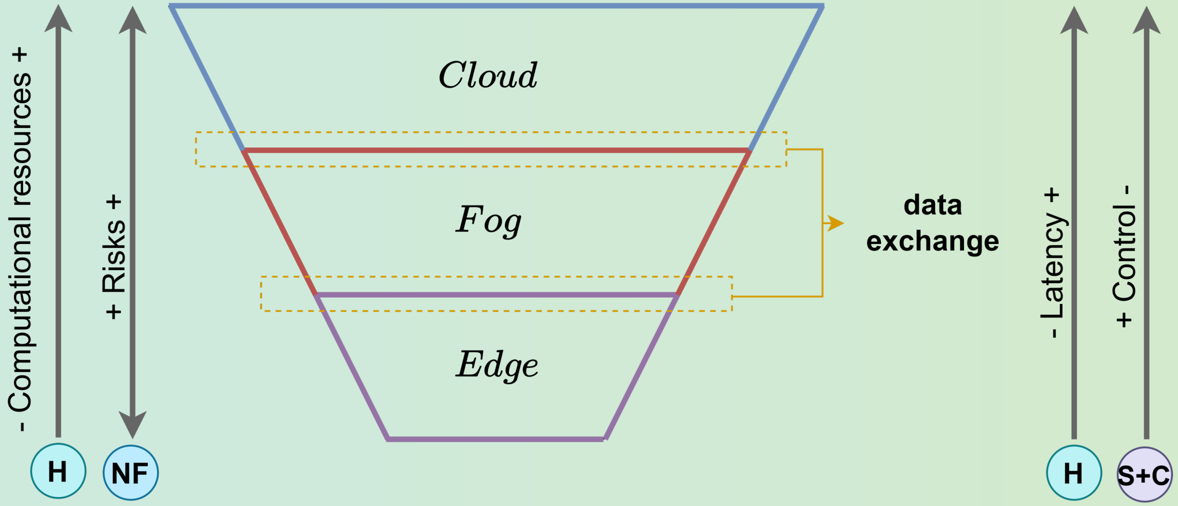
Latency to store and access data

Multi-tier serverless architectures

- Multi-tier serverless architectures allows to create a geographically distributed data service.
  - Deployed dynamically following applications needs

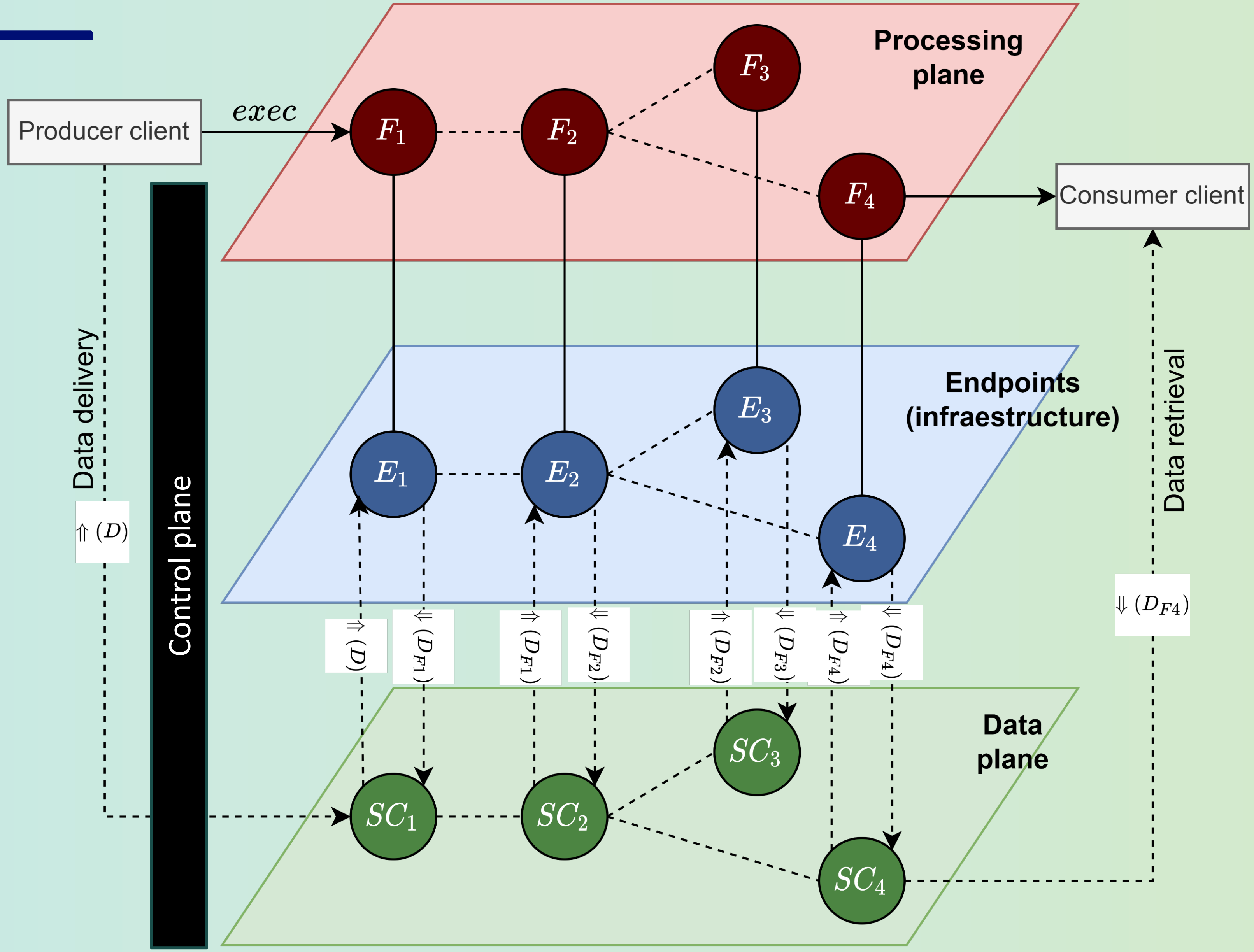
- Challenges:
  - **Latency** between infrastructures.
  - Storage **Capacity** (persistent, volatile)
  - **Synchronization** and global availability of data.
  - To manage the **input/output operations**.
  - Enforcing **Non-Functional Requirements** for the data.





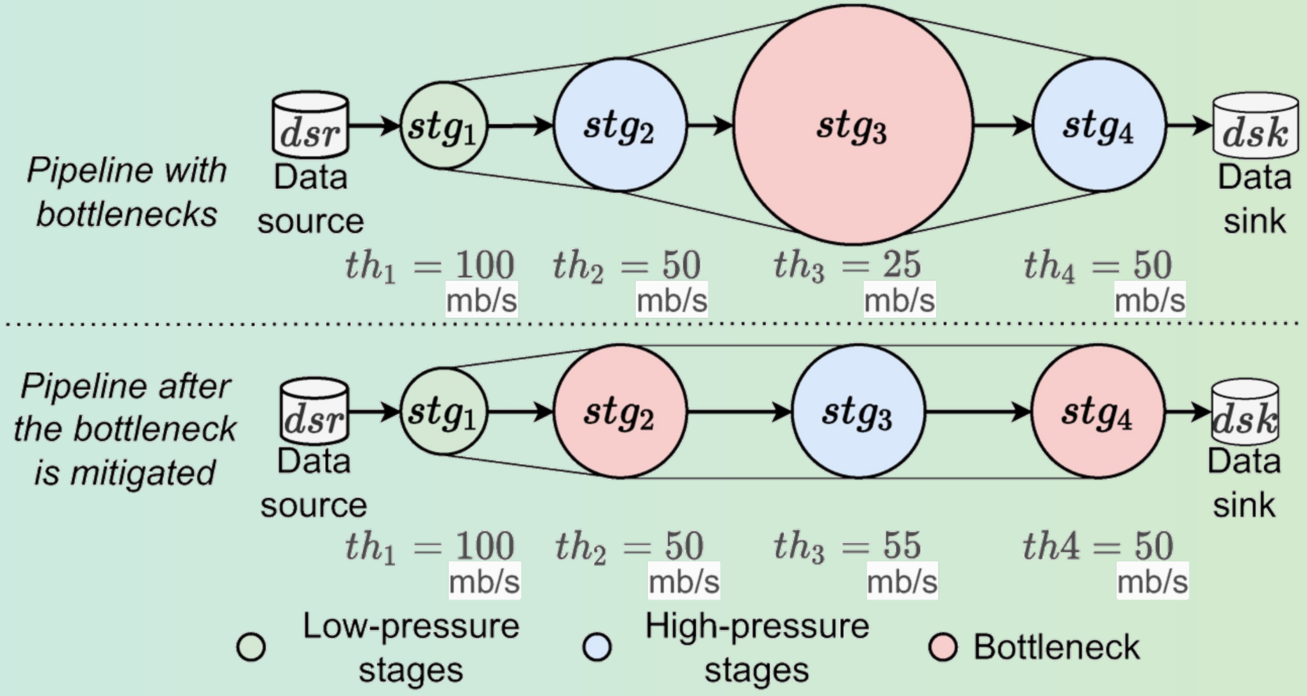
MeshStore: General architecture

- Deployment of systems on the computing continuum.
- Automatic orchestration of data and tasks.
- Continuous monitoring of tasks.
- Implicit parallelism.
- Automatic management of data storage operations.
- Auto-scaling to mitigate bottlenecks.
- Added as a transversal layer to computing continuum systems.
- **Endpoints:** personal computers, servers, clusters, cloud instances, virtual machines, and virtual containers.



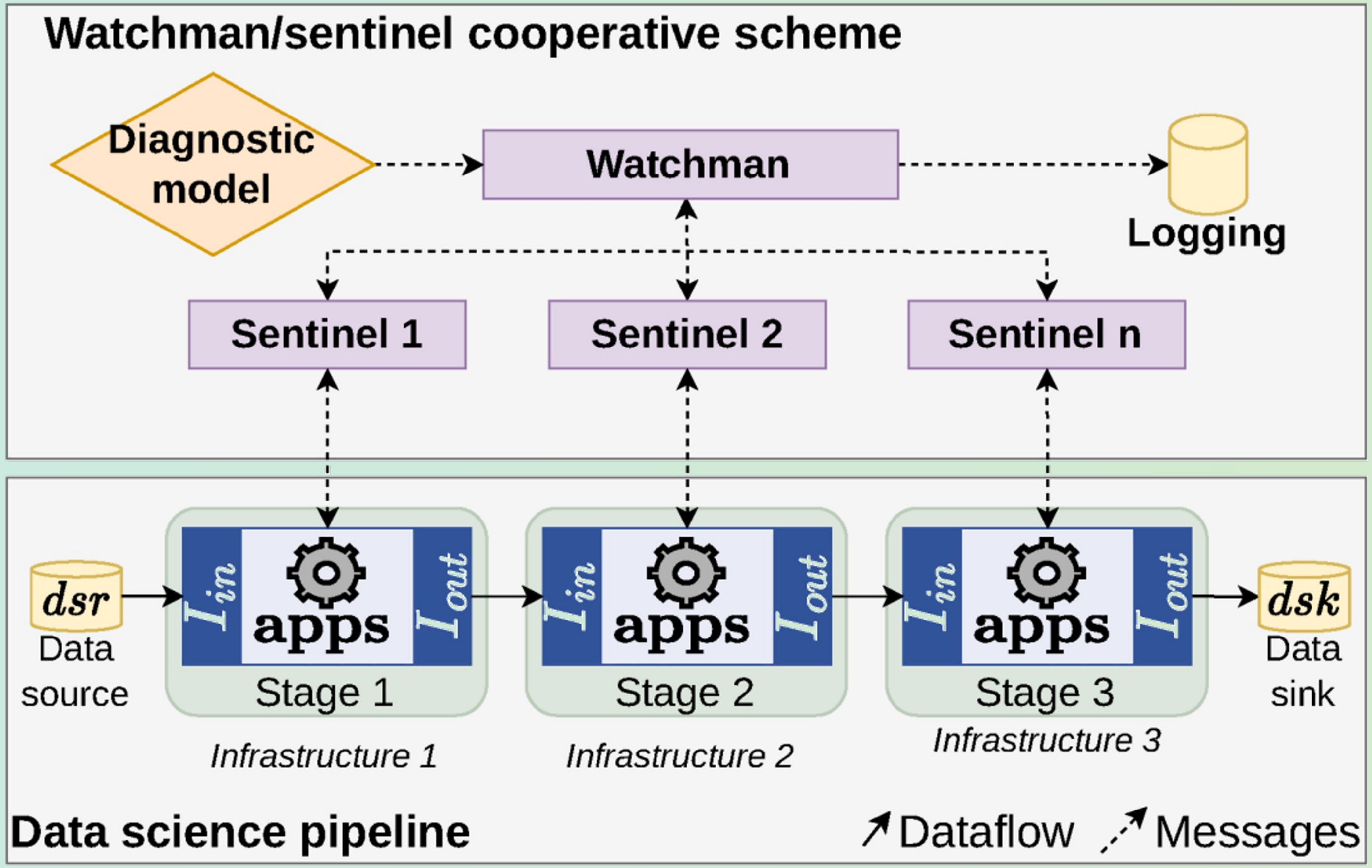
Control plane: diagnostic model to identify bottlenecks

- The performance of a system is modeled based on a Bernoulli principle metaphor.
- We mapped the following variables and elements:
    - Dataflow = a flow in a streamline
    - Throughput = velocity of a fluid
    - Pressure = input workload stored in the input buffer
    - The fastest stages = low pressure points
    - The slowest stages = high pressure points.
- Stages are classified according to their throughput.



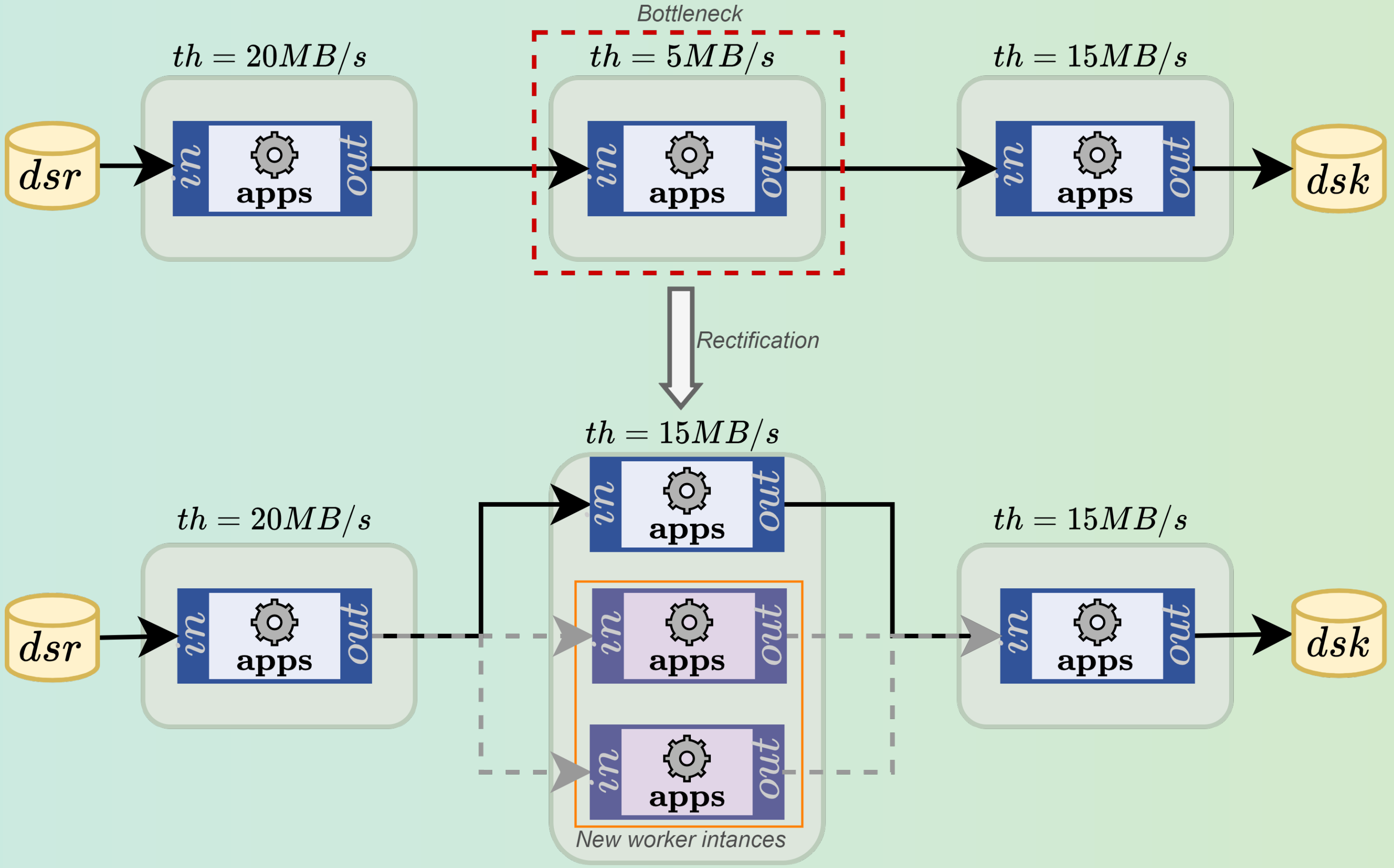
Control plane: continuous monitoring and rectification scheme

- The throughput, response time, and input buffer utilization of the stages are monitored using entities called **sentinels**.
- The metrics are delivered to a watchman entity.
  - Identifies the bottleneck based on their throughput.
  - $Bottleneck = \min(th_i) i \cdots n$
    - th = throughput
- The watchman and sentinels are added as a transversal layer to the stages.



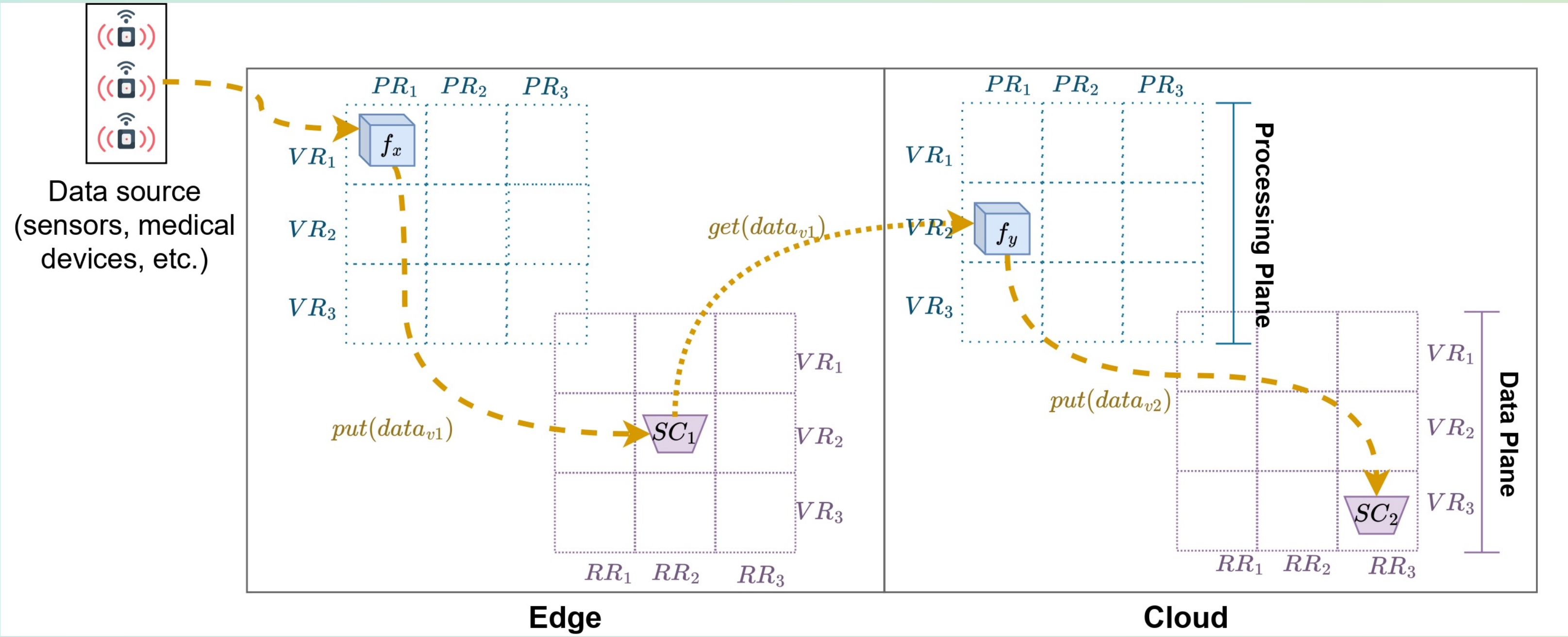
Control plane: rectification scheme to solve bottlenecks

- Bottlenecks are mitigated using a manager/worker parallel pattern.
- The number of workers is obtained from the response time of the bottleneck and a metric called takt time.
    - $workers = \min(\lceil \frac{RT_{Btl}}{TkT} \rceil, N_{cores}),$
    - The maximum number of workers is limited to the number of cores in the machine.
    - Takt time: maximum service time required to process an objective demand.
    - $TkT = \frac{MRT}{|inBuff|}$
        - MRT = median response time of the stages near in performance to the bottleneck.
        - inBuff = size of the bottleneck's input buffer.



Data plane: moving data through the computing continuum

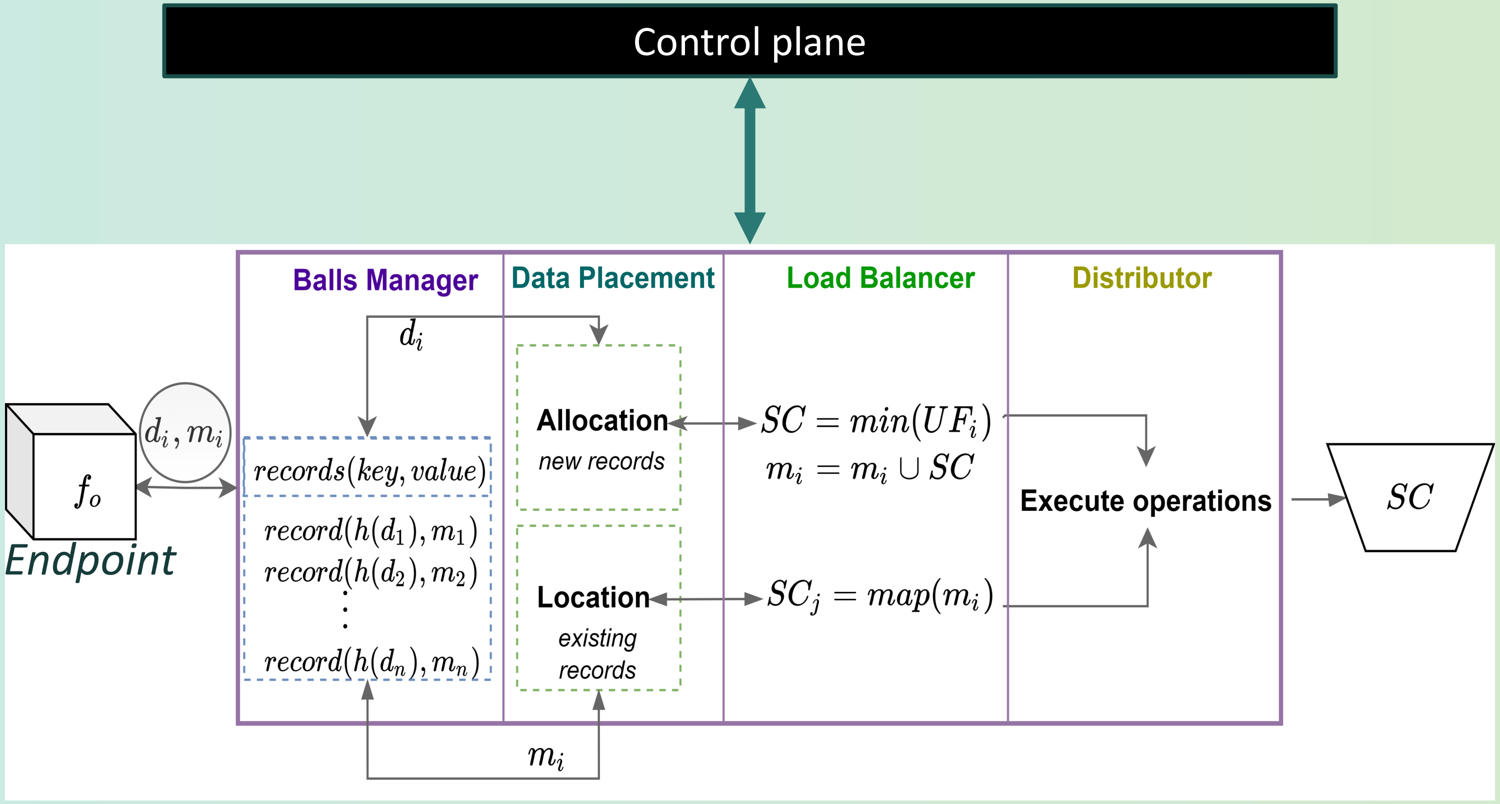
- Applications use put and get operations to access to data in the data plane.
- Data plane:
  - Managed as a mesh.
  - Composed of storage containers.
  - Interfaces:
    - Filesystem
    - Memory
    - Network
- Creates a content delivery network that connects multiple infrastructures.



PR: Physical resources VR: Virtual resources

Data plane: storage scheme

- The allocation/location of data is based on a balls-into-bins metaphor.
- The data placement (allocation) is based on a two choices load-balacing algorithm with an **utilization factor (UF_i)**.
  - $SC_j = \min(UF_i = 1 - (\frac{C_i - Ui}{C}))$
  - $U_i = SC_i$ usage
  - $C = \Sigma c_{i,\ i = 1 \ldots n}$
  - $C_i = SC_i$ capacity
- Metadata maps are generated for each content (m_i) to be stored in a storage container.
  - Location, NFR, …

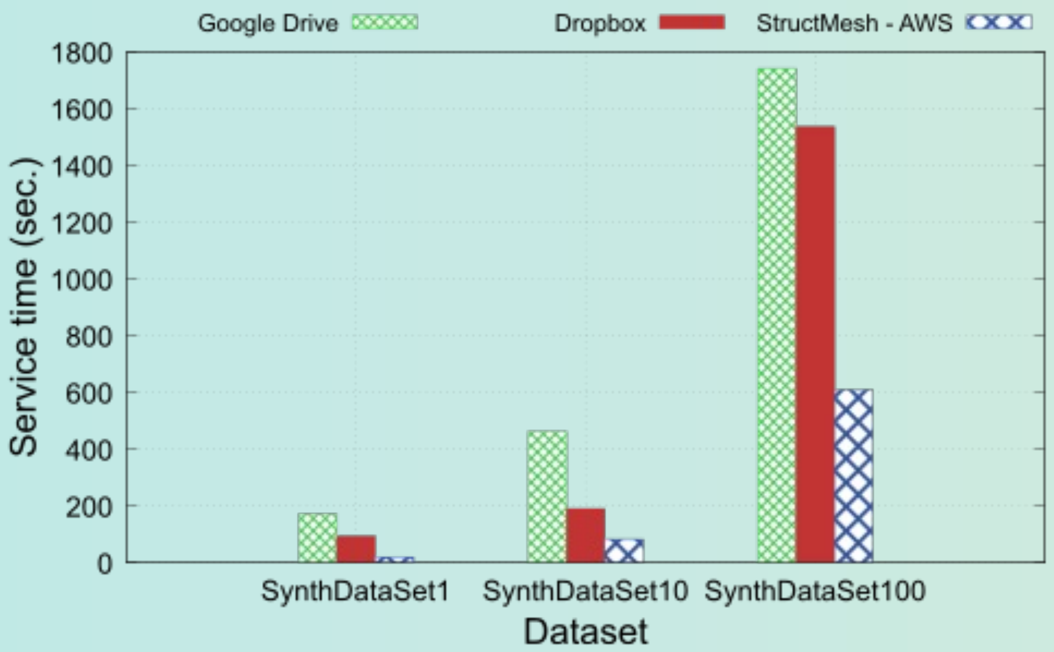


Experimental evaluation

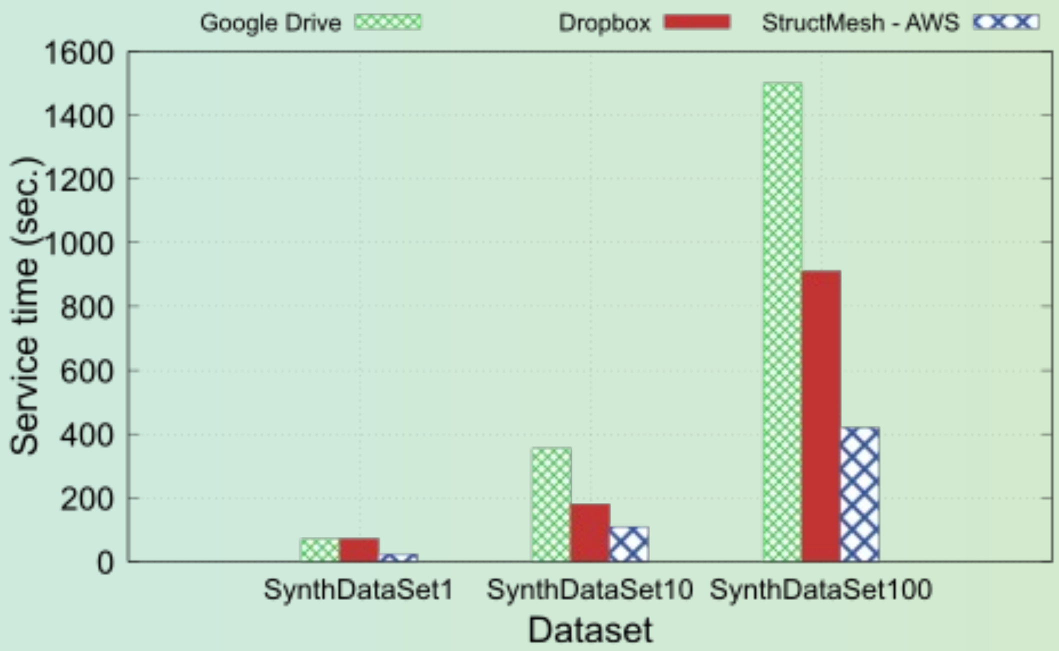
- Evaluation performed using synthetic data and real meteorological traces.
- Evaluated using simultaneously distributed infrastructure available at Mexico, Spain, and Amazon AWS.
  - Mexico. 1 edge, 3 fog.
  - Spain. 1 edge, 2 fog.
  - AWS. Shared storage instance.
- A storage mesh was created using that infrastructure.

Comparing uploading/downloading operations with comercial tools

- We evaluate the time to share 100 files of 1, 10, and 100 MB from the Tamaulipas, Mexico to Madrid, Spain.
  - Comparison between Google Drive, Dropbox, and MeshStore.
  - We connected MeshStore to storage containers deployed on AWS



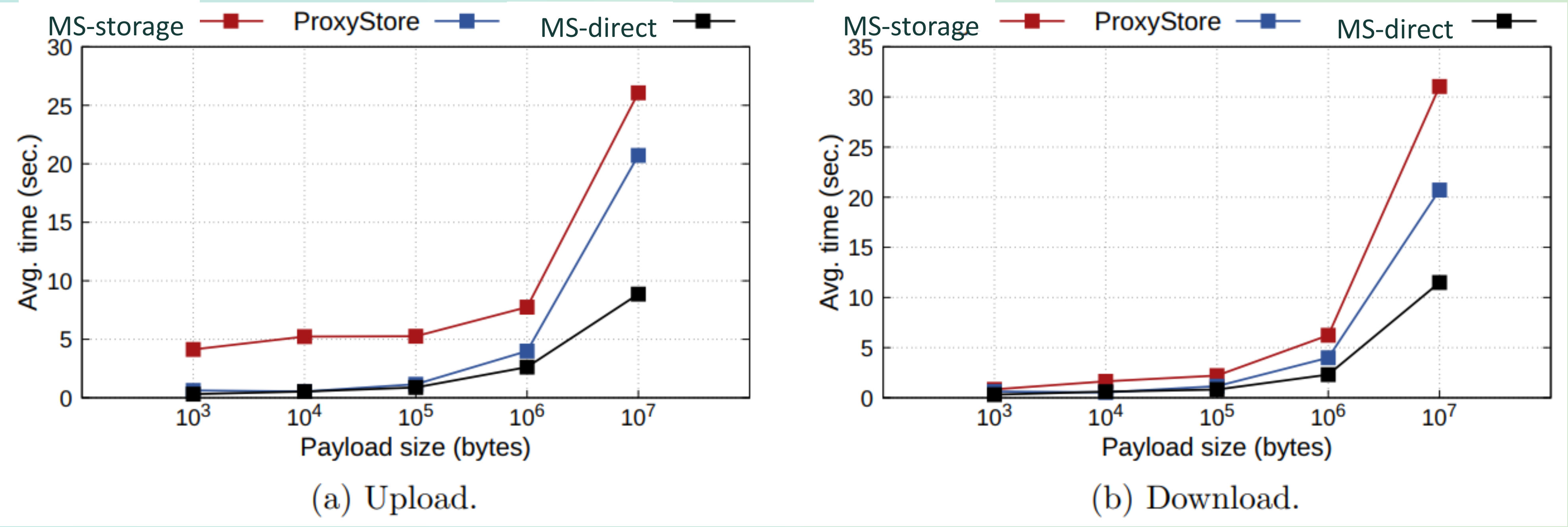
Data uploading from Mexico to the cloud.



Data downloading from the cloud to Madrid.

Data movement evaluation

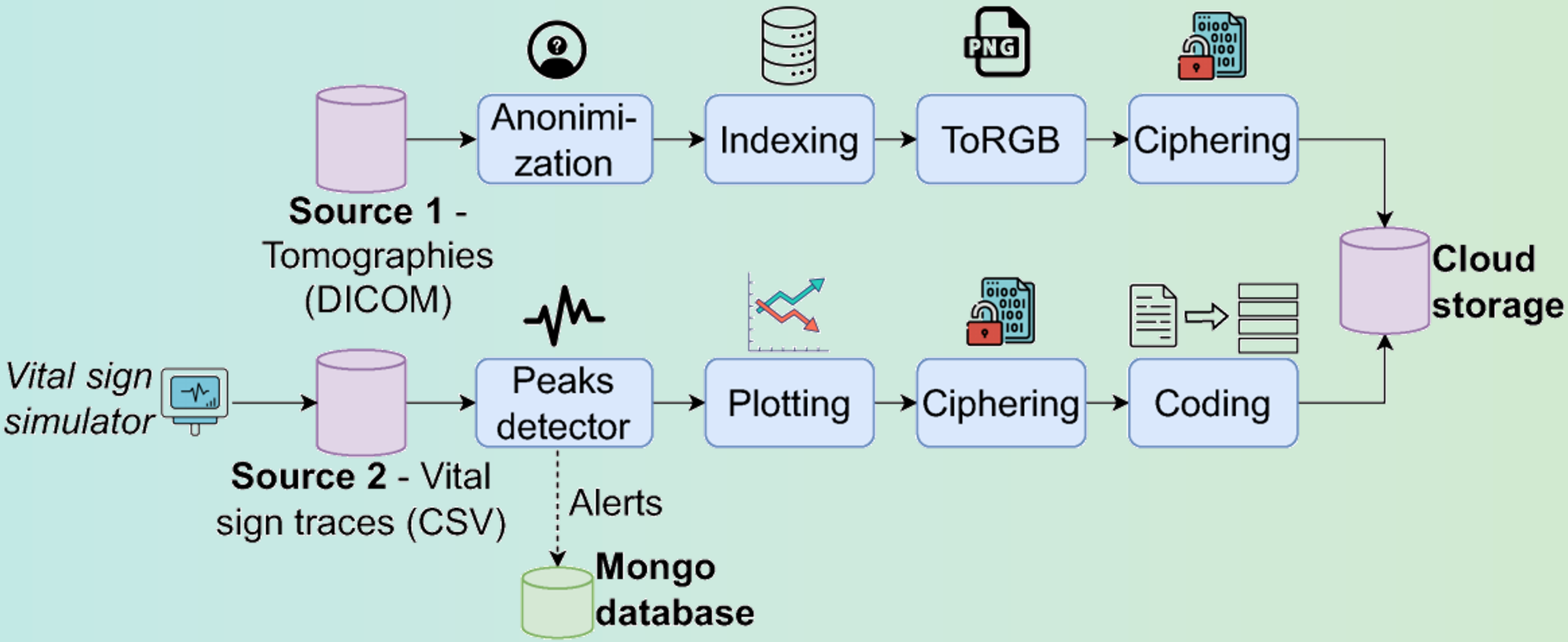
- Point-to-point transmission of data from a computer in the UC3M (Madrid) to a virtual machine in Amazon EC2 (US East - N. Virginia).
  - **ProxyStore[1]** to transfer data on inter-site environments (Point to Point data transmission).
  - **MeshStore-direct**: a direct transmission of the data (Point to Point data transmission).
  - **MeshStore-storage**: including the storage of the data for their long-time preservation on storage containers (serverless).



[1]Pauloski, J. G., Hayot-Sasson, V., Ward, L., Hudson, N., Sabino, C., Baughman, M., ... & Foster, I. (2023). Accelerating Communications in Federated Applications with Transparent Object Proxies.

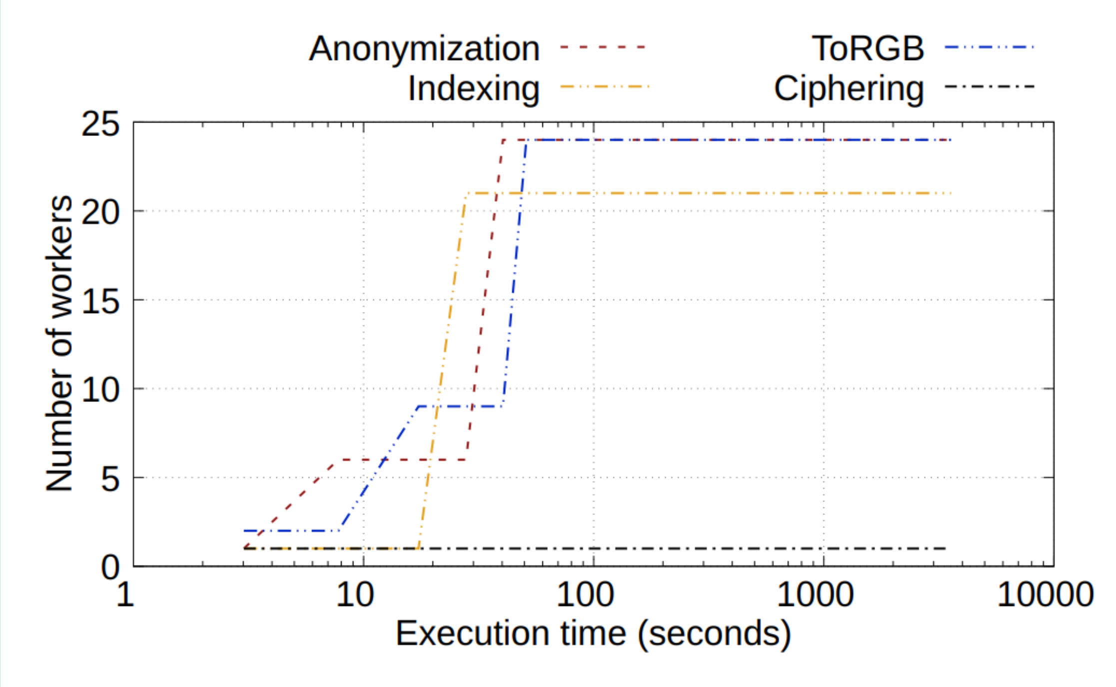
Case study: management of medical data

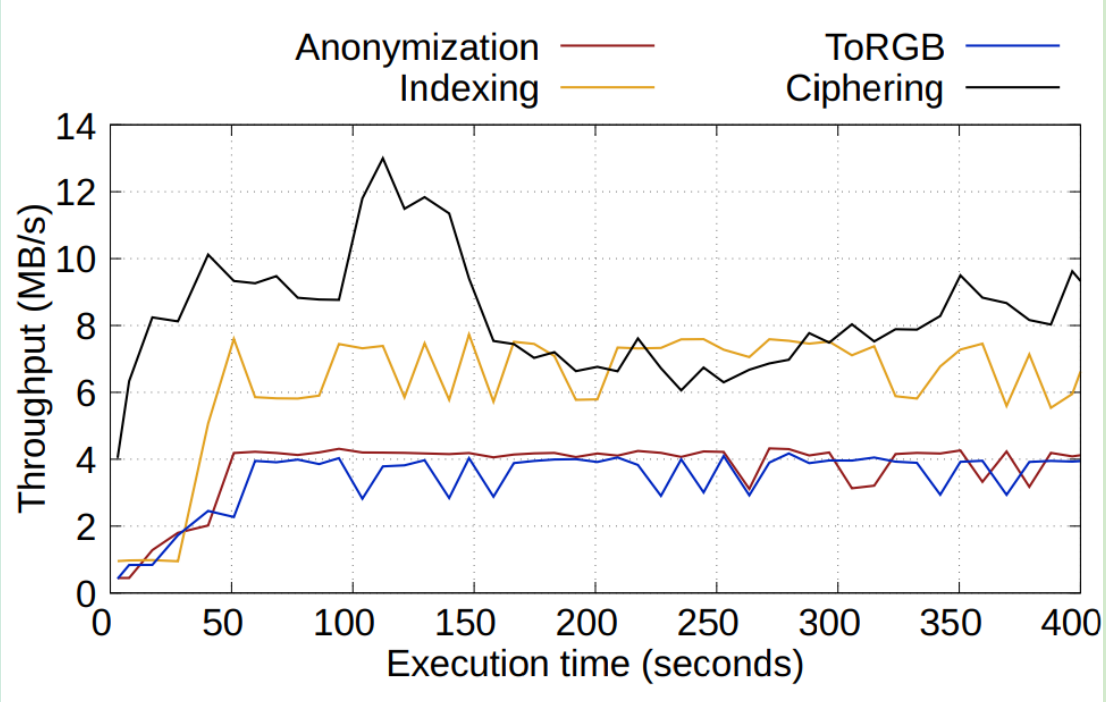
- System deployed on fog and EC2 infrastructures.
  - Data source 1: 9533 tomography images with a total size of 4.7 GB.
  - Data source 2: 1000 CSV files (57.3 MB) generated with a vital sign simulator.



Case study: management of medical images

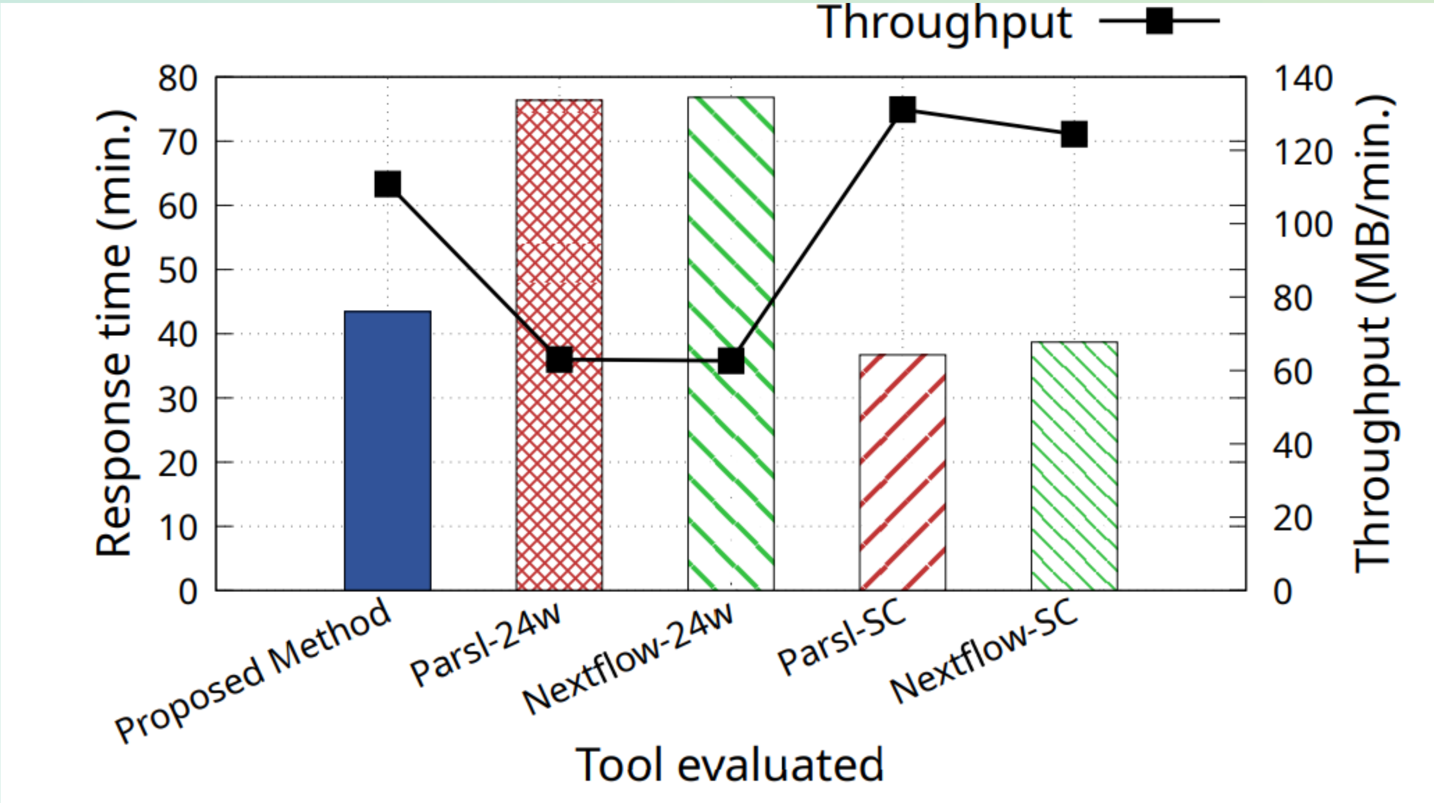
- The method achieves a speed-up of 3.94x and 3.74x on the stages identified as bottlenecks.
- The maximum number of workers is 24, which is equal to the number of cores on the infrastructure.
- The improvement of the performance of bottlenecks has a direct impact on the performance of the fastest stage (Ciphering), as it can process more data per second.





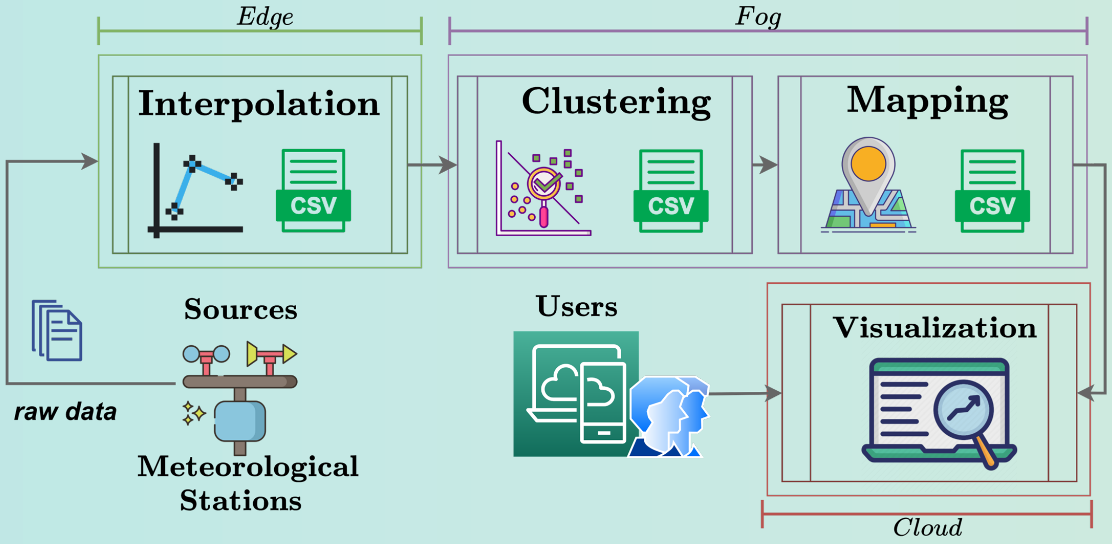
Case study: management of medical images

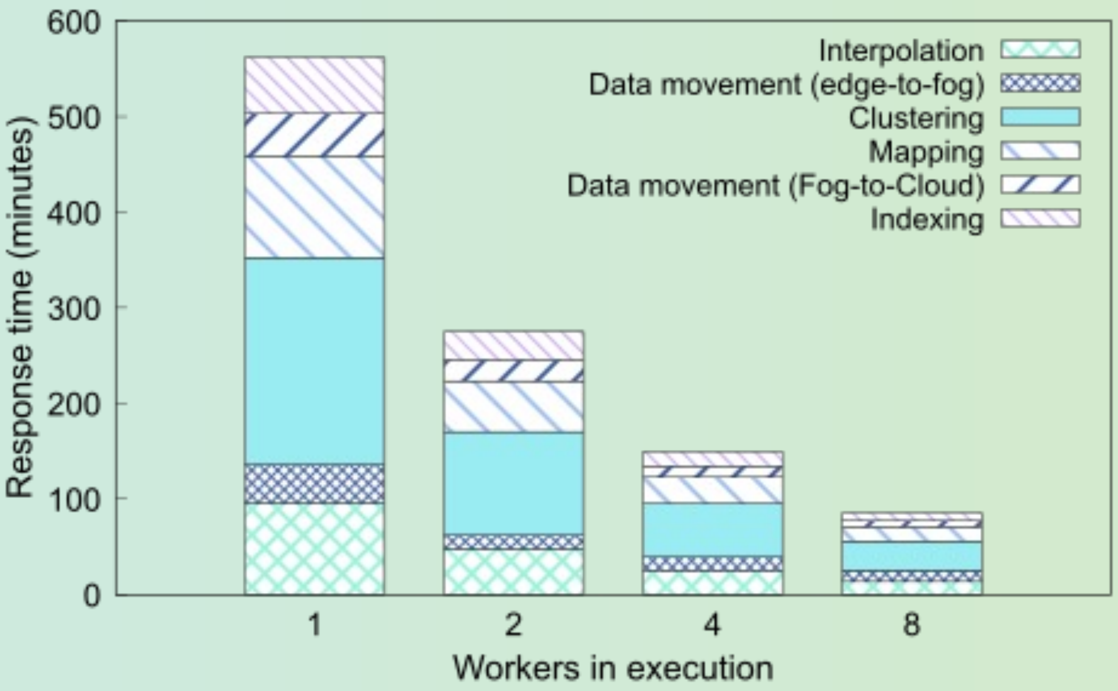
- **Parsl-24w** and **Nextflow-24w**: using all available resources without managing bottlenecks.
- **Parsl-SC** and **Nextflow-SC**: using the steady configuration obtained by our method.
- Our method reduces the response time of 43.14% and 43.46% in comparison with Parsl and Nextflow, respectively.



Case study for the management of meteorological data

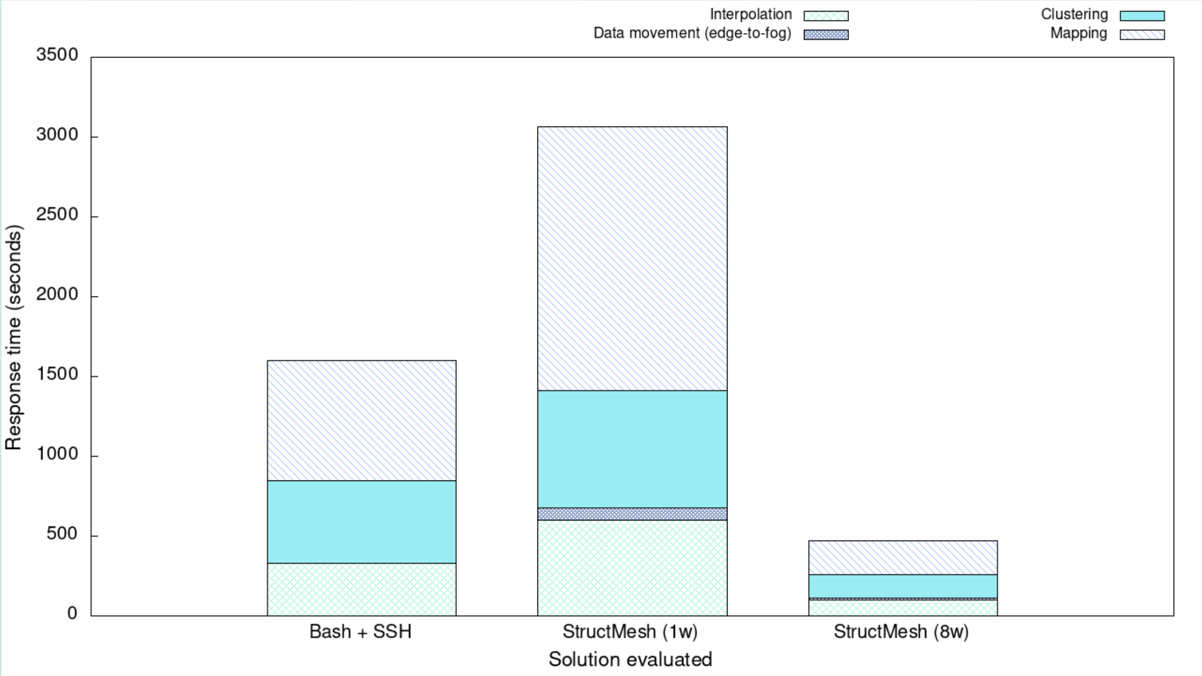
- We evaluate the performance of the data management workflow by scaling the number of available workers.
- For each stage in the workflow, we executed 15817 functions (one for each file in the MERRA-2 dataset).
- Execution time for one worker is 805.07 minutes. For 8 workers, 119.79 minutes.





Case study for the management of meteorological data

- Evaluation of a solution using bash scripts to execute the functions and SSH to move the data from the edge to the fog.
  - It requires the complex management of SSH credentials and the installation of the functions and their dependencies in the available infrastructure.
- The spatial variables are for the Yucatan peninsula in Mexico, whereas the temporal variables were limited to 2016.



Conclusions

- MeshStore is based on storage structures that represent maps of storage resources available on multiple infrastructures.
- Automatically manages the data required and produced by serverless functions.
- Automatically identifies bottlenecks on computing continuum systems.
- Creates a representation of the state of functions and applications based on the Bernoulli equation.
- A unified storage layer is added in a transversal manner to serverless functions.

Ongoing work

- Integration of MeshStore with a blockchain model to keep the traceability of the data and exploitation through smart contracts.
- Study of self-adaptable mechanisms to choose the number of workers and virtual containers in a storage mesh.
- Enhancing data distribution by alleviating I/O bottlenecks.
- Using ad-hoc storage deployments per workflow to enhance I/O in HPC systems

A data science pipeline synchronization method for edge-fog-cloud continuum

Dante D. Sanchez-Gallegos, J. L. Gonzalez-Compean, **Jesus Carretero**, Heidy Marin-Castro

jcarrete@inf.uc3m.es

The 18th Workshop on Workflows in Support of Large-Scale Science (WORKS23)