Telefónica Innovación Digital

Scheduling Inference Workloads on Distributed Edge Clusters with Reinforcement Learning

Francisco Álvarez Terribas, Ferran Diego Andilla

23/01/2025 – Telefonica Scientific Research

NebulOuS
Grant Agreement No: 101070473

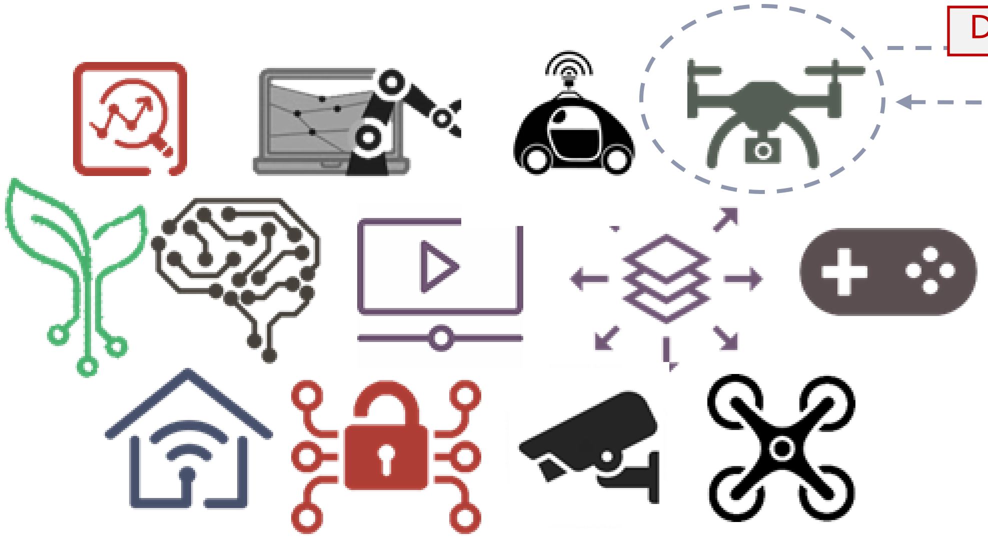
EDGELESS
Grant Agreement No: 101092950

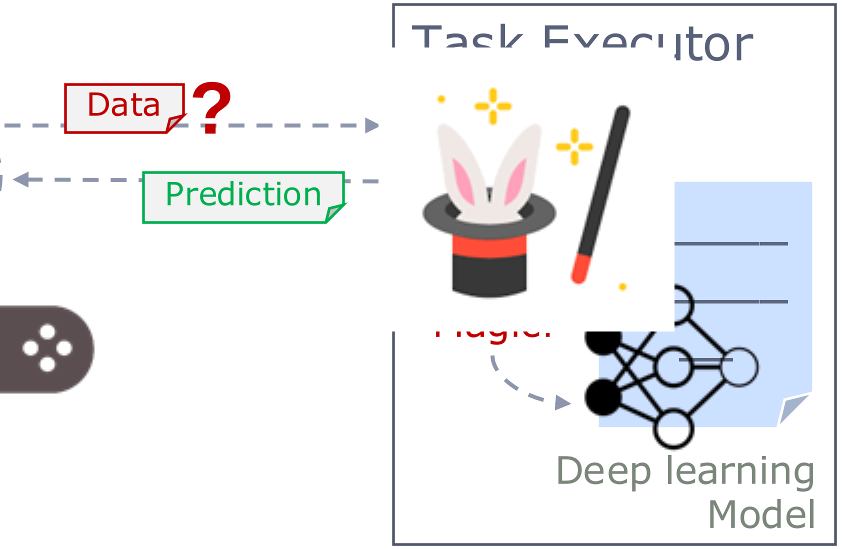
FLUIDOS
Grant Agreement No: 101070516

Cloud Computing as a modern REVOLUTION

In search for computing as a utility

Application Backgound





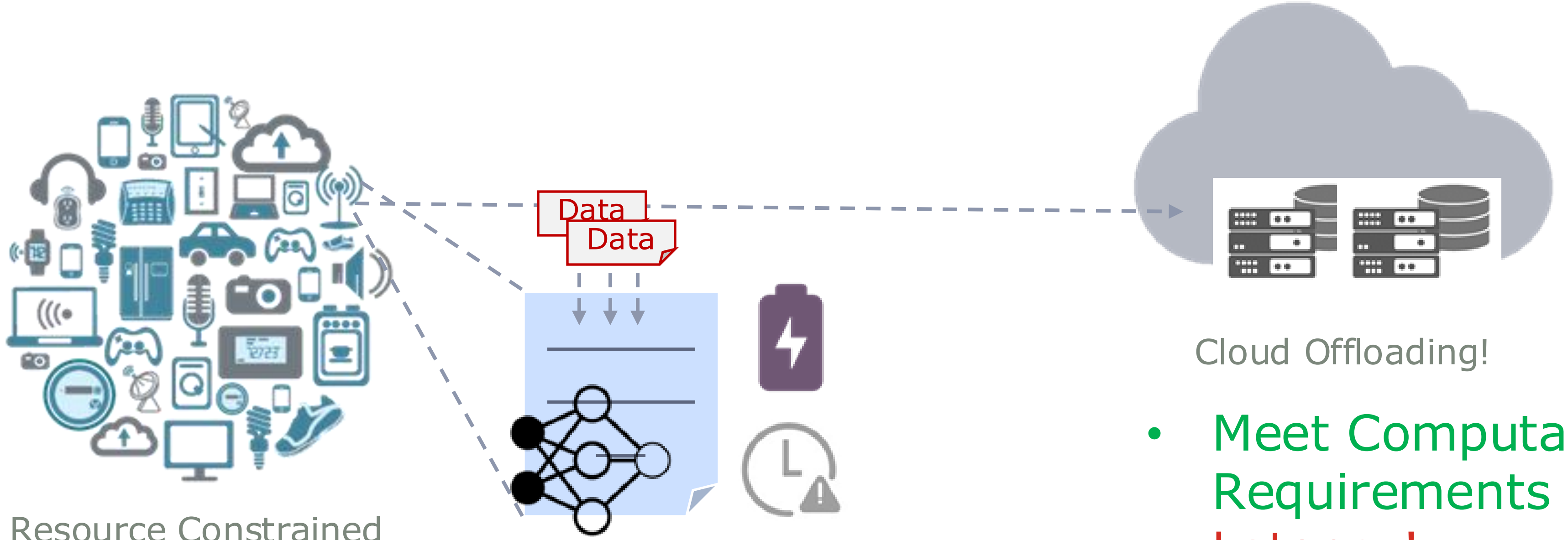
- A large variety of application components exploits Deep Learning
  - Computer vision
  - Natural language processing
  - Big data analysis
  - More …

Something has been left behind

- A class of Next Generation Applications (VR, XR …)
    - Low-latency
    - High-bandwidth

Edge Computing

Cloud is not always the best option!



Resource Constrained
End Devices

- Computer vision
- Natural language processing
- Big data analysis
- More …

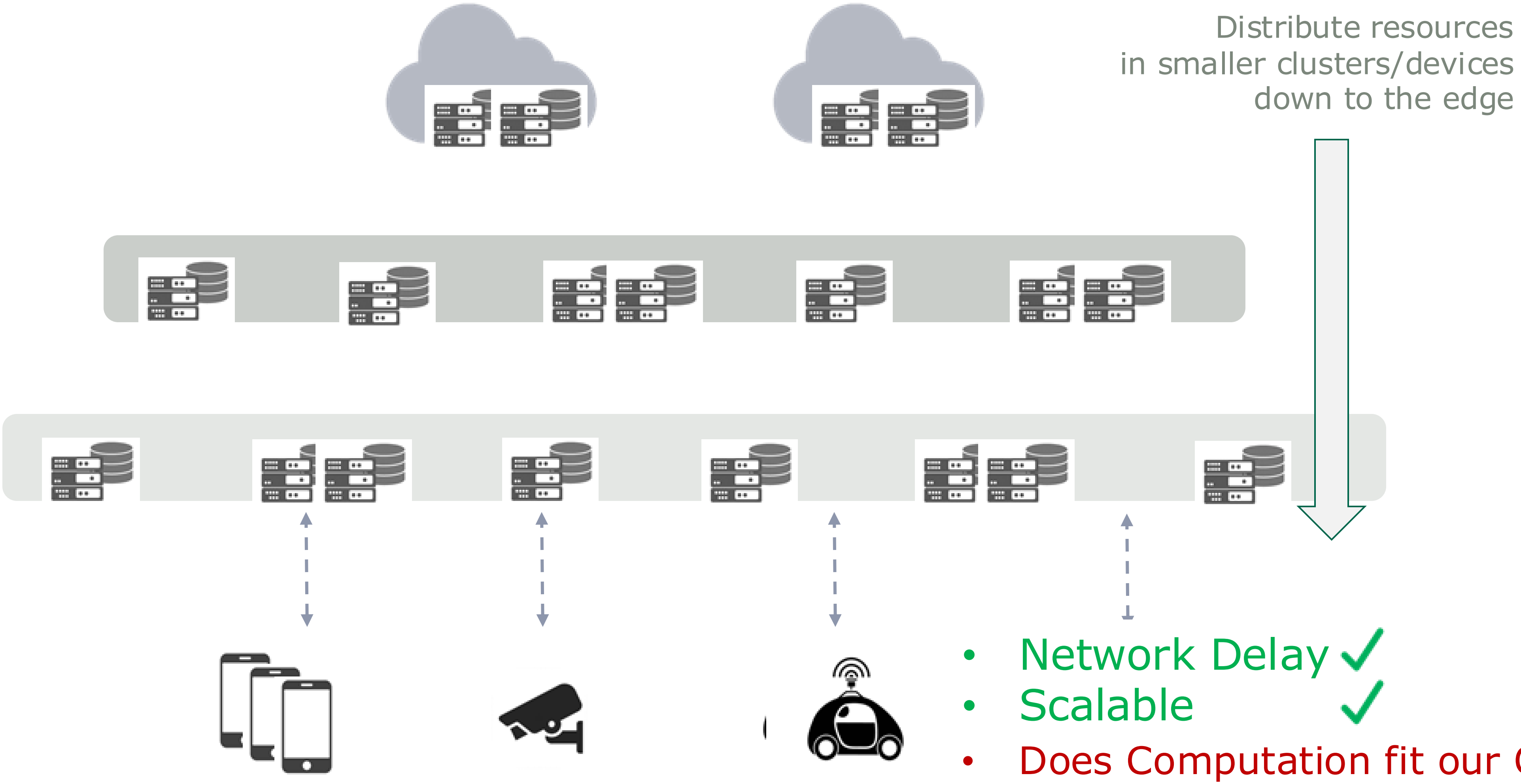
- High Accuracy!
- High Computation Requirements

Cloud Offloading!

- Meet Computation Requirements
- Latency!

Edge Computing

Cloud offloading challenges



- Network Delay ✓
- Scalable ✓
- Does Computation fit our QoS?

Edge Computing

Cloud offloading challenges

Deep learning demands high computation

\+ Edge Nodes feature less resources!

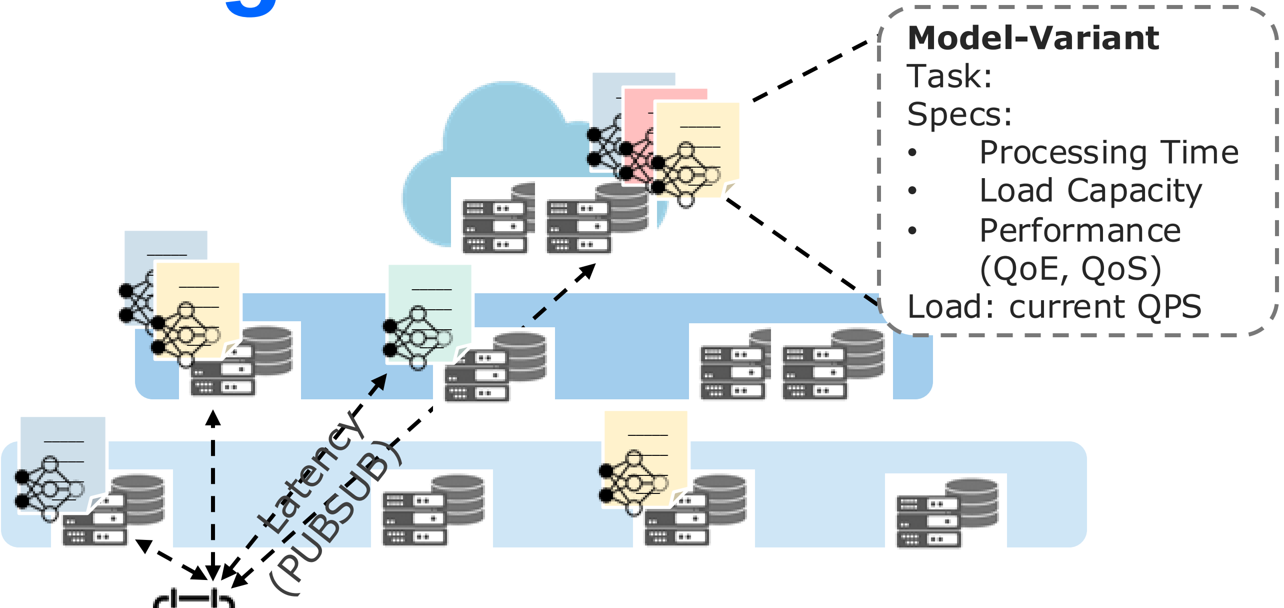
= Way more challenging resource management!

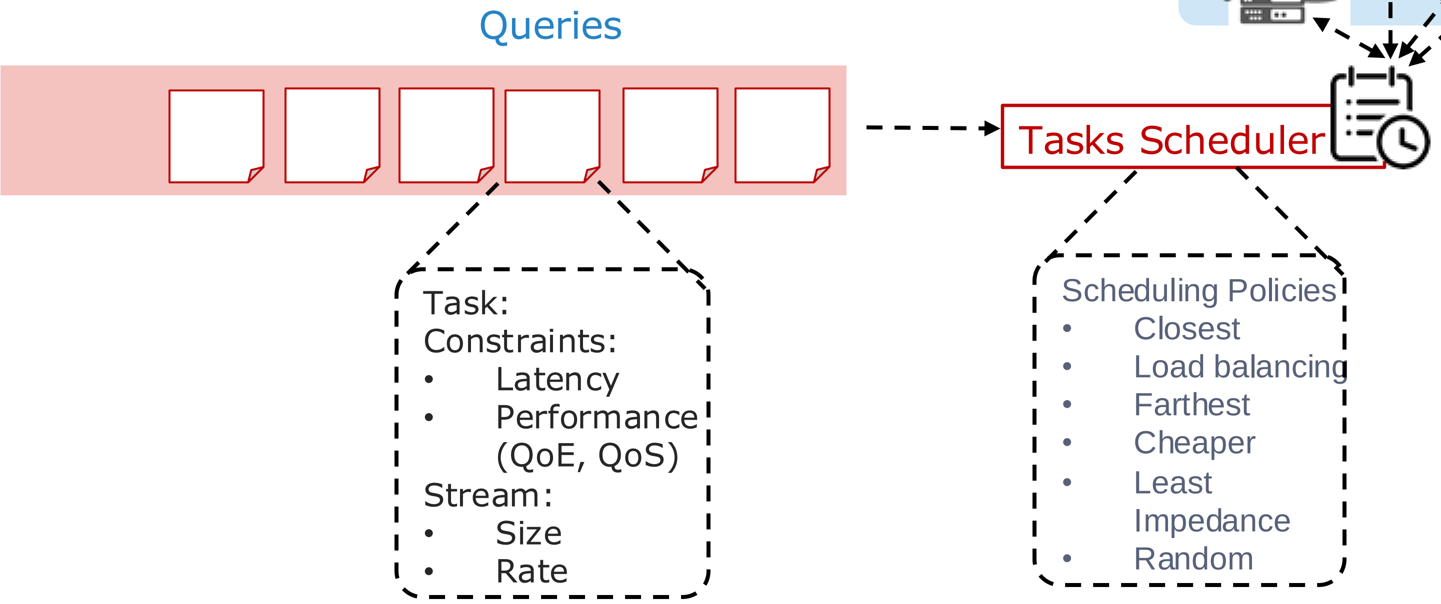
Differently from the cloud:

- We cannot rely on potentially unlimited resources
- We may not be able to always deploy, **for each task**:
  - The **most performant** model
  - At **every** edge location
  - **Scale it** up indefinitely

Adaptive Scheduling of Edge Tasks

Task Scheduler: An overview





- Keeps information about model deployment and their load
- Decides: **which model** on **which nodes** should serve queries
  - Avoid saturating nearest and most performant models, potential future queries!
  - The farther the node, the higher the uncertainty on network latency!

Adaptive Scheduling of Edge Tasks

Scheduling outputs

Closest: sent to the worker n$*$ that features the lower network latency

$$n^* = \arg\min_{n \in N} (d_n^s + 2\sigma_n^s)$$

Load balancing: may lead to unfair allocation when latency-sensitive tasks are in the minority

$$(v^*, n^*) = \arg\min_{v,n \in V^m \times N} L_{vn}(t)$$

Farthest: send to the worker n $*$ with the highest (still feasible) network latency

$$n^* = \arg\max_{v \in N} (d_n^s + 2\sigma_n^s)$$

Cheaper: send to worker n$*$ such that the expected end-to-end delay (roundtrip and processing time) is maximized

$$\arg\min_{v,n \in V^m \times N} (2(d_n^s + 2\sigma_n^s) + D_v(\zeta_i))$$

Random-propotional latency: guarantees that, on a large enough number of streams, bindings are proportionate to end-to-end delays

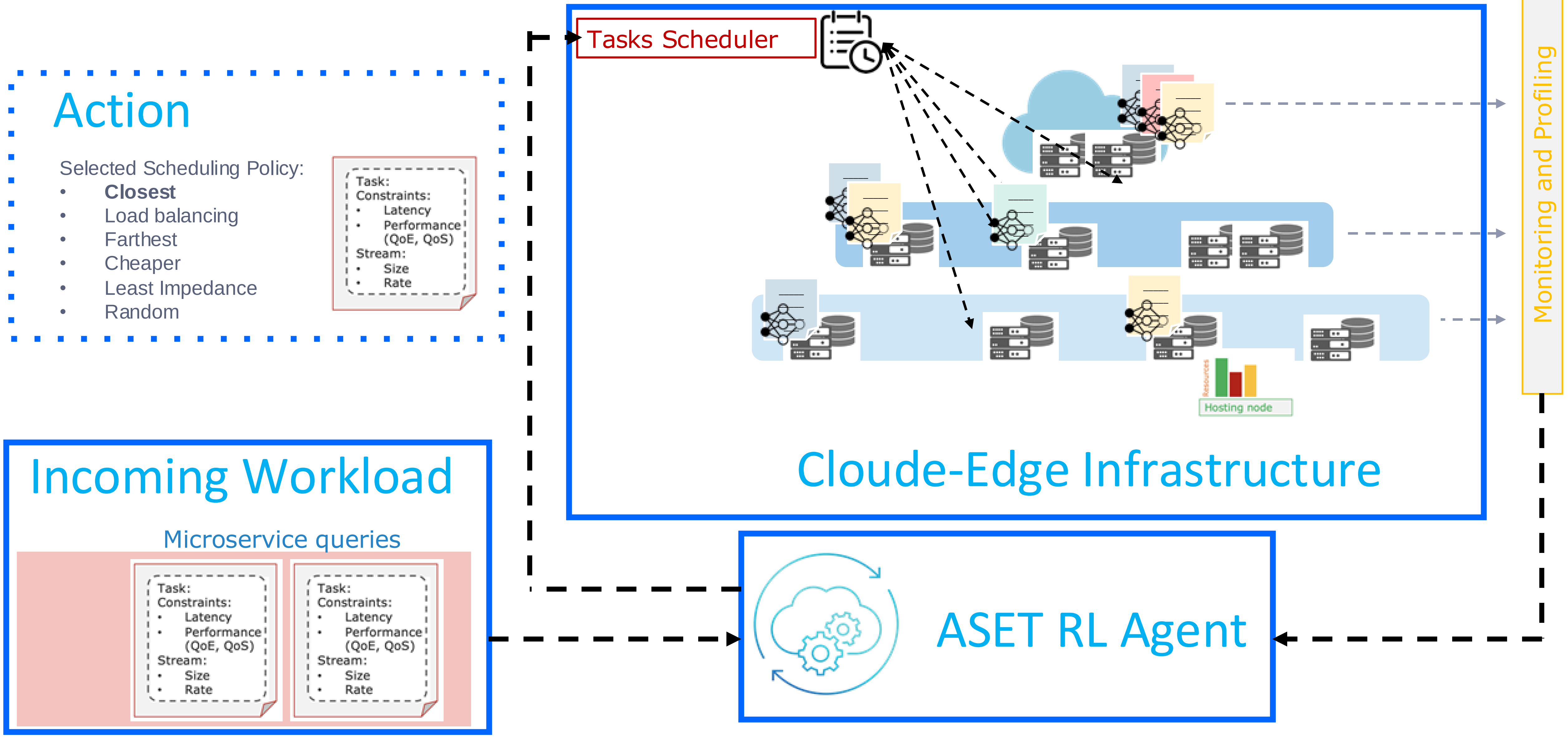
Random-proportional load: guarantees that, on a large enough number of streams, bindings are proportional to the capacity of each model variant.

Least impedance: send to the worker such that the end-to-end latency to s is minimized

$$(v^*, n^*) = \arg\min_{v,n \in V^m \times N} (2(d_n^s + 2\sigma_n^s) + D_v(\zeta_i))$$

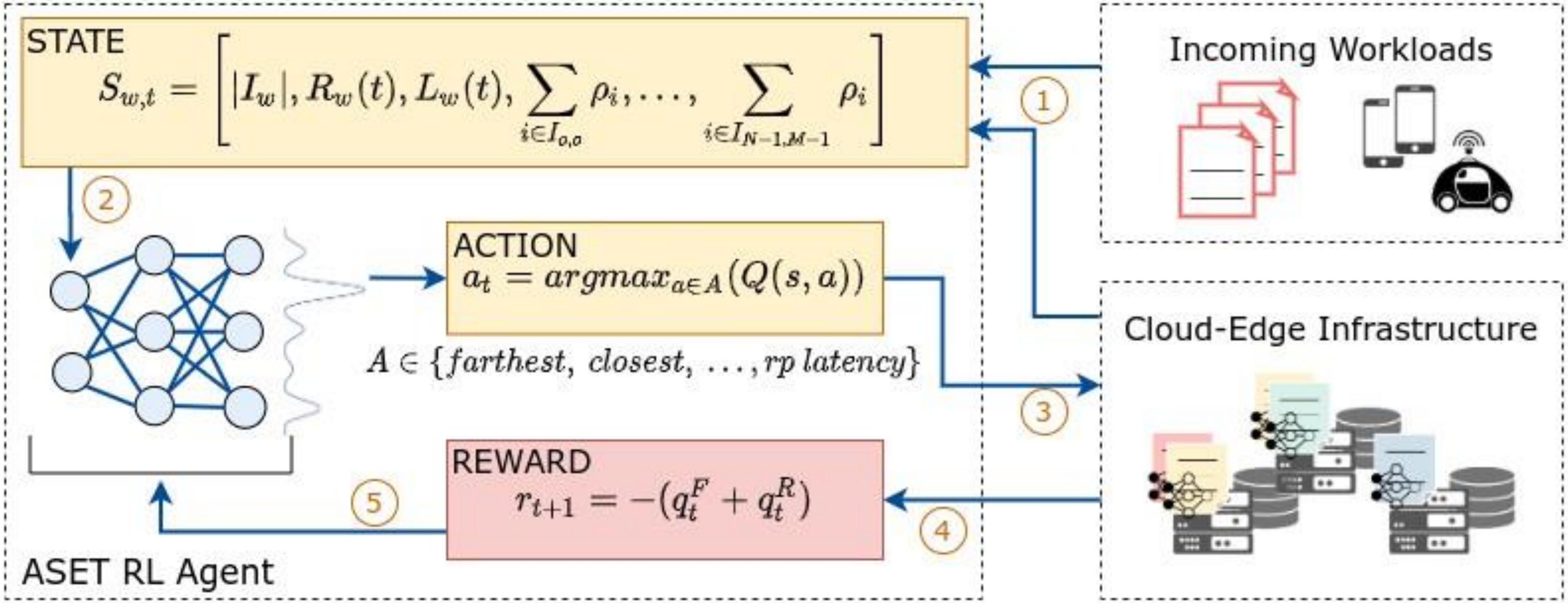
Adaptive Scheduling of Edge Tasks

Adaptive Task Scheduler: RL prespective



Adaptive Scheduling of Edge Tasks

Learning ASET RL Agent



Adaptive Scheduling of Edge Tasks

RL agent

receives the state from the scheduler and it sends back the best action, or in other words the policy.

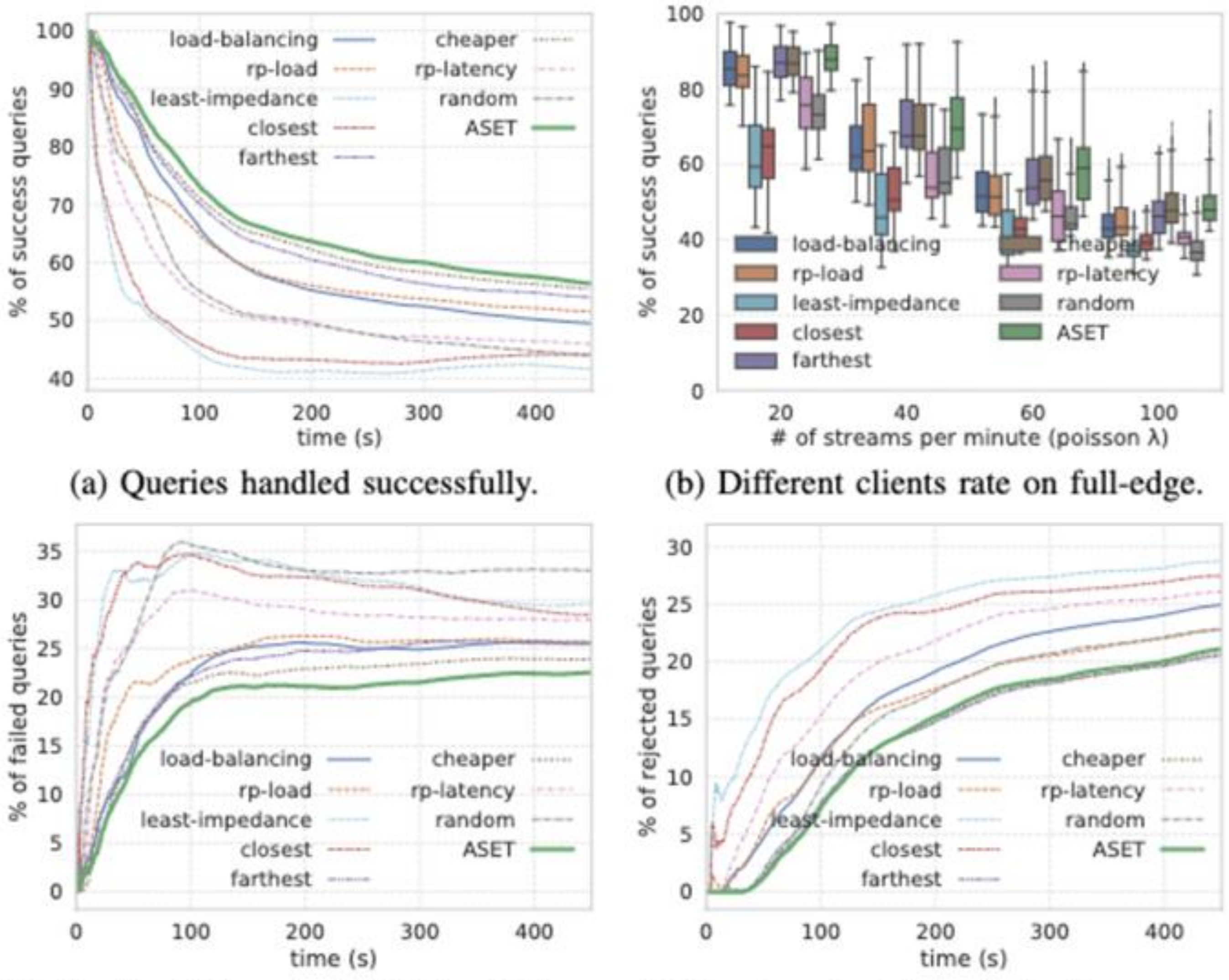
The features that we are using as an input for the agent are the followings:

    qps = queries per second
    rps = responses per second
    clients = number of clients in the simulator
    load node 1 = load in terms of qps in node 1
    ...
    load node n

As a reward we are currently using 1 - % instant failures in a time window.

We are also using a target network for better convergence and a buffer to store past samples, from which we sample when training.

Experiments



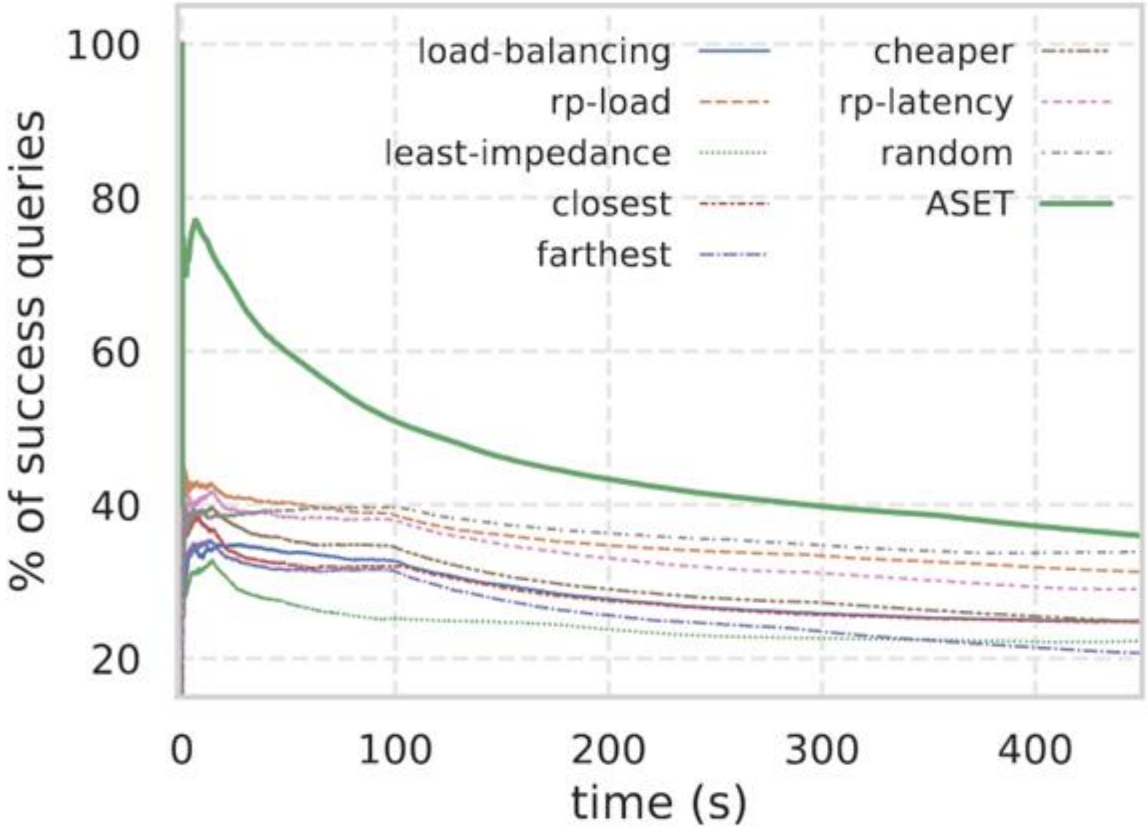
(a) Queries handled successfully.
(b) Different clients rate on full-edge.
(c) Queries delivered with QoS violations.
(d) Queries rejected for lack of resources.

Fig. 6: Performance of ASET compared with static policies for the full-edge topology. (a) (c) and (d) show averages of multiple runs with $\lambda = 60$.

Experiments

Cluster	Worker	RAM	CPUs	GPU
Cluster 1	Worker 1	16 GB	8	8 GB VRAM
	Worker 2	16 GB	8	8 GB VRAM
Cluster 2	Worker 1	16 GB	8	8 GB VRAM
Cluster 3	Worker 1	8 GB	4	8 GB VRAM
	Worker 2	8 GB	4	8 GB VRAM

TABLE III: Cluster and Worker Configuration for the real-deployment scenario.



(b) Time avg on a *real GPU-equipped* dc-cloud for $\lambda = 60$.

ASET schedules efficiently multi-tenant machine learning tasks in the Computing Continuum

Telefónica Innovación Digital

FLUIDOS
Grant Agreement No.: 101070473

EDGELESS
Grant Agreement No.: 101092950

NebulOuS
Grant Agreement No.: 101070516