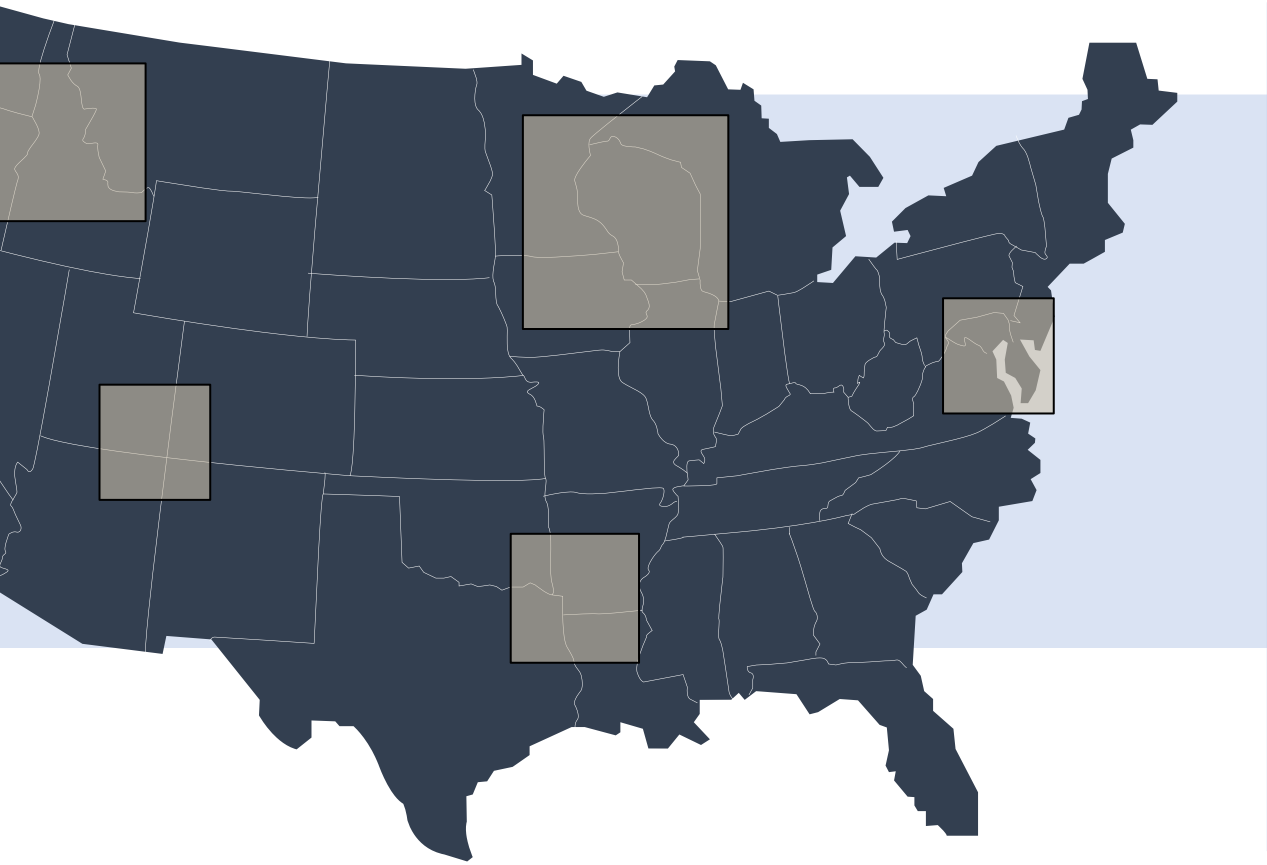
Balancing Energy Efficiency and Infrastructure Knowledge in Cloud-to-Edge Task Distribution Systems

Stefano Galantino (Politecnico di Torino)
Andrea Pinto (Saint Louis University)
Flavio Esposito (Saint Louis University)
Antonio Manzalini (Telecom Italia Mobile (TIM))
Fulvio Risso (Politecnico di Torino)

THE COMPUTING SCENARIO



The computing continuum allows to create **geographically distributed** infrastructures - from edge devices like smartphones and IoT (Internet of Things) sensors to central cloud data centers. This spectrum ensures that computation can occur anywhere along this continuum, based on the most efficient location for processing.

The use of renewable energy sources, energy-efficient hardware, and advanced software algorithms for workload management can significantly **reduce the environmental footprint** of computing resources across the continuum.

CENTRALIZED vs DISTRIBUTED

CENTRALIZED

Centralized task placement in computing infrastructures concentrates workload management and resource allocation in a single, authoritative system, streamlining decision-making for efficiency.

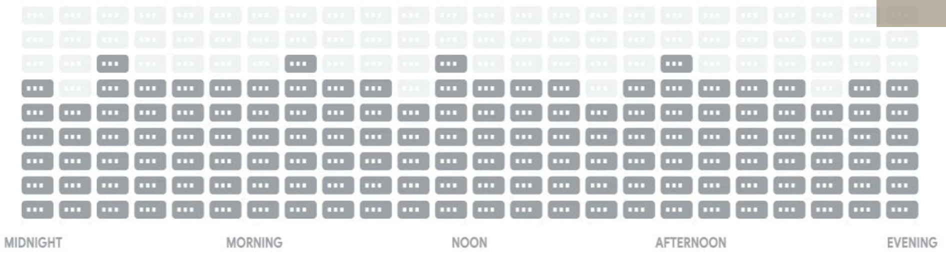
DISTRIBUTED

Distributed task placement in computing infrastructures distributes the workload management in each node participating in the federation, so that each node can (potentially) perform the task allocation phase.

LIMITATION OF CENTRALIZED

HORIZONTAL SCALABILITY

The complexity of the centralized allocation algorithm scales with the number of compute nodes. Heuristics can speed up the convergence, but compromise on the quality of the final allocation.

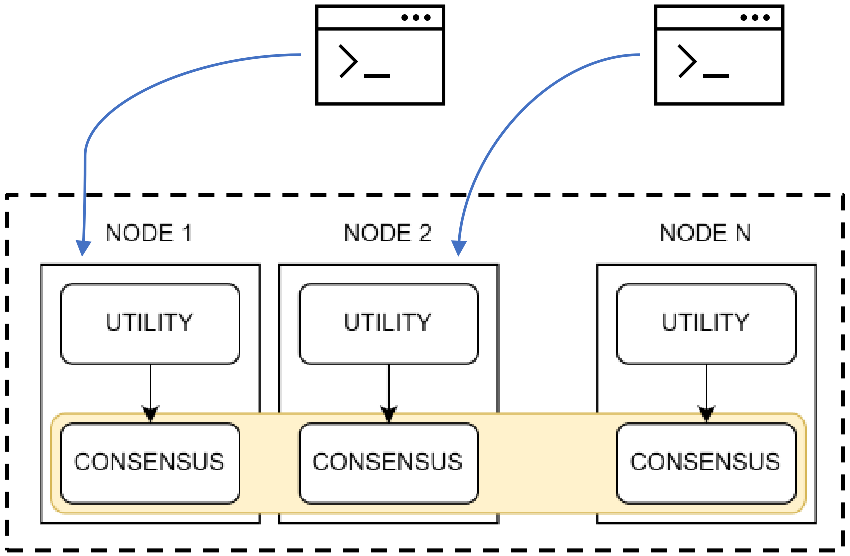


COMPLETE KNOWLEDGE

Centralized allocation algorithms require complete knowledge on the infrastructure to perform the allocation. This means that all the nodes must be willing to share such information.

SYSTEM ARCHITECTURE (I)

On top of this continuum infrastructure, applications must be **efficiently allocated** (i.e., satisfying all the execution requirements), and nodes should **compete** to identify the most suitable location to host the execution.
To this end, each node participating in the continuum is equipped with the following:

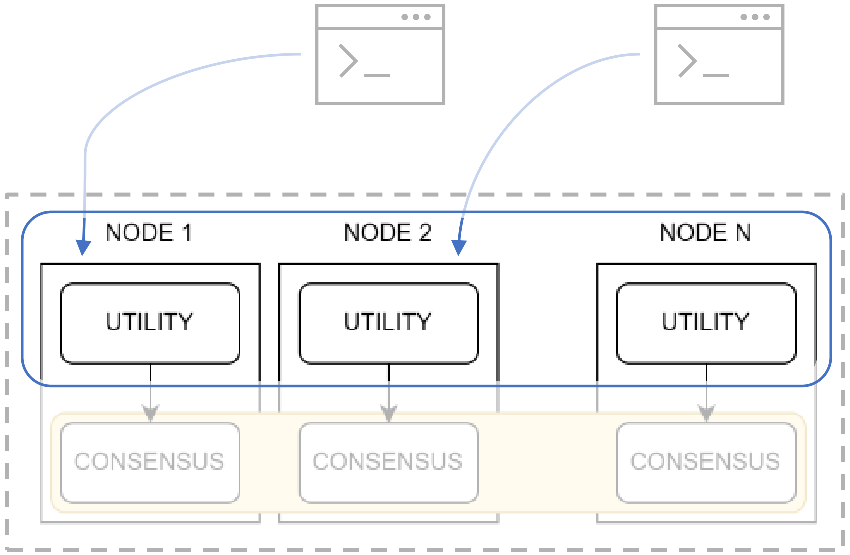


Utility function, that maps how much a given node is willing to host one application or a part thereof (in the case of microservice-based applications)

Distributed consensus-based RAFT-like algorithm to reach a consensus among the participants in the continuum to select the node with the highest utility value.

SYSTEM ARCHITECTURE (II)

We assume that each hosting node n has a (private) utility function U_n , and we are seeking an allocation solution that maximizes the sum of the utilities of all nodes



$$
\begin{aligned}
\max_{x} \quad & \sum_{n \in \mathcal{N}} \sum_{j \in \mathcal{J}} \sum_{m \in \mathcal{M}_j} U_n(y_j) & \\
\text{s.t.} \quad & \sum_{j \in \mathcal{J}} \sum_{m \in \mathcal{M}_j} y_j r_{m_j} \leq c_n & \forall n \in \mathcal{N} \\
& \sum_{n \in \mathcal{N}} \sum_{m \in \mathcal{M}_j} y_j = M_j & \forall j \in \mathcal{J} \\
& \sum_{m \in \mathcal{M}_j} y_j \leq 1 & \forall n \in \mathcal{N}, \forall j \in \mathcal{J}
\end{aligned}
$$

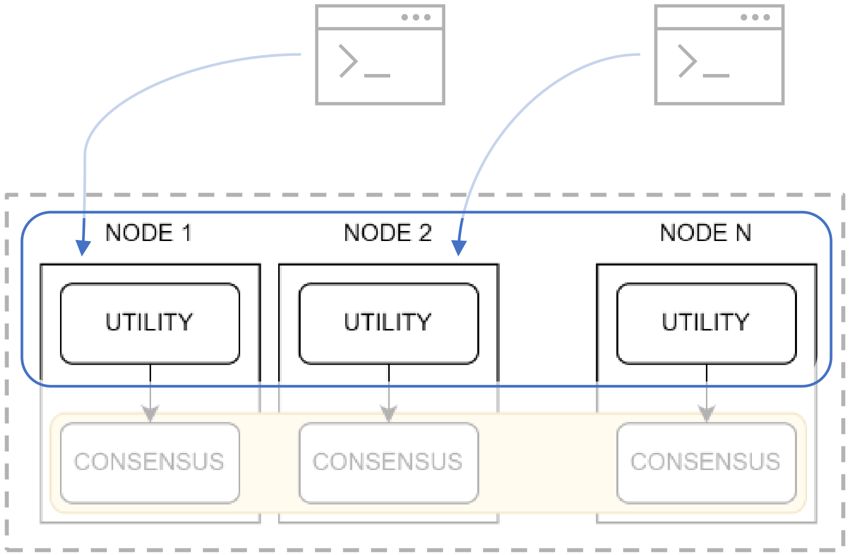
In the scope of this work, we design the utility function for a generic node n to be as follows:

$$
U_n(x) = \frac{1}{P_n(x) - P_n(x_0)}
$$

where $U_n(x)$ is a function of the CPU usage x.

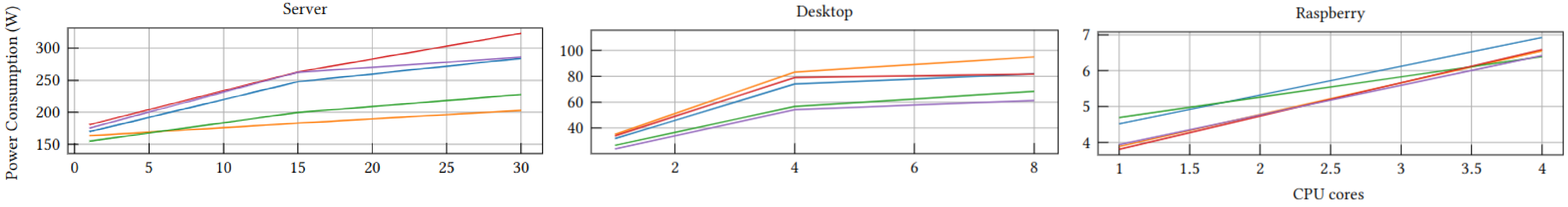
SYSTEM ARCHITECTURE (III)

Starting from real measurements, we extrapolated the results to obtain an accurate mathematical model.



$$P(x) = \begin{cases} \alpha x + \beta, & \text{if } 0 < x \leq \rho \\ \delta x + (\alpha\rho + \beta - \delta\rho), & \text{if } \rho < x \leq 2\rho \end{cases}$$

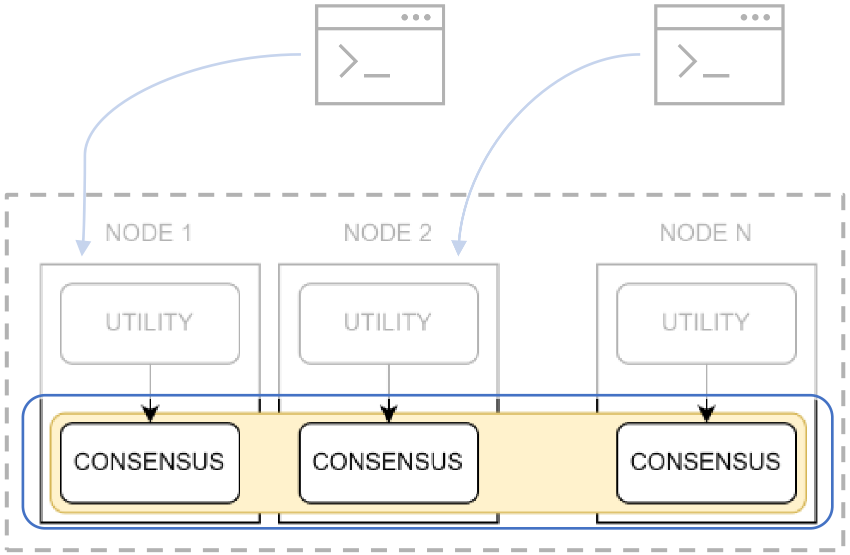
where ρ represents the number of physical cores of the system. In detail, devices experience different power consumption patterns, depending on whether or not the current CPU usage x exceeds the number of physical cores (i.e., logical cores are also required to sustain the load).



SYSTEM ARCHITECTURE (IV)

We then use a distributed consensus-based RAFT-like algorithm to reach a consensus among the participants in the continuum to select the node with the highest utility value.
Our distributed protocol operates in two phases:

- **Task dispatching**
- **Consensus**



DISPATCHING

A node receives the allocation request of an application → The node computes the value of the utility function →

CONSENSUS

The node share the output value of its utility function → The information of the highest utility is replicated among all the nodes in the continuum

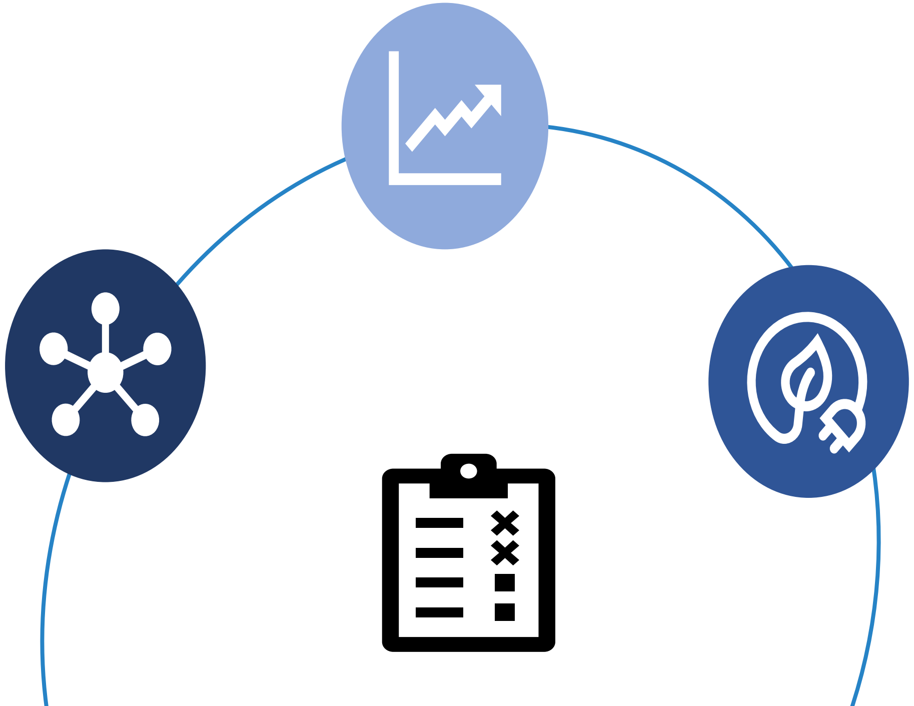
EVALUATION SETUP

INCREASE NOISY DEVICES

Evaluate the performance on our distributed allocation policy, increasing the number of devices in the process using the utility function described.

INFRASTRUCTURE

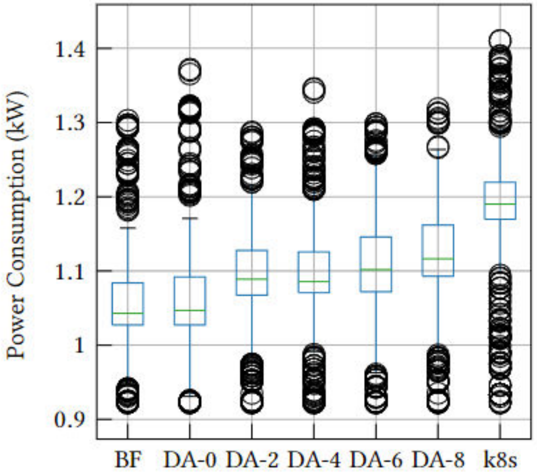
The infrastructure is composed of 5 Desktops, 5 Servers and 5 Raspberry Pis that generate requests to allocate tasks.
The submitted tasks have a randomized duration ranging from 60s to 140s depending on the scenario.



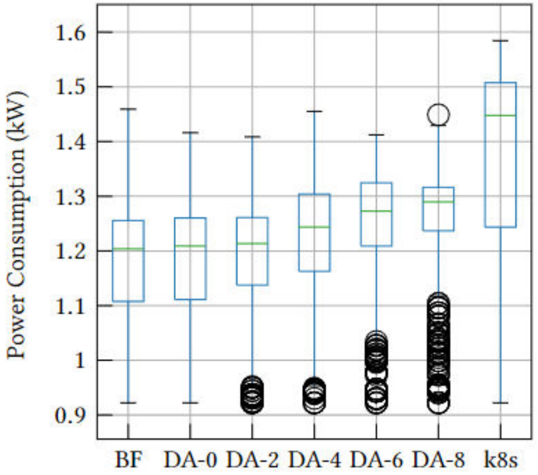
MEASURE POWER CONSUMPTION

Compare the results with the optimal online allocation policy to assess the performance of the proposed framework.

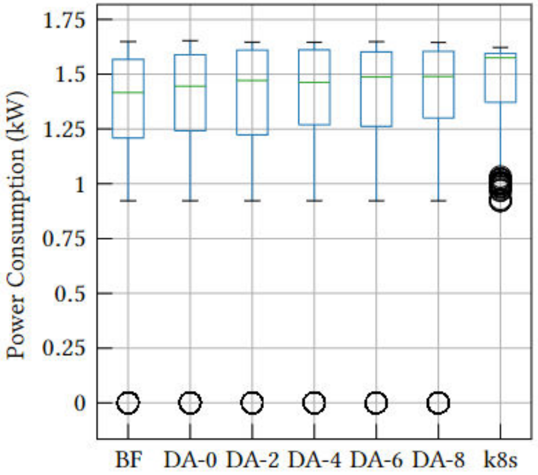
EVALUATION (I)



Duration range 1-60s



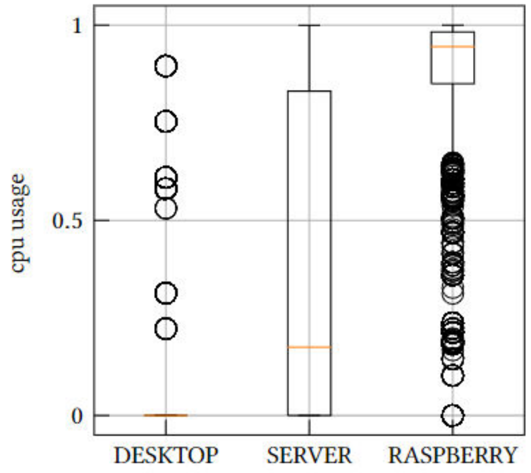
Duration range 1-100s



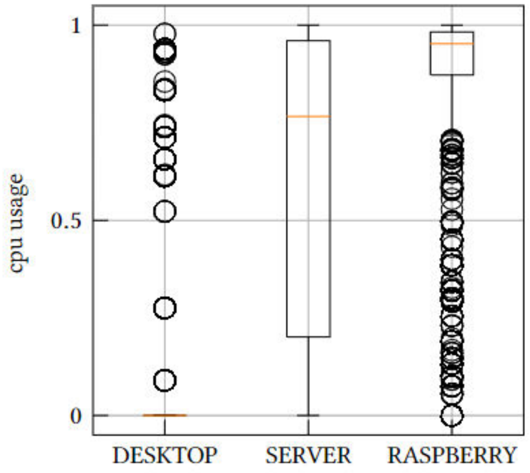
Duration range 1-140s

- If all nodes use the energy-aware utility function (DA-0), it is possible to obtain almost the same results as the optimal brute force approach.
- If we increase the number of devices not participating in the energy-aware allocation, the gap increases, but it is always possible to achieve between 5% and 20% of energy savings w.r.t. the energy-unaware allocation.
- With highly saturated infrastructures the possible solutions for the allocation problem are extremely limited, resulting in almost comparable results for all the configurations.

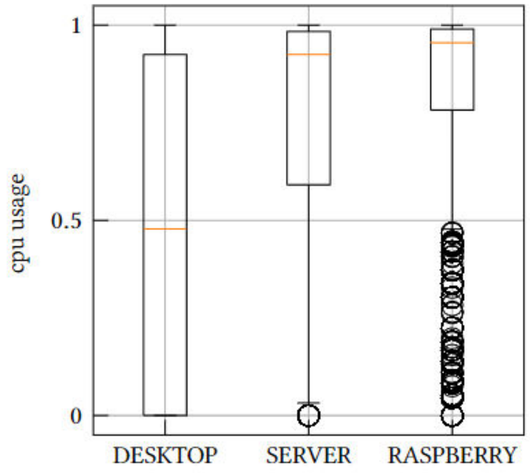
EVALUATION (II)



Duration range 1-60s



Duration range 1-100s



Duration range 1-140s

The energy-aware allocation policy can **preemptively select only the most efficient nodes** for the job execution while leaving the less efficient ones only in case of saturation of resources.
For the case of DA-0, with low saturated infrastructure, our proposal relies almost entirely on servers and Raspberry PIs to host the submitted tasks (being the most energy-efficient devices of the lot).
Instead, if we increase the load submitted to the infrastructure, the system has to select less energy-efficient nodes to host the workload.

CONCLUSIONS

Framework Overview:

- Distributed framework presented for optimizing energy consumption in heterogeneous computing environments.
- Utilizes custom utility functions to optimize task allocation while minimizing energy consumption.
- Ensures privacy by bypassing the need to share private resource information.

Key Findings:

- Outperforms traditional centralized allocation policies and Kubernetes-like scheduling algorithms.
- Significant improvements in energy efficiency observed without compromising node privacy.

Future Directions:

- Evaluation of temporal shifting of workloads alongside geographical shifting to enhance outcomes.
- Incorporation of energy proportion data (green vs. brown energy) to promote sustainable infrastructure.
- Trade-off analysis between Quality of Experience (QoE) and energy consumption for high-priority applications.

THANKS!

Stefano Galantino

Email: stefano.galantino@polito.it

Website: stegala.github.io