Real-time and Energy-aware Scheduling for Edge-to-Cloud Continuum based on Reinforcement Learning

A. Panceri, G. Proietti Mattia, R. Beraldi

Department of Computer, Control and Management Engineering "Antonio Ruberti"
Sapienza University of Rome

DS-RT '24, Urbino, Italy - October 9, 2024

Table of Contents

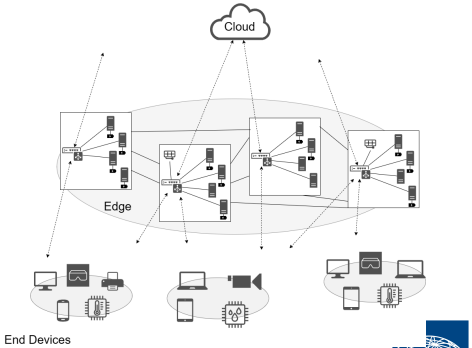
Table of Contents

The Edge-to-Cloud continuum

The Edge (Fog) Computing paradigm has few relevant characteristics:

- **real-time** processing with **low-latency** requirements;
- **privacy preserving** computation;
- **distributed** nature of the computation.

We call it *continuum* when the Edge computing layer is also assisted by the Cloud one.



Motivation and Related Works

We suppose that Edge nodes are powered by **batteries** and they continuously receive tasks to be processed. The main aim of this work is to design a **real-time** scheduling algorithm which is able to **maximize** the lifespan of the nodes and the number of **satisfied in-deadline** requests.
This work is a continuation of[1], with respect major works in literature[2][3] we offer a real-time task modeling with a RL approach that is more suitable for real-time decision making.

---

[1] Gabriele Proietti Mattia and Roberto Beraldi. "Leveraging Reinforcement Learning for online scheduling of real-time tasks in the Edge/Fog-to-Cloud computing continuum". In: *2021 IEEE 20th International Symposium on Network Computing and Applications (NCA)*. 2021, pp. 1–9. DOI: 10.1109/NCA53618.2021.9685413.

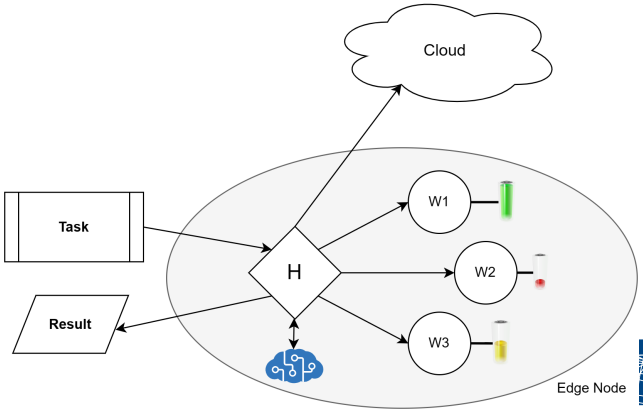
[2] Yu Sui and Shiming Song. "A multi-agent reinforcement learning framework for lithium-ion battery scheduling problems". In: *Energies* 13.8 (2020), p. 1982.

[3] Xiong Xiong et al. "Resource allocation based on deep reinforcement learning in IoT edge computing". In: *IEEE Journal on Selected Areas in Communications* 38.6 (2020), pp. 1133–1146.

Table of Contents

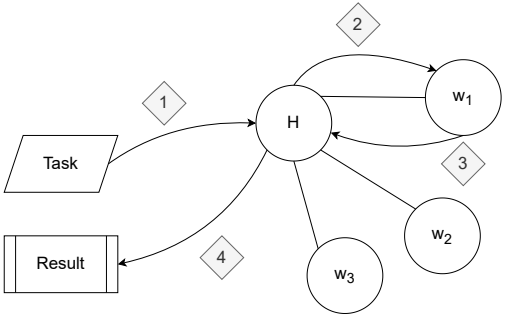
Environment

- Task Management.
- Request Handling.
- Worker Node Characteristics.
- Power Considerations.



Task and Delay models

- Task generated and added to scheduler node H queue (using bandwidth B_{cs}).
- Scheduler H forwards task to worker node w1 (using bandwidth B_{sw}).
- Task result returned to H.
- Result finally sent to client (using same bandwidth and payload size).



The Agent

- Primary objective is to learn an effective scheduling policy, denoted as π, which is a function of the current state:

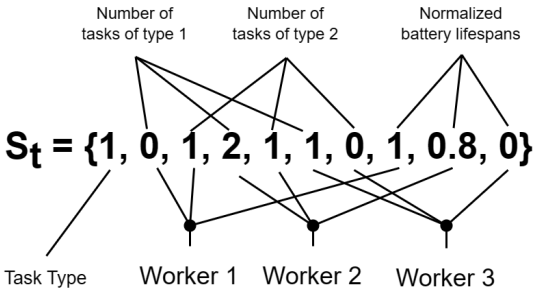
$$\pi : S \rightarrow A \tag{1}$$

- The policy π maps a given state $s \in S$ to an action $a \in A$, where:

$$A = \{reject, cloud\} \cup W_i \tag{2}$$

- Multi-objective reinforcement learning framework, we incorporate a parameter $\alpha \in [0, 1]$. This parameter allows for the adjustment of reward criteria, providing flexibility in optimizing various objectives simultaneously.
- The value 0 prioritize batteries management, 1 prioritize fps requirements.

State Representation



- The raw state representation is not directly utilized in the learning process. Instead, we employ the tiling technique to map the vector into a 24-dimensional vector space.

Reward

$$\text{reward} = \text{reward\_fps} \times \alpha + (1 - \alpha) \times \text{reward\_batteries} \tag{3}$$

- The parameter α plays a crucial role in this formulation as it enables the prioritization between the two performance measures. By selecting an appropriate value for α, we can optimize the results and achieve the best tradeoff between FPS performance and battery lifespan.

Differential Semi-Gradient Sarsa

The optimal policy, which maximizes the long-term reward, is defined by the optimal q_* function, as shown below:

$$q_*(s,a) = \sum_{r,s'} p(s', r|s, a) \cdot \left[r + \max_{\pi} r(\pi) + \max_{a'} q_*(s', a') \right] \tag{4}$$

The Sarsa algorithm is used to learn the policy, and at a certain time t, the differential form of the error, δ_t:

$$\delta_t = R_{t+1} - \overline{R}_{t+1} + \hat{q}(S_{t+1}, A_{t+1}, \vec{w}_t) - \hat{q}(S_t, A_t, \vec{w}_t) \tag{5}$$

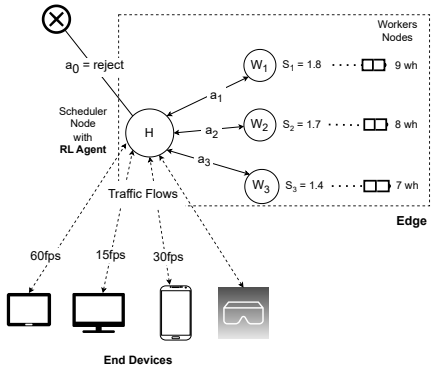
The reward_fps is never immediate, as it is only known after a task has been executed or rejected and returned to the client.

Performance Parameters

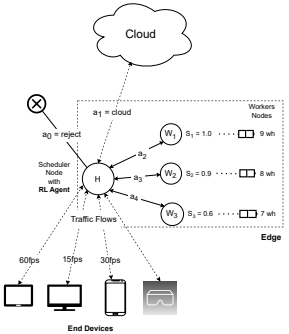
- α: Represents the alpha value used, with $\alpha \in [0, 1]$.
- σ: Signifies the variance of battery WH difference, indicating the spread of the differences in battery capacity ($\downarrow$).
- δ: Represents the difference between the maximum and minimum WH (Watt-hour) when the first node discharges ($\downarrow$).
- M: Denotes the maximum lifespan of a component or system, while m represents the minimum lifespan. Higher M and higher m indicate longer lifespans ($\uparrow$)
- γ: Indicates the percentage of jobs meeting requirements, reflecting the efficiency or success rate of a system ($\uparrow$).
- γ_i: Represents the percentage of jobs meeting requirements of a specific type i, providing insight into the performance of different job types ($\uparrow$).
- ts: Represents the total service time of all workers, reflecting the overall workload or processing time ($\uparrow$).

Table of Contents

Setting

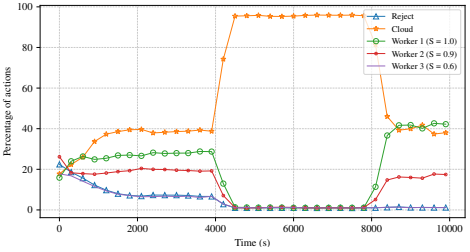


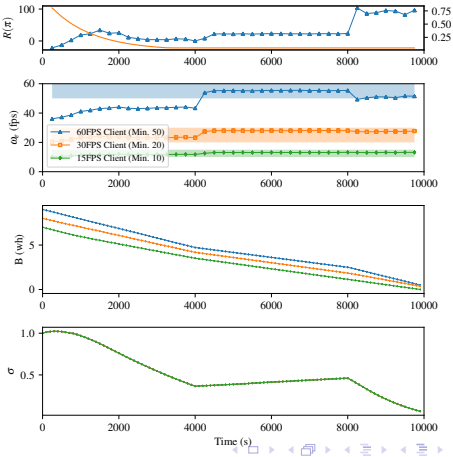
(a) Only Workers



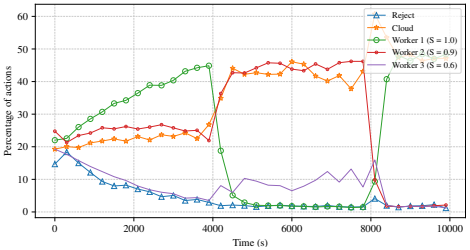
(b) Edge-to-cloud continuum

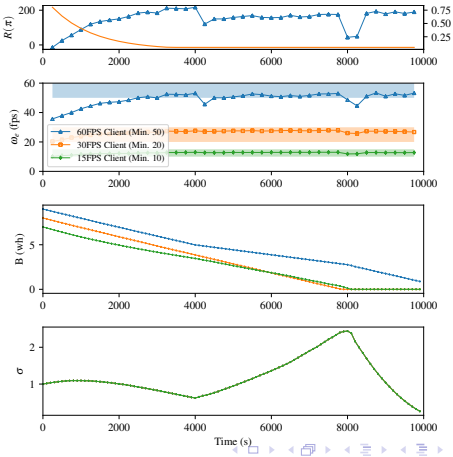
Adaptability to change (α = 0.0)





Adaptability to change (α = 1.0)



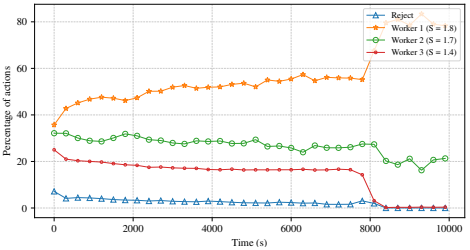


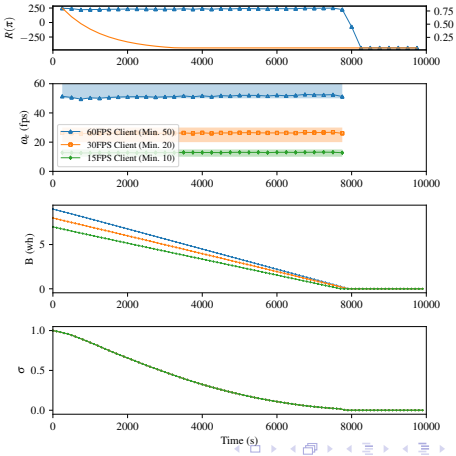
Only Workers - Table

α	$\sigma \downarrow$	$\delta \downarrow$	$m \uparrow$	$M \uparrow$	$\gamma \uparrow$	$\gamma_0 \uparrow$	$\gamma_1 \uparrow$	$\gamma_2 \uparrow$	ts $\uparrow$
0.00	**0.08**	**0.00**	8055	8056	70.9	18.5	94.2	**99.9**	24166
0.20	0.08	0.10	7870	7957	82.2	49.3	98.5	**99.9**	23784
0.30	0.10	0.24	7711	7912	91.3	74.5	**99.4**	**99.9**	23533
0.40	0.14	0.49	7541	7941	91.8	76.7	98.8	**99.9**	23415
0.50	0.14	0.48	7548	7942	94.5	84.9	98.7	**99.9**	23385
0.60	0.14	0.47	7534	7926	**95.6**	**88.7**	98.4	99.9	23325
0.70	0.22	0.88	7348	8077	92.7	84.8	94.2	99.2	23224
0.80	0.31	1.22	7175	8202	90.1	80.4	90.6	99.2	23120
0.90	0.33	1.25	7197	8259	89.0	78.3	89.2	99.5	23097
1.00	0.37	1.38	7110	8306	88.5	78.3	88.3	98.9	23054
LL	0.41	1.48	7167	8384	84.67	64.8	89.2	99.9	23347
MLIF	0.12	**0.00**	**9452**	**9452**	41.1	0.3	25.0	98.1	**28356**
RAND	1.130	2.62	7155	8804	39.39	0.00	34.92	83.25	15959

Table Results of the experiment in the case of only workers with batteries values $B_1 = 9$ Wh, $B_2 = 8$ Wh, and $B_3 = 7$ Wh.

Only Workers - (α = 0.3)



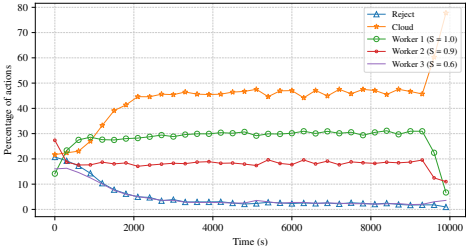


Edge-to-Cloud continuum - Table

α	$\sigma \downarrow$	$\delta \downarrow$	$m \uparrow$	$M \uparrow$	$\gamma \uparrow$	$\gamma_0 \uparrow$	$\gamma_1 \uparrow$	$\gamma_2 \uparrow$	ts $\uparrow$
0.00	0.13	**0.00**	8545	8546	65.2	25.2	81.9	90.6	25636
0.10	0.14	**0.00**	8877	8877	72.6	45.3	83.0	89.4	26631
0.20	0.16	**0.00**	9200	9201	83.7	62.1	91.4	97.2	27602
0.30	0.17	0.01	**9462**	**9471**	92.8	80.4	**97.8**	99.9	**28403**
0.40	0.17	**0.00**	9380	9385	**93.2**	**81.9**	**97.8**	**100**	28149
0.50	0.17	**0.00**	9076	9077	92.3	79.4	97.6	100	27230
0.60	0.15	**0.00**	8715	8717	90.9	75.6	97.3	99.9	26147
0.70	0.15	0.20	8261	8436	88.9	71.2	95.6	**100**	25069
0.80	0.21	0.57	8064	8577	89.5	72.4	96.3	99.8	24961
0.90	0.29	1.08	7590	8652	92.0	79.1	97.1	99.9	24727
1.00	0.32	1.19	7517	8671	90.6	75.1	96.8	99.8	24389
LLAC	0.20	1.1	6272	7081	88.3	70.9	93.9	99.9	20221
MLIF	**0.11**	**0.0**	9218	9219	16.0	0.0	0.1	48.0	27656
RAMD	1.10	3.1	5635	8669	29.9	0.0	15.1	74.6	21672

Table Results of the experiment in the case of three workers with batteries values B_1 = 9 Wh, B_2 = 8 Wh, and B_3 = 7 Wh, and cloud available.

Edge-to-Cloud continuum - (α = 0.3)



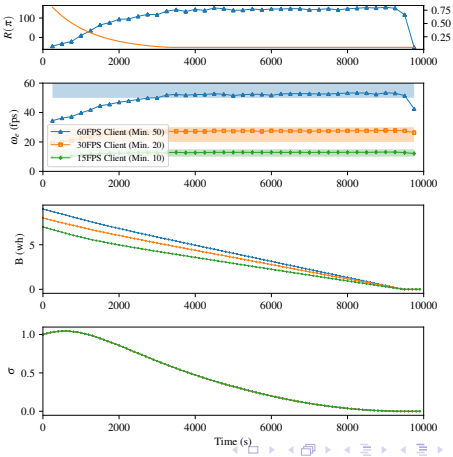


Table of Contents

Conclusions and Future Work

With this work we showed that:

- RL is suitable for **dynamic task scheduling** in resource-constrained environments.
- Our approach allows us to decide the **trade-off** between energy management and real-time feasibility.

Future research directions can regard:

- Implementation on **real Edge devices** to validate efficacy.
- Consideration of more **advanced** RL techniques (e.g., policy gradient methods) but always keeping them as lightweight as possible.