Darly: Deep Reinforcement Learning for QoS-aware scheduling under resource heterogeneity Optimizing serverless video analytics

Dimitrios Giagkos, Achilleas Tzenetopoulos, Dimosthenis Masouros, Dimitrios Soudris, Sotirios Xydis
Microprocessors Laboratory and Digital Systems Lab (MicroLab)
School of Electrical and Computer Engineering, National Technical University of Athens
Email: dimgiagos@gmail.com, {atzenetopoulos, demo.masouros, dsoudris, sxydis}@microlab.ntua.gr

***Abstract*—Today, video analytics are becoming extremely popular due to the increasing need for extracting valuable information from videos available in public sharing services through camera-driven streams. Typically, video analytics are organized as a set of separate tasks, each of which has different resource requirements (e.g., computational- vs. memory-intensive tasks). The serverless computing paradigm forms a very promising approach for mapping such types of applications, as it enables fine-grained deployment and management in a per-function manner. However, modern serverless frameworks suffer from performance variability issues, due to *i)* the interference introduced due to the co-location of third-party workloads with the serverless functions and *ii)* the increasing hardware heterogeneity introduced in public clouds. To this end, this work introduces Darly, a QoS- and heterogeneity-aware *Deep Reinforcement Learning-based Scheduler* for serverless video analytics deployments. The proposed framework incorporates a DRL agent which exploits low-level performance counters to identify the levels of interference and the degree of heterogeneity in the underlying infrastructure and combines this information along with user-defined QoS requirements to dynamically optimize resource allocations by deciding the placement, migration, or horizontal scaling of serverless functions. Promising results are produced within our experiments, which are accompanied by the intent to further build upon this groundwork.**

***Index Terms*—Cloud computing, Serverless Computing, Deep Reinforcement Learning, Quality-of-Service, Dynamic Scheduling, Resource Management**

I. Introduction

Video traffic has already been and is projected to be further increased over the next years [1]. Video analytics are typically offloaded to the Cloud, due to the quasi-unlimited computing capacity it offers. Serverless computing is an emerging paradigm offering a very high-level abstraction of the cloud infrastructure to end-users. However, it comes with decreased control over the infrastructure itself, leading to limitations regarding the efficient management of resources that often result in Quality of Service (QoS) violations, due to the high degree of performance variability [5].

Serverless workloads are managed by open-source runtimes like OpenFaas and Openwhisk, leveraging container orchestrators such as Kubernetes for scheduling and deployment. Nonetheless, workload orchestrators (e.g., the native Kubernetes scheduler) usually apply resource management decisions once, at the beginning of each job, neglecting future system states or QoS targets. State-less scheduling decisions for meeting QoS invest in resource over-provisioning, application workload modelling [2] and horizontal/vertical scaling [6] but occasionally fail due to i) the heterogeneity of hardware resources [4], ii) interference phenomena in multi-tenant environments [7]; as a consequence of the resource sharing and contention, the workload execution interferes with the execution of other applications and iii) unawareness of workload specific features both in function and workflow scope [8].

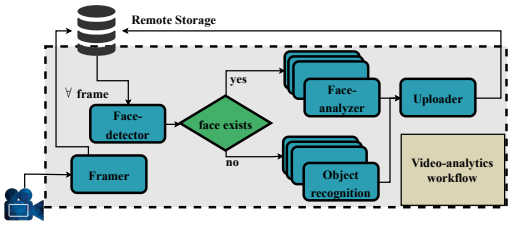


Fig. 1: Video-analytics workflow

Improving scheduling efficiency in modern cloud environments needs continuous feedback loops for boosting the algorithm's heterogeneity, interference and performance awareness and thus ease informed decision-making for serving multiple QoS requirements. Deep Reinforcement Learning (DRL) forms a very effective solution in modelling environmental variability so as to derive orchestration strategies for online management of serverless workflows.

We present *Darly*, a DRL-based scheduling framework for managing video analytics pipelines in serverless infrastructures, which in principle can be also applied by design to granular serverless workflows from other domains. *Darly* exploits low-level system metrics to identify underlying cluster interference and along with user-defined QoS requirements, aims to regulate end-to-end latency of video analytics pipelines, through horizontal scaling and migration of the pipeline's functions. Our solution manages to dynamically orchestrate the deployed functions under various run-time conditions, i.e., system-level dynamic resource fluctuations due to interference and/or dynamically changing QoS criteria.

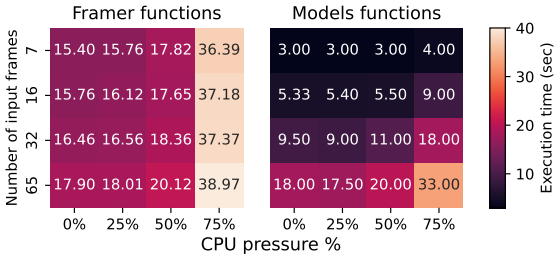


Fig. 2: Impact of interference on serverless functions

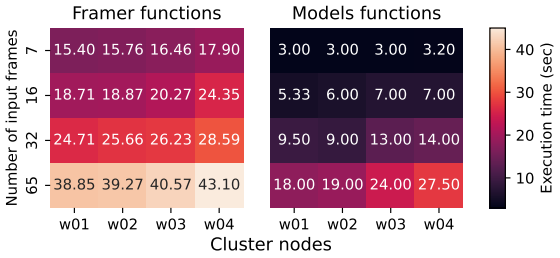


Fig. 3: Impact of heterogeneity on serverless functions

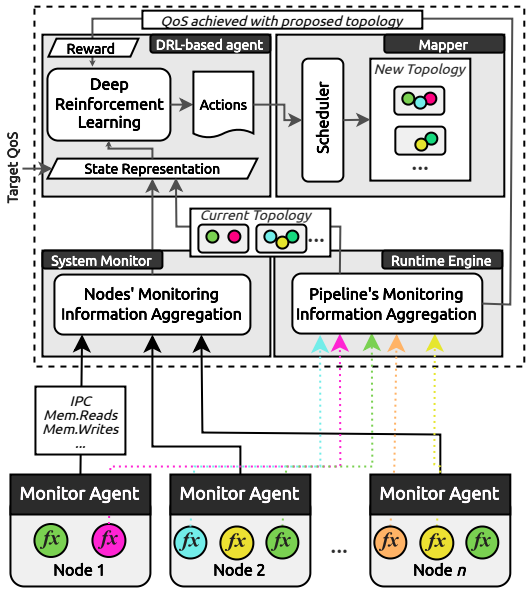


Fig. 4: Framework overview

II. Motivation

We develop a video-analytics workflow reflecting a real-world pipeline [9], that performs computer vision (CV) inference in frames of a video. We explore interference's and heterogeneity's influence in its performance under various scenarios, in a cluster of four Virtual Machines (VMs) deployed on top of an on-premise, highly-heterogeneous, high-end server infrastructure. Our pipeline is represented as a DAG in Fig. 1, and consists of 5 separate functions: i) *Framer* which extracts frames from the input `mp4` video file, ii) *Face-detector* which detects whether a human face exists or not in a frame and forwards the processed frame to iii) or iv), iii)*Face-analyzer* that performs emotion recognition to a detected human face, iv) *Object-recognition* which classifies objects existing in the frame and v) *Uploader* that aggregates results from iii), iv) and uploads them to remote storage. We invoke our pipeline with four distinct input sizes and measure the average execution latency for the individual functions, which sum up to the end-to-end latency.

Impact of interference: We spawn different amounts of `cpu` microbenchmarks from the iBench suite [3] and apply four levels of interference to w01: 0%, 25%, 50% and 75% of the total available cores, as portrayed in Fig. 2. Great, non-linear performance variability is presented w.r.t CPU interference that reaches up to 57.6% for the 16-frames input and up to 47.2% for the 32-frames input in the *Framer* and CV models functions respectively.

Impact of heterogeneity: Fig. 3 shows the performance variation of the examined functions, w.r.t. hardware heterogeneity. VMs with fewer cores, provide less multi-threading capacity to the hosted functions which result in poorer latency. For the *Framer* function we find deltas with a maximum value of 23% and a minimum of 10% performance variation in the 16-frames and 65-frames inputs respectively. For the ML-models functions, the measured deltas have a maximum value of 34% and a minimum of 5% variations respectively.

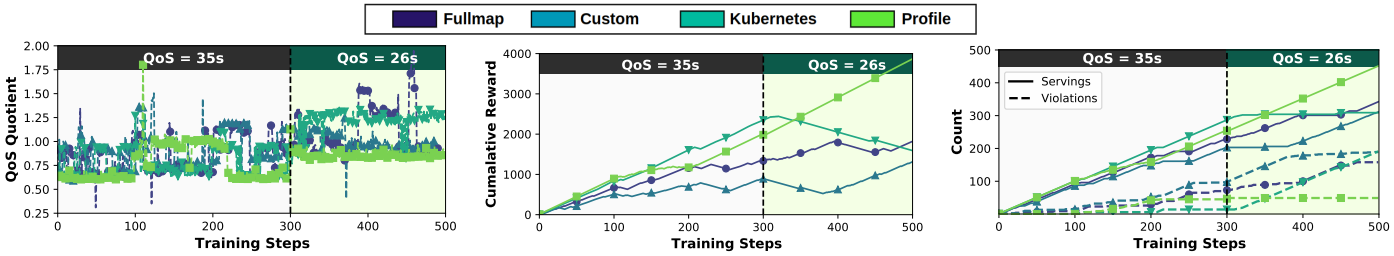
III. *Darly*: A Dynamic DRL-based Scheduler

We design a dynamic scheduler for modern cloud environments that is aware of node heterogeneity, resource interference from third-party co-located workloads and resistant to fluctuations caused by unpredictable user demand. Based on the findings outlined in Section II, we leverage DRL to implement *Darly*, a dynamic DRL-based scheduler for the widely adopted open-source serverless platform, OpenFaas.

Our scheduler receives a video-request with a QoS constraint and after scanning the cluster state, orchestrates the workflow functions' topology in order to optimize resource utilization and regulate end-to-end latency without exceeding the user-defined QoS. The proposed framework, shown in Fig. 4, consists of four components: a `System Monitor` which collects low-level metrics (i.e., IPC, L3-cache misses, Memory Reads/Writes, C-states) depicting the system's state, a `DRL-based Agent` which reads the system metrics and calculates the next action to be performed on the deployed functions regarding the specified QoS, a `Runtime Engine` that given the current functions' placement accommodates the execution of workflow instance and a function `Mapper` which maps a function to a node according the agent's latest decision.

The DRL-based agent, utilizing a Deep Q-Network (DQN), interacts with the *environment* at discrete time steps t and aims to maximize the received reward over time by choosing an action A_t among a discrete set of available functions and force the transition of the environment to a new state.

Action set A: Includes: i) per function *horizontal-scaling*, ii) function *migration* to different nodes or iii) *inactivity*, i.e., preserving the function topology as is.



(a) QoS quotient over training (b) Cumulative reward over training (c) QoS violations over training

Fig. 5: Comparative evaluation of different schedulers during training of the DRL

Rewarding Strategy: The incentive behind the reward function (Eq. 1) is the regulation of the execution latency (L_a) by striving not to violate the latency threshold set by the user (L_t) while attempting to minimize both the number of utilized servers (sp) and the replica count (r) for each function.

$$R = \begin{cases} \frac{N}{sp} + \frac{Rs}{r} + \frac{L_a}{L_t} \times k_1, & \text{if } L_a \leq L_t \\ \max(-k_2, -k_3 - \frac{L_a}{L_t}), & \text{otherwise} \end{cases} \quad (1)$$

IV. Evaluation

We evaluate our framework w.r.t its efficiency to identify the appropriate actions for satisfying the pre-defined latency constraint while allocating the least amount of resources. We quantify the scheduler's performance by the `QoS quotient` (i.e., execution latency achieved divided by the user-defined QoS), agent's `cumulative reward` over time and `QoS violations` that are depicted in Fig. 5.

Experimental Conditions: We set two QoS levels, relying on the performance characterization presented in Sec. II, i.e., 35 and 26 seconds, which correspond to looser and stricter constraints respectively. Training on discrete levels of QoS is essential for exposing the DRL-based agent on a wide enough spectrum of states to facilitate its ability in identifying patterns among ($state$, $action$) pairs. During the experiments we dynamically change the underlying interference on the cluster, by randomly altering the number of `cpu` micro-benchmarks per VM in the same way as explained in Sec. II.

Examined Schedulers: We examine four different schedulers, so as to determine the inter-relationship between the DRL-agent's proposed actions and the employed scheduling mechanism. We aim to quantify the impact of i) the scheduling granularity when migrating functions and ii) heterogeneity- and interference-awareness with our proposed framework. Specifically, we developed four distinct schedulers, all differing in their actionspace: Fullmap-based, Custom-based, Kubernetes-based, Profile-based. The former two decide both the migration and the destination of a function, while the latter two decide just whether a function should be migrated or not and a third-party scheduler, Kubernetes and Profile respectively, locates the migrating function to a node, with its own policy. The Profile-based scheduler leverages offline profiling information (Sec. II) of the performance of deployed functions and decides the optimal scheduling policy accordingly.

Results: As depicted in Fig. 5a, the Custom- and Profile-based schedulers manage to adapt effectively to all changes in resource stress levels while regularizing the `QoS quotient` (i.e., remaining close to the upper bound of value of 1). A similar but less stable result is achieved by the fullmap-based scheduler which, due to a larger action space, is less prone to convergence. Last comes the Kubernetes-based approach due to its heterogeneity- and interference-unawareness fails to adjust its migration decisions to the occurring conditions.

V. Acknowledgements

This work has been partially funded by EU H2020 Research and Innovation programme AI@EDGE under Grant Agreement No 101015922 (https://aiatedge.eu/) and supported by the Hellenic Foundation for Research and Innovation (HFRI) under the 3rd Call for HFRI PhD Fellowships (Fellowship Number: 5349).

References

[1] Video analytics market by component (software services), application (facial recognition, video telematics), end-user (bfsi, education, healthcare, government) - global forecast to 2029. Website.

[2] Anirban Das, Andrew Leaf, Carlos A. Varela, and Stacy Patterson. Skedulix: Hybrid cloud scheduling for cost-efficient execution of serverless applications, 2020.

[3] Christina Delimitrou and Christos Kozyrakis. ibench: Quantifying interference for datacenter applications. In *2013 IEEE international symposium on workload characterization (IISWC)*, pages 23–33. IEEE.

[4] Christina Delimitrou and Christos Kozyrakis. Paragon: Qos-aware scheduling for heterogeneous datacenters. *ACM SIGPLAN Notices*, 48(4):77–88, 2013.

[5] Samuel Ginzburg and Michael J. Freedman. Serverless isn't server-less: Measuring and exploiting resource variability on cloud faas platforms. In *Proceedings of the 2020 Sixth International Workshop on Serverless Computing*, WoSC'20, page 43–48, New York, NY, USA, 2021. Association for Computing Machinery.

[6] Swaroop Kotni et al. Faastlane: Accelerating Function-as-a-Service workflows. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pages 805–820. USENIX Association, July 2021.

[7] Dimosthenis Masouros et al. Rusty: Runtime interference-aware predictive monitoring for modern multi-tenant systems. *IEEE Transactions on Parallel and Distributed Systems*, PP:1–1, 08 2020.

[8] Francisco Romero et al. Llama: A heterogeneous & serverless framework for auto-tuning video analytics pipelines. *CoRR*, abs/2102.01887, 2021.

[9] Miao Zhang, Fangxin Wang, Yifei Zhu, Jiangchuan Liu, and Zhi Wang. Towards cloud-edge collaborative online video analytics with fine-grained serverless pipelines. In *Proceedings of the 12th ACM Multimedia Systems Conference*, MMSys '21, page 80–93, New York, NY, USA, 2021. Association for Computing Machinery.