Learning-Based Anomaly-Tolerant Resource Management in Cloud-Edge Continuum

Submitted on: **23-02-2024**

Efstathios Trigkas
efstathios.trigkas@student.uva.nl
University of Amsterdam
Amsterdam, The Netherlands

Hongyun Liu
h.liu@uva.nll
University of Amsterdam
Amsterdam, The Netherlands

1 INTRODUCTION

In the ever-evolving landscape of cloud computing, the Cloud-Edge Continuum (CEC) has emerged as a pivotal architecture, promising unparalleled scalability, responsiveness, and efficiency. The edge to cloud continuum describes the distributed computing and network infrastructure management to meet the performance, security, and cost efficiencies of an edge ecosystem [8]. However, with the increasing complexity of cloud-edge continuum, more anomalies get involved in the resource management process, e.g. hardware failures and system dynamics making the task of resource management to be considered as an NP-hard problem [1].

Traditionally, in computer systems the way to tackle scheduling of resources like CPU, Memory and I/O devices etc. is by designing heuristic algorithms that prioritize finding good solutions quickly. Nevertheless, due to their fixed nature when anomalies arise they tend to under-perform. For that reason, there is a clear need for an anomaly-tolerant orchestrator.

Previous research has demonstrated that machine learning offers a feasible alternative to conventional resource management heuristics [2, 7, 9, 10] . These studies have shown that deep reinforcement learning can be applied to resource management problems and perform comparably to state-of-the-art heuristics. A notable attempt that of H. Mao [7] demonstrated the potential of DRL in their algorithm DeepRM which outperformed popular scheduling heuristics such as Shortest-Job-First. In a nutshell, inside the core of RL, an agent (i.e., the decision maker) interacts with the environment and learns automatically from the experience based on the reward from the environment. In the case of Resource management problem, the agent learns to automatically assign incoming jobs to computational resources by experience while optimize a variety of different goals, such as minimizing average job slowdown.

While optimizing scheduling policies for efficiently managing resources in the Cloud-Edge Continuum is crucial, the detection and integration of anomalies in the scheduling process is equally important, especially in scenarios where abnormalities are a frequent phenomenon. Although DRL-based scheduling algorithms have performed advantages in reviewed literature, they are only tested at the laboratory simulation experiments where tasks do not reflect the real variety of Cloud environment.

With this research, we aim to bridge this research gap by developing a system that addresses the challenges of resource management and anomaly tolerance simultaneously. In this paper we ask: *To what extent can a learning based resource scheduler become anomaly-tolernat in order to address the challenges of the Cloud-Edge Continuum ?*

The following research can be further divided into the following sub-questions:

- How well can an anomaly detection module be integrated in the resource management algorithm in order to detect anomalous workloads.
- How well can the resource management algorithm manage the incoming anomalous jobs and migrate them in the scheduling process while gradually optimizing their policy.

2 LITERATURE REVIEW

The proposed research will contribute to closing the research gap within the fields of learning based resource management and anomaly tolerance in computing systems. We start by reviewing the problem of machine learning methods for resource scheduling and suggested solutions in the literature. Subsequently, we introduce scientific work related to false tolerance and anomaly detection. Lastly, we conclude by summarising the relevant research in the overlap of these two areas.

2.1 Learning based resource management

Recent studies have demonstrated the effectiveness of Machine Learning based methods for resource scheduling. A paper that summarizes this field is that of G Zhou [10] . In their research they surveyed methods of resource scheduling with focus on DRL-based scheduling approaches in Cloud computing, reviewed the application of DRL as well as discussed challenges and future directions of DRL in scheduling of Cloud computing. This research concludes that DRL is one of effective methods to solve the dynamic resource scheduling of large-scale Cloud computing. However, the paper states several challenges that these algorithms encounter one of them being that they are only tested at the laboratory simulation experiments where tasks do not represent the complexity in real Cloud environments. Furthermore, real scheduling relies on predicting dynamic tasks without preemptive or prior knowledge. DRL-based methods fail to adequately address this challenge.

Another paper that is considered a state-of-the-art in the field of DRL-based scheduling is that of Mao [7]. In his research, he proposed an algorithm named DeepRL that is comparable and sometimes better than ad-hoc heuristics for a multi-resource cluster scheduling problems. He concludes that if we can make the model work in a practical context, this could offer a real alternative to current heuristic based approaches.

Their have been several studies that are based on Maos' work most notably the research of Yufei Ye [3] and Wenxia Guo [9]. Their work is based on DeepRM but their solutions have faster convergence speed and better scheduling efficiency with regarding to average slowdown time. This is achieved through the implementation of Imitating learning techniques to accelerate convergence of the model and convolution neural networks to better capture the state of resources.

2.2 False Tolerance and Anomaly detection

As the number of connected devices in the cloud-edge continuum continues to expand, its complexity has grown proportionally. This increased complexity has rendered the continuum more susceptible to anomalies, thereby reducing its fault tolerance.

As shown in previous research [6] the design of distributed systems places significant emphasis on fault tolerance. In the event of a hardware or software failure within the system, termed as a fault, the system's functionality is disrupted. To ensure uninterrupted operation despite these faults, techniques are employed to tolerate failures. These techniques aim to detect and correct errors, enabling the system to maintain its functionalities. According to their research their are two main directions for achieving false tolerance. Reactive fault tolerance techniques are used to reduce the impact of failures on a system when the failures have occurred and Proactive fault tolerance expects the faults proactively and places healthy (working) components in place of the faulty components, to prevent recovery from faults.

An essential component of fault tolerance revolves around effectively detecting anomalies. Considerable research has been devoted to this area, with S. Han's study [4] serving as a benchmark for anomaly detection algorithms. In their paper, they introduce ADBench, an anomaly detection benchmark with 30 algorithms and 57 benchmark datasets. Through comprehensive analyses from various perspectives, they shed light on the significance of supervision, the value of prior understanding of anomaly categories, and the fundamentals of crafting resilient detection algorithms.

2.3 False tolerant RL resource management

An overlap between the areas of learning-based resource management and fault tolerance can be observed in the research work of S. Moghaddam [5]. The proposed solution utilizes an anomaly detection module to detect persistent performance problems in the system, triggering the decision-making module of reinforcement learning (RL) to initiate a scaling action for correcting the issue. This solution combines the two concepts; however, the agent is triggered only when an anomaly is detected to correct the problem and does not serve as the main job scheduler.

3 METHODOLOGY

In this section, we outline our chosen resources including datasets and software tools, our research methods to answer the research question and the methods we will use to evaluate the results.

3.1 Data and Software

For this research, our approach builds upon previous methodologies and strategies. More specifically, we extend the research conducted by Mao [7] by utilizing the DeepRM algorithm as our foundation. This choice is motivated by the fact that DeepRM is regarded as a state-of-the-art in the field, and additionally, its code is open source and accessible to everyone.

The DeepRM Architecture consists of a cluster with d resource types (e.g., CPU, memory, I/O), a sequential flow of jobs that arrive to the cluster in discrete time steps and a scheduler that chooses one or more of the waiting jobs to schedule at each time step.

In our case, the synthetic dataset which is considered the sequential flow of jobs is randomly created in the background by a job generator and follows a specific prototype. Each job is composed of the following main features:

- The job ID: We denote the job ID as j_i, where $j \in J$ is the set of all jobs, and i is the unique ID of job j.
- The job resource requirement: Denoted as $r_j = \{[r_{1j1}, r_{2j1}], \ldots, [r_{1jn}, r_{2jn}]\}$, where r represents the amount per type of resource required by job j.
- The job length: Represented as j_L, the length of job j is in the range of Lower Boundary to Upper Boundary: $L_n = \{L \in \mathbb{N} \mid LB \leq L \leq UB\}$.

For the needs of our research we will modify the generation module to produce "anomalous" jobs with either an abnormal length of execution or a resource requirement higher than normal jobs.

3.2 Our Approach

The innovation of our research lies in bridging the gap between the domains of learning-based resource management and anomaly tolerance. In order to address the resource question we posed, we need to simultaneously solve these two problems under a single algorithm.

The anomaly tolerance aspect can be addressed by either modifying the reward function of DeepRM to encourage the agent to detect anomalous jobs, penalizing it for misclassifying jobs, or by training a separate classification module to identify anomalous jobs and informing the agent accordingly.

The aspect of migrating anomalous jobs to the scheduling process requires extensive testing and evaluation to identify the optimal method of incorporating these anomalous jobs effectively.

3.3 Evaluation

The assessment of our proposed model will focus on two primary aspects. As outlined earlier, the model aims to achieve two objectives: detecting and migrating anomalous tasks, all the while minimizing job slowdown. Subsequently the evaluation will focus on how well can the model perform these two tasks.

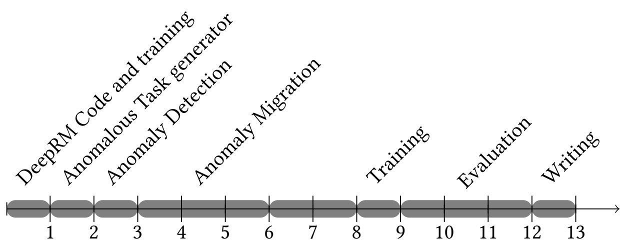
- To assess the performance of the model we will compare the average job slowdown of with that of the baseline model DeepRM.
- To assess the ability of the model to detect anomalies we will compare the predictions labels with the ground Truth labels and we will derive classification metrics such as F1-Score, precision and recall

4 RISK ASSESSMENT

The primary risk in our research stems from the prolonged duration required for the model to train. For context, the DeepRM algorithm takes approximately a day to train on a system lacking a GPU, such as one equipped with an M1 chip. This extended training period presents a substantial constraint on our research endeavors, particularly as the introduction of anomalous job instances could impede the model's convergence to an optimal solution.

5 PROJECT PLAN

In order to successfully complete my research work on time, I need to follow a series of steps. Firstly, I must fully understand the DeepRM code and train a model to use as a reference in the evaluation part. Secondly, I need to build the anomalous job generator to create a training dataset. Then, I will focus on the anomaly detection and anomaly migration tasks. Finally, I will train the model and evaluate the results. Throughout this process, I will ensure to fix any issues and refine the model as needed.



REFERENCES

[1] Faraz Fatemi Moghaddam, Mohammad Ahmadi, Samira Sarvari, Mohammad Eslami, and Ali Golkar. 2015. Cloud computing challenges and opportunities: A survey. In *2015 1st International Conference on Telematics and Future Generation Networks (TAFGEN)*. 34–38. https://doi.org/10.1109/TAFGEN.2015.7289571

[2] Sepideh Goodarzy, Maziyar Nazari, Richard Han, Eric Keller, and Eric Rozner. 2020. Resource Management in Cloud Computing Using Machine Learning: A Survey. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 811–816. https://doi.org/10.1109/ICMLA51294.2020.00132

[3] Wenxia Guo, Wenhong Tian, Yufei Ye, Lingxiao Xu, and Kui Wu. 2021. Cloud Resource Scheduling With Deep Reinforcement Learning and Imitation Learning. *IEEE Internet of Things Journal* 8, 5 (2021), 3576–3586. https://doi.org/10.1109/JIOT.2020.3025015

[4] Songqiao Han, Xiyang Hu, Hailiang Huang, Mingqi Jiang, and Yue Zhao. 2022. ADBench: Anomaly Detection Benchmark. arXiv:2206.09426 [cs.LG]

[5] Sara Kardani-Moghaddam, Rajkumar Buyya, and Kotagiri Ramamohanarao. 2021. ADRL: A Hybrid Anomaly-Aware Deep Reinforcement Learning-Based Resource Scaling in Clouds. *IEEE Transactions on Parallel and Distributed Systems* 32, 3 (2021), 514–526. https://doi.org/10.1109/TPDS.2020.3025914

[6] Abdeldjalil Ledmi, Hakim Bendjenna, and Sofiane Mounine Hemam. 2018. Fault Tolerance in Distributed Systems: A Survey. In *2018 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS)*. 1–5. https://doi.org/10.1109/PAIS.2018.8598484

[7] Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. 2016. Resource Management with Deep Reinforcement Learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks* (Atlanta, GA, USA) *(HotNets '16)*. Association for Computing Machinery, New York, NY, USA, 50–56. https://doi.org/10.1145/3005745.3005750

[8] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal* 3, 5 (2016), 637–646. https://doi.org/10.1109/JIOT.2016.2579198

[9] Yufei Ye, Xiaoqin Ren, Jin Wang, Lingxiao Xu, Wenxia Guo, Wenqiang Huang, and Wenhong Tian. 2018. A New Approach for Resource Scheduling with Deep Reinforcement Learning. *CoRR* abs/1806.08122 (2018). arXiv:1806.08122 http://arxiv.org/abs/1806.08122

[10] Guangyao Zhou, Wenhong Tian, and Rajkumar Buyya. 2021. Deep Reinforcement Learning-based Methods for Resource Scheduling in Cloud Computing: A Review and Future Directions. *CoRR* abs/2105.04086 (2021). arXiv:2105.04086 https://arxiv.org/abs/2105.04086