A Survey of Kubernetes Container Scheduling Techniques

Vidhi Sutaria
Department of Computer Science and Engineering,
Uka Tarsadia University, Bardoli, 394350, Gujarat, India
Email:- vidhi.sutaria@utu.ac.in

Dr. Dharmendra Bhatti
Department of Computer Science and Engineering,
Uka Tarsadia University, Bardoli, 394350, Gujarat, India
Email:- dgbhatti@utu.ac.in,

Parth Shah
Maverick Quantum India Pvt. Ltd,
Knowledge City, Hyderabad, 500081, India.
Email:- parth@parthhosting.com

Abstract:

Today, most of the things we do on the Internet are possible only due to cloud technology. Among various technologies used to serve appli- cations, cloud containers are considered the most suitable technological solution. Containers have become the leading virtualization technology in cloud services — powering everything from microservices and smart vehicles to IoT and fog/edge computing. Their popularity comes from being highly flexible, portable, and easily scalable, making them an ideal choice for modern computing environments. Maintaining a large set of containers is a challenging task, to solve that problem, Kubernetes were invented which became the de facto standard for container orchestration. Effortless scheduling of container services is necessary for effective per- formance and cost minimization, yet a well-defined survey of container scheduling techniques is missing. To identify this gap, this paper presents a classification of various scheduling techniques and categorizes them into These four approaches offer different ways to tackle complex problems, especially when making decisions or optimizing solutions. Mathematical, Heuristics, Metaheuristics, and Machine Learning. By defining the merits and demerits of each method based on performance, this paper aims to guide further research in this field. Finally, by highlighting future scope opportunities, this paper aims to unlock the full potential of containers.

1 Introduction

1.1 Cloud Computing

This paper offers a summary of existing literature on scheduling strategies ap- plicable to Kubernetes. Reviews approximately 70 papers published in various journals and conferences. Each paper addresses different problem statements and provides corresponding solutions, highlighting potential areas for improve- ment and opportunities for future research in specific domains [1].

In this review paper, we identify different scheduling algorithms that are used to achieve different performance metrics such as availability, resource utiliza- tion, energy consumption, reliability, cost, load balancing, and response time. In that many surveys only describe the difference between the VM and the con- tainer [2]. Although containers are widely used, there is still no thorough survey describing different container scheduling methods. Having such a survey is cru- cial to pinpointing important research areas that could improve how containers are scheduled.

In our review paper, we initially found basic scheduling methods and cate- gorized them according to optimized algorithms and performance metrics. In examining each category

of scheduling algorithms, we conducted a comprehen- sive analysis of two methods which is possible to merge in the future and give better performance in given performance metrics. The primary objective of this paper is to present the most effective container technology based on research. [3] Incorporating an efficient container scheduling approach presents a significant challenge in light of dynamic workloads and diverse resource needs, among other factors, when considering research-based perspectives [4]. After conducting a thorough examination of the container scheduling technique, we have identified the two most effective scheduling techniques, and for future reference, we can combine these two techniques to achieve higher efficiency.

In the next section, we briefly introduce each four techniques, and after that, the effective two techniques will be described with a literature survey according to the research.

2 Scheduling Classification

Scheduling techniques are grouped into four main categories, each offering a unique approach to organizing and managing tasks efficiently. Here are the four categories:

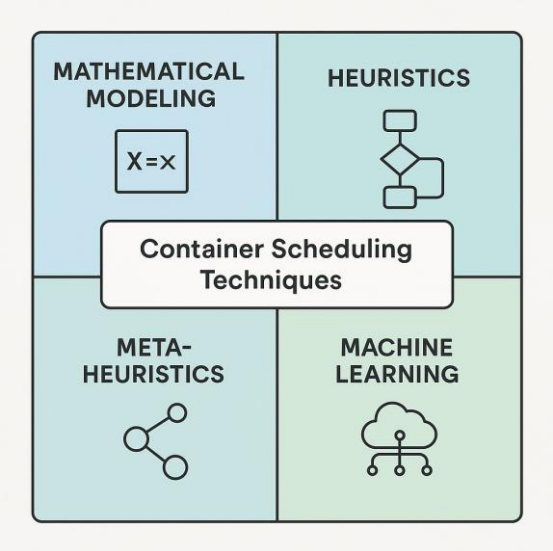

Figure 1: Different methods of container scheduling

The first technique in this set is mathematical modeling, which uses equa- tions with defined parameters, such as integer linear programming (ILP), to solve problems. Follow a structured approach to find effective solutions. How- ever, the downside is that it becomes impractical for larger problems because it requires a lot of computing power.

The second method is a heuristic approach, which is used to find approximate solutions to schedule large-size problems. The heuristic approach is best suited for finding an

effective solution due to its low complexity and the expected outcome in a given technique. So, mainly the researcher works and uses this technique to find the appropriate solution to the problem.

Next, we have the population-based optimization algorithm, which is known as metaheuristics technique. This technique is used for the intelligent processes and behavior of given data. The two most important parameters of this tech- nique are stable selection and adaptation to the environment.

The last one is the most important and very emerging technique known as the machine learning technique. Today, it has achieved great success in various areas and provides a highly effective solution in the domain of container scheduling technology [5]. The effectiveness of machine learning techniques depends on big data. It also depends on training and testing data. However, there is still no suitable effective solution provided by this technique.

So, to solve this, we can combine any of these techniques and take advantage of them. So, for an effective solution of container scheduling, we combine these techniques. For this solution, we combine the two most effective techniques, heuristic and machine learning, for a more effective solution of the container scheduling technique.

In the next section, we describe the detailed literature survey of these two most useful techniques. According to different research papers from 2017 to 2024. In all the papers, researchers use different techniques that provide a large number of data.

3 Container Scheduling Approaches

In this section, we examine the heuristic and machine learning approaches to container scheduling developed in recent years. We highlight their benefits, lim- itations, and distinctive features, as well as the contexts in which they are best suited.

3.1 Heuristic Techniques

Mao et al. (2017) The creator designed and executed the Dynamic and Resource-Aware Placement Scheme (DRAPS) in Docker Swarm to efficiently oversee a diverse array of nodes. This approach utilizes research-derived meth- ods to improve resource usage and refine node allocation [7]..

Dziurzanski and Indrusiak (2018) Researchers adapt a heuristic ap- proach for container distribution in Docker Swarm to maximize workload value. This research does not take memory or network utilization into account when evaluating performance [8].

Zhang et al. (2018) In this paper, the network bandwidth is reduced. It also reduces the execution time of Docker. In this research, it provided a combination of various algorithms [9].

Song et al. (2018) This research presents the Gaia algorithm, a GPU- based scheduling method designed to improve efficiency. It operates using a tree-based resource access model that is cost-conscious to optimize performance. Using it increases dynamic scheduling decisions. The result describes a 10% improvement compared to other approaches [10].

Lv et al. (2019) This research represents a dual-phase scheduling method. This method aims to reduce communication costs while efficiently distribut- ing resource usage. It operates in two stages: first, the Communication-Aware Worst Fit Decreasing (CAWFD) algorithm is used to improve resource utiliza- tion and decrease communication overhead. Then, the Sweep & Search algo- rithm is used to further enhance performance [11]. By integrating these two approaches, the system effectively optimizes container scheduling.

Hu et al. (2019) It uses a vector bin packing method to efficiently schedule containers. In this approach, multiple objectives are used, such as load balanc- ing, resource utilization, and network dependency. It helps the container to be scheduled with different parameters [12].

Menouer and Darmon (2019) In this investigation, the proposed ap- proach is used for overall performance by locating containers with interdepen- dencies. It was on the same host compared to the Kubernetes scheduler [13].

Xu et al. (2018, 2019) Given the method used for energy consumption that uses brownout. This service reduces energy consumption and proposes a new approach to research. The results indicate that this approach achieves 10-40% energy savings compared to other methods [14].

Hu et al. (2020) The research introduced a minimum-cost flow model. That is, to tackle the simultaneous multi resource container scheduling issue. This research also shows practical proof of its higher resource efficiency and performance in contrast to Kubernetes and Docker Swarm. However, it did not consider container dependencies [15].

Menouer (2020) This paper introduces KCSS, a multi-criteria container scheduling approach designed to optimize parameters. The author evaluates six different criteria to

determine the most suitable node for scheduling. The disadvantage is that KCSS does not handle fault tolerance and increases latency [16].

Rodrigues et al. (2020) The GPU-based Scheduler for containers (GPUACS) was presented. This approach tackles the challenge of modeling joint network resources and containers while incorporating Quality of Service (QoS). The au- thors take into account multiple criteria and GPU factors to improve the speed and scalability of the scheduling process [17].

Toka et al.(2021) This research employs a brief evaluation cycle to ex- amine and contrast the effectiveness of various machine learning forecasting techniques. It also introduces a compact management parameter to help ap- plication providers balance over-provisioning and prevent SLA violations. It improves Kubernetes auto-scaling decisions in cloud-based applications [18].

Balla et al.(2020) In this paper, we suggest an adaptive autoscaler known as Libra. This approach autonomously identifies the optimal resource configura- tion for each pod while overseeing the horizontal scaling process. A key feature is its ability to adjust pod resources and improve scaling efficiency. As a result, it reduces CPU usage by up to 48% and memory usage by 39% [19].

Toka et al.(2020) This paper introduces several AI-driven forecasting methods designed for short-term evaluation loops, using simulated web traces for analysis. Based on our research, this method significantly reduces dropped requests and a modest increase in available resources [20].

Wang et al.(2020) In this research, enhance the functionality of Kuber- netes resource scaling. The autoscale prediction and autoscale speed provided for Kubernetes can help enhance resource scaling efficiency in Kubernetes [21].

Kang et al.(2021) This paper introduces an improved automatic scaling strategy for Kubernetes, leveraging different node types with preloaded images. The proposed approach helps stabilize active clusters and boosts system perfor-mance, especially under high-load conditions. By incorporating the strengths of various types of nodes, this method improves the total scalability and efficiency of the system [22].

Vu et al.(2022) The proposed technique safeguards critical operations while preventing resource shortages in cluster nodes. It is particularly useful for Kubernetes infrastructures hosting network services, enhancing availability and resilience. This approach ensures

stable performance across different net- work services within the Kubernetes environment [23].

Turin, et al.(2023) Research introduced Kubernetes and real-time ABS. It also provided a new model for online shopping applications and predicted resources for usage in CI/CD pipelines [24].

Jian et al.(2023) In this research the implementation of a new scheduler named DRS was proposed. Used CPU memory and disk utilization as optimiza- tion parameters. The result is compared with the default scheduler [25].

Here figure 2 summarizes the heuristic algorithm.

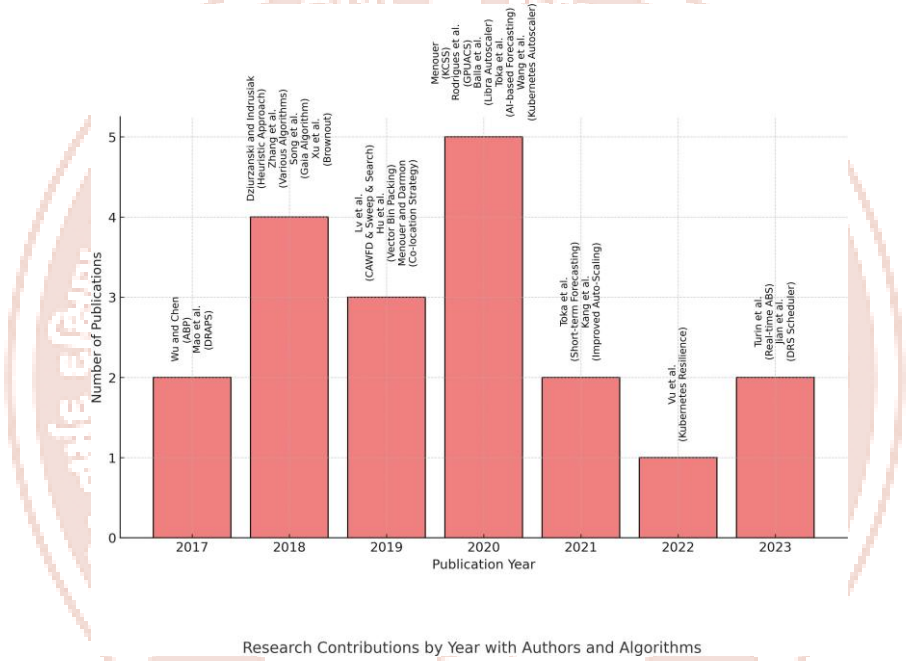

Figure 2: Summary of heuristic approach

Only a limited number of heuristic approaches have been dedicated to en- hancing the availability of the system. These heuristics are valued for their ability to quickly identify solutions and can be combined with other optimiza- tion techniques to improve the quality of the final schedules.

3.2 Machine Learning Techniques

Machine learning algorithms utilize data to autonomously learn and make in- formed decisions. ML utilizes a complex neural network system to acquire knowledge and make sophisticated decisions independently. Deep learning algo- rithms have demonstrated significant success in the domains of image processing and computer vision and are presently being widely implemented across diverse fields. Resource optimization poses a

significant challenge, especially in the realm of container consolidation. The goal is to efficiently assign containers to the fewest possible computing nodes, maximizing resource utilization while minimizing energy consumption.

Nanda and Hacker (2018) A groundbreaking deep reinforcement learn- ing method was introduced to optimize container consolidation and improve resource utilization;It outperforms traditional techniques such as the Shortest Job First and random placement algorithms in terms of efficiency [26].

Lv et al.(2019)A new approach was developed using a random forest re- gression model to more accurately predict how many containers are needed. This strategy provides precise forecasts of future resource requirements, allowing for swift adjustments to abrupt changes in workloads. The algorithm exhibited substantial improvements in both speed and accuracy compared to alternative machine learning techniques, such as support vector machines [27].

Mehta et al. (2020) The study introduced WattsApp, a power-aware con- tainer scheduling method that helps prevent servers from exceeding their power limits, ensuring smooth and stable performance for all containers [28].

Nath et al.(2020) They have employed Bayesian optimization-based sta- tistical online learning techniques to address container consolidation problems with limited data points, and their experimental findings demonstrate that the proposed program has effectively reduced total consumption of energy in data centers compared to different existing methods [29].

Peng et al.(2021) A flexible and efficient deep learning cluster scheduler was developed to make the most of expensive deep learning resources [30].

Mao et al.(2022) Improved scheduling for deep learning applications in Kubernetes clusters by optimizing resource management and ensuring that re- sources are used to their full potential [31].

Huang et al.(2020)RLSK, a job scheduling system powered by deep re- inforcement learning, was developed to dynamically assign independent batch processes across various federated cloud computing clusters. This adaptive ap- proach ensures an efficient distribution of the workload, optimizing performance and resource utilization [32].

Wang et al.(2020) A new approach was designed to consider both the temporal and spatial characteristics of machine learning jobs. During system overload, this load control

mechanism intelligently eliminates tasks that con- tribute minimal or no benefit, ensuring efficient resource utilization [33].

Liu et al.(2022) A deep learning task planning strategy for Kubernetes has been introduced, analyzing task resource characteristics; the approach improves execution efficiency and ensures balanced workload distribution [34].

Rahali et al.(2021) The proposed approach optimizes resource allocation in the Kubernetes infrastructure that hosts network services, ensuring the pro- tection of essential functions while preventing resource shortages [35].

Chandra et al.(2023) compared the default and custom scheduler and implemented it on AWS EC2. The custom scheduler was addressing which resources were allocated more accurately from the result compared to the default [36]. Figure 3 describes the summary of machine learning approaches.

Figure 3: Summary of machine learning approach

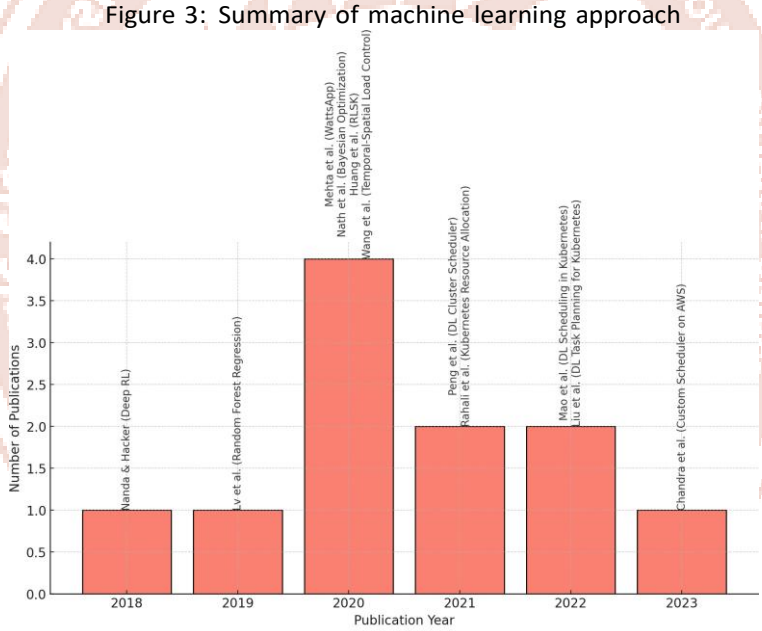

Figure: Research Contributions by Year with Authors and Algorithms

Figure 3: Summary of machine learning approach

4 Conclusion

In recent years, the adoption of containers for cloud service delivery has in- creased, making efficient resource management through real-time container schedul- ing a critical aspect of cloud computing. This survey provides a comprehensive analysis of container scheduling technologies, highlighting key advances, chal- lenges, and future directions.

We first classified scheduling methods into four categories based on the opti- mization algorithms used to develop the schedules. Afterward, we assessed the optimization goals to measure the efficacy of the produced schedules.

5 Future Work

Based on the above survey, we found a large research gap in developing an efficient and cost effective container scheduling algorithm. Research in this area can provide a beneficial outcome for the overall cloud computing landscape as day-by-day resource usage in the cloud is increasing.

References

1. Ahmad, Imtiaz, et al. ”Container scheduling techniques: A survey and as- sessment.” Journal of King Saud University-Computer and Information Sci- ences (2021).
2. Menouer, Tarek. ”KCSS: Kubernetes container scheduling strategy.” The Journal of Supercomputing 77.5 (2021): 4267-4293.
3. Balla, David, Csaba Simon, and Markosz Maliosz. ”Adaptive scaling of Kubernetes pods.” NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2020.
4. Carri´on, Carmen. ”Kubernetes Scheduling: Taxonomy, ongoing issues and challenges.” ACM Computing Surveys (CSUR) (2022).
5. Arunarani, A. R., Dhanabalachandran Manjula, and Vijayan Sugumaran. ”Task scheduling techniques in cloud computing: A literature survey.” Fu- ture Generation Computer Systems 91 (2019): 407-415.
6. Mao, Y., Oak, J., Pompili, A., Beer, D., Han, T., Hu, P., 2017. Draps: Dynamic and resource-aware placement scheme for docker containers in a heterogeneous cluster. In: 2017 IEEE 36th International Performance Com- puting and Communications Conference (IPCCC). IEEE, pp. 1–8.
7. Dziurzanski, P., Indrusiak, L.S., 2018. Value-based allocation of docker con- tainers. In: 2018 26th Euromicro International Conference on Parallel, Dis- tributed and Network-based Processing (PDP). IEEE, pp. 358–362.
8. Zhang, W., Liu, Y., Wang, L., Li, Z., Goh, R.S.M., 2018. Cost-efficient and latency-aware workflow scheduling policy for container-based systems. In: 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS). IEEE, pp. 763–770.

9. Song, S., Deng, L., Gong, J., Luo, H., 2018. Gaia scheduler: A kubernetes-based scheduler framework. In: 2018 IEEE Intl Conf ona- tions(ISPA/IUCC/BDCloud/SocialCom/SustainCom). IEEE, pp. 252–259.
10. Lv, J., Wei, M., Yu, Y., 2019. A container scheduling strategy based on machine learning in a microservice architecture. In: 2019 IEEE International Conference on Services Computing (SCC).
11. Hu, Y., De Laat, C., Zhao, Z., et al., 2019. Multi-objective container de- ployment on heterogeneous clusters. In: Proc. 19th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.(CCGRID), pp. 592–599.
12. Menouer, T., Darmon, P., 2019. Containers scheduling consolidation ap- proach for cloud computing. In: International Symposium on Pervasive Sys- tems, Algorithms and Networks. Springer, pp. 178–192.
13. Xu, M., Buyya, R., 2019. Brownoutcon: A software system based on brownout and containers for energy-efficient cloud computing. J. Syst. Soft. 155, 91–103.
14. Hu, Y., Zhou, H., de Laat, C., Zhao, Z., 2020. Concurrent container scheduling on heterogeneous clusters with multi-resource constraints. Fut. Gen. Comput. Syst. 102, 562–573.
15. Menouer, T., 2020. KCSS: Kubernetes container scheduling strategy. J. Supercomput., 1–27.
16. Rodrigues, L.R., Pasin, M., Alves Jr, O.C., Miers, C.C., Pillon, M.A., Felber, P., Koslovski, G.P., 2019. Network-aware container scheduling in multi-tenant data center.
17. Toka L, Dobreff G, Fodor B, Sonkoly B (2021) Machine Learning-Based Scaling Management for Kubernetes Edge Clusters. IEEE Trans Netw Serv Manage 18(1):958–972
18. Balla D, Simon C, Maliosz M (2020) Adaptive scaling of Kubernetes pods.IEEE/IFIP Network Operations and Management Symposium 2020: Management in the Age of Softwarization and Artificial Intelligence, NOMS
19. Toka L, Dobreff G, Fodor B, Sonkoly B (2020) Adaptive AI-based autoscal- ing for Kubernetes. IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing, CCGRID. pp 599–608

20. Wang M, Zhang D, Wu B (2020) A Cluster Autoscaler Based on Multiple Node Types in Kubernetes. IEEE 4th Information Technology, Networking,Electronic and Automation Control Conference, ITNEC. pp 575–579
21. Kang R, Zhu M, He F, Sato T, Oki E (2021) Design of Scheduler Plugins for Reliable Function Allocation in Kubernetes. 17th International Conference on the Design of Reliable Communication Networks, DRCN
22. Vu DD, Tran MN, Kim Y (2022) Predictive hybrid autoscaling for con- tainerized applications. IEEE Access 10:109768–109778
23. Turin, G., Borgarelli, A., Donetti, S., Damiani, F., Johnsen, E.B. and Tarifa, S.L.T., 2023. Predicting resource consumption of Kubernetes con- tainer systems using resource models. Journal of Systems and Software, 203, p.111750.
24. Jian, Z., Xie, X., Fang, Y., Jiang, Y., Li, T. and Lu, Y., 2023. DRS: A Deep Reinforcement Learning Enhanced Kubernetes Scheduler for Microservice- based System.
25. Nanda, S., Hacker, T.J., 2018. Racc: resource-aware container consolida- tion using a deep learning approach. In: Proceedings of the First Workshop on Machine Learning for Computing Systems, pp. 1–5.
26. Lv, J., Wei, M., Yu, Y., 2019. A container scheduling strategy based on machine learning in microservice architecture. In: 2019 IEEE International Conference on Services Computing (SCC). IEEE, pp. 65–71.
27. Mehta, H., Harvey, P., Rana, O., Buyya, R., Varghese, B., 2020. Wattsapp: Power-aware container scheduling.
28. Nath, S.B., Addya, S.K., Chakraborty, S., Ghosh, S.K., 2020. Green con- tainerized service consolidation in the cloud. In: ICC 2020–2020 IEEE In- ternational Conference on Communications (ICC). IEEE, pp. 1–6.
29. Peng Y, Bao Y, Chen Y, Wu C, Meng C, Lin W (2021) DL2: A Deep Learning-Driven Scheduler for Deep Learning Clusters. IEEE Trans Parallel Distri[81]b Syst 32(8):1947–1960
30. Mao Y, Fu Y, Zheng W, Cheng L, Liu Q, Tao D (2022) Speculative Con- tainer Scheduling for Deep Learning Applications in a Kubernetes Cluster. IEEE Syst J 16(3):3770–3781

31. Huang J, Xiao C, Wu W (2020) RLSK: A Job Scheduler for Federated Kubernetes Clusters based on Reinforcement Learning. IEEE International Conference on Cloud Engineering, IC2E. pp 116–123
32. Wang H, Liu Z, Shen H (2020) Job scheduling for large-scale machine learn- ing clusters. Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies. pp 108–120
33. Liu Z, Chen C, Li J, Cheng Y, Kou Y, Zhang D (2022) KubFBS: A fine- grained and balance-aware scheduling system for deep learning tasks based on Kubernetes. Concurrency Computat Pract Exper 34(11):e6836. https:// doi.org/ 10. 1002/ cpe. 6836
34. Rahali M, Phan CT, Rubino G (2021) KRS: Kubernetes Resource Sched- uler for resilient NFV networks. IEEE Global Communications Conference
35. Chandra, M., 2023. Effective Memory Utilization using Custom Scheduler in Kubernetes