WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Towards Intelligent Kubernetes Scheduling: Bridging Latency, Cost, and Trust in Multi-Cloud Environments

Mitul Raj*

School of Engineering, P P Savani University, Surat, India

A B S T R A C T :

The adoption of microservice based architectures in the cloud native environment has also accelerated the shift towards the multi cloud environment to avoid vendor lock-in, enhanced availability and disaster recovery by distributing workloads across multiple cloud providers. To efficiently manage this micro-services across multiple cloud environments, "Kubernetes" has been emerged as the leading platform to automate the deployment and orchestrate the cloud-native workloads across multi cloud environments. However, Kubernetes native scheduling approaches are primarily designed for single cloud environment which leaves significant gap when used in multi cloud setup. The multi cloud deployment is complex and offers challenges like cross-cloud latency, data locality, resource allocation which impede the performance of Kubernetes in the multi cloud environment.

The aim of this paper is to identify the limitations of Kubernetes scheduling in the multi cloud setup and highlights the need to have alternative solutions which can address this limitations and improve the performance. The novel Kubernetes Scheduling algorithm can bridge this gap if specifically designed for multi cloud environments. We propose Multi Cloud optimize scheduling algorithm to enhance the performance of Kubernetes by optimizing latency, resource utilization across for a specific category of workload in the multiple cloud platforms.

Keywords: Kubernetes, Scheduler, Multi cloud, federated cloud, latency, cost optimization.

1. Introduction

The increased adoption of cloud technologies and the evaluation of organizational requirements has accelerated the sift from cloud native environment to multi cloud native strategies. Organizations, specifically those who has global presence, increasingly seek for the solution for their dynamic requirements of flexibility, resilience and cost effectiveness[Ref : Netflix Multi Cloud Native Paper 1st one].

According to Cloud Native Computing Foundation (CNCF) Annual surveys till 2023 [Ref CNCF], growing prominence of multi cloud strategies are highlighted for modern infrastructure managements. In 2020, multi cloud adoption by the organization where raised with slight decreased in the usage of hybrid cloud. By 2022, the trend where strengthened due to adoption of containerised technologies by the organizations with the usage of container orchestration tools like Kubernetes[ref]. Recent survey also highlights that 56% of organizations use hybrid or multi cloud setups compared to 28% that relied on public cloud. On average they uses services from 2 to 3 cloud service providers with multi cloud increasing that numbers. This shift highlights the importance of multi cloud for managing workloads effectively.

Multi Cloud typically means "using workloads across different clouds based on the type of cloud that fits the workload best." [3] . By distributing the workload across multiple cloud environments it ensures optimized returned on investments meeting the organisations functional and non-functional requirements such as reduced vendor lock-in, cost optimizations, low latency, real time processing etc. As a result many businesse with their global presence such as Netflix (Entertainment Business), energy, autonomous driving, e-health found that multi cloud adoption is highly advantageous solution for them[4. Refer Links for Industries adopting Multi cloud].

Importance of Kubernetes in container orchestration:

The Cloud native application development with the support of container technology has transformed the way how applications are built and run on cloud infrastructures. Containers are light weight application virtualization technology which provides logical packaging of applications along with dependencies. The container creates portable environment for the applications which stands consistent from developer machines to production environment which is critical in cloud native applications where microservices are often deployed on different environments. The rapid application deployment, easy scaling and efficient resource utilization across dynamic and distributed environments also poses significant challenges such as Load balancing, Orchestration, scaling and fault tolerance.

To address this challenges, the container orchestration tools such as (Kubernetes[6], Borg[7], Docker Swarm[7] are used to automate the deployment and effective management of containerized applications. Kubernetes(K8s), an open source container orchestration tool developed by Google is leader in the market. It empowers the organizations to fully leverage the benefits of container technology while maintaining the operational efficiency and reliability. Its architecture includes components like Kubernetes API server, scheduler and control plane which enables developers to manage the complex workload efficiently. Kubernetes uses built in schedulers to determine where containers should be run based on the available resources. However, despite widespread adoption of Kubernetes, it experiences significant challenges in the Multi cloud environments. As different organizations leveraging multiple cloud providers to avoid the vendor lock in and optimisation of performance and cost, Kubernetes native schedulers and its mechanism struggles with the multi cloud specific challenges. [Kubernetes Scheduling Taxonomy]
The Scheduling in Kubernetes is active research field Scheduling Special Interest Group (SIG) is community interested in various scheduling related questions on Kubernetes. Scheduling for containerized application is an area of growing interest and this research will focus on the identification of limitations with the Kubernetes scheduling mechanism with respect to the Multi cloud environments and research gap leading to proposal of new scheduling algorithm optimised for muti cloud Kubernetes.

2. Background

Kubernetes has transformed the way containerized applications are deployed and managed. At its core lies the scheduler, a critical component that ensures efficient workload distribution across available compute resources. In parallel, the industry trend toward multi-cloud computing—where organizations deploy workloads across multiple cloud service providers—adds layers of complexity and opportunity that existing Kubernetes schedulers are not inherently designed to handle. This section introduces the foundational concepts of Kubernetes scheduling and outlines the key challenges faced in multi-cloud environments.

2.1 Kubernetes Scheduling Basics

Kubernetes scheduling is the process of assigning newly created pods to appropriate nodes in a cluster. The default Kubernetes scheduler operates through the process of filtering and Scoring.

- Filtering – Eliminates nodes that don't satisfy the pod's requirements (e.g., CPU, memory, affinity rules, taints, tolerations).
- Scoring – Ranks the remaining candidate nodes based on priorities (e.g., least requested resources, balanced resource allocation).

The default scheduler supports extensions via scheduler profiles, plugins, and custom scheduling policies. However, it is designed primarily for single-cluster, homogeneous environments. This makes it less effective for latency-sensitive workloads, Stateful applications with data locality constraints, Workloads spanning across cloud vendors with different characteristics.

2.2 Multi-Cloud Concepts and Challenges

A multi-cloud environment involves the simultaneous use of two or more cloud platforms (e.g., AWS, Azure, GCP) to host different segments of an organization's workloads. It offers significant benefits, such as - Redundancy and High Availability - Reduces dependency on a single cloud vendor. Also Cost Optimization Enables selection of the most cost-effective resources from different providers and performance Optimization allows deploying services closer to end-users across regions.

However, managing Kubernetes workloads in a multi-cloud setup introduces several complex challenges like 1. Interoperability Issues where each cloud provider has its own APIs, services, resource types, and compliance standards, making seamless integration difficult. 2. Latency and Bandwidth Variations between Inter-cloud communication introduces non-deterministic delays and variable network throughput, impacting service performance. 3. Diverse Cost Models where used by different cloud service providers where Pricing schemes vary widely (e.g., on-demand vs spot instances, ingress/egress charges), complicating cost-aware scheduling. 4. Data Residency and Compliance: Legal and organizational policies may restrict where certain types of data can be processed or stored. 5. Lack of Unified Control Plane: Kubernetes is typically limited to single-cluster or federated clusters, with few native tools for intelligent cross-cloud scheduling.

While Kubernetes federation solutions like KubeFed, Karmada, and Istio multi-cluster architectures attempt to bridge some gaps, they lack deep integration with scheduling logic that dynamically considers network latency, cloud costs, and trust-levels of resources.

3. Literature Review

Kubernetes is rapidly emerging as the de facto standard for orchestrating containerized applications. However, scheduling across multi-cloud environments introduces complexity that surpasses its default design. Various frameworks and research contributions in your uploaded documents provide insight into existing solutions, but collectively expose important gaps in current Kubernetes scheduling models for multi-cloud.

3.1 Interoperability and Cloud Architecture Challenges:

Shukla and Patil [6] provide an extensive review of frameworks to achieve interoperability in multi-cloud systems. The proposed Multi-Cloud Interoperability Framework (MCIF) emphasizes cloud-agnostic APIs and federated identity management. However, it doesn't address scheduling

strategies or workload placement in real-time operational Kubernetes clusters. Similarly, Jamshidi et al. propose migration patterns to enable transparent application transitions to multi-cloud architectures [7] . Yet, their focus is on application-level migration rather than runtime pod-level scheduling decisions or network latency trade-offs in Kubernetes clusters.

3.2 Resource Allocation, Monitoring, and Visibility:

Sharma's work integrates Prometheus and Grafana with Kubernetes for monitoring multi-cloud deployments [10]. While useful for observability, this setup is reactive. *It doesn't aid in proactive scheduling*, especially under high traffic or latency-sensitive conditions.

3.3 Trust, Fault, and Security Models in Multi-Cloud:

Gupta et al. and Alajroush et al. emphasize trust models to evaluate service provider reliability in multi-cloud environments[11] [12]. These models support decision-making at the provider level but are not designed for pod-to-node scheduling decisions in Kubernetes environments. Mohanraj et al. address IDS in cloud and fog computing but the paper is retracted [14]. Hence, its findings should be excluded from academic influence..

3.4 SaaS and Federation-Centric Management Systems:

Ouchaou et al. proposed a distributed SaaS management system that builds trust and service discoverability through ontology-based clustering[]. While effective in service discovery, it overlooks *runtime orchestration and pod placement challenges* inherent in multi-cloud Kubernetes clusters.

4. Research Gap

From the survey and reviewed literature, several key gaps are identified:

- ***Lack of Multi-Cloud-Aware Kubernetes Scheduling Algorithms***: No reviewed paper proposes a comprehensive, real-time scheduling algorithm for multi-cloud-aware pod placement based on latency, cost, or trust levels.
- ***Neglect of Latency as a Scheduling Metric***: While network topology is considered at the infrastructure level, **latency-aware pod scheduling** (e.g., placing latency-critical services in closer cloud regions) remains largely unaddressed.
- ***Absence of Cost-Aware Scheduling Integration***: Although cost optimization is a primary driver for multi-cloud adoption, no scheduling framework natively integrates **real-time pricing models** (e.g., spot vs reserved) into pod placement logic.
- ***Insufficient Use of Trust and Fault Tolerance in Scheduling***: Trust frameworks exist (e.g., broker-based trust models), but are not applied to **Kubernetes pod scheduling**, especially in fault-prone multi-cloud clusters.
- ***Federated Control Plane Limitations:***Most frameworks focus on service management rather than **distributed scheduling logic** across federated Kubernetes clusters. There is a need for **cross-cluster schedulers** that operate as global controllers.

5. Suggested Future Work:

To overcome the limitations identified in Kubernetes' default scheduling in multi-cloud environments, several promising research directions can be pursued. One critical direction is the development of a *latency-aware global scheduler* that can dynamically evaluate round-trip time (RTT) between cloud zones and place pods based on latency-sensitive requirements. This is particularly important for microservices and real-time applications that depend on low-latency inter-service communication. Another direction involves designing a *cost-sensitive scheduling algorithm* that leverages real-time cloud pricing APIs, such as those provided by AWS, Azure, or GCP, to intelligently allocate resources based on the most economical options like spot or savings plans. Furthermore, a *trust-enhanced scheduling mechanism* is necessary to incorporate historical reliability and service-level performance of cloud regions or providers into the scheduling decision, ensuring that critical workloads are assigned to trusted infrastructure. A *federated cross-cluster meta-scheduler* could enable global policy-driven orchestration across multi-cloud Kubernetes clusters, factoring in latency, cost, and compliance constraints. Finally, a *multi-cloud simulation framework* using tools like Minikube, KIND, or KubeSim should be established to model inter-cloud variability and test the proposed scheduling strategies in controlled environments before real-world deployment.

6. Conclusion:

Multi-cloud environments offer significant advantages such as improved redundancy, cost-efficiency, and global reach, but they also introduce substantial challenges for workload orchestration using Kubernetes. The current body of research provides valuable contributions in areas like service federation, trust management, and cloud interoperability, but it falls short in addressing the intricacies of real-time, intelligent pod scheduling across multiple cloud platforms. This survey has highlighted these critical gaps and underscored the need for a novel, custom-built Kubernetes scheduler that is equipped to operate in federated, heterogeneous, and latency-sensitive environments. Such a scheduler must be capable of optimizing for key factors like network latency, cost variability, trustworthiness, and compliance—elements that are often overlooked in standard scheduling policies. By addressing these needs, the research community can significantly enhance the scalability, efficiency, and resilience of containerized applications in the rapidly evolving multi-cloud landscape.

REFERENCES

1. Flexera: 2024 state of the cloud report: https://info.flexera.com/CM-REPORT-State-of-the-Cloud?id=ELQ-Redirect Last Accessed on 17-01-2025 (2025)
2. CNCF (https://www.cncf.io/)

3. Alonso, J., Orue-Echevarria, L., Casola, V. et al. Understanding the challenges and novel architectural models of multi-cloud native applications – a systematic literature review. J Cloud Comp 12, 6 (2023). https://doi.org/10.1186/s13677-022-00367-6

4. How leading industries are driving multi cloud adoption | ITProPortal. https://www.itproportal.com/features/how leading industries are driving multi cloud adoption/. Accessed 1 Jan 2022

5. Senjab, K., Abbas, S., Ahmed, N. et al. A survey of Kubernetes scheduling algorithms. J Cloud Comp 12, 87 (2023). https://doi.org/10.1186/s13677-023-00471-1

6. P. R. Shukla and V. M. Patil, "A Comprehensive Review of Frameworks for Achieving Interoperability in Multi-Cloud Environments," 2023 IEEE Int. Conf. on Informatics (ICI), pp. 1–7.

7. L. Ouchaou, H. Nacer, and C. Labba, "Towards a Distributed SaaS Management System in a Multi-Cloud Environment," Cluster Computing, vol. 25, pp. 4051–4071, 2022.

8. P. Jamshidi et al., "Cloud Migration Patterns: A Multi-Cloud Architectural Perspective," ICSOC 2014, LNCS 8954, pp. 6–19, Springer, 2015.

9. V. Sharma, "Managing Multi-Cloud Deployments on Kubernetes with Istio, Prometheus and Grafana," 8th Int. Conf. on Advanced Computing and Communication Systems (ICACCS), 2022.

10. P. Gupta and P. K. Gupta, *Trust & Fault in Multi Layered Cloud Computing Architecture*, Springer Nature, 2020.

11. A. Alajroush, R. Latif, and T. Saba, "Trust Management Frameworks in Multi-Cloud Environment: A Review," IEEE Access, vol. 11, pp. 3456–3467, 2023.

12. T. Mohanraj and R. Santhosh, "Security and Privacy Issues in Multi-Cloud Accommodating Intrusion Detection System," *Retracted Article*, Distributed and Parallel Databases, 2021.

13. A. Ghaffari, M. Shojafar, and E. Baccarelli, "Latency-Aware and Energy-Efficient Scheduling in Fog-Cloud Computing," IEEE Transactions on Sustainable Computing, vol. 4, no. 2, pp. 121–132, Apr. 2019.

14. M. Abedi, A. Khonsari, and M. H. Yaghmaee, "A Trust-Based Cloud Scheduling Mechanism for Multi-Tenant Architectures," IEEE Transactions on Cloud Computing, vol. 8, no. 1, pp. 282–295, Jan.–Mar. 2020.

15. M. Zhang, R. Ranjan, and L. Wang, "KubeEdge: A Kubernetes-Native Edge Computing Framework," IEEE Internet of Things Journal, vol. 8, no. 4, pp. 2252–2264, Feb. 2021.

16. R. Mao, X. Lin, and M. Dong, "Cost-Aware Scheduling for Deadline-Constrained Workflows in Hybrid Cloud," IEEE Transactions on Cloud Computing, vol. 7, no. 1, pp. 45–58, Jan.–Mar. 2019.

17. Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2322–2358, Q4 2017.

18. M. Morabito, "Power Consumption of Virtualization Technologies: An Empirical Investigation," 2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom), pp. 522–527.

19. A. Singh, M. Korupolu, and D. Mohapatra, "Server-Storage Virtualization: Integration and Load Balancing in Data Centers," Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, 2008.

20. K. Bilal et al., "Cloud Computing: A Taxonomy, Survey, and Issues," IEEE Access, vol. 1, pp. 205–225, 2013.

21. R. Jain and S. Paul, "Network Virtualization and Software Defined Networking for Cloud Computing: A Survey," IEEE Communications Magazine, vol. 51, no. 11, pp. 24–31, Nov. 2013.