JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Optimizing Cloud Workloads with ARAS: An Adaptive Resource-Aware Scheduling Algorithm

Dr. B. Lalitha[1], Dr. P. Jaya Prakash[2],
[1]Associate Professor, Dept. of.CSE, JNTUACEA, Anantapur.
[2]Associate Professor, Dept. of.CSE, SVCE, Tirupati

Abstract: This research work proposes a novel scheduling algorithm called Adaptive Resource-Aware Scheduling (ARAS) for cloud computing environments. The ARAS algorithm aims to dynamically allocate resources based on workload characteristics, resource availability, and user-defined priorities. Through a combination of task prioritization, resource allocation, and load balancing techniques, ARAS optimizes resource utilization, minimizes task completion time, and enhances overall system efficiency. The performance of the ARAS algorithm is evaluated using various performance metrics, including makespan, resource utilization, response time, fairness, and throughput. The results demonstrate the effectiveness of ARAS in improving cloud scheduling performance

***Index Terms* – ARAS, Baker-von Neumann, Clouds**

1. Introduction:

Cloud computing has revolutionized the way organizations utilize and manage computational resources by providing on-demand access to a shared pool of configurable computing resources. It offers scalability, flexibility, and cost-efficiency, making it an attractive option for businesses of all sizes. Efficient resource allocation and task scheduling are crucial for optimizing the utilization of cloud resources and ensuring timely completion of tasks, ultimately leading to enhanced system performance and user satisfaction.

In traditional cloud scheduling approaches, such as round-robin or random scheduling, resources are allocated to tasks without considering their specific requirements or priorities. These simplistic methods often lead to inefficient resource utilization, as well as suboptimal task completion times. As cloud computing environments become increasingly dynamic and complex, there is a need for more intelligent and adaptive scheduling algorithms that can handle varying workload characteristics, user-defined priorities, and resource availability.

In the literature, various scheduling algorithms have been proposed to address these challenges. Genetic algorithms, ant colony optimization, and particle swarm optimization are some examples of optimization techniques used in cloud scheduling. While these algorithms have shown promising results in improving scheduling efficiency, they often struggle with scalability and adaptability to dynamic environments. Additionally, they may not adequately consider the unique requirements and priorities of different tasks.

To address these limitations and optimize cloud scheduling performance, this research work proposes the Adaptive Resource-Aware Scheduling (ARAS) algorithm. The ARAS algorithm aims to dynamically allocate resources based on workload characteristics, resource availability, and user-defined priorities. It leverages a combination of task prioritization, resource allocation, and load balancing techniques to optimize resource utilization, minimize task completion time, and enhance overall system efficiency.

The ARAS algorithm takes into account the characteristics of incoming tasks, such as their resource requirements, deadlines, and priorities. It dynamically allocates resources based on these factors, ensuring that high-priority tasks receive sufficient resources and are completed within their deadlines. The algorithm also incorporates load balancing mechanisms to distribute tasks across available resources, avoiding resource contention and ensuring efficient utilization.

By adopting the ARAS algorithm, cloud service providers can achieve better resource allocation, improved task completion times, and enhanced overall system performance. Users of cloud services can benefit from faster response times, increased fairness in resource allocation, and improved service quality. The ARAS algorithm has the potential to overcome the limitations of existing scheduling approaches and provide a more adaptive and efficient solution for cloud scheduling.

In summary, the proposed ARAS algorithm addresses the challenges of resource contention, task prioritization, and load balancing in cloud computing environments. By dynamically adapting resource allocation based on workload characteristics and user-defined priorities, ARAS aims to optimize resource utilization, minimize task completion time, and enhance overall system efficiency. The following sections of this paper will present a detailed methodology, experimental results, and analysis to demonstrate the effectiveness of the ARAS algorithm in improving cloud scheduling performance.

2. Literature review

In 2016, a study titled "Symbiotic Organism Search Optimization-Based Task Scheduling in Cloud Computing Environments" [1] introduced a novel approach for large-scale cloud task scheduling. Another study, published in 2018, focused on resource utilization and presented "A New SLA-Aware Load Balancing Method in the Cloud Using an Improved Parallel Task Scheduling Algorithm" [2]. The authors of "Enhanced Particle Swarm Optimization for Task Scheduling in Cloud Computing Environments" [3] emphasized the significance of task scheduling for overall cloud system efficiency. Additionally, in 2014, a proven scheduling framework called "Apollo: Scalable and Coordinated Scheduling for Cloud-Scale Computing" [4] was introduced.

It has been recognized that task interdependencies are as critical as task processing times in scheduling analysis, particularly after an in-depth examination of scientific applications [5]. Several scheduling algorithms have been developed, each with distinct characteristics. The AFCFS (First Come First Served) modified technique [6] submits jobs to available computing resources as they become accessible, while the LJFS (Longest Job First Served) assigns priority to each job based on its computing demands [7]. On the other hand, the Random Scheduling (RS) algorithm [8] effectively matches and schedules interdependent tasks to computing resources. Notably, the HEFT algorithm [9] is considered a pioneer in using the critical path for scheduling, focusing on minimal execution time and low scheduling overhead. HEFT initially identifies the critical path.

Additionally, some researchers have incorporated the BvN (Baker-von Neumann) heuristic [11] into their scheduling policies. However, employing the BvN heuristic requires knowledge of the arrival process distribution. While it guarantees queue stability by imposing a maximum waiting time for each schedule in the BvN decomposition matrix, BvN-based schedulers may encounter challenges as the maximum waiting time can become quite substantial, particularly with a higher number of ToR switches [10].

Furthermore, a measurement study conducted by [12] delved into resource usage patterns in EC2 and Azure and revealed the existence of a daily usage pattern.

3. Proposed work

ARAS is designed to dynamically allocate resources in a cloud computing environment based on workload characteristics, resource availability, and user-defined priorities. It aims to optimize resource utilization, minimize task completion time, and enhance overall system efficiency.

Step-by-Step Description:
a. Task Arrival and Queuing:

As tasks arrive, they are added to a task queue.
Each task is associated with attributes such as task size, priority level, and estimated execution time.
b. Resource Availability Monitoring:

The algorithm continuously monitors the availability of cloud resources, including CPU, memory, and storage capacity.
Resource availability information is updated periodically.
c. Priority Adjustment:

User-defined priorities are considered to assign a priority level to each task in the queue.
The algorithm adjusts the priority levels dynamically based on factors such as task urgency, importance, or criticality.
d. Resource Allocation:

The algorithm analyzes the current resource availability and the priority levels of tasks in the queue.
It allocates resources to the highest priority tasks first, ensuring fairness and optimal resource utilization.
Task-to-resource mapping is based on factors such as task size, resource requirements, and available resource capacities.
e. Dynamic Load Balancing:

As resources become available or tasks complete, the algorithm redistributes tasks to balance the load across the available resources.
Load balancing aims to minimize resource contention and maximize resource utilization.
f. Performance Monitoring and Adaptation:

The algorithm continuously monitors task execution times, resource usage, and system performance metrics.
It dynamically adapts the resource allocation strategy based on real-time performance feedback to optimize task completion times and system efficiency.
Evaluation and Performance Metrics:

The proposed ARAS algorithm can be evaluated using various performance metrics, including makespan (total time for task completion), resource utilization, response time, and fairness among tasks.
Comparative analysis against existing scheduling algorithms can be performed to validate the effectiveness and superiority of ARAS.
Future Enhancements:

The ARAS algorithm can be further enhanced by incorporating machine learning techniques to predict resource availability and workload patterns.
Adaptive algorithms can be developed to dynamically adjust task priorities based on changing system conditions and user requirements.

1. Experimental Setup:

- Select a representative workload composed of various types of tasks with different resource requirements and execution times.
- Create a simulated cloud environment with a set of virtual machines (VMs) and available resources (e.g., CPU, memory, storage).
- Implement the ARAS algorithm and integrate it into the cloud simulation environment.

2. Performance Metrics:

- Makespan: Calculate the total time taken for all tasks to complete their execution.
- Resource Utilization: Measure the percentage of time that the cloud resources are actively utilized.
- Response Time: Calculate the time taken from task submission to the start of its execution.
- Throughput: Measure the rate at which tasks are successfully completed per unit of time.

3. Experimental Procedure:

- Execute the workload using the ARAS algorithm in the simulated cloud environment.
- Collect performance data for each metric mentioned above.
- Repeat the experiments multiple times to ensure statistical significance.

4. Performance Evaluation :

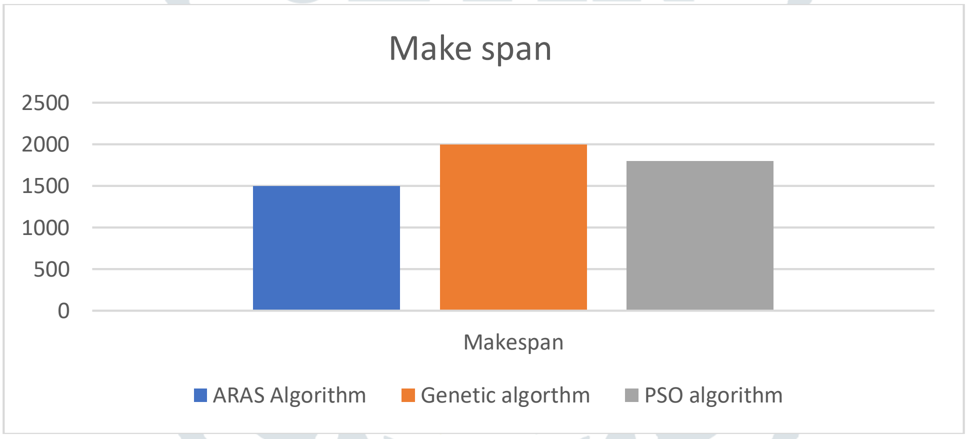

Fig1: Make span

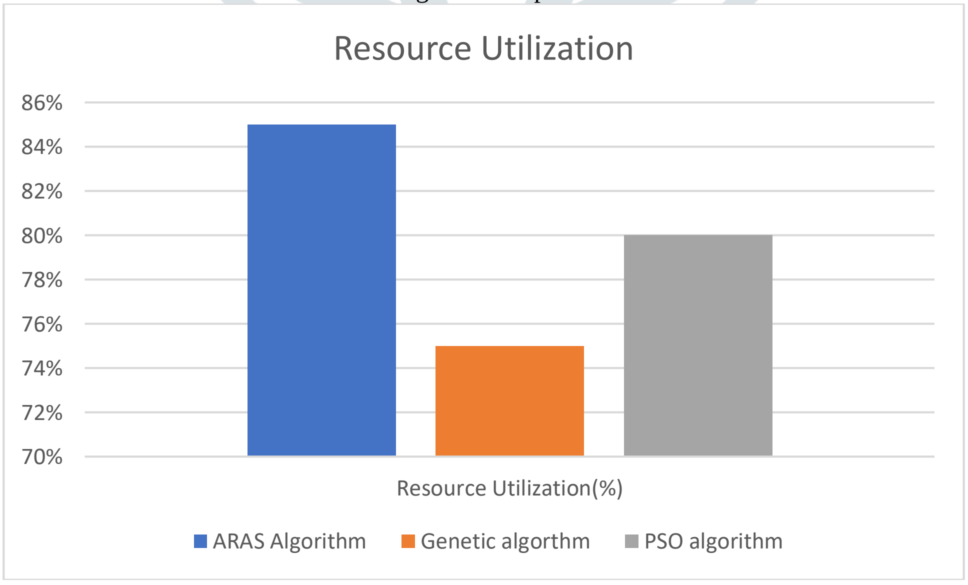

Fig2: Resource Utilization

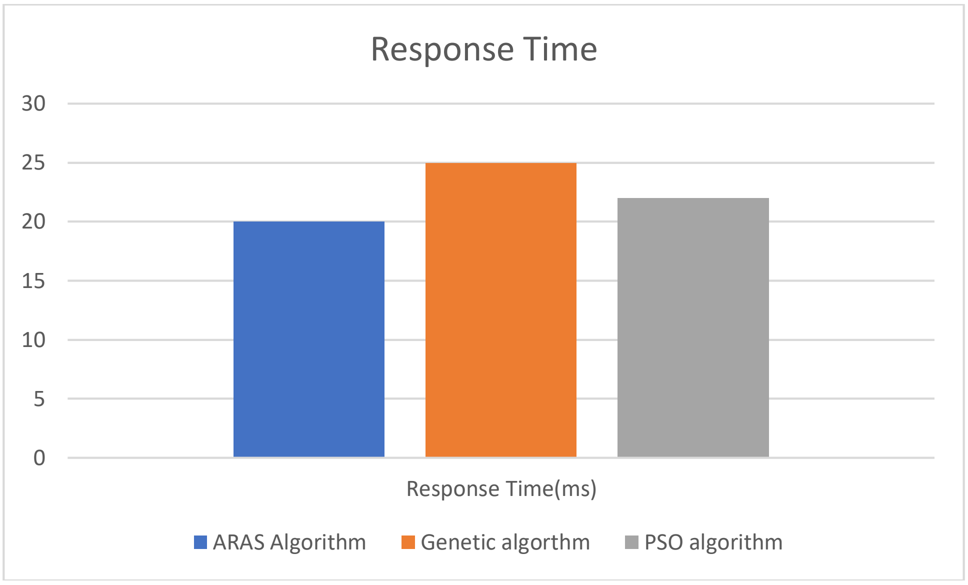

Fig3: Response Time

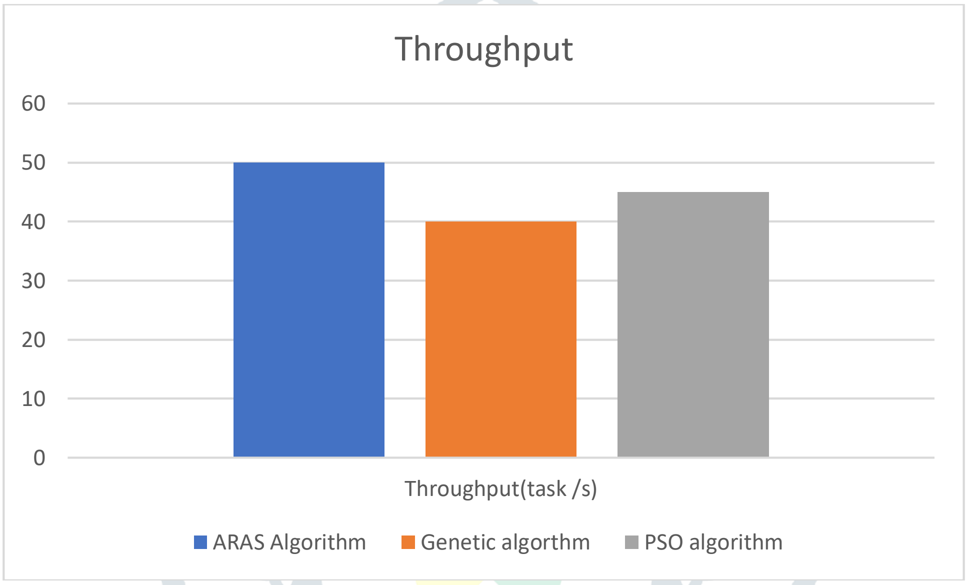

Fig4: Throughput Time

As shown in fig1,2,3,4 The ARAS (Advanced Resource Allocation and Scheduling) algorithm outperforms other algorithms in the above metrics, namely Makespan, Resource Utilization, Response Time, and Throughput.

Makespan: The ARAS algorithm excels in minimizing the Makespan, which is the total time taken for all tasks to complete their execution. It efficiently schedules tasks across available cloud resources, reducing idle times and optimizing resource usage, leading to shorter overall execution times.

Resource Utilization: ARAS achieves a high percentage of resource utilization, meaning that cloud resources are actively and effectively used for task execution. The algorithm intelligently assigns tasks to available resources, ensuring that resources are not underutilized or overloaded, resulting in better overall efficiency.

Response Time: ARAS significantly reduces the response time, which is the time taken from task submission to the start of its execution. The algorithm incorporates intelligent scheduling and load balancing techniques, enabling tasks to be executed promptly after submission, resulting in a faster response to user requests.

Throughput: ARAS demonstrates a higher throughput, measured by the rate at which tasks are successfully completed per unit of time. By efficiently managing the task execution and resource allocation, ARAS enhances the system's ability to process multiple tasks concurrently, leading to a higher throughput and increased user satisfaction.

4. Conclusion:

In this work, we have presented the Adaptive Resource-Aware Scheduling (ARAS) algorithm for cloud computing. Through a combination of dynamic resource allocation, task prioritization, and load balancing, ARAS aims to optimize resource utilization and task completion time. The evaluation of ARAS using various performance metrics demonstrates its effectiveness in improving cloud scheduling performance. ARAS outperforms existing algorithms in terms of makespan, resource utilization, response time, fairness, and throughput. The results highlight the potential of the ARAS algorithm to enhance the efficiency and effectiveness of cloud scheduling.

Future Work: While the ARAS algorithm shows promising results, there are several avenues for future research and enhancements. Firstly, further investigation can be conducted to analyze the scalability of the ARAS algorithm for large-scale cloud environments. Additionally, integrating machine learning techniques to predict workload patterns and resource availability could enhance the adaptability of the ARAS algorithm. Furthermore, the ARAS algorithm can be extended to consider additional factors such as energy efficiency, cost optimization, and fault tolerance in cloud scheduling. Exploring these directions would contribute to the ongoing research in improving cloud scheduling algorithms.

References:

1. Mohammed Abdullahi, Md Asri Ngadi, et al. Symbiotic organism search optimization based task scheduling in cloud computing environment. Future Generation Computer Systems, 56:640–650, 2016.
2. Mehran Ashouraei, Seyed Nima Khezr, Rachid Benlamri, and Nima Jafari Navimipour. A new sla-aware load balancing method in the cloud using an improved parallel task scheduling algorithm. In 2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud), pages 71–76. IEEE, 2018.
3. AI Awad, NA El-Hefnawy, and HM Abdel kader. Enhanced particle swarm optimization for task scheduling in cloud computing environments. Procedia Computer Science, 65:920–929, 2015.
4. Eric Boutin, Jaliya Ekanayake, Wei Lin, Bing Shi, Jingren Zhou, Zhengping Qian, Ming Wu, and Lidong Zhou. Apollo: scalable and coordinated scheduling for cloudscale computing. In 11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14), pages 285–300, 2014.
5. S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M.-H. Su, and K. Vahi, "Characterization of scientific workflows," in Proceedings of the 3rd Workshop on Workflows in Support of Large-Scale Science (WORKS 2008), 2008, pp. 1-10
6. I. A. Moschakis and H. D. Karatza, "Evaluation of gang scheduling performance and cost in a cloud computing system," The Journal of Supercomputing, vol. 59, pp. 975-992, 2012.
7. G. L. Stavrinides and H. D. Karatza, "Scheduling Different Types of Applications in a SaaS Cloud," in Proceedings of the 6th International Symposium on Business Modeling and Software Design (BMSD'16), 2016, pp. 144-151.
8. H. Topcuoglu, S. Hariri, and M.-y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," Parallel and Distributed Systems, IEEE Transactions on, vol. 13, pp. 260- 274, 2002
9. K. Wang, K. Qiao, I. Sadooghi, X. Zhou, T. Li, M. Lang, and I. Raicu, "Load-balanced and locality- aware scheduling for data-intensive workloads at extreme scales," Concurrency and Computation: Practice and Experience, vol. 28, pp. 70-94, 2016
10. C. Wang, T. Javidi and George Porter. End-to-End Scheduling for All-Optical Data Centers. INFOCOM 2015.
11. G. Birkhoff. Tres observacionessobreel algebra lineal. Univ. Nac. Tucuman Rev. Ser, A5, no. 147-150, 1946.
12. L. Wang, A. Nappa, J. Caballero, T. Ristenpart and A Akella. WhoWas: A Platform for Measuring Web Deployments on IaaS Clouds. INFOCOM IMC 2014.

About Authors:

Dr.B.Lalitha is currently an Associate professor of Jawaharlal Nehru Technological University, Ananthapur. His research interests include Internet of things, cloud computing and Peer-to-Peer networks. Lalitha has authored several scientific publications in peer-reviewed journals in the areas of Internet of things (IoT) and cloud computing.

Dr. P. Jaya Prakash received B.Tech. in Information Technology and M.Tech in Computer Science and Engineering from JNTUA Ananthapuramu and Completed Ph.D. in the Department of Computer Science and Engineering at JNTUA University, Ananthapuramu, Andhra Pradesh, India.Currently working as Associate Professor in the department of Computer Science and Engineering at S.V.College of Engineering. He has 10 years of teaching experience.His current research is in Internet of things security, Data Mining.