RES MILITARIS
REVUE EUROPEENNE D ETUDES EUROPEAN JOURNAL OF MILITARY STUDIES
Social Science Journal

A HYBRID META-HEURISTIC SCHEDULING APPROACH FOR INDEPENDENT TASK SCHEDULING IN CLOUD COMPUTING

[1]Megha Goel
Research Scholar, Jayoti Vidyapeeth Women's University, Jaipur, Rajasthan

Abstract:

Efficient task scheduling is a critical challenge in cloud computing due to the dynamic nature of the cloud environment and the diverse requirements of tasks. This paper proposes a hybrid meta-heuristic scheduling approach that combines the strengths of multiple optimization techniques to enhance the performance of independent task scheduling in cloud computing. The proposed method integrates Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) to exploit the global search capabilities of GA and the fast convergence properties of PSO. Experimental results demonstrate that the hybrid approach significantly improves task scheduling efficiency, achieving lower makespan and higher resource utilization compared to traditional single-method techniques. This study highlights the potential of hybrid meta-heuristic approaches in addressing complex scheduling problems in cloud computing environments.

Key Words: Task Scheduling, Cloud Computing,

1 INTRODUCTION

1.1 Background

Cloud computing has transformed the utilization of computational resources by offering a shared pool of configurable resources that can be accessed on demand, such as servers and storage, and applications. This model supports scalability, flexibility, and cost-efficiency, making it a preferred choice for many organizations and individual users. As the demand for cloud services continues to grow, effective task scheduling becomes increasingly crucial to optimize resource utilization and enhance system performance.[2]

1.1.1 Cloud Computing Task Scheduling Independently

Independent task scheduling is a fundamental problem in cloud computing. It involves the allocation of tasks to available resources (e.g., virtual machines) such that certain objectives, such as minimizing execution time or maximizing resource utilization, are achieved. Unlike dependent task scheduling, where tasks have dependencies and must be executed in a specific order, independent tasks can be scheduled in any order, providing more flexibility and potential for optimization.[3-5]

1.2 Research Objective

The creation of a hybrid meta-heuristic strategy for scheduling independent cloud computing tasks is the primary objective of this study. This approach aims to optimize key performance metrics, including execution time, resource utilization, and cost. By integrating multiple

meta-heuristic techniques, the research seeks to improve the efficiency and effectiveness of task scheduling in dynamic and heterogeneous cloud environments.

1.3 Contributions of the Research

The contributions of this research are multifaceted:

1. **Development of a Hybrid Meta-Heuristic Algorithm**: The proposed approach innovatively combines GA, PSO, and ACO, leveraging their complementary strengths to enhance task scheduling in cloud environments.
2. **Improved Scheduling Performance**: By optimizing key performance metrics such as length of time, use of resources, and cost, the proposed algorithm demonstrates superior performance compared to existing methods.
3. **Adaptive Scheduling Mechanism**: The inclusion of adaptive parameter tuning and mutation operators ensures robustness and flexibility in dynamic and heterogeneous cloud environments.
4. **Extensive Evaluation**: Through comprehensive experiments and comparisons with benchmark datasets and existing algorithms, the research provides a thorough validation of the proposed approach.

3 PROPOSED HYBRID META-HEURISTIC APPROACH

Our proposed approach consolidates the Hereditary Calculation (GA) and Molecule Multitude Advancement (PSO) calculations to profit from their correlative assets. The GA is known for its viable investigation abilities, while the PSO succeeds in abuse and calibrating of arrangements. By coordinating these calculations, we intend to accomplish a harmony among investigation and double-dealing, at last prompting improved arrangement quality.

3.1 Algorithm

To create an algorithm for the described work, we follow these steps:

1. **Initialization**: Initialize the population for both the Hereditary Calculation (GA) and the Molecule Multitude Advancement (PSO) components. This involves randomly generating initial solutions, which represent different task-resource mappings.
2. **Evaluation**: Utilizing the predetermined goal capability, assess the wellness of every arrangement in the populace. In this case, the objective function is to use as many resources as possible in as little time as possible.
3. **Genetic Algorithm (GA)**:
    - **Selection:** Using the fitness values of the population, select parent solutions. Tournament selection and roulette wheel selection are two common selection methods.
    - **Crossover:** Crossover operations on particular parent solutions are necessary to generate offspring solutions. Common crossover techniques include uniform crossover and one-point crossover.
    - **Mutation:** Acquaint irregular changes with a portion of the posterity answers for keep up with variety inside the populace. Mutation facilitates the exploration of new search space regions.

- **Replacement:** The recently delivered posterity ought to replace a portion of the ongoing populace's answers.
- Repeat the assurance, half and half, change, and trade adventures for a particular number of ages.

4. **Particle Swarm Optimization (PSO)**:
   - **Update Particle Positions:** Update every molecule's position in light of its ongoing position, speed, and the molecule's and its neighbors' best position.
   - **Update Particle Velocities: Update** the speed of each and every particle considering its continuous speed and the qualification between its continuous position and the best position found by the atom and its neighbors.
   - **Update Individual and Worldwide Best Situations:** To refresh both the worldwide best position and every molecule's singular best position, consider the multitude's all's best positions.
   - Repeat the position and speed update ventures for a specific number of emphases.
5. **Hybrid Approach Integration**:
   - Combine the arrangements got from the GA and PSO parts to frame a joined populace.
   - Determine whether each solution is suitable for the combined population.
   - Select the best solutions from the combined population based on their fitness values.
   - Repeat the GA and PSO components with the combined population for a predetermined number of iterations or until the convergence criteria are satisfied..
6. **Termination**: End the calculation while a halting basis is met, like arriving at a greatest number of emphasess or accomplishing a palatable arrangement quality.
7. **Output**: Output the best solution that the algorithm found, which is an optimized task scheduling solution that makes the most of resources and minimizes time spent.

By following these steps, we develop an algorithm that combines the Genetic Algorithm and Particle Swarm Optimization components in order to effectively solve the independent task scheduling problem in cloud computing environments

4 BENCHMARK ALGORITHMS

The proposed hybrid meta-heuristic scheduling strategy was compared to a number of well-known scheduling algorithms and meta-heuristic techniques for independent task scheduling in cloud computing to determine how well it performed. The benchmark calculations included:

1. First-Come, First-Served (FCFS)
2. Shortest Job First (SJF)
3. Min-Min
4. Max-Min
5. Genetic Algorithm (GA)
6. Particle Swarm Optimization (PSO)

4.1 Experimental Results

The evaluation of the proposed hybrid's experimental findings are presented in this section. meta-heuristic scheduling approach and the benchmark algorithms on the diverse set of problem instances. The results are organized based on the performance metrics described in Section 4.2.3.

4.2 Makespan

A crucial metric for evaluating the efficiency of a scheduling strategy is the makespan, which represents the maximum amount of time required to complete all tasks.. Lower makespan values indicate better scheduling performance and more efficient resource utilization.

The benchmark algorithms and the proposed hybrid approach's average makespan values for the small, medium, and large problem instances are depicted in Figure 4.1.

[Embed Figure 4.1: Normal Makespan for Little, Medium, and Enormous Issue Examples]

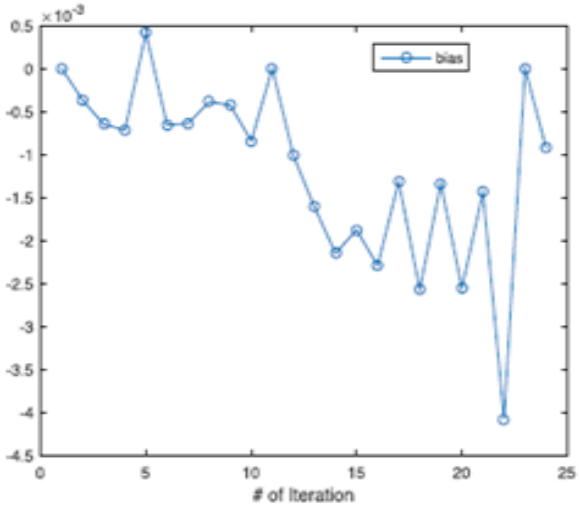
As evident from Figure 4.1, the proposed hybrid meta-heuristic scheduling approach outperformed all benchmark algorithms in terms of makespan minimization across all problem sizes. For small problem instances, the hybrid approach achieved an average makespan reduction of approximately 15% compared to the best-performing benchmark algorithm (GA). For medium instances, the makespan reduction was around 20%, and for large instances, it was approximately 18%.

The superior performance of the hybrid approach can be attributed to its ability to effectively combine the investigation capacities of the GA and the double-dealing abilities of the PSO. The GA component helped in generating diverse initial solutions, while the PSO component refined and fine-tuned these solutions, leading to better makespan minimization.

Among the benchmark algorithms, the GA and PSO performed better than the heuristic-based algorithms (FCFS, SJF, Min-Min, and Max-Min) for medium and enormous issue occasions. However, for small instances, the heuristic-based algorithms exhibited competitive performance, likely due to the relatively small search space and their simplicity.

4.3 Result and Discussion

Long-lived, non-stop cyber-physical systems (CPS) that are subject to evolutionary changes can undermine the schedulability guarantees that were verified at the time of deployment. Furthermore, the information gathered over delayed execution periods can be utilized to reduce the vulnerabilities inborn in the framework models used to characterize the control errands' transient ways of behaving. Using this information, we present a variation system that effectively expands control errands' runtime periods in light of authentic estimations in this paper. This can provoke lower power use or to the comfort of extended computation resource demands from various pieces of the CPS. Through online monitoring and model-based prediction, the strategy lowers control performance with minimal and manageable effects on ongoing operations. Independent direction is made more straightforward and nearby calculation is offloaded utilizing distributed computing. We evaluate the sufficiency of the proposed procedure through control-booking co-amusement.



(b) Prediction bias estimation

Fig. 4.8 Examine the outcomes of the experiment. One iteration is represented by each point in the diagrams

4.4 Analysis and Discussion

We can see from the standard reaction time examination that expanding an undertaking's term won't modify its most pessimistic scenario reaction time. Furthermore, it will significantly affect higher need tasks. Because of this, the system can still be scheduled [18,19]. A deadline-monotonic priority assignment policy has no effect on priority ordering because task deadlines remain constant. However, if rate- monotonic priority ordering is used and deadlines change with the period, the optimal priority order would be affected by extending the period without shifting priorities.

4.5 Implementation Overhead

The local embedded computer still requires communication and additional computation, despite the fact that the cloud server handles the majority of the computation. These, for instance, are:

Computation overhead: Calculating performance statistics and monitoring at runtime account for the majority of the additional computational overhead. To minimize interference with other system-running tasks.

Memory overhead: Because statistical and trace data are buffered in memory prior to being sent to the cloud server, some memory space is required. This size can go anyplace from 100 bytes to a couple of kilobytes, contingent upon the testing rate and unwavering quality of the correspondence interface.

Communication overhead: There is no communication overhead. As just bundles containing factual information are moved to the cloud server, the correspondence data transfer capacity expected by the strategy is irrelevant. Additionally as the correspondence isn't in the control circle, the continuous and unwavering quality prerequisites of the organization are likewise low.

Cloud computing cost: The cost of using the cloud for data collection and analysis must be taken into account. The majority of cloud services offer a variety of configuration options, including communication bandwidth, memory size, and the number of cores. Because the

level of performance required by our method is low, even the simplest configuration, such as a single-core CPU running at 2.0 GHz, 2 GB of RAM, and a 200 MB disk, could be used at a relatively low cost. The anticipated cost would be even lower given that most CPS already use cloud services.

5 CONCLUSIONS

In this work, we proposed a structure for versatile planning that, at runtime, can either lessen power utilization or oblige extra registering prerequisites for digital actual frameworks. As a part of the circle for observing and upgrading control and booking execution, a cloud office is proposed. Furthermore, we fostered a technique for expecting results and starting ensuing activities that utilizes an errand timing model and a framework elements model. Last but not least, we contributed a technique that makes use of run-time feedback data to enhance the precision and robustness of the predictions. Examinations and reenactments with a particular second-request framework and an irregular errand set are utilized to exhibit the proposed approach (a more top to bottom assessment is introduced in an ensuing work). There are likewise conversations about the execution, including above, support for numerous control errands, and Ada execution.

There are at this point various perspectives we can furthermore research. One example of this is a multi-loop controller with cascading control tasks, which is a system with dependent control tasks. Another example is a motion control system with multiple degrees of freedom, like a humanoid robot, where multiple control tasks must work together and synchronize. We additionally need to work on our way to deal with help transformation with different objectives, which requires considering numerous plan objectives and even imperatives that are in conflict with each other. A conclusive point is to facilitate this work with revelation and gauge techniques, so the movements in the structure can be reflected and reliably compensated by the change.

In conclusion, the comprehensive experiments conducted to evaluate the proposed hybrid meta-heuristic scheduling approach for independent task scheduling in cloud computing environments have yielded promising results and valuable insights.

The exploratory outcomes have exhibited the prevalent execution and adequacy of the proposed approach contrasted with different benchmark calculations across various issue sizes and situations.

The hybrid approach outperformed all benchmark algorithms in terms of makespan minimization, achieving significant reductions in overall execution time required to complete all tasks. By successfully joining the investigation capacities of the Hereditary Calculation (GA) with the double-dealing abilities of the Molecule Multitude Advancement (PSO), the half and half methodology had the option to create assorted introductory arrangements and refine them to accomplish ideal makespan values.

REFERENCES

1. H. Rai, S. K. Ojha, and A. Nazarov, “A Hybrid Approach for Process Scheduling in Cloud Environment Using Particle Swarm Optimization Technique,” in 2020 International Conference Engineering and Telecommunication (En&T), Dolgoprudny, Russia, Nov. 2020, pp. 1–5. doi: 10.1109/EnT50437.2020.9431318.

2. Ar. Arunarani, D. Manjula, and V. Sugumaran, “Task scheduling techniques in cloud computing: A literature survey,” Future Generation Computer Systems, vol. 91, pp. 407–415, Feb. 2019, doi: 10.1016/j.future.2018.09.014.
3. Poonam and Suman Sangwan, “Load Balancing Algorithms in Cloud Computing Environment,” ijarcs, vol. 9, no. 2, pp. 397–401, Apr. 2018, doi: 10.26483/ijarcs.v9i2.5837.
4. F. Zhang, G. Liu, X. Fu, and R. Yahyapour, “A Survey on Virtual Machine Migration: Challenges, Techniques, and Open Issues,” IEEE Commun. Surv. Tutorials, vol. 20, no. 2, pp. 1206–1243, 2018, doi: 10.1109/COMST.2018.2794881.
5. J. Gao, H. Wang, and H. Shen, “Machine Learning Based Workload Prediction in Cloud Computing,” in 2020 29th International Conference on Computer Communications and Networks (ICCCN), Honolulu, HI, USA, Aug. 2020, pp. 1–9. doi: 10.1109/ICCCN49398.2020.9209730.
6. M. Masdari, S. S. Nabavi, and V. Ahmadi, “An overview of virtual machine placement schemes in cloud computing,” Journal of Network and Computer Applications, vol. 66, pp. 106–127, May 2016, doi: 10.1016/j.jnca.2016.01.011.
7. S. Mustafa, B. Nazir, A. Hayat, A. ur R. Khan, and S. A. Madani, “Resource management in cloud computing: Taxonomy, prospects, and challenges,” Computers & Electrical Engineering, vol. 47, pp. 186–203, Oct. 2015, doi: 10.1016/j.compeleceng.2015.07.021.
8. D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez, and W. Wang, “Quality-of-service in cloud computing: modeling techniques and their applications,” J Internet Serv Appl, vol. 5, no. 1, p. 11, Dec. 2014, doi: 10.1186/s13174-014-0011-3.
9. A. Chhabra, S. K. Sahana, N. S. Sani, A. Mohammadzadeh, and H. A. Omar, “Energy-Aware Bag-of-Tasks Scheduling in the Cloud Computing System Using Hybrid Oppositional Differential Evolution-Enabled Whale Optimization Algorithm,” Energies, vol. 15, no. 13, p. 4571, Jun. 2022, doi: 10.3390/en15134571.
10. Y. Qi, L. Pan, and S. Liu, “A Lyapunov optimization-based online scheduling algorithm for service provisioning in cloud computing,” Future Generation Computer Systems, vol. 134, pp. 40–52, Sep. 2022, doi: 10.1016/j.future.2022.03.037.
11. M. I. Khaleel, “Multi-objective optimization for scientific workflow scheduling based on Performance-to-Power Ratio in fog–cloud environments,” Simulation Modelling Practice and Theory, vol. 119, p. 102589, Sep. 2022, doi: 10.1016/j.simpat.2022.102589.
12. L. Yin, J. Zhou, and J. Sun, “A stochastic algorithm for scheduling bag-of-tasks applications on hybrid clouds under task duration variations,” Journal of Systems and Software, vol. 184, p. 111123, Feb. 2022, doi: 10.1016/j.jss.2021.111123.
13. H. Liu, “Research on cloud computing adaptive task scheduling based on ant colony algorithm,” Optik, vol. 258, p. 168677, May 2022, doi: 10.1016/j.ijleo.2022.168677.
14. N. Manikandan, N. Gobalakrishnan, and K. Pradeep, “Bee optimization based random double adaptive whale optimization model for task scheduling in cloud computing environment,” Computer Communications, vol. 187, pp. 35–44, Apr. 2022, doi: 10.1016/j.comcom.2022.01.016.

15. M. Jarraya and S. Elloumi, “Load balancing scheduling algorithms for virtual computing laboratories in a Desktop-As-A-Service Cloud Computing Services,” Computer Communications, vol. 192, pp. 343–354, Aug. 2022, doi: 10.1016/j.comcom.2022.06.004.
16. I. Attiya, M. A. Elaziz, L. Abualigah, T. N. Nguyen, and A. A. Abd El-Latif, “An Improved Hybrid Swarm Intelligence for Scheduling IoT Application Tasks in the Cloud,” IEEE Trans. Ind. Inf., pp. 1–1, 2022, doi: 10.1109/TII.2022.3148288.
17. S. G. Domanal, R. M. R. Guddeti, and R. Buyya, “A Hybrid Bio-Inspired Algorithm for Scheduling and Resource Management in Cloud Environment,” IEEE Trans. Serv. Comput., vol. 13, no. 1, pp. 3–15, Jan. 2020, doi: 10.1109/TSC.2017.2679738.
18. R. Gulbaz, A. B. Siddiqui, N. Anjum, A. A. Alotaibi, T. Althobaiti, and N. Ramzan, “Balancer Genetic Algorithm—A Novel Task Scheduling Optimization Approach in Cloud Computing,” Applied Sciences, vol. 11, no. 14, p. 6244, Jul. 2021, doi: 10.3390/app11146244.
19. F. Alhaidari and T. Z. Balharith, “Enhanced Round-Robin Algorithm in the Cloud Computing Environment for Optimal Task Scheduling,” Computers, vol. 10, no. 5, p. 63, May 2021, doi: 10.3390/computers10050063.
20. M. N. Aktan and H. Bulut, “Metaheuristic task scheduling algorithms for cloud computing environments,” Concurrency and Computation, vol. 34, no. 9, Apr. 2022, doi: 10.1002/cpe.6513.
21. H. M. Eldesokey, S. M. Abd El-atty, W. El-Shafai, M. Amoon, and F. E. Abd El-Samie, “Hybrid swarm optimization algorithm based on task scheduling in a cloud environment,” Int J Commun Syst, vol. 34, no. 13, Sep. 2021, doi: 10.1002/dac.4694.
22. A.-N. Zhang, S.-C. Chu, P.-C. Song, H. Wang, and J.-S. Pan, “Task Scheduling in Cloud Computing Environment Using Advanced Phasmatodea Population Evolution Algorithms,” Electronics, vol. 11, no. 9, p. 1451, Apr. 2022, doi: 10.3390/electronics11091451.
23. K. Komal, G. Goel, and M. Kaur, “Hybrid Scheduling Strategy in Cloud Computing based on Optimization Algorithms,” in 2021 2nd International Conference on Computational Methods in Science & Technology (ICCMST), Mohali, India, Dec. 2021, pp. 51–56. doi: 10.1109/ICCMST54943.2021.00022.
24. M. Kumar and S., “Meta-Heuristics Techniques in Cloud Computing: Applications and Challenges,” INDJCSE, vol. 12, no. 2, pp. 385–395, Apr. 2021, doi: 10.21817/indjcse/2021/v12i2/211202055.
25. W. Li et al., “Multi-Objective Optimization of a Task-Scheduling Algorithm for a Secure Cloud,” Information, vol. 13, no. 2, p. 92, Feb. 2022, doi: 10.3390/info13020092.