A Probabilistic Evolutionary Hybrid (EDA-GA) for Multi-Objective Task Scheduling in Cloud Computing

Ms. Kale Jyoti S.[1], Dr. Gokuldhev M. [2]

[1]Research Scholar, Department of CSE, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology,Avadi, Chennai, India, Pincode: 600062 Email ID: vtd696@veltech.edu.in

[2]Associate Professor,Department of CSE, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Avadi, Chennai, India, Pincode: 600062 Email ID: drmgokuldhev@veltech.edu.in

HOW TO CITE:

Kale Jyoti S., Gokuldhev M. (2026). A Probabilistic Evolutionary Hybrid (EDA-GA) for Multi-Objective Task Scheduling in Cloud Computing. International Journal of Special Education, 41(11s), 637-648

COPYRIGHT STATEMENT:

Copyright: © 2026 Authors. Open access publication underthe terms and conditions ıe Creative CommonsAttribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

ABSTRACT:

Cloud computing requires effective task scheduling to allocate heterogeneous workloads on virtual machines while maintaining quality of service and resource utilisation. The study aims to solve the problem of minimising task completion time and improving load balance simultaneously in cloud environments. To tackle this problem, the EDA-GA hybrid algorithm is proposed which is composed of the probabilistic solution generation of Estimation of Distribution Algorithm and the crossover and mutation operations of Genetic Algorithm. The experiment was implemented in a simulation environment based on CloudSim with synthetic CSV datasets containing parameters of cloudlets, virtual machine configurations and scheduling scenarios. The proposed method was compared with standalone GA and standalone EDA based on makespan, load imbalance and optimisation time. Results show that EDA-GA outperformed EDA, reducing the average makespan from 93.7858 to 90.7256 and the average load imbalance from 0.2050 to 0.1287. However, the best overall average makespan and load balance was achieved by GA. This study introduces a hybrid scheduling framework and highlights its advantages, shortfalls and future optimisation requirements.

Keywords: EDA-GA hybrid algorithm, task scheduling, cloud computing, makespan optimization, load balancing

1. Introduction

Cloud computing has been emerged as a dominant computing paradigm due to its ability to provide users with on-demand and scalable access to configurable computing resources, storage and applications. Its growth has changed the way organisations deploy applications, manage infrastructure and deliver computing services without maintaining large local data centers [1]. The continuous growth of cloud services makes the efficient management of computing resources a critical research issue, mainly because cloud platforms are needed to process many heterogeneous tasks submitted by different users under dynamic workload conditions [2].

Cloud computing has many important issues and one of them is task scheduling which directly affects execution time, resource utilisation, quality of service, energy consumption and

system reliability. In cloud environments, user tasks need to be assigned to the available virtual machines (VMs) in a way to satisfy the multiple objectives such as reducing completion time, improving load balancing and avoiding inefficient resource allocation [3]. The traditional scheduling approaches often have huge differences in the task size and the capacity of the virtual machines and the user requirements. Thus, the task scheduling becomes a complicated optimisation problem, especially when multiple conflicting objectives are to be considered simultaneously [4].

Multi-objective optimisation is appropriate for cloud task scheduling as enhancing one objective may have adverse effects on another. For instance, minimising makespan may overload high performance virtual machines, and improving load balancing may increase total completion time. Such problems have been widely solved using evolutionary algorithms as they can search large solution spaces and yield close to optimal schedules without exhaustive search [5]. Among these methods genetic algorithms are useful as selection, crossover and mutation allow to explore diverse scheduling solutions. But they may suffer from premature convergence or require careful parameter tuning.

Another direction of optimisation is estimation of distribution algorithms, which generates new solutions based on learned probability model from promising candidate schedules [6]. This stochastic mechanism can guide the search towards more promising regions and improve the convergence behaviour. However, the too fast convergence of the probability model may restrict the diversity of the standalone EDA. Likewise, GA alone can preserve diversity but may need more iterations to find good schedules. Thus, the combination of EDA and GA is a meaningful approach, because EDA can lead the solution generation and GA can enhance the exploration by crossover and mutation.

In cloud computing, several nature-inspired and hybrid scheduling algorithms have been proposed to improve load balancing and task completion time [7]. Furthermore, multi-objective scheduling algorithms have been presented to deal simultaneously with makespan, load balancing and resource utilisation [8]. However, existing methods still have limitations in balancing the convergence speed and search diversity. Some of these methods perform well for specific workload sizes but may not be stable when the number of tasks or virtual machines changes [9]. This creates a need for hybrid scheduling models that are better able to balance exploitation and exploration.

The present study is entitled "An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing". The scope of study is limited to independent task scheduling in simulated cloud environment using synthetic CloudSim based datasets. The study evaluates the proposed approach on makespan, load imbalance, and optimisation time. It does not consider real-time task arrivals, task dependency graph, energy-aware scheduling, security constraints, or cost-based service level agreement. So, the results are to be interpreted within the scope of simulation-based task scheduling.

The importance of this work lies in its attempt to combine probability-based learning and evolutionary search for improving cloud task scheduling. As cloud systems are expected to provide scalable, efficient and balanced resource allocation under uncertain workloads, hybrid optimisation approaches are gaining more importance [10]. Efficient scheduling is also critical in modern edge-cloud and service computing environments were resource allocation decisions impact execution efficiency and service quality [11]. The comparison of GA, EDA and EDA-GA under same experimental conditions yields useful evidence on the strengths and limitations of hybrid approach. The novelty of this work is the integration of probabilistic solution generation based on EDA with crossover and mutation based on GA for the multi-objective cloud task scheduling problem. The hybrid framework proposed differs from the standalone EDA or GA by integrating guided convergence and search diversity. It is compared over the same CloudSim-based scenarios in terms

of makespan, load imbalance and optimisation time, which gives a specific evaluation of hybrid scheduling performance in simulated cloud environments. The objectives of this research are:

- To design an EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing.
- To evaluate the proposed algorithm using makespan, load imbalance, and optimization time.
- To compare the performance of the proposed EDA-GA algorithm with standalone GA and standalone EDA under the same CloudSim-based experimental scenarios.

2. Literature review

Task scheduling in cloud computing has been widely studied since it directly affects the execution time, resource utilisation, load balancing, energy consumption and service quality. We studied in detail workflow scheduling techniques and found that generally cloud scheduling problems are constrained by factors such as execution cost, deadlines, resource diversity, and quality-of-service requirements. Existing workflow scheduling methods provide useful optimisation techniques but are often designed for specific workflow structures and may not scale well to independent task scheduling or highly dynamic cloud workloads [12].

A broader survey on workflow scheduling domains further classified scheduling research on different computing infrastructures and highlighted the different goals considered in cloud, grid, cluster and distributed environments. The work showed an increasing focus on multi-objective optimisation in scheduling studies but also pointed to the difficulty of direct algorithm comparison due to the lack of standard experimental settings [13]. Recent survey work on machine learning for cloud resource allocation and workflow scheduling also pointed out the increasing use of intelligent methods to improve the cloud efficiency. However, the scheduling based on machine learning techniques often needs enough historical workload data and the simulation-based optimisation methods are still useful in the absence of real workload traces or when real workload traces are difficult to obtain [14].

In recent studies, several hybrid and nature-inspired approaches for multi-objective task scheduling have been proposed. A hybrid artificial bee colony algorithm with reinforcement learning for improving load balancing in cloud environments was presented. The main advantage of the system is that it merges swarm intelligence and learning-based decision improvement. Such methods, however, may increase the computational load and parameter complexity [15]. In another study, a decision tree-based algorithm was used for multi-objective task scheduling and the results showed that decision-support approaches can improve scheduling decisions. The disadvantage of such methods is that their effectiveness may depend on the structure and representativeness of data used for learning or decision-making [16].

Another important research direction is energy-efficient scheduling. A hybrid scheduling algorithm was proposed to reduce the energy consumption, while keeping the efficiency of task scheduling in cloud environment. Its strength lies in the consideration of energy-aware resource allocation, but the focus on energy efficiency may reduce the direct emphasis on makespan and load-balancing performance [17]. Likewise, an improved whale optimisation algorithm was proposed for independent task scheduling showing that metaheuristic algorithms can provide competitive performance in complex scheduling environments. However, whale optimisation and other similar swarm-based methods may require careful tuning to avoid slow convergence or instability in performance across workload sizes [18].

Methods based on genetic algorithms have been widely used because they are able to handle complex search spaces and multi-objective scheduling requirements. A non-dominated sorting genetic algorithm II model was proposed for fog-cloud task scheduling , showing the relevance of evolutionary multi-objective optimisation in distributed computing environments. Its strength is the ability to handle

conflicting objectives. However, such methods may still be subject to computational cost and be sensitive to parameters [19]. Energy and reliability-aware scheduling have also been studied for cloud workflows with the objective of not only reducing cost but also enhancing reliability under resource constraints. The studies show that scheduling is more considered as a multi-objective problem and not only as a single makespan minimisation problem [20].

Recent systematic literature reviews confirm that task scheduling in cloud computing is still an active research area. The existing literature adopts heuristic, metaheuristic, hybrid, and machine learning based approaches. The most common objectives are minimising makespan, cost, energy, reliability, and load balancing. However, many techniques are tested in different simulation settings, and it is hard to find a generally better approach [21]. In a recent review on the multi-objective optimisation techniques, the hybrid optimisation methods are also reported to be increasingly used due to the limitations of search diversity, convergence speed or workload adaptability of the single algorithms [22].

In general, previous studies have shown that cloud task scheduling has been addressed by some optimisation methods such as decision trees, artificial bee colony algorithms, whale optimisation, genetic algorithms and other hybrid techniques . The proposed methods provide useful improvements but still have room for improvement in terms of balancing fast convergence, strong search ability, makespan reduction and load balancing in a single scheduling framework. This paper addresses this gap by designing and evaluating an EDA-GA hybrid algorithm, which combines probability-based solution generation with evolutionary crossover and mutation, for multi-objective task scheduling in cloud computing.

3. Methodology

3.1 Research Design

In this paper, we evaluate the proposed EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing through experimental simulation-based design. The scheduling problem is formulated as the assignment of independent cloudlets to heterogeneous virtual machines, with every candidate solution representing a complete mapping of tasks to virtual machines. The objective is to minimise makespan with enhanced load distribution over available VMs. The proposed model is a combination of the probabilistic learning ability of the Estimation of Distribution Algorithm (EDA) and the evolutionary search ability of the Genetic Algorithm (GA).

The experimental framework was implemented in Java using CloudSim/CloudSim Plus as a simulation platform. The workflow starts with loading task, VM, and scenario data from CSV files. These records are converted to Cloudsim cloudlet and VM objects. Then three scheduling methods are executed in the same conditions, standalone GA, standalone EDA and proposed EDA-GA algorithm. Each algorithm generates a task-to-VM allocation, executes the schedule in the simulated cloud environment and logs makespan, load imbalance and optimisation time.

The EDA-GA workflow involves four major steps, as illustrated in Figure 1 below. First, a population of feasible schedules is generated. Second, the EDA builds a probabilistic model based on promising candidate schedules and samples new solutions. Third, the GA-based operations, including selection, crossover, and mutation, enhance search diversity and improve solution quality. Finally, each candidate schedule is evaluated using a multi-objective fitness function, and the best schedule is selected as the simulation output.

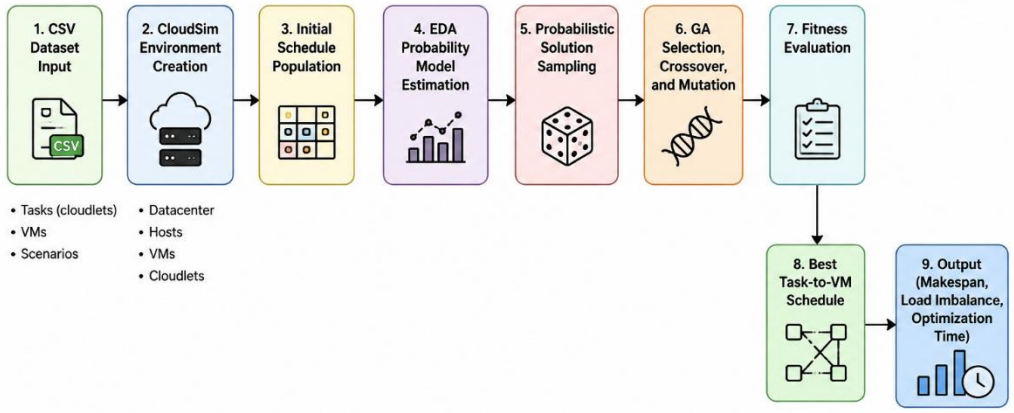

Figure 1. Workflow of the Proposed EDA-GA Hybrid Task Scheduling Framework.

3.2 Dataset Description and Input Variables

The experiment is performed on a synthetic dataset prepared in CSV format based on CloudSim. Algorithmic comparison was performed using synthetic workloads under controlled task and VM conditions. The dataset consists of three files, that are cloudsim_tasks.csv, cloudsim_vms.csv and experiment_scenarios.csv.

The task file specifies the task attributes at the cloudlet level including task ID, task length in million instructions, number of processing elements, input file size, output file size, priority and deadline. The VM file describes resource level properties like VM ID, MIPS capacity, number of processing elements, RAM, bandwidth, storage size and scheduler type. In the scenario file, workload configurations are defined by the number of tasks and the number of virtual machines.

The main independent variables are the number of tasks, the number of VMs, the length of tasks and the processing capacity of VMs. The dependent performance variables are makespan, load balance and optimisation time. All CSV records are validated before simulation and parsed into numerical values and converted into CloudSim compatible cloudlet and VM objects.

3.3 Experimental Scenarios and Sampling Strategy

The experimental population consists of independent cloud task scheduling scenarios where multiple user tasks are submitted to heterogeneous virtual machines. We take a scenario-based sampling approach to explore algorithm behaviour given controlled workload conditions. The selected scenarios are 100 and 200 tasks with 5, 10 and 20 VMs.

To be fair, GA, EDA and EDA-GA are performed on the identical task records, VM configurations and simulation settings in all the scenarios. This prevents bias from the dataset and guarantees any differences in performance are due to the scheduling algorithm and not to changes in the structure of the workload. For a fair comparison, each algorithm is given the same set of cloudlets and the same pool of VMs.

3.4 Algorithmic Procedure

The schedules in GA baseline are encoded as chromosomes; each gene represents the VM that is assigned to a given task. The GA begins with a randomly initialised population and uses fitness-based selection, crossover, mutation and replacement to produce improved schedules.

The EDA baseline selects promising candidate schedules from the population and estimates a probability distribution over task-to-VM assignments. New candidate schedules are then drawn from this probability model. This enables EDA to guide the search towards frequent and good quality VM assignments.

These mechanisms are combined in the proposed EDA-GA algorithm. Initial EDA learns from high quality solutions to create probability-guided schedules. GA then refines these schedules through crossover and mutation to maintain

diversity and prevent premature convergence. The hybrid structure is designed to combine guided convergence and exploratory search.

3.5 Fitness Evaluation and Performance Metrics

A multi-objective fitness function based on makespan and load imbalance is evaluated for each candidate schedule. The makespan can be defined as the total time for the completion of all assigned cloudlets. A lower makespan means faster execution of tasks. Load imbalance is the uneven distribution of workload among VMs. The smaller the imbalance, the more the resources are utilised and the more balanced the workload distribution is.

The performance of GA, EDA and EDA-GA is compared based on three metrics: makespan, load imbalance and optimisation time. Scheduling quality is evaluated by makespan and load imbalance, while computational overhead is measured by optimisation time. The output of the simulation is exported for analysis into makespan_results.csv and load_balance_results.csv.

4. Results

4.1 Experimental Performance

The experimental evaluation was performed to assess the scheduling performance of the proposed EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing. This is not a classification/prediction task. So, metrics like accuracy, precision, recall, f1-score, AUC, loss, etc do not apply. Instead, the relevant performance metrics are makespan, load imbalance, and time to optimise. Makespan is the total time required to execute all the cloudlets and load imbalance is the uniformity of the task's distribution across the virtual machines. Better scheduling performance is reflected in lower values for both metrics.

The experiment was conducted on 6 scheduling scenarios with 100 and 200 tasks and 5, 10 and 20 virtual machines. The makespan results for the GA, EDA and the proposed EDA-GA algorithm are shown in Table 1. The proposed method achieved the best makespan in two cases. EDA-GA achieved 140.2859 of makespan for 100 tasks and 5 VMs, while GA achieved 141.0254 and EDA achieved 153.5050. For 200 tasks and 5 VMs EDA-GA got 273.4806 compared to 275.1629 for GA and 289.5843 for EDA. The obtained results show that the proposed method is effective in case of limited number of available VMs.

Table 1. Makespan Comparison of Scheduling Algorithms

Task Count	VM Count	GA	EDA	EDA-GA	Best Algorithm
100	5	141.0254	153.5050	140.2859	EDA-GA
100	10	23.2444	24.0704	27.5171	GA
100	20	14.3446	14.9687	20.5627	GA
200	5	275.1629	289.5843	273.4806	EDA-GA
200	10	44.7584	51.9759	51.5135	GA
200	20	23.0759	28.6108	30.9936	GA

But GA got the best makespan in four scenarios. The average makespan for GA was 86.9352, for EDA-GA was 90.7256 and for EDA was 93.7858. Therefore, the proposed EDA-GA algorithm performed better than the stand-alone EDA, but it was not always better than the stand-alone GA for all the scenarios considered.

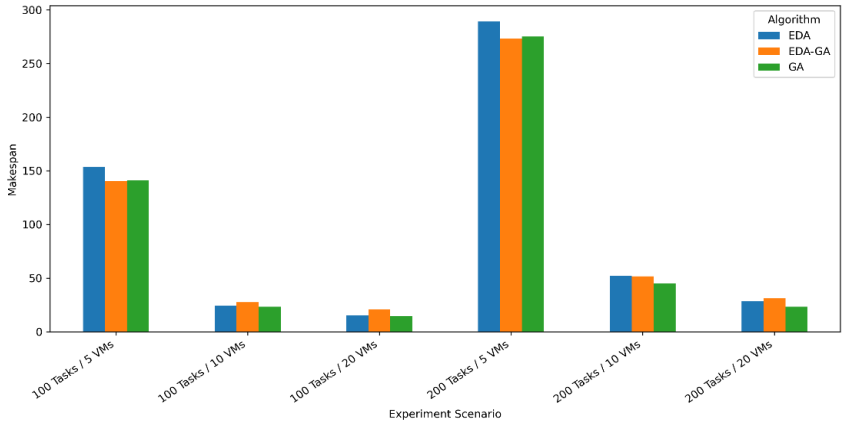

Figure 2. Makespan Comparison of GA, EDA, and EDA-GA Across Different Task and VM Configurations.

Figure 2. illustrates the makespan values obtained by GA, EDA and the proposed EDA-GA hybrid algorithm for different task and virtual machine configurations. A lower makespan means faster completion of tasks. The results show that EDA-GA achieves the lowest makespan in the 100-task/5-VM and 200-task/5-VM scenarios, while GA performs better in the other configurations.

4.2 Baseline Comparison

We compare the proposed EDA-GA algorithm with two baseline approaches, standalone GA and standalone EDA. The purpose of this comparison was to find out if the combination of EDA-based probability sampling and GA-based crossover and mutation enhances the task scheduling performance. The results of load imbalance are shown in Table 2.

Table 2. Load Imbalance Comparison of Scheduling Algorithms

Task Count	VM Count	GA	EDA	EDA-GA	Best Algorithm
100	5	0.0119	0.0309	0.0170	GA
100	10	0.0792	0.2386	0.0710	EDA-GA
100	20	0.1447	0.4560	0.3161	GA
200	5	0.0117	0.0436	0.0112	EDA-GA
200	10	0.0534	0.2156	0.1200	GA
200	20	0.1098	0.2456	0.2370	GA

The results show that GA achieved the best load imbalance in four scenarios and EDA-GA achieved the best in two scenarios. For 100 tasks and 10 VMs the least value of load imbalance was 0.0710 for EDA-GA, 0.0792 for GA and 0.2386 for EDA. For 200 tasks and 5 VMs, EDA-GA reached 0.0112, which is slightly better than GA and clearly better than EDA (0.0117 and 0.0436 respectively).

The average load imbalance was 0.0684 for GA, 0.1287 for EDA-GA and 0.2050 for EDA. These values indicate that EDA-GA improves load balancing with respect to EDA, but GA is still stronger for most of the tested cases. Thus, the proposed method provides performance improvement over EDA and competitive performance against GA, but it is a trade-off rather than a total improvement over all baselines.

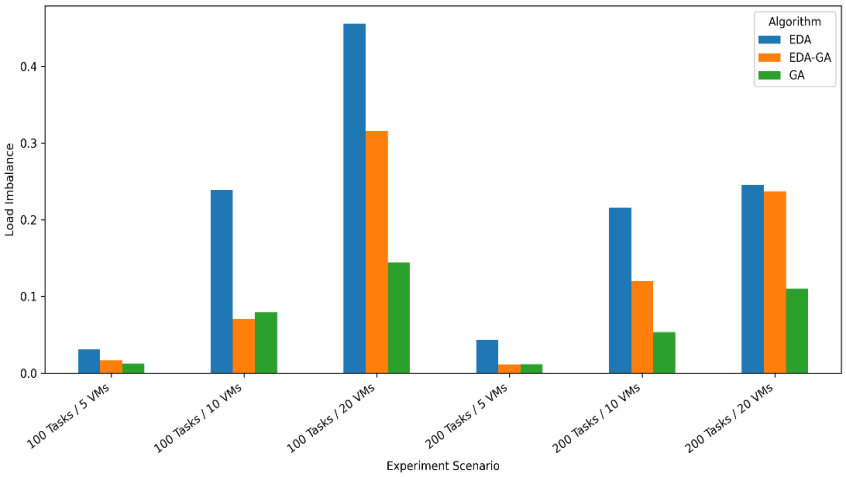

Figure 3. Load Imbalance Comparison of EDA-GA Against GA and EDA Baseline Algorithms.

Figure 3. shows the load imbalance values obtained by GA, EDA and the proposed hybrid EDA-GA algorithm for different configurations of tasks and virtual machines. The smaller the load imbalance, the better the workload distribution among virtual machines. The results show that EDA-GA improves load balancing with respect to stand-alone EDA, while GA has the lowest imbalance in most scenarios tested.

4.3 Ablation Study

The ablation analysis studies the impact of removing or modifying algorithmic components on the scheduling performance. In this experiment, we consider standalone EDA and standalone GA as ablated versions of the proposed hybrid EDA-GA algorithm. The EDA is the probability modelling part without the GA based evolutionary refinement, and GA is the evolutionary part without the EDA based probability sampling. The average performance of three algorithms over all tested scenarios is summarised in Table 3.

Table 3. Average Performance Summary

Algorithm	**Average Makespan**	**Average Load Imbalance**	**Average Optimization Time MS**
GA	86.9352	0.0684	197.5321
EDA	93.7858	0.2050	162.3880
EDA-GA	90.7256	0.1287	228.0971

The results indicate that the GA component is the most significant contributor to the final scheduling performance in the current experimental setup. GA had the smallest average makespan and the smallest average load imbalance. In contrast, EDA provided the lowest average values for both metrics. The hybrid algorithm of EDA-GA was in between GA and EDA, which means that the addition of GA-based crossover and mutation improves the search process when added to EDA.

The EDA component is still included in the hybrid method since the EDA-GA performs better than the standalone EDA for both average makespan and average load imbalance. Specifically, EDA-GA improved the average makespan from 93.7858 to 90.7256 and the average load imbalance from 0.2050 to 0.1287 in comparison to standalone EDA. Despite the better overall performance of GA, the comparison at the component level shows that the combination of the probability-based sampling with evolutionary crossover and mutation performs better than the EDA alone in terms of scheduling performance. This shows that the hybrid design presented in this paper can improve the search behaviour in the selected cloud task and VM configuration.

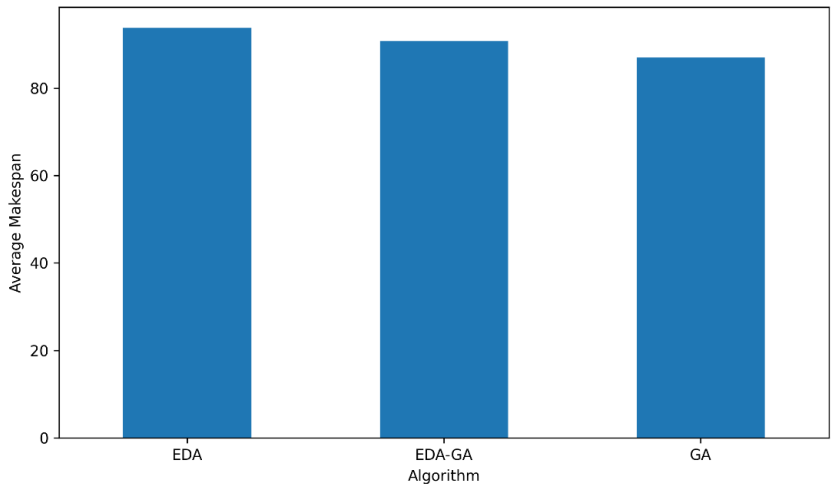

Figure 4. Ablation Study Comparison of GA-Only, EDA-Only, and Complete EDA-GA Hybrid Scheduling Model.

Figure 4. Performance effect of standalone GA component, standalone EDA component and complete EDA-GA hybrid model. The comparison shows how evolutionary search and probability-based sampling contribute to the performance in terms of makespan reduction and load balancing.

4.4 Additional Analysis

In a further analysis the computational efficiency was measured in terms of optimisation time. The average optimisation time for GA, EDA and EDA-GA was 197.5321 ms, 162.3880 ms and 228.0971 ms, respectively. This shows that EDA took the minimum time for optimisation, GA took the second place and EDA-GA took the maximum time as it is a hybrid of EDA based probability modelling with GA based crossover and mutation.

The overall results show that EDA-GA is better than the standalone EDA, and competitive to GA. The hybrid method was better in makespan for 100-task/5-VM, 200-task/5-VM, and better in load imbalance for 100-task/10-VM, 200-task/5-VM. These results show that the EDA-GA can be useful in selective cloud scheduling situations, especially when hybrid search can improve the probability-based scheduling. However, GA still showed a better performance in most of the scenarios tested, indicating that further tuning of the parameters could make the proposed hybrid model more competitive.

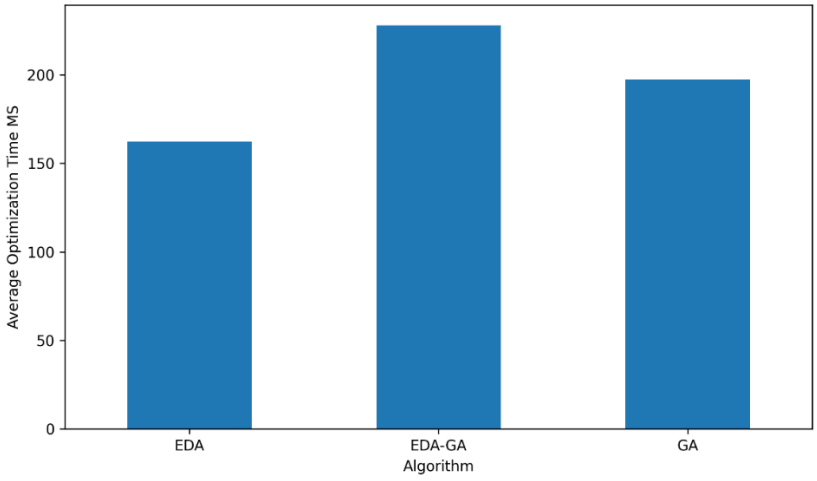

Figure 5. Average Optimization Time Comparison of GA, EDA, and EDA-GA.

Figure 5. compares the average optimisation time required by the three scheduling algorithms for all the experimental scenarios. EDA has the least average optimisation time, followed by GA, while the proposed EDA-GA hybrid algorithm has the highest optimisation time due to the combination of probability-based sampling with crossover and mutation operations.

5. Discussion

The experimental results show that the proposed EDA-GA hybrid algorithm is better in scheduling than standalone EDA but not always better than standalone GA for all the tested cases. The average makespan of GA, EDA and EDA-GA were 86.9352, 93.7858 and 90.7256 respectively. In a similar way, the average load imbalance was 0.0684 for GA, 0.2050 for EDA, and 0.1287 for EDA-GA. These values show that the hybrid method decreases the task completion time and workload imbalance in comparison to EDA. However, GA obtained the lowest average makespan and load imbalance, suggesting that the evolutionary search component played a stronger role than the probability-modeling component in the present experimental configuration.

The results are in line with the previous work that models the cloud task scheduling as a multi-objective optimisation problem considering execution time, resource utilisation, load balancing, and quality-of-service constraints [12]. They also support more general scheduling surveys that expose the challenges of workload heterogeneity, algorithm comparability and objective trade-offs in cloud and distributed computing environments [13]. As for hybrid scheduling, previous work based on artificial bee colony optimisation with reinforcement learning proved that the combination of search strategies can improve the scheduling quality but may lead to the increase of parameter complexity and computational overhead [15]. The pattern is like the current study where EDA-GA was better than EDA but took the highest average optimisation time of 228.0971 ms. The results are also in line with the literature on energy-efficient hybrid scheduling where usually an increase in optimisation capability comes at the cost of runtime, complexity and tuning effort [17].

The results are of practical importance for research on scheduling in clouds. First, hybrid optimisation can improve weaker standalone methods by coupling complementary search mechanisms. In this study, EDA search process is enhanced with GA-based crossover and mutation, which achieved lower average makespan and load imbalance than standalone EDA. Second, the algorithm performance seems to be highly dependent on the task and VM configuration. The hybrid method of EDA-GA has achieved the best makespan for the 100-task/5-VM and 200-task/5-VM cases. This indicates that this hybrid method may be more useful when the availability of VMs is limited. Third, the results confirm that scheduling algorithms should be evaluated by multiple metrics, not just makespan, because load imbalance and optimisation time give additional insight into the quality of scheduling and computational cost.

The study has several limitations. The experiments have been carried out using synthetic CloudSim based datasets and not real cloud workload traces. Hence the results should be interpreted in a simulation context. The evaluation was done for selected task and VM configurations, with 100 and 200 tasks and 5, 10 and 20 VMs. While these scenarios provide useful evidence of the behaviour of GA, EDA and EDA-GA in controlled conditions, a wider variety of workloads would further strengthen the generalisability of the findings. Moreover, the ablation analysis was performed to compare the standalone GA and EDA as component-level, while the individual effects of crossover, mutation, selection and probability updating can be further studied in future work.

Future work should extend the experiment to a broader set of workload conditions using real cloud traces and a more diverse set of VM configurations. The EDA-GA parameters, including population size, crossover probability, mutation rate, elitism, and probability update strategy, can be further optimised. Other

objectives such as energy consumption, execution cost, deadline violation, and resource failure can also be added for better realism. Future validation with repeated simulation settings and wider cloud scheduling environments can further investigate consistency of GA, EDA and EDA-GA performance.

6. Conclusion

This study addressed the problem of multi-objective task scheduling in cloud computing where heterogeneous tasks need to be assigned to virtual machines in a manner to reduce the completion time and improve the load balance. To tackle this problem, an EDA-GA hybrid algorithm is proposed, which combines EDA-based probabilistic solution generation with GA-based crossover and mutation. The evaluation of the method was made based on the makespan, load imbalance, and optimisation time by implementing a simulation environment based on CloudSim and comparison with standalone GA and standalone EDA. The results show that EDA-GA has better performance than EDA alone. The average makespan was reduced from 93.7858 to 90.7256 and the average load imbalance from 0.2050 to 0.1287 . GA achieves the lowest average load imbalance of 0.0684 and the best overall average makespan of 86.9352. Therefore, EDA-GA demonstrated competitive performance but did not always surpass GA for all tested cases. The main contribution of this paper is the design and evaluation of a hybrid EDA-GA scheduling framework for cloud task allocation. The work is limited by its dependence on synthetic CloudSim-based datasets and a selected set of task-VM configurations. Future work should consider wider workload conditions, real cloud traces, other objectives such as energy and cost, and better parameter tuning of the proposed hybrid algorithm.

References

M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “Above the clouds: A Berkeley view of cloud computing,” EECS Department, University of California, Berkeley, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2009-28, Feb. 2009.

R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” Future Generation Computer Systems, vol. 25, no. 6, pp. 599–616, 2009, doi: 10.1016/j.future.2008.12.001.

A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing,” Future Generation Computer Systems, vol. 28, no. 5, pp. 755–768, 2012, doi: 10.1016/j.future.2011.04.017.

S. Abrishami, M. Naghibzadeh, and D. H. J. Epema, “Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds,” Future Generation Computer Systems, vol. 29, no. 1, pp. 158–169, 2013, doi: 10.1016/j.future.2012.05.004.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002, doi: 10.1109/4235.996017.

M. Hauschild and M. Pelikan, “An introduction and survey of estimation of distribution algorithms,” Swarm and Evolutionary Computation, vol. 1, no. 3, pp. 111–128, 2011, doi: 10.1016/j.swevo.2011.08.003.

L. D. Dhinesh Babu and P. Venkata Krishna, “Honey bee behavior inspired load balancing of tasks in cloud computing environments,” Applied Soft Computing, vol. 13, no. 5, pp. 2292–2303, 2013, doi: 10.1016/j.asoc.2013.01.025.

M. Abdullahi, M. A. Ngadi, S. I. Dishing, S. I. M. Abdulhamid, and B. I. Ahmad, “An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment,” Journal of Network and Computer Applications, vol. 133, pp. 60–74, 2019, doi: 10.1016/j.jnca.2019.02.005.

Y. Li, S. Wang, X. Hong, and Y. Li, “Multi-objective task scheduling optimization in cloud computing based on genetic algorithm and differential evolution algorithm,” in Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 2018, pp. 4489–4494, doi: 10.23919/ChiCC.2018.8483505.

L. Yang, Y. Xia, L. Ye, R. Gao, and Y. Zhan, “A fully hybrid algorithm for deadline-constrained workflow scheduling in clouds,” IEEE Transactions on Cloud Computing, 2023, doi: 10.1109/TCC.2023.3269144.

Z. Sun, B. Zhang, C. Gu, R. Xie, B. Qian, and H. Huang, “ET2FA: A hybrid heuristic algorithm for deadline-constrained workflow scheduling in cloud,” IEEE Transactions on Services Computing, vol. 16, no. 3, pp. 1807–1821, 2023, doi: 10.1109/TSC.2022.3196620.

M. Masdari, S. ValiKardan, Z. Shahi, and S. I. Azar, “Towards workflow scheduling in cloud computing: A comprehensive analysis,” Journal of Network and Computer Applications, vol. 66, pp. 64–82, 2016, doi: 10.1016/j.jnca.2016.01.018.

L. Versluis and A. Iosup, “A survey of domains in workflow scheduling in computing infrastructures: Community and keyword analysis, emerging trends, and taxonomies,” Future Generation Computer Systems, vol. 123, pp. 156–177, 2021, doi: 10.1016/j.future.2021.04.009.

Y. Kumar, S. Kaul, and Y.-C. Hu, “Machine learning for energy-resource allocation, workflow scheduling and live migration in cloud computing: State-of-the-art survey,” Sustainable Computing: Informatics and Systems, vol. 36, Art. no. 100780, 2022, doi: 10.1016/j.suscom.2022.100780.

B. Kruekaew and W. Kimpan, “Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning,” IEEE Access, vol. 10, pp. 17803–17818, 2022, doi: 10.1109/ACCESS.2022.3149955.

H. Mahmoud, M. Thabet, M. H. Khafagy, and F. A. Omara, “Multiobjective task scheduling in cloud environment using decision tree algorithm,” IEEE Access, vol. 10, pp. 36140–36151, 2022, doi: 10.1109/ACCESS.2022.3163273.

N. K. Walia, N. Kaur, M. Alowaidi, K. S. Bhatia, S. Mishra, N. K. Sharma, S. K. Sharma, and H. Kaur, “An energy-efficient hybrid scheduling algorithm for task scheduling in the cloud computing environments,” IEEE Access, vol. 9, pp. 117325–117337, 2021, doi: 10.1109/ACCESS.2021.3105727.

Z. A. Khan, I. A. Aziz, N. A. B. Osman, and S. Nabi, “Parallel enhanced whale optimization algorithm for independent tasks scheduling on cloud computing,” IEEE Access, vol. 12, pp. 23529–23548, 2024, doi: 10.1109/ACCESS.2024.3364700.

I. M. Ali, K. M. Sallam, N. Moustafa, R. Chakraborty, M. Ryan, and K.-K. R. Choo, “An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems,” IEEE Transactions on Cloud Computing, vol. 10, no. 4, pp. 2294–2308, 2022, doi: 10.1109/TCC.2020.3032386.

E. Cao, S. Musa, M. Chen, T. Wei, X. Wei, X. Fu, and M. Qiu, “Energy and reliability-aware task scheduling for cost optimization of DVFS-enabled cloud workflows,” IEEE Transactions on Cloud Computing, vol. 11, no. 2, pp. 2127–2143, 2023, doi: 10.1109/TCC.2022.3188672.

O. L. Abraham, M. A. B. Ngadi, J. B. M. Sharif, and M. K. M. Sidik, “Task scheduling in cloud environment—Techniques, applications, and tools: A systematic literature review,” IEEE Access, vol. 12, pp. 138252–138279, 2024, doi: 10.1109/ACCESS.2024.3466529.

O. L. Abraham, M. A. B. Ngadi, J. B. M. Sharif, and M. K. M. Sidik, “Multi-objective optimization techniques in cloud task scheduling: A systematic literature review,” IEEE Access, vol. 13, pp. 12255–12291, 2025, doi: 10.1109/ACCESS.2025.3529839.