Proceedings of the International Conference on Industrial Engineering and Operations Management

Publisher: IEOM Society International, USA DOI: 10.46254/GC02.20240097
Published: December 01, 2024

Enhancing Cloud Architecture's Efficiency with an Adaptive Elitism-Based Genetic Algorithm for Task Scheduling

Muhammad Osama
Department of Computer Science
National University of Computer & Emerging Sciences
Karachi, Pakistan
k237800@nu.edu.pk

Syed Shah Sultan Mohiuddin Qadri
Department of Industrial Engineering
Çankaya University
Ankara, Türkiye
syedshahsultan@cankaya.edu.tr

Muhammad Farrukh Shahid
Department of Artificial Intelligence
National University of Computer & Emerging Sciences
Karachi, Pakistan
mfarrukh.shahid@nu.edu.pk

Ahmet Kabarcık
Department of Industrial Engineering
Çankaya University
Ankara, Türkiye
a.kabarcik@cankaya.edu.tr

Abstract

A contemporary business concept that has become popular is cloud computing (CC), which gives consumers unlimited access to virtual resources. This paradigm provides facilities for virtual machines (VMs) and data centers. Infrastructure as a service (IaaS) is one of the several service models. The secret to improving data center efficiency and lowering energy usage is the effective use and administration of the resources offered by these services. Because of the variety and computational complexity of this activity, effective task scheduling—which moves cloud jobs onto virtual machines (VMs)—is essential. To enhance this process, meta-heuristic techniques are frequently applied. This research offers a new adaptive elitism-based genetic algorithm (AGA-E), which integrates elitism with conditional parameter adjustment to increase convergence speed and solution quality. The conditional parameter tuning technique constantly modifies the algorithm's parameters in response to population diversity and fitness levels, whereas the elitist method maintains the best-performing solution over successive generations. The results of the proposed algorithm were compared with traditional approaches, including Min-Min, Max-Min, Adaptive Incremental GA (AIGA), and Standard GA (SGA). Experimental evaluations conducted on Amazon EC2 demonstrated that the proposed approach outperforms these existing methods regarding task completion time, resource utilization, and convergence performance. These results suggest that a method combining elitism and adaptive algorithms is effective in building a scalable and stable solution for work schedules in high-demand cloud environments.

Keywords
Task Scheduling, Cloud Computing, Adaptive Genetic Algorithm, Virtual Machines, and Optimization

1. Introduction
The domain of organizational computing conventionally relied on in-house infrastructures, which were composed of dedicated physical servers. These systems were standardized and designed to manage computational processes, making them generally sufficient for managing small-scale operations. However, their inherited expensive nature often led to the utilization of resources in a suboptimal manner, which resulted in the formation of data silos (Lepore et al.,2023; Johnny & Trevor, 2023), and considerable restrictions on the term of scalability (Ahmad et al., 2023). As the requirements for real-time responsiveness and the capability to handle complex computations escalated, the limitations of these traditional systems became glaringly apparent, rendering them increasingly inefficient and cumbersome to manage (Tran, 2013).

The emergence of cloud computing (CC) is considered a model framework shift in the consideration of computing resources being deployed and employed. By considering demand scalable, which can grow or shrink based on the demand and services provided by software over the internet (Maheswaran et al., 1999; Pirozmand et al., 2022). These cloud formats allow organizations to separate the hardware limitations, which results in the separation of assigning resources flexibly. Service models include Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (Attiya et al., 2020). These system models empowered several healthcare, finance, and educational institutions to achieve valuable performance that will result in reducing functional costs (Kumar & Sharma, 2018; Pencheva et al., 2009). These options cover a variety of sectors, which have evolved into private, public, and hybrid frameworks that offer excellence in flexibility towards usage alignment in the domain of organizational priorities.

However, the urge to make a smooth schedule of computational tasks across cloud resources is still considered a complicated and impossible challenge. In hybrid and diverse environments where processor capability, memory size, and network capacity are assigned tasks requiring intricate decision-making mechanisms (Wickremasinghe et al., 2010; Yuan et al., 2020). Furthermore, clients do submit their tasks in accordance with certain restrictions such as ongoing demands of CPU or memory, excessive data dependence, and high risk over execution deadlines, which must be ensured in respect of service level agreements (SLAs) and responsiveness of system competency.

Furthermore, as long as performance is valued, the one factor that can become a highlighted issue in modern cloud processing is energy in an efficient manner. As data centers contribute significantly to global electricity consumption along with carbon emissions, this results in the rise of green computing (Radu, 2017; Yuan et al., 2020). This whole domain is considered in formatting a computing system that can be both high in performance and environmentally responsible (Yuan et al., 2020). Throughout a lot of strategies considered in this domain, energy-aware task scheduling is considered to be highly impactful, as this can reduce the overconsumption of power without compromising the reliability and validity of the system.

Throughout the decades, a massive range of scheduling algorithms has been evaluated. Early developed heuristic methods, such as Min-Min, Max-Min, and Round-Robin, provide foundational and practical approaches but result in inefficient performance under diverse and large-scale conditions. Throughout the era, cloud workloads have become more advanced and complex, which has caused researchers to turn toward meta-heuristic algorithms. Which includes Genetic algorithm (GA), Ant Colony Optimization (ACO), and Grey Wolf Optimizer (GWO) (Kumar & Sharma, 2018; Pencheva et al., 2009; Sreenu & Sreelatha, 2019). These methods have led to notable improvements in the reduction of makespan and the conservation of energy. However, they often struggle with reliability and adaptability in the context of workloads, and an ideal way to utilize those resources is in over-categorized environments.

In the previous study (Osama et al., 2024), GA-based task scheduling model was proposed, which also addresses some challenges by enhancing its makespan along with energy usage. Furthermore, it was considered to have a restriction bound towards considering one task per VM. However, the allocations framework is actually limiting overall resource utilization and flexibility in scheduling. While the study does provide the foundation for energy scheduling, however, it did not overall exploit the diverse nature of a virtualized cloud environment (VCE).

1.1 Objectives

The current study evolved into an enhanced and more diverse model through the emergence of this foundation. The newly launched framework supports multi-task apportionment per VM. This enables each VM to perform several tasks simultaneously based on on-time availability, which will improve the system in terms of reducing makespan and enhancing load balancing. Additionally, this framework provides a GA based system to induce high-quality task schedules throughout the generations. However, customized crossover and mutation operators are employed in order to improve diversity and adaptability in terms of solutions. Through critical exposure, these models emerge energy-aware heuristics to incorporate with green computing objectives, while also incorporating system resilience and error tolerance in the domain of excessive workloads.

In summary, the research emerges as an intelligible, scalable system and environmentally active approach in the task scheduling sector in CC. By expanding on our earlier work and integrating multitask VM allocation with advanced GA techniques, the proposed framework contributes meaningfully to the development of next-generation, sustainable cloud infrastructure.

2. Literature Review

As CC matured, the task of scheduling workloads across virtualized resources evolved from simple rules to sophisticated optimization techniques. In the early stages, researchers relied on heuristic-based algorithms like Min-Min, Max-Min, and Round-Robin to assign tasks based on estimated execution times. While these methods provided a foundation, they proved inadequate in real-world, dynamic cloud environments. Their deterministic nature lacked adaptability, especially when dealing with heterogeneous resources and fluctuating workloads.

To address the shortcomings of static heuristics, more adaptive methods such as Load Balanced Min-Min (LBMM) and Traveling Salesman Cloudlet Scheduling (TSACS) were introduced. These refinements aimed to redistribute tasks more evenly and optimize task routing. However, they still struggled with scalability, especially as the number of users and tasks increased (Chen et al., 2013; Nasr et al., 2019).

As CC evolved, the task scheduling workloads across software resources upgraded from common rules to more complex and advanced optimization techniques. In the early decades, researchers depended entirely on traditional heuristic-based algorithms such as Min-Min, Max-Min, and Round-Robin in order to assign tasks in accordance with execution times. However, these methods incorporate the foundation but result in a lack of a diverse or real-world cloud environment. The nature of these terms lacks adaptability, especially when emerging with diverse workloads and heterogeneous consideration of resources.

In accordance with the traditional domain of static heuristics, more adaptive methods have been introduced, such as load-balanced Min-Min (LBMM) and Traveling Salesman Cloudlet Scheduling (TSACS), which emerged in order to distribute the task evenly. However, they still struggled with scalability, especially when the task and users both increased in size.(Chen et al., 2013; Nasr et al., 2019).

This led to the emergence of meta-heuristic algorithms, which not only evolved intelligibility and flexibility but also provided approaches for solving complex and advanced problems. Among these, different variants of GA stood out for their ability to explore large solution spaces and adapt to changing conditions. (Liu & Wang, 2020) demonstrated that GAs could dynamically reduce makespan and improve load balance through custom fitness operators. This marked a shift toward more responsive and efficient scheduling solutions.

Now, performance optimization alone couldn't be sufficient, which makes researchers move towards multi-objective scheduling to bring energy efficiency, cost, and the utilization of several resources into account. In this regard, (Kumar Shukla et al., 2021), introduced an energy-aware GA that employed Dynamic voltage and frequency scaling (DVFS), which reduces power consumption without compromising performance. (Radu, 2017) demonstrated that efficient energy relates not only to the economic domain but also significantly impacts the environmental aspect, paving the way for green computing, which should be the primary consideration towards scheduling models.

Furthermore, the hybrid metaheuristic is evaluated in order to enhance the quality of solutions and speed. (Peng et al., 2022) combined GA with the MapReduce framework for showing support towards large-scale DAG with prioritized constraints. (Abualigah & Alkhrabsheh, 2022) proposed a Multi-Verse Optimizer integrated with GA (MVO-GA), which intelligibly balanced exploration and exploitation for high-load cloud domains. Similarly, (Yuan et al., 2020) evaluate a Greedy Genetic Hybrid (MGGS), a scheduler that will improve convergence while maintaining accuracy. Despite these advancements, one continuous restriction was quite evident in many frameworks, which was the limitation of assigning one task per VM only. This limitation will eventually lead to the inefficient usage of resources and make the makespan longer in the context of high-demand scenarios. However recent emergence of efforts ought to overcome this issue by indulging multi-tasking per VM. For example, (Iranmanesh & Naji, 2021) introduced DCHG-TS, a hybrid GA-based method that allowed task grouping while respecting deadlines and budget constraints. Similarly, (Imene et al., 2022) employed an NSGA III model to optimize execution time and energy consumption simultaneously.

More recent studies have started to incorporate error tolerance and workload awareness into scheduling strategies. (Naz et al., 2023) emerged a workload-aware GA with the usage of asymmetric traces in accordance with diverse conditions. (Rizvi et al., 2023) developed a fuzzy adaptive GA to handle unpredictable situations, while (Xie et al., 2023) introduced a multi-population GA, which provides convergence with balancing workloads intelligibly over distributed cloud nodes.

While these contributions are significant, several research gaps persist. Notably, many frameworks still lack integrated elitism strategies, have limited support for multitasking, and do not fully incorporate energy-aware heuristics in dynamic and large-scale cloud environments. These gaps leave room for further improvement in adaptability, efficiency, and environmental responsibility.

Addressing these limitations, the current study builds upon our previous work (Osama et al., 2024), which implemented a GA-based scheduling framework with energy-aware capabilities but was restricted to single-task-per-VM allocation. The present research extends that foundation by introducing a multi-task-per-VM strategy, paired with elitism-based evolution, custom genetic operators, and dynamic parameter adaptation. The proposed framework offers a comprehensive solution for intelligent and sustainable cloud task scheduling by prioritizing energy efficiency, fault tolerance, and scheduling robustness.

Four comparative algorithms have been selected to rigorously evaluate the proposed model: Min-Min, Max-Min, Adaptive Improved Genetic Algorithm (AIGA), and Standard Genetic Algorithm (SGA). Min-Min and Max-Min, being traditional heuristics, are widely used benchmarks known for their effectiveness in handling tasks of varying execution times (Duan et al., 2018; Liu & Wang, 2020). AIGA and SGA, both GA variants, have been extensively implemented in metaheuristic research and have shown promising results in resource optimization (Duan et al., 2018). Their inclusion in the comparative analysis ensures a comprehensive and meaningful assessment of the adaptive advantages introduced by the proposed methodology.

3. Proposed Methodology

Task scheduling is difficult in CC because different tasks have high processing needs that need to be met. Task distribution and scheduling are necessary for *VMs* to maintain quality of service (QoS). Makespan and load balance are crucial characteristics used in the scheduling process to improve performance and optimum resource efficiency. The creation of the Adaptive Elitism-based Genetic Algorithm (AGA-E) and its component elements are covered in this section.

3.1 Task Scheduling Problem

A data center in cloud architecture comprises various physical servers running *VMs*. Depending on the needs of each VM, user tasks are delegated to different ones. Scheduling has two fundamental problems: First, on which host to deploy the VM. Cloud data center performance, resource utilization, and energy consumption depend on VM scheduling techniques. Certain scheduling algorithms, such as "first-come, first-served" or greedy algorithms, can enable more physical servers, significantly when the needs of VMs vary greatly. Another issue is which task to assign to which VM. In the cloud, large tasks are divided into smaller tasks using the Map/Reduce technique and then assigned to appropriate cloud servers. Tasks are allocated according to each VMs's current status and service requirements. A task's completion time and cost are affected by the scheduling strategy. Using appropriate scheduling methods is essential for better performance and lower costs for cloud users and providers.

These services are responsible for scheduling VMs and tasks according to user requirements and criteria. Service providers seek to maximize efficiency and minimize costs and energy consumption, while consumers are primarily concerned with the cost of services. The objective of this study is to minimize the makespan, which can be defined as the duration of time required to complete all jobs is known as the makespan. In other words, the time between the start and finish of a sequence of tasks is known as the makespan. The other factors are also important, but they are somewhat related to task execution time (makespan). We model the task scheduling problem as the following:

Suppose we have n VMs $(vm_1, vm_2, \ldots, vm_n)$ and m tasks $(t_1, t_2, \ldots, t_m)$ Assigning these tasks to VMs is an NP-complete problem with n^m possible solutions. VMs can be defined based on CPU, memory bandwidth, and storage. A compute unit (CU) is a measure that some cloud service providers, such as Amazon, use to describe CPU capacity. A CPU capacity equivalent to a 1.0–1.2 GHz 2007 Opteron or 2007 Xeon processor is called an EC2 Compute Unit (ECU) (Wen et al., 2015). The required execution time can be predicted using CU before requesting cloud resources, as cloud services are charged based on the hours used.

The length of the task and the CU of the VM vm_j determine the execution time of a task t_i when it is attached to a vm_j. Its formula is:

$$t_{ij} = \frac{L_i}{C_j} \tag{1}$$

where $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$. L_i is the length of the task, and C_j is the CU of the virtual machine. The total occupancy time for a vm_j can be calculated as:

$$T_j = \sum_{i=1}^{m} t_{ij} \cdot x_{ij} \tag{2}$$

where $x_{ij} \in \{0, 1\}$ represents a mapping between a t_i and a vm_j. When $x_{ij} = 1$, task t_i is allocated to vm_j. Because the tasks are executed in parallel on the VMs, the overall execution time is the largest of all VM occupation times. To minimize the makespan, the objective function can be expressed as:

$$min(max(T_j)) = min\left(max\left(\sum_{i=1}^{m} t_{ij} \cdot x_{ij}\right)\right) \tag{3}$$

where x_{ij} represents the allocation state and t_{ij} represents the execution time.

3.2 Integrating Meta-Heuristic in the Cloud for Task Scheduling

Figure 1 shows an example of a CC design incorporating a meta-heuristic-based scheduler to optimize job scheduling. Through the Internet, users make requests, which are processed by the metaheuristic-based scheduler to create an effective scheduling map. A Cloud Broker analyzes this map and allocates jobs to the right VMs inside data centers. Using metaheuristic algorithms to generate optimal or nearly optimal scheduling solutions, the scheduler maximizes task distribution across cloud resources, improving operational efficiency and resource usage. This system exemplifies how sophisticated scheduling algorithms can control the scalable and dynamic demands of CC environments, enhancing performance and cutting expenses.

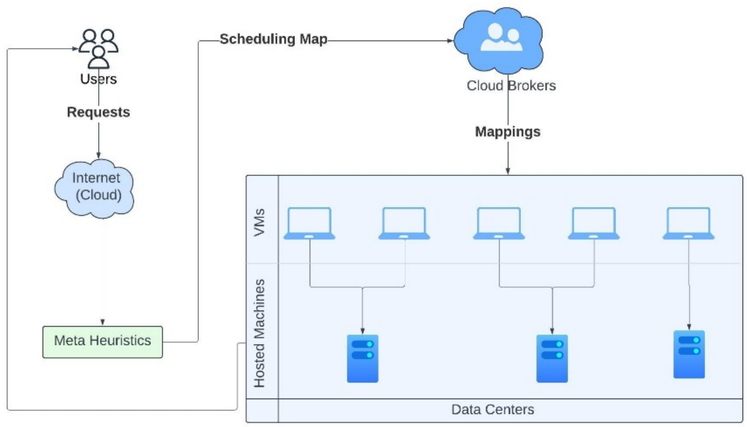


Figure 1. Abstract Representation of the Cloud Environment

3.3 Adaptive Elitism-Based Genetic Algorithm

GA is an inspired algorithm based on Darwin's Theory of Evolution and Mendelian Genetics. In GA, Selection, crossover, and mutation are the primary operators. The Selection operator in GA mimics the process of natural selection. The goal of selection is to preserve a "good" individual. The greater the fitness rating, the greater the likelihood that the individual will be able to procreate. Conversely, those less valuable in fitness will be retained less frequently. Selection techniques influence the population's pace of convergence and direction of evolution. Gene recombination is the process of crossover. The perfect characteristics of both parents should ideally be inherited by the children created by crossing two parents. A mutation operation modifies a few genes to create a new individual. Finding the best possible answer and adding to population diversity is crucial.

Given the varying complexities of a cloud environment, a data center arranges for multiple tasks to be scheduled in its request list. In SGA processes, where evaluation, crossover, and mutation operations involve computationally heavy steps, finding the optimal solution can take a long time. In this paper, AGA-E is proposed to solve this problem. This suggests that the number of VMs will be adapted to the duration of tasks and that elitism techniques will ensure that the most successful individual or solution in one generation is passed on to the next generation without change. Figure 2 describes the complete process of AGA-Elitism, while the details of the proposed AGA-E are provided in subsequent figures.

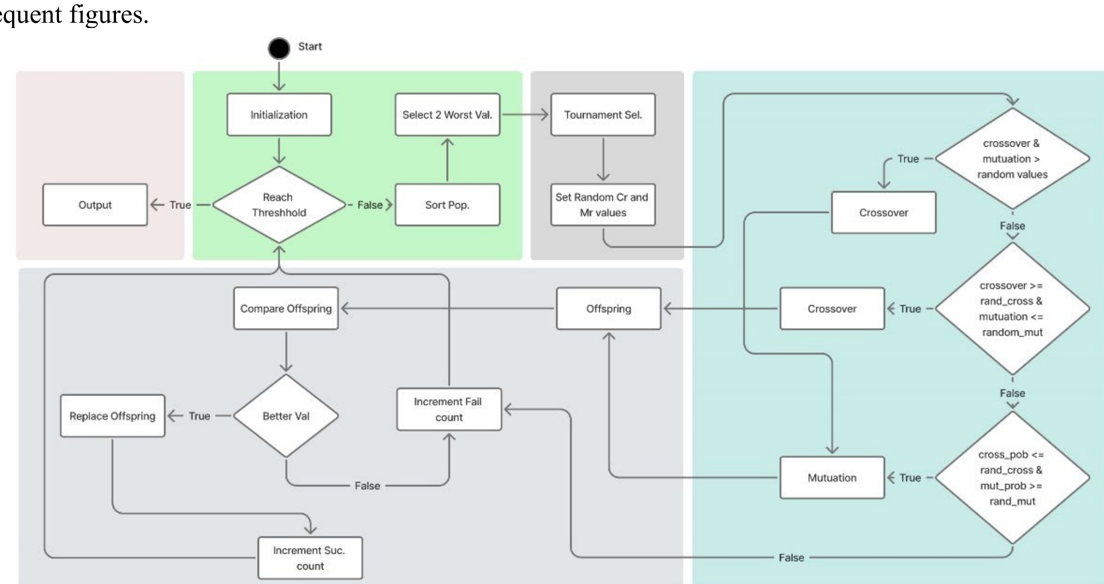


Figure 2. Flow Diagram of AGA-E

3.4 Encoding
In the task scheduling problem, chromosomes that represent a feasible solution are encoded with real numbers, where the length of the chromosome is equal to the number of tasks. The value of the gene determines which task will be assigned to which VM.

3.5 Fitness Function
Selecting the best candidate is a crucial matter, and for this purpose, the fitness function is used, which helps to evaluate the quality of candidate solutions. Generally, a fitness function can be defined for a given situation, or the value of an objective function can be used to measure fitness directly. Equation (3) provides a formula for calculating fitness, which represents the efficiency of a solution. The fitness value represents the time until all *VMs* have completed their longest task.

3.6 Selection
Selection plays a vital role as it populates the suitable individuals. There are different selection strategies like sorting, roulette, and tournaments. For this approach, we have used tournament-based selection in which a subset of people is selected randomly, and the most fit individual is chosen to procreate. Until the desired number of parents is selected, it repeats. It assists in striking a balance between diversity and selection pressure, allowing stronger candidates to win while preserving population diversity.

3.7 Crossover
The crossover operator plays an important role in generating new offspring. The literature contains several methods for this procedure, including homogeneous crossover, two-point crossover, and single-point crossover. In this study, a two-point crossover was employed, the detail of which is illustrated in Figure 3. Every paternal chromosome represents one of the solutions for task scheduling problems, in which every gene (or cell) corresponds to a particular task (Task 0 to Rask 9). The numbers in the cells designate the exact $\boldsymbol{VM}$ ($\boldsymbol{VM}$ 0 to $\boldsymbol{VM}$ 9) to which a given task is allocated. Two randomly selected vertical green lines separate the crossover regions. These lines represent locations where the genetic material from the parents will be swapped by randomly selecting the part to create new offspring.

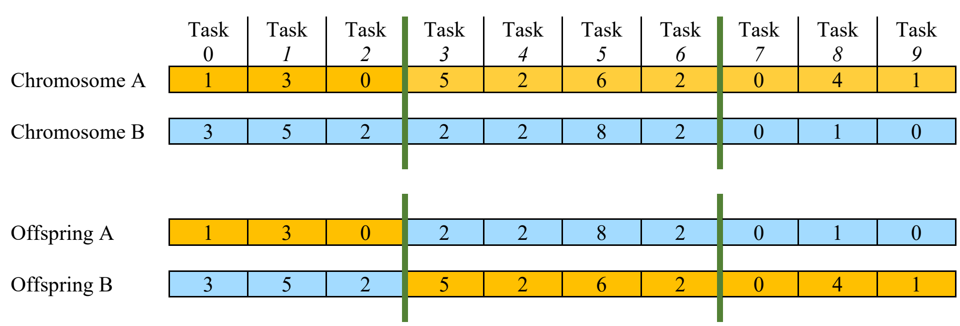


Figure 3. Crossover Illustration

3.8 Mutation
Mutation operators play an important role when it comes to diversity; they help alter genes, eliminate premature convergence, and help find new solution spaces. We have different Mutation Strategies, such as uniform mutation, Gaussian mutation, etc. A throas mutation operator (Otman et al., 2012a)was utilized for this study. Figure [number] is an example of Throas Mutation.

3.9 Parameter Tuning
A study was conducted using populations of 75, 100, and 125 to determine the best parameter setting for the proposed algorithm to enhance its effectiveness. The study included a methodical examination of both crossover and mutation rates, focusing on the hypothesis that the higher crossover rate value than the mutation rate value gives better results. In an AGA-E setup, Figure 4 below shows heatmaps illustrating the fitness values associated with various combinations of crossover and mutation rates across various population sizes. The fitness results for 75, 100, and 125

populations are shown in each heatmap correspondingly. These fitness score-annotated visualizations offer a clear picture of how the settings of the genetic algorithm affect how successful the created solutions are.

Figure 4. Heatmap Illustrating the Fitness Values Associated with Population Size

4. Experimental Results

The specifics of the experimental study have been detailed in this section. The suggested AGA-E was tested against two conventional scheduling techniques, Min-Min and Max-Min, as well as the meta-heuristic algorithms SGA, and AIGA, which are described in (Duan et al., 2018). The optimal parameter values that were chosen for this study in order to minimize the makespan while using the fewest number of VMs are shown in Table 1 below.

Table 1. Optimal Parameter Settings for AGA-E

Parameters	**Values**
Population Size	100
Crossover	0.8
Mutation	0.2
Success Threshold	350
Failure Threshold	10
Tournament Size	5

In order to accomplish the shortest unassigned tasks on the system in the least period of time, the Min-Min algorithm gives them priority when scheduling them. After identifying a group of unallocated tasks, it determines the Minimum Completion Time (MCT) for every task across all available machines. The task with the minimum MCT is assigned to the respective machine, and the machine's available time is adjusted. This process continues until all tasks are scheduled. The Max-Min algorithm operates in a similar manner but focuses on scheduling the largest task first on the machine with the minimum MCT.

Min-Min and Max-Min are known for their simplicity and efficiency in scheduling tasks based on completion time; however, they lack adaptability in dynamic environments. Min-Min prioritizes shorter tasks, which can lead to longer tasks waiting in the queue. In contrast, Max-Min focuses on allocating the longest tasks first, potentially increasing wait times for shorter tasks (Peng et al., 2022). In contrast, SGA and AIGA leverage genetic operations to achieve near-optimal solutions over successive generations. However, these algorithms can suffer from slower convergence (Asghari et al., 2021; Peng et al., 2022). The proposed AGA-E, by incorporating adaptive parameter tuning and elitism, addresses these limitations by improving convergence speed and ensuring task distribution consistency across *VMs* Regarding the specifications and length of tasks, the authors suggest referring to the Amazon Elastic Compute Cloud scheme used by Duan et al. (2018) for more details. Table 2 provides information on the instance types of VMs and their corresponding Compute Units (CU).

Table 2. VM Types and their CU

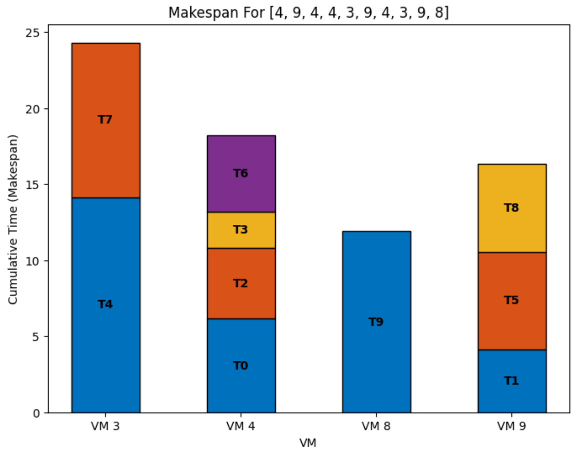
VM No.	**Instance Type**	**CU**
1	c3: large	8
2	c3: xlarge	16
3	c3: 2xlarge	31
4	c3: 3xlarge	62
5	c3: 4xlarge	132
6	c4: xlarge	7
7	c4: 2xlarge	14
8	c4: 3xlarge	28
9	c4: 4xlarge	55
10	c4: 8xlarge	108

For this study, the task lengths have been used ranging from 10 to 1000 seconds that are [814.0, 447.0, 610.0, 319.0, 876.0, 692.0, 663.0, 631.0, 626.0, 655.0]. The execution time consumed by each VM for task completion for the given tasks is calculated using (1), and values are represented in Table 3.

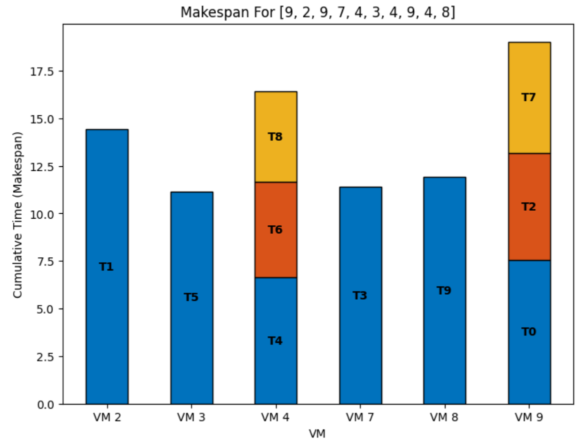
Table 3. Execution Time of Each VM

		Virtual Machines (VM_i)									
		VM0	**VM1**	**VM2**	**VM3**	**VM4**	**VM5**	**VM6**	**VM7**	**VM8**	**VM9**
Tasks	**0**	101.75	50.88	26.26	13.13	6.17	116.29	58.14	29.07	14.8	7.54
	1	55.88	27.94	14.42	7.21	3.39	63.86	31.93	15.96	8.13	4.14
	2	76.25	38.13	19.68	9.84	4.62	87.14	43.57	21.79	11.09	5.65
	3	39.88	19.94	10.29	5.15	2.42	45.57	22.79	11.39	5.8	2.95
	4	109.5	54.75	28.26	14.13	6.64	125.14	62.57	31.29	15.93	8.11
	5	86.5	43.25	22.32	11.16	5.24	98.86	49.43	24.71	12.58	6.41
	6	82.88	41.44	21.39	10.69	5.02	94.71	47.36	23.68	12.05	6.14
	7	78.88	39.44	20.35	10.18	4.78	90.14	45.07	22.54	11.47	5.84
	8	78.25	39.13	20.19	10.1	4.74	98.43	44.71	22.36	11.38	5.8
	9	81.88	40.94	21.13	10.56	4.96	93.57	46.79	23.39	11.91	6.06

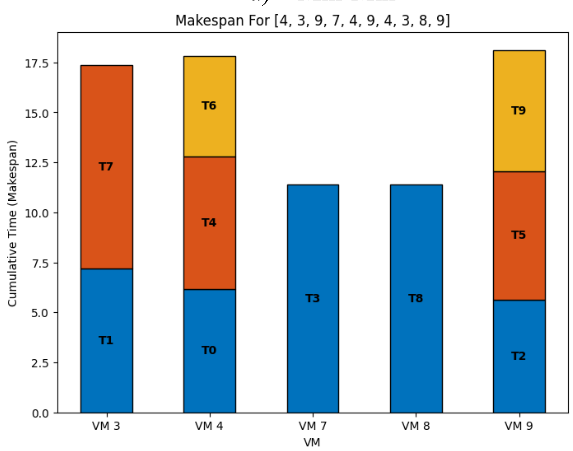
By the time the algorithm concluded, the following task orders were produced: [3, 1, 2, 8, 7, 9, 6, 5, 0, 4], [4, 0, 5, 6, 9, 7, 8, 2, 1, 3], [0, 2, 1, 4, 5, 8, 3, 6, 9, 7], and [1, 8, 9, 6, 4, 3, 2, 7, 5, 0]. These task orders were produced by the Min-Min, Min-Max, AIGA, and the proposed AGA-E algorithms. In that order, these algorithms took 290.92, 366.077, 343.74, and 265.25 seconds to complete. For the Makespan, which represents the maximum execution time after all the tasks-to-VMs scheduling, the results are also compared with SGA as well. The makespan with Min-Min, Min-Max, AIGA, SGA, and AGA-E are 24.31, 19.5, 18.12, 21.25 and 17.29 seconds for these algorithms, respectively. These assignments and makespan are depicted in Figure 5 as follows: a) Min-Min, b) Max-Min, c) AIGA, d) SGA, and e) AGA-E.



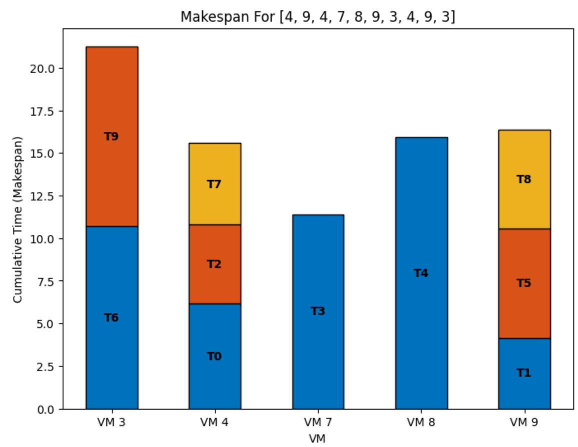
a) Min-Min



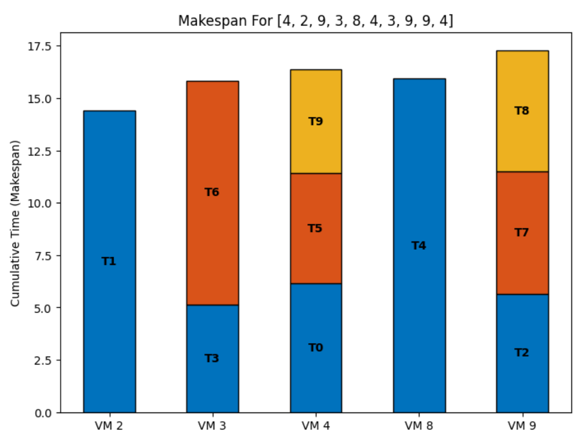
b) Max-Min



c) AIGA



d) SGA



e) AGA-E

Figure 5. Tasks-to-VMs Assignment and Makespan by Algorithms

To statistically validate the performance results, we conducted significance testing to determine the confidence level of the AGA-E algorithm's performance compared to other approaches. We employed a t-test to compare the mean makespan and resource utilization values of AGA-E against each of the other algorithms, observing a p-value of less than 0.05, indicating a statistically significant improvement (Liu & Wang, 2020; Kumar Shukla et al., 2021).

Additionally, confidence intervals were calculated for each algorithm's performance metrics, allowing us to quantify the error margins associated with each approach. This statistical analysis adds rigor to the experimental findings, reinforcing the reliability of AGA-E's superior task scheduling performance.

Figure 6 shows a violin plot that compares the distribution of mean total times for AGA-E and SGA using scattered data points. The data density and dispersion are more clearly visualized in the plot. The effectiveness of the AGA-E is highlighted by its narrow form, which indicates a more compact distribution and concentrated data points around lower mean total times. The SGA group's figure, in comparison, is thicker and broader at higher time values, indicating larger total times and greater dispersion. Each violin's dispersed points represent a single data point, revealing information about the distribution and outliers within each group. The visualizations underscore the comparative analysis of task processing efficiencies between both strategies, clearly favoring the 'Elitism' approach for its consistency and effectiveness in reducing processing times.

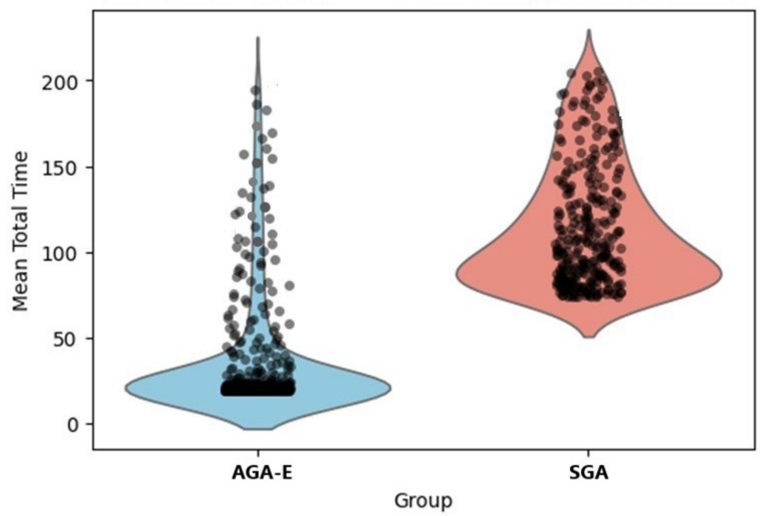


Figure 6. Violin Plots with Scattered Points- AGA-E vs SGA

To compare the job processing efficiency of AGA-E and SGA further, figure 7 shows a grouped box plot of mean total time. The median, interquartile range, and any outliers for each method are highlighted in the box plot, showing the total processing times distribution. In order to clearly show the average total time for each algorithm, mean markers were also included. This graphic representation enhances the violin plot by offering more information about the distribution and central tendency of the processing times. Plotting shows that AGA-E is more efficient and consistent in task processing than SGA, with a lower mean total time and a more compact distribution.

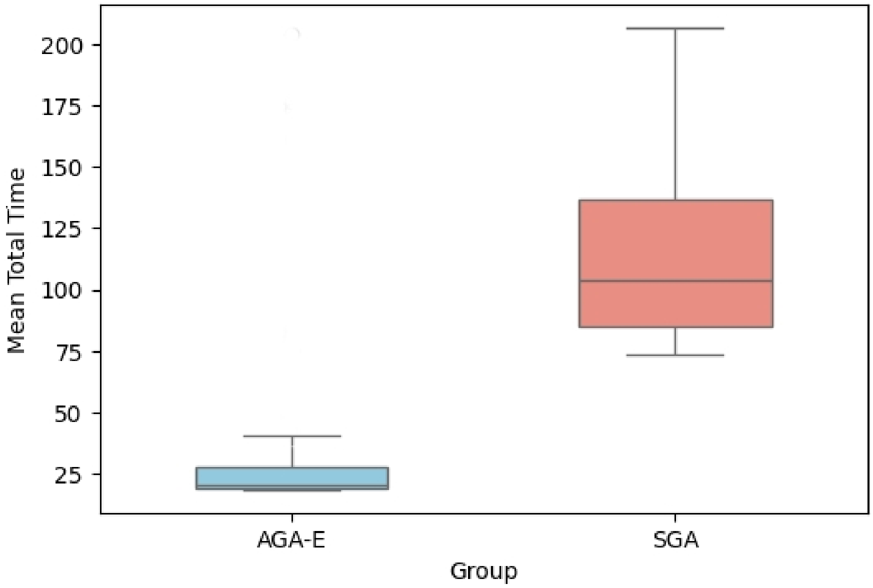


Figure 7. Box Plot of Mean Total Time for Task Processing Efficiency – AGA-E vs. SGA

5. Conclusion

During task scheduling in a cloud environment, time consumption is directly related to financial cost to users and energy consumption. Therefore, proper resource allocation is imperative to reduce the makespan in processing task batches. GA and other biologically motivated techniques, which effectively solve NP-complete problems, often suffer from slow convergence when applied to high-dimensional data. To face these difficulties, an AGA-E technique is proposed for large-scale task scheduling in cloud environments, reducing the makespan and computational time. The method adopts a conditional approach to selecting and tuning parameters, which allows efficient and optimal allocation to multiple tasks with low computational overhead. The task scheduling problem is framed as an objective optimization challenge, in which the number of tasks defines the complexity of the problem. Experiments show that AGA-E outperforms traditional scheduling methods such as Min-Min, Max-Min, AIGA, and SGA in sophisticated large task volumes and provides more efficient solutions in less computational time.

Future research will seek to develop an advanced mathematical model to understand the complexities of task scheduling on IaaS platforms, incorporating task dependencies and dynamic variations. In addition, a detailed study of AGA-E will examine other objectives, such as energy consumption and resource utilization, as well as multi-objective optimization scenarios.

References

Abualigah, L., & Alkhrabsheh, M., Amended hybrid multi-verse optimizer with genetic algorithm for solving task scheduling problem in cloud computing. *Journal of Supercomputing*, *78*(1), 740–765, 2022. https://doi.org/10.1007/s11227-021-03915-0

Ahmad, R., Rafiq, M., & Arif, M., Global trends in digital preservation: Outsourcing versus in-house practices. *Journal of Librarianship and Information Science*, *56*(4), 1114-1125, 2024.

Asghari, A., Sohrabi, M. K., & Yaghmaee, F., Task scheduling, resource provisioning, and load balancing on scientific workflows using parallel SARSA reinforcement learning agents and genetic algorithm. *Journal of Supercomputing*, *77*(3), 2800–2828, 2021. https://doi.org/10.1007/s11227-020-03364-1

Attiya, I., Abd Elaziz, M., & Xiong, S., Job Scheduling in Cloud Computing Using a Modified Harris Hawks Optimization and Simulated Annealing Algorithm. *Computational Intelligence and Neuroscience*, *2020*. https://doi.org/10.1155/2020/3504642

Chen, H., Wang, F., Helian, N., & Akanmu, G., User-priority guided min-min scheduling algorithm for load balancing in cloud computing. *2013 National Conference on Parallel Computing Technologies, PARCOMPTECH 2013*. https://doi.org/10.1109/ParCompTech.2013.6621389

Duan, K., Fong, S., Siu, S. W. I., Song, W., & Guan, S. S. U., Adaptive incremental Genetic Algorithm for task scheduling in cloud environments. *Symmetry*, *10*5), 2018. https://doi.org/10.3390/sym10050168

Imene, L., Sihem, S., Okba, K., & Mohamed, B., A third generation genetic algorithm NSGAIII for task scheduling in cloud computing. *Journal of King Saud University - Computer and Information Sciences, 2022*. https://doi.org/10.1016/j.jksuci.2022.03.017

Iranmanesh, A., & Naji, H. R., DCHG-TS: a deadline-constrained and cost-effective hybrid genetic algorithm for scientific workflow scheduling in cloud computing. *Cluster Computing*, *24*(2), 667–681, 2021. https://doi.org/10.1007/s10586-020-03145-8

Johnny, R., & Trevor, J., "Addressing Data Silos with Governance Frameworks in Financial Organizations," ResearchGate, Sep. 2023. [Online]. Available: https://www.researchgate.net/publication/387174207

Kumar, M., & Sharma, S. C., Deadline constrained based dynamic load balancing algorithm with elasticity in cloud environment. *Computers and Electrical Engineering*, *69*, 395–411, 2018. https://doi.org/10.1016/j.compeleceng.2017.11.018

Kumar Shukla, D., Ali, S., & Trivedi, M. C., Energy Aware Scheduling of Tasks in Cloud environment. In *Turkish Online Journal of Qualitative Inquiry (TOJQI)* (Vol. 12, Issue 6) , 2021.

Lepore, D., Montgomery, A., Siepe, G., & Siepe, F., Shifting Away from Silos. In *From Silos to Network: A New Kind of Science for Management: A New Kind of Science for Managing Complexity* (pp. 1-24). Cham: Springer International Publishing, 2023.

Liu, S., & Wang, N., Collaborative Optimization Scheduling of Cloud Service Resources Based on Improved Genetic Algorithm. *IEEE Access*, *8*, 150878–150890, 2020. https://doi.org/10.1109/ACCESS.2020.3016762

Maheswaran, M., Ali, S., Siegel, H. J., Hensgen, D., & Freund, R. F., Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems. *Journal of Parallel and Distributed Computing*, *59*, 107–131, 1999. https://doi.org/10.1006/jpdc.1999.1581

Nasr, A. A., El-Bahnasawy, N. A., Attiya, G., & El-Sayed, A., Using the TSP Solution Strategy for Cloudlet Scheduling in Cloud Computing. *Journal of Network and Systems Management*, *27*, 366–387, 2019. https://doi.org/10.1007/s10922-018-9469-9

Naz, I., Naaz, S., Agarwal, P., Alankar, B., Siddiqui, F., & Ali, J., A Genetic Algorithm-Based Virtual Machine Allocation Policy for Load Balancing Using Actual Asymmetric Workload Traces. *Symmetry*, *15*(5) , 2023. https://doi.org/10.3390/sym15051025

Osama, M., Sultan Mohiuddin Qadri, S. S., Shams Malick, R. A., Shahid, M. F., & Dawood, K., Enhanced Task Scheduling in IaaS Cloud Environments Using Elitism-Based Genetic Algorithms. *Proceedings - ELTICOM 2024: 8th International Conference on Electrical, Telecommunication and Computer Engineering: Tech-Driven Innovations for Global Organizational Resilience*, 62–67, 2024. https://doi.org/10.1109/ELTICOM64085.2024.10865043

Otman, A., Abouchabaka, J., Tajani, C., & Abdoun, O., Analyzing the Performance of Mutation Operators to Solve the Travelling Salesman Problem Analyzing the Performance of Mutation Operators to Solve the Traveling Salesman Problem. *Int. J. Emerg. Sci*, *2*(1), 61–77, 2012. https://www.researchgate.net/publication/221700559

Pencheva, T., Atanassov, K., & Shannon, A., Modelling of a Roulette Wheel Selection Operator in Genetic Algorithms Using Generalized Nets. *International Journal Bioautomation*, *13, 2009*.

Peng, Z., Pirozmand, P., Motevalli, M., & Esmaeili, A., Genetic Algorithm-Based Task Scheduling in Cloud Computing Using MapReduce Framework. *Mathematical Problems in Engineering*, *2022*. https://doi.org/10.1155/2022/4290382

Pirozmand, P., Javadpour, A., Nazarian, H., Pinto, P., Mirkamali, S., & Ja'fari, F., GSAGA: A hybrid algorithm for task scheduling in cloud infrastructure. *Journal of Supercomputing*, *78*, 17423–17449, 2022. https://doi.org/10.1007/s11227-022-04539-8

Radu, L. D., Green cloud computing: A literature survey. In *Symmetry* (Vol. 9, Issue 12). MDPI AG, 2017. https://doi.org/10.3390/sym9120295

Rizvi, N., Ramesh, D., Wang, L., & Basava, A., A Workflow Scheduling Approach With Modified Fuzzy Adaptive Genetic Algorithm in IaaS Clouds. *IEEE Transactions on Services Computing*, *16*(2), 872–885, 2023. https://doi.org/10.1109/TSC.2022.3174112

Sreenu, K., & Sreelatha, M., W-Scheduler: whale optimization for task scheduling in cloud computing. *Cluster Computing*, *22*, 1087–1098, 2019. https://doi.org/10.1007/s10586-017-1055-5

Viet-Trung Tran. Scalable data-management systems for Big Data. Other [cs.OH]. École normale supérieure de Cachan- ENS Cachan, 2013. English. NNT: 2013DENS0001

Wickremasinghe, B., Calheiros, R. N., & Buyya, R., CloudAnalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, 446–452, 2020. https://doi.org/10.1109/AINA.2010.32

Xie, Y., Gui, F. X., Wang, W. J., & Chien, C. F., A Two-stage Multi-population Genetic Algorithm with Heuristics for Workflow Scheduling in Heterogeneous Distributed Computing Environments. *IEEE Transactions on Cloud Computing*, *11*(2), 1446–1460, 2023. https://doi.org/10.1109/TCC.2021.3137881

Yuan, H., Bi, J., & Zhou, M. C., Profit-Sensitive Spatial Scheduling of Multi-Application Tasks in Distributed Green Clouds. *IEEE Transactions on Automation Science and Engineering*, *17*, 1097–1106, 2020. https://doi.org/10.1109/TASE.2019.2909866

Biographies

Muhammad Osama is a Master's student at the National University of Computer and Emerging Sciences (FAST-NUCES), currently pursuing advanced studies in computer science. He completed his Bachelor's degree in Computer Science from Sindh Madressatul Islam University in 2022. His research interests span across meta-heuristics, natural language processing (NLP), machine learning, Web3 technologies, and blockchain. Muhammad Osama is passionate about exploring innovative solutions and applying cutting-edge technologies to tackle complex problems in these domains.

Syed Shah Sultan Mohiuddin Qadri is an Assistant Professor in the Department of Industrial Engineering at Çankaya University, Ankara, Türkiye. He holds a Ph.D. in Industrial Engineering from Yasar University, Izmir, Türkiye. Dr. Qadri earned his BS and MS degrees in Applied Mathematics from the University of Karachi and NED University of Engineering & Technology, Karachi, Pakistan, in 2010 and 2013, respectively. He has worked as a researcher on a TÜBİTAK-funded project, and his research interests include simulation optimization, intelligent transportation systems, heuristic optimization, and traffic modeling. Dr. Qadri is dedicated to advancing optimization techniques for solving complex real-world problems.

Muhammad Farrukh Shahid received his Ph.D. in a Joint Doctorate in Interactive and Cognitive Environments under a joint degree program between Queen Mary University of London, UK, and the University of Genoa, Italy in 2021. During his Ph.D. research study, he worked on developing a framework based on deep learning techniques and Bayesian probabilistic models to mitigate the effects of abnormal attacks in the AI-enabled IoT network and published his research work in IEEE Transactions and top-ranked signal processing conferences. He wrote two book chapters on the topics of realizing cognitive radios for wireless sensor networks and IoT and deep learning integration for industry 4.0. Currently, he is serving as an Assistant Professor in the Computer Science Department at FAST-NUCES Karachi, where he teaches courses in Artificial Intelligence, Machine Learning, and Statistical and Mathematical Methods for Data Science. Recently, he supervised a final-year undergraduate project titled "Virtual Apparel Try Room", which won two awards in the National competitions held in Karachi and was also selected by the National Incubation Centre Karachi for incubation.

Ahmet Kabarcık is a Lecturer at the Department of Industrial Engineering at Çankaya University in Ankara, Turkey. He earned his PhD in Operations Research from the Turkish Military Academy in 2013. He completed his Bachelor's and Master's degrees in Computer Engineering at Çankaya University in 2002 and 2007, respectively. His research interests include network optimization and security, optimal database design and management, enterprise resource planning, object-oriented programming, and computer-aided design and manufacturing. Recently, he has developed a strong interest in artificial intelligence and machine learning and is actively working to enhance his knowledge in these fields.