Grenze International Journal of Engineering and Technology, June Issue

Multi-Objective Genetic Algorithm for Software Project Scheduling

Kuldeep Vaydande[1], Parth Nakti[2], Pranav Nagur[3], Viraj Pabale[4], Rohan Murudkar[5] and Malik Mujawar[6]
[1-6]Vishwakarma Institute of Technology, Pune, India
Email: kuldeep.vayadande@gmail.com

***Abstract*— Software project scheduling is an important aspect which influences the duration, cost, and risks involved in a software project. The conventional scheduling approaches like Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT) consider mainly the time factor and do not help in optimizing the multi-objective problem. In this paper, a scheduling algorithm based on Multi-Objective Genetic Algorithm (MOGA) is designed for managing the software project schedule. The proposed system takes into account the dependencies between different software tasks, their durations, cost, resources, and risks involved. The genetic algorithm operations like selection, crossover, and mutation are performed to provide an optimized schedule by considering dependency and resource constraints. The proposed system optimizes the project schedule by minimizing its duration, cost, and risks. The performance of the proposed system is compared with the conventional methods CPM and PERT. It has been found from experimental analysis that the proposed MOGA-based scheduling approach improves scheduling efficiency and provides optimized schedules with minimum duration, cost, and risk.**

***Keywords*— Software Project Scheduling, Genetic Algorithm, Multi-Objective Optimization, CPM, PERT, Risk-Aware Scheduling, Evolutionary Algorithms.**

I. Introduction

Scheduling of the software project is one of the key activities in software project management because it directly impacts on the completion time of the project, development cost, resource utilization, and risk associated with the software project [1]. Modern software projects comprise of several interrelated activities, including requirement analysis, design, implementation, testing, and deployment. Scheduling helps in effective utilization of the available resources and meeting the deadline and cost budget of the project.

Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT) are two traditional methods commonly used in project management [9]. The former technique deals with identification of the activities that impact on the total completion time of the project, while the latter technique takes into account estimation of the duration of tasks in an uncertain environment. Both techniques focus primarily on the project duration and fail to optimize for multiple objectives [6]. The traditional methods of scheduling are mostly appropriate in an environment where the project is relatively small and certain. However, the nature of modern software development projects is characterized by constantly changing requirements, dispersed project team members, uncertainty in activity duration, and limited resource availability. This means that the scheduling process becomes more complex and static methods of scheduling become less efficient.

Grenze ID: 01.GIJET.12.2.149_26

Thus, the project manager needs to balance different conflicting objectives continuously. Furthermore, incorrect scheduling can result in late software delivery, increase in development cost, conflict of resources, and decrease in software quality. Effective scheduling not only increases the efficiency of the project but also increases the efficiency of resource utilization and reliability of the project. Hence, scheduling methods based on optimization have been gaining importance. Scheduling for software projects, therefore, can be formulated as a multi-objective optimization problem that involves competing objectives such as minimizing project time, decreasing development costs, and minimizing scheduling risks [5]. It is difficult to achieve all these objectives at once using deterministic optimization techniques.

The genetic algorithm, which mimics nature, is an efficient way of doing optimization for solving complicated scheduling problems [7]. GA utilizes a pool of candidate solutions that are improved by the evolutionary operators of selection, crossover, and mutation to come up with optimal solutions [8]. In this research, a multi-objective genetic algorithm (MOGA) based scheduling approach is proposed for the scheduling of software projects. This MOGA approach will simultaneously minimize project duration, cost, and risks while meeting other requirements. This research will compare the performance of the proposed approach against other conventional approaches like CPM and PERT.

II. LITERATURE SURVEY

Software Project Scheduling Problem (SPSP) is one of the most important topics of research in software engineering because of its direct effect on project completion time, development cost, and project resource management. There have been extensive applications of various traditional methods like CPM and PERT to estimate the project schedule and identify critical activities in the project workflow [9]. Despite providing a structured approach to scheduling, these techniques only concentrate on time estimation and cannot optimize all aspects of the project. In practical software development processes, there are many other parameters that need to be considered by project managers such as project duration, development cost, resource assignment, and risk management during scheduling. Therefore, traditional deterministic scheduling techniques may be considered inadequate for software project management.

In an attempt to address the challenges associated with these scheduling methods, various studies have employed evolutionary and metaheuristic algorithms for the resolution of challenging scheduling problems. Genetic Algorithms (GA) have been among the most commonly used optimization algorithms due to their capacity to conduct searches on a large solution space and produce nearly optimal scheduling solutions [7]. In contrast to conventional scheduling approaches, GA-based scheduling algorithms incorporate evolutionary operators including selection, crossover, and mutation in order to enhance the quality of scheduling solutions. Chicano et al. and Luna et al. implemented multi-objective evolutionary algorithms in order to optimize project duration and development costs and produce Pareto-optimal scheduling solutions for software projects [6], [8].

In addition, several other sophisticated scheduling techniques have been developed to enhance the adaptability and scalability of the dynamic environment of software projects. Shen et al. suggested the use of a memetic algorithm based on Q-learning for dynamically changing project scheduling considering the varying project circumstances and resource availability [1]. Subsequently, some other approaches that considered cooperative coevolutionary technique were suggested for solving large-scale software scheduling problems [2]. In a similar way, adaptive dynamic scheduling techniques have been proposed by Silva et al. using particle swarm optimization (PSO) [4]. Although these adaptive optimization techniques enhanced scheduling adaptability and convergence speed, most of them did not incorporate any risk-awareness into their optimization processes.

It has been noted recently that scheduling optimization models must also include project risks. Risk-aware scheduling models were introduced by Milani et al., who considered uncertainty, dependency delays, and resource overload when generating a schedule [5]. Uncertainty-aware scheduling models have been further explored by Li et al., where they tried to develop a schedule that would be robust enough considering various task durations and workload [3]. Even though their approaches helped make schedules more reliable, most scheduling optimization models deal either with cost optimization or uncertainty.

It may be seen from the literature review that multi-objective evolutionary algorithms perform better than deterministic methods for scheduling. There are many research gaps in the field of scheduling techniques. Most of the existing models do not provide optimization for both project time, cost, and risk together. Some of the models lack capability of handling dynamic rescheduling while some lack support for implementation of software engineering projects. Hence, there is a need for developing an intelligent multi-objective scheduling model which is capable of producing adaptive and optimized schedules. Hence, the Multi-Objective Genetic Algorithm approach seeks to overcome these limitations through the incorporation of dependency-aware scheduling, risk-aware optimization, adaptive evolutionary operations, and visualization capabilities into a single software project scheduling system. The objective of the framework is to ensure that the software project schedule generated is optimized, scalable, and realistic.

III. Methodology

In the proposed approach, MOGA is applied for scheduling optimization in software projects by considering minimization of project completion time, software development costs, and scheduling risks concurrently. The approach considers a software project as an aggregation of interrelated activities with characteristics like activity duration, resource requirement, dependencies, costs, and risks among others.

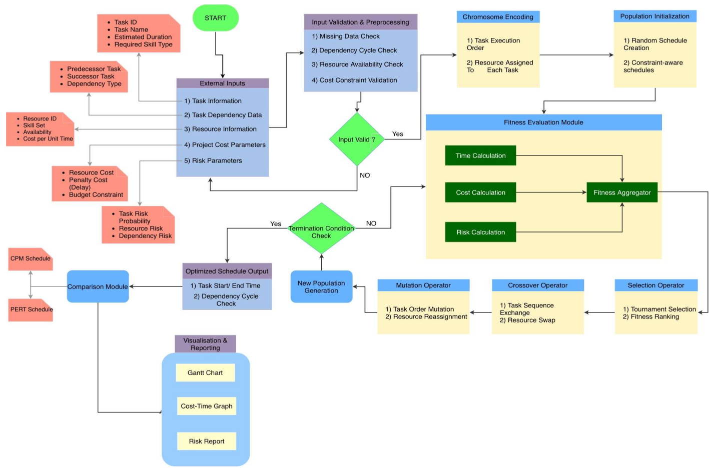


Figure 1. Workflow of the Proposed Moga-based Scheduling System

Fig. 1. shows the steps involved in the MOGA-based framework for scheduling software projects. Scheduling starts with the acquisition of project information, which includes task information, dependency information, resource information, parameters associated with project costs, and parameters relating to project scheduling risks. Preprocessing ensures that all information is valid by ensuring that dependency cycles do not exist, no missing values are included, sufficient resources are available, and that there is adequate budget. With the information collected, it is encoded into the chromosome structures. Chromosomes encode different scheduling solutions based on the order of tasks to be performed and the resources used in performing them. From there, initial population of the schedules is achieved using random dependency-sensitive scheduling.

Once the population of the schedules is obtained, their fitness is assessed using the fitness evaluation module. The fitness evaluation module computes the values of the different objectives including project duration, total project cost, and scheduling risks. The objectives are combined using weights within the fitness function to determine the quality of the schedules. From there, tournament selection is carried out to select the high-quality scheduling solutions, while crossover and mutations create new offspring schedules. After each iteration, the population evolves and certain stopping criteria such as maximum number of generations or optimum fitness value are considered. After successful optimization, the optimization framework generates an optimized schedule that contains information on when tasks should be executed and if there are any dependency issues. The optimized schedules are then compared to the conventional methods used for scheduling such as CPM and PERT by means of Gantt and cost-time charts.

The evaluation of solutions is done by using a weighted multi-objective fitness function. There are three main objectives that are considered in the optimization procedure, which include project duration, project cost, and project risk. Fitness function definition is as follows:

$$Fitness = w_1 D + w_2 C + w_3 R$$

where:

- D represents project duration,
- C represents project cost,
- R represents scheduling risk,
- w1, w2, w3 are weighting coefficients.

Genetic Algorithm makes use of evolutionary operations like selection, crossover, and mutation to optimize the solution for the problem. Schedules with high quality are obtained through tournament selection while ordered crossover with dependency repairing helps in generating proper offspring. Mutation operations help in changing the order of tasks as well as resource allocation.

A. Genetic Algorithm Configuration:

The proposed algorithm makes use of a population based Multi-Objective Genetic Algorithm using 500 generations and tournament selection with tournament size of 3. The swap mutation operator is employed with a probability of 20%. The dependency aware ordered crossover operator is used to generate valid offspring by considering the dependencies. Very high penalties are imposed on dependency violation, deadline violation, and budget overflow scenarios.

B. Chromosome Representation and Initialization

In the proposed scheduling scheme, every chromosome corresponds to one of the possible software project schedules. Each chromosome holds information related to the task execution order, resource assignment, and execution modes of the respective software project activities. The encoding method guarantees the preservation of all dependencies between tasks in the schedule. Chromosomes are generated in the initialization stage based on multiple feasible schedules created through dependency-aware randomization. Topological sorting according to Kahn's algorithm is used to ensure correct task execution order while generating chromosomes. Random tie breaking is implemented in initialization to enhance the population diversity and explore alternative scheduling solutions.

The execution modes of every task are also included in the chromosome. In the proposed model, there are three possible execution modes: Fast mode, Normal mode, and Slow mode. Execution in Fast mode leads to minimal execution time but higher resource costs. On the other hand, Slow mode results in reduced execution costs but takes more time than other execution modes. Normal mode is a compromise between execution time and software development cost.

C. Fitness Evaluation Strategy

Each schedule produced by the algorithms is tested for its efficiency with respect to the project duration, cost of the entire project, and risk associated with the scheduling. The project duration can be obtained from the relationships between different tasks and their completion durations, while the cost includes both the cost of performing the tasks and the cost of scheduling. In addition to the constraints mentioned before, the risks associated with the project include delay, duration uncertainty, and overloading of resources.

D. Evolutionary Operations

MOGA is designed to enhance scheduling results through the application of evolutionary techniques including tournament selection, ordered crossover, and swap mutation. The tournament selection technique helps select the best performing chromosomes in the population, whereas crossover helps create offspring through the exchange of task sequence and resource allocation details. Mutation adds diversity in the population through alterations in task sequence and task execution modes.

The dependency aware scheduling approach is among the main features of the suggested framework. In the process of optimization, each possible schedule is tested for its compliance with dependencies within the project. The approach allows avoiding creation of an infeasible schedule and increasing practical value of the suggested optimization framework. In addition, the choice of adaptive execution mode helps optimize the scheduling process in terms of balancing execution speed and cost. Important tasks are executed with faster modes of execution where necessary, while unimportant tasks use balanced execution modes to minimize unnecessary costs associated with the project.

The optimal schedules obtained using the suggested framework are compared with the conventional scheduling methods like Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT). In CPM, the critical path for minimum project completion time is identified, while in PERT, the expected completion time for each task is estimated using probabilistic scheduling analysis.

$$T_E = \frac{O + 4M + P}{6}$$

where:

➢ = Optimistic Time

➢ M = Most Likely Time
➢ P = Pessimistic Time

This process is continued until certain predetermined stopping criteria have been met. In the end, the optimized schedule will show enhanced performance in comparison to traditional scheduling techniques in terms of project duration, cost effectiveness, and risk avoidance.

IV. Results And Discussion

TABLE I. Comparison of GA, CPM and Pert Results

Metric	GA	CPM	PERT
Project Duration	21.0 days	29.0 days	30.2 days
Total Project Cost	₹32100	₹33450	₹34488
Daily Overhead	₹400/day	₹400/day	₹400/day
Risk Exposure	2.27	2.53	2.57

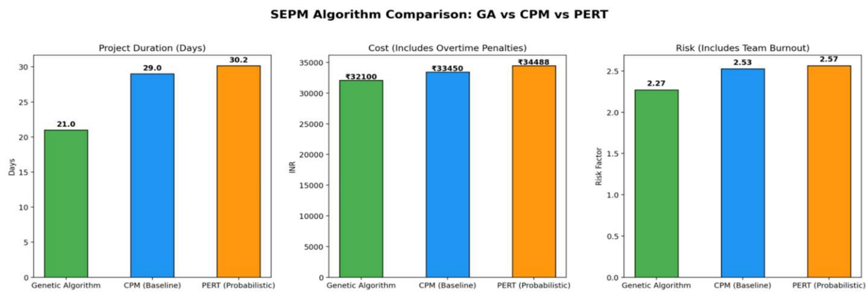
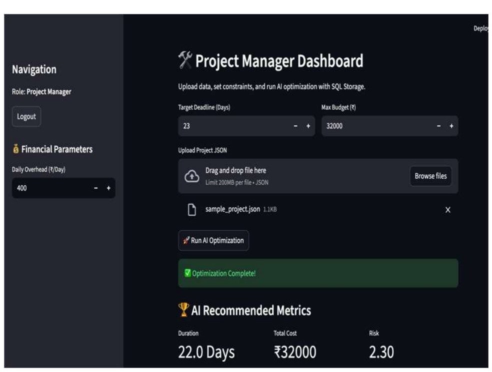


Figure 2. Comparison of GA, CPM and Pert Results

From the results shown in Table I and Fig. 2. above, it can be seen that the suggested MOGA-based scheduling method performs better than conventional scheduling techniques when it comes to time taken to complete the project, cost of development, and risk reduction. The Genetic Algorithm helped in completing the project in 21.0 days while CPM and PERT took 29.0 days and 30.2 days respectively. Moreover, the proposed framework resulted in a total cost of ₹32100, which is lower than ₹33450 in CPM and ₹34488 in PERT. This decrease in cost is mainly due to efficient resource utilization and better sequencing. In addition, the proposed framework has resulted in the least value of risk exposure, i.e., 2.27.

The shortened duration of the project obtained from the suggested MOGA-based system is proof of how well the evolutionary optimization algorithm can be applied to solve scheduling issues that involve software dependencies. In contrast to other deterministic scheduling techniques like CPM and PERT, the suggested system allows the simultaneous exploration of various scheduling options. This helps deal with parallel processing and resource allocation. This framework can also be used to improve cost optimization performance via adaptive choice of execution modes and resource usage. Fast execution modes are only selected for critical scheduling operations whereby faster execution will have a significant effect on reducing the time taken to complete the entire project. Non-critical operations are carried out by using balanced or slow execution modes to avoid incurring unnecessary costs.



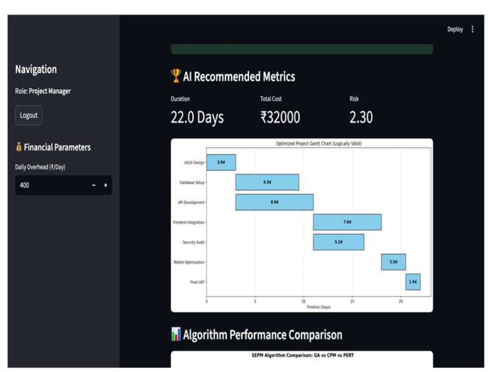


Figure 3. Project Manager Dashboard Interface Figure 4. Optimized Schedule

Fig. 3. shows the Graphical User Interface that was designed for the proposed Multi-Objective Genetic Algorithm-based scheduling approach. The dashboard serves as the main interface between the project manager and the scheduling tool. Through this interface, users will be able to load their project data sets in formats like JSON and set crucial parameters such as project deadline, maximum budget, and daily overhead project costs. This interface will help make scheduling easy by enabling the user to conduct optimization operations without having to interact with the actual algorithmic implementation. The dashboard will also enable users to start the Genetic Algorithm optimization and monitor the execution process. Once the optimization process is complete, the results will be presented on the dashboard where users will be able to view the output information including the project duration, total project cost, and risks involved. This makes the interface user-friendly and more efficient in analysis of scheduling performance.

Fig. 4. illustrates the optimized project schedule produced through the proposed Multi-Objective Genetic Algorithm framework. This illustration involves the use of a Gantt chart that indicates task execution times, sequence of tasks, dependencies between tasks, and resource allocation for each phase of the project. Tasks are placed in terms of their optimal start and end time which gives insight into how the scheduling scheme produced by the framework looks like. From the Gantt chart, it is evident that the proposed framework effectively ensures logical dependencies among tasks while minimizing the total project duration. Parallel execution of independent tasks is also possible which means efficient use of available resources and scheduling bottlenecks avoided. From the visualization perspective, the project manager is able to understand workflow allocation, identify critical tasks, and evaluate the effectiveness of the scheduling scheme.

V. Conclusion and Future Work

The suggested Multi-Objective Genetic Algorithm (MOGA)-based scheduling approach is an appropriate means to solve scheduling problems of software projects with multiple objective functions. In contrast to conventional scheduling algorithms such as CPM and PERT, the suggested system optimizes the software project scheduling problem by minimizing the project duration, cost, and scheduling risk subject to the dependencies and resource limitations. The results of experiments show that the suggested scheduling framework leads to better scheduling performance by minimizing the project duration, cost, and risk exposure. The application of evolutionary techniques including selection, crossover, and mutation in the MOGA-based scheduling process facilitates searching of the scheduling space and generating optimized scheduling solutions.

Moreover, the proposed framework also allows more flexibility and adaptability to the changing environment of software projects than the deterministic approach to scheduling. The use of visualization in form of Gantt charts will help project managers to analyze the optimized schedule and resource allocation strategy. As far as future research is concerned, the proposed framework can be improved in various ways. First of all, one can consider incorporating AI tools like Machine Learning and Reinforcement Learning for improving adaptive scheduling capabilities. Second, it will be useful to combine Genetic Algorithms with PSO or ACO for faster convergence and more effective optimization. Finally, integration with industrial project management software like Jira and Microsoft Project will help increase the efficiency of the proposed framework.

References

[1] X.-N. Shen, L. L. Minku, N. Marturi, Y.-N. Guo, and Y. Han, “A Q-learning-based memetic algorithm for multi-objective dynamic software project scheduling,” Information Sciences, vol. 428, pp. 1–29, 2018. doi: 10.1016/j.ins.2017.10.041.

[2] X.-N. Shen, Y.-N. Guo, and A. Li, “Cooperative coevolution with an improved resource allocation for large-scale multi-objective software project scheduling,” Applied Soft Computing, vol. 88, Art. no. 106059, 2020. doi: 10.1016/j.asoc.2019.106059.

[3] H. Li, H. Zhu, L. Zheng, and F. Xie, “Software project scheduling under activity duration uncertainty,” Annals of Operations Research, vol. 338, no. 1, pp. 477–512, 2024. doi: 10.1007/s10479-023-05343-0.

[4] G. F. D. Silva, L. Silva, and A. Britto, “Dynamic software project scheduling problem with PSO and dynamic strategies based on memory,” in Intelligent Systems – BRACIS 2020, Lecture Notes in Computer Science, Springer, 2020. doi: 10.1007/978-3-030-61377-8_6.

[5] S. Naderi, D. Rahmani, and R. Tavakkoli-Moghaddam, “Optimizing resource-constrained project scheduling using hybrid metaheuristics: A probabilistic look,” 2022. doi: 10.1155/2022/7711383.

[6] F. Milani et al., “Risk-aware project scheduling using metaheuristics,” Computers & Industrial Engineering, 2021. doi: 10.1016/j.cie.2020.106876.

[7] F. Chicano, F. Luna, A. J. Nebro, and E. Alba, “Using multi-objective metaheuristics to solve the software project scheduling problem,” in Proc. Genetic and Evolutionary Computation Conference (GECCO), 2011. doi: 10.1145/2001576.2001833.
[8] E. Alba and F. Chicano, “Software project management with genetic algorithms,” Information Sciences, vol. 177, no. 11, pp. 2380–2401, 2007. doi: 10.1016/j.ins.2006.12.020.
[9] D. Debels, B. De Reyck, R. Leus, and M. Vanhoucke, “A hybrid scatter search/electromagnetism meta-heuristic for project scheduling,” European Journal of Operational Research, vol. 169, no. 2, pp. 638–653, 2006. doi: 10.1016/j.ejor.2004.08.020.
[10] P. Kunekar, K. Vayadande, O. Kulkarni, K. Ingale, R. Kadam and S. Inamdar, "OCR based Cheque Validation using Image Processing," 2023 5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE), Navi Mumbai, India, 2023, pp. 1-5, doi: 10.1109/ICNTE56631.2023.10146687.
[11] K. Vayadande, P. Sheth, D. Pawal, A. Pathak, K. Paralkar and S. Patil, "Simulation of CPU Scheduling Algorithms for Efficient Execution of Processes," 2023 International Conference for Advancement in Technology (ICONAT), Goa, India, 2023, pp. 1-6, doi: 10.1109/ICONAT57137.2023.10080113.
[12] Kuldeep Vayadande; Komal Sunil Munde; Amol A. Bhosle; Aparna R. Sawant; Ashutosh M. Kulkarni, "Software Engineering in Machine Learning Applications," in How Machine Learning is Innovating Today's World: A Concise Technical Guide , Wiley, 2024, pp.159-172, doi: 10.1002/9781394214167.ch12.
[13] Kuldeep Vayadande; Ashutosh M. Kulkarni; Gitanjali Bhimrao Yadav; R. Kumar; Aparna R. Sawant, "A Comprehensive Analysis of Various Tokenization Techniques and Sequence-to-Sequence Model in Natural Language Processing," in How Machine Learning is Innovating Today's World: A Concise Technical Guide , Wiley, 2024, pp.1-11, doi: 10.1002/9781394214167.ch1.
[14] K. Vayadande et al., "Heart Disease Prediction using Machine Learning and Deep Learning Algorithms," 2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), Greater Noida, India, 2022, pp. 393-401, doi: 10.1109/CISES54857.2022.9844406.
[15] K. Vayadande, O. Mirajkar, S. Mhetre, H. Mehta, S. Markande and S. Mohol, "The Future of Skin Cancer Detection: Machine Learning and Convolutional Neural Networks," 2024 5th International Conference on Data Intelligence and Cognitive Informatics (ICDICI), Tirunelveli, India, 2024, pp. 1282-1290, doi: 10.1109/ICDICI62993.2024.10810972
[16] K. Vayadande, N. Wakodikar, K. Oswal, T. Chauhan, V. Shivade and S. Ubale, "Traffic Flow Optimizer: Traffic Signal Control With Vehicle Density and Algorithms," 2023 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), B G NAGARA, India, 2023, pp. 1-6, doi: 10.1109/ICRASET59632.2023.10420165.
[17] P. Dahiwal et al., "NeuroBridge-AI: A Hybrid Biophysical and Machine Learning Framework for Multi-Scale Neural Simulation," in IEEE Access, doi: 10.1109/ACCESS.2026.3700047.
[18] Vayadande, K., Bodhe, Y., Patil, A., Jadhav, A.N., Mokashi, D.K., Mane, S.C., Mane, R.M. and Jadhav, R.V. (2026). Hybrid Optimization Techniques for Ayurvedic Medicine Recommendation. In Driving Innovation by Dynamic Optimization (eds A. Dey, S.N. Mohanty, R. Kumar and T.R. Kumar). https://doi.org/10.1002/9781394315727.ch15
[19] Vayadande, K., Munde, K. S., Bhosle, A. A., Sawant, A. R., & Kulkarni, A. M. (2024). Software Engineering in Machine Learning Applications: A Comprehensive Study. How Machine Learning is Innovating Today's World: A Concise Technical Guide, 159-172.
[20] Mumbare, S., Shivam, K., Lokhande, P., Zaware, S., Deshpande, V., & Vayadande, K. (2022). Software Controller using Hand Gestures. In ITM Web of Conferences (Vol. 50, p. 01008). EDP Sciences.
[21] Vayadande, K., Sheth, P., Pawal, D., Pathak, A., Paralkar, K., & Patil, S. (2023, January). Simulation of CPU scheduling algorithms for efficient execution of processes. In 2023 International Conference for Advancement in Technology (ICONAT) (pp. 1-6). IEEE.
[22] Vayadande, K., Bodhankar, A., Mahajan, A., Prasad, D., Dhakalkar, R., & Mahajan, S. (2022, December). An Improved Way to Implement Round Robin Scheduling Algorithm. In International Conference on Computer Vision, High-Performance Computing, Smart Devices, and Networks (pp. 403-414). Singapore: Springer Nature Singapore.
[23] Vayadande, K., Shaikh, U., Ner, R., Patil, S., Nimase, O., & Shinde, T. (2023, May). Mood detection and emoji classification using tokenization and convolutional neural network. In 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 653-663). IEEE.