SINTECHCOM: Science, Technology and Communication Journal
Vol. 5, No. 2, February 2025, pp. 67-80, DOI: 10.59190/stc.v5i2.275

Final assignment exam scheduling optimization using genetic algorithms with tournament selection techniques and violated directed mutation (VDM)

Dian Meliani Kusuma Dewi, Aina Musdholifah*
Department of Computer Science and Electronics, Universitas Gadjah Mada, Sleman 55281, Indonesia

ABSTRACT

Scheduling the final assignment exam is an important process that requires careful planning to ensure smooth implementation for each student. This process involves the stages of archiving final assignment submission files, determining supervisors and examiners, as well as preparing seminar and trial schedules. However, obstacles that often arise include conflicting schedules, long execution times, and low fitness values. To overcome this problem, the genetic algorithm approach is used to optimize scheduling. This algorithm can handle complex problems with a wide search space, although it has weaknesses in selecting appropriate parameters and the time required to reach the optimal solution. Genetic algorithm optimization techniques such as violated directed mutation (VDM) and tournament selection are used in this research. Previous research shows that VDM provides better results than other methods, while tournament selection improves the desired solution. It is hoped that the use of genetic algorithms with VDM and tournament selection will overcome the problem of conflicting schedules and increase the execution speed in final project exam scheduling.

ARTICLE INFO

Article history:

Received Jan 15, 2025
Revised Feb 16, 2025
Accepted Feb 23, 2025

Keywords:

Final Assignment Exam
Genetic Algorithm
Scheduling
Tournament Selection
VMD

* **Corresponding Author**
E-mail address: aina_m@ugm.ac.id

1. INTRODUCTION

Scheduling final project exams is a complex process that requires structured planning to ensure that all students can participate smoothly [1-4]. This process involves several stages, including managing the submission files for final projects, determining supervisors and examiners, and arranging schedules for seminars and final project defenses [5, 6]. The information is consolidated and analyzed to produce an efficient and effective exam schedule.

The scheduling process begins with data collection and needs analysis. Next, scheduling planning and development are carried out [7-11]. The final stage involves implementing the schedule and monitoring it to ensure all requirements are met [12, 13]. However, challenges such as schedule conflicts, long execution times, and low fitness values often arise during this process.

Combining genetic algorithms (GA) with other algorithms can reduce scheduling errors [14-17]. However, when processing large datasets, this approach may result in slower execution than a standalone genetic algorithm [18, 19]. The efficiency and output of a genetic algorithm largely depends on the operators used [20-23]. Therefore, it is essential to select an appropriate combination of operators to optimize the genetic algorithm, ensuring accurate scheduling with faster processing times [24, 25]. This study proposes using the violated directed mutation (VDM) technique during the mutation phase and Tournament Selection during the parent selection phase of the genetic algorithm.

This research focuses on enhancing the genetic algorithm by incorporating VDM and Tournament Selection techniques to produce conflict-free final project exam schedules. By applying these techniques to the genetic algorithm, scheduling performance is expected to improve. The

resulting schedules will be compared with those generated using conventional genetic algorithm methods to evaluate their effectiveness.

2. MATERIALS AND METHOD

This study focuses on developing a Genetic Algorithm using the Violated Directed Mutation (VDM) and Tournament Selection techniques to produce final project exam schedules with minimal conflicts. By applying these techniques within the stages of the Genetic Algorithm, it is expected to enhance scheduling performance. The scheduling results obtained with this approach will be compared to those generated using conventional Genetic Algorithm stages or the standard GA process.

2.1. Genetic Algorithm Stages

2.1.1. Experiment Data Input

The research begins with inputting the data required for scheduling final project exams. The data includes student information, rooms, exam times, lecturers, and lecturer schedules. Lecturer schedules are necessary to ensure that assigned examiners are not scheduled for other activities during the specified exam times. Once all data is input into the system, the population initialization process will begin.

2.1.2. Initial Population Initialization

The next step is initializing the population with a size determined by the input parameter. This initialized population will serve as the starting population for the scheduling process. In the chromosome representation, value encoding is used with a solution string for each gene in the chromosome. Each gene represents a combination of room code, exam time code, and the lecturers assigned to examine the students during the final project exam. The chromosome will contain genes equal to the number of students scheduled for exams.

Table 1. Gene representation in chromosomes.

Design of genes in the chromosome					Example of gene representation
Room code	Time code	Examiner 1	Examiner 2	Examiner 3	4, 2, 5, 14, 23

In Table 1, each gene in the chromosome consists of 5 parameters. The explanation of each section is as follows:

a. Room Code: This represents the room code used as the location for the final project exam.
b. Exam Time Code: This represents the exam time code, which includes a combination of the day's name and the start time of the exam. All possible combinations are considered for generating scheduling solutions.
c. Lecturers 1, 2, and 3: These are the examiners assigned to each student, chosen randomly from the available lecturer data.

The chromosome formation process begins by inserting student data into the genes of the chromosome. Subsequently, this is combined with room data, exam time data, and lecturer data. For the lecturer data, random selection is performed with the condition that Lecturer 1, 2, and 3, along with the student's supervising lecturer, must not be the same. The flowchart of the population initialization process in this research can be seen in Figure 1.

The result of this population initialization will serve as the initial population in the final project exam scheduling process for this research. This initial population will be used in all experiments, ensuring that the initial population remains the same for each trial. An example of the initial population initialization process is as follows in Table 2.

2.1.3. Determination of GA Parameters

In this research, the chromosome length (g) corresponds to the number of students participating in the final project exams. The population size (p) will be set to a minimum value equal to the size of the chromosome. The mutation probability (pm) will be adjusted throughout the genetic algorithm process.

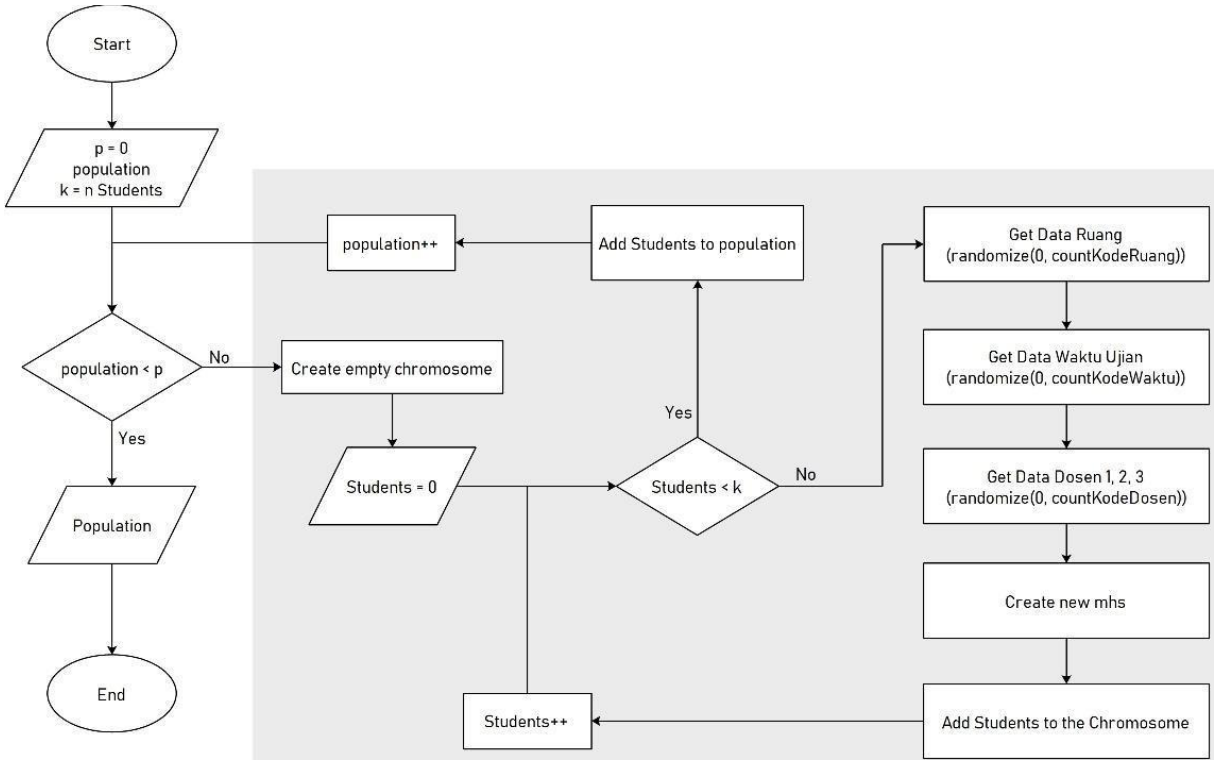

Figure 1. Flowchart of population initialization.

Table 2. Example of chromosome results population initialization.

Individual	Chromosome				
Individual 1	3, 11, 21, 26, 2	2, 11, 8, 28, 25	2, 2, 10, 24, 9	2, 6, 27, 10, 6	2, 15, 17, 24, 19
Individual 2	4, 11, 23, 30, 21	1, 2, 28, 23, 6	2, 11, 17, 7, 26	1, 11, 23, 17, 1	3, 13, 13, 22, 15
Individual 3	4, 11, 23, 30, 12	1, 4, 2, 17, 2	4, 7, 11, 24, 17	2, 6, 22, 21, 16	4, 5, 25, 29, 2
Individual 4	3, 3, 17, 27, 22	2, 10, 4, 16, 5	3, 13, 11, 19, 24	1, 12, 10, 5, 29	3, 7, 6, 24, 7
Individual 5	3, 3, 17, 27, 10	1, 3, 8, 21, 11	1, 11, 23, 9, 14	1, 12, 10, 5, 14	3, 7, 6, 24, 5
Individual 6	3, 11, 21, 26, 30	1, 3, 19, 30, 12	2, 3, 25, 3, 21	4, 6, 20, 12, 23	4, 2, 17, 20, 4

The pm value is determined through multiple experiments on the population. These experiments aim to obtain the average fitness value and the time required for the genetic algorithm process. The best pm value, which provides the highest fitness and the shortest processing time, will be selected.

The parameter search is conducted within systems using genetic algorithms, including systems employing Violated Directed Mutation and Tournament Selection within their algorithm stages. The results from experiments yielding the highest average fitness will be used for system testing.

2.1.4. Running Basic GA

The first genetic algorithm used is the basic GA, which utilizes Roulette Wheel Selection for selection and Random Mutation for mutation. Figure 2 illustrates the flow of the basic GA and the steps performed.

Experiments with the basic GA were conducted twice, using a fitness approach with and without weighting. After the initial population was created, fitness was calculated, followed by a selection phase using Roulette Wheel Selection to choose two individuals as parents for crossover. The next step is the uniform crossover between the two parents, followed by random mutation. The crossover and mutation processes are repeated according to the determined population size. Subsequently, the fitness of the new individuals is calculated, and all individuals from the initial population, along with children resulting from crossover and mutation, are sorted. The best individuals are then selected for the next generation, matching the determined population size.

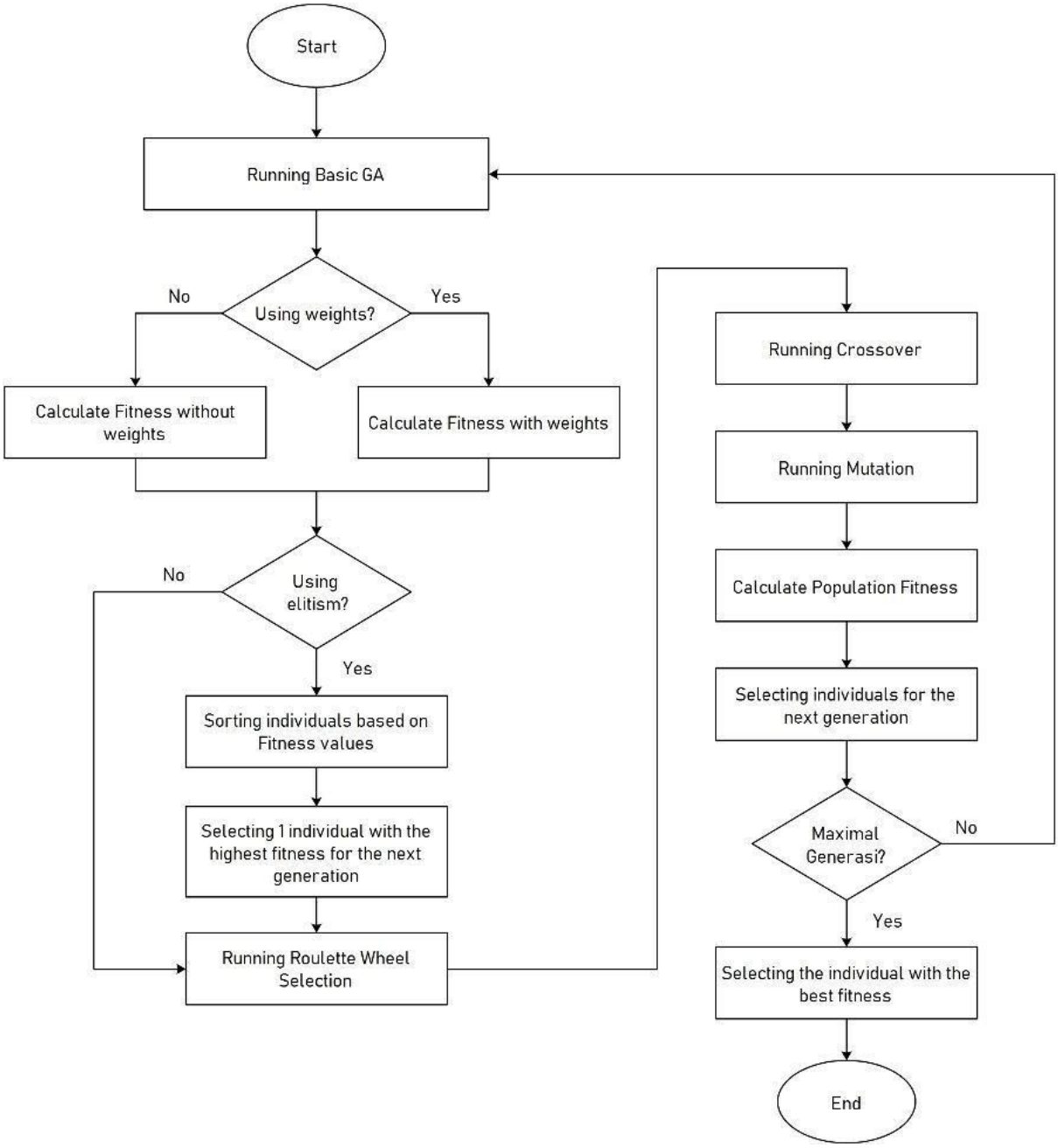

Figure 2. Basic genetic algorithm flow.

2.1.5. Running Modified GA

The next step of the research involves using a modified GA with Tournament Selection in the selection phase and Violated Directed Mutation in the mutation phase. Figure 3 illustrates the flow of the modified GA and the steps performed.

The modified GA experiment was conducted twice, using a fitness approach with and without weighting. Starting with the initial population, fitness values were calculated, and the selection process used tournament selection to choose two individuals as parents for crossover. The crossover stage, using uniform crossover, produces new individuals, which are then mutated using Violated Directed Mutation (VDM). VDM performs mutation in a targeted manner, focusing only on genes that violate constraints. All individuals from the initial population and children resulting from crossover and mutation are sorted, and the best individuals are selected for the next generation based on the determined population size.

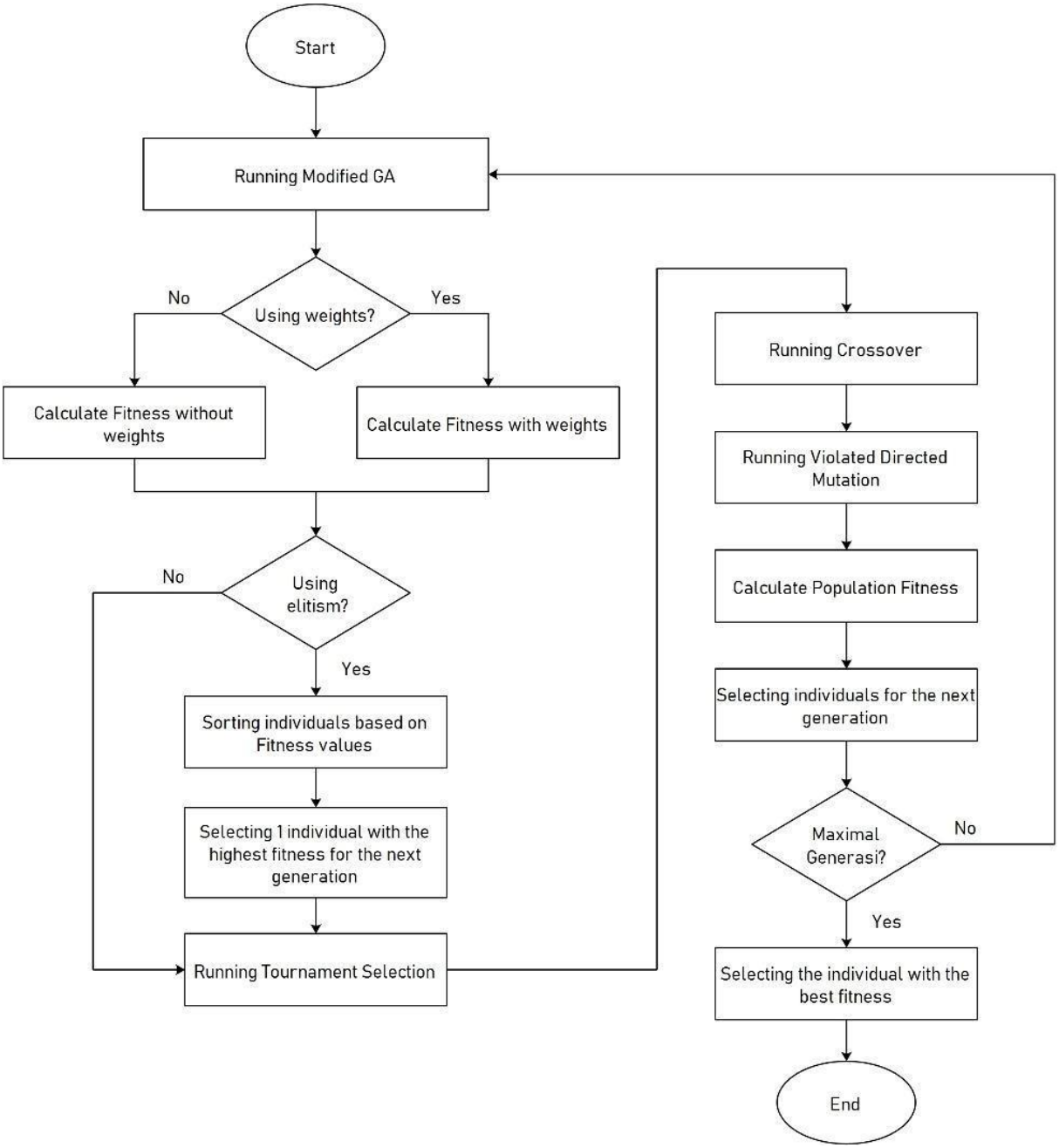

Figure 3. Modified genetic algorithm flow.

2.2. Genetic Algorithm Design

2.2.1. Population Evaluation

After the initial population is formed, the fitness of each chromosome is calculated based on the number of violated constraints, using a penalty strategy for each violation. In this research, fitness calculation is divided into three main aspects: schedule, academic lab, and lecturer compatibility. The calculation rules used for each aspect are explained in the following Table 3.

Regarding academic lab compatibility between examiners and students, the calculation is based on the compatibility scheme between academic labs. Compatibility is calculated such that the more compatible the pair, the lower the compatibility score. This is done to minimize the constraint value. Thus, a lower compatibility score indicates a better match between the academic labs of the examiner and the student. The compatibility point scheme is shown in the Table 4.

Table 3. Weighting of constraint values.

No	Calculation aspects	Weight
1	Schedule	
	a. Exam time - room conflict	+1
	b. Examiner lecturers 1, 2, and 3 are added to the unavailable list	+1
	c. Examiner lecturers 1, 2, and 3 are the same as the supervising lecturer	+1
2	Lecturer compatibility	
	a. Examiner lecturers 1, 2, and 3 from the previous exam are not the same as examiner lecturers 1, 2, and 3 in the chromosome	+1

Table 4. Scheme of compatibility between student's lab and examiner's lab.

	SC	AK	SKJ	RPL	EK
SC	1	2	5	3	3
AK	2	1	4	2	5
SKJ	5	4	1	2	3
RPL	3	3	2	1	5
EK	3	5	3	5	1

After evaluating the aspects mentioned above, the resulting constraint values are summed. In the third and fourth experiments to be conducted, each aspect will be assigned a weight based on the priority of its impact on the individual. Using the predetermined weights, the fitness function used for calculating fitness values is as follows:

$$f(k) = \frac{1}{(c_{schedule} c_{lab} c_{lecturer})} \quad (1)$$

For the fitness function using weights, the fitness calculation will use the following Equation (2):

$$f(k) = \frac{1}{(0.5 \times c_{schedule}) + (0.35 \times c_{lab}) + (0.15 \times c_{lecturer})} \quad (2)$$

where, $c_{schedule}$ is the constraint generated from the schedule aspect, c_{lab} is the constraint generated from the academic lab aspect, and $c_{lecturer}$ is the constraint generated from the lecturer compatibility aspect.

The closer the fitness value is to 1, the more optimal the solution obtained, as no constraints are violated by the individual. Next, the population with calculated fitness values enters the selection phase. Figure 4 illustrates the complete process of fitness calculation in this research.

2.2.2. Elitism

After the fitness values of the population are calculated, the elitism process will be executed. Elitism is applied if the previously entered elitism parameter is set to true. Elitism involves selecting individuals with the highest fitness values to be included in the next generation as the best solutions. The goal of elitism is to preserve the best solutions to prevent them from being lost due to crossover or mutation processes.

2.2.3. Selection

2.2.3.1. Roulette Wheel Selection

The selection phase in GA basic uses Roulette Wheel Selection, where this selection is carried out similarly to a roulette wheel game, randomly selecting individuals based on the wheel spins. The portion on the wheel is based on the fitness values of each individual, ensuring that individuals with higher fitness have a greater chance of being selected. The flowchart of the roulette wheel selection process can be seen in the Figure 5.

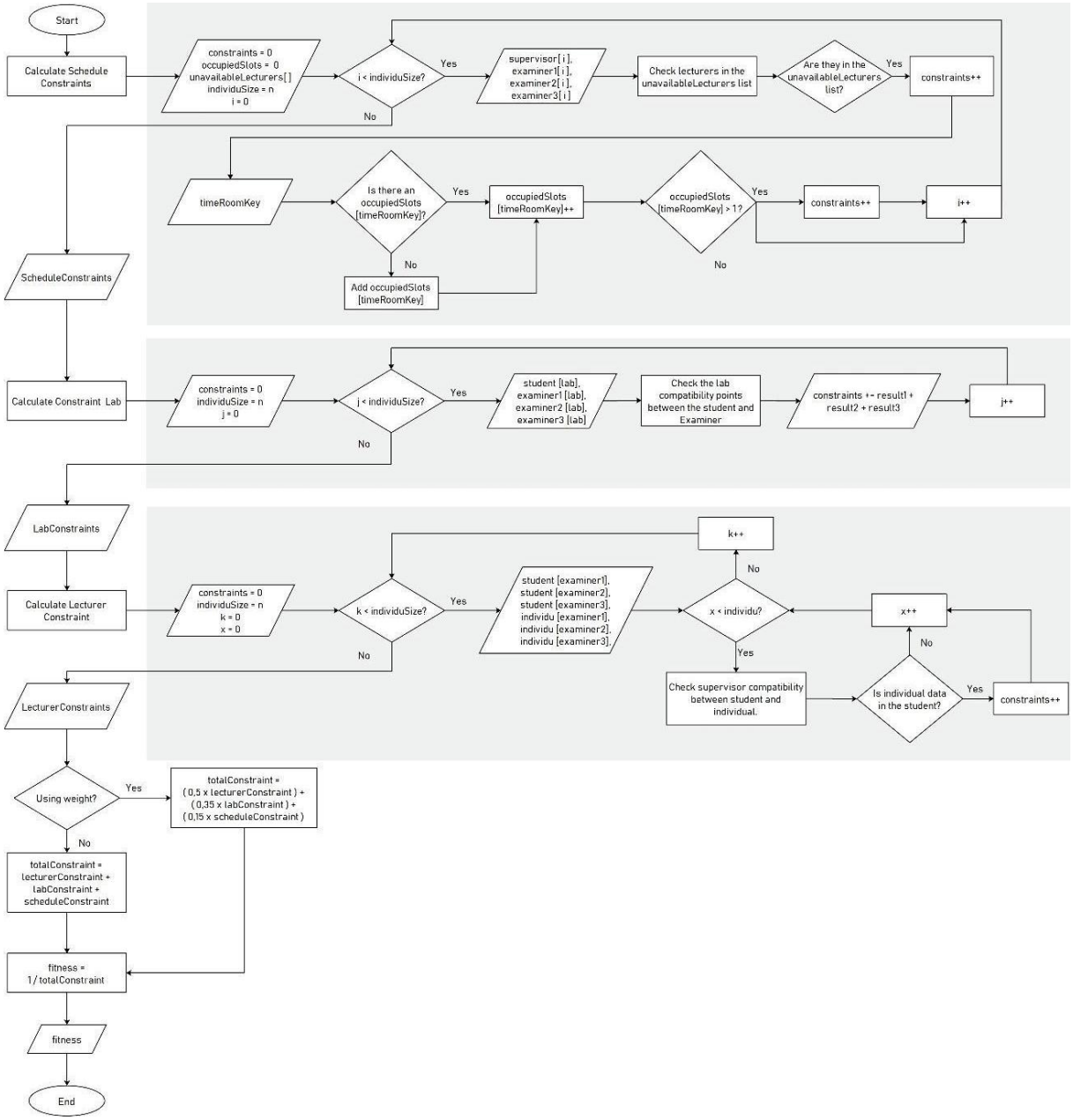

Figure 4. Flowchart of fitness function.

2.2.3.2. Tournament Selection

In this GA selection phase, Tournament Selection is used, where a random selection of n chromosomes is made. These chromosomes compete to determine the best chromosome by comparing their fitness values. The flowchart of the selection process can be seen in the Figure 6.

2.2.4. Crossovers

The crossover process is performed to obtain new chromosomes resulting from the combination of parents selected through the previous selection process. The individuals chosen in the previous selection process become parents in this crossover phase, which utilizes uniform crossover. Crossover is carried out based on the provided pattern, where if the pattern is 1, genes from parent 1 are used, and if the pattern is 0, genes from parent 2 are used. The flowchart of this uniform crossover process can be seen in the Figure 7.

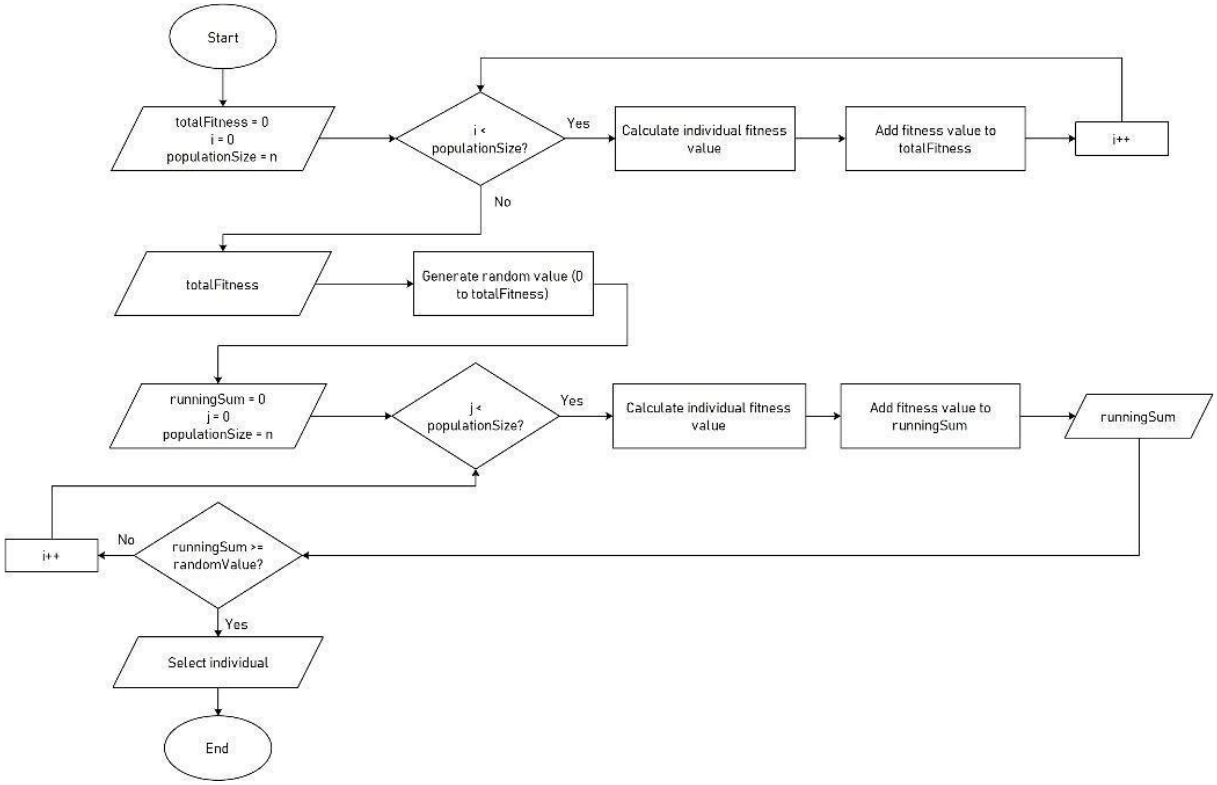

Figure 5. Flowchart of Roulette wheel selection.

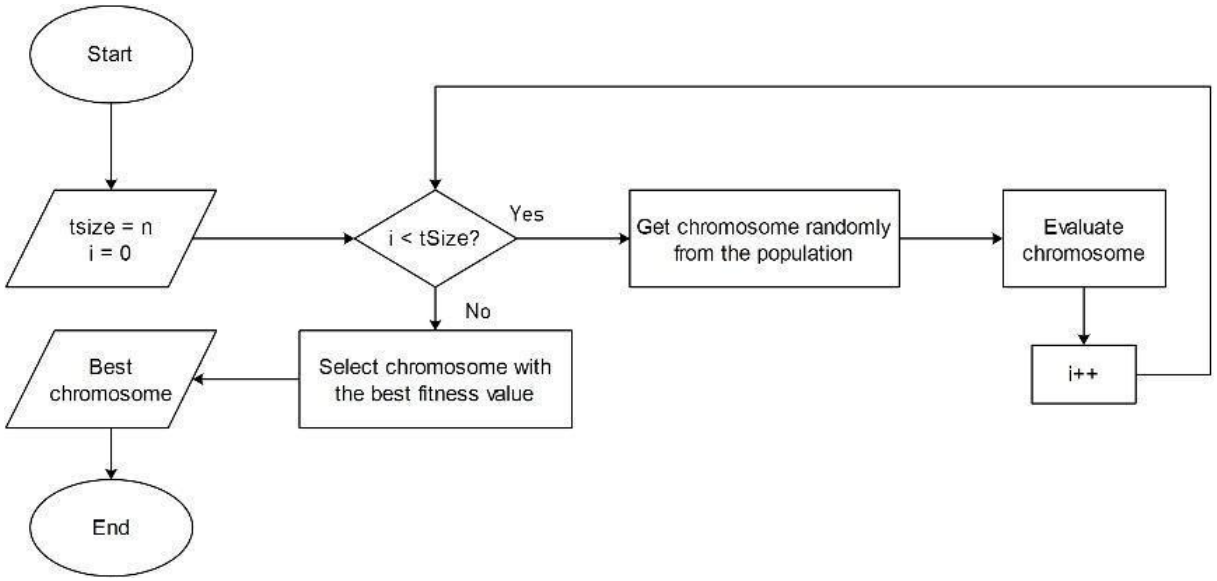

Figure 6. Flowchart of tournament selection.

2.2.5. Mutations

2.2.5.1. Random Mutations

The basic GA mutation phase uses random mutation. This mutation is performed on each chromosome gene by filling in random values for mutated genes. A gene will mutate if the mutation rate is less than the random value present in the gene. Figure 8 illustrates the flowchart of the random mutation process.

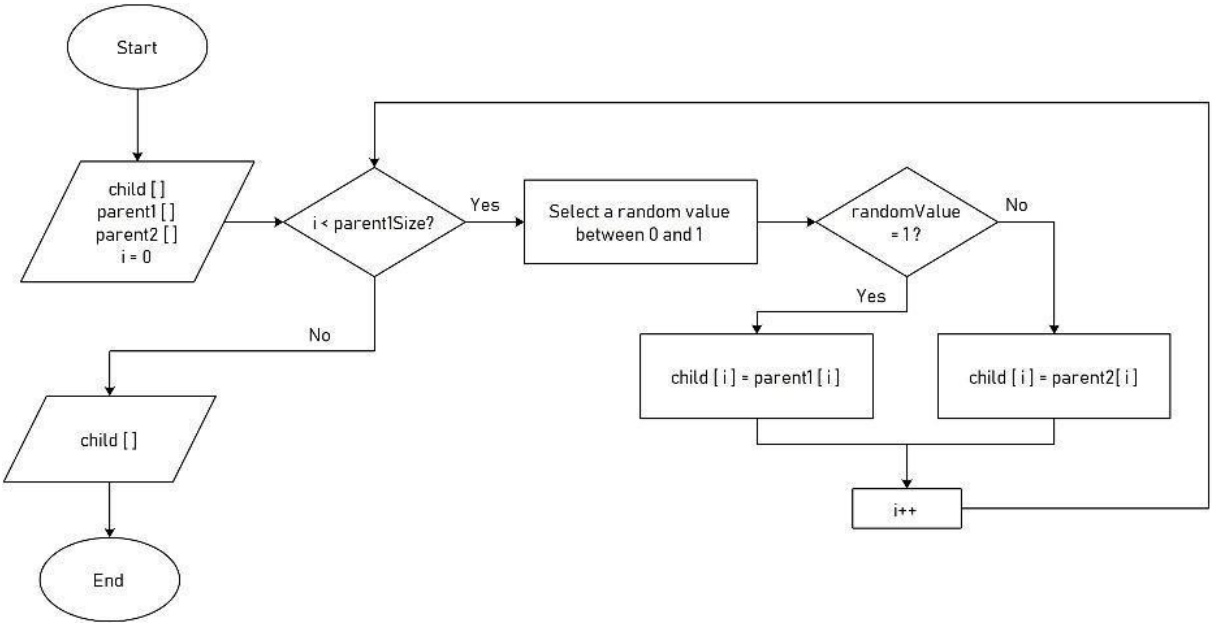

Figure 7. Flowchart of crossover.

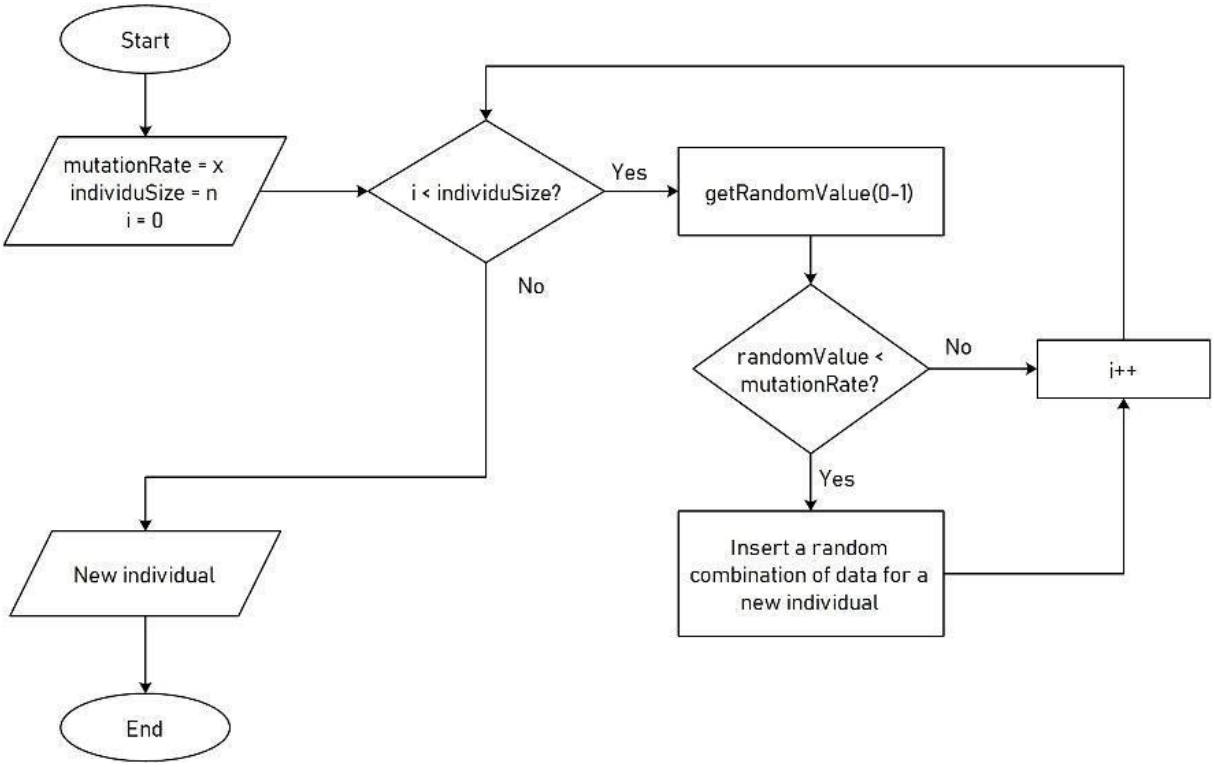

Figure 8. Flowchart of random mutation.

2.2.5.2. Violated Directed Mutation

The modified GA mutation phase uses Violated Directed Mutation (VDM). VDM involves mutating genes that contain violations. For genes with violations, a random allele value is modified with another value that does not violate the rules. Figure 9 illustrates the flowchart of the VDM mutation process.

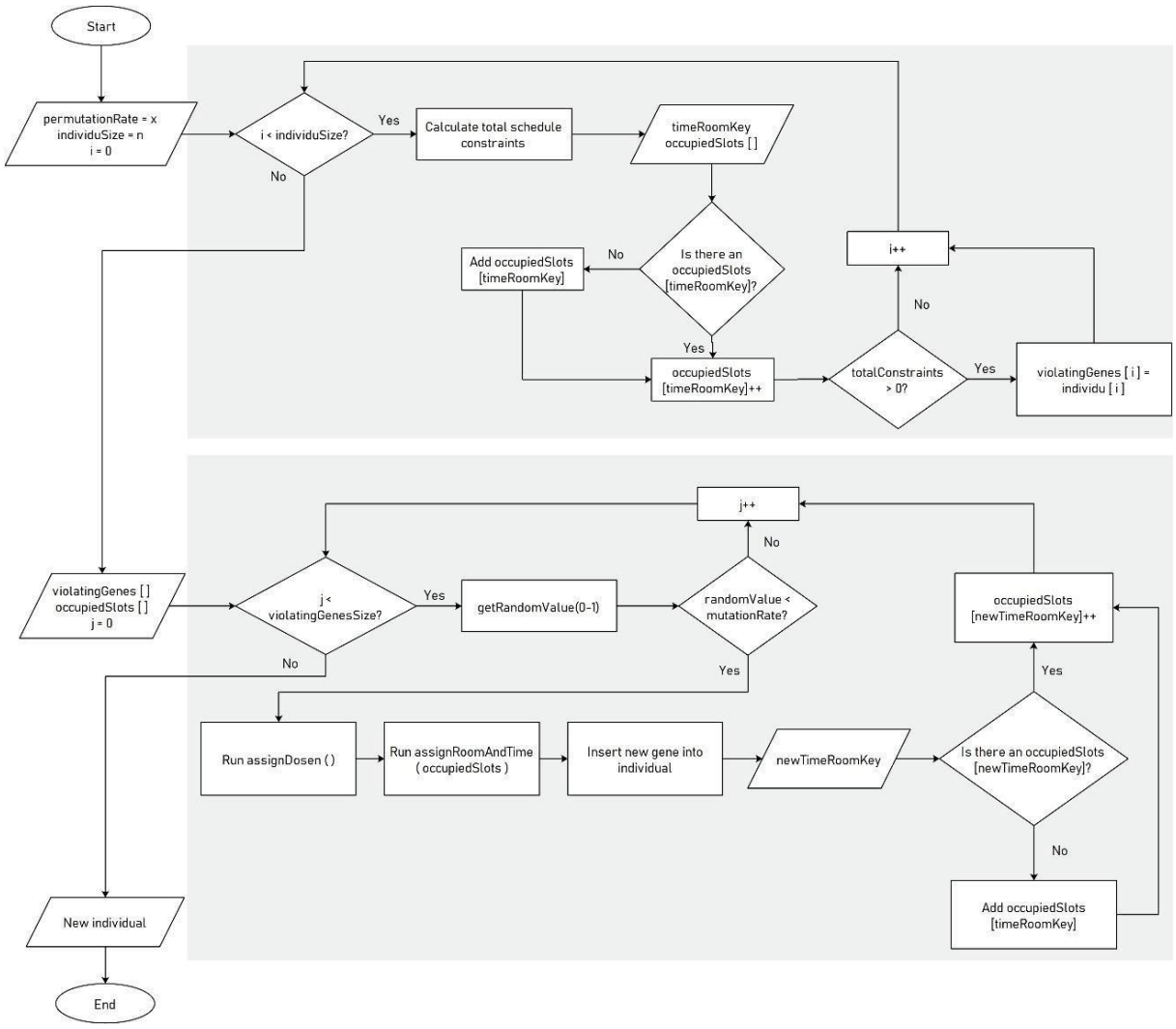

Figure 9. Violated directed mutation flowchart.

2.2.6. Update Generation

The population for the next generation is updated using a generational model with elitism. All members are removed except for the chromosome with the highest fitness from the previous generation. The remaining population is filled with new chromosomes generated through crossover and mutation from offspring. The number of members follows the initial parameters set at the beginning.

2.2.7. Best Chromosome Representation

The best chromosome represents the final solution for the exam scheduling task, containing the highest fitness value from the last generation. Each gene in the chromosome represents a student undertaking the final exam, the room used for the exam, the time the exam is held, and the instructors involved in the exam.

2.3. Testing and Evaluation

Testing in this research was conducted through two schemes: comparing the average fitness values generated by each genetic algorithm (GA) with specific parameters and comparing the time taken to complete the process. The testing involves two types of GAs, basic and modified, and experiments using weights in fitness calculations. The testing stage is carried out with a mutation probability parameter (Pm), where several experiments were conducted to estimate time and the best fitness values obtained.

The experiments are performed three times with varying numbers of generations and different Pm values. The average fitness value for scheduling processes using GA basic and GA modified is recorded to determine which algorithm has the highest average fitness value in each trial. The Pm parameter used is the best value from each algorithm based on experiments with sample data yielding the highest average fitness value. The testing design is presented in Table 5.

Table 5. Testing scheme.

No	Data sample	Max gen	GA basic	GA modified	GA basic (with weight)	GA modified (with weight)
1	Proposal seminar	50	…	…	…	…
		75	…	…	…	…
		100	…	…	…	…
2	Thesis seminar	50	…	…	…	…
		75	…	…	…	…
		100	…	…	…	…
3	All data	50	…	…	…	…
		75	…	…	…	…
		100	…	…	…	…

3. RESULTS AND DISCUSSIONS

3.1. Parameter Determination

The parameters used in this testing process include the maximum number of generations, population size, and mutation probability (Pm) across all GAs. The best PM value is obtained from 10 trials conducted on the GAs developed. The Pm value with the highest fitness-to-time ratio will be used in the subsequent research stages. Testing will use data from Master's thesis students at the Faculty of Computer Science, Gadjah Mada University, for the year 2023. The data is divided into three sets based on the type of exam the students will undertake. To determine the best PM value, data from students who will conduct Thesis Seminars will be used. Testing will be conducted with Pm values ranging from 0.01 to 0.1. The results of trials for determining the best Pm value are presented in Table 6.

Table 6. Finding the best Pm parameters.

Nilai Pm	Fitness / Time			
	GA basic	GA modified	GA basic (with weight)	GA modified (with weight)
0.01	0.000272	0.000651	0.001338	0.003483
0.02	0.000269	0.000759	0.001375	0.003193
0.03	0.000657	0.001748	0.003145	0.007122
0.04	0.000701	0.001669	0.003182	0.007424
0.05	0.000654	0.001632	0.002995	0.006910
0.06	0.000654	0.001552	0.002871	0.007844
0.07	0.000730	0.001781	0.003274	0.008409
0.08	0.000701	0.001919	0.003201	0.007640
0.09	0.000710	0.001742	0.003237	0.008268
0.10	0.000668	0.001835	0.003588	0.007579

From the data above, the Pm value with the highest ratio between fitness and time is 0.07. Where Pm equals 0.07, the highest ratio was obtained in both Basic GA and Modified GA with weighting. With the Pm parameters already obtained, 3 experiments will be conducted for each GA. These experiments will use data from students who will conduct theses seminars, with a population size of 15 individuals per generation. Other parameters will be adjusted according to the testing scheme. The details of the parameters used in each GA experiment are as follows:

a. Experiment 1: Proposal exam dataset with a maximum of 50, 75, and 100 generations.
b. Experiment 2: Thesis Seminar exam dataset with a maximum of 50, 75, and 100 generations.
c. Experiment 3: All exam datasets with a maximum of 50, 75, and 100 generations.

The results from all experiments will be selected based on the maximum generation with the highest fitness-to-time ratio for each dataset. The results of the experiments can be found in the attached pages.

3.2. Comparison of Research Results

The experiments were conducted using student datasets based on the type of examination and with predetermined parameters. The results of experiments using GA are displayed in bar charts showing the fitness-to-time ratio and generations. For each generation, the results from each dataset are displayed. Figure 10 shows a bar chart containing the fitness-to-time ratio with time using Basic GA and different maximum generations.

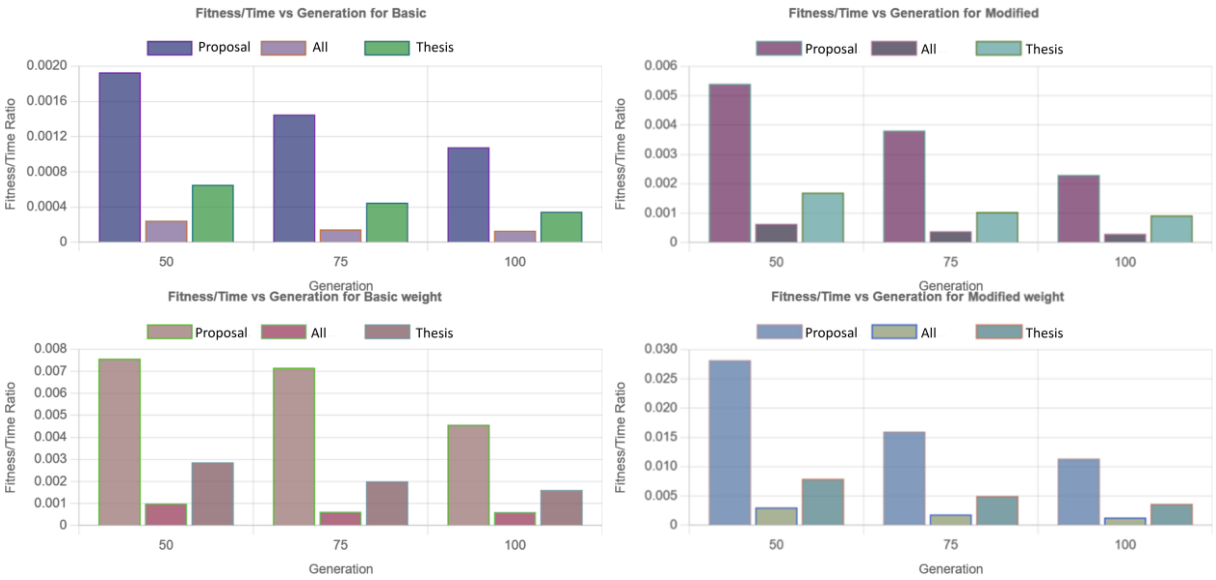

Figure 10. Experiment results.

For all datasets and GAs, the fitness-to-time ratio at a maximum of 50 generations is the highest. This indicates that in every GA used, the maximum generation achieves the best fitness outcome, and the fastest time is at 50. Therefore, the smaller the maximum generation, the higher the fitness-to-time ratio achieved. The results of the research in complete table form obtained from the conducted experiments can be found in the attached pages.

3.3. Best Results

After determining the optimal and best maximum generations for each GA on the dataset, the best values achieved with the same parameters will be compared. The experimental results are presented in Table 5.

Based on the three experiments conducted earlier, the results for each experiment using the same parameters—maximum optimal generation of 50 and mutation rate of 0.07—can determine which GA has the most optimal ratio. In each dataset experiment, the GA modified with weighting achieved the highest ratio compared to other GAs. Although the best fitness values between GA basic and modified are not significantly different, when compared to execution time, GA modified shows a lower value, indicating that the execution time is faster.

In the results from both GA basic and GA modified, the highest ratio between fitness and time is achieved by GA modified. The fitness values are not drastically different, which suggests that the resulting exam schedules have a similar structure, leading to almost identical best fitness values. However, the proposed Modified GA takes less time than basic GA, making it faster.

From the above results, it can be concluded that the modified GA or modified GA can produce exam schedules with the same fitness values but in a shorter time. This is influenced by the differences in selection and mutation techniques used in each GA. Violated Directed Mutation in GA modified

creates more structured and non-repetitive data mutations, thereby reducing the execution time required.

Table 5. Best results of each GA.

Dataset	Algen	Max gen optimal	Best fitness	Execution time	F/W
Proposal	GA basic	50	0.00433	2.24425	0.001929
	GA modified	50	0.00437	0.80884	0.005403
	GA basic (with weight)	50	0.01955	2.58341	0.007568
	GA modified (with weight)	50	0.02125	0.75436	0.028170
Thesis	GA basic	50	0.00216	5.60440	0.000385
	GA modified	50	0.00437	1.26915	0.003443
	GA basic (with weight)	50	0.01011	3.53393	0.002861
	GA modified (with weight)	50	0.00999	1.26192	0.007917
All	GA basic	50	0.00137	3.30269	0.000415
	GA modified	50	0.00134	2.09186	0.000641
	GA basic (with weight)	50	0.00594	5.89040	0.001008
	GA modified (with weight)	50	0.00601	1.98994	0.003020

4. CONCLUSION

Based on the analysis of the modified Genetic Algorithm using Tournament Selection and Violated Directed Mutation, it is evident that the modified approach provides efficient solutions for exam scheduling with minimal conflicts. The modified GA maintains high fitness values comparable to the basic GA while significantly reducing execution time. Violated Directed Mutation and Tournament Selection enhances the selection process, ensuring faster convergence towards optimal solutions. Furthermore, combining these techniques results in a structured mutational process that minimizes unnecessary iterations. Overall, the modified GA is a more efficient and effective method for handling complex scheduling problems, offering high-quality solutions and reduced processing time. For future research, enhancing the Violated Directed Mutation method and exploring hybrid algorithms could optimize performance and flexibility in solving intricate scheduling challenges.

REFERENCES

[1] Amrulloh, A. & Sela, E. I. (2021). Optimasi proses penjadwalan mata kuliah menggunakan algoritme genetika dan pencarian tabu. *Jurnal Teknologi dan Sistem Komputer*, **9**(3), 157–166.

[2] Dener, M. & Calp, M. H. (2018). Solving the exam scheduling problems in central exams with genetic algorithms. *Mugla Journal of Science and Technology*, **4**(1), 102–115.

[3] Putra, D. M. D. U. & Subanar, S. (2011). Penerapan Algoritma Genetika Untuk Menyelesaikan Permasalahan Penjadwalan Perawat Dengan Fuzzy Fitness Function. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, **6**(2).

[4] Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, **80**, 8091–8126.

[5] Iskandar, A. P. S. (2021). Optimasi Penjadwalan Ujian Tugas Akhir Dengan Menggunakan Algoritma Genetika. *Journal of Computer Science and Informatics Engineering (J-Cosine)*, **5**(1), 40–48.

[6] Kristiadi, D. & Hartanto, R. (2019). Genetic Algorithm for lecturing schedule optimization. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, **13**(1), 83–94.

[7] Kurniati, N. I., Rahmatulloh, A., & Rahmawati, D. (2019). Perbandingan performa algoritma koloni semut dengan algoritma genetika–tabu search dalam penjadwalan kuliah. *Comput. Eng. Sci. Syst. J*, **4**(1), 17.

[8] Marte, M. (2002). *Models and algorithms for school timetabling: A constraint programming approach*. Doctoral dissertation, München, Univ., Diss.

[9] Sari, Y., Alkaff, M., Wijaya, E. S., Soraya, S., & Kartikasari, D. P. (2019). Optimasi Penjadwalan Mata Kuliah Menggunakan Metode Algoritma Genetika dengan Teknik Tournament Selection. *J. Teknol. Inf. dan Ilmu Komput*, **6**(1), 85.

[10] Shima, Y., Kadir, R. A., & Ali, F. H. (2021). A novel approach to the optimization of a public bus schedule using k-means and a genetic algorithm. *IEEE Access*, **9**, 73365–73376.
[11] Suyanto. (2010). An informed genetic algorithm for university course and student timetabling problems. *International Conference on Artificial Intelligence and Soft Computing*, 229–236.
[12] Swari, M. H. P., Putra, C. A., & Handika, I. P. S. (2022). Analisis perbandingan algoritma genetika dan modified improved Particle Swarm Optimization dalam penjadwalan mata kuliah. *Jurnal Nasional Pendidikan Teknik Informatika: JANAPATI*, **11**(2), 92–101.
[13] Tayyab, A. & Saif, U. (2022). A two-stage genetic artificial bee colony algorithm for solving integrated operating room planning and scheduling problem with capacity constraints of downstream wards. *IEEE Access*, **10**, 131109–131127.
[14] Yadav, S. L. & Sohal, A. (2017). Study of the various selection techniques in genetic algorithms. *International Journal of Engineering, Science and Mathematics*, **6**(3), 198–204.
[15] Zhou, Z., Li, F., Zhu, H., Xie, H., Abawajy, J. H., & Chowdhury, M. U. (2020). An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. *Neural Computing and Applications*, **32**, 1531–1541.
[16] Squires, M., Tao, X., Elangovan, S., Gururajan, R., Zhou, X., & Acharya, U. R. (2022). A novel genetic algorithm based system for the scheduling of medical treatments. *Expert Systems with Applications*, **195**, 116464.
[17] Abualigah, L. & Alkhrabsheh, M. (2022). Amended hybrid multi-verse optimizer with genetic algorithm for solving task scheduling problem in cloud computing. *The Journal of Supercomputing*, **78**(1), 740–765.
[18] Harada, T. & Alba, E. (2020). Parallel genetic algorithms: a useful survey. *ACM Computing Surveys (CSUR)*, **53**(4), 1–39.
[19] Mehanović, D., Kečo, D., Kevrić, J., Jukić, S., Miljković, A., & Mašetić, Z. (2021). Feature selection using cloud-based parallel genetic algorithm for intrusion detection data classification. *Neural Computing and Applications*, **33**, 11861–11873.
[20] Alhijawi, B. & Awajan, A. (2024). Genetic algorithms: Theory, genetic operators, solutions, and applications. *Evolutionary Intelligence*, **17**(3), 1245–1256.
[21] Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, **80**, 8091–8126.
[22] Xue, Y., Zhu, H., Liang, J., & Słowik, A. (2021). Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification. *Knowledge-Based Systems*, **227**, 107218.
[23] Hussain, A. & Muhammad, Y. S. (2020). Trade-off between exploration and exploitation with genetic algorithm using a novel selection operator. *Complex and Intelligent Systems*, **6**(1), 1–14.
[24] Zhou, Z., Li, F., Zhu, H., Xie, H., Abawajy, J. H., & Chowdhury, M. U. (2020). An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. *Neural Computing and Applications*, **32**, 1531–1541.
[25] Shang, L., Shang, Y., Hu, L., & Li, J. (2020). Performance of genetic algorithms with different selection operators for solving short-term optimized reservoir scheduling problem. *Soft Computing*, **24**(9), 6771–6785.