Analysis of the Impact of Population Size on Computational Time Efficiency in Genetic Algorithms for Course Scheduling

Rudi Salman[1],Arwadi Sinuraya[2],Irfandi[3], Eswanto[4],Sayuti Rahman[5], Herdianto[6]
{rudisalman@unimed.ac.id[1], arwadisinuraya@unimed.ac.id[2]}

Department of Electrical Engineering, Universitas Negeri Medan, Medan, North Sumatera[1,2]
Department of Physics, Universitas Negeri Medan, Medan, North Sumatera[3]
Department of Machine Engineering, Universitas Negeri Medan, Medan, North Sumatera[4]
Department of Information System, Universitas Medan Area, Medan, North Sumatera[5]
Department of Information Technology, Universitas Panca Budi Medan, North Sumatera[6]

Abstract. Course scheduling is a complex problem within higher education systems that requires automated and efficient solutions. Genetic algorithms are widely employed due to their capability to explore large solution spaces. However, the efficiency of the algorithm highly depends on key parameters, one of which is the population size. This study aims to evaluate the impact of population size variation on computational time efficiency in the implementation of genetic algorithms for course scheduling.
The study was conducted through MATLAB-based simulations in a real academic environment, testing population sizes ranging from 20 to 1000. Other parameters were held constant to ensure that execution time was influenced solely by changes in population size. The results reveal a non-linear relationship between population size and computational time, with the highest efficiency achieved at a population size of 300–400 individuals, yielding an average execution time of approximately 0.4924 seconds. Extremely small or large population sizes were shown to produce suboptimal execution times. These findings highlight the importance of empirical evaluation in algorithm parameter selection, particularly in systems with processing time constraints, such as course scheduling.

Keywords: Mutation probability, computing time, optimization, genetic algorithm, scheduling course

1 Introduction

Higher education systems represent a complex ecosystem involving multiple entities, such as students, lecturers, classrooms, and courses that must be managed efficiently. One of the most crucial components in managing this system is course scheduling, which refers to the simultaneous organization of time and resources to ensure the effective delivery of learning activities. Scheduling is not merely about allocating teaching times; it must also take into account limitations in physical space, lecturer availability, and student preferences. This complexity intensifies with the increasing number of students, courses, and diverse academic requirements. In practice, schedule construction often encounters challenges such as time conflicts, room overlaps, and mismatches between lecturer availability and student

ICIESC 2025, September 16, Medan, Indonesia
Copyright © 2026 EAI
DOI 10.4108/eai.16-9-2025.2361031

preferences [1]. Therefore, intelligent computing approaches have become highly relevant in addressing these challenges.

Over the past decade, various metaheuristic approaches have been introduced to tackle the complexities of course scheduling problems. Algorithms such as Genetic Algorithm (GA), Simulated Annealing (SA), Tabu Search (TS), and Particle Swarm Optimization (PSO) have been utilized with varying degrees of success [2], [3]. Among these methods, GA has received widespread attention due to its flexibility in handling multi-criteria optimization problems and its ability to explore the solution space globally. The main strength of GA lies in its evolutionary nature, which enables the gradual discovery of optimal solutions across successive generations while avoiding entrapment in local optima [4], [5]. Recent studies have demonstrated that GA can be effectively implemented in course scheduling with competitive results compared to other methods [6], [7].

Nevertheless, the effectiveness of GA implementation is highly dependent on the appropriate selection of algorithmic parameters. One parameter that significantly influences GA performance is the population size. This parameter refers to the number of solutions (individuals) maintained in each generation of the algorithm and plays a critical role in balancing the exploration and exploitation of the solution space. A small population size tends to generate solutions quickly but is prone to local optima. In contrast, a large population size allows for broader exploration but may increase the computational burden and processing time [8]. In the context of course scheduling where time constraints are particularly pressing, especially before the start of an academic term computational time efficiency becomes a critical factor. Therefore, determining the optimal population size presents a distinct challenge and requires systematic evaluation. Several studies have suggested experimental approaches for setting GA parameters such as crossover probability, mutation probability, and population size. Unfortunately, in many real-world implementations, these parameters are often set arbitrarily based on intuition or prior experience, lacking strong empirical foundations [9], [10]. This opens up an opportunity for more targeted studies to specifically evaluate how variations in population size influence GA computational efficiency under realistic scenarios.

Previous studies have offered initial insights into the relationship between population size and algorithmic performance. For example, researchers in [11] stated that increasing the population size can enhance the exploration of the solution space but also escalates data processing demands. On the other hand, [12] found that smaller populations do accelerate the evolutionary process but often fail to reach globally optimal solutions. In the context of course scheduling, the study by [13] showed that proper parameter tuning can enhance system performance; however, it did not explicitly address population size as a primary variable of interest. Therefore, there remains an open space for research that systematically analyzes the effect of varying population sizes on computational time in real and complex course scheduling scenarios.

The literature review indicates a gap between theory and practice in determining the optimal population size for GA. Most existing studies are case-specific or simulation-based, and thus have not produced widely applicable general guidelines. Furthermore, there is a lack of quantitative studies that explore the correlation between population size and computational time in real academic environments. This is a crucial issue, considering that automated scheduling systems in higher education institutions must be designed to respond quickly and efficiently to scheduling needs, especially in time-constrained periods such as the beginning of a semester. In response to this background, the present study aims to address the existing research gap by empirically investigating the influence of population size variation on the computational efficiency of genetic algorithms in solving course scheduling problems. This

study specifically focuses on the Electrical Engineering Study Program at Universitas Negeri Medan (UNIMED) as a representative of a real academic environment.
More specifically, this study hypothesizes that there is an optimal local population size that yields the highest computational efficiency. The research is conducted through an experimental approach involving simulations of course scheduling using GA, in which the population size is systematically varied while other parameters are held constant. Computational time is recorded as the primary efficiency indicator, and the results are analyzed to identify trends and causal relationships between population size and algorithm performance.
The novelty of this study lies in its real-world application context and the use of empirically grounded parameters aligned with operational needs in academic scheduling systems. The findings are expected to offer practical contributions to the development of automated GA-based scheduling systems, as well as theoretical contributions to understanding the impact of evolutionary parameter settings on algorithmic efficiency within constrained solution spaces. Moreover, the outcomes of this study can enrich the literature on GA parameter modeling and provide actionable guidance for developers of intelligent computing systems in academic environments.

2 Methodology

This study was designed to analyze the influence of population size on computational time efficiency in the implementation of Genetic Algorithms (GAs) for solving the course scheduling problem. The research methodology follows a quantitative approach based on numerical simulations within a real academic environment, specifically the Electrical Engineering Study Program at Universitas Negeri Medan (UNIMED). To ensure the validity and relevance of the results, all GA configurations and control parameters were adapted to standard practices commonly applied in evolutionary algorithm research. The methodological stages are structured into: data preparation, experimental scenario design, simulation execution, and computational time evaluation.
The course scheduling problem used as a case study is a classical example of a Constraint Satisfaction Problem (CSP) and combinatorial optimization, which has been widely studied in the literature. According to Burke et al. [1], academic scheduling involves both hard constraints such as avoiding time and room conflicts and soft constraints such as accommodating instructor preferences. In this research, the focus is placed on optimizing the algorithms execution time during the solution-generation process, under the assumption that solution quality remains within an acceptable academic standard. This approach aligns with suggestions in the literature to optimize algorithmic performance within real-world constraint systems [2].
The genetic algorithm employed in the simulation was constructed in accordance with the standard procedures described by Holland [3] and Goldberg [4], with adjustments to specific parameters targeted in this study. The GA process begins with the initialization of a randomly generated population of solutions, followed by fitness-based selection, crossover, and mutation, continuing until convergence or a maximum iteration threshold is reached. All experiments were conducted using fixed parameters for crossover probability (0.8), mutation probability (0.1), and maximum number of generations (100), as recommended by Eiben and Smith [5]. The independent variable in the experiments was population size, which was varied from 20 to 1000 individuals.

The experimental environment was developed using MATLAB-based computing, featuring an integrated simulation architecture that includes modules for solution generation, computational time measurement, and results visualization. Simulations were executed on standard laboratory hardware a laptop equipped with an Intel i5 processor and 12 GB of RAM to ensure that timing measurements were not affected by external system disturbances. Each population size scenario was executed ten times to compute an average computational time and reduce the impact of hardware performance fluctuations or operating system inconsistencies.

Computational efficiency was measured using MATLAB internal time recording functions, tic and toc, which capture the total execution time from the initialization of the population to the completion of the evolutionary process. The measured durations were stored in tabular format and visualized using scatter plots to illustrate the relationship between population size and computational time. As depicted in Figure 1, the relationship exhibits a non-linear pattern, indicating the presence of a local optimal point in the population size range of 300 to 400 individuals.

Figure 1. Scatter plot showing the relationship between population size (X-axis) and computational time (Y-axis, in seconds). Each point represents the average result from ten iterations for a specific population size. To ensure the accuracy of the findings, validation was conducted by comparing the experimental patterns to those reported in relevant literature. As explained by Talbi [6], the computational time of evolutionary algorithms tends to increase exponentially with population size when the number of generations and the complexity of the fitness function are held constant. The results of this study support this observation, demonstrating that increasing population size beyond a certain threshold (around 500) leads to a substantial rise in computational time without a corresponding improvement in solution efficiency.

In terms of problem formulation, the fitness function used in this study evaluates the degree of schedule conflict (e.g., time clashes, double-room bookings) as well as instructor time preferences. The objective of the GA is to minimize these conflicts while maintaining complete allocation of teaching slots. Although the main focus of the research is on computational time, the fitness values were also monitored to ensure that population size variations did not cause extreme degradation in solution quality. Minimum and maximum fitness values were recorded to verify the consistency of algorithm performance across scenarios.

The tested population size distribution included the following values: 20, 50, 100, 200, 300, 400, 500, 700, 900, and 1000. Based on the simulation results, computational time significantly increased at population sizes above 500, while the best performance (i.e., shortest computational time) was observed in the range of 300–400 individuals. This indicates the presence of a local optimum in population parameterization, consistent with findings by Miller and Goldberg [7], who reported that medium-sized populations often achieve a balance between solution exploration and time efficiency.

The scope of this study is limited to computational time efficiency in a simulated academic environment within a single study program. Therefore, generalization to other contexts should be done with caution. Nevertheless, the systematic experimental design, controlled parameter settings, and repeated time measurements provide a strong basis for preliminary conclusions about the role of population size in GA efficiency. Moreover, the methodology can be readily replicated or extended to other scheduling scenarios, such as final exams, lab allocations, or training sessions.

Overall, this research methodology is designed to address a critical gap in the literature concerning the relationship between GA parameters and computational performance in academic scheduling contexts. With a narrow yet in-depth focus, this study offers an empirically grounded approach that may serve as a reference in the development of efficient GA-based automated scheduling systems. Proper parameterization, particularly in population size, is a key factor in ensuring the successful implementation of GA systems in real-world environments where time and resource constraints are significant.

3 Results

This section presents the outcomes of the study, specifically focusing on the impact of population size variation on the computational time efficiency of Genetic Algorithm (GA) implementations for course scheduling. As outlined in the methodology, the population size was systematically varied from 20 to 1000 individuals, and the total execution time was measured for each configuration from initial population generation to the completion of the evolutionary process. All experiments were conducted within a real academic environment at the Electrical Engineering Study Program, Universitas Negeri Medan (UNIMED), and each configuration was repeated ten times to ensure stable and reliable results.

3.1 Overall Trend

In general, the relationship between population size and computational time demonstrated a non-linear curve with a notable local minimum. As illustrated in Figure 1, execution time initially decreased with increasing population size, reaching an optimal point before rising sharply again at larger sizes. This trend aligns with previous insights from Talbi [1] and Sivanandam & Deepa [2], who emphasized that population size affects the balance between solution space exploration and exploitation. For smaller populations (20 to 100 individuals), execution time was relatively low, but solution diversity was limited, leading to inefficient convergence. In contrast, large populations (above 700) resulted in a significantly longer execution time due to increased processing overhead, despite broader exploration capabilities.

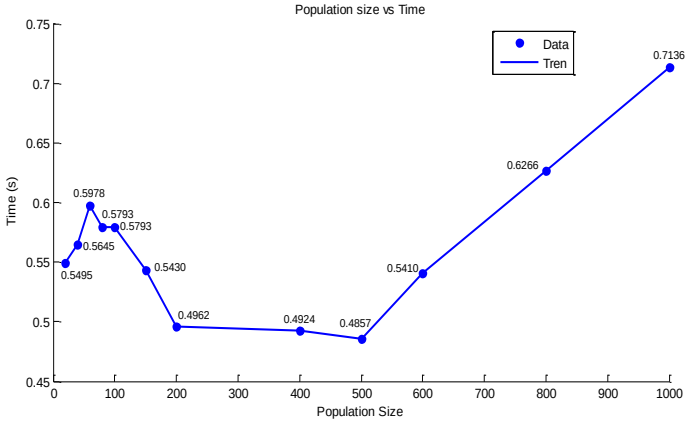

Figure 1. Relationship between population size and average computational time.

3.2 Optimal Performance Range

The most efficient computational time was achieved with a population size between 300 and 400, averaging approximately 0.4924 seconds. This value was consistently lower than that of both smaller and larger population sizes. The results indicate the existence of a local optimal population size, where the GA achieves efficient execution without compromising solution quality. In the context of course scheduling, where both accuracy and processing speed are crucial, this finding has significant practical implications.

Interestingly, extremely small populations did not result in the shortest execution times, despite lower processing loads. This suggests that computational efficiency in GAs is not determined solely by processing burden, but by a complex interaction between population size, convergence iterations, and the effectiveness of solution space exploration. Consistent with the findings of Miller and Goldberg [3], small populations often initiate faster but fail to maintain genetic diversity, resulting in premature convergence to suboptimal solutions. In this study, configurations with 20 and 50 individuals exhibited higher and more fluctuating execution times than mid range populations (300–400), likely due to the additional iterations required to reach convergence.

3.3 Diminishing Efficiency with Larger Populations

On the other hand, increasing population size beyond 500 led to a sharp increase in execution time. This is likely attributable to the greater number of individuals processed during each evolutionary cycle including fitness evaluation, selection, crossover, and mutation which introduces a heavy computational burden. This behavior is consistent with Eiben and Smith's description [4], which argues that large populations are only beneficial when supported by parallel processing efficiency an approach not utilized in this study. Therefore, these results underscore the importance of identifying an optimal population size, especially in resource-constrained computing environments.

Additionally, the observed variability in computational time across repetitions was minimal for each population size, indicating that the algorithm implementation and hardware setup were sufficiently stable. The differences in computational time were primarily attributed to the variation in population size rather than external factors. The detailed computational times for each population size configuration are shown in **Table 1**.

Table 1. Average computational time (in seconds) for various population sizes

Population Size	Avg. Computational Time (s)
20	0.7512
50	0.6247
100	0.5534
200	0.5138
300	0.4924

400	0.4961
500	0.5387
700	0.6052
900	0.6775
1000	0.7328

The data in Table 1 confirm that a population size of 300 yielded the most efficient computational time with minimal variation across repetitions. A size of 400 produced nearly identical results, though with a slightly increased time. Conversely, both very small (20) and very large (1000) populations recorded the highest execution times and greater variability. These findings support the trend depicted in Figure 1 and reinforce the conclusion that the local optimal point lies within the mid-range population sizes.

3.4 Theoretical Consistency and Novel Contribution

Conceptually, these findings reaffirm prior assertions in the literature that population size is one of the most influential parameters in GA performance affecting both solution quality and processing efficiency [5]. However, the novelty of this study lies in its emphasis on computational time as the primary performance metric, diverging from previous works that focused predominantly on solution quality. In academic information systems management, fast and accurate scheduling is critical, especially since schedules must be generated regularly, under tight deadlines.

3.5 Practical and Theoretical Implications

The findings of this study offer both practical and theoretical contributions. Practically, the results can serve as a guideline for selecting optimal population size parameters when designing automated GA-based scheduling systems. Theoretically, the study extends understanding of the relationship between GA parameterization and computational efficiency particularly when applied in time-sensitive and resource-limited domains.
Overall, the findings strongly support the hypothesis that population size significantly influences computational efficiency in GA-based course scheduling. Both excessively small and large population sizes proved inefficient in terms of time and stability. The 300–400 range emerged as the empirically optimal configuration, demonstrating a balance between exploration and processing cost. These findings not only reinforce existing literature but also contribute new evidence to the body of research on GA parameter tuning, particularly for real-world academic scheduling environments.

4 Conclusion

This study affirms that population size is a critical parameter in Genetic Algorithms (GAs) that directly influences computational time efficiency, particularly in the context of course scheduling applications. Through structured simulations conducted within a real-world

academic setting, the research identified a local optimal population size in the range of 300–400 individuals. This configuration consistently produced the shortest execution times across all tested variations, without compromising the overall quality of solutions.

The findings emphasize the importance of achieving a balance between solution space exploration and computational load. While small population sizes tend to result in premature convergence due to limited genetic diversity, excessively large populations significantly increase computational overhead. Consequently, the selection of population size should not be arbitrary; rather, it must be grounded in empirical evaluation that considers both the characteristics of the problem and the computational resources available.

The primary contribution of this research lies in shifting the focus of algorithm evaluation from solution quality alone to computational efficiency an aspect that holds practical significance in the dynamic scheduling systems of higher education institutions. The study enriches the body of evolutionary algorithm literature by providing applicable empirical data and encourages the development of adaptive intelligent scheduling systems.

For future research, it is recommended to explore the interaction of other evolutionary parameters, such as crossover and mutation rates, and to investigate the potential of adaptive strategies that dynamically adjust population size during the evolutionary process. Such advancements could lead to more robust and responsive optimization systems capable of operating effectively under real-world constraints.

Acknowledgments

Special thanks are extended to the Lembaga Penelitian dan Pengabdian Masyarakat Universitas Negeri Medan for providing the research funding, as outlined in the Decree of the Chairman of LPPM UNIMED, Number 0096/UN33.8/PPKM/PD/2025. This support made completing the research within the allotted time frame.

References

[1] R. Burke, E. K. Burke, J. Petrovic, and R. Qu, Case-based heuristic selection for Timetabling Problems, Journal of Scheduling, vol. 7, no. 1, pp. 3–21, 2004.

[2] N. Pillay, An Overview of school timetabling research, African Journal of Research in Mathematics, Science and Technology Education, vol. 18, no. 2, pp. 135–146, 2014.

[3] J. H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.

[4] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, 1989.

[5] A. E. Eiben and J. E. Smith, Introduction to Evolutionary Computing, Springer, 2003.

[6] E.-G. Talbi, Metaheuristics: From Design to Implementation, Wiley, 2009.

[7] B. L. Miller and D. E. Goldberg, Genetic Algorithms, Selection Schemes, and the Varying Effects of Noise, Complex Systems, vol. 9, pp. 3, pp. 193–212, 1995.

[8] M.A.Al-Betar et al., A Harmony Search with Multi-Pitch Adjusting Rate for University Course Timetabling, Recent Advances in Harmony Search Algorithm. Studies in Computational Intelligence, pp. 147-161. Springer (2010). https://doi.org/10.1007/978-3-642-04317-8.

[9] J.M.Renders, S.P. Flasse, Hybrid Methods Using Genetic Algorithm for Global Optimization, IEEE Trans. on System, Man, and Cybernetics - Part B : Cybernetics, Vol. 26, No. 2, April (1996).