Elite-Refined Genetic Algorithm with Hill Climbing Local Search for University Course Scheduling

Heru Purnomo Kurniawan
Informatika
Universitas Islam Negeri Siber Syekh Nurjati Cirebon
Cirebon, Indonesia
herupurnomo@uinssc.ac.id

Lia Farhatuaini
Informatika
Universitas Islam Negeri Siber Syekh Nurjati Cirebon
Cirebon, Indonesia
farhatuaini@uinssc.ac.id

Nurul Bahiyah
Informatika
Universitas Islam Negeri Siber Syekh Nurjati Cirebon
Cirebon, Indonesia
nurulbahiyah@uinssc.ac.id

Ardi Susanto
Informatika
Universitas Islam Negeri Siber Syekh Nurjati Cirebon
Cirebon, Indonesia
ardisusanto@uinssc.ac.id

Muhammad Iszul Wilsa
Informatika
Universitas Islam Negeri Siber Syekh Nurjati Cirebon
Cirebon, Indonesia
iszulwilsa@uinssc.ac.id

Gina Khayatun Nufus
Informatika
Universitas Islam Negeri Siber Syekh Nurjati Cirebon
Cirebon, Indonesia
ginakhn@uinssc.ac.id

Abstract— This paper proposes a hybrid Genetic Algorithm with Hill Climbing (GA-HC) for the university course timetabling problem. The approach integrates GA's global exploration capability with generational Hill Climbing refinement applied to elite individuals using Lamarckian learning, aiming to minimize both hard and soft constraint violations. Using a real-world dataset comprising 56 course sections, 18 lecturers, three rooms (one lecture room and two computer laboratories), and 11 daily time slots across five working days, the hybrid GA-HC consistently outperforms standalone GA and HC in terms of convergence speed, solution quality, and stability across multiple runs. Parameter tuning further reveals that moderate mutation rates combined with limited HC iterations per generation provide the best trade-off between computational cost and timetable quality. The proposed framework demonstrates superior robustness and effectiveness for complex, resource-constrained academic scheduling problems.

Keywords— Genetic Algorithm, Hill Climbing, Hybrid Optimization, Course Scheduling, Metaheuristic, Timetabling

I. INTRODUCTION

Course scheduling is one of the fundamental and critical components in academic management at higher education institutions, as it affects multiple operational aspects across the campus [1], [2], [3]. A well-structured timetable not only ensures smooth and orderly execution of teaching and learning activities but also plays a key role in optimizing the use of limited resources such as classrooms, lecture times, and teaching staff with specific capacities and availabilities [4], [5]. Ineffective scheduling may lead to various issues, including course conflicts, wasted time for lecturers and students, and inefficient use of facilities, which can ultimately degrade the quality of academic services [6], [7], [8].

These challenges are particularly pronounced in resource-constrained university environments where rapid enrollment growth, a high proportion of practical (laboratory-based) courses, and limited room availability significantly increase scheduling complexity. The program must accommodate theoretical courses, practical laboratory sessions, and project-based classes, each with its own room-type and capacity constraints [9]. In addition, scheduling must incorporate lecturer availability, prerequisite relationships between courses, and policy-based rules related to course prioritization and semester progression [10]. Because these constraints interact in non-trivial ways, modifying the placement of a single course may propagate conflicts across multiple days or rooms. Such interdependency has been widely reported in literature as a characteristic of multi-constraint academic timetabling, where the complexity quickly exceeds what can be handled through manual or rule-based strategies [11], [12].

The course timetabling problem is known to be NP-Hard [13], meaning that no deterministic algorithm can solve large-scale instances optimally within polynomial time. Consequently, heuristic and metaheuristic methods are often adopted to generate near-optimal solutions [14]. Among them, the Genetic Algorithm (GA) [15] is widely recognized for its strong global exploration ability, as it simulates natural evolutionary processes such as selection, crossover, and mutation [16]. GAs have been successfully applied to various scheduling problems, including course timetabling, curriculum scheduling and exam scheduling [17], [18], [19]. However, GAs sometimes suffer from premature convergence and the risk of getting trapped in local optima, particularly when lacking an effective local improvement mechanism [20]. To address this limitation, this study enhances the standard GA with a Hill Climbing (HC) [21] procedure that performs local refinement on selected best individuals in each generation.

Previous research has successfully applied hybrid approaches that combine genetic algorithms with local search or hill-climbing techniques. For instance, study in [22] investigated a hybrid GA-HC model for optimizing the hyper-parameters of convolutional neural networks and demonstrated faster convergence and higher accuracy compared to either method alone. Similarly, [23] proposed a hybrid meta-heuristic for large-scale course timetabling, combining adaptive local search and instance decomposition, and obtained significant reductions in constraint violations and solution time on real-world university datasets. Within the domain of university course scheduling itself, [24] developed a hybrid improved parallel GA with local search (IPGALS) that achieved superior fitness values on benchmark timetabling instances compared to standard GA. More recently, [25] applied a GA-HC hybrid to a university timetable scheduling problem and showed that the hybrid outperformed pure GA and pure HC in terms of both feasibility and runtime.

Variants of hybrid evolutionary methods also incorporate advanced local improvement strategies such as Simulated Annealing, Variable Neighborhood Search, or tabu-based enhancements, further demonstrating the advantage of combining global exploration with local refinement [26], [27], [28]. Another key development in hybrid evolutionary computation is the adoption of Lamarckian learning, where improvements discovered during local search overwrite the original chromosome, allowing refined structural features to persist across generations and accelerating convergence [29], [30]. In our study, we similarly employ an initial global search using a GA framework, followed by a HC-based local refinement step on the top individuals each generation, with replacement of refined chromosomes (Lamarckian update) [31]. The key distinction of our method lies in the integration of a generational HC refinement applied to the elite subset at every iteration, whereas previous studies either refine offspring only occasionally or apply local search as a post-processing step rather than in-loop per generation.

Given the limitations observed in previous studies, this research aims to enhance the balance between exploration and exploitation in the optimization of university course timetabling by proposing a hybrid Genetic Algorithm with Hill Climbing refinement (GA-HC). The proposed model leverages the global search capability of the GA to explore diverse scheduling configurations while employing the HC mechanism as a local improvement operator to refine high-quality solutions during each generation. This generational hybridization strategy is expected to accelerate convergence toward feasible timetables and reduce both hard and soft constraint violations more effectively than traditional GA or post-hoc local search methods. By systematically integrating global and local optimization within a unified evolutionary framework, this study contributes to the growing body of research on hybrid metaheuristics for educational timetabling, offering a robust and adaptable approach for complex academic scheduling scenarios.

II. PROPOSED METHOD

The proposed method combines the global exploration ability of the Genetic Algorithm (GA) with the local exploitation capability of Hill Climbing (HC) to create a hybrid GA-HC framework for course timetabling. This integration allows the algorithm to maintain diversity within the population while refining high-quality individuals through local improvement at each generation [32], [33]. The overall objective is to minimize the total penalty score derived from both hard and soft constraint violations.

A. Scheduling Data

The scheduling data used in this study consist of several main components that represent the real-world elements of the Informatics Study Program's course timetable at Universitas Islam Negeri Siber Syekh Nurjati Cirebon. These components are courses, lecturers, rooms, and time slots, each of which has specific attributes and constraints that must be satisfied during the scheduling process. Each course is identified by a unique code, course name, number of credit hours (SKS), and course type (theoretical or practical). Practical courses require laboratory facilities, while theoretical courses are held in standard lecture rooms. Each lecturer has assigned teaching preferences that include available days and preferred teaching hours. These preferences are treated as soft constraints to maintain schedule flexibility while respecting instructor comfort. Each room has a specific capacity and type, categorized as either a lecture room or a laboratory room, ensuring that class assignments comply with facility requirements. Finally, the time slots represent available teaching hours for each working day, ranging from Monday to Friday, with predefined start times corresponding to the university's class schedule.

The dataset used in this experiment includes 56 course classes taught by 18 lecturers, utilizing three rooms: one lecture room (Ruang 1) and two computer laboratories (Lab Informatika 1 and Lab Informatika 2). Each with a capacity of 30 students. The schedule spans five working days (Monday–Friday), with 11 available time slots per day starting from 07:30 to 16:00. These parameters reflect the actual academic scheduling conditions of the Informatics Study Program, ensuring that the optimization scenario closely represents a real institutional context. The scheduling data is subject to both hard and soft constraints. Hard constraints include conflicts such as assigning a lecturer or room to multiple classes at the same time, scheduling practical courses in lecture rooms (or vice versa), and exceeding available time slots within a day. Soft constraints include assigning classes outside a lecturer's preferred days or times and distributing lecturer's courses across too many separate days. These components form the basis of the optimization process, where each complete schedule is constructed from a combination of all possible course, lecturer, room, and time slot assignments. In the next stage, these scheduling elements are encoded into a chromosome structure so that they can be processed using the Genetic Algorithm's evolutionary operators.

B. Chromosome Representation

In the proposed hybrid GA–HC framework, each candidate timetable is encoded as a chromosome, while every gene within the chromosome represents a single class assignment. A class assignment is defined as a tuple of elements (c, r, d, t), where c denotes the course, r the assigned room, d the teaching day, and t the starting time slot. This encoding reflects the structure of the scheduling data and allows the algorithm to evaluate and modify timetables using standard evolutionary operations such as crossover and mutation.

Each chromosome thus represents a complete feasible timetable covering all courses in the dataset. The number of genes in one chromosome equals the total number of courses (56 in this study). During initialization, every course is assigned randomly to a compatible room, day, and time slot, subject to room-type compatibility: practical courses

are restricted to laboratory rooms, while theoretical courses are placed in lecture rooms. This ensures that even at the initial stage, the algorithm produces syntactically valid solutions, though they may still violate soft and hard constraints that will be handled during fitness evaluation and optimization.

The representation is designed to be both compact and expressive, enabling efficient manipulation while preserving schedule consistency. For example, the gene:

$$(IF102.B, LabKomputer2, Tuesday, 08{:}20)$$

represents the class "Algoritma dan Pemrograman I (Group B)" scheduled on Tuesday at 08:20 AM in Lab Komputer 2. Through genetic operations, these assignments are modified to explore alternative combinations that may reduce overall penalty and improve schedule quality. Formally, a chromosome S can be represented as:

$$S = [g_1, g_2, \dots, g_n]$$

where each gene $g_i = (c_i, r_i, d_i, t_i)$ corresponds to the scheduling decision for course i, and n is the total number of courses. This representation aligns directly with the internal data model implemented in the algorithm, allowing seamless interaction between the Genetic Algorithm and the Hill Climbing refinement process described in the following sections.

C. Fitness Function

The fitness function evaluates the quality of each chromosome by quantifying the level of constraint violations in the generated timetable [34]. The objective of the optimization process is to minimize the total penalty, where each penalty value represents a violation of hard or soft constraints that occur in the schedule.

Each chromosome (schedule) is decoded into its constituent class assignments, and several constraint checks are performed for every assignment to ensure schedule validity. The function assigns higher penalties to severe conflicts (hard constraints) and smaller penalties to less critical violations (soft constraints). The total fitness value $F(S)$ of a schedule S is computed as the sum of all penalties incurred.

a. Hard Constraints

Hard constraints represent violations that make a schedule infeasible and must be strictly avoided. These include conflicts related to lecturer availability, room allocation, and time-slot validity. The hard constraints considered in this research, which represent infeasible scheduling conditions that must be avoided by the algorithm, are listed in Table 1.

Table 1. Hard Constraint

No	Constraint Description	Penalty Value
H1	A lecturer is assigned to two classes at the same time	10
H2	A room is assigned to more than one class at the same time	10
H3	A practical course is scheduled in a lecture room (or vice versa)	15
H4	Class duration exceeds available time slots in a single day	5

b. Soft Constraints

Soft constraints are desirable preferences that improve schedule quality but are not mandatory. Violations of these constraints add minor penalties, allowing the algorithm to flexibly explore the solution space. The soft constraints, which influence the quality of feasible solutions and help guide the algorithm toward more balanced and preference-aware timetables, are presented in Table 2.

Table 2. Soft Constraint

No	Constraint Description	Penalty Value
S1	The class is scheduled on a day not preferred by the lecturer	1
S2	The class is scheduled at a time not preferred by the lecturer	1
S3	A lecturer teaches on more than one separate day	2

The inclusion of soft constraints allows the optimization process to balance between feasibility and comfort, leading to schedules that are both valid and practical in real-world academic contexts.

c. Fitness Evaluation Formula
For each chromosome S, the total penalty is computed as:

$$F(S) = \sum_{i=0}^{n} \left(\sum_{j=1}^{m} P_{ij} \right)$$

where n is total number of class assignments, m is total number of constraint types, and P_{ij} is penalty incurred by the class i for constraint j.
The optimization objective is to minimize the total fitness value:

$$min\ F(S)$$

Lower fitness values indicate schedules with fewer constraint violations and thus higher feasibility and quality. A fitness score of zero represents a perfectly feasible schedule with no conflicts.

d. Integration with the Optimization Process
The fitness function plays a crucial role in both the GA and HC stages of the hybrid model. In GA, it determines selection probabilities and guides crossover and mutation processes toward better solutions. In HC, it serves as a local improvement metric, allowing incremental refinements by accepting only neighbor solutions that reduce the penalty value. This unified fitness mechanism ensures consistent evaluation criteria across both global (GA) and local (HC) optimization phases.

D. Genetic Algorithms Process

The Genetic Algorithm (GA) serves as the global search mechanism in the proposed hybrid optimization framework. Inspired by the principles of natural selection, GA explores a large and complex solution space through a population-based stochastic process. Each individual in the population represents a complete timetable encoded as a chromosome, whose fitness value reflects the degree of constraint satisfaction as defined in the fitness evaluation function.

The algorithm iteratively improves the population through selection, crossover, mutation, and elitism until a termination criterion is met. The overall GA procedure in this study consists of the following steps:

1. Initialization
   An initial population $P_0 = \{S_1, S_2, \ldots, S_N\}$ of N chromosomes is generated randomly. Each chromosome represents a feasible but potentially suboptimal schedule, constructed by assigning every course to a valid day, time slot, and room that matches its type (lecture or laboratory). This random initialization ensures diversity of solutions, allowing the search to start from multiple regions of the solution space and reducing the risk of premature convergence.

2. Fitness Evaluation
   Each chromosome S_i is evaluated using the penalty-based fitness function $F(S_i)$ described in previous section. The total penalty quantifies the degree of violation across all constraints, and lower values indicate higher-quality solutions. This fitness evaluation forms the foundation for selection, guiding the evolutionary process toward more feasible and optimized schedules.
3. Selection
   The selection operator determines which chromosomes are chosen from the current population to produce offspring in the next generation. In this study, a tournament selection strategy is adopted due to its simplicity, robustness, and ability to maintain a balance between selective pressure and population diversity. In each tournament, a small subset of individuals is randomly sampled from the population, and the chromosome with the lowest fitness value which represents the best solution within the subset is selected as a parent.
   Formally, the selected parent S_p can be expressed as:
   $$S_p = \underset{S_i \in T}{min}(f(S_i))$$
   where T denotes the tournament subset and $f(S_i)$ is the fitness value of individual S_i.
   This mechanism ensures that individuals with better fitness have a higher probability of being selected for reproduction, while still allowing less-fit chromosomes to occasionally participate in the process, thus preserving population diversity and preventing premature convergence.
4. Crossover
   The crossover operator is responsible for combining genetic material from two parent chromosomes to produce a new offspring. In this study, a single-point crossover strategy is implemented, where each chromosome is represented as an ordered sequence of class assignments. A random crossover point c is selected within the chromosome, and the offspring inherits genes from the first parent up to point c, while the remaining genes are taken from the second parent. This process can be formally represented as:
   $$O = \left[S_1^{(p1)}, S_2^{(p2)}, \ldots, S_c^{(p1)}, S_{c+1}^{(p2)}, \ldots, S_n^{(p2)}\right]$$
   where O denotes the offspring chromosome, $S_i^{(p1)}$ and $S_i^{(p2)}$ represent genes (class assignments) from parent 1 and parent 2 respectively, and n is the total chromosome length.
   This mechanism enables the algorithm to combine feasible partial schedules from both parents, promoting exploration of the solution space and preservation of building blocks that represent high-quality scheduling patterns.
5. Mutation
   The mutation operator introduces random alterations to individual chromosomes to maintain population diversity and prevent premature convergence. In this study, mutation is performed by randomly selecting one class assignment and modifying one or more of its attributes, such as its assigned day, room, or starting timeslot. Formally, given a mutation probability P_m, a gene g_i within chromosome S is modified as:
   $$g_i' = \begin{cases} Mutate(g_i), & if\ r < P_m \\ g_i, & otherwise \end{cases}$$
   where r is a uniformly distributed random number $r \in [0,1]$, and $Mutate(g_i)$ donates a stochastic operator that generates a new feasible value for the selected attribute.

This mutation process ensures that the algorithm can escape local optima and explore alternative feasible schedules while still respecting hard constraints defined in the fitness function.

6. Elitism and Population Update
   To preserve the best solutions, elitism is applied by directly copying a small number of top-performing individuals into the next generation. In this study, the top four individuals with the lowest penalty values are retained unaltered.
   The remaining population slots are filled with offspring produced by crossover and mutation. This mechanism ensures that the algorithm never loses the best solutions found so far, while still introducing sufficient variation to maintain progress.
7. Termination
   The evolutionary process continues for a fixed number of generations G or until a solution with zero penalty (fully feasible schedule) is found. At the end of the process, the chromosome with the lowest fitness value is selected as the best global solution, denoted by S^*:
   $$S^* = \min_{S_i \in P_G} (f(S_i))$$
   This solution is then passed to the Hill Climbing stage for local refinement, where further improvements are made through focused neighborhood search.

E. Hill Climbing Process

HC algorithm is employed as a local search mechanism to refine promising solutions produced by the Genetic Algorithm. While GA provides strong global exploration capability, its stochastic nature may lead to slow convergence or suboptimal solutions trapped near local minima. HC complements this process by performing an exploitation-oriented search that iteratively improves an individual solution through small, guided modifications in its neighborhood.

1. Local Search Principle
   HC operates by taking an initial solution S_0, typically one of the best chromosomes from the GA population, and generating neighboring solutions through small perturbations of its genes. Each neighbor differs from the current solution by a minor modification, such as changing the day, time slot, or room of a single class assignment.
   The new solution S' is evaluated using the same fitness function $F(S')$. If $F(S') < F(S)$, the algorithm accepts S' as the new current solution. This iterative process continues for a predefined number of iterations H, representing the maximum number of local refinement steps.
   Formally, the process can be defined as:
   $$S_{t+1} = \begin{cases} S', & if\ F(S') < F(S\_t) \\ S\_t, & otherwise \end{cases}$$
   where S_t denotes the current solution at iteration t, and *F(S)* is the penalty-based fitness value.
2. Neighborhood Generation
   The neighborhood of a solution, denoted by $N(S_t)$, is defined by all possible one-gene modification of S_t. In each iteration, one random gene is selected and mutated by changing one of its attributes: day assignment, starting time slot, room selection (subject to course type compatibility). This local modification strategy maintains solution feasibility while enabling the exploration of nearby configurations that might reduce penalty violations. By using simple but targeted neighborhood operations, HC efficiently exploits the local region around a candidate solution without disrupting its overall structure.
3. Stopping Criterion

The Hill Climbing procedure is executed for a fixed number of iterations H, typically between 10 and 20, depending on the complexity of the dataset.
The process terminates when either no improvement is observed for a predefined number of consecutive iterations, or the maximum number of iterations H is reached. At the end of the refinement process, the best local solution S^* is returned. This refined solution typically exhibits lower penalty scores and serves as an enhanced version of the original chromosome provided by GA.

4. Role in Hybrid Framework
Within the hybrid GA-HC framework, HC acts as a secondary exploitation phase that strengthens the quality of the best individuals found during the global search. By embedding HC after each GA generation, the model balances exploration and exploitation which that a key principle in metaheuristic optimization.

F. Hybrid GA-HC Integration

The proposed Hybrid Genetic Algorithm with Hill Climbing (GA-HC) combines the global exploration strength of the GA with the local exploitation capability of the HC method. The integration is designed to overcome the typical limitation of standard GA, which tends to converge slowly or stagnate in local minima due to its stochastic nature. By embedding HC within the evolutionary cycle, the hybrid framework achieves a more balanced optimization process that promotes both diversity and precision in the search for feasible timetables.

In this hybrid approach, the GA functions as the main optimization framework responsible for exploring a wide range of potential scheduling configurations. The process begins by initializing a population of candidate timetables, each encoded as a chromosome that represents a complete set of class assignments. Through iterative processes of selection, crossover, mutation, and elitism, the GA progressively evolves the population toward regions of lower penalty values based on the fitness function described earlier.

At the beginning of each generation, before the reproduction stage, the algorithm applies a local refinement process to enhance the quality of the best individuals. Specifically, the top k chromosomes with the lowest penalty values are refined using the HC procedure. Let $P_g = \{S_1, S_2, \dots, S_N\}$ denote the population at generation g. After evaluating the fitness of all individuals, the top-performing subset undergoes local search as follows:

$$S_i' = HillClimb(S_i, H)$$

where H represents the number of HC iterations performed for each selected chromosome S_i. During each iteration, HC explores a neighboring solution by applying a small perturbation to the chromosome, such as modifying the assigned day, timeslot, or room.

Once the refinement is completed, the improved individuals replace their previous versions in the population (Lamarckian learning) [31]. These refined individuals are then used in the subsequent GA reproduction phase, serving as both elites and candidates for parent selection during the tournament process. The next generation is then formed through elitism, crossover, and mutation applied to this updated population. Formally, the population update rule for the hybrid algorithm can be represented as:

$$P_{g+1} = Elitism(P_g') \cup Offspring\left(GA(P_g')\right)$$

where $P_g' = RefineTopK(p_g, k, H)$ denotes the population after the refinement of the top k individuals via HC.

This integration of GA and HC provides several advantages for solving the complex, constraint-based timetabling problem. By refining high-quality individuals before reproduction, the algorithm accelerates convergence and enhances the overall quality of the population. It reduces constraint violations early in the optimization process while maintaining sufficient diversity through stochastic crossover and mutation. Because HC is

selectively applied to only a few top individuals, the algorithm achieves a balance between exploration and exploitation, preventing premature convergence. The consistent use of the same fitness function across both GA and HC ensures that the global exploration and local exploitation stages operate under a unified optimization objective.

The parameters used in the hybrid GA-HC algorithm are summarized in Table 3. These include a population size of 30 individuals, a total of 100 generations, an elitism count of 4, and a mutation rate of 0.35. During each generation, the top $k = 2$ individuals are refined through $H = 25$ HC iterations. These settings provide a balance between computational efficiency and solution quality. The overall process of the hybrid algorithm is summarized in Algorithm 1, which outlines the integration of GA's population-based evolution with HC's local refinement stage. By combining the stochastic search capability of GA and the deterministic improvement mechanism of HC, the proposed model achieves a more robust and efficient optimization performance, producing high-quality timetables that satisfy both hard and soft constraints more consistently than either method alone.

Algorithm 1: Hybrid Genetic Algorithm with Hill Climbing (GA–HC)

Initialize population P with N random schedules
for $g \leftarrow 1$ **to** G **do**
  // Pre-breeding local refinement (Lamarckian)
  Sort P by fitness (ascending)
  for $i \leftarrow 1$ **to** $\min(k, |P|)$ **do**
    $P[i] \leftarrow \text{HillClimbing}(P[i], H)$
  end
  Sort P by fitness (ascending)
  if $\mathit{BestFitness}(P) = 0$ **then**
    break
  end
  // Form next generation
  $P_{\text{new}} \leftarrow \text{CloneTop}(P, E)$ // elitism after HC
  while $|P_{new}| < N$ **do**
    $p_1 \leftarrow \text{TournamentSelect}(P[1{:}10], k{=}3)$
    $p_2 \leftarrow \text{TournamentSelect}(P[1{:}10], k{=}3)$
    $c \leftarrow \text{Crossover}(p_1, p_2)$
    if $\mathit{rand}() < p_m$ **then**
      Mutate(c)
    end
    Append c to P_{new}
  end
  $P \leftarrow P_{\text{new}}$
end
return $\arg\min_{S \in P} \mathit{Fitness}(S)$

III. RESULT AND DISCUSSION

This section presents the experimental results and performance analysis of the proposed Hybrid Genetic Algorithm with Hill Climbing (GA-HC) for solving the university course scheduling problem. The objective of this section is to evaluate the algorithm's capability in minimizing constraint violations and improving convergence behavior compared to the baseline methods, namely the pure Genetic Algorithm and pure Hill Climbing.

A. Experimental Setup

The experimental setup was designed to evaluate the performance of the proposed GA–HC algorithm using the same dataset and constraint configurations described in the previous section. All experiments were implemented in Python 3.10 within the Google Colab environment. Each algorithm, including the pure Genetic Algorithm (Pure GA), the pure Hill Climbing (Pure HC), and the hybrid GA-HC, was executed for 100 generations with a population size of 30. The mutation rate, elitism count, and local refinement parameters were kept consistent with the settings presented in Section II. to ensure

comparability. To enhance the robustness of the results, all experiments were repeated ten times, and the average performance across runs was reported for statistical reliability.

B. Comparative Performance of GA, HC and GA-HC

To comprehensively evaluate the performance of the proposed hybrid optimization model, three algorithmic variants were implemented and compared under identical experimental conditions, namely the pure GA, the pure HC, and the hybrid GA-HC. Each method was executed using the same dataset and fitness evaluation criteria described in Section 3. The comparison focused on four primary performance indicators: (1) best fitness value, representing the lowest total penalty achieved; (2) average fitness, indicating solution stability across generations; (3) the number of hard and soft constraint violations; and (4) average computational time.

Table 3. Comparative performance metrics for GA, HC and GA-HC algorithm.

Algorithm	Population Size	Best Fitness	Convergence Generation	Average Lecturer Conflict	Average Room Conflict	Average Room Type Mismatch	Average Course Credits Overflow	Average Unpreferred Day	Average Unpreferred Time
Pure GA	30	440	300	0.10	16.47	0.00	0.00	89.90	136.13
Pure HC	30	579	300	2.00	27.00	0.00	0.00	92.00	137.00
GA-HC (Hybrid)	30	386	300	0.00	15.60	0.00	0.00	66.20	130.10

Table 3 presents the comparative quantitative results of the three algorithmic variants evaluated in this study, including the pure GA, the pure HC, and the hybrid GA-HC. The table reports key performance metrics such as the best fitness values, convergence generation, and average numbers of hard and soft constraint violations. As shown in the table, the proposed hybrid GA-HC achieved the lowest overall penalty and the smallest number of constraint violations, demonstrating its superior effectiveness in generating high-quality timetables within the defined constraints. These results indicate that the integration of the HC procedure into the GA framework significantly enhances local exploitation without compromising global exploration, leading to faster and more stable convergence toward optimal scheduling solutions.

The convergence behavior of the three algorithms is further analyzed to illustrate their optimization dynamics across generations. Figure 1 shows the evolution of fitness values throughout the optimization process for GA, HC, and GA-HC.

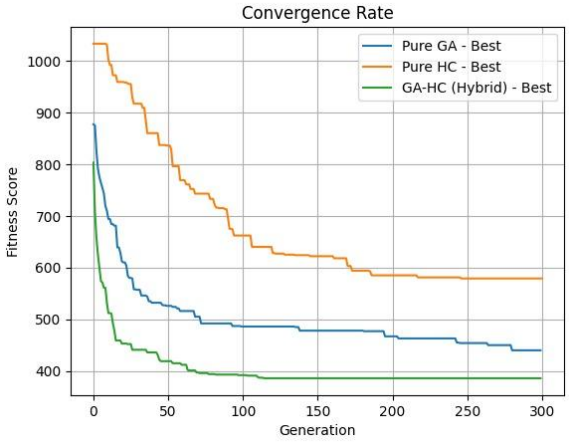

Figure 1. Convergence curves of GA, HC, and GA-HC over 300 generations

The curve corresponding to the hybrid GA-HC demonstrates a steeper initial decline and earlier stabilization compared to both the pure GA and HC, confirming its ability to

achieve faster convergence while maintaining solution stability. The pure HC shows rapid improvement during early iterations but quickly stagnates, reflecting its tendency to get trapped in local minima. Meanwhile, the pure GA displays a gradual but steady reduction in fitness values, indicating consistent exploration but slower optimization. The hybrid approach effectively combines these complementary characteristics, achieving both rapid convergence and enhanced final solution quality through its integrated local refinement mechanism.

C. Comparative Performance under Tuned Parameters

After establishing the comparative performance among GA, HC, and GA-HC in the previous section, a further investigation was carried out to examine how parameter tuning influences the performance of GA and GA-HC [35]. Since the standalone Hill Climbing algorithm demonstrated substantially weaker performance, it was excluded from this stage of analysis. The objective of this experiment was to observe the sensitivity of key parameters, including population size, elitism count, mutation rate, and the local refinement settings in GA-HC, specifically the number of top individuals refined ($refine\_top\_k$) and the number of iterations for each refinement (H). All experiments were conducted under the same dataset, fitness formulation, and constraint configurations to ensure comparability, and each configuration was executed multiple times to mitigate randomness.

The results are summarized in Table 4, which presents the mean best-fitness values obtained for both GA and GA-HC under several parameter configurations. The data clearly indicate that GA-HC consistently achieves lower mean fitness values across all settings, confirming its superior ability to minimize total scheduling penalties. The best performance was achieved with a population size of 30, an elitism count of 4, and a mutation rate of 0.35. For the hybrid algorithm, optimal results were obtained when $refine\_top\_k = 3$ and $H = 20$, suggesting that local refinement provides the most effective trade-off between solution quality and computational cost. Further increasing the number of iterations beyond this threshold produced improvements while significantly extending runtime, indicating diminishing returns.

Table 4. Mean best fitness results for parameter tuning of GA and GA-HC.

Algorithm	Parameters	Runs	Mean Fitness	Std Fitness	Best (min)	Worst (max)	Mean Runtime (s)
GA #0	$generations = 300$, $pop\_size = 30$, $elite = 4$, $mutation\_rate = 0.35$	3	434	6.164	427	442	10.887
GA #1	$generations = 300$, $pop\_size = 40$, $elite = 5$, $mutation\_rate = 0.3$	3	445.333	8.807	433	453	14.318
GA-HC #2	$generations = 300$, $pop\_size = 30$, $elite = 4$, $mutation\_rate = 0.35$, $refine\_top\_k = 2$, $hc\_iters\_each = 15$	3	427.333	16.049	415	450	13.963
GA-HC #3	$generations = 300$, $pop\_size = 30$, $elite = 4$, $mutation\_rate = 0.35$, $refine\_top\_k = 3$, $hc\_iters\_each = 20$	3	406	18.833	382	428	14.973
GA-HC #4	$generations = 300$, $pop\_size = 40$, $elite = 5$, $mutation\_rate = 0.3$, $refine\_top\_k = 2$,	3	400.333	11.086	386	413	17.691

	$hc\_iters\_each = 15$						
GA-HC #5	$generations = 300$, $pop\_size = 40$, $elite = 5$, $mutation\_rate = 0.3$, $refine\_top\_k = 3$, $hc\_iters\_each = 20$	3	403.333	6.799	394	410	19.37

Figure 2 illustrates the convergence patterns of both algorithms under their tuned parameters. The curve for GA-HC exhibits a notably steeper decline during the early generations, followed by smoother stabilization at later stages, indicating a faster and more stable convergence behavior. In contrast, the standard GA converges more gradually and displays greater fluctuation, suggesting that it explores a wider but less directed search space. The integration of Hill Climbing allows GA-HC to exploit promising local regions effectively while maintaining diversity through mutation and crossover, resulting in faster convergence toward high-quality feasible schedules.

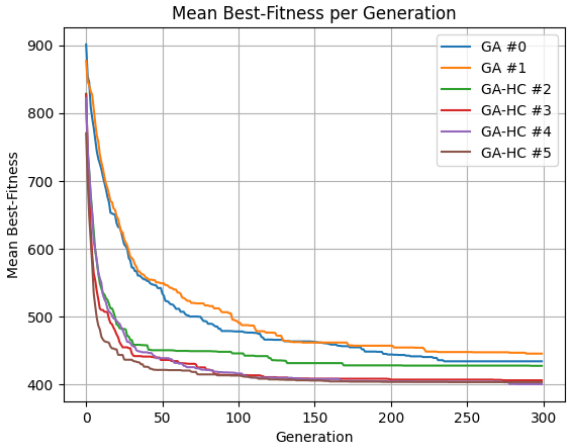

Figure 2. Mean best-fitness per generation for tuned GA and GA-HC algorithm

Further examination of parameter sensitivity reveals that the performance of the proposed hybrid GA-HC model is strongly influenced by key parameters governing population diversity and local refinement intensity. While higher population sizes and elitism levels in the pure GA tend to increase computational time without improving solution quality, the hybrid GA-HC demonstrates greater robustness, benefiting from moderate parameter settings that balance exploration and exploitation effectively. Excessive local refinement in the HC component may lead to premature convergence by over-exploiting local minima, whereas insufficient refinement limits the algorithm's ability to fine-tune promising regions. These findings highlight the importance of maintaining a balanced trade-off between global exploration and local exploitation to sustain performance stability. Statistical comparisons of mean best fitness values across multiple runs confirm that GA-HC achieves a significant improvement over the GA method, demonstrating its superior convergence speed, solution quality, and adaptability under varying parameter configurations.

The experimental results demonstrate that the hybrid GA-HC algorithm effectively integrates the global exploration capability of the Genetic Algorithm with the local exploitation strength of Hill Climbing, yielding superior convergence behavior and solution quality. Across varying population sizes and elitism levels, GA-HC consistently achieved lower mean fitness values and fewer constraint violations than the standard GA, indicating enhanced robustness and stability under parameter variations. These outcomes confirm that the incorporation of local search within the evolutionary framework significantly improves the algorithm's efficiency in navigating dense constraint-based

scheduling problems, thereby highlighting the potential of hybrid metaheuristic strategies as a robust approach to complex optimization tasks.

IV. CONCLUSION

This study presented a hybrid optimization approach combining the Genetic Algorithm (GA) and Hill Climbing (HC) for solving the university course scheduling problem in the Informatics Study Program at Universitas Islam Negeri Siber Syekh Nurjati Cirebon. The proposed GA-HC algorithm integrates the global exploration capability of GA with the local exploitation precision of HC, aiming to minimize both hard and soft constraint violations in course timetabling.

Experimental results demonstrated that the hybrid GA-HC consistently outperformed both pure GA and pure HC in terms of best fitness value, convergence speed, and solution stability. The inclusion of the HC-based local refinement stage within the GA evolutionary process effectively enhanced convergence and reduced constraint violations, producing more feasible and balanced timetables. Parameter tuning further reinforced these advantages, showing that moderate levels of mutation and local refinement yield the best trade-off between computational cost and optimization quality.

The findings confirm that the hybrid GA-HC approach provides a robust and efficient framework for handling complex, constraint-based scheduling problems in higher education. However, the results have not yet reached an optimal point where all hard constraints are fully satisfied, which may be attributed to the high density of the scheduling data. This limitation should be acknowledged and addressed in future studies. Future research may extend this work by incorporating adaptive parameter control, employing multi-objective optimization to simultaneously consider lecturer satisfaction and room utilization, or exploring hybridization with other local search methods to further enhance scalability and performance.

REFERENCES

[1] L. R. Lattuca and J. S. Stark, *Shaping the college curriculum: Academic plans in context*. John Wiley & Sons, 2009.

[2] J. S. Tan, S. L. Goh, G. Kendall, and N. R. Sabar, "A survey of the state-of-the-art of optimisation methodologies in school timetabling problems," *Expert Syst Appl*, vol. 165, p. 113943, 2021, doi: https://doi.org/10.1016/j.eswa.2020.113943.

[3] K. Song, S. Kim, M. Park, and H.-S. Lee, "Energy efficiency-based course timetabling for university buildings," *Energy*, vol. 139, pp. 394–405, 2017, doi: https://doi.org/10.1016/j.energy.2017.07.176.

[4] B. A. Jhonlawi, "Optimizing Educational Scheduling: ACO-Based Lecture Schedule Preparation Application," *International Journal of Enterprise Modelling*, vol. 18, no. 2, pp. 52–62, 2024.

[5] M. Lindahl, A. J. Mason, T. Stidsen, and M. Sørensen, "A strategic view of University timetabling," *Eur J Oper Res*, vol. 266, no. 1, pp. 35–45, 2018, doi: https://doi.org/10.1016/j.ejor.2017.09.022.

[6] A. Muluk, H. Akpolat, and J. Xu, "Scheduling problems—an overview," *J Syst Sci Syst Eng*, vol. 12, no. 4, pp. 481–492, 2003.

[7] M. C. Chen, S. N. Sze, S. L. Goh, N. R. Sabar, and G. Kendall, "A Survey of University Course Timetabling Problem: Perspectives, Trends and Opportunities," *IEEE Access*, vol. 9, pp. 106515–106529, 2021, doi: 10.1109/ACCESS.2021.3100613.

[8] A. Gülcü and C. Akkan, "Robust university course timetabling problem subject to single and multiple disruptions," *Eur J Oper Res*, vol. 283, no. 2, pp. 630–646, 2020, doi: https://doi.org/10.1016/j.ejor.2019.11.024.

[9] S. and B. A. and E. H. and H. P. and M. K. Aschinger M. and Applebee, "New Constraints and Features for the University Course Timetabling Problem," in *Operations Research Proceedings 2016*, A. and G. M. J. Fink Andreas and Fügenschuh, Ed., Cham: Springer International Publishing, 2018, pp. 95–101.

[10] H. Babaei, J. Karimpour, and A. Hadidi, "A survey of approaches for university course timetabling problem," *Comput Ind Eng*, vol. 86, pp. 43–59, 2015, doi: https://doi.org/10.1016/j.cie.2014.11.010.

[11] N. and V. J. and L. O. and M. J. and V. M. Silva Jose and Varela, "Comparison of Bioinspired Algorithms Applied to the Timetabling Problem," in *Computational Methods and Data Engineering*, V. K. and K. S. and P. R. B. Singh Vijendra and Asari, Ed., Singapore: Springer Singapore, 2021, pp. 427–437.

[12] T. Song, M. Chen, Y. Xu, D. Wang, X. Song, and X. Tang, "Competition-guided multi-neighborhood local search algorithm for the university course timetabling problem," *Appl Soft Comput*, vol. 110, p. 107624, 2021, doi: https://doi.org/10.1016/j.asoc.2021.107624.

[13] A. A. Lazarev, "Solution of the NP-hard total tardiness minimization problem in scheduling theory," *Computational Mathematics and Mathematical Physics*, vol. 47, no. 6, pp. 1039–1049, 2007.

[14] A. I. Diveev, O. V Bobr, D. E. Kazaryan, and O. Hussein, "Some methods of solving the NP-difficult problem of optimal schedule for the university," *Procedia Comput Sci*, vol. 150, pp. 410–415, 2019.

[15] B. Alhijawi and A. Awajan, "Genetic algorithms: Theory, genetic operators, solutions, and applications," *Evol Intell*, vol. 17, no. 3, pp. 1245–1256, 2024.

[16] M. A. Albadr, S. Tiun, M. Ayob, and F. Al-Dhief, "Genetic algorithm based on natural selection theory for optimization problems," *Symmetry (Basel)*, vol. 12, no. 11, p. 1758, 2020.

[17] D. Kristiadi and R. Hartanto, "Genetic Algorithm for lecturing schedule optimization," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 13, no. 1, pp. 83–94, 2019.

[18] A. K. Nugroho, I. Permadi, A. R. Yasifa, and others, "Optimizing course scheduling faculty of engineering unsoed using genetic algorithms," *JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer)*, vol. 7, no. 2, pp. 91–98, 2022.

[19] E. A. Abdelhalim and G. A. El Khayat, "A utilization-based genetic algorithm for solving the university timetabling problem (uga)," *Alexandria Engineering Journal*, vol. 55, no. 2, pp. 1395–1409, 2016.

[20] Y. G. Xu, G. R. Li, and Z. P. Wu, "A novel hybrid genetic algorithm using local optimizer based on heuristic pattern move," *Applied Artificial Intelligence*, vol. 15, no. 7, pp. 601–631, 2001.

[21] F. Sabry, *Hill Climbing: Fundamentals and Applications*, vol. 80. One Billion Knowledgeable, 2023.

[22] K. Sarode and S. R. Javaji, "Hybrid Genetic Algorithm and Hill Climbing Optimization for the Neural Network," *arXiv preprint arXiv:2308.13099*, 2023.

[23] J. Almeida, J. R. Figueira, A. P. Francisco, and D. Santos, "A hybrid meta-heuristic for the generation of feasible large-scale course timetables using instance decomposition," *arXiv preprint arXiv:2310.20334*, 2023.

[24] A. Rezaeipanah, S. S. Matoori, and G. Ahmadi, "A hybrid algorithm for the university course timetabling problem using the improved parallel genetic algorithm and local search," *Applied Intelligence*, vol. 51, no. 1, pp. 467–492, 2021.

[25] A. Hussain, H. Ashas, A. Shahid, S. Qureshi, and S. Karrila, "Hybrid approach involving genetic algorithm and hill climbing to resolve the timetable scheduling for a university," in *Future of Information and Communication Conference*, 2024, pp. 72–83.

[26] E. Mirsadeghi and S. Khodayifar, "Hybridizing particle swarm optimization with simulated annealing and differential evolution," *Cluster Comput*, vol. 24, no. 2, pp. 1135–1163, 2021, doi: 10.1007/s10586-020-03179-y.
[27] K. Sun *et al.*, "Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system," *Expert Syst Appl*, vol. 215, p. 119359, 2023, doi: https://doi.org/10.1016/j.eswa.2022.119359.
[28] F. H. Awad, A. Al-Kubaisi, and M. Mahmood, "Large-scale timetabling problems with adaptive tabu search," *Journal of Intelligent Systems*, vol. 31, no. 1, pp. 168–176, Jan. 2022, doi: 10.1515/jisys-2022-0003.
[29] T. A. El-Mihoub, A. A. Hopgood, and L. Nolle, "Self-adaptive learning for hybrid genetic algorithms," *Evol Intell*, vol. 14, no. 4, pp. 1565–1579, 2021, doi: 10.1007/s12065-020-00425-5.
[30] Z. Sharifi, K. Soltanian, and A. Amiri, "Developing Convolutional Neural Networks using a Novel Lamarckian Co-Evolutionary Algorithm," in *2023 13th International Conference on Computer and Knowledge Engineering (ICCKE)*, 2023, pp. 399–408. doi: 10.1109/ICCKE60553.2023.10326238.
[31] B. J. Ross, "A Lamarckian evolution strategy for genetic algorithms," in *Practical handbook of genetic algorithms*, CRC Press, 2019, pp. 1–16.
[32] H. Zhang and M. Ishikawa, "An extended hybrid genetic algorithm for exploring a large search space," in *Proceedings of the 2nd International Conference on Autonomous Robots and Agents*, Dec. 2004, pp. 244–248.
[33] A. Rjoub, "Courses timetabling based on hill climbing algorithm," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, 2020, doi: 10.11591/ijece.v10i6.pp6558-6573.
[34] H. Alghamdi, T. Alsubait, H. Alhakami, and A. Baz, "A review of optimization algorithms for university timetable scheduling," *Engineering, Technology & Applied Science Research*, vol. 10, 2020, doi: 10.48084/etasr.3832.
[35] R. Bellio, S. Ceschia, L. Di Gaspero, A. Schaerf, and T. Urli, "Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem," *Comput Oper Res*, vol. 65, pp. 83–92, 2016.