Hybrid GA-GWO with Dual-Vector Encoding for Indonesian School Timetabling

Akbar Muhammad Sadat [1], Alqis Rausanfita [2,*], Pima Hani Safitri [3]
[1] Informatics Study Program, Telkom University, Surabaya, Indonesia
[2, 3] Center of Excellent Motion Technology for Safety Health and Wellness, Telkom University, Surabaya, Indonesia
Email: [1] akbarmuhammadsadat@student.telkomuniversity.ac.id, [2] alqisfita@telkomuniversity.ac.id,
[3] phanisafitri@telkomuniversity.ac.id
*Corresponding Author

***Abstract*—School timetabling is a complex combinatorial optimization problem that involves assigning subjects, teachers, and classes to predefined time slots while satisfying numerous institutional constraints. In many Indonesian junior high schools, scheduling is still performed using manual approaches, which are often time-consuming and prone to conflicts. Compared with university timetabling, school timetabling presents additional challenges due to fixed class groups, rigid subject allocations, teacher availability constraints, and institutional regulations. To address these challenges, this study proposes a hybrid optimization framework that combines a Guided Genetic Algorithm (GA) and Grey Wolf Optimizer (GWO) for the school timetable. The proposed framework incorporates dual-vector solution encoding to provide a structured representation of scheduling components and support efficient constraint handling during the optimization process. In addition, a majority-voting and guided mutation strategy is employed to enhance the balance between exploration and exploitation. The proposed method was evaluated using real-world scheduling data from an Indonesian junior high school consisting of 27 classes, 54 teachers, 13 subjects, and 36 time slots. Experimental results show that the proposed hybrid GA-GWO achieved a fitness improvement of 95.84%, reducing the fitness value from 16,120 to 670, compared with improvements of 89.83% and 94.31% obtained by Traditional GA and Guided GA, respectively. Although the proposed method required approximately 28 minutes of execution time, it produced the highest overall timetable quality among the evaluated approaches. These findings demonstrate that the integration of dual-vector encoding, majority voting, and guided mutation within a hybrid GA-GWO framework can effectively improve timetable optimization for real-world Indonesian school scheduling environments.**

Keywords*—*School Timetabling; Genetic Algorithms; Grey Wolf Optimizer; Hybrid Metaheuristic; Constraint Optimization

I. INTRODUCTION

School timetabling plays an essential role in managing academic activities in educational institutions, particularly in Indonesian junior high schools (Sekolah Menengah Pertama / *SMP*), where scheduling processes are still largely performed manually [1], [2]. This process requires the allocation of classes, teachers, subjects, and time slots under multiple institutional constraints, making it a complex combinatorial optimization problem [2]-[4]. In the Indonesian education system, scheduling complexity is further increased by nationally standardized curricula, fixed subject structures, teacher certification requirements, and inter-school teacher working groups (Musyawarah Guru Mata Pelajaran/*MGMP*), which introduce additional constraints beyond standard timetabling models [2]. These factors make timetable construction highly context-dependent and difficult to generalize across schools [5], [6].

The complexity of the timetable problem increases as the quantity of components grows. In a typical *SMP* environment, dozens of classes and teachers must be scheduled simultaneously, resulting in a highly constrained problem with a large search space [7], [8]. This often leads to scheduling conflicts such as overlapping teaching assignments and imbalanced workloads, which can disrupt teaching and learning activities [9]-[11]. Table 1 illustrates the complexity of scheduling multiple classes within a single day (Monday). Furthermore, resolving conflicts manually may require considerable time and effort, reduce administrative efficiency and delaying academic processes [2], [9].

Table 1. Example of Scheduling Complexity in a Single School Day

Time Slot	Class 7A	Class 7B	Class 7C
07:00 – 07:40	Flag Ceremony		
07:40 – 08:00	Reflective Journal Activity		
08:00 – 08:40	Mathematics (T1)	Indonesian (T2)	English (T3)
08:40 – 09:20	Mathematics (T1)	Indonesian (T2)	English (T3)
09:20 – 09:40	First Break		
09:40 – 10:20	Mathematics (T1)	Indonesian (T2)	Mathematics (T1)
10:20 – 11:00	Science (T4)	Science (T5)	Mathematics (T1)
11:00 – 11:40	Science (T4)	Science (T5)	Indonesian (T6)
11:40 – 12:20	Informatics (T1)	English (T2)	Indonesian (T6)
12:20 – 13:20	Second Break		
13:20 – 14:00	Informatics (T1)	English (T2)	Science (T4)
14:00 – 14:40	Indonesian (T6)	English (T2)	Science (T4)

Despite its importance, many schools still rely on manual scheduling methods that are highly prone to human error and inefficiency [2], [9]. Such approaches often produce suboptimal timetables characterized by teacher conflicts, unbalanced workload distribution, and violations of institutional constraints [11], [12]. Moreover, manual scheduling places a significant administrative burden on curriculum staff, particularly in schools with limited digital infrastructure, resulting in delays in academic planning and reduced operational efficiency [8]. These limitations highlight the need for automated scheduling approaches capable of generating feasible and high-quality timetables in a more efficient manner [3].

From a computational perspective, the timetabling problem is classified as an NP-hard combinatorial optimization problem, where the solution space grows exponentially with the number of classes, teachers, and time slots [2]-[4]. The

problem involves multiple hard and soft constraints, including teacher availability, subject requirements, and classroom capacity constraints [7], [13], [14]. In Indonesian junior high schools, the problem is further complicated by institutional constraints such as fixed homeroom structures, teacher certification assignments, MGMP coordination schedules, and limited flexibility in subject-hour distribution. These constraints significantly increase the difficulty of generating feasible and optimal timetables, making exact optimization methods impractical for real-world applications.

To address this problem, various approaches have been proposed, including exact methods, hyper-heuristics, machine learning-assisted scheduling techniques, and metaheuristic algorithms [2], [8]. Table 2 summarizes representative studies in timetabling optimization research. Exact methods such as Integer Linear Programming and Constraint Satisfaction Problems can produce optimal solutions for small instances but become computationally expensive for large-scale problems [3]. Hyper-heuristics improve adaptability by dynamically selecting or generating heuristics, while deep learning-based approaches attempt to learn scheduling patterns from historical data [15]. However, these approaches often require significant computational resources, large datasets, or extensive domain knowledge [8]. Consequently, metaheuristic algorithms remain widely adopted due to their ability to efficiently explore large search spaces and produce near-optimal solutions [2], [7].

Table 2. Recent Timetabling Optimization Studies

Ref	Title	Method	Dataset Characteristics	Objective Function	Performance Metric	Main Limitation
[13]	Student Timetabling Genetic Algorithm Accounting For Student Preferences	Genetic Algorithm (GA)	King's College London with varying complexity	Allocate students to classes while maximizing student preference satisfaction and maintaining timetable feasibility	Student preference satisfaction rate, timetable feasibility, solution quality, computational efficiency	Focuses on student allocation rather than complex teacher–class scheduling. Performance depends on problem-specific operators and repair mechanisms
[25]	A Simulated Annealing Algorithm For The Faculty-Level University Course Timetabling Problem	Simulated Annealing (SA)	Faculty-level university course timetabling with shared classrooms, double-major, and minor-program constraints. Evaluated using random test instances and a real university case study	Minimize timetable conflicts while satisfying institutional constraints, including classroom availability, student schedules, and double-major/minor requirements	Objective function improvement, timetable feasibility, solution quality, computational time	Performance depends on parameter tuning and is primarily designed for university course timetabling rather than highly constrained school timetabling environments
[21]	A Novel Method for Solving the University Course Timetabling Problem Based on the Grey Wolf Optimizer Algorithm	Grey Wolf Optimizer (GWO) + Graph Coloring	Real university course timetabling dataset from the Information Technology Department, 20 August 1955 Skikda University	Generate feasible timetables by satisfying all hard constraints and maximizing soft-constraint satisfaction	Hard-constraint satisfaction rate, soft-constraint satisfaction rate, solution validity, computational efficiency	Requires an initial population of feasible solutions generated through graph coloring. Focuses primarily on search optimization rather than solution representation and constraint-handling mechanisms
[15]	A parallelised hyper-heuristic framework for the Integrated Healthcare Timetabling Competition 2024	Parallelised Hyper-Heuristic Framework	Integrated Healthcare Timetabling Competition (IHTP 2024) benchmark instances	Generate feasible schedules and improve timetable quality through heuristic selection and solution acceptance mechanisms	Feasibility of generated schedules, solution quality, heuristic effectiveness, and computational performance	Did Not Achieve Competition-Winning Performance. Effectiveness Depends On Heuristic Selection And Acceptance Strategies. Requires Further Adaptation For Domain-Specific Scheduling Problems
[26]	Train Timetabling with General Learning Environment and Multi-Agent Deep Reinforcement Learning	Multi-Agent Deep Reinforcement Learning (MADRL) with Actor–Critic framework	Real-world railway timetabling instances, including single-track and double-track railway systems	Optimize train scheduling while satisfying operational constraints (track conflicts, timing constraints) in both single-track and double-track environments	Solution optimality, computational time, feasibility of schedules, performance compared to baseline methods	Requires complex environment modeling as Markov Decision Process. High computational cost due to deep neural networks. Scalability may depend on training stability and reward design
[27]	Gradual Optimization of University Course Scheduling Problem Using Genetic Algorithm and Dynamic Programming	Hybrid Genetic Algorithm + Dynamic Programming (POGA-DP)	Real university course scheduling instances from Beijing Forestry University	Minimize scheduling conflicts while improving overall timetable quality, optimizing classroom utilization, and handling both joint and independent course constraints simultaneously	Fitness value improvement, scheduling quality gain, classroom utilization rate, convergence speed, stability of scheduling across iterations	Performance evaluation is limited to a single institutional dataset. Generalization across different universities, constraint structures, and large-scale heterogeneous timetabling environments is not fully validated

Among metaheuristic approaches, Genetic Algorithms (GA) are widely used due to their strong exploration capability through evolutionary operators such as selection, crossover, and mutation [16]-[18]. However, GA often suffers from premature convergence and slow exploitation in later iterations [19], [20]. In contrast, Grey Wolf Optimizer (GWO) provides strong exploitation capability by guiding candidate solutions toward the best-performing individuals, but it may lack sufficient exploration in early search stages [16], [17]. These complementary characteristics suggest that neither method is sufficient on its own for complex timetabling problems, motivating the need for hybrid optimization strategies [14], [21].

To overcome these limitations, hybrid metaheuristic approaches have been introduced by combining the strengths of multiple algorithms [2], [14]. One promising approach is the integration of Genetic Algorithm and Grey Wolf Optimizer, where GA provides global exploration while GWO enhances local exploitation. Previous studies have reported improvements in convergence speed and solution quality using hybrid GA-GWO approaches [21], [22]. However, most existing studies primarily focus on improving search performance, while relatively limited attention has been given to how solution representation influences constraint handling and feasibility preservation during optimization. As a result, hard-constraint violations often persist during optimization [1], [7].

Most existing timetable studies employ conventional single-vector solution representations, where all scheduling information is encoded within a single chromosome structure [23], [24]. While effective for simpler scheduling environments, such representations may become inefficient when handling large numbers of classes, teachers, and institutional constraints [3]. In school timetabling, these two types of scheduling decisions often exhibit different constraint characteristics, making unified encoding structures more difficult to optimize efficiently. In addition, many existing approaches rely heavily on repair mechanisms to restore feasibility after crossover and mutation [14]. These limitations indicate the need for more efficient encoding strategies that can better support constraint handling and optimization performance.

Based on the identified limitations, the research gap addressed in this study is not only the limited application of hybrid GA-GWO approaches in school timetabling but also the lack of effective solution representations and constraint-handling mechanisms in existing optimization frameworks. Current studies primarily focus on improving convergence behavior, while insufficient attention is given to encoding structures and their impact on feasibility preservation [2]. Furthermore, most hybrid GA-GWO implementations do not explicitly address the decomposition of scheduling variables, leading to inefficient constraint management. Therefore, a more structured optimization framework that integrates solution representation with hybrid search mechanisms is required.

In this study, a hybrid Genetic Algorithm–Grey Wolf Optimizer framework with a dual-vector solution representation is proposed for school timetabling optimization. Unlike conventional single-vector encoding, the proposed representation separates scheduling decisions into distinct components, enabling more efficient constraint evaluation and reducing the complexity of feasibility checking. The proposed integration strategy combines GA-based global exploration with GWO-inspired local refinement to improve solution quality while maintaining feasibility throughout the optimization process. This structured separation allows the search process to operate on more manageable subspaces, reducing constraint violations and improving computational efficiency.

The primary contributions of this study are as follows. First, a dual-vector solution representation is introduced to improve constraint-handling efficiency in school timetabling optimization by decomposing scheduling variables into structured components. Second, a hybrid optimization strategy combining Genetic Algorithm and Grey Wolf Optimizer is developed, where GA enhances global exploration and GWO improves local exploitation, resulting in a balanced search process that improves both feasibility and solution quality. The proposed framework is validated using a real-world Indonesian junior high school dataset characterized by strict institutional constraints.

II. METHODOLOGY

A. Problem Formulation

School timetabling is a constraint-based scheduling problem that involves assigning subjects, teachers, classes, and time slots into a structured timetable while satisfying multiple institutional requirements [4], [7], [28]. Unlike university timetabling, school timetabling is characterized by fixed class groups, predefined subject-hour allocations, teacher workload requirements, and institutional regulations, making the problem highly constrained and difficult to solve [2], [3].

In this study, the timetabling problem consists of four primary scheduling components such as a set of classes $C = \{c_1, c_2, \ldots, c_n\}$, a set of teachers $T = \{t_1, t_2, \ldots, t_m\}$, a set of subjects $S=\{s_1,s_2,\ldots,s_p\}$, and a set of available time slots $L = \{l_1, l_2, \ldots, l_q\}$. The objective is to generate a feasible timetable by assigning subject-teacher combinations to each class and time slot while satisfying all mandatory scheduling constraints.

The timetabling problem is formulated as a minimization problem. The timetabling problem is formulated as a minimization problem, where the objective is to minimize the total penalty associated with constraint violations:

$$\min\ f(x) = \sum_{i=1}^{m} w_i\, V_i(x) \tag{1}$$

where $f(x)$ represents the total penalty of timetable solution x, $V_i(x)$ denotes the number of violations associated with the constraint i, w_i represents the penalty weight assigned to constraint i, and m denotes the total number of evaluated constraints. Lower fitness values indicate fewer constraint violations and higher timetable quality.

In this study, constraints are categorized into hard constraints. Hard constraints must always be satisfied to ensure timetable feasibility. Since the primary objective of this research is to generate feasible school timetables, hard-constraint violations are assigned substantially larger penalties than soft-constraint violations.

Table 3 summarizes the hard constraints considered in this study. These constraints are derived from institutional scheduling requirements observed in Indonesian junior high schools and represent mandatory conditions that must be satisfied during timetable optimization.

As an illustrative example, a teacher conflict occurs when a teacher is assigned to multiple classes during the same time slot. This constraint can be formally expressed as:

$$\sum_{c \in C} I(t,c,l) \leq 1, \quad \forall t \in T, \forall l \in L \tag{2}$$

where $I(t,c,l) = 1$ if teacher t is assigned to class c at time slot l, and 0 otherwise. This constraint ensures that a teacher cannot teach more than one class simultaneously.

Table 4 presents an example of a teacher conflict violation, where Teacher T1 is assigned to two classes within the same time slot. Under the proposed dual-vector representation, such violations can be efficiently identified through the relationship between teacher assignments and time-slot allocations.

Table 3. List of Constraints

No	Constraint	Description
1	Teacher Conflict	A teacher unable to teach more than one class at the same time
2	Subject Distribution	Subjects must follow predefined distribution patterns
3	Physics Subject	Physics subjects should not be scheduled in afternoon slots
4	Teacher Workload	Teaching hours must match predefined workload limits
5	MGMP Time	Subject scheduling must follow MGMP daily limits
6	Homeroom Teacher	Homeroom teachers must teach their assigned class
7	Teacher Consistency	A subject in a class must be taught by the same teacher

Table 4. Example of Constraint Violation (Teacher Conflict)

Slot	Class A	Class B
1	T1 (violation)	T1 (violation)
2	T1 (violation)	T1 (violation)
3	T2	T3
4	T2	T3
5	T2	T4

B. Dataset Description

The dataset used in this study was obtained from real scheduling data of SMP Negeri 5 Jombang for the 2025/2026 academic year. The dataset consists of multiple scheduling entities, including teachers, subjects, classes, time slots, and teacher–subject relationships that define permissible teaching assignments. The dataset represents a real-world scheduling scenario characterized by fixed class assignments, predefined teacher–subject relationships, workload requirements, and institutional scheduling constraints, making it suitable for evaluating school timetabling optimization methods. The data was collected from the school administration system and manually verified to ensure consistency. A summary of the dataset is presented in Table 5.

Table 5. List of Datasets

No	Dataset	Quantity
1	Teachers	54
2	Subjects	13
3	Classes	27
4	Time Slots	36
5	Teacher-subject Relations	93

C. Experimental Environment

All experiments were conducted in a standard computing environment to evaluate the performance of the proposed method. The implementation was developed using Python with supporting libraries such as NumPy and Pandas. The hardware specifications are summarized in Table 6 to ensure reproducibility.

Table 6. Experimental Device Specification

No	Components	Specification
1	Processor	Intel Core i7-13650HX
2	RAM	12 GB DDR5
3	Storage	SSD 512 GB
4	Operating System	Windows 11 25H2
5	Programming Language	Python 3.9
6	Libraries	Pandas, NumPy, Random, Collections

D. Proposed Method

This study proposes a hybrid optimization framework that combines a Guided Genetic Algorithm (GA) with a Grey Wolf Optimizer (GWO) to solve the school timetabling problem. Genetic Algorithm is employed as the primary global search strategy because its evolutionary operators, including selection, crossover, and mutation, are well suited for exploring large combinatorial search spaces and generating diverse timetable solutions [23], [24]. However, GA often experiences slow exploitation and premature convergence during later generations [13], [24]. To overcome these limitations, Grey Wolf Optimizer is incorporated as a complementary local search strategy. GWO utilizes the three best candidate solutions (alpha, beta, and delta) to guide the search toward promising regions of the solution space, thereby strengthening exploitation while maintaining search stability [22]. By integrating the exploration capability of GA with the exploitation capability of GWO, the proposed hybrid framework aims to achieve a more balanced optimization process and improve timetable quality under complex scheduling constraints.

Based on Fig. 1, the optimization process begins by loading the scheduling dataset, which contains teachers, classes, subjects, and available time slots. These scheduling components constitute the decision variables used throughout the optimization process and provide the information required for evaluating timetable feasibility under institutional constraints.

During pre-processing, all categorical scheduling information, including teacher identities, subject names, and class labels, is transformed into numerical identifiers through a data dictionary. This transformation enables efficient manipulation of scheduling data during optimization because integer-based representations require less memory and faster comparison operations than string-based representations [20]. Consequently, computational overhead during crossover, mutation, and fitness evaluation can be reduced.

After preprocessing, an initial population of candidate timetable solutions is randomly generated. The purpose of this initialization is to provide diverse starting solutions that cover different regions of the search space. Population diversity at the beginning of the optimization process is important because it reduces the risk of premature convergence and increases the opportunity for the evolutionary process to discover high-quality timetable solutions.

Each candidate solution is evaluated using the proposed fitness function to determine its scheduling quality based on the weighted constraint violations. The resulting fitness values serve three purposes for selecting parent individuals during tournament selection, identifying the alpha, beta, and delta leaders for the Grey Wolf Optimizer, and determining the best timetable returned as the final optimization result.

Tournament selection is adopted to select parent individuals for crossover because it increases the probability that high-quality solutions participate in the reproduction process while preventing excessively random parent selection. Selecting individuals with better fitness values reduces the likelihood of propagating poor scheduling structures into subsequent generations, thereby improving convergence efficiency and reducing unnecessary repair operations.

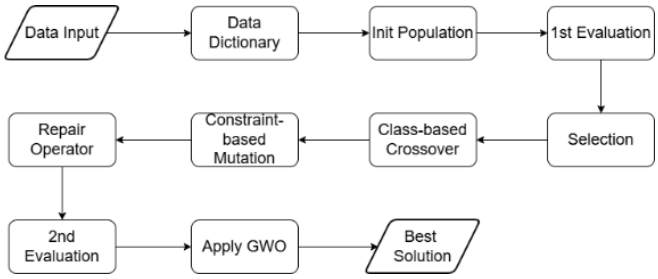

Fig. 1. Overall procedure process of Hybrid Guided GA-GWO

A class-based crossover operator is employed to generate new timetable solutions while preserving class scheduling structures. Unlike conventional one-point or multi-point crossover, which frequently disrupt partially optimized class schedules by exchanging arbitrary chromosome segments, the proposed operator exchanges complete class schedules between parent individuals. This strategy preserves subject-hour distributions and class consistency while still introducing sufficient diversity into the population. Consequently, fewer scheduling conflicts are introduced after crossover, reducing the need for extensive repair operations. The detailed procedure of the class-based crossover is presented in Fig. 2.

Algorithm 1 Class-Based Crossover

Input: Parent1 (S_1, T_1), Parent2 (S_2, T_2)
Output: Child1, Child2
Randomly select split point $k \in [1, \text{numClasses} - 1]$
Initialize Child1 and Child2
for each class index i **do**
  if $i < k$ **then**
    Child1 ← Parent1
    Child2 ← Parent2
  else
    Child1 ← Parent2
    Child2 ← Parent1
  end if
end for
Return Child1, Child2

Fig. 2. Class-Based Crossover

Unlike conventional random mutation, the proposed mutation operator prioritizes the constraint with the highest number of violations before applying modifications. This strategy is based on preliminary observations obtained during timetable construction and consultation with curriculum administrators. By directing mutation toward the dominant source of infeasibility, computational effort is concentrated on resolving the most critical conflicts first, resulting in faster convergence and more efficient optimization. Remaining constraints generally involve localized scheduling adjustments and therefore can be resolved more easily after the major conflicts have been eliminated. The mutation process is described in Fig. 3.

Algorithm 2 Constraint-Based Mutation

Input: Individual (S, T)
Output: Mutated Individual
Evaluate constraint violations
Identify most violated constraint c
Randomly select class k
if c is Subject Distribution **then**
  Swap two adjacent slots in class k
else if c is Teacher Conflict **then**
  Select random slot
  Replace teacher with valid teacher
else
  Swap two random slots in class k
end if
Return updated (S, T)

Fig. 3. Constraint-based Mutation

The repair operator is applied after mutation to restore timetable feasibility before the next fitness evaluation. Mutation intentionally introduces modifications that may temporarily violate hard constraints. Therefore, applying repair immediately afterward prevents infeasible solutions from propagating into subsequent optimization stages. Applying repair before mutation would be ineffective because new violations could still be introduced by the mutation process. Likewise, applying repair immediately after crossover is unnecessary because crossover primarily preserves class structures through the proposed class-based mechanism, whereas most hard-constraint violations are introduced during targeted mutation. The repair process focuses on teacher conflicts because this constraint carries the highest penalty weight and directly determines timetable feasibility. Resolving teacher conflicts first substantially reduces the overall fitness value and improves the effectiveness of subsequent optimization. The repair process is described in Fig. 4.

Algorithm 3 Repair Operator

Input: Individual (S, T)
Output: Repaired Individual
Evaluate teacher conflicts
if conflict exists **then**
  for each time slot **do**
    Identify teachers assigned multiple times
    for each conflicting teacher **do**
      Select one class randomly
      Replace teacher with valid alternative
    end for
  end for
end if
Return repaired (S, T)

Fig. 4. Repair Operator

Unlike the original Grey Wolf Optimizer developed for continuous optimization problems, the proposed method employs a discrete adaptation suitable for school timetabling. Since timetable solutions consist of discrete subject-teacher assignments rather than continuous numerical variables, the original position-update equations cannot be directly applied.

Therefore, the alpha, beta, and delta leaders guide solution refinement through a majority-voting mechanism that determines the most appropriate scheduling decisions for selected timetable components. The detailed process is shown in Fig. 5.

Algorithm 4 Grey Wolf Optimizer (GWO)

```
Input: Population P
Output: Updated Population
Identify α, β, δ (best 3 individuals)
for each individual i in population do
    if i is α then
        Keep unchanged
    else
        Select random classes and slots
        for selected slots do
            Perform majority voting from α, β, δ
            Update subject and teacher
        end for
        Apply local swap
        Apply repair operator
        if new fitness better then
            Accept solution
        end if
    end if
end for
Return updated population
```

Fig. 5. Grey Wolf Optimizer

Majority voting is introduced as the discrete adaptation of the leadership mechanism in Grey Wolf Optimizer as shown in Fig. 6. For each selected scheduling component, the corresponding values from the alpha, beta, and delta solutions are compared, and the value supported by at least two leaders is selected as the new candidate assignment. This mechanism enables the exploitation process to preserve high-quality scheduling patterns while remaining compatible with discrete timetable representations.

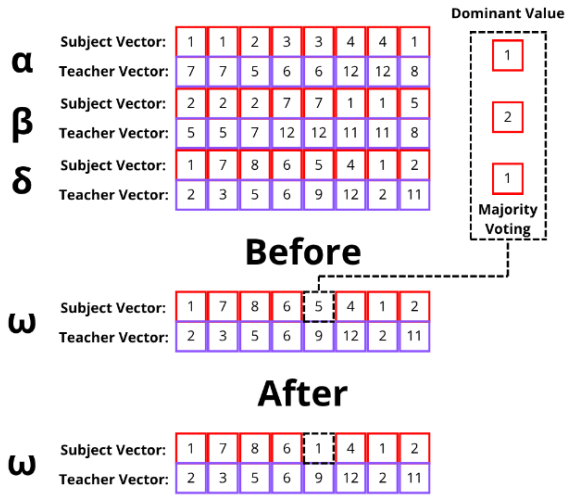

Fig. 6. Illustration of Majority Voting

After majority voting, a local swap operator is applied by exchanging two selected timetable positions within the candidate solution. This neighborhood search refines local scheduling arrangements while preserving the overall timetable structure, thereby improving exploitation around promising solutions.

Fig. 6 illustrates the majority voting mechanism used in the proposed discrete Grey Wolf Optimizer. The values from the alpha, beta, and delta leaders are compared to determine the dominant value. In this example, the values $\{1, 1, 2\}$ produce a majority value of 1, replacing the current value 5 to preserve promising scheduling patterns during local exploitation.

Finally, the population is updated with improved individuals, and the process repeats until the termination condition is satisfied. The best individual found during the optimization process is selected as the final timetable solution. This hybrid approach effectively combines the global exploration ability of the Genetic Algorithm with the local exploitation strength of the Grey Wolf Optimizer, resulting in improved solution quality and convergence performance.

E. Proposed Dual-Vector Encoding

Solution representation is one of the most critical components in evolutionary timetabling because all genetic operators directly manipulate the encoded timetable structure. An effective representation should preserve feasible scheduling patterns while allowing crossover, mutation, and constraint evaluation to be performed efficiently. Therefore, the quality of the encoding strategy significantly influences the effectiveness of the optimization process, particularly for highly constrained school timetabling problems.

Most existing school timetabling studies employ conventional single-vector representations, where all scheduling information, including subjects, teachers, and timetable assignments, is encoded within a single chromosome. Although such representations are relatively simple to implement, modifications performed during crossover or mutation frequently affect multiple scheduling attributes simultaneously. Consequently, feasible timetable structures may be disrupted, resulting in additional constraint violations and increasing the need for repair operations after genetic manipulation.

To overcome these limitations, this study proposes a dual-vector encoding that separates subject assignments from teacher assignments into two independent but synchronized vectors. Both vectors have identical dimensions, where each position corresponds to the same class and time-slot combination. The subject vector stores the scheduled subject for each timetable position, whereas the teacher vector stores the corresponding teacher assigned to deliver the subject. This representation maintains a one-to-one correspondence between both vectors while allowing each scheduling component to be manipulated independently. An example of the proposed encoding is presented in Table 7.

Table 7. Example of Dual-Vector Representation (Partial Illustration)

No	Slot	1	2	3	4	5	6	...	971	972
1	Subject Vector	3	3	3	9	9	12	...	7	7
2	Teacher Vector	36	36	36	42	42	55	...	8	8

Fig. 7 and Fig. 8 illustrate the difference between the conventional single-vector representation and the proposed dual-vector encoding. In the conventional representation (Fig. 7), subject, teacher, and class information are stored together within a single chromosome, causing multiple scheduling attributes to be modified simultaneously during genetic operations. In contrast, the proposed representation

(Fig. 8) separates subject assignments and teacher assignments into two synchronized vectors while maintaining the same timetable positions. This separation enables each scheduling component to be manipulated independently, facilitating more targeted genetic operations and more efficient constraint handling.

```
[ (3,36,C1), (3,36,C1), (7,37,C1), (4,47,C1), ... ]
```

Fig. 7. Conventional Single-Vector Representation

```
Subject Vector : [3, 3, 7, 4, 9, ...]
Teacher Vector : [36,36,37,47,42,...]
```

Fig. 8. Proposed Dual-Vector Encoding

The proposed encoding reduces the dependency between scheduling variables during evolutionary operations. Since subject assignments and teacher assignments are represented separately, genetic operators can modify one component without unnecessarily altering the other. As a result, crossover and mutation become more localized, preserving feasible timetable structures while reducing unintended changes to unrelated scheduling decisions. This modular representation improves the stability of the evolutionary search and supports more effective constraint handling.

The separation of subject and teacher assignments also enables more targeted optimization strategies. During constraint-based mutation, modifications can be directed specifically toward teacher assignments when resolving teacher conflicts, while subject allocations remain unchanged. Likewise, the proposed class-based crossover exchanges complete class schedules without unnecessarily disrupting subject-hour distributions. These characteristics reduce the number of infeasible offspring generated during evolutionary operations and consequently decrease the reliance on extensive repair mechanisms.

The proposed representation also supports the discrete adaptation of the Grey Wolf Optimizer employed in this study. Since subject and teacher assignments are stored independently, the majority-voting mechanism can evaluate each scheduling component separately before constructing a new timetable solution. This structured representation simplifies the implementation of local exploitation while preserving scheduling consistency during the leader-guided refinement process.

Although the overall search space remains combinatorial, the proposed dual-vector encoding improves the practical efficiency of timetable optimization by simplifying constraint evaluation, reducing variable dependency, and minimizing unnecessary repair operations after genetic modifications. Consequently, the proposed representation complements the hybrid GA-GWO optimization framework and contributes to generating feasible timetable solutions more efficiently under complex school scheduling constraints.

F. Parameter Configuration

The parameter configuration used in this study is summarized in Table 8. The parameter values were not selected arbitrarily but were determined through a series of preliminary experiments. Multiple parameter combinations were evaluated to examine their effects on convergence behavior, solution quality, and computational time. The final configuration was empirically selected because it consistently produced feasible timetables with stable convergence while maintaining a reasonable computational cost. This empirical tuning strategy improves the reproducibility of the proposed optimization framework while providing a practical balance between exploration and exploitation.

Table 8. Parameter Settings

No	Parameter	Value
1	Population Size	50
2	Maximum Iteration	1500
3	Mutation Probability	0.8
4	Crossover Probability	0.7
5	Tournament Size	10
6	GWO Interval	20
7	GWO Iteration	20

Each parameter serves a specific purpose in controlling the optimization process. The population size was set to 50 because smaller populations produced insufficient solution diversity, whereas larger populations significantly increased computational time with only marginal improvements in timetable quality. The maximum iteration was fixed at 1500, allowing the evolutionary process sufficient opportunity to converge while avoiding unnecessary computational overhead after convergence had stabilized.

The crossover probability was set to 0.7 to maintain an effective balance between preserving high-quality scheduling structures and generating new offspring through recombination. Meanwhile, the mutation probability was assigned a relatively high value of 0.8 because the proposed mutation operator is constraint-guided rather than purely random. Frequent mutation enables the algorithm to correct dominant constraint violations, particularly teacher conflicts and subject-distribution constraints, without excessively disrupting feasible timetable structures.

A tournament size of 10 was selected to provide sufficient selection pressure, ensuring that individuals with better fitness values have a higher probability of reproduction while maintaining adequate population diversity throughout the optimization process.

The GWO interval was set to 20 generations, meaning that the Grey Wolf Optimizer is activated periodically rather than after every GA iteration. Applying GWO at every generation considerably increased computational cost with limited additional improvement, whereas longer intervals reduced the contribution of local exploitation. Executing GWO every 20 generations provided an effective compromise between the global exploration performed by the Genetic Algorithm and the local exploitation performed by the Grey Wolf Optimizer.

During each activation, the Grey Wolf Optimizer performs 20 iterations. This value was selected because it allows the alpha, beta, and delta leaders sufficient opportunity to refine promising timetable solutions without dominating the overall optimization process. Preliminary experiments indicated that increasing the number of GWO iterations produced only marginal improvements while noticeably increasing execution time. Therefore, 20 iterations provided

a practical balance between exploitation capability and computational efficiency.

G. Fitness Function

The optimization model introduced in the Problem Formulation subsection is implemented using a weighted-penalty fitness function to evaluate the quality of each candidate timetable throughout the evolutionary process. The objective of the optimization is to minimize the total penalty caused by constraint violations, where lower fitness values indicate better timetable quality. The individual with the lowest fitness value is selected as the best solution at the end of the optimization process.

The fitness value of each candidate timetable is computed by aggregating the weighted penalties associated with all evaluated constraints. The fitness function is defined as follows:

$$f(x) = \sum_{i=1}^{n} w_i \cdot C_i(x) \tag{3}$$

where $f(x)$ denotes the total fitness value of timetable solution (x). n represents the total number of scheduling constraints. w_i is the penalty weight assigned to constraint i. and $C_i(x)$ denotes the number of violations associated with constraint i in solution x. Since all constraints considered in this study are treated as hard constraints, every detected violation contributes directly to the total fitness value according to its assigned penalty weight.

The overall fitness value is obtained by summing the weighted penalties of all constraints. This penalty aggregation mechanism enables different types of scheduling violations to be evaluated simultaneously while preserving their relative importance during optimization. Consequently, candidate solutions with fewer violations obtain lower fitness values and are more likely to survive during the evolutionary process.

The weighted penalty configuration used in this study is summarized in Table 9. The penalty coefficients were determined according to the relative importance of each institutional scheduling constraint. Teacher Conflict receives the highest penalty because assigning a teacher to multiple classes simultaneously immediately invalidates the timetable and therefore represents the most critical scheduling violation. Subject Distribution and Teacher Consistency are assigned moderate penalties because they directly affect curriculum compliance and instructional continuity. Teacher Workload receives a lower penalty because workload imbalance primarily influences timetable quality and can generally be adjusted after major scheduling conflicts have been resolved. MGMP Time, Physics Subject, and Homeroom Teacher constraints receive smaller penalties because they mainly affect timetable quality rather than fundamental timetable feasibility.

No explicit weight normalization is applied because the proposed optimization relies on relative penalty magnitudes to prioritize constraint satisfaction during the evolutionary search. Consequently, violations of high-priority constraints contribute substantially larger penalties than lower-priority constraints, directing the optimization process to resolve the most critical scheduling conflicts before improving secondary timetable requirements.

Table 9. Weighted Penalty Configuration

No	Constraint	Weight
1	Teacher Conflict	100
2	Subject Distribution	20
3	Teacher Consistency	20
4	Teacher Workload	10
5	MGMP Time	5
6	Physics Subject	5
7	Homeroom Teacher	5

III. Results and Discussion

A. Best Fitness Improvement Rate

The optimization results presented in Table 10 demonstrate that all evaluated algorithms successfully reduce the initial fitness values, indicating their capability to improve timetable quality through iterative optimization. However, the improvement rate varies depending on the search mechanism and constraint-handling strategy employed by each method.

Table 10. Best Fitness Improvement Rate Comparison

No	Method	Initial Fitness	Final Fitness	Runtime	Improvement Rate
1	Traditional GA	19635	1995	8 min	89.83%
2	Guided GA	16340	930	13 min	94.31%
3	Guided GA-GWO	16120	670	28 min	95.84%
4	Original GWO	14565	5730	24 min	60.66%
5	PSO	22320	12505	27 min	43.97%
6	Simulated Annealing	17925	10305	0.2 min	42.51%
7	Tabu Search	21030	1490	9 min	92.91%

Traditional GA achieves an improvement rate of 89.83%, reducing the fitness value from 19,635 to 1,995. This improvement is mainly obtained through crossover and mutation operations that provide broad exploration of the search space. However, without specific guidance toward highly violated constraints, the algorithm tends to experience premature convergence.

The Guided GA improves the optimization performance by achieving a 94.31% improvement rate with a final fitness value of 930. The improvement is attributed to the constraint-aware mutation mechanism, which directs the search process toward resolving critical constraint violations and produces more feasible solutions.

The Hybrid Guided GA-GWO obtains the lowest final fitness value of 670 with an improvement rate of 95.84%. The integration of GWO provides additional population guidance through leader-based exploration, improving convergence stability and enabling better solutions. However, the improvement over Guided GA is relatively limited when considering the computational cost. The fitness reduction from 930 to 670 requires an increase in runtime from 13 to 28 minutes, indicating a trade-off between solution quality and computational efficiency.

Compared with other optimization methods, including Original GWO, PSO, Simulated Annealing, and Tabu Search, the proposed approaches achieve competitive fitness improvements. However, the results also indicate that higher algorithmic complexity does not always guarantee

proportional improvement, as simpler approaches such as Tabu Search can still achieve competitive performance.

B. Convergence Behavior

The convergence behavior of each optimization algorithm is presented in Fig. 9. Best Fitness Convergence Comparison. The convergence curve illustrates the ability of each method to reduce fitness values throughout the optimization process and provides insight into the balance between exploration and exploitation during the search process. A faster reduction in fitness indicates a stronger capability to discover promising regions of the solution space, while continuous improvement throughout iterations reflects better exploitation and avoidance of premature convergence.

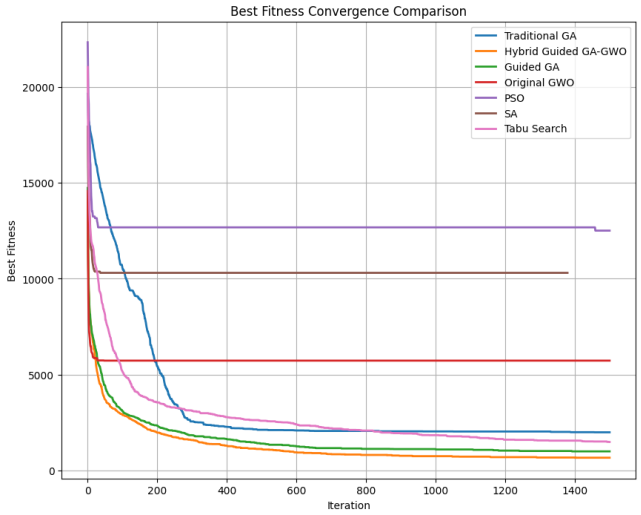

Fig. 9. Best Fitness Convergence Comparison

As shown in Fig. 9, each algorithm demonstrates different convergence characteristics during the optimization process. Traditional GA achieves a rapid fitness reduction in the early iterations due to the exploration capability of crossover and mutation operators. However, the convergence curve becomes nearly stagnant in later iterations, indicating premature convergence caused by the lack of guidance toward highly violated constraints.

Guided GA shows a similar initial reduction but continues improving gradually throughout the iterations. This behavior indicates that the constraint-aware mutation mechanism improves exploitation capability by directing the search toward critical constraint violations. The Hybrid Guided GA-GWO achieves the lowest fitness value among the evaluated variants, with a rapid initial reduction followed by continuous refinement. The additional guidance from GWO helps maintain search diversity and improves the ability to explore promising regions. However, this improved convergence behavior requires additional computational cost due to the integration of the GWO mechanism

The benchmark algorithms exhibit different convergence behaviors. Original GWO, PSO, and Simulated Annealing reduce fitness rapidly during early iterations but experience earlier stagnation, limiting further improvement. In contrast, Tabu Search demonstrates a more consistent reduction throughout iterations due to its memory-based search mechanism, although it does not achieve a lower final fitness compared with Hybrid Guided GA-GWO.

The average fitness convergence behavior of all evaluated algorithms is presented in Fig. 10. Average Fitness Convergence Comparison. Unlike the best fitness analysis, average fitness reflects the overall optimization progress and search stability of each algorithm during the iterative process. A consistent reduction in average fitness indicates that the algorithm is able to improve the overall population quality rather than relying only on individual best solutions.

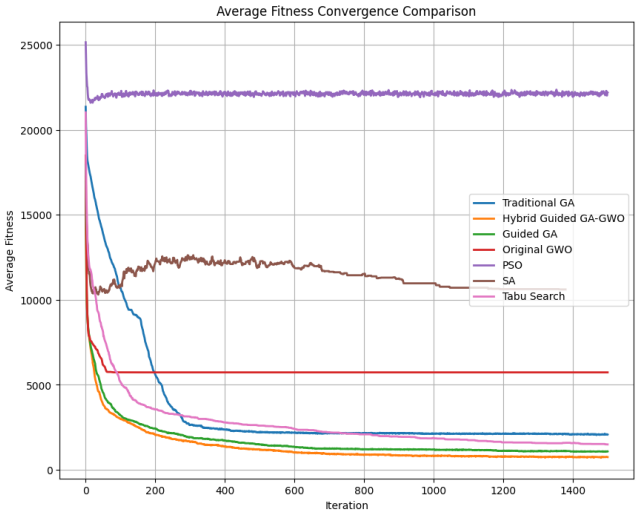

Fig. 10. Average Fitness Convergence Comparison

As shown in Fig. 10, Traditional GA achieves a significant reduction in average fitness during the early iterations, followed by a short stagnation period before continuing to improve. However, the improvement rate gradually decreases in the final iterations, indicating reduced search effectiveness as the population approaches convergence. Guided GA presents a more consistent reduction pattern, where the average fitness decreases rapidly at the beginning and continues improving gradually throughout the optimization process due to the constraint-aware mutation mechanism.

The Hybrid Guided GA-GWO demonstrates the strongest average convergence performance, achieving the lowest average fitness among the evaluated methods. The algorithm rapidly reduces average fitness in the initial iterations and continues gradual improvement until the end of optimization, indicating a better balance between exploration and exploitation. In contrast, Original GWO shows the fastest initial reduction but experiences early stagnation, suggesting limited capability for further refinement after reaching promising regions.

The remaining benchmark algorithms exhibit different convergence behaviors. PSO and Simulated Annealing achieve substantial initial reductions but show fluctuating average fitness values throughout later iterations, indicating unstable search behavior and limited improvement capability. Tabu Search demonstrates a more stable decreasing trend, with continuous improvement across iterations due to its memory-based search mechanism. However, its average fitness remains higher compared with Hybrid Guided GA-GWO.

C. Constraint Satisfaction

Constraint satisfaction analysis was conducted to evaluate the ability of each optimization algorithm to reduce violations of individual scheduling constraints rather than relying solely on the overall fitness value. Although the fitness function represents the total weighted penalty, analyzing each constraint separately provides a clearer understanding of the optimization behavior and identifies which scheduling requirements remain difficult to satisfy.

Table 11 and Table 12 present the average number of constraint violations before and after optimization for all evaluated algorithms. Before optimization, Teacher Consistency, Subject Distribution, and Physics Subject allocation exhibit the highest numbers of violations because these constraints involve multiple interdependent scheduling decisions, resulting in a highly constrained search space. In contrast, MGMP Time and Homeroom Teacher constraints produce fewer violations since they are associated with a smaller subset of scheduling activities.

After optimization, all algorithms substantially reduce constraint violations compared with their initial conditions. The Hybrid Guided GA-GWO achieves the lowest overall fitness value and produces the best results for several high-impact constraints, particularly Teacher Conflict and Teacher Consistency. However, its superiority is not consistent across all constraint categories. Guided GA achieves fewer violations in Teacher Workload, MGMP Time, and Homeroom Teacher constraints, indicating that the hybridization process favors global fitness optimization rather than uniformly improving every individual constraint.

Compared with the benchmark algorithms, Original GWO performs competitively on several constraints but tends to stagnate during later iterations, limiting its ability to further refine timetable quality. PSO records the highest violations in Teacher Conflict and Teacher Consistency, while Simulated Annealing exhibits fluctuating search behavior that reduces its effectiveness in satisfying multiple interacting constraints. Tabu Search achieves competitive improvements through its memory-based search mechanism; however, it remains less effective than the proposed Hybrid Guided GA-GWO in minimizing the overall weighted constraint violations.

The remaining violations also indicate the influence of the fitness weighting scheme used during optimization. Constraints assigned higher penalty weights, such as Teacher Conflict and Teacher Consistency, receive greater optimization priority and are resolved more effectively. Conversely, several lower-weight constraints, including MGMP Time and Homeroom Teacher assignment, retain residual violations in some algorithms. This suggests that the weighting strategy directly affects the search direction and the distribution of optimization effort across different scheduling constraints.

Table 11. Average of Initial Constraint Comparison

Method	Teacher Conflict	Subject Distribution	Teacher Consistency	Teacher Workload	MGMP Time	Physics Subject	Homeroom Teacher
Traditional GA	25	120	200	80	13	140	27
Guided GA	20	110	190	70	13	130	27
Hybrid Guided GA-GWO	18	105	185	65	13	125	27
Original GWO	42	95	141	88	35	78	40
PSO	116	170	290	103	30	40	28
Simulated Annealing	113	85	170	85	50	45	40
Tabu Search	119	150	220	98	40	60	50

Table 12. Average of Final Constraint Comparison

Method	Teacher Conflict	Subject Distribution	Teacher Consistency	Teacher Workload	MGMP Time	Physics Subject	Homeroom Teacher
Traditional GA	14	80	166	50	13	100	12
Guided GA	0	17	62	1	2	2	0
Hybrid Guided GA-GWO	0	21	20	36	10	5	11
Original GWO	0	45	204	51	16	22	10
PSO	40	119	270	52	12	21	8
Simulated Annealing	61	46	120	68	19	14	8
Tabu Search	0	34	27	20	0	1	13

D. Generated Schedule

Generated schedule is presented in Table 13, which illustrates the schedule for multiple classes on Monday. Each row represents a scheduled time slot, while each column corresponds to a class. Instructional periods are represented by pairs (S, T), where S denotes the subject identifier and T denotes the assigned teacher. Non-instructional activities, including the flag ceremony, reflective journal session, and break periods, are incorporated directly into the timetable to reflect the actual daily scheduling structure implemented.

The generated timetable demonstrates that instructional activities are systematically allocated across all classes while preserving the daily school timetable structure. Based on the optimization results presented in the previous subsection, the proposed method successfully eliminates teacher conflicts and produces a feasible schedule that satisfies the majority of high-priority scheduling constraints. The inclusion of fixed school activities within the timetable also illustrates the ability of the proposed method to accommodate institutional scheduling requirements without disrupting regular teaching sessions.

Table 13. Generated Weekly Schedule for Monday

Time Slot	7A	7B	7C	7D	7E
07:00 – 07:40	Flag Ceremony				
07:40 – 08:00	Reflective Journal Activity				
08:00 – 08:40	(2, 39)	(10, 15)	(8, 23)	(10, 31)	(3, 36)
08:40 – 09:20	(2, 39)	(10, 15)	(8, 23)	(10, 31)	(3, 36)
09:20 – 09:40	First Break				
09:40 – 10:20	(5, 48)	(9, 38)	(12, 51)	(3, 22)	(4, 47)
10:20 – 11:00	(5, 48)	(9, 38)	(12, 51)	(3, 22)	(4, 47)
11:00 – 11:40	(4, 47)	(3, 22)	(9, 42)	(3, 22)	(11, 37)
11:40 – 12:20	(4, 47)	(3, 22)	(9, 42)	(3, 22)	(11, 37)
12:20 – 13:20	Second Break				
13:20 – 14:00	(4, 47)	(5, 18)	(7, 21)	(4, 47)	(5, 48)
14:00 – 14:40	(4, 47)	(5, 18)	(7, 21)	(4, 47)	(5, 48)

Although the generated timetable achieves the best overall solution quality, several lower-priority constraints, including Teacher Workload, MGMP Time, and Homeroom Teacher assignments, still contain a small number of residual violations. This behavior is consistent with the weighted fitness function, which prioritizes constraints with higher penalty values during the optimization process. Consequently, the generated schedule represents a practical trade-off between minimizing overall constraint violations and maintaining computational efficiency.

The timetable also demonstrates that the proposed optimization framework is capable of producing a structured schedule that can be readily interpreted by users. However, the practical applicability of the generated timetable has not yet been evaluated by school administrators or compared with manually prepared schedules. Such evaluations would provide additional evidence regarding the usability and acceptance of the proposed approach and are therefore recommended for future work.

E. Representation Complexity

The proposed Dual-Vector Encoding is one of the main contributions of this study. Unlike conventional timetable representations that combine multiple scheduling components into a single data structure, the proposed encoding separates subject assignments and teacher assignments into two aligned matrices. This separation reduces variable interdependency during genetic operations, allowing modifications to one scheduling component with minimal impact on the other. As a result, constraint evaluation becomes more localized and repair operations are required less frequently during the optimization process.

To highlight the advantages of the proposed encoding, Table 14 compares the computational characteristics of several commonly used timetable representations. Single-vector representation is computationally efficient during genetic operations but suffers from strong dependency among scheduling variables, making constraint evaluation increasingly expensive as the number of constraints grows. The 2D array representation provides a natural timetable structure but requires crossover and mutation to manipulate multiple rows and columns simultaneously. Meanwhile, nested-array representation improves modularity but introduces additional traversal overhead during evaluation due to its hierarchical organization.

The proposed Dual-Vector Encoding maintains separate matrices for subject and teacher assignments with an overall representation size of $O(C \times S)$, where C denotes the number of classes and S denotes the number of time slots. Based on the implemented evaluation procedures, mutation can be performed in constant time, while crossover remains linear with respect to the timetable size. Constraint evaluation requires approximately $O(C \times S \log S)$, mainly due to grouping and aggregation operations performed during timetable validation. Although the representation requires additional memory compared with conventional encodings, the separation of scheduling variables substantially simplifies constraint handling throughout the optimization process.

The experimental results presented in the previous subsections support the effectiveness of the proposed representation. The reduced dependency between subject and teacher assignments contributes to the stable convergence behavior observed in Fig. 9 and Fig. 10, enabling the Hybrid Guided GA-GWO to continue improving solution quality after the initial exploration phase. Furthermore, the simplified constraint evaluation facilitates the guided mutation and GWO-based refinement processes, which together produce the lowest final fitness value among the evaluated algorithms. These findings indicate that the additional memory overhead introduced by the Dual-Vector Encoding represents a reasonable trade-off for achieving improved convergence stability and timetable quality.

Table 14. Comparison of Representation Complexity

No	Representation	Complexity	Evaluation	Weakness
1	Single Vector	$O(n)$	$O(n \cdot c)$	High Dependency
2	2D Array	$O(C \times S)$	$O(C \times S)$	Expensive Crossover
3	Nested Array	$O(n)$	$O(n \log n)$	Tranversal Overhead
4	Dual-Vector	$O(C \times S)$	$O(C \times S \log S)$	Memory Overhead

F. Discussion

The experimental results demonstrate that the proposed Hybrid Guided GA-GWO achieves the best overall solution quality among the evaluated algorithms by producing the lowest final fitness value and maintaining stable convergence throughout the optimization process. Compared with Traditional GA, the integration of constraint-aware mutation significantly improves the exploitation capability by directing the search toward highly violated constraints. Furthermore, the incorporation of the Grey Wolf Optimizer enhances global exploration through leader-based guidance, enabling the population to continue refining solutions after the initial search phase and reducing the likelihood of premature convergence.

Although the proposed hybrid method achieves the lowest overall fitness value, its superiority is not uniform across all scheduling constraints. The Hybrid Guided GA-GWO performs particularly well on high-priority constraints, including Teacher Conflict and Teacher Consistency, which contribute substantially to the overall fitness value. However, Guided GA produces fewer violations for Teacher Workload, MGMP Time, and Homeroom Teacher constraints. This observation indicates that the hybridization process primarily optimizes the weighted objective function rather than minimizing every individual constraint simultaneously. Consequently, the weighting scheme plays a critical role in determining the optimization priorities and the distribution of residual violations.

The proposed Dual-Vector Encoding also contributes to the overall optimization performance by reducing the dependency between subject assignments and teacher assignments. This representation simplifies constraint evaluation and enables more effective genetic operations without extensively disrupting feasible timetable components. The stable convergence behavior observed in Fig. 9 and Fig. 10 suggests that the proposed representation supports both the guided mutation strategy and the GWO refinement process, thereby improving search stability and solution quality despite introducing additional memory overhead.

A trade-off is observed between solution quality and computational cost. Although the Hybrid Guided GA-GWO improves the final fitness value from 930 to 670 compared with Guided GA, the execution time increases from approximately 13 minutes to 28 minutes. This indicates that the additional computational effort produces a relatively modest improvement in the objective function. Therefore, the proposed hybrid method is more appropriate for timetabling scenarios where obtaining higher-quality schedules is prioritized over execution time, whereas Guided GA may provide a more practical alternative when faster optimization is required.

IV. Conclusion

This study proposed a Hybrid Guided Genetic Algorithm-Grey Wolf Optimizer (Hybrid Guided GA-GWO) with Dual-Vector Encoding to address the Indonesian school timetabling problem involving multiple hard constraints. Experimental evaluation demonstrates that the proposed method reduced the fitness value from 16,120 to 670, corresponding to an improvement rate of 95.84%, which represents the best overall solution quality among the evaluated algorithms. The combination of constraint-aware mutation, GWO-based leader guidance, and Dual-Vector Encoding improves convergence stability and enhances the optimization of highly constrained scheduling problems.

Despite achieving the lowest overall fitness value, the proposed method does not consistently outperform the compared algorithms across every individual constraint and requires substantially longer execution time than Guided GA. These findings indicate that the proposed hybrid framework improves the weighted objective function at the expense of increased computational cost, representing a trade-off between solution quality and optimization efficiency. From a practical perspective, the proposed approach has the potential to support educational management systems by generating higher-quality timetables while reducing manual scheduling effort for complex school environments.

The present study has several limitations. The experimental evaluation was conducted using a single case-study dataset, and the reported results are based on a single execution for each algorithm without statistical validation. Although several additional metaheuristic algorithms were included for comparison, the evaluation has not yet been validated using public timetabling benchmark datasets. Furthermore, the performance of the proposed approach remains dependent on parameter settings and incurs higher computational cost than simpler optimization methods.

Future work should therefore investigate adaptive parameter tuning to improve robustness, parallel implementations to reduce execution time, provide additional evidence, and evaluation on larger-scale and publicly available timetabling benchmark datasets. Incorporating statistical significance analysis through multiple independent runs and practical validation with school administrators would further strengthen the reliability and applicability of the proposed framework for real-world educational scheduling systems.

Acknowledgment

The author expresses sincere gratitude to Allah SWT for His blessings and grace, which made this research possible. The author would also like to extend heartfelt appreciation to family, friends, and lecturers for their continuous support, guidance, and encouragement throughout the completion of this study.

References

[1] N. Pillay, "A survey of school timetabling research," *Annals of Operations Research*, vol. 218, no. 1, pp. 261–293, 2014, https://doi.org/10.1007/s10479-013-1321-8.

[2] I. Gusti Agung Premananda, A. Tjahyanto, and A. Muklason, "Timetabling Problems and the Effort Toward Generic Algorithms: A Comprehensive Survey," *IEEE Access*, vol. 12, pp. 143854–143868, 2024, https://doi.org/10.1109/ACCESS.2024.3463721.

[3] S. Ceschia, L. Di Gaspero, and A. Schaerf, "Educational timetabling: Problems, benchmarks, and state-of-the-art results," *European Journal of Operational Research*, vol. 308, no. 1, pp. 1–18, 2023, https://doi.org/10.1016/j.ejor.2022.07.011.

[4] A. Schaerf, "Survey of automated timetabling," *Artificial Intelligence Review*, vol. 13, no. 2, pp. 87–127, 1999, https://doi.org/10.1023/A:1006576209967.

[5] M. Kaur and S. Saini, "A review of metaheuristic techniques for solving university course timetabling problem," in *Lecture Notes in Networks and Systems*, vol. 135, V. Goar, M. Kuri, R. Kumar, and T. Senjyu, Eds., Springer, pp. 19–25, 2021, https://doi.org/10.1007/978-981-15-5421-6_3.

[6] R. Hoshino and I. Fabris, "Optimizing Student Course Preferences in School Timetabling," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, E. Hebrard and N. Musliu, Eds., Springer, pp. 283–299, 2020, https://doi.org/10.1007/978-3-030-58942-4_19.

[7] 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 Systems with Applications*, vol. 165, p. 113943, 2021, https://doi.org/10.1016/j.eswa.2020.113943.

[8] X. Gu, M. Krish, S. Sohail, S. Thakur, F. Sabrina, and Z. Fan, "From Integer Programming to Machine Learning: A Technical Review on Solving University Timetabling Problems," *Computation*, vol. 13, no. 1, p. 10, 2025, https://doi.org/10.3390/computation13010010.

[9] P. T. Nsulangi, W. E. Ngongi, M. R. Likamba, O. B. Sarehe, and M. A. Mkwande, "A Comparative Analysis of Manual and Automatic Timetabling Approaches for Resource Utilisation in Tertiary Higher Learning Institution," *International Journal of Computer Science and Mobile Computing*, vol. 13, no. 12, pp. 65–76, 2024, https://doi.org/10.47760/ijcsmc.2024.v13i12.007.

[10] T. Birbas, S. Daskalaki, and E. Housos, "School timetabling for quality student and teacher schedules," *Journal of Scheduling*, vol. 12, no. 2, pp. 177–197, 2009, https://doi.org/10.1007/s10951-008-0088-2.

[11] R. Saltos and S. Maldonado, "School Timetabling Problem: A Scheduling Problem for High-School Institutions," *INFORMS Transactions on Education*, vol. 24, no. 1, pp. 95–99, 2023, https://doi.org/10.1287/ited.2022.0276ca.

[12] M. H. Cruz-Rosales *et al.*, "Metaheuristic with Cooperative Processes for the University Course Timetabling Problem," *Applied Sciences (Switzerland)*, vol. 12, no. 2, p. 542, 2022, https://doi.org/10.3390/app12020542.

[13] A. R. Mahlous and H. Mahlous, "Student timetabling genetic algorithm accounting for student preferences," *PeerJ Computer Science*, vol. 9, p. e1200, 2023, https://doi.org/10.7717/peerj-cs.1200.

[14] R. P. Badoni *et al.*, "An Exploration and Exploitation-Based Metaheuristic Approach for University Course Timetabling Problems," *Axioms*, vol. 12, no. 8, 2023, https://doi.org/10.3390/axioms12080720.

[15] M. Davison, G. Burgess, R. Hamm, B. Lowery, and A. Page, "A parallelised hyper-heuristic framework for the Integrated Healthcare Timetabling Competition 2024," *SSRN*, 2025, https://doi.org/10.2139/ssrn.5601273.

[16] B. Alhijawi and A. Awajan, "Genetic algorithms: theory, genetic operators, solutions, and applications," *Evolutionary Intelligence*, vol. 17, no. 3, pp. 1245–1256, 2024, https://doi.org/10.1007/s12065-023-00822-6.

[17] T. Alam, S. Qamar, A. Dixit, and M. Benaida, "Genetic algorithm: Reviews, implementations and applications," *International Journal of Engineering Pedagogy*, vol. 10, no. 6, pp. 57–77, 2021, https://doi.org/10.3991/IJEP.V10I6.14567.

[18] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press, 1992, https://doi.org/10.7551/mitpress/1090.001.0001.

[19] M. H. Hashem, H. S. Abdullah, and K. I. Ghathwan, "Grey Wolf Optimization Algorithm: A Survey," *Iraqi Journal of Science*, vol. 64, no. 11, pp. 5964–5984, 2023, https://doi.org/10.24996/ijs.2023.64.11.40.

[20] B. Li and X. Xia, "A Self-Adjusting Search Domain Method-Based Genetic Algorithm for Solving Flexible Job Shop Scheduling Problem," *Computational Intelligence and Neuroscience*, vol. 2022, 2022, https://doi.org/10.1155/2022/4212556.

[21] K. Sarra, Z. Djamel, and M. Smaine, "A Novel Method for Solving the University Course Timetabling Problem Based on the Grey Wolf Optimizer Algorithm," in *2023 3rd International Conference on Theoretical and Applicative Aspects of Computer Science, ICTAACS 2023*, pp. 1–8, 2023, https://doi.org/10.1109/ICTAACS60400.2023.10449623.

[22] K. Li, S. Li, Z. Huang, M. Zhang, and Z. Xu, "Grey Wolf Optimization algorithm based on Cauchy-Gaussian mutation and improved search strategy," *Scientific Reports*, vol. 12, no. 1, 2022, https://doi.org/10.1038/s41598-022-23713-9.

[23] I. A. Abduljabbar and S. M. Abdullah, "An evolutionary algorithm for solving academic courses timetable scheduling problem," *Baghdad Science Journal*, vol. 19, no. 2, pp. 399–408, 2022, https://doi.org/10.21123/BSJ.2022.19.2.0399.

[24] Q. Zhang, "An optimized solution to the course scheduling problem in universities under an improved genetic algorithm," *Journal of Intelligent Systems*, vol. 31, no. 1, pp. 1065–1073, 2022, https://doi.org/10.1515/jisys-2022-0114.

[25] H. Erdoğan Akbulut, F. Özçelik, and T. Saraç, "A simulated annealing algorithm for the faculty-level university course timetabling problem," *Pamukkale University Journal of Engineering Sciences*, vol. 30, no. 1, pp. 17–30, 2024, https://doi.org/10.5505/pajes.2023.00483.

[26] H. Yan, Z. Cui, X. Chen, and X. Ma, "Distributed Multiagent Deep Reinforcement Learning for Multiline Dynamic Bus Timetable Optimization," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 469–479, 2023, https://doi.org/10.1109/TII.2022.3158651.

[27] X. Han and D. Wang, "Gradual Optimization of University Course Scheduling Problem Using Genetic Algorithm and Dynamic Programming," *Algorithms*, vol. 18, no. 3, 2025, https://doi.org/10.3390/a18030158.

[28] D. Abramson, "Constructing school timetables using simulated annealing. Sequential and parallel algorithms," *Management Science*, vol. 37, no. 1, pp. 98–113, 1991, https://doi.org/10.1287/mnsc.37.1.98.