Metaheuristic Approaches for Solving the School Timetabling Problem

1st Thiago E. dos Santos, 2nd Heber F. Amaral, 3rd Antonio R. Santana
Instituto Federal do Sudeste de Minas Gerais
Bom Sucesso, Brasil
thiaagosantos33@gmail.com, heber.amaral@ifsudestemg.edu.br, antonio.santana@ifsudestemg.edu.br

***Abstract*—This paper investigates the application of metaheuristic techniques to solve the School Timetabling Problem (STP), a classic NP-hard combinatorial optimization challenge. Specifically, it addresses the scheduling needs of a Brazilian federal institution (IF Sudeste MG), whose timetable structure does not fully conform to standard STP formulations, thus requiring customized constraint modeling and algorithmic adjustments. The proposed approach integrates a Randomized Constructive Heuristic with local search-based metaheuristics: Iterated Local Search (ILS), Variable Neighborhood Search (VNS), and Iterated Tabu Search (ITS). The problem formulation considers both hard and soft constraints, which are incorporated into a penalized cost function. Experiments conducted on real-world instances from four academic terms show that all proposed metaheuristics generate feasible solutions. Among them, ITS achieved the best average performance, while ILS demonstrated superior efficiency in limited time scenarios. The results confirm the effectiveness of metaheuristic strategies in generating high-quality solutions for realistic and institution-specific timetabling problems.**

***Index Terms*—Metaheuristics, Combinatorial Optimization, ILS, VNS, ITS, Timetabling.**

I. Introduction

The School Timetabling Problem (STP) involves assigning classes to available time slots and rooms, subject to constraints that ensure the feasibility of the academic schedule. According to [1], the problem requires that no classroom be assigned to more than one class simultaneously, and that no teacher be scheduled to teach more than one lesson during the same time period. Manually constructing such timetables is a complex and often time-consuming task due to the large number of possible combinations, which may lead to suboptimal solutions.

Automating this process brings substantial benefits in terms of operational efficiency by enabling the identification of more appropriate combinations of courses, instructors, and available physical resources. Such automation helps reduce idle periods and contributes to more continuous and productive routines, positively impacting the quality of education and the satisfaction of both students and faculty [2]. Combinatorial optimization techniques make it possible to address constraints such as teacher preferences, room availability, and undesired scheduling gaps. The elimination or minimization of such gaps improves time management and promotes the rational use of institutional space.

This work was supported by the Brazilian National Council for Scientific and Technological Development (CNPq) and the Minas Gerais Research Support Foundation (FAPEMIG), whose contributions were essential through the institutional undergraduate research program at IF Sudeste MG.

The STP is widely recognized as a classical NP-hard combinatorial optimization problem [3], making it impractical to obtain optimal solutions using exact methods within polynomial time. Consequently, the use of heuristics and metaheuristics has become common practice, as they can yield high-quality solutions within acceptable computational time even though they do not guarantee optimality.

While existing literature, as reviewed in the following section, extensively explores various metaheuristic approaches for different timetabling problem variants, many studies rely on specific datasets like those from the International Timetabling Competition (ITC) or address university-specific constraints. This work distinguishes itself by investigating the application of the metaheuristics Iterated Local Search (ILS), Variable Neighborhood Search (VNS), and Iterated Tabu Search (ITS) to solve the STP, taking into account the unique operational particularities of a real-world federal educational institution, which involve a hybrid structure combining features of both high school and university timetabling problem. Specifically, our problem formulation includes a set of hard and soft constraints that partially differ from those commonly found in standard benchmarks, necessitating a tailored approach to generate robust and feasible solutions for practical deployment.

The remainder of this paper is structured as follows: Section II reviews related work; Section III introduces classical problem variants and details the adopted modeling; Section IV discusses the metaheuristic approaches used; experimental results are presented and analyzed in Section V; finally, Section VI presents the conclusions and directions for future work.

II. Related Work

Several studies in the literature address the School Timetabling Problem (STP) using metaheuristic techniques. This section highlights relevant research that has contributed to the advancement of methodologies applied to this context.

In the work of [4], the *Simulated Annealing* and *Iterated Local Search* (ILS) techniques are applied to the *High School Timetabling Problem* (HSTP), solving instances from the *International Timetabling Competition* (ITC). The initial solution is generated using the KHE instance manipulation platform proposed in [5], which is capable of producing high-quality

starting points. The results showed that ten out of seventeen generated solutions were feasible, and seven out of twenty-one instances outperformed the best-known solutions at the time.

In [6], a solution is presented for the scheduling problem of the Department of Computing at CCA-UFES. The method employs the GRASP metaheuristic to generate an initial solution and refines it using *Simulated Annealing*. The proposed algorithm produced feasible solutions that outperformed manually constructed timetables within an acceptable computational time.

The work by [7] addresses the problem by dividing it into two subproblems: classroom allocation and timetable scheduling, applied to IME/UERJ. The initial solution is constructed by assigning each event to a feasible time slot and a compatible classroom, treating both subproblems simultaneously. The metaheuristics used to improve this initial solution include classical *Simulated Annealing* and *Late Acceptance Hill-Climbing*, which compares neighboring solutions with previously accepted ones stored in a historical list.

In the study by [8], ILS is combined with the *maxmatch* perturbation strategy, starting from an initial solution generated through a greedy procedure that schedules events period by period. The study uses a dataset of 60 instances developed in [9], obtaining feasible solutions for 58 of them and demonstrating that ILS significantly outperforms other algorithms, particularly for more complex instances.

In [10], Vanden Berghe et al. propose a graph-based hyper-heuristic framework for solving university course timetabling problems, emphasizing the integration of GRASP, VNS, and hybrid approaches. The study highlights the flexibility of the proposed model, which allows for various heuristic selection and acceptance strategies within a high-level search structure. Through experiments conducted on instances from the International Timetabling Competition (ITC), the authors demonstrate that hybrid combinations within the hyper-heuristic framework can produce high-quality solutions, effectively adapting to the specific features of each instance.

This work proposes a solution for the School Timetabling Problem tailored to the specific constraints of a real educational institution, which partially differ from those defined in the ITC and the previously cited works. The proposed approach starts from an initial solution generated by a Randomized Constructive Heuristic and refines it using local search-based techniques, namely ILS, VNS, and ITS.

III. Problem Description

The *High School Timetabling Problem* (HSTP) is a variant of the scheduling problem that operates with fixed structures. In this model, students remain in the same classroom, while teachers move between different rooms. This structure is typical of secondary education institutions. Additionally, subject workloads are standardized across multiple classes of the same grade level [11].

The *University Course Timetabling Problem* (UCTP), on the other hand, presents a more flexible structure with respect to students, time slots, and classrooms. A single student may be enrolled in multiple classes, since each course enrollment generates a distinct class. In this context, both students and instructors move between rooms, and one of the main objectives of the UCTP is to minimize this movement between consecutive classes [12].

The *Post-Enrolment Course Timetabling Problem* (PE-CTT) involves the allocation of lectures after the student enrollment process has been completed. This variant primarily focuses on minimizing individual student conflicts, reducing the number of gaps in schedules, and limiting consecutive lectures. Although the PE-CTT is a distinct formulation, its most critical constraints remain relatively consistent with those of the other variants [13].

The institution studied in this work does not perfectly fit into either of the two main formulations but instead adopts features from both. The structure of class groupings resembles the HSTP, as the schedule is created by the institution itself. However, as in the UCTP, lectures can be assigned to different rooms, requiring student movement between them. Moreover, IF Sudeste MG offers multiple educational levels—including integrated technical education (with high school), subsequent technical education, and higher education—which adds another layer of complexity to the problem. Faculty members often teach across more than one modality, increasing the number of constraints to be respected and the scheduling conflicts to be avoided. Given that the problem formulation aligns more closely with HSTP. Student-level considerations are only introduced in exceptional cases, such as when offering courses outside the standard curriculum. In such cases, a special class is created with a schedule that avoids conflicts with regular courses.

A. Constraints

In combinatorial optimization problems [14], constraints are generally divided into two categories: hard constraints, which define solution feasibility, and soft constraints, which influence solution quality. Violating any hard constraint renders a solution infeasible, regardless of its associated cost. Soft constraints, on the other hand, may be violated but incur penalties based on their severity. Although the solution remains feasible, its overall cost may increase significantly.

The **hard constraints** are:

- HC1 All lectures must be scheduled.
- HC2 Two or more subjects assigned to the same class group must not be scheduled at the same time.
- HC3 No teacher should be assigned to teach more than one class in the same time slot.
- HC4 A classroom cannot host more than one lecture simultaneously.
- HC5 Preassigned lectures must not be altered in terms of time or room.
- HC6 Lectures must be scheduled in the same session (e.g., morning or evening) as the offering.
- HC7 The total number of weekly lectures allocated to each subject must match its required workload.

The **soft constraints** are:

SC1 The number of gaps between lectures of the same subject should be minimized.
SC2 Teachers' preferred time slots should be respected.
SC3 Teachers' workloads should be concentrated on their preferred days.
SC4 Courses requiring specific resources should preferably be assigned to rooms that fulfill those requirements.

B. Penalties

TABLE I
PENALTY VALUES FOR EACH CONSTRAINT TYPE

Constraint	Penalty
HC*	10,000
SC1	5
SC2	25
SC3	20
SC4	20

In this work, a relaxed version of the School Timetabling Problem is adopted, in which hard constraints are incorporated into the objective function through the assignment of high penalty costs, rather than being strictly enforced as constraints. This approach allows the use of metaheuristics without requiring strict feasibility throughout the search process. A fixed penalty of 10,000 units is applied for each violation of a hard constraint, strongly discouraging infeasible solutions. Soft constraints are also associated with penalty values, adjusted according to their frequency and practical importance. For instance, SC1—being more likely to occur—is assigned a higher weight relative to the others. Table I summarizes the penalty values assigned to each type of soft constraint violation.

C. Problem Formulation

The objective of the model is to minimize the weighted sum of violations of hard and soft constraints. The objective function is defined as follows:

$$\begin{aligned}\min f(s) = &\sum_{k=1}^{|HC|} \alpha \cdot HC_k(s) \\ &+ \beta_1 \sum_{j=1}^{|D|}\sum_{k=1}^{|H|}\sum_{m=1}^{|I|} w_j(D_j, H_k, I_m) \\ &+ \beta_4 \sum_{j=1}^{|D|}\sum_{k=1}^{|H|}\sum_{m=1}^{|I|} w_d(D_j, H_k, I_m) \\ &+ \beta_2 \sum_{l=1}^{|P|}\sum_{n=1}^{|H|}\sum_{p=1}^{|I|} w_p(P_l, H_n, I_p) \\ &+ \beta_3 \sum_{l=1}^{|P|}\sum_{n=1}^{|H|}\sum_{p=1}^{|I|} w_r(P_l, H_n, I_p) \end{aligned} \quad (1)$$

Where:

- s: set of classrooms;
- H: set of time slots;
- D: set of subjects;
- I: set of all lectures across all subjects;
- P: set of teachers;
- HC: set of hard constraints;
- $HC_k(s)$: number of violations of the k-th hard constraint in solution s;
- $w_j(D_j, H_k, I_m)$: returns 1 if subject D_j has a gap at time H_k between its first and last scheduled lectures;
- $w_d(D_j, H_k, I_m)$: returns 1 if a lecture of D_j is allocated to a room without the required resources;
- $w_p(P_l, H_n, I_p)$: returns 1 if a lecture for teacher P_l is scheduled at an unavailable time slot;
- $w_r(P_l, H_n, I_p)$: returns 1 if a lecture for teacher P_l is scheduled outside their preferred teaching days;
- α: fixed penalty weight for each hard constraint violation;
- $\beta_1, \beta_2, \beta_3, \beta_4$: penalty weights for violations of soft constraints SC1, SC2, SC3, and SC4, respectively.

D. Neighborhood Structure

The algorithm employs several auxiliary data structures, the most important of which is a 2D matrix representing the **Room × Time Slot** relationship. Each timeslot in room j at time k can store an event e_i, where each event corresponds to a weekly lecture of a subject. This structure ensures that hard constraint HC4 is never violated, since it is not possible to allocate more than one event in the same timeslot and room within the timetable matrix.

The exploration of the search space is performed through neighborhood moves that alter the current allocation by randomly selecting events (lectures) and resources (rooms and time slots). The two neighborhood moves used are described below:

- ***Move.*** An event a_1 is removed from a timeslot t_1 and allocated to an empty timeslot t_2, selected at random.

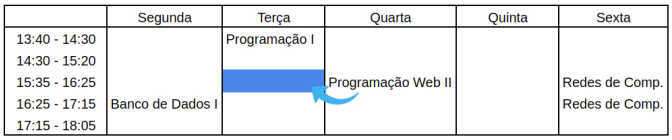

Fig. 1. Example of a *Move* neighborhood operation

- ***Swap.*** Two events, a_1 and a_2, are randomly selected and exchanged between their respective timeslots.

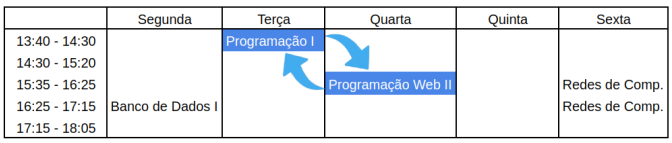

Fig. 2. Example of a *Swap* neighborhood operation

The choice of which move to apply is made randomly, but according to predefined probabilities: in the current configuration, 60% of the iterations apply the *Move* operation, while

40% apply the *Swap*. However, this ratio can be adjusted depending on the characteristics of the instance. In scenarios with a high number of occupied timeslots and limited availability of empty slots, the *Swap* operation tends to yield better results. Conversely, when many timeslots are available, the *Move* operation is generally more effective.

IV. Heuristic Methods

Local search-based heuristic methods aim to improve a current solution by exploring its neighborhood and moving to a new solution if it is better. However, metaheuristics require an initial feasible solution to operate effectively. Therefore, an initial solution generator adapted to the characteristics of the problem was developed.

A. Randomized Constructive Heuristic with Repair

Based on the approach proposed by [15], the Randomized Constructive Heuristic (RCH), as presented in Algorithm 1, iterates over a set of unscheduled events, assigning each event to a randomly selected room j and time slot k, while verifying the feasibility of the solution after each allocation. Frequently, the algorithm encounters events for which no conflict-free timeslot is available. In such cases, a repair mechanism is triggered. A conflicting resource—such as a teacher, room, or time—is selected and reallocated. If the conflict persists, a conflicting event is randomly selected and returned to the list of unallocated lectures, allowing the current event to be inserted into the desired timeslot.

Algorithm 1: Randomized Constructive Heuristic

```
Input: Instance I
Output: Feasible solution s
LC ← ListOfUnscheduledLectures();
while |LC| > 0 do
    Select a random lecture a_i;
    Select a random time slot k and room j;
    if s is feasible then
        Allocate a_i in the original matrix;
        Remove a_i from LC;
    else
        RepairSolution();
return s;
```

B. Iterated Local Search

Iterated Local Search (ILS), as presented in Algorithm 2, is a variant of basic local search that aims to escape local optima by perturbing the current solution. The algorithm receives a solution s already improved through local search. After a fixed number of iterations without improvement, it applies a perturbation operation that generates a new solution s', followed by a new local search phase resulting in solution s''. After each perturbation and local improvement, s'' is compared to s, and if better, s is updated to s''. Although the perturbation may temporarily increase the cost, it enables the algorithm to explore new regions of the search space [16].

Algorithm 2: *Iterated Local Search*

```
Input: s_0, Time_max
Output: best s
s ← LocalSearch(s_0);
while Time < Time_max do
    s' ← Perturb(s);
    s'' ← LocalSearch(s');
    if f(s'') ≤ f(s) then
        Iter ← 0;
        s ← s'';
return s;
```

C. Variable Neighborhood Search

Variable Neighborhood Search (VNS), as presented in Algorithm 3, as defined in [17], operates over a series of neighborhood structures $(N_1, N_2, ..., N_k)$ and consists of two main phases: a shaking phase, which introduces a controlled perturbation to generate a random neighbor by applying the k-th neighborhood operation, and a local search phase, which applies the same procedure described in the previous section.

Algorithm 3: *Variable Neighborhood Search*

```
Input: s_0, Time_max
Output: best s
s ← LocalSearch(s_0);
while Time < Time_max do
    s' ← Shake(s);
    s'' ← LocalSearch(s');
    if f(s'') ≤ f(s) then
        Iter ← 0;
        s ← s'';
return s;
```

D. Iterated Tabu Search

Originally proposed in [18], *Tabu Search* (Algorithm 4) is a memory-based local search metaheuristic that maintains a tabu list to store previously visited moves, thereby preventing cycles and promoting diversification. The algorithm starts with an empty tabu list of fixed size. Each newly performed move is added to the list, and the oldest entries are discarded as needed to make room for new ones.

To approach the global optimum, [19] enhanced the classic Tabu Search by proposing the *Iterated Tabu Search* (ITS) (Algorithm 5). This method combines the perturbation phase of ILS with the memory-based mechanism of Tabu Search. After each perturbation and TS phase, the resulting solution s'' is compared with the current one s, and if it is better, s is updated accordingly—similar to the acceptance criterion in ILS with local search.

Algorithm 4: *Tabu Search*

Input: s_0, $Time_{max}$
Output: best s
$\mathcal{T} \leftarrow \emptyset$;
$s \leftarrow s_0$;
$Iter \leftarrow 0$;
while $Iter < Iter_{max}$ **do**
  $\mathcal{N} \leftarrow$ GenerateNeighborhood(S);
  $m^* \leftarrow$ SelectBestMove$(\mathcal{N}, \mathcal{T})$;
  $s' \leftarrow$ ApplyMove(s, m^*);
  if $f(s') \leq f(s)$ **then**
    $s \leftarrow s'$;
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{m^*\}$;
  UpdateTabuList$(\mathcal{T})$;
  $Iter \leftarrow Iter + 1$;
return s;

Algorithm 5: *Iterated Tabu Search*

Input: s_0, $Time_{max}$
Output: best s
$s \leftarrow TabuSearch(s_0)$;
while $Time < Time_{max}$ **do**
  $s' \leftarrow Perturb(s)$;
  $s'' \leftarrow TabuSearch(s')$;
  if $f(s'') \leq f(s)$ **then**
    $Iter \leftarrow 0$;
    $s \leftarrow s''$;
return s;

V. Results

All experiments were conducted on a computer equipped with a Ryzen 5 5600G processor clocked at 3.9GHz and 16GB of RAM. The implementations were developed in C++ and compiled using G++ 11.4.0, running on Ubuntu 22.04 LTS.

For the experiments, data provided by the educational institution for academic terms from 2023/1 to 2024/2 were used. These datasets included information about instructors, student groups, available rooms, time slots, and scheduling preferences. Table II summarizes the characteristics of the instances used in the experiments. Distance learning (EAD) lectures were excluded from the allocation process.

TABLE II
Characteristics of the Institutional Instances by Academic Term

Term	Teachers	Rooms	Class Groups	Courses	Lectures
2024/2	16	9	9	53	131
2024/1	17	9	10	60	157
2023/2	15	9	11	61	154
2023/1	16	9	10	81	168

A. Results Analysis

Although feasible, the initial solutions produced by the RCH algorithm exhibited significantly higher costs compared to those refined by metaheuristic methods, as shown in Table III.

TABLE III
Solution Obtained by RCH

Term	$f(s)$
2024/2	4110
2024/1	4305
2023/2	4120
2023/1	3685

For the ILS metaheuristic, an adaptive perturbation technique was employed. Initially, 3% of the solution is perturbed. If no improvement is observed after 50 perturbations, the perturbation rate increases to 5%. After 100 iterations without improvement, a more aggressive perturbation of 10% is applied. In the ITS, the same perturbation strategy as ILS is adopted, but with a fixed perturbation rate of 5% and a tabu list size of 100 moves. For the VNS, the neighborhoods alternate systematically between *Move* and *Swap* operations throughout the 300-second runtime. Table IV presents the results obtained after executing each method under a 300-second time limit. The table reports the best solution cost $f(s^*)$ and the average solution cost $f(\bar{s})$ for each metaheuristic.

TABLE IV
Results Obtained by Metaheuristcs

	ILS		VNS		ITS	
Período	$f(s^*)$	$f(\bar{s})$	$f(s^*)$	$f(\bar{s})$	$f(s^*)$	$f(\bar{s})$
2024/2	40	72,5	40	81,9	40	**42,7**
2024/1	255	370,3	255	391,2	255	**346,5**
2023/2	245	391,8	245	406,5	245	**252,1**
2023/1	340	458,7	340	470,9	340	**351,1**

While RCH provided feasible initial solutions for all instances, their costs were considerably higher. The use of ILS, VNS, and ITS led to solutions that ranged from 1% to 10% of the cost obtained by RCH. Although all three metaheuristics achieved comparable best-case results, ITS consistently yielded the best average performance, outperforming ILS, which in turn outperformed VNS across all instances.

To further analyze performance, the TTTplots technique [20] was applied to visualize the probability of each method reaching a predefined target solution cost within a given time frame.

A total of 200 independent runs were executed for each algorithm and each instance, with a maximum execution time of 900 seconds per run. The target cost was set to 10% above the best-known solution for most instances, except for the 2024/2 instance, for which a 13% margin was used due to its notably lower baseline cost, which made target attainment more challenging.

For the 2023/1 instance, all runs of ITS reached the target. Eight runs failed to reach the target with ILS, and six with

VNS. For 2023/2, fifteen ILS and eleven VNS runs failed, while ITS reached the target in all runs. For 2024/1, thirteen ILS, seven VNS, and two ITS runs failed to reach the target. Lastly, for 2024/2—which presented the most challenging scenario—twenty-six ILS and nineteen VNS runs failed, while all ITS runs succeeded in reaching the target.

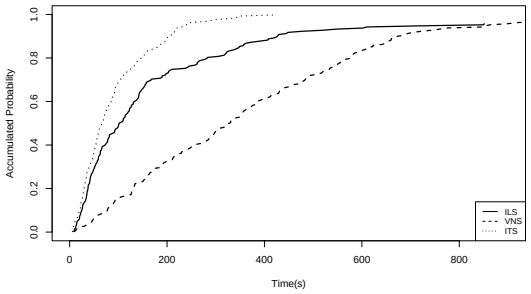

Fig. 3. TTTplot 2023/1 - target = 370

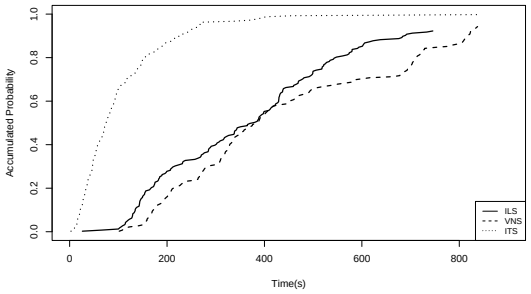

Fig. 4. TTTplot 2023/2 - target = 270

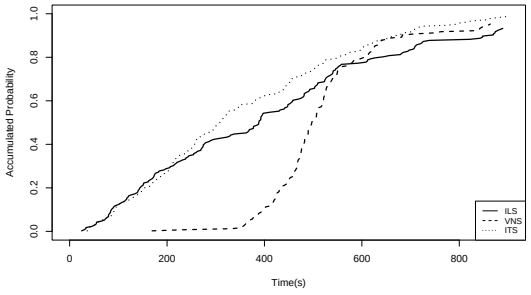

Fig. 5. TTTplot 2024/1 - target = 280

B. Discussion of Results

The experimental results presented in this work reveal consistent evidence regarding the effectiveness and limitations of the proposed metaheuristic approaches for solving the School Timetabling Problem (STP). From the data in Tables III and IV, it is evident that the Random Constructive Heuristic (RCH) serves as an efficient method for producing feasible initial solutions; however, the high associated costs highlight its inadequacy as a final solution strategy. Instead, its role is best suited as a starting point for local search-based improvement processes.

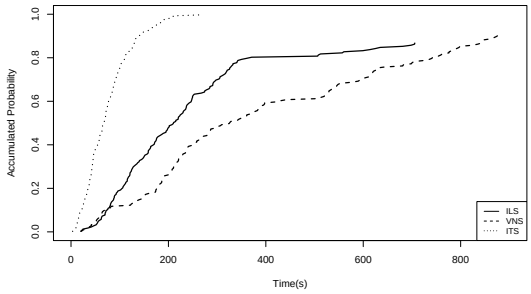

Fig. 6. TTTplot 2024/2 - target = 45

Among the metaheuristics analyzed, the Iterated Tabu Search (ITS) achieved the most robust performance. It consistently delivered the best average solution cost across all instances, while also reaching the minimum solution cost ($f(s^*)$) in every case, demonstrating its high effectiveness in both solution quality and reliability. The Iterated Local Search (ILS), while slightly less effective in more complex instances such as 2024/2, still outperformed the Variable Neighborhood Search (VNS) in average performance and showed good convergence properties under limited time budgets (300 seconds). On the other hand, although VNS did not reach the same quality of results under time constraints, the TTTplot analysis indicates a more gradual and stable convergence behavior when the execution time is extended. This suggests that VNS can be particularly beneficial in scenarios with higher computational tolerance, where broader search diversification is desired.

The use of adaptive perturbation techniques in ILS and ITS was shown to be crucial in escaping local optima, a recurring challenge in combinatorial optimization problems such as the STP. The ITS strategy, in particular, leverages memory-based mechanisms through the tabu list, which prevents cycling and promotes search diversification, enabling the algorithm to explore new regions of the solution space efficiently. The combination of adaptive perturbation and memory-based intensification allows ITS to strike a balance between diversification and intensification, resulting in more reliable performance.

The application of TTTplots proved essential for assessing the temporal robustness of each method. The ITS demonstrated superior resilience, reaching the target solution in 100% of the runs across all instances. In contrast, ILS and VNS showed a greater number of failures, particularly under strict targets such as the one defined for instance 2024/2. These findings reinforce the importance of employing time-to-target analysis in future studies, as traditional evaluation metrics may fail to capture performance stability over time.

Overall, the results validate the effectiveness of ILS, VNS,

and ITS in generating high-quality solutions while strictly satisfying hard constraints. The empirical evidence supports the claim that ITS is the most promising standalone approach for practical applications of school timetabling. Furthermore, the complementary characteristics observed among the metaheuristics suggest promising directions for hybrid algorithms that integrate the strengths of each method. Future work should also consider extending these approaches to other variants of the timetabling problem and exploring multi-objective optimization scenarios where criteria such as fairness, equity, and energy efficiency may play relevant roles.

VI. Conclusion

The experiments conducted demonstrate the feasibility and efficiency of the proposed metaheuristic strategies for solving real-world instances of the School Timetabling Problem (STP). Among the evaluated methods, the Iterated Tabu Search (ITS) stood out by achieving the best average performance and the most consistent results across all tested instances.

Although Iterated Local Search (ILS) and Variable Neighborhood Search (VNS) also delivered feasible and competitive solutions, their performance was more sensitive to instance complexity and runtime limits. VNS exhibited more gradual convergence, suggesting its suitability in scenarios with extended execution time.

While the results confirm the effectiveness of the proposed methods for the tested institutional instances, it is essential to investigate their scalability. Larger and more complex instances may expose limitations in search strategies or parameter tuning, potentially impacting solution quality and runtime. Therefore, future studies should include experiments on larger benchmark datasets, particularly those provided by the *International Timetabling Competition* (ITC), to further assess robustness and generalizability.

The empirical evidence supports the use of ITS as a robust standalone method. Additionally, the observed complementary behaviors of the algorithms point to promising directions for hybridization. Future research should also explore hybrid metaheuristics, multi-objective formulations, and the development of practical tools to facilitate adoption in educational institutions. Furthermore, studying the integration of mathematical heuristics, such as matheuristic approaches that combine exact optimization techniques with metaheuristic frameworks, represents an additional promising line of investigation to enhance solution quality and scalability.

References

[1] C. C. Gotlieb, "The construction of class-teacher time-tables." in *IFIP Congress*. North-Holland, 1962, pp. 73–77.

[2] S. S. Brito, G. H. Fonseca, T. A. Toffolo, H. G. Santos, and M. J. Souza, "A sa-vns approach for the high school timetabling problem," *Electronic Notes in Discrete Mathematics*, vol. 39, pp. 169–176, 2012.

[3] S. Even, A. Itai, and A. Shamir, "On the complexity of time table and multi-commodity flow problems," in *16th annual symposium on foundations of computer science (sfcs 1975)*. IEEE, 1975, pp. 184–193.

[4] G. H. Fonseca, T. A. Toffolo, S. S. Brito, and H. G. Santos, "Técnicas de busca local para o problema da programaçao de horários escolares," *Anais do SBPO*, pp. 1092–1103, 2012.

[5] J. H. Kingston, "Hierarchical timetable construction," in *Proceedings of the 6th International Conference on Practice and Theory of Automated Timetabling VI*. Berlin, Heidelberg: Springer-Verlag, 2006.

[6] L. V. Moreira, R. C. Monteiro, E. H. Kampke, and G. R. Mauri, "Metaheurística grasp para o problema de tabela-horário de disciplinas do departamento de computação do cca-ufes," *Anais do XLVIII Simpósio Brasileiro de Pesquisa Operacional*, pp. 2171–2182, 2016.

[7] D. H. d. S. Costa, I. M. Coelho, and P. E. D. Pinto, "Programação de horários e alocação de salas de aula no ime/uerj com simulated annealing e lahc," 2017, pp. 2077–2088.

[8] T. Song, S. Liu, X. Tang, X. Peng, and M. Chen, "An iterated local search algorithm for the university course timetabling problem," *Applied Soft Computing*, 2018.

[9] R. Lewis, B. Paechter, and B. Mccollum, "Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition," *Cardiff University, Cardiff Business School, Accounting and Finance Section, Cardiff Accounting and Finance Working Papers*, 2007.

[10] R. Qu and E. K. Burke, "Hybridizations within a graph-based hyper-heuristic framework for university timetabling problems," *Journal of the Operational Research Society*, vol. 60, no. 9, pp. 1273–1285, 2009.

[11] A. Schaerf, "A survey of automated timetabling," *Artificial intelligence review*, vol. 13, pp. 87–127, 1999.

[12] E. Burke, K. Jackson, J. H. Kingston, and R. Weare, "Automated university timetabling: The state of the art," *The computer journal*, vol. 40, no. 9, pp. 565–571, 1997.

[13] S. L. Goh, G. Kendall, and N. R. Sabar, "Improved local search approaches to solve the post enrolment course timetabling problem," *European Journal of Operational Research*, 2017.

[14] F. Peres and M. Castelli, "Combinatorial optimization problems and metaheuristics: Review, challenges, design, and development," *Applied sciences*, vol. 11, no. 14, p. 6449, 2021.

[15] Z. Lü and J.-K. Hao, "Adaptive tabu search for course timetabling," *European Journal of Operational Research*, vol. 200, pp. 235–244, 2010.

[16] H. R. Lourenço, O. C. Martin, and T. Stützle, *Iterated Local Search: Framework and Applications*. Boston, MA: Springer US, 2010, pp. 363–397.

[17] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, 1997.

[18] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Computers & Operations Research*, vol. 13, no. 5, pp. 533–549, 1986, applications of Integer Programming.

[19] A. Misevicius, A. Lenkevicius, and D. Rubliauskas, "Iterated tabu search: an improvement to standard tabu search," *Information Technology and Control*, vol. 35, no. 3, 2006.

[20] R. M. Aiex, M. G. C. Resende, and C. C. Ribeiro, "Ttt plots: a perl program to create time-to-target plots," *Optimization Letters*, 2007.