Transactions on Energy Systems and Engineering Applications

Article

UniSchedApi: A comprehensive solution for university resource scheduling and methodology comparison

Alexandra La Cruz[1,*], Luis Herrera[1], Jeisson Cortes[1], Andrés García-León[2], Erika Severeyn[3]

[1] Engineering Faculty, Universidad de Ibagué, Tolima, Colombia.
[2] Escuela de Ingeniería y Ciencias, Tecnológico de Monterey, Mexico, Mexico.
[3] Department of Thermodynamics and Transfer Phenomena, Universidad Simón Bolívar, Caracas, Venezuela
[*] Correspondence: alexandra.lacruz@unibague.edu.co

Received: 23 January 2024; Accepted: 21 October 2024; Published: 26 October 2024

Abstract: This paper introduces UniSchedApi, an API-based solution that revolutionizes optimized university resource scheduling. The primary focus of the research is the detailed evaluation of two automatic resource allocation methods: Tabu Search (TS) and Genetic Algorithm (GA). The paper thoroughly explores how these methods address challenges associated with resource allocation in university environments, considering critical factors such as teacher availability, student time constraints, classroom features (including computers, projectors, TV's, specialized laboratories, specialized equipment, etc.), among others. The evaluation is carried out meticulously, measuring the performance and memory resource usage of both algorithms, considering the comparison with the manual scheduling. The results reveal that the TS algorithm excels in terms of temporal efficiency and computational resource usage. Based on these findings, UniSchedApi implements GA and TS but uses TS as the default algorithm, ensuring more efficient and optimized management of academic resources. This research not only presents a practical solution with UniSchedApi but also provides a deep understanding of the methods for evaluating and selecting algorithms to address specific challenges in university resource allocation. These results lay the groundwork for future improvements in academic resource management.

1. Introduction

In the university context, the enrollment of courses process is one of the most crucial and complex administrative procedures, demanding significant effort from both students and educational institutions. In this regard, this project focuses on the development of a software system for scheduling the undergraduate enrollment process at the University of Ibagué (Colombia) as study case. One of the responsibilities of the Admissions and Academic Registry Office at the University of Ibagué is to schedule calendars for administrative staff, classrooms, full-time faculty, adjunct professors, and undergraduate students for the

How to cite this article: La Cruz, A.; Herrera, L.; Cortes, J.; García-León, A., and Severeyn, E.. UniSchedApi: A comprehensive solution for university resource scheduling and methodology comparison. *Transactions on Energy Systems and Engineering Applications*, 5(2): 633, 2024. DOI:10.32397/tesea.vol5.n2.633

current semester. The current process has been carried out as follows: before the enrollment period, an estimate of the number of students is obtained through a survey to determine the scope of the scheduling. This information is then analyzed and interpreted to reserve resources during the enrollment of courses stage.

Resource allocation has been performed without considering technical and engineering elements, relying on an empirical approach based on the input of involved parties. For instance, department assistants manually establish schedules for each course with their respective professors. The information from each academic unit is consolidated by the University Planning Office, which manually assigns classrooms, and if necessary, makes adjustments by shifting time slots or compensating the time of professors primarily to progress in the process. Both the student and faculty experiences are significantly affected when such cases arise, resulting in numerous complaints. Additionally, resource allocation, including infrastructure investment, maintenance, and human capital, is impacted. Moreover, the expected solution does not always meet quality standards. This problem has been observed in some universities and organizations, where the approach to tackling the issue is quite similar [1]. While linear programming can provide a solution, it becomes inefficient when the number of variables exceeds a certain limit, it is not capable of solving any kind of NP-hard problem [2], as is the case here. In this context, the field of intelligent agents plays a crucial role, utilizing heuristics to address this problem effectively. Previous contributions to similar problems have been made [1, 3–8], however non of them are available or describe the details of methods used for solving the problem. The added value for this problem lies in achieving high-quality solutions with parameters defined by the University of Ibagué as a study case.

The main objective of this research is to design and develop a system automating the planning and organization of the enrollment process to enhance efficiency and the quality of services offered by the institution. The proposed system enables faculty administrators to schedule courses more effectively, optimizing available time and resources while simplifying enrollment management tasks. A comprehensive review of scheduling and enrollment concepts, as well as technologies and methodologies for software system design, has been conducted. The analysis of the University of Ibagué's specific processes and requirements has identified the system's needs. Additionally, a front-end application for the UniSchedAPI has been included, enhancing the system's visualization and usability.

This work contributes to improving enrollment management and organization at the University of Ibagué and other educational institutions through the implementation of an efficient software system. The findings hold practical applications beyond the immediate context, particularly in optimizing administrative processes and services for students. To achieve this, two algorithms, Genetic Algorithm (GA) and Tabu Search (TS), known for their effectiveness in addressing complex optimization problems, were strategically selected and implemented.

The study presents a significant contribution by developing a software-based API for practical use in the university context, specifically at the University of Ibagué. While numerous studies propose solutions using various meta-heuristics, this work stands out as it validates the implementation of these proposals in software development. The results demonstrate the improvement of management and organization in the enrollment process not only at the University of Ibagué but also in other educational institutions. The article emphasizes the strategic selection and presentation of GA and TS as solutions to address the problem.

In addition to addressing scheduling challenges, recognizing the importance of user interaction and practical testing, we incorporated a user-friendly front-end interface. This interface allows administrators and users to interact with the scheduling system, visualize results, and perform real-world test scenarios. The inclusion of an API and a front-end interface adds usability and accessibility to our scheduling solution, marking a significant advancement in the practical implementation of scheduling systems in university contexts.

The structure of the article is as follows: In the section 2, an exhaustive review of several algorithms used for solving the scheduling planning problem in university settings and other fields is presented. Subsequently, the section 3 describes the data utilized for evaluating the methods in our case study, along with the constraints and a description of the employed algorithms. Following this, the section 4 showcases the outcomes of the implemented algorithms, evaluated within our case study, and make an analysis of findings about the results and weakness identified in our solution. Finally, the section 5 presents the ultimate conclusions drawn from this study.

2. Related work

The problem of event scheduling is present in various scenarios, such as schools (with teachers, students, and classroom schedules), transportation services (scheduling bus, plane, train trips, routes), and hospitals (scheduling patient appointments, surgeries, consultations) [9]. Through the use of artificial intelligence algorithms, these problems have been successfully addressed in several ways, leading to high-quality solutions that are close to optimal [10].

Regarding student scheduling, the problem can be classified into five groups according to Adriaen et al. [5], some of them are: the Faculty Timetabling (FTT), which involves assigning teachers to subjects; the Class-Teacher Timetabling (CTTT) assigning subjects with minimal temporal conflicts among groups of students; the Course Timetabling (CTT) assigning subjects with minimal temporal conflicts among individual students; Examination Timetabling (ETT) scheduling exams for students, ensuring that no student takes two tests simultaneously; and the Classroom assignment (CATT) which after assigning classes to teachers, the class-teacher pairs are allocated to classrooms.

Several algorithms have been employed to solve these problems, including: Graph Coloring [11], Constraint Satisfaction Programming (CSP) [12], Metaheuristic-Based Methods [4], IP/LP (Integer Programming/Linear Programming) [13], Genetic Algorithms [3, 6, 14], Memetic Algorithms [14], Tabu Search [1, 15, 16], Simulated Annealing [17], Local Search [8], Hyper-Heuristic Approach [18]. Notably, GAs emulate natural selection processes, exploring solution spaces effectively and addressing intricate relationships [19]. Memetic Algorithms [8] incorporate local search procedures, enhancing their adaptability. TS [1, 15, 16] excels in navigating solution landscapes due to its adaptive memory mechanism, adept at avoiding local optima. Simulated Annealing [17] mimics the annealing process in metallurgy, allowing the algorithm to escape sub-optimal solutions and explore a broader solution space. Local Search [8] intensifies exploration in the vicinity of solutions, contributing to the refinement of schedules.

There are on literature different approaches and strategies that have been proposed to address this problem, with each solution tailored to the specific constraints of the problem. The variety of methods used for scheduling also depends on the available resources, as some solutions have been found to consume fewer resources in terms of software quality [20]. Considering the richness of available approaches, the selection of TS and GA for our study case is rooted in their documented effectiveness and their application as a hybrid combination in previous research [21]. This combination often results in a synergistic effect, leveraging the strengths of both algorithms for enhanced solution approximation.

In our investigation, we aim to integrate TS and GA into a cohesive software solution, leveraging their complementary strengths to address the nuanced challenges of event scheduling in educational institutions. The extensive evaluation of these algorithms seeks to provide valuable insights into their performance and potential synergies, contributing to the advancement of efficient and adaptive event scheduling solutions in educational settings.

3. Methods and Materials

3.1. Data

The aim of this study is to address the challenge of the timetable schedulling problem for generating optimal schedulles in a specific university context. Within this context, the following components were considered:

- Subjects (A_n): The diverse subjects offered by the university.
- Resources (R_m): General resources such as computers, projectors, etc.
- Classrooms (C_o): Availability of classrooms for classes.
- Time slots (TS_p): Time slots available within the university.
- Constraints (RS_q): Internal criteria for filtering and evaluating solutions.
- Requirements (AR_r): List of necessary resources, both mandatory and optional, for each subject.
- Classroom Constraints (RC_s): Specific constraints related to the classrooms ('No classes during lunchtime' or 'Classes only in the mornings', among others, they can vary)
- Solution: Set of selected assignments (A_t) that solve the problem.
- Assignment: Link between a time slot (TS_i) and a classroom (C_j) for a subject (A_k).

The input process requires a subject (A_k) and a number N, indicating the number of groups to initiate the calculation. Optionally, a quality function can be added, which is a list of constraints (AR_r) that evaluate the final solution. By default, there is a list of constraints (RS_s) that evaluate the resulting solution (S). Subsequently, the required resource set (AR_r) is identified for the subject, allowing the selection of available classrooms (C_j). From these available classrooms (C_j), time slots (TS_p) are generated, within which assignments (A_k) can be made. Additionally, classroom constraints (RC_s) and filter constraints (RS_q) are extracted from the subject (A_k), aiming to create an initial set of conflict-free time slots that meet the desired requirements. This initial set is fed into an algorithm (TS or GA) to execute the process of generating an optimal solution (S).

In detail, each subject includes information such as code, credit count, study hours, name, and number of subgroups per week. Each classroom has a maximum capacity, name, and code. Each time slot has a code, day of the week, duration, start and end time. Each constraint has a code, name, alphanumeric description of the filter or evaluator, and a state (active or inactive). Each assignment includes a time slot, classroom code, and subject code. The remaining relationships are derived from the normalization of this data, through the intersection.

3.2. Tabu Search

Tabu Search method is a local-search optimization method, widely used on trying Np-hard problems, it is also known as adaptive memory programming, and considered as a robust optimization method [22]. It's used to find optimal solutions that maximize desired outcomes or minimize negative factors. TS's adaptive memory programming allows it to effectively explore solution spaces, making it a key tool for finding optimal solutions in various practical scenarios. As stopping criterion it was defined a maximum number of iterations, when it is reached, the algorithm stop and return the best current solution. The algorithm is described in Algorithm 1. This summarized workflow outlines the steps and decision points within the algorithm's operation.

Algorithm 1 Tabu Search.

Initialize: Set parameters and initialize data structures
$currentIteration \leftarrow 0$
$maxIterations \leftarrow$ maximum allowed iterations
$bestSolution \leftarrow$ initial solution
$tabuList \leftarrow$ empty list
While $currentIteration < maxIterations$ **do**
  Select room randomly
  Select a random time slot from the valid time slot group
  Validate the combination and subsequent time slots
  Add the combination and its remaining to the selection group
  If Validate solution is not in the tabu list **then**
    Evaluate the solution and assign a score
    Add the solution to the tabu list
    Compare with the current best solution
    If Comparison score is lower **then**
      Continue to the next iteration
    Else
      Replace the current best solution
    EndIf
  EndIf
EndWhile

3.3. Genetic Algorithm

The GA aims to create an initial population by randomly selecting rooms and time slots along with their validations for each requested group. Subsequently, it combines assignment features (chromosomes) through crossover. Here is the algorithm adapted to the aforementioned problem. The procedure folowed for our study case can be see in Algorithm 2.

Algorithm 2 Genetic Algorithm.

while not stopping condition **do**
  if current iteration is less than maximum allowed **then**
    return best current solution
  end if
  Evaluate the solution
  Compare with current best solution
  if if positive **then**
    replace current best solution
  end if
  Validate if crossover is possible
  if Validation is negative **then**
    Mutate chromosome by changing to new time slot;
    if not possible, mutate to new room until achieve it
  else
    Perform Uniform Midpoint Crossover,
    swapping rooms and their time slots
  end if
end while

The stopping condition for the GA is also the maximum number of iterations. The algorithm score is evaluated comparing the last solution with the current best solution; if positive, replace the current best solution.

4. Results and Discussion

As a result, these two algorithms were implemented, evaluated, and actually they were incorporated to a software based on the architecture described in the Figure 1.

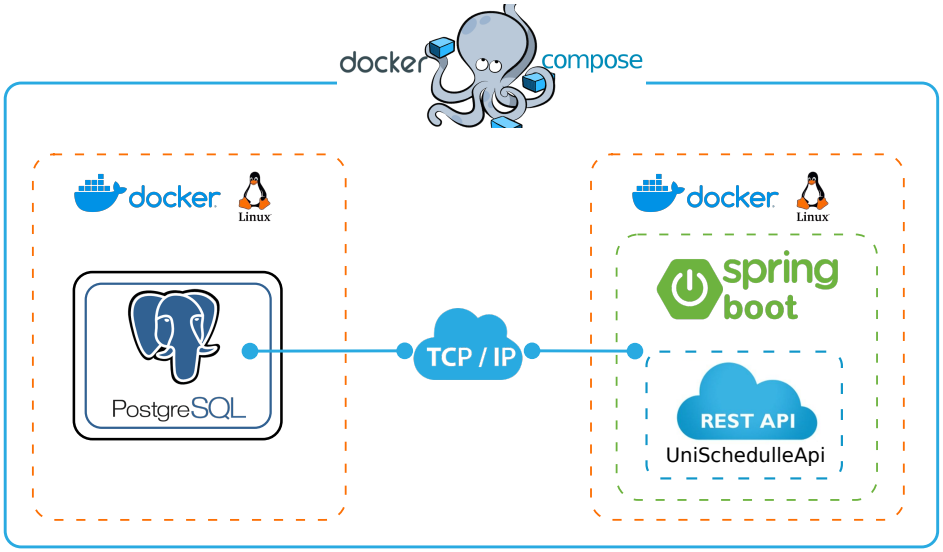

Figure 1. Architecture of UniSchedApi Software, where TS and GA are available for Universidad de Ibague (Colombia).

4.1. Algorithm Implementation and Performance Evaluation

The algorithms were implemented and evaluated in a realistic setting using representative data from a typical university scheduling scenario. They were applied for the last two semester (2023-A and 2023-B, first and second semester of the year 2023, respectively) of the Universidad de Ibagué. For comparing purpose, the process were done manually as the administrative staff were used to do it, and using UniSchedApi. The results were promising, showing significant improvement over existing solutions. For the evaluation of the better algorithm several considerations were taking into account in both of them:

- **Runtime:** Both algorithms displayed acceptable execution times, even for large datasets, making it practical for real-time implementation.
- **Space Efficiency:** Both algorithms optimized physical space allocation, reducing duplication and schedule overlaps.
- **Constraint Satisfaction:** Specific problem constraints, such as resource availability and instructor preferences, were successfully met.
- **Adaptability:** Both algorithms exhibited flexibility in accommodating last-minute changes and adjustments to previously set conditions.

4.2. Performance Test and Resource Consumption Analysis

The performance evaluation considers a machine with a 2.5 GHz CPU, 6GB RAM, and 20GB disk, meeting recommended execution requirements. The concurrent use of a PostgresSQL database is accounted for, impacting the application only during initialization. The test involved stressing the REST service and monitoring its memory consumption over time. This validates if constructing "MemoryObjects" effectively reduces resource-intensive processes such as instances, calls, and validations. A 10-minute stress test was conducted using both the TS and GA on a request for a subject (Algorithm and Object Programming). The subject involves 7 groups with 3 weekly classes each, limited to 100,000 iterations. The Figure 1 shows as The GA after second 200 has more memory use than the TS algorithm, showing that both algorithm are quite similar before second 200, after, the best is the TS method, based on the memory use in megabyte (see Figure 2).

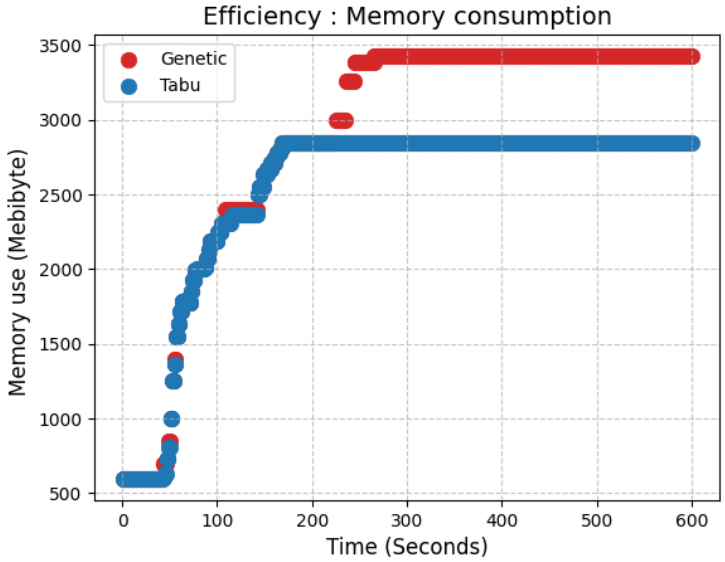

Figure 2. Comparative of the memory consumption between algorithms.

4.3. Varying the Performance Conditions: High vs. Low Request Difficulty

Due to changing performance conditions based on request complexity, two stress tests were performed for both algorithms, simulating high and low difficulty scenarios. The high complexity scenario was defined using seven (7) groups or courses, one time zone constraint allowed (assignable hours) and one not allowed (non-assignable hours), five (5) mandatory resources that the classrooms must meet, and (7) optional resources that the classrooms may require and a group clash constraint. The low complexity scenario was instead defined using a single group, a single unallowable time zone constraint, and an optional classroom resource.

Utilizing the same machine parameters as the efficiency test and maintaining the same maximum iteration count, diverse statistical metrics were collected to assess response times. This analysis was pivotal as it can uncover the optimal algorithm for specific cases, influencing the selection of new algorithms for system integration. Figures 3 and 4 show the statistical summary respect to the time in milliseconds with a high and low problem difficulty.

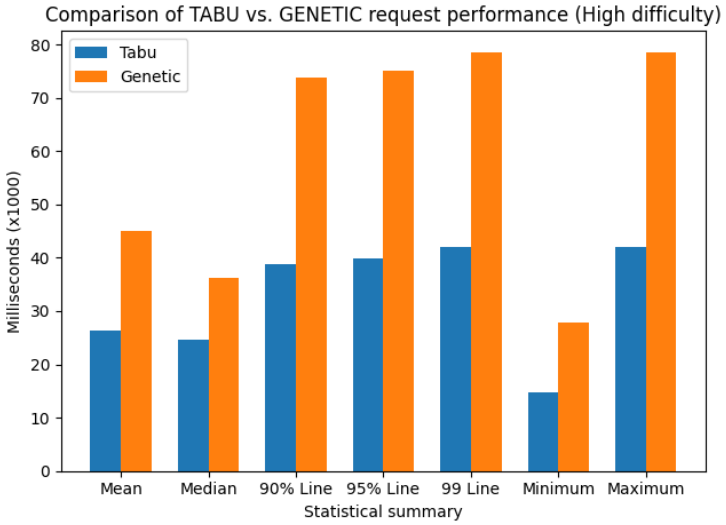

Figure 3. Comparative of the request difficulty (hight difficulty) between algorithms.

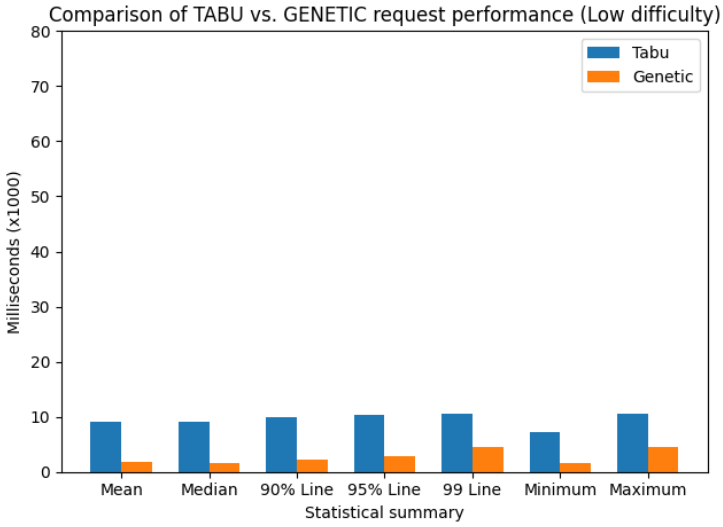

Figure 4. Comparative of the request difficulty (low difficulty) between algorithms.

4.4. Analysis of Iteration Test

From the analysis of the iteration test results, it can be inferred that fewer than 86 percent of requests necessitate fewer than 60,000 iterations to achieve an optimal solution (see Fig. 5). This analysis suggests the possibility of reducing the iteration count to this value to achieve quicker results if needed. However, maintaining a higher iteration count is still recommended to explore a wider array of feasible solutions and increase the likelihood of identifying an optimal maximum, which is shown to be unlikely in prior tests. This probability improves with more iterations and well-calibrated resource allocation aligned with subject requirements. To prevent infinite loops, algorithms were confined to a maximum iteration count. Stress tests employ a considerably high iteration count (100,000), strategically chosen to analyze various behaviors previously examined. Metadata from solution outputs of both algorithms in high-difficulty stress tests were extracted for the Figure 5.

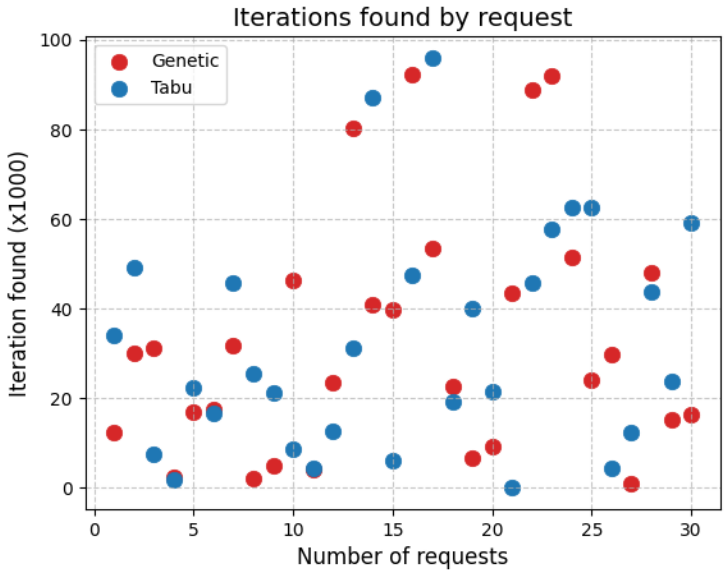

Figure 5. Comparative of iterative values against the number of requests by algorithms (TS and GA).

4.5. Identified weakness concerning the Algorithmic Efficiency

While the process has proven to be functional and effective in time allocation, weaknesses were identified when dealing with exceptional cases, such as the global restriction to respect lunchtime for students and faculty, per example. In this context, the algorithm scheduled classes at noon for three out of five groups, contrary to the university's expectation of equitable assignments. Despite these shortcomings, the algorithms retain the initial observations based on stress tests.

The comparison of algorithms (TS *vs* GA) with the traditional process under the two scenarios tested (2023-A ans 2023-B) showed that the algorithms maintain their timings and adhere to all desired restrictions to improve the well-being of students and faculty. However, it is crucial to note that the complexity of allocation increases with the inclusion of more restrictions, indicating a potential weakness in the algorithm in terms of processing times and the identification of bottlenecks. Although negative aspects observed, these can be improved through meticulous refinement of quality criteria and restrictions. Precision in assignment hours is an area for improvement, as assignments, while feasible and optimized in terms of resources, may not be comfortable due to external factors such as sunlight or temperature. This analysis can be complemented by a study on the perception of the university community on the subject.

To address these issues, future work will focus on the following improvements:

- **Enhancing Accuracy in Hour Allocation:** Refining our algorithms to better account for the equitable distribution of classes, particularly around crucial times like lunchtime, to meet the university's expectations more accurately.
- **Incorporating Comfort Factors:** Considering external factors such as sunlight and temperature in the scheduling process to ensure that assignments are not only feasible and optimized in terms of resources but also comfortable for students and faculty.
- **Continuous Improvement:** Conducting meticulous refinements of quality criteria and restrictions, ensuring that we can handle the increasing complexity of allocations without significant processing delays or bottlenecks.

- **Community Feedback:**Complementing algorithmic improvements with studies on the perception of the university community to ensure that our solutions are well-received and effective in practice.

These enhancements aim to address the identified weaknesses and further improve the effectiveness of the proposed scheduling solution, ensuring better alignment with the university's expectations and the overall well-being of students and faculty.

4.6. User Interface

Beyond optimizing times and processes, the goal is to enhance the comfort of the community, significantly contributing to their academic or professional performance. This approach aims not only for efficiency but also for the satisfaction and well-being of those within the university community. Besides a front-end application was implemented within the framework of our scientific project with the purpose of enhancing the visualization of data flows in conjunction with the defined meta-heuristic and UniSchedApi (see figure 6). This added component provides an interface that facilitates efficient exploration and understanding of the processes and results derived from the meta-heuristic for researchers. The incorporation of this application enriches the usability of the platform, specially designed to meet the needs of our scientific project, enabling a deeper and faster understanding of the data flows generated by the meta-heuristics and UniSchedApi.

The ability to access detailed visualizations empowers users to make informed decisions and effectively optimize the parameters of the meta-heuristics. This advancement underscores our ongoing commitment to improving the usability and effectiveness of our solutions within the specific context of our scientific project, providing researchers with the necessary tools to maximize the value of the meta-heuristics and UniSchedApi in the research. This development is presented as a contribution to the scientific field and

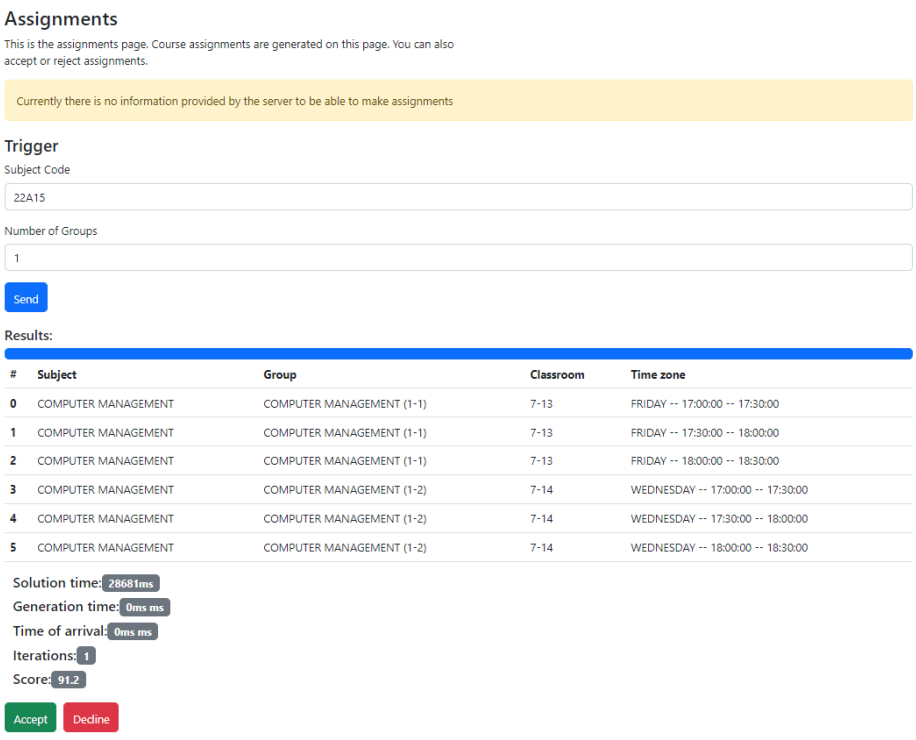

Figure 6. Image representing the application for interacting with UniSchedApi.

adds to research. We are confident that the inclusion of this functionality will enrich our methodological approach and provide more robust results for the advancement of our project.

5. Conclusions

We are pleased to introduce UniSchedApi, a robust and innovative API designed for resource planning and allocation in university environments. UniSchedApi offers a REST API along with a user-friendly interface, allowing for the execution and retrieval of resource allocation results. This interface provides users with the ability to accept, reject, or reimplement the results, enhancing the adaptability and user-driven nature of the resource allocation process.

With its comprehensive capabilities, UniSchedApi stands as a significant contribution to the field of university resource scheduling. The integration of this API into our methodology facilitates a practical and user-centric approach, aligning with the transformative potential of algorithm-based methods to enhance the efficiency and organization of education. This study not only advances the academic discourse on optimization but also underscores the tangible impact that innovative technologies like UniSchedApi can have on reshaping resource management in educational settings. The default configuration was implemented by using TS due that this methods showed better performance in high and low complexity scenario, memory uses and allocation results compared to the manual allocation.

Throughout this comprehensive research, we undertook the meticulous implementation and evaluation of two algorithms within a realistic educational environment. The main objective was to address the challenges related to the optimization of space and time allocation in educational environments and to present a simple interface to visualize the results of the automatic allocation. The results of this effort have been really encouraging, showing a substantial advance compared to existing solutions in this field.

The consistent superiority of the Tabu Search (TS) algorithm over the Genetic Algorithm (GA) methods in all tests performed is noteworthy. Furthermore, the TS algorithm demonstrated its ability to provide high quality solutions in much shorter timescales than conventional methods, making it not only highly competitive, but also extraordinarily efficient. Interestingly, the results in terms of efficiency, memory usage, and optimal solution remained consistent across both the Semester 2023A and Semester 2023B applications, representing two tested cases from different semesters and highlighting the robustness of the algorithms.

However, it is worth mentioning that some weaknesses were identified in the implemented algorithms. One notable issue was the deviation from equitable assignments. The algorithm struggled with exceptional cases, such as adhering to a global restriction to respect lunchtime, which led to inequitable class assignments for certain groups. Future work will focus on refining the algorithms to better account for equitable distribution of classes around crucial times, ensuring fair and balanced scheduling for all groups. Another weakness was the increased complexity with additional restrictions. The comparison of algorithms (TS vs GA) highlighted a potential weakness in the algorithm's processing times and identification of bottlenecks when faced with a higher number of restrictions. To address this, meticulous refinements of the quality criteria and restrictions are required to enhance the algorithm's efficiency in handling complex scenarios without significant processing delays.

Additionally, there was a lack of precision in assignment hours. While optimized for resources, the algorithm may not account for comfort factors such as sunlight or temperature, indicating a need for refinement to improve precision and consider additional types of facilities. Incorporating external factors into the scheduling process will ensure that assignments are both feasible and comfortable for students and faculty. Furthermore, external factors impacting comfort were not adequately considered, making some assignments uncomfortable despite being feasible. This underscores the importance of refining the

algorithm to consider comfort aspects during scheduling. Conducting studies on the perception of the university community will ensure that the solutions are well-received and effective in practice.

In essence, this study represents a fundamental achievement in the effective management of schedules and spaces in the educational setting. It is imperative to underscore the pivotal role played by algorithmic approaches in shaping the landscape of educational scheduling. The remarkable achievements demonstrated by the TS algorithm, particularly its efficiency and competitive advantage, argue for the transformative potential of algorithm-based methods to foster more effective and well-organized education. This study not only contributes to the academic discourse on optimization, but also accentuates the tangible impact that algorithms can have on education.

Acknowledgments

The Research Direction of Tecnológico de Monterey, located in Monterrey, Mexico, played a pivotal role as the primary financial supporter of this project. Additionally, appreciation is extended to Universidad Simón Bolívar, in Caracas, Venezuela, and Universidad de Ibagué, in Tolima, Colombia, for their invaluable support and resources throughout the project's development.

Funding: This research received no external funding

Author contributions: In accordance with the guidelines for authorship, each contributor to this research paper made substantial contributions to different aspects of the study. The conception and design of the research were collectively developed by all authors. Conceptualization, Metholdology and Investigation, Cortes, J., Herrera, L., García-León, A., and La Cruz A.; Software development and Validation, Cortes, J. and Herrera, L.; Investigation, C.A.; Writing – Review & Editing, all authors and collaborators.

Disclosure statement: The authors declare no conflict of interest.

References

[1] A.R Mushi. Tabu search heuristic for university course timetabling problem. *African Journal of Science and Technology*, 7(1), 2006.

[2] H. Raoofpanah and V. Ghezavati. Extended hybrid tabu search and simulated annealing algorithm for location-inventory model with multiple products, multiple distribution centers and multiple capacity levels. *Production Engineering Research and Development*, 13:649–663, 2019.

[3] X. Deng, Y. Zhang, B. Kang, J. Wu, X. Sun, and Y. Deng. An application of genetic algorithm for university course timetabling problem. In *Proceedings of the 23rd Chinese Control and Decision Conference (CCDC 2011)*, pages 2119–2122, 2011.

[4] Rhydian Lewis. A survey of metaheuristic-based techniques for university timetabling problems. *OR Spectrum*, 30:167–190, 01 2008.

[5] Marieke Adriaen, Patrick De Causmaecker, and Piet Demeester. Tackling the university course timetabling problem with an aggregation approach. In *Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2006)*, pages 330–335, 2006.

[6] Ahmed A. Mahiba and Chitharanjan A. D. Durai. Genetic algorithm with search bank strategies for university course timetabling problem. *Procedia Engineering*, 38:253–263, 2012.

[7] Michael R. R. Lewis. *Metaheuristics for University Course Timetabling*. PhD thesis, Napier University, 2006.

[8] M. Joudaki, M. Imani, and N. Mazhari. Using improved memetic algorithm and local search to solve university course timetabling problem (ucttp). Doroud, Iran, 2010. Islamic Azad University.

[9] Robert Pellerin, Nathalie Perrier, and François Berthaut. A survey of hybrid metaheuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 280(2):395–416, 2020.

[10] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019.

[11] P. Nandal, Ankit Satyawali, Dhananjay Sachdeva, and Abhinav Singh Tomar. Graph coloring based scheduling algorithm to automatically generate college course timetable. In *2021 11th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pages 210–214, 2021.

[12] Sally C. Brailsford, Chris N. Potts, and Barbara M. Smith. Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research*, 119(3):557–581, 1999.

[13] Tadeusz Sawik. *Scheduling in Supply Chains Using Mixed Integer Programming*. Wiley, 2011.

[14] L. Buriol, P.M. França, and P. Moscato. A new memetic algorithm for the asymmetric traveling salesman problem. *Journal of Heuristics*, 10:483–506, 2004.

[15] Marek Mika, Grzegorz Waligóra, and Jan Węglarz. Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. *European Journal of Operational Research*, 187(3):1238–1250, 2008.

[16] Cuneyt Aladag and Gulay Hocaoglu. A tabu search algorithm to solve a course timetabling problem. *Hacettepe Journal of Mathematics and Statistics*, pages 53–64, 2007.

[17] Juan Frausto-Solís, Francisco Alonso-Pecina, and Jaime Mora-Vargas. An efficient simulated annealing algorithm for feasible solutions of course timetabling. In *Proceedings of the 10th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2008)*, pages 675–685, 2008.

[18] Juan Soria-Alcaraz, Gabriela Ochoa, Jerry Swan, Miguel Carpio, Héctor Puga, and Edmund Burke. Effective learning hyper-heuristics for the course timetabling problem. *European Journal of Operational Research*, pages 77–86, 2014.

[19] S. Castillo-Rivera, J. De Antón, R. del Olmo, J. Pajares, and A. López-Paredes. Genetic algorithms for the scheduling in additive manufacturing. *International Journal of Production Management and Engineering*, 8(2):59–63, 2020.

[20] *Scheduling under Resource Constraints*, pages 425–475. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[21] S.N. Jat and S. Yang. A hybrid genetic algorithm and tabu search approach for post enrolment course timetabling. *Journal of Scheduling*, 14:617–637, 2011.

[22] Fred Glover and Manuel Laguna. *Tabu Search*, pages 3261–3362. Springer New York, New York, NY, 2013.