Optimization and Scheduling of Examination Dates

Lilly Abraham
Responsible for RL PPO model and GUI development and contributed to sections 4, 6 and 2

Deva Dharshni Rajarajeswari
Responsible for RL-SA-HH model and Genetic Algorithm and contributed to sections 4 and 5

Megha Eldho
Contributed to RL-SA-HH framework, GUI and responsible for documentation and wrote sections 1,2,3,4,6

Akshay Augustine Sheby
Contributed to RL-SA-HH framework and contributed to sections 4,5,6

ABSTRACT

This paper proposes mainly 3 approaches- Reinforcement learning simulated annealing hyper-heuristic framework, Reinforcement Learning PPO framework, and Genetic Algorithm. Each of these methods generate an automated examination schedule for the University of Applied Sciences, Würzburg – Schweinfurt. Creating an ideal examination schedule poses a formidable challenge due to multiple factors, such as scheduling conflicts, restricted time frame and room availability. Our objective is to explore different sources and methods in artificial intelligence technologies to automate the creation of an examination timetable and implement a system to provide a suitable examination plan for the University. The general and specific requirements for the examination scheduling for the University were identified. Multiple frameworks were constructed using the reinforcement learning approach, and also with genetic algorithm. An examination schedule was formulated and assessed for its effectiveness. The system strives to fulfill all the hard and soft constraints, while reducing the manual effort involved in scheduling examinations.

KEYWORDS

Examination Scheduling, Reinforcement learning simulated annealing framework, Reinforcement Learning PPO framework, Genetic Algorithm.

1 INTRODUCTION

Universities face the challenge of examination timetabling as it is a complex and time-consuming task, which requires a great deal of human effort. The University Würzburg-Schweinfurt's faculty of Computer Science and Business Information Systems currently handles examination scheduling manually for each semester. To address this issue, our study aims to develop an automated examination timetable using artificial intelligence technologies. The process of examination scheduling requires the allocation of specific dates and times for exams, aiming to optimize resource utilization while accommodating various constraints and preferences. Hard constraints are non-negotiable conditions such as adhering to room capacity, number of days and ensuring that exams with common students are not scheduled at the same time. On the other hand, soft constraints include individual preferences such that students have sufficient time in between exams. By effectively balancing these constraints, examination scheduling algorithms can create examination plans that meet necessary requirements while optimizing the student and faculty preferences.

Through our literature review from multiple research papers we identified the most suitable methods to address the scheduling problem. This paper focuses on three distinct approaches that were employed to tackle the examination timetabling issue: Reinforcement learning simulated annealing hyper-heuristic framework[6], Reinforcement Learning PPO framework, and using genetic algorithm. The primary objectives of our project are:

- Study and analyze various methods and algorithms that can be used for scheduling examinations, taking into account factors like room availability and student preferences.
- Evaluate the performance of scheduling algorithms in terms of meeting both hard constraints and soft constraints.
- Explore the potential of incorporating reinforcement learning to improve the accuracy of examination scheduling solutions.

2 BACKGROUND

Literature review over time shows that, there have been attempts to optimize examination scheduling using heuristics and constraint programming. Techniques such as backtracking, branch and bound, and graph coloring algorithms were applied to generate feasible schedules, despite its limited scalability and optimality.

Metaheuristics gained popularity later due to the complex nature of timetabling, as they offer more generalized solutions compared to traditional methods[2]. Genetic algorithms (GAs), simulated annealing, tabu search and ant colony optimization emerged as popular metaheuristic approaches[9], offering scalable and adaptable solutions to the examination scheduling problem. These algorithms provided schedulers with tools to generate high-quality schedules while considering both hard and soft constraints. Simulated annealing (SA) is a heuristic method for global optimization that is inspired by the cooling process of physical systems. It is based on statistical mechanics concepts and does not rely on any specific assumptions about the objective function[5]. Tabu search also is a heuristic search technique that offers the benefit of an internal memory. This allows for easier avoidance of local optima and enables the exploration of global or near-global optimal solutions within a shorter timeframe [4] [8]. Genetic Algorithm is a machine learning technique that utilizes natural selection principles to perform iterative search, optimization, and adaptation[11] and the potential of this system to address examination scheduling problems at the university level is quite promising[10].

In recent years, there has been a growing interest in hybridizing different scheduling approaches[6], combining the strengths of multiple algorithms to improve solution quality and robustness. Hybrid approaches often integrate metaheuristic algorithms with machine learning techniques such as reinforcement learning, deep learning enabling schedulers to adaptively optimize schedules. In their study, Chiarandini et al. [3]introduced a hybrid metaheuristics algorithm that combines various techniques such as heuristics, tabu search, variable neighborhood descent, and simulated annealing to address the university course timetabling problem efficiently. It has been demonstrated that combining various selection Hyper-heuristic components leads to favorable outcomes. In [6], a Hyper-heuristic approach was introduced, which integrates Reinforcement Learning techniques with a Simulated Annealing algorithm during the high-level heuristic selection phase. This approach is a non-deterministic Hyper-heuristic that focuses on single-point heuristic selection. The framework encompasses multiple stages, such as generating an initial solution, selecting heuristics, move acceptance, and designing low-level heuristics. The incorporation of Reinforcement Learning algorithms occurs specifically in the heuristic selection stage. This research served as the primary basis for our work on the RL-SA-HH framework.

3 FORMULATION OF THE PROBLEM

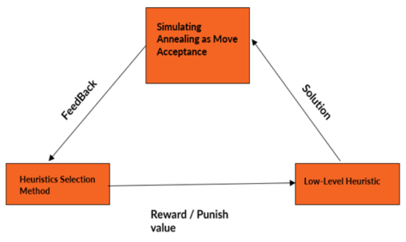
The University of Applied Sciences Würzburg-Schweinfurt's Faculty of Computer Science and Business Information Systems currently has a large student population. Each of the approximately 1,000 students in the Faculty of Informatics and Business Informatics are required to take six exams per semester, resulting in a total of about 4,500 exams written each semester. Additionally, there are numerous oral examination dates that are individually arranged with the professors. Three input files were provided, containing information about students registering for exams, examination halls, and course details. The Exam Registration file includes the matriculation ID of students and the Course ID of the exams they registered for. The Course file includes the Course ID and Course Name. The Room Capacity file primarily includes the Room ID and the corresponding capacity of the rooms.

However, there are limited resources available, including only 13 exam rooms and a time frame of just 3 weeks to conduct all the exams due to legal regulations. And there are 4 exam time slots a day. This scheduling problem was attempted to be solved based on the constraint solver OptaPlanner[7]. However, the results did not meet expectations as it violated some soft constraints. Our goal is to employ artificial intelligence technologies to solve this problem while ensuring compliance with all hard and soft constraints.

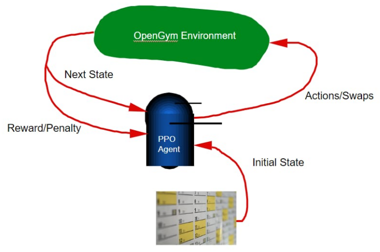
The schedule for examinations must be created in accordance with specific constraints. There are 2 types of constraints in this context, hard constraints and soft constraints. In order for a schedule to be deemed valid, it is imperative that all hard constraints are met. The hard constraints are considered crucial and must be fulfilled. On the contrary, soft constraints are merely desirable features of the timetable.

Hard Constraints:

(1) An exam will be scheduled for each course.
(2) A student can not give more than one exam at a time.
(3) All exams must be held between 9 AM and 5 PM.
(4) The capacity of the exam should not exceed the room capacity.
(5) No exam scheduled in the same room at the same time.


Simulating Annealing as Move Acceptance
FeedBack
Solution
Heuristics Selection Method
Low-Level Heuristic
Reward / Punish value

(a) RL-SA-Hyperheuristic Framework



(b) RL-PPO Framework

Figure 1: Frameworks

(6) No exams are scheduled on Saturdays and Sundays.

Soft Constraints:

(1) Two exams in a row: minimize student sitting exams on consecutive days.
(2) One hour of break for faculty meeting.
(3) Two exams in a day: minimize student sitting more than two exams in a day.
(4) Period penalty: minimize the number of exams scheduled in the period with a penalty.
(5) Room penalty: minimize the number of exams scheduled in a room with a penalty.
(6) Larger examination schedule late in the timetable: minimize the number of large exams that appear 'late' in the timetable.

4 METHODOLOGY

Three methods have been used to solve the examination scheduling problem. The first one is a Reinforcement learning simulated annealing hyper-heuristic framework, the second one is based on Reinforcement Learning PPO framework and the last one is a Genetic Algorithm.

4.1 Reinforcement learning simulated annealing hyper-heuristic framework

Traditional Reinforcement Learning (RL) approaches has several limitations in examination scheduling due to the larger state and action spaces that exist in such problems. To tackle this problem, researchers have proposed combining Reinforcement learning with other techniques to improve the overall scheduling process. One such approach is integrating Reinforcement learning concepts with Hyper-heuristic algorithms, utilizing their capability to efficiently explore solution spaces. This combination transforms the Hyper-heuristic searching process into a Reinforcement learning framework, with heuristic selection methods as learning policies and low-level heuristics[1] as actions. The reward function in this context evaluates the performance of heuristics.

$$P(dE) = \exp\left(\frac{dE}{kT}\right) \tag{1}$$

The proposed framework is shown in Figure 1a, it combines heuristic selection method with a Simulated Annealing-based move acceptance strategy. The selection methods within this framework aim to enhance the initial solution through exploration and exploitation. Simulated Annealing[6] serves as the environment, facilitating the selection of optimal solutions, which is defined by the equation 1, where $P(dE)$ represents the probability of a change in energy dE, k is the Boltzmann constant, and T is the temperature. In this framework, each action (i.e., low level heuristic) is assigned a reward value or penalty value based on its performance. To make a balance between exploration and exploitation, an ϵ - greedy algorithm is employed, where the probability parameter ϵ controls the likelihood of exploration versus exploitation. As the learning progresses, ϵ diminishes.

This approach was not able to yield the expected results as it was only trying to mimic the reinforcement learning framework and was not able to handle the complexity of our constraints.

4.2 Reinforcement Learning PPO framework

Our implementation of the exam scheduling problem using Reinforcement Learning Proximal policy optimization (PPO) model with the 'MLP' policy from stable-baselines library and the OpenGym environment, which is shown in Figure 1b.

$$L(\theta) = \mathbb{E}\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right] \tag{2}$$

PPO is a state of the art on-policy RL algorithm developed by OpenAI which uses a policy gradient method. It introduces a clipped surrogate objective function[12] to prevent large policy updates, balancing exploration and exploitation in reinforcement learning, which is defined by the equation 2, where $L(\theta)$ maximizes the objective, $r_t(\theta)$ is the policy ratio, $\hat{A}_t$ is the advantage estimate, and ϵ controls clipping.

The initial schedule was constructed such that all hard constraints are satisfied and almost all soft constraints. This was fed into the RL agent as the initial state which was a 3d tensor of dimensions days, timeslots and rooms. Each entry in the tensor corresponded to a course id that was scheduled. The environment was considered discrete for both the observation and action space. We had defined 7 actions as swapping across multiple dimensions and combination of these dimensions within the initial state. Both hard and soft constraint checks were put into place and weights were assigned per category violation as penalty values. The reward mechanism was constructed in a way to minimize the overall penalty. Our model struggled to learn even when we gave a worse-off state. The constraints were too tricky for the model to converge as RL agent cannot learn a strategy but merely the most suitable choice of actions/swaps to be performed to generate the optimal schedule. Hence, the need to try out optimization algorithms for the problem.

4.3 Genetic Algorithm

Our approach using Genetic algorithm to solve the problem in hand aims to take advantage of the natural selection methodology it offers from the concept of biology. Over a number of generations, the algorithm is able to converge to the most optimal schedule based on it's fitness score. The latter is calculated on the basis of the count in constraints being violated.

```
Data: Days: List of tuples containing the date, day of the
        week, and day index;
  Exam Start Timing: List of tuples containing the start time
  of exams and its index;
  Exam Duration: Duration of each exam;
  Courses: List of courses;
  Class Rooms: List of available class rooms;
  Registrations: List of student registrations for each course;
  Total Class Rooms: Total number of available class rooms;
  Individual: Named tuple representing an individual in the
  population;
  Population Size: Size of the population;
  Crossover Probability, Mutation Probability:
  Probabilities for crossover and mutation operations;
Result: Exam schedule using genetic algorithm and
        constraints handling
1  Initialization;
   /* Initialize data structures and parameters   */
2  ;
3  Generate Initial Population;
   /* Randomly generate initial exam schedules    */
4  ;
5  while not at end of termination condition do
6      Select Parents for Crossover;
       /* Use selection methods such as roulette
          wheel selection                          */
7      ;
8      Apply Crossover and Mutation;
       /* Generate new individuals from selected
          parents                                  */
9      ;
10     Calculate Fitness Values;
       /* Evaluate fitness of each individual      */
11     ;
12     Select Fittest Individuals;
       /* Choose individuals for next generation   */
13     ;
14 end
15 Display Exam Schedule;
   /* Visualize exam schedule                      */
16 ;
```

Algorithm 1: Exam Scheduling with Genetic Algorithm and Constraints Handling

Even without having to penalise the constraints separately, the model is able to converge without getting stuck in local minima. The proposed GA is shown in the Algorithm 1.

Objective: The objective of the implemented examination scheduling method is to fine-tune the task sequences within each exam session, focusing on reducing conflicts and optimizing the overall schedule's efficiency.

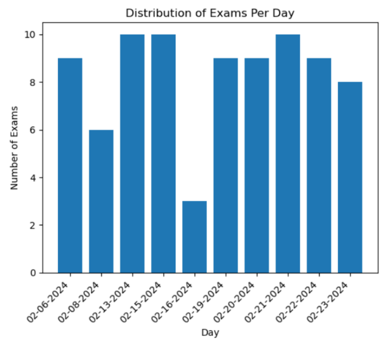
Implementation Details:

- **Chromosomes:** In this context, chromosomes represent potential examination schedules, where each chromosome lists exams assigned to specific days and timeslots.
- **Elitism:** The best individuals from the previous generation should be preserved in the next generation without undergoing crossover or mutation. Elitism can help maintain the best solutions across generations
- **Crossover (Exam Rescheduling):** This process involves randomly selecting two exams and exchanging their timings, aiming to generate better schedules.
- **Parent Chromosomes:** In genetic algorithms, parent chromosomes represent existing exam schedules, each composed of exams.
- **Offspring Chromosomes After Crossover:** Offspring chromosomes are newly generated schedules formed by mixing parts of two parent schedules, with aim to generate schedules with minimum constraint violation.
- **Selection:** Based on a fitness score that is build on our constraint violation rules selection involves choosing schedules that are more optimal.
- **Fitness Function:** The fitness of each chromosome is calculated, as defined by the equation 3, where C represents a chromosome, $Makespan(C)$ represents the total duration of the schedule, $Conflicts(C)$ represents the number of conflicts in the schedule, and λ is a weighting factor that balances the importance of minimizing conflicts and optimizing efficiency.

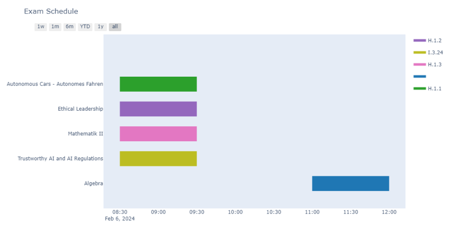
$$Fitness(C) = \frac{1}{Makespan(C) + \lambda \times Conflicts(C)} \tag{3}$$

5 RESULTS AND DISCUSSION

- Among the above used techniques, the genetic algorithm provides the most optimal solution, satisfying all hard constraints and soft constraints, except 'Two exams in a row' constraint.
- Addition of Elitism and proper arrangement of large exam helped us to minimize number of students attending consecutive exams on consecutive days.
- The bar graph and gantt chart shown in Figure 2a and Figure 2b highlight the algorithm's usefulness, will give readers a clear understanding of the scheduling procedure and results.
- We are able to generate the timetable as a pdf file containing the Date, Timeslot, Course and the RoomId.



(a) Bar graph distribution of Exam Per Day



(b) Gantt Chart of the exam schedule for first five exams

Figure 2: Bar Graph and Gantt Chart

6 CONCLUSION

In this research, we introduce three artificial intelligence techniques for automating the examination timetabling process at our University. Among these techniques, the genetic algorithm has proven to be the most fit, because the constraints taken into account are restrictive and to be scheduled within very tight time frame. Genetic Algorithm is an abstract concept of reinforcement learning wherein we do not have an agent but the other concepts of action, state, environment space and reward utility functions are considered the same. The reason why Genetic Algorithm super seeded over the RL-PPO methodology is because the environment in the case of RL-PPO approach is complicated to customize(as the constraints are very much dependent on the initial state of the model) and the predefined environment in PPO do not give expected results. The reason why the RL-SA-HH approach could not have yielded expected results is that the initial solution was not optimized before feeding into the framework. Further research into using algorithms like Squeaky Wheels Optimization [6] in initial solution could prove to be more helpful. We have also attempted to use the Google OR tools-CP-SAT model to train the model to optimize the schedule but our constraints are too complicated for this method as these model work very well for multiple but for much simpler constraints. This study indicates that there is still room for further improving the exam schedules in case where the constraints are too restrictive.

ACKNOWLEDGMENTS

We thank Prof. Dr. Frank Michael Schleif for supervising our project and Prof. Dr. Peter Braun for guiding us with the project requirement. We are also grateful to Mr.Thomas Rust for giving us insight into how the exam schedule is currently being built on.

For our project, Trello has provided a wonderful platform for effectively managing our tasks and meetings. With the help of tools such as Canva, Quillbot, Overleaf, we were able to transition across multiple stages of our project presentation and documentation.

REFERENCES

[1] Aftab Ahmed, Abdul Raziq, Jan Muhammad, Manzoor Brohi, and Muhammad Khan. 2015. Formulation of low level heuristics. *International Journal of Basic and Applied Sciences* 4 (01 2015), 44–50. https://doi.org/10.14419/ijbas.v4i1.4001

[2] Saman Almufti, Awaz Shaban, Rasan Ali, and Jayson Fuente. 2023. Overview of Metaheuristic Algorithms. *Polaris Global Journal of Scholarly Research and Trends* 2 (04 2023), 10–32. https://doi.org/10.58429/pgjsrt.v2n2a144

[3] Marco Chiarandini, M. Birattari, Krzysztof Socha, and Olivia Rossi-Doria. 2006. An effective hybrid algorithm for university course timetabling. *Journal of Scheduling* 9 (2006), 403–432. https://api.semanticscholar.org/CorpusID:16997081

[4] Shu-Chuan Chu and H.L. Fang. 2000. Genetic algorithms vs. Tabu search in timetable scheduling. 492 – 495. https://doi.org/10.1109/KES.1999.820230

[5] Crescenzio Gallo and Vito Capozzi. 2019. A Simulated Annealing Algorithm for Scheduling Problems. *Journal of Applied Mathematics and Physics* 7 (10 2019). https://doi.org/10.4236/jamp.2019.simann

[6] Kehan Han. 2018. Using reinforcement learning in solving exam timetabling problems. Queen's University Belfast. Faculty of Engineering and Physical Sciences, December. https://api.semanticscholar.org/CorpusID:208089745

[7] Justin Seegets Janik Hemrich, Stella Konieczek. [n. d.]. Automation of the Examination Timetabling. ([n. d.]).

[8] Hamid Lawal, Ibrahim Adeyanju, Elijah Omidiora, Oladiran Arulogun, and Oluyinka Omotosho. 2014. University Examination Timetabling Using Tabu Search. *International Journal of Scientific and Engineering Research* 5 (10 2014), 785–788.

[9] Rhydian Lewis. 2008. A survey of metaheuristic-based techniques for University Timetabling problems. *OR Spectrum* 30 (01 2008), 167–190. https://doi.org/10.1007/s00291-007-0097-0

[10] Mazin Mohammed, Mohd Khanapi Abd Ghani, Abd Ghani, Omar Ibrahim Obaid, Salama Mostafa, Mohd, Mohd Ahmad, Dheyaa Ahmed, and Mohd Burhanuddin. 2017. A Review of Genetic Algorithm Application in Examination Timetabling Problem. *Journal of Engineering and Applied Sciences* 12 (01 2017), 5166–5181.

[11] José Moreira. 2008. A System for Automatic Construction of Exam Timetable Using Genetic Algorithms. *Tékhne - Revista de Estudos Politécnicos* 6 (06 2008).

[12] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 http://arxiv.org/abs/1707.06347