

Artificial intelligence

(report File)



*SIR:  
Haseeb Arshad*

Member name

19f-0240 Abdullah Awan

“Exam Schedule Generator Using Local

Search Algorithms”

**1. Introduction:**

In this Project, we implemented an exam schedule generator using Simulated Annealing algorithm to solve the exam timetable problem. The project aims to create a exam schedule for two weeks while satisfying various constraints

* Hard constraints
* Soft constraints

The solution displays the sorted exam schedule with zero clash using Graphical user interface (GUI) built using tkinter.

**2. Libraries Used:**

The code uses the following libraries:

|  |  |
| --- | --- |
| **numpy** | For numerical operations |
| **pandas** | For reading CSV files and storing data in Data Frame objects |
| **copy** | For creating deep copies of dictionaries |
| **math** | For mathematical functions |
| **random** | For generating random values in simulated annealing |
| **tkinter** | For creating GUI graphical user interface for exam schedule |
| **tkinter.ttk** | For styling the tkinter widgets |
| **tkinter.messagebox** | For displaying message boxes in the GUI |

**4. Working of the Code:**

1. The code starts by importing the necessary libraries and defining some constants related to the exam schedule generation process.
2. The function import\_data(fileDir) reads data from CSV files containing teacher, student, room, course, and student-course mapping data.
3. The function random\_solution(courses, students, rooms) generates a random initial solution for the exam schedule by randomly assigning rooms, teachers, dates, and times to each course.
4. The function calculate\_fitness(exam\_schedule, student\_courses, num\_days) calculates the fitness (number of constraint violations) of a given exam schedule based on hard constraints such as students not giving more than one exam at a time and teachers not invigilating two exams at the same time.
5. The function generate\_neighbor\_solution(current\_solution, rooms) generates a neighbor solution by randomly changing the date and room for a randomly selected exam in the current solution.
6. The function simulated\_annealing(initial\_solution, student\_courses, num\_days, max\_iterations, initial\_temperature, cooling\_rate) applies the simulated annealing algorithm to find the best exam schedule that minimizes the number of constraint violations. It iteratively generates neighbor solutions and probabilistically accepts them based on the Metropolis criterion.
7. There are functions to check various soft constraints such as no exams at 1 PM, no consecutive exams for students, preferred order of MG and CS courses, and faculty meeting constraints.
8. The function calculate\_soft\_constraints(exam\_schedule, student\_courses, num\_days) calculates the number of violations for each soft constraint.
9. The function display\_schedule\_gui(exam\_schedule\_df, soft\_constraints\_df) creates a GUI using tkinter to display the generated exam schedule and soft constraint violations in a table-like format.

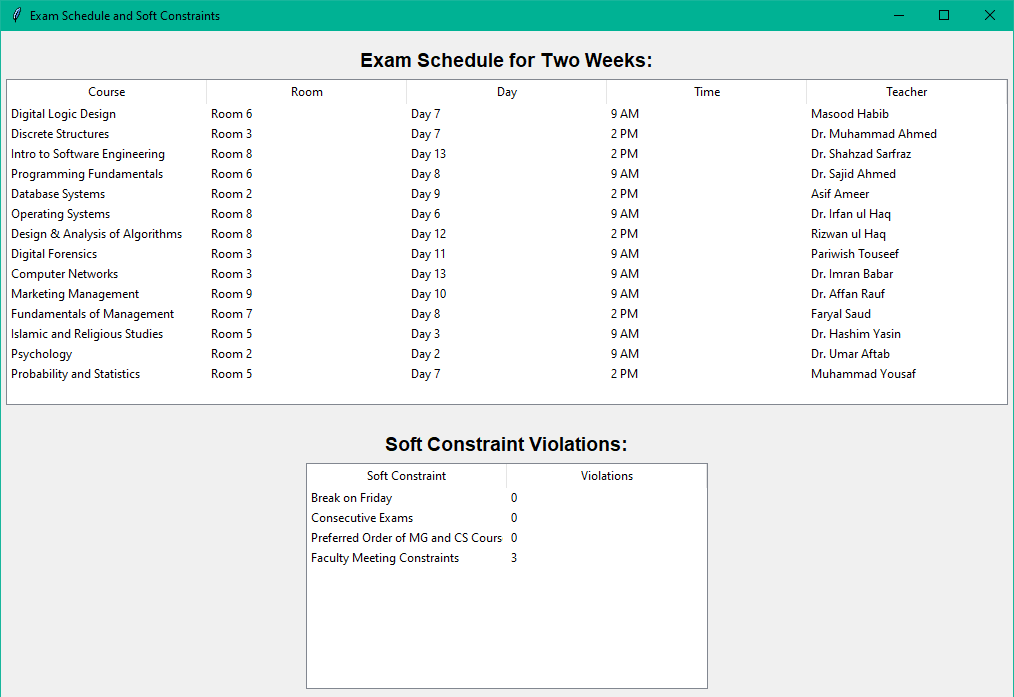
5. Terms Used:

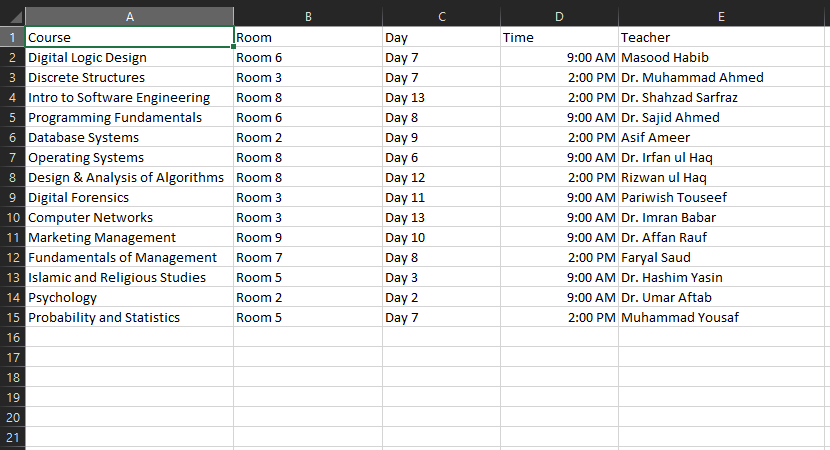
|  |  |
| --- | --- |
| Term | Description |
| Simulated Annealing | A probabilistic optimization algorithm inspired by the annealing process in metallurgy. It is used to find the global optimum in complex search spaces. |
| Hard Constraints | Constraints that must be satisfied in any valid solution. Failing to satisfy these results in an invalid solution. |
| Soft Constraints | Constraints that are desired to be satisfied, but it is acceptable if they are violated to some extent. |
| Fitness Function | A function that measures how well a solution satisfies the constraints. In this case, it is the number of constraint violations. |
| Neighbor Solution | A solution that is obtained from the current solution by making a small change or modification. Used in the search process. |
| Metropolis Criterion | A rule used in simulated annealing to decide whether to accept a neighbor solution or not based on its fitness and temperature. |
| GUI | Graphical User Interface, a visual way to interact with a computer program using buttons, menus, etc. |

**6. Conclusion:**

To find the best solution after applying other algorithms we finally use Simulated Annealing algorithm which generate not only the exams schedule also minimize the constraint violations and also generate CSV file of solutions with Graphical User Interface. Not only that code can also handle the more complex constraints and optimization for exams scheduling in real world scenarios. More than that our structure contains the best solution and use of pandas not only enhances the code functionality but also enhancing the usability of code.

Test Program output:





Code shows you GUI one of exam schedule timetable and second one is for soft constraints.

