# 1. Introduction

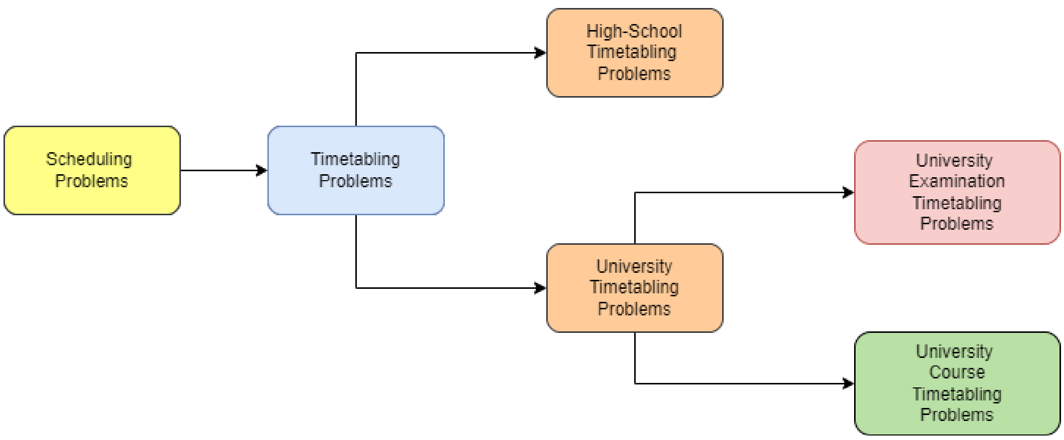
Timetabling : a very common multi-dimensional assignment problem, which needs to be solved at almost all educational institutions. It involves the assignment of courses to faculty members and the assignment of classrooms and time slots to these courses, such that the resources available are used optimally. The timetable should follow certain constraints like - no single teacher teaches more than one class at the same time, no single room is allocated for more than one class at the same time, no student has two different classes at the same time, etc.

Further, we need to achieve certain objectives such as maximum utilization of classrooms, assigning teachers to their preferred courses and preferred time slots, etc. In reality, a typical university timetabling problem would comprise thousands of courses, thousands of students, hundreds of instructors, hundreds of classrooms and other resources. Moreover, the timetabling problem has been classified as NP-hard optimization problem (i.e., no polynomial time algorithm is known to solve the problem), meaning that if all combinations were to be examined, the time to solution for reasonable problems would rise dramatically. **Therefore, in order to find optimal solutions to such problems, it is necessary to consider all the possible solutions and choose the best one that satisfies a wide range of constraints, preferences, and participants. Also, it should be solved in a reasonable amount of time**.

Although the University Timetable is a major, regular and complex administrative activity in most of the academic institutions, only a few organizations possess reliable automated timetable solvers, and fewer still possess solvers that require no manual intervention. However, most institutions employ the knowledge and experience of expert personnel with regard to the production of a good timetable that satisfies all given requirements.

The University Timetabling problems can be classified into two main categories: Course timetabling problems and Examination timetabling problems, where each problem has its own sets of constraints and requirements. A lot of research has been invested in order to provide automated support for solving a real-world timetabling problem. Researchers from different fields of operations research (graph coloring, network flow techniques) and artificial intelligence (simulated annealing, tabu search, genetic algorithms and constraint satisfaction) have contributed in solving the timetabling problem.

In this report, we have focused on University Course Timetabling Problem (UCTP).




**Classification of Scheduling Problems**

## 2. Description of the problem

In a University Course Timetabling Problem (UCTP), we are given the number of classes (C), teachers (T), venues (V), periods per day (P) and days in a week (D). We need to assign the classes and the teachers in such a manner that there are no clashes. In this problem, the number of times a class has to be held by a particular teacher in the given venue is given in the form of a matrix. Number of working days and number of periods are specified using which total number of time slots can be calculated. The problem has been designed to be totally constrained, i.e., each class, teacher and venue is required for each period. The optimal objective function for each of these problems is to allot class, teacher and venue for each of the time periods and ensure zero clashes among them.

## 3. Formulating the Problem

Let the :

- Number of Classes = C
- Number of Teachers = T
- Number of Venues = R
- Number of Periods per day = P
- Number of days per week = D
- Number of slots per week = K = P · D
- Decision Variable :

$$x_{ijkl} = \begin{Bmatrix} 1; & \text{if the i}^{th} \text{ class is taken by the } j^{th} \text{ teacher in the } k^{th} \text{ slot and in the } l^{th} \text{ room} \\ 0; & \text{otherwise} \end{Bmatrix}$$

### 3.A. Objective Function

The objective function for the problem is to minimize the number of clashes. Since the problem is to find a feasible solution, we may use other objective functions as well.

One such function can be to **minimize the summation of all decision variables** : This would ensure that our IP solver is not biased towards any decision variable. But the best objective function would just be a constant value as it won't take computational time to calculate the objective function value. If CPLEX is able to solve the IP, we will get a feasible solution.

**Objective Function :** Minimize $\sum_i \sum_j \sum_k \sum_l x_{ijkl}$

### 3.B. Constraints for our problem

#### 3.B.I. Requirement Constraint

$$\sum_k x_{ijkl} = M_{[C\cdot l+i],[j]} \ \forall \ i, j, l \text{ (where M = Matrix)}$$

#### 3.B.II. No clashes in classes

$$\sum_j \sum_l x_{ijkl} \leq 1 \ \forall \ k, i$$

#### 3.B.III. No clashes for teachers

$$\sum_i \sum_l x_{ijkl} \leq 1 \ \forall \ k, j$$

#### 3.B.IV. No clashes for rooms

$$\sum_i \sum_j x_{ijkl} \leq 1 \ \forall \ k, l$$

## 4. Sample input data set

We took sample data from the OR library provided to us.

- Number of Classes = 5
- Number of Teachers = 5
- Number of rooms available (Venues) = 5
- Number of Periods per day = 6
- Number of days per week = 5
- Number of slots per week = 30

**Matrix for the Problem (25 x 5)**

$$\text{Venue 1} \begin{bmatrix} 1 & 0 & 0 & 1 & 2 \\ 2 & 0 & 1 & 3 & 1 \\ 0 & 0 & 1 & 2 & 1 \\ 1 & 2 & 3 & 1 & 0 \\ 0 & 2 & 1 & 4 & 1 \end{bmatrix}$$

$$\text{Venue 2} \begin{bmatrix} 4 & 0 & 3 & 1 & 3 \\ 0 & 1 & 0 & 1 & 3 \\ 0 & 1 & 2 & 1 & 1 \\ 1 & 0 & 3 & 0 & 0 \\ 2 & 1 & 0 & 2 & 0 \end{bmatrix}$$

$$\text{Venue 3} \begin{bmatrix} 2 & 2 & 0 & 0 & 2 \\ 3 & 2 & 1 & 2 & 1 \\ 1 & 0 & 3 & 1 & 1 \\ 3 & 0 & 0 & 0 & 3 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\text{Venue 4} \begin{bmatrix} 1 & 2 & 0 & 0 & 1 \\ 0 & 2 & 3 & 1 & 0 \\ 2 & 0 & 2 & 0 & 4 \\ 1 & 3 & 0 & 2 & 1 \\ 1 & 1 & 0 & 2 & 1 \end{bmatrix}$$

$$\text{Venue 5} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 3 & 0 & 0 & 0 \\ 1 & 2 & 3 & 1 & 0 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 3 & 2 & 2 & 1 \end{bmatrix}$$

Where, the first five rows of matrix indicate the number of times each class-teacher combination will meet each other in venue 1 across the P periods. The next five rows indicate the number of times each class-teacher combination will meet each other in venue 2 across the P periods, and similarly for Venue 3, Venue 4 and Venue 5.

## 5. Computational results

We solved the problem using CPLEX with C++ in Visual Studio 2022. A 5-Dimensional matrix was created to store the decision variables. The time taken by the program to get the feasible arrangement of teachers and classes is shown below :

| S. No. | No. of classes | No. of teachers | No. of rooms | Time Taken (sec) |
|---|---|---|---|---|
| 1 | 4 | 4 | 4 | 0.20 |
| 2 | 5 | 5 | 5 | 0.43 |
| 3 | 6 | 6 | 6 | 0.87 |
| 4 | 7 | 7 | 7 | 28.49 |
| 5 | 8 | 8 | 8 | 32.34 |

As we can see in the $5^{th}$ case, the program takes nearly 0.5 minutes for the problem size of 8 classes, 8 teachers and 8 rooms. Therefore, if we apply it to the real life problems with hundreds of teachers and classes, it may take hours to arrive at a proper assignment.

## 6. Conclusion

We solved the University Course Timetabling Problem by the exact method. We formulated it as an Integer Programming problem. Since the data set had a smaller number of classes and teachers, we were able to solve it by exact method. The IP approach will take much more time for larger data sets. The time elapsed to get the feasible distribution of the classes and teachers for a week was 32.54 seconds for the problem size of 8 classes, 8 teachers and 8 rooms. **So, we requrie a more sophisticated algorithm to solve real life university timetable problems efficiently under reasonable time.**

## 7. References


1. https://www.itc2019.org/home

2. Schimmelpfeng, K., Helber, S. (2007). Application of a real-world university-course timetabling model solved by integer programming. 783–803.

3. El-sakka, T. (n.d.). University Course Timetable using Constraint Satisfaction and Optimization. 4(3), 83–95.

4. Rudová, H., Müller, T., Murray, K. (2016). Complex university course timetabling. May 2010, 187–207.

5. Babaei, H., Karimpour, J., Hadidi, A. (2015). Computers Industrial Engineering A survey of approaches for university course timetabling problem. Computers Industrial Engineering, 86, 43–59.

6. Zuzana, M. (2019). University course timetabling and International Timetabling Competition 2019.

7. Hav, J., Olsson, A., Persson, J. (n.d.). Modeling and optimization of university timetabling A case study in integer programming as.

8. Sandström, V. (2012). Scheduling of Teaching Resources and Classes using Mixed integer linear programming - A case study in a university of applied sciences.