International Journal of Advance and Innovative Research
Volume 12, Issue 2: April - June 2025
ISSN 2394 - 7780

TIME TABLING PROBLEM: A CASE STUDY USING INTEGER LINEAR PROGRAMMING MODEL

Ms. Rituparna Choudhary[1] and Dr. Ritu Khanna[2]
[1]Assistant Professor, Thakur College of Science and Commerce, Kandivali – East
[2]Professor, Department of Mathematics, Pacific University, Udaipur

ABSTRACT: -
Linear Programming (ILP) is a powerful Mathematical model and optimization technique that provides a structured and systematic approach to modelling and solving many real-life problems. Developing timetable for any institute can also be modelled using integer linear programming. In recent time, modelling and formulating a timetabling problem using integer linear Programming (ILP) is a highly studied and practically important optimization problem in the field of operations research and computational Mathematics. It consists of scheduling a set of events such as classes, examinations, or work shifts into specific time slots with considering several constraints. Integer

In this paper we will try to formulate the Mathematical model of a timetabling problem of a reputed college of Mumbai. After deriving the model, we will also try to optimize this model using algorithms which are used for integer linear programming problem. Various algorithms are present to solve integer linear programming problems. Brunch and bound method or cutting plan method is one of them. In this paper we will try to solve this problem using Python. In Python there are external packages which are capable for solving any integer linear programming problem.

Genetic algorithm is another optimization techniques which use the similar Mathematical principles to optimize various complex real-life systems. In this paper we will try to optimize same real-life timetabling problem using Genetic algorithms. In this way we can do a comparison analysis between these two optimization techniques.

Keywords: - *Optimization, Timetabling, ILP, GA, Comparison.*

INTRODUCTION

Modelling timetabling problem using integer linear Programming (ILP) is a highly studied and practically important optimization problem in computational Mathematics and in the field of operations research. **Daskalaki et al. [1]** proposed that a timetable consists of scheduling a set of events like classes, examinations, or work hours into specific time slots with considering several constraints. Resource availability, conflict avoidance, and preferences can be considered as the most relevant constraints which make that the problem highly combinatorial and complex. **Maaroufi et al. [7]** defined Integer Linear Programming (ILP) as a powerful Mathematical framework that provides a structured and systematic approach to modelling and solving such problems.

In this introduction, we constate into the fundamental aspects of the timetabling problem, the key motivations for employing ILP, which is defined by **Lawrie [17]**, and the critical modelling considerations.

The Nature of the Timetabling Problem: Real-life timetabling problems arise in very different application domains, such as transportation networks, universities, hospitals, colleges, schools and personnel scheduling. **Prabodanie [10]** explains the application of timetabling related to the fundamental problem of how to use resources like classrooms, personnel, or machines on specific events such as lectures, practices, examinations, shifts, to optimize an objective function against a set of constraints that must be satisfied. **Choudhari and Khanna [20]** defined that the problems of this type are often very hard due to their inherent complexity, which rises from large solution space and variety of constraints.

The size of the feasible schedules grows exponentially in number of events and time slots. **Laisupannawong, et al. [16]** defined many constraining requirements and constraints. For example, avoid clashes in between constrained resources, and learners' preferences, are generally in conflict with one another and demand non-trivial trade-offs. All the constraints and objectives of the problem are very sensitive and significantly different from application to application.

The Integer Linear Programming Approach: Integer linear programming provides a strong mathematical approach for solving the timetabling problem. **Torres et al. [15]** explained that solution of any timetabling problem is achieved by formulating the problem as a set of linear equations and inequalities with the choice of integer decision variables that genuinely encapsulate both structure and constraints of the problem under

ISSN 2394 - 7780

consideration. **Jain et al. [19]** stated that there are many benefits of integer linear programming (ILP). Some of the benefits of using ILP are:

(a) Expressiveness: Complex constraints can be explicitly represented, such as mutual exclusivity, fairness, or capacity limits.

(b) Optimality Guarantees: Solution can be obtained by integer linear programming (ILP) solvers, such as Gurobi or PuLP,.

(c) Flexibility: Integer linear programming (ILP) models can be adapted and extended to include new complex constraints or objectives with minimal reformulation.

Important Components of an integer linear programming (ILP) Model for Timetabling

In past, **Benitez [14]** defined that to formulate a good integer linear programming (ILP) model for this timetabling problem, the following important components need to be considered carefully:

(a) Decision Variables: These are the variables for assigning events to time slots and resources. For example, a binary variable (0 or 1) x_{ijt} can be used as an indicator of whether event i is assigned to time slot t and resource j.

(b) Objective Function: The measure of the quality of a solution is done by the objective function. It may be used to minimize scheduling conflicts, maximize resource utilization, or balance workloads. Example: Minimize the total deviations from preferred time slots of all the courses.

(c) Types of Constraints: There are mainly two types of constraints in a ILP model. **Pereira et al. [8]** defined these two types of constraints as hard and soft constraint as a component of fitness function.

Hard Constraints: These must be violated at no point of time, like it being impossible for two overlapping events like lecture of same teacher to be conducted in same time slot.

Soft Constraints: These are desirable but not obligatory. **Tyagi and Tyagi [5]** explained an example, lecturers' specific subject classes are preferable but not made possible due to conflicting with their desires and arrangements for violations.

In this paper a modelling is done of a real-life time tableting problem using integer linear programming approach. The model formulated above was solved using linear programming solver of Python known as PuLP At the last of this paper a sample timetable is formed which meets all the limitations and constraints of the specific problem defined. This gives an idea of a Mathematical model of a real life problem and also explains programming approach to solve any Mathematical model.

OBJECTIVES OF THE STUDY

The four aims of this paper are as follows:

1) The ILP Model: A mathematical formulation of the time-tabling problem using Integer Linear Programming that defines decision variables, constraints, and an objective function that incorporates the problem requirements and goals.

2) Solution of the ILP Model: The model devised will be shown to be capable of providing feasible and optimal time-tabling solutions using an appropriate solver or algorithm.

3) Python PuLP solver: A python programming will be developed which will be capable of solving this model with satisfying all limitations and to get a sample solution as timetable using this programming method.

METHODOLOGY

The methodology ensures a systematic and comprehensive evaluation of both models in solving the time-tabling problem.

1. Problem Definition and Formulation

2. Development of the ILP Model the problem

3. Implementation and Testing

5. Conclusions and Recommendations

MODELLING AND ANALYSIS

Before starting with the modelling of the timetable of Bachelor oi Mass Media (BMM) department of a College of Mumbai. Let us consider the following considerations.

ISSN 2394 - 7780

1. There are 21 different courses in BMM programme. The courses are named as Introduction to New Media (INM), Basics of Radio & TV (BR&TV), Overview of Print Production (OPP), History of Media (HOM), Leadership in Management (LM), Marketing Mix – II (MM-II), Translation Skills (TS), Introduction to Computers – II(IC-II), AEC, Electronic Media – Il (EM II), Writing and Editing for Media (WEM), Media Law & Ethics (MLE), Mass Media Research (MMR), Film Communication – II (FC II), Computers and Multimedia – II (CMM II), Digital Media (DM), Advertising Design (Projects) (AD), Media Planning and Buying (MPB), Brand Management (BR), Entertainment and Media Marketing (EMM), Advertising in Contemporary Issues (ACI)

2. There is total 4 faculties of the department who will going to take lectures in those programme and courses. The short name of the faculties are as follows: AC, NM, FZG and RK.

3. There are six time slots available for the lectures of these classes. These timeslots are as follows: (7:30 am – 8:30 am), (8:30 am – 9:30 am), (10:00 am-11:00 am), (11:00 am-12:00 pm) and (12:00 pm -1:00 pm) and (1:30 pm – 2:30 pm)

4. There are 3 different group of students who will going to attain these sessions. They are as follows: TYBAMMC, SYBAMMC, FYBAMMC.

Now let us check some constraints for modelling this timetable problem. The main constraints are:

1) A student group or a teacher group cannot assign two classes simultaneously and that situation will be called as clash in timetable.

2) An individual teacher cannot be allotted more than 16 lectures in a single week. They should get equal or less than 16 lectures.

3) For different courses there are different requirement of lectures and practical per week.

Sr No	Courses	Sort Form	Lecture
1	Introduction to New Media	INM	2
2	Basics of Radio & TV	BR&TV	2
3	Overview of Print Production	OPP	2
4	History of Media	HOM	2
5	Leadership in Management	LM	2
6	Marketing Mix - II	MM-II	2
7	Translation Skills	TS	1
8	Introduction to Computers - II	IC-II	2
9		AEC	2
10	Electronic Media - Il	EM 2	3
11	Writing and Editing for Media	WEM	3
12	Media Law & Ethics	MLE	3
13	Mass Media Research	MMR	3
14	Film Communication - II	FC 2	3
15	Computers and Multimedia - II	CMM II	3
16	Digital Media	DM	3
17	Advertising Design (Projects)	AD	3
18	Media Planning and Buying	MPB	3
19	Brand Management	BM	3
20	Entertainment and Media Marketing	EMM	3
21	Advertising in Contemporary Issues	ACI	3
Total Departmental Workload			**53**

The specification of teachers are as follows:

Teacher Name	Course	Sort Form	No Lecture
AC	Introduction to New Media	INM	2
	Basics of Radio & TV	BR&TV	2
	Film Communication-II	FC 2	3
	Writing and Editing for Media	WEM	3
	Ad Design	AD	3

ISSN 2394 - 7780

	Brand Management	BR	3
	Total		**16**
NM	Overview of Print Production	OPP	2
	History of Media	HOM	2
	Media Law & Ethics	MLE	3
	Mass Media Research	MMR	3
	Entertainment and Media Marketing	EMM	3
	Advertising in Contemporary Issues	ACI	3
	Total		**16**
FZG		AEC	2
	Leadership in Management	LM	2
	Marketing Mix-II	MM-II	2
	Translation Skills	TS	1
	Electronic Media-II	EM-II	3
	Digital Media	DM	3
	Consumer Behavour	CB	3
	Total		**16**
RK	Introduction to Computers - II	IC-II	2
	Computers and Multimedia-II	CMM-II	3
	Total		**05**

5) All the programmes are starting at different time slots. Now let us list down all different time slots for the courses

Programme	7:30-8:30	8:30-9:30	10:00-11:00	11:00-12:00	12:00-1:00	1:30-2:30
FYBAMMC	Lecture	Lecture	Lecture	Lecture		
SYBAMMC	Lecture	Lecture	Lecture	Lecture		
TYBAMMC				Lecture	Lecture	Lecture

6) Lectures will be continued for six days of a week {MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY}

Mathematical Linear Integer Programming Model

To solve this timetable problem, the following Linear Integer Programming (LIP) model is formulated. The goal is to allocate teachers and time slots to all courses while satisfying the constraints. **Sánchez et al. [3]** defined various set of a time tabling problem as follows:

1. Set and Indices:

Course C = {“INM”, “BR&TV”, “OPP”, “HOM”, “LM”, “MM-II”, “TS”, “IC-II”, “AEC”, “EM II”, “WEM”, “MLE”, “MMR”, “FC II”, “CMM II”, “DM”, “AD”, “MPB”, “BR, “EMM”, “ACI”}

Time slots T = {(7:30 am – 8:30 am), (8:30 am – 9:30 am), (10:00 am – 11:00 am), (11:00 am -12:00 pm), (12:00 pm -1:00 pm) and (1:30 pm -2:30 pm)}.

Gorup of students G = {TYBAMMC, SYBAMMC, FYBAMMC}.

Faculties F = {AC, NM, FZG, RK}

Days of the week D = {MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY}

2. Decision Variables:

Define a binary decision variable as follows:

$X_{d,t,f,c,g} \in \{0, 1\}$, with $X_{d,t,f,c,g} = 1$

if faculty f teachers course c to the group g on day d at time slot t , Otherwise,

$X_{d,t,f,c,g} = 0$.

Constraints:

1. Faculty workload constraint:

Each faculty must meet its required number of lectures per week:

ISSN 2394 - 7780

$\sum_{d\in D}\sum_{t\in T}\sum_{g\in G} X_{d,t,f,c,g} = W_f \ , \forall\ f \in F$, where, W_f is the predefined workload of faculty f . (16 for AC, NM and FZG and 5 for RK)

2. Course requirement constraint:
A course should meet its specified or required number of lectures :

$$\sum_{d\in D}\sum_{t\in T}\sum_{f\in F}\sum_{g\in G} X_{d,t,f,c,g} = W_c \ , \forall\ c \in C$$

Where W_c is the number of lectures required for course c which is given in the problem statement.

3. Faculty Specialization Constraint:
Ensure a faculty member only teachers course assigned to them.

$X_{d,t,f,c,g} = 0, \forall\ (f,c) \notin$ Faculty Specialization.

4. Timeslot and conflict constraint:
Each faculty member can only teach one slot per course per time slot.

$$\sum_{c\ \in\ C}\sum_{g\ \in\ G} X_{d,t,f,c,g} \leq 1, \forall\ d \in D, t \in T, f \in F$$

5. Student Group Constraint:
Each group can attend only one lecture per time slot.

$$\sum_{c\ \in\ C}\sum_{f\ \in\ F} X_{d,t,f,c,g} \leq 1, \forall\ d \in D, t \in T, g \in G$$

Objective Function:
Minimize the total number of unused slots:

Minimize $\sum_{d\in D}\sum_{t\in T}(\ 1 - \sum_{f\in F}\sum_{c\in C}\sum_{g\in G}\ X_{d,t,f,c,g})$

Which minimizes the number of empty lecture slots in the timetable.

In this model the **objective function** is to minimize the unused time slots to get an optimal time table schedule. There are four constraints. These constraints ensure that

1)The required number of lectures for each course are satisfied.

2)There should not be any clash in schedule for teacher and student.

3)Teachers are assigned to courses according to their specialization and capacity limits.

4) Permissible time slots for each course are maintained.

In this paper the above model is solved using a Mixed Integer Programming (MIP) solver known as PuLP in Python. Python's PuLP model can solve this type of problem and a required result can be obtained.

The explanation of the Python code written is as follows
The following Python script generates a timetable for faculty members, student groups, and courses while considering faculty specialization and workload. The final schedule is saved as a CSV file. Let's break it down step by step:

Step 1: Import Necessary Modules

Python Code:

```
import random

import csv
```

Where, random module is used to randomly allocate faculty, courses, and student groups and csv module is used to write the final timetable into a CSV file.

ISSN 2394 - 7780

Step 2: Define Available Data

The data of the timetabling problem is defined in this step.

Python Code:

```
courses = { "INM": 2, "BR&TV": 2, "OPP": 2, "HOM": 2, "LM": 2, "MM-II": 2, "TS": 1, "IC-II": 2,
"AEC": 2,    "EM II": 3, "WEM": 3, "MLE": 3, "MMR": 3, "FC II": 3, "CMM II": 3, "DM": 3, "AD": 3,
"MPB": 3,    "BR": 3, "EMM": 3, "ACI": 3 }.
```

(a) A dictionary courses holds course names as keys and the number of times each course should be scheduled as values

Python Code:

```
faculty_specialization = {"AC": ["INM", "BR&TV", "FC II", "WEM", "AD", "BR"],
   "NM": ["OPP", "HOM", "MLE", "MMR", "EMM", "ACI"],
   "FZG": ["AEC", "LM", "MM-II", "TS", "EM II", "DM", "MPB"], "RK": ["IC-II", "CMM II"]}
```

(b) Faculty specialization maps faculty members to the courses they are qualified to teach.

Python Code:

```
faculty_workload = {"AC": 16, "NM": 16, "FZG": 16, "RK": 5}
```

(c) Faculty workload sets the maximum number of sessions each faculty member can handle.

Python Code:

```
time_slots = ["7:30-8:30", "8:30-9:30", "10:00-11:00", "11:00-12:00", "12:00-1:00", "1:30-2:30"]
days = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]
students_groups = ["FYBAMMC", "SYBAMMC", "TYBAMMC"]
```

(d) Time slots, days, and student's groups define available scheduling options.

Step 3: Initialize Schedule and Course Allocation Tracking

Schedule is a variable which stores the generated timetable.

Python Code:

```
schedule = []
allocated_courses = {course: 0 for course in courses}
```

allocated_courses is the second variable which keeps track of how many times each course has been assigned

Step 4: Assign Courses in the Initial Loop

Python Code:

```
for day in days:
   for slot in time_slots:
```

(a) Iterates the loop over each day and time slot.

ISSN 2394 - 7780

Python Code:

```
available_faculty = [f for f in faculty_specialization.keys() if faculty_workload[f] > 0]

if not available_faculty:

    continue
```

(b) Selects faculty members who still have workload capacity. If no faculty is available, skips the current time slot.

Python Code:

```
faculty = random.choice(available_faculty)

available_courses = [c for c in faculty_specialization[faculty] if allocated_courses[c] < courses[c]]:
```

(c) Randomly selects a faculty member. Identifies courses that the selected faculty can teach and are not fully scheduled

Python Code:

```
if available_courses:

    course = random.choice(available_courses)

    student_group = random.choice(students_groups)
```

(d) Randomly picks a course and a student group

Python Code:

```
schedule.append([day, slot, faculty, course, student_group])

allocated_courses[course] += 1

faculty_workload[faculty] -= 1
```

(e) Adds the class to the schedule. Updates course allocation and faculty workload.

Python Code:

```
if allocated_courses[course] == courses[course]:

    for fac in faculty_specialization:

        if course in faculty_specialization[fac]:

            faculty_specialization[fac].remove(course)
```

(f) If a course reaches its required allocation, it is removed from all faculty specialization lists.

Step 5: Fill Remaining Workload

Python Code:

```
while sum(faculty_workload.values())> 0:
```

(a) Ensures that all faculty workload is exhausted.

ISSN 2394 - 7780

Python Code:

```
for faculty in faculty_workload:
    if faculty_workload[faculty] > 0:
```

(b) Loops through each faculty to check if they still have pending workload.

Python Code:

```
for course in faculty_specialization[faculty]:
    if allocated_courses[course] < courses[course]:
```

(c) Finds a course for the faculty that is not fully allocated.

Python Code:

```
day = random.choice(days)
slot = random.choice(time_slots)
student_group = random.choice(students_groups)
schedule.append([day, slot, faculty, course, student_group])
allocated_courses[course] += 1
faculty_workload[faculty] -= 1
```

(d) Randomly assigns an available time slot and student group.

Python Code:

```
if allocated_courses[course] == courses[course]:
    faculty_specialization[faculty]. remove(course)
```

(e) Removes the course from faculty specialization if it reaches full allocation.

Python Code:

```
if faculty_workload[faculty] == 0:
    break
```

(f) Stops assigning further slots if the faculty has no remaining workload.

Step 6: Save the Timetable to a CSV File

Python Code:

```
csv_filename = "bmm_timetable.csv"
with open(csv_filename, "w", newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["Day", "Time Slot", "Faculty", "Course", "Student Group"])
    writer.writerows(schedule)
```

(a) This code opens a file in write mode, writes the column headers: Day, Time Slot, Faculty, Course, Student Group and writes the generated schedule into the file.

ISSN 2394 - 7780

Python Code:

```
print(f"Timetable has been generated and saved as {csv_filename}."
```

(b) Below code displays a confirmation message.

The output in csv. format will look as follows:

(1)

Day	Time Slot	Faculty	Course	Student Group
Monday	7:30-8:30	AC	AD	TYBAMMC
Monday	8:30-9:30	AC	BR&TV	TYBAMMC
Monday	11:00-12:00	AC	INM	SYBAMMC
Tuesday	8:30-9:30	AC	BR	TYBAMMC
Tuesday	12:00-1:00	AC	BR&TV	FYBAMMC
Wednesday	7:30-8:30	AC	INM	TYBAMMC
Wednesday	11:00-12:00	AC	WEM	SYBAMMC
Friday	1:30-2:30	AC	BR	FYBAMMC
Saturday	7:30-8:30	AC	WEM	TYBAMMC
Saturday	8:30-9:30	AC	BR	SYBAMMC
Saturday	12:00-1:00	AC	FC II	TYBAMMC
Saturday	1:30-2:30	AC	WEM	SYBAMMC
Friday	1:30-2:30	AC	FC II	SYBAMMC
Friday	1:30-2:30	AC	AD	FYBAMMC
Tuesday	7:30-8:30	AC	FC II	SYBAMMC
Thursday	11:00-12:00	AC	AD	TYBAMMC

(2)

Day	Time Slot	Faculty	Course	Student Group
Tuesday	7:30-8:30	FZG	MPB	SYBAMMC
Tuesday	10:00-11:00	FZG	DM	FYBAMMC
Tuesday	1:30-2:30	FZG	TS	FYBAMMC
Wednesday	12:00-1:00	FZG	LM	SYBAMMC
Thursday	11:00-12:00	FZG	LM	FYBAMMC
Friday	11:00-12:00	FZG	EM II	FYBAMMC
Friday	12:00-1:00	FZG	DM	SYBAMMC
Saturday	11:00-12:00	FZG	DM	FYBAMMC
Tuesday	11:00-12:00	FZG	AEC	TYBAMMC
Tuesday	8:30-9:30	FZG	MM-II	TYBAMMC
Monday	10:00-11:00	FZG	EM II	FYBAMMC
Monday	12:00-1:00	FZG	MPB	TYBAMMC
Thursday	1:30-2:30	FZG	AEC	TYBAMMC
Monday	7:30-8:30	FZG	EM II	SYBAMMC
Tuesday	11:00-12:00	FZG	MM-II	FYBAMMC
Wednesday	1:30-2:30	FZG	MPB	TYBAMMC

(3)

Day	Time Slot	Faculty	Course	Student Group
Monday	10:00-11:00	NM	HOM	TYBAMMC
Monday	1:30-2:30	NM	HOM	SYBAMMC
Wednesday	10:00-11:00	NM	ACI	FYBAMMC
Wednesday	1:30-2:30	NM	ACI	TYBAMMC
Thursday	10:00-11:00	NM	MLE	FYBAMMC
Thursday	12:00-1:00	NM	MMR	SYBAMMC
Thursday	1:30-2:30	NM	ACI	TYBAMMC

ISSN 2394 - 7780

Friday	7:30-8:30	NM	OPP	TYBAMMC
Friday	8:30-9:30	NM	EMM	TYBAMMC
Friday	10:00-11:00	NM	OPP	FYBAMMC
Saturday	10:00-11:00	NM	MLE	TYBAMMC
Thursday	8:30-9:30	NM	MLE	FYBAMMC
Tuesday	8:30-9:30	NM	EMM	TYBAMMC
Saturday	10:00-11:00	NM	MMR	TYBAMMC
Thursday	11:00-12:00	NM	EMM	SYBAMMC
Wednesday	12:00-1:00	NM	MMR	SYBAMMC

(4)

Day	Time Slot	Faculty	Course	Student Group
Monday	12:00-1:00	RK	CMM II	SYBAMMC
Tuesday	11:00-12:00	RK	IC-II	SYBAMMC
Wednesday	8:30-9:30	RK	CMM II	SYBAMMC
Thursday	7:30-8:30	RK	IC-II	FYBAMMC
Thursday	8:30-9:30	RK	CMM II	TYBAMMC

CONCLUSION:

Hence from the above model and solution we can conclude that LIP is a mathematical optimization approach for problems with linear objective functions and constraints, where some or all variables are restricted to integer values. Exact Solutions can be determined using linear programming models. If the problem is feasible, LIP provides an optimal solution. This method is very efficient for Well-Structured Problems: LIP solvers (like Python PuLP) are highly optimized for linear problems with integer constraints. LIP excels in handling strict, well-defined constraints. But there are some weaknesses. The solution of large-scale or very complex problems (e.g., having many variables or constraints) can be computationally expensive. Again, LIP only supports linear models, which makes it unsuitable for nonlinear or black-box objective functions. For highly combinatorial problems, LIP can get stuck in a locally optimal solution or may require preprocessing for improved performance.

Hence, LIP should be used when the objective and constraints of a problem are linear or in a applications where exact solutions required or for structured problems in which solvers are well-tuned for the problem size and complexity

REFERENCES:

[1] **Daskalaki, S., Birbas, T., & Housos, E.** (2004). An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research, 153*(1), 117–135.

[2] **Aldy Gunawan, K. M. Ng and H. L. Ong** (2008). A Genetic Algorithm for the Teacher Assignment Problem for a University in Indonesia, Information and Management Sciences Volume 19, Number 1, pp. 1-16, 2008.

[3] **Sánchez-Partida, D., Martínez-Flores, J. L., & Olivares-Benítez, E.** (2014). An integer linear programming model for a university timetabling problem considering time windows and consecutive periods. Journal of Applied Operational Research, 6(3), 158–173. UPAEP University, Puebla, México.

[4] **Thongsanit, K.** (2014). Solving the Course - Classroom Assignment Problem for a University. Silpakorn U Science & Tech J, 8(1). Department of Industrial Engineering and Management, Faculty of Engineering and Technology, Silpakorn University, Nakhon Pathom, Thailand.

[5] **Tyagi, K., & Tyagi, K.** (2015). A comparative analysis of optimization techniques. International Journal of Computer Applications, 131(10), 6.

[6] **Haldurai, L., Madhubala, T., & Rajalakshmi, R.** (2016). A study on genetic algorithm and its applications. International Journal of Computer Sciences and Engineering Open Access Review Paper, 4(10), 139–143.

[7] **Maaroufi Mars, F., Camus, H., & Korbaa, O.** (2016). A Mixed Integer Linear Programming Approach to Schedule the Operating Room. In 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 9-12). DOI: 10.1109/SMC35644.2016.

[8] **Pereira, V., & Costa, H. G.** (2016). Linear integer model for the course timetabling problem of a faculty in Rio de Janeiro. *Advances in Operations Research, 2016*, Article 7597062. https://doi.org/10.1155/2016/7597062

[9] **Herath, Achini Kumari**, (2017) "Genetic Algorithm For University Course Timetabling Problem". Electronic Theses and Disser tations . 443.

[10] **Prabodanie, R. A. R.** (2017). An integer programming model for a complex university timetabling problem: A case study. *Industrial Engineering & Management Systems, 16*(1), 141–153. https://doi.org/10.7232/iems.2017.16.1.141.

[11]. **Maidamisa, A. A., & Odiniya, H. A.** (2017). Integer linear programming applied to nurses rostering problem. *International Journal of Science and Research (IJSR), 6*(8), 2319-7064.

[12]. **Bei Wang, Yuejie Geng and Zhigang Zhang** (2019), Applying genetic algorithm to university classroom arrangement problem, Journal of Physics: Conference Series 1325 (2019) 012157, IOP Publishing doi:10.1088/1742-6596/1325/1/012157.

[13] **Alghamdi, H., Alsubait, T., Alhakami, H., & Baz, A.** (2020). A Review of Optimization Algorithms for University Timetable Scheduling. Engineering, Technology & Applied Science Research, 10(6), 6410-6417.

[14]. **Benitez Galle, M. D.** (2020). Implementation of an integer linear programming model for the timetabling problem of the Louvain School of Management (Promoter: Professor Daniele Catanzaro). Louvain School of Management, UCLouvain.

[15]. **Torres, M. C., Villegas, K. K. S., & Gavina, M. K. A.** (2021). Solving faculty-course allocation problem using integer programming model. *Philippine Journal of Science, 150*(4), 679-689.

[16]. **Laisupannawong, T., & Sumetthapiwat, S.** (2024). An integer linear programming model for the examination timetabling problem: A case study of a university in Thailand. *Advances in Operations Research, 2024*, Article ID 6636563, 17 pages.

[17] **Lawrie, N. L**. (1969). An integer linear programming model of a school timetabling problem.307–316.

[18]. **Bradley, S. P., Hax, A. C., & Magnanti, T. L**. (1977). Applied Mathematical Programming (Chapter 9, pp. 272-319). Addison-Wesley.

[19]. **Jain, N., Jain, S., & Khanna, R.** (2019**).** Goal optimization for electricity management. *International Journal of Engineering and Technical Research (IJETR)*, 9(6), 1–7.

[20]. **Choudhari, S. D., & Khanna, R**. (2014). Flow shop scheduling problems: A survey. *International Journal of Scientific & Engineering Research*, 5(12), 34–38.

[21]. **Khanna, R., & Patel, P.** (2023). Practical and adaptable applications of goal programming: A literature review. *Journal of Advanced Zoology*, 44(5).

[22]. **Dahake, R., Lakshmi, G. P., & Khanna, R.** (2018). Advanced analysis of passengers flow for automated service devices at airport terminal. *[PDF document]*. Retrieved from ResearchGate: https://www.researchgate.net