CAUCHY – Jurnal Matematika Murni dan Aplikasi
Volume 11 (1) (2026), Pages 61-74
p-ISSN: 2086-0382; e-ISSN: 2477-3344

University Scheduling Optimization Using Integer Programming: A Case Study

Gayus Simarmata[1*], Rajainal Saragih[2], and Anil Hakim Syofra[3]

[1]*Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas HKBP Nommensen Pematangsiantar, Pematangsiantar, Indonesia*
[2]*Department of Computer Engineering, Faculty of Science and Technology, Politeknik Bisnis Indonesia, Pematangsiantar, Indonesia*
[3]*Department of Mathematics Education, Faculty of Education and Teacher Training, Universitas Asahan, Kisaran, Indonesia*

Abstract

This paper presents an Integer Linear Programming (ILP) model to construct a weekly lecture timetable for the Mathematics Study Program at HKBP Nommensen University, Pematangsiantar. The case study comprises 25 courses, three rooms (RK 11, RK 12, and LAB 1), five teaching days (Monday–Friday), and 13 time periods per day. The model enforces hard constraints on room, lecturer, and cohort non-overlap; consecutive periods according to credit load; room-type compatibility between theory and practicum sessions; and an institutional worship-time restriction on Tuesday. Lecturers' availability is represented by a binary acceptance matrix collected at the course level, and rejected time periods are penalized in the objective. The ILP is implemented in Python using the PuLP (Python Linear Programming) library and solved with the CBC (Coin-or Branch and Cut) solver. For the real instance, the solver returns an optimal solution with objective value $Z^* = 0$ (no scheduled period falls in a rejected slot) in approximately 94 seconds. The resulting timetable is conflict-free and operationally interpretable, with a weekly room-time utilization of about 31.3%. To support verification and communication to stakeholders, the paper also provides a heatmap of the acceptance matrix and a graphical timetable by room and day.

Keywords: Branch-and-cut; Course Timetabling; Integer Linear Programming; PuLP; University Scheduling.

1. Introduction

Course scheduling is a recurrent operational task in higher education institutions. Each semester, a timetable must coordinate courses, lecturers, student cohorts, rooms, and a finite set of time periods under institutional rules. In practice, manual scheduling is time-consuming and prone to clashes, especially when lecturers teach multiple courses and cohorts must attend a fixed set of classes without overlap. From an optimization perspective, university timetabling is widely recognized as a combinatorial problem that is commonly classified as NP-hard, because it requires identifying a best allocation over many interacting assignments within limited resources [1].

[*]**Corresponding author**. E-mail: gayuspermata224@gmail.com

Submitted: November 27, 2025 Reviewed: December 16, 2025 Accepted: December 29, 2025
DOI: https://doi.org/10.18860/cauchy.v11i1.37815

University course timetabling problems are often categorized into student-based and curriculum-based variants [2, 3]. This study considers a curriculum-based setting, where courses within a study program are assigned to predefined time and room slots. The assignment must satisfy hard constraints, such as avoiding conflicts involving lecturers, cohorts, and rooms [4], while also addressing soft considerations, such as lecturers' time preferences and the reduction of undesirable placements [5].

A broad spectrum of solution methods has been proposed for university timetabling. Exact approaches, including Integer Linear Programming (ILP) and related formulations such as graph coloring, aim to obtain provably optimal solutions under a precise mathematical model [6, 7]. However, exact approaches may face scalability challenges as the problem size and constraint complexity grow. In contrast, heuristic and metaheuristic methods—such as Genetic Algorithms, Tabu Search, and evolutionary strategies—are frequently adopted because they can produce high-quality schedules within reasonable computation time for complex, real-world constraints [8–13].

Despite the popularity of heuristics, ILP remains attractive in departmental or program-level scheduling because it offers transparency, interpretability, and direct encoding of institutional rules. Several studies have reported successful ILP deployments in practical contexts, although applications are often limited to a single department or relatively moderate instances [14, 15]. In operational environments, a common strategy is to solve scheduling at the program level while coordinating room usage within the local infrastructure, which helps maintain implementability while keeping the model scope manageable [16, 17].

This paper addresses the weekly timetabling of the Mathematics Study Program at HKBP Nommensen University, Pematangsiantar, under institutional constraints including multi-period course delivery, differentiation between theory and practicum rooms, and a fixed worship-time restriction. The study uses real academic data to evaluate whether an optimization model can produce a conflict-free timetable while aligning assignments with lecturers' declared availability.

The novelty and contribution of this work are as follows. First, we develop a program-level ILP formulation that simultaneously enforces room, lecturer, and cohort non-overlap while guaranteeing consecutive time blocks for multi-credit courses through start-time variables. Second, we incorporate room-type compatibility (regular rooms versus LAB 1) and an institutional worship-time restriction within the same constraint system to ensure operational feasibility. Third, we integrate lecturers' availability using a binary acceptance matrix collected at the course level and minimize rejected assignments via a transparent penalty objective, complemented by visual diagnostics (heatmap and schedule overlay) to support validation. Accordingly, the objective of this study is to generate a feasible and interpretable timetable that minimizes preference violations under the stated constraints.

The remainder of this paper is organized as follows: Section 2 describes the case-study setting, data, and ILP formulation; Section 3 reports the computational results and their interpretation; and Section 4 concludes the paper and outlines future work.

2. Methods

This study adopts an applied quantitative approach with a case-study design [16]. The optimization model is developed and evaluated using real academic scheduling data from the Mathematics Study Program, Faculty of Mathematics and Natural Sciences, HKBP Nommensen University, Pematangsiantar. A case-study design is appropriate because university timetabling is an operational decision problem whose constraints and feasibility requirements depend strongly on institutional policies, available resources, and local academic practices.

2.1. Data collection and scheduling setting

The study begins with the collection of information required to define a realistic timetabling instance. The collected data include: (i) the list of courses and their credit loads; (ii) lecturer

assignments for each course; (iii) cohort groupings (class/semester) that must not overlap; (iv) available rooms and their functional types (regular rooms versus laboratory); (v) the weekly scheduling horizon in terms of days and time periods; and (vi) institutional restrictions and policies, including worship time and non-lecture days. These data constitute the empirical basis of the ILP formulation and ensure that the optimization problem accurately reflects real operational conditions.

The dataset is derived from academic records covering the 2022–2025 academic years. The weekly instance considered in this paper involves 25 courses (active learning groups) to be scheduled within one academic week. The scheduling horizon consists of five lecture days (Monday–Friday) and 13 time periods per day, where each period has a fixed duration of 50 minutes. The available infrastructure includes three rooms: two regular classrooms (RK 11 and RK 12) and one laboratory (LAB 1). The course data (including classes/semesters and supporting lecturers) are summarized in Table 1.

Table 1: Data on Courses, Classes, and Supporting Lecturers

Label	MK Code	Subject	Credits	Class/ Semester	Supporting Lecturer
1	AM0111	Basic Calculus Response	1	AM.1/1	FLOG
2	AM0231	Introduction to Logic and Sets	3	AM.1/1	MS
3	AM0431	Number Theory	3	AM.1/1	HS
4	AM0631	Analytical Geometry	3	AM.1/1	RBM
5	FM0131	Calculus	3	FMNS/1	YOP
6	FM0231	Physics	2	FMNS/1	STPL
7	UN0121	Religion	2	AM.1/1	Rev. RS
8	AM0933	Programming Design and Algorithms	3	AM.3/3	PA
9	AM1033	Elementary Linear Algebra	1	AM.3/3	RT
10	AM1133	Statistics	3	AM.3/3	YOP
11	AM1333	Multivariate Calculus	3	AM.3/3	TMS
12	UN0423	Pancasila	2	UN/3	SDS
13	UN0523	Multimedia	2	UN/3	JTH
14	AM2235	Real Analysis II	3	AM.5/5	DRS
15	AM2335	Combinatorics	3	AM.5/5	DGS
16	AM2435	Sampling Theory and Methods	1	AM.5/5	RS
17	AM2535	Decision Support System	3	AM.5/5	DJAS
18	AM2635	Mathematical Statistics	3	AM.5/5	RS
19	AM2725	Entrepreneurship Mathematics Science	2	AM.5/5	DGS
20	AM2835	Science Research Methodology	3	AM.5/5	DGS
21	AM4025	Queue Theory	3	AM.5/5	RS
22	AM3637	Applied Mathematics of Industry Based on Pro Deo Et Patria	3	AM.7/7	RS
23	AM3737	Stochastic Process	1	AM.7/7	CS
24	AM3837	KKN	3	AM.7/7	YOP
25	AM4627	Dynamic Program	2	AM.7/7	RS

2.2. Time periods, rooms, and institutional rules

The time structure is represented by 13 periods per day. The complete period definition is provided in Table 2. An institutional worship-time restriction is imposed every Tuesday from 09:10 to 10:00 AM. Since this interval overlaps parts of the second and third periods, the affected Tuesday morning periods are treated as unavailable for lectures in the optimization model, and the formulation enforces this restriction explicitly.

Table 2: Time Period Definition

Period	Start Time	Finish Time	Duration (min)	Period	Start Time	Finish Time	Duration (min)
1	08:00	08:50	50	8	13:50	14:40	50
2	08:50	09:40	50	9	14:40	15:30	50
3	09:40	10:30	50	10	15:30	16:20	50
4	10:30	11:20	50	11	16:20	17:10	50
5	11:20	12:10	50	12	17:10	18:00	50
6	12:10	13:00	50	13	18:00	18:50	50
7	13:00	13:50	50				

The available lecture rooms are summarized in Table 3. In this case study, RK 11 and RK 12 are treated as regular classrooms, while LAB 1 is the only facility room used for practicum or computer-based sessions.

Table 3: Available Lecture Rooms

No	Lecture Room
1	RK 11
2	RK 12
3	Laboratory

Based on Table 1, Table 2, and Table 3, the scheduling environment consists of three rooms operating from 08:00 to 18:50 over five lecture days (Monday–Friday). This setting naturally creates conflicts among cohorts, lecturers, and room usage if assignments are performed manually, motivating the use of an optimization-based approach such as Integer Linear Programming (ILP) to generate a feasible and conflict-free timetable.

2.3. Preference assumptions and computational procedure

Because the teaching system follows fixed cohorts, this study does not model individual student preferences. Student feasibility is enforced through cohort non-overlap constraints. Lecturer availability is incorporated through a binary acceptance matrix collected at the course level, which identifies acceptable and rejected time periods; rejected assignments are discouraged via a penalty-based objective. Institutional rules, including the Tuesday worship-time restriction and the absence of lecture activities on Saturday, are treated as hard constraints.

The ILP model is implemented in Python using the PuLP library as the modeling interface and solved using the CBC solver. The computational experiments produce an optimal timetable under the stated constraints and policies. The resulting schedule is then evaluated in terms of feasibility (no lecturer, cohort, or room clashes; consecutive periods for multi-credit courses; and room-type compatibility) and preference compliance through the reported objective value. The detailed results and their interpretation are presented in Section 3.

3. Results and Discussion

This section reports the outcomes of applying the proposed ILP formulation to the real scheduling data of the Mathematics Study Program. We begin by describing the practical scheduling context and the main sources of conflicts, including lecturer assignments, cohort-based non-overlap requirements, room limitations, and the institutional worship-time restriction. We then present how these elements are reflected in the preference matrix and the resulting optimized timetable, and we interpret the solution quality through feasibility checks and the objective value produced by the solver.

3.1. Lecture Scheduling Problems

Lecture scheduling in this study was carried out for the odd semester in the Mathematics Study Program, Faculty of Mathematics and Natural Sciences, HKBP Nommensen University,

Pematangsiantar. The main objective was to optimize the allocation of lecture time and classroom space in order to avoid conflicts related to lecturer availability, room usage, and class overlaps, while complying with institutional regulations such as designated worship times.

This study applies an Integer Linear Programming (ILP) approach, implemented in Python using the PuLP library as the modeling interface and the CBC solver for optimization. The scheduling model simultaneously considers courses, lecturers, rooms, lecture days, and time periods. Each lecture day is divided into 13 time periods with a fixed duration of 50 minutes. The model is designed to produce a feasible and implementable timetable by enforcing structural constraints (e.g., room and lecturer conflicts) and incorporating lecturer time-slot preferences as soft considerations.

The relationship between lecturers and the courses they teach is summarized in Table 4. Several lecturers are responsible for multiple courses, which increases problem complexity and highlights the need to prevent lecturer time conflicts across cohorts and semesters.

Table 4: Types of Courses and Instructors

No.	Lecturer	Course Label	No.	Lecturer	Course Label
1	FLOG	1	10	TMS	11
2	MS	2	11	SDS	12
3	HS	3	12	JTH	13
4	RBM	4	13	DRS	14
5	YOP	5, 10, 24	14	DGS	15, 19, 20
6	STPL	6	15	RS	16, 18, 21, 22, 25
7	Rev. RS	7	16	DJAS	17
8	PA	8	17	CS	23
9	RT	9			

In addition to lecturer–course assignments, the scheduling problem is structured by grouping courses according to class cohorts and semesters. Table 5 lists the study groups and their corresponding course labels. Each study group represents a cohort of students who must attend all listed courses without overlap, thereby imposing strict time-slot constraints.

Table 5: Study Groups and Course Labels

No.	Class (Class/Semester)	Course Label
1	AM.1/1 and FMNS/1	1, 2, 3, 4, 5, 6, 7
2	AM.3/3 and UN/3	8, 9, 10, 11, 12, 13
3	AM.5/5	14, 15, 16, 17, 18, 19, 20, 21
4	AM.7/7	22, 23, 24, 25

The information in Table 4 and Table 5 reflects the inherent combinatorial nature of the scheduling task: lecturers may teach multiple courses, and each cohort must attend a fixed set of courses without conflicts. These conditions justify the use of an optimization-based approach such as ILP to systematically generate a feasible, conflict-free timetable.

Lecturer time-slot preferences are represented by a binary acceptance matrix, where a value of 1 indicates that the corresponding time period is acceptable and a value of 0 indicates rejection. This acceptance matrix is summarized in Table 6 and is incorporated into the ILP objective function to discourage assignments in rejected time periods.

Table 6: Lecturer time-slot acceptance matrix for each course (1 = acceptable, 0 = rejected). Rows denote periods, columns denote course labels.

Time Session	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	1	1	1	0	1	1	0	0	1	1	1	1	0	1	1	0	1	0	1	0	1	0	0	0	0
2	1	1	1	0	1	1	0	0	1	1	1	1	0	1	1	0	1	0	1	0	1	0	0	0	0
3	1	1	1	0	1	1	0	0	1	1	1	1	0	1	1	1	1	0	1	0	1	0	0	0	0
4	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	0	1	1	1	0	0	0	0
5	1	1	1	1	1	1	0	1	1	0	1	1	0	0	1	1	1	0	1	1	1	0	0	0	0
6	1	1	1	1	1	1	0	1	1	0	1	1	0	0	1	1	1	0	1	1	1	0	0	0	1
7	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	0	1
8	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1
9	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
10	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	1
11	1	1	1	0	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	0	0	1	1	1	1
12	1	1	1	0	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	0	0	1	1	1	1
13	1	1	1	0	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	0	0	1	1	1	1

To improve readability, the lecturer time-slot acceptance matrix in Table 6 is also visualized in Fig. 1. This visualization highlights rejected periods (0) and acceptable periods (1) across all courses, making the overall preference structure easier to interpret at a glance.

Fig. 1: Heatmap visualization of the lecturer time-slot acceptance matrix Q_{tk} (1 = acceptable, 0 = rejected). Rows represent time periods ($t = 1, \ldots, 13$) and columns represent course labels ($k = 1, \ldots, 25$).

To reduce model complexity and clearly define the scope of the study, several assumptions are adopted. Lecturer assignments for each course are fixed and predetermined. Students are assumed to have no individual time preferences, so cohort-based non-overlap constraints are used as the primary student-related restriction. All available classrooms may be used subject to room availability constraints. Courses with three credit units are assumed to be delivered in three consecutive periods, and courses that require laboratory or computer-based facilities are scheduled exclusively in LAB 1. In addition, no lectures are permitted during the designated worship time held every Tuesday from 09:10 to 10:00 AM, and no lecture activities are scheduled on Saturdays.

3.2. Design of Integer Linear Programming Model

The lecture scheduling problem in this study is formulated as an Integer Linear Programming (ILP) model by explicitly considering limitations related to space, time, and lecturer availability [16, 18]. The model aims to generate an efficient and conflict-free lecture timetable based on predefined time periods, available lecture rooms, and lecturer time-slot preferences (acceptance/rejection).

Through this formulation, institutional constraints and lecturer preferences can be integrated into a single optimization framework.

Indices and sets. Let $\mathcal{K} = \{1, 2, \ldots, K\}$ denote the set of courses (in this study, $K = 25$), $\mathcal{R}$ the set of rooms, $\mathcal{D}$ the set of lecture days, and $\mathcal{T} = \{1, 2, \ldots, T\}$ the set of time periods per day (here, $T = 13$). We use:

$$k \in \mathcal{K} \text{ (course)}, \quad r \in \mathcal{R} \text{ (room)}, \quad d \in \mathcal{D} \text{ (day)}, \quad t \in \mathcal{T} \text{ (time period)}.$$

In this case study, $\mathcal{R} = \{\text{RK } 11, \text{RK } 12, \text{LAB } 1\}$. The day index can be defined as $\mathcal{D} = \{1, 2, 3, 4, 5\}$ (Monday–Friday); if Saturday is included for bookkeeping, it must be constrained as non-lecture day in the constraint set.

To distinguish room categories and course types, define the room subsets

$$\mathcal{R}_p = \{\text{LAB } 1\} \quad \text{(facility room)}, \qquad \mathcal{R}_t = \{\text{RK } 11, \text{RK } 12\} \quad \text{(regular rooms)}.$$

Let $G_p \subseteq \mathcal{K}$ be the set of courses that require laboratory or computer-based facilities (e.g., programming/multimedia sessions) and therefore must be scheduled in the laboratory (LAB 1). Let $G_t = \mathcal{K} \setminus G_p$ be the set of courses that can be scheduled in regular classrooms. In this case study, LAB 1 is treated as the only facility room, hence any course categorized in G_p is restricted to LAB 1 by the room-type compatibility constraints.

To model cohort non-overlap constraints, define the cohort sets (Table 5):

$$G_{S1} = \{1, 2, 3, 4, 5, 6, 7\}, \quad G_{S3} = \{8, 9, 10, 11, 12, 13\},$$

$$G_{S5} = \{14, 15, 16, 17, 18, 19, 20, 21\}, \quad G_{S7} = \{22, 23, 24, 25\}.$$

These sets ensure that, within each cohort, no two courses are scheduled at the same day and time period. If lecturer-based non-overlap constraints are used, it is recommended to index lecturers by $\ell \in \mathcal{L}$ and define lecturer course sets $G_{L\ell} \subseteq \mathcal{K}$, where each $G_{L\ell}$ contains the course labels taught by lecturer ℓ.

Parameters. Let S_k denote the number of face-to-face periods required for course k (derived from its credit load). Lecturer time-slot preference is given by the binary acceptance matrix Q_{tk}:

$$Q_{tk} = \begin{cases} 1, & \text{if time period } t \text{ is acceptable for course } k, \\ 0, & \text{if time period } t \text{ is rejected for course } k. \end{cases}$$

To penalize rejected assignments in a minimization framework, define the penalty parameter

$$P_{tk} = 1 - Q_{tk},$$

so that $P_{tk} = 1$ indicates a rejected time period and $P_{tk} = 0$ indicates an acceptable one.

Decision variables. A binary decision variable is introduced:

$$x_{rdtk} = \begin{cases} 1, & \text{if course } k \text{ is scheduled in room } r \text{ on day } d \text{ at period } t, \\ 0, & \text{otherwise.} \end{cases}$$

The variable x_{rdtk} captures the time-room occupancy, while y_{rdk} supports consistency constraints between room assignment and time allocation.

Objective function. The objective is to minimize the total time-slot rejection penalty implied by lecturer preferences. Using $P_{tk} = 1 - Q_{tk}$, the objective function is:

$$\min Z = \sum_{r\in\mathcal{R}} \sum_{d\in\mathcal{D}} \sum_{t\in\mathcal{T}} \sum_{k\in\mathcal{K}} x_{rdtk}\, P_{tk}.$$

A smaller value of Z indicates that fewer course-period assignments fall into rejected time slots, hence the resulting timetable is more consistent with lecturer preferences while still satisfying all hard constraints related to room availability, lecturer/cohort conflicts, worship time, and non-lecture days.

3.3. Constraints

The scheduling model incorporates a set of constraints that represent academic regulations, spatial limitations, lecturer availability, cohort coherence, and institutional policies such as worship periods and non-lecture days. These constraints ensure that the resulting timetable is feasible, conflict-free, and compliant with all predefined requirements.

Auxiliary start-time variable (to enforce consecutiveness). To model consecutive time periods for multi-credit courses in a linear ILP form, we introduce an auxiliary binary variable

$$s_{rdtk} = \begin{cases} 1, & \text{if course } k \textit{ starts} \text{ in room } r \text{ on day } d \text{ at period } t, \\ 0, & \text{otherwise.} \end{cases}$$

The original binary variable x_{rdtk} remains an *occupancy* variable (1 if course k occupies room r on day d at period t). The variables are linked as follows.

(C1) Each course starts exactly once per week.

$$\sum_{r\in\mathcal{R}} \sum_{d\in\mathcal{D}} \sum_{t\in\mathcal{T}} s_{rdtk} = 1, \quad \forall k \in \mathcal{K}.$$

(C2) Feasible start times. A course of length S_k cannot start after period $T - S_k + 1$.

$$s_{rdtk} = 0, \quad \forall r \in \mathcal{R},\, d \in \mathcal{D},\, k \in \mathcal{K},\, t > T - S_k + 1.$$

(C3) Linking start-time and occupancy (consecutive periods). For each course k, if it starts at period τ, then it occupies periods $\tau, \tau+1, \ldots, \tau+S_k-1$ on the same day and room. This is enforced by:

$$x_{rdtk} = \sum_{\tau=\max(1,\, t-S_k+1)}^{\min(t,\, T-S_k+1)} s_{rd\tau k}, \quad \forall r \in \mathcal{R},\, d \in \mathcal{D},\, t \in \mathcal{T},\, k \in \mathcal{K}.$$

With this definition, the consecutive requirement is satisfied automatically, and each scheduled course occupies exactly S_k periods.

(C4) Room capacity (no room clashes). Each room can host at most one course in the same day and time period:

$$\sum_{k\in\mathcal{K}} x_{rdtk} \le 1, \quad \forall r \in \mathcal{R},\, d \in \mathcal{D},\, t \in \mathcal{T}.$$

(C5) Lecturer non-overlap (no lecturer clashes). Let $\mathcal{L}$ be the set of lecturers and $G_{L\ell} \subseteq \mathcal{K}$ the set of courses taught by lecturer ℓ. Then:

$$\sum_{r\in\mathcal{R}} \sum_{k\in G_{L\ell}} x_{rdtk} \le 1, \quad \forall \ell \in \mathcal{L},\, d \in \mathcal{D},\, t \in \mathcal{T}.$$

(C6) Cohort non-overlap (no student group clashes). Using the cohort sets defined in the model (Table 5):

$$\sum_{r\in\mathcal{R}}\sum_{k\in G_{S1}} x_{rdtk} \le 1, \quad \forall d \in \mathcal{D},\, t \in \mathcal{T},$$

$$\sum_{r\in\mathcal{R}}\sum_{k\in G_{S3}} x_{rdtk} \le 1, \quad \forall d \in \mathcal{D},\, t \in \mathcal{T},$$

$$\sum_{r\in\mathcal{R}}\sum_{k\in G_{S5}} x_{rdtk} \le 1, \quad \forall d \in \mathcal{D},\, t \in \mathcal{T},$$

$$\sum_{r\in\mathcal{R}}\sum_{k\in G_{S7}} x_{rdtk} \le 1, \quad \forall d \in \mathcal{D},\, t \in \mathcal{T}.$$

(C7) Room-type compatibility (theory vs practicum). Let $\mathcal{R}_p = \{\text{Lab}\}$ be practicum rooms and $\mathcal{R}_t = \{\text{RK } 11, \text{RK } 12\}$ be regular rooms. Then:

$$x_{rdtk} = 0, \quad \forall k \in G_t,\, r \in \mathcal{R}_p,\, d \in \mathcal{D},\, t \in \mathcal{T},$$

$$x_{rdtk} = 0, \quad \forall k \in G_p,\, r \in \mathcal{R}_t,\, d \in \mathcal{D},\, t \in \mathcal{T}.$$

(C8) Worship time restriction (Tuesday 09:10–10:00). As the worship time overlaps parts of periods 2 and 3 on Tuesday, those periods are treated as unavailable on day $d = 2$:

$$x_{r,2,t,k} = 0, \quad \forall r \in \mathcal{R},\, k \in \mathcal{K},\, t \in \{2, 3\}.$$

This also prevents any multi-period course from starting in a way that would occupy the restricted periods, due to constraint (C3).

(C9) Binary requirements.

$$x_{rdtk} \in \{0, 1\}, \quad s_{rdtk} \in \{0, 1\}, \quad \forall r \in \mathcal{R},\, d \in \mathcal{D},\, t \in \mathcal{T},\, k \in \mathcal{K}.$$

3.4. Computational Results and Discussion

This subsection reports the computational results of the proposed ILP model and discusses the feasibility, preference satisfaction, and practical interpretability of the obtained timetable. The problem instance consists of 25 courses scheduled over five lecture days (Monday–Friday), with 13 time periods per day and three rooms (RK 11, RK 12, and Laboratory/LAB 1). The model minimizes violations of lecturers' time-slot preferences represented by the binary acceptance matrix Q_{tk} (Table 6) by using the penalty coefficient $P_{tk} = 1 - Q_{tk}$.

To facilitate reproducibility and interpretation, the results are presented in a stepwise manner. First, we report the solver configuration and the resulting solution status to confirm that an optimal integer solution was reached. Next, we present the finalized timetable in a compact tabular format, followed by a graphical visualization to support quick verification of room usage patterns. Finally, we discuss the feasibility of the obtained schedule with respect to each hard constraint and interpret the objective value in terms of lecturers' time-slot preference satisfaction.

3.4.1. Solver setup and solution status

The ILP model was implemented in Python using PuLP as the modeling interface and solved using the CBC (Coin-or Branch and Cut) solver. CBC returned an *Optimal* status, indicating that a feasible integer solution satisfying all hard constraints was found and that the objective value reached its minimum under the proposed formulation. The computational experiments were conducted on a computer equipped with an Intel Core i5 processor and 8 GB RAM. The reported runtime was approximately 1 minute and 34 seconds, as summarized in Table 7.

Table 7: Solver summary (PuLP/CBC).

Item	Value
Solver	CBC (via PuLP)
Solution status	Optimal
Objective value Z^*	0
Runtime (reported)	$\approx$ 1 minute 34 seconds
Binary variables (upper bound)	$\lvert x\rvert = 3 \times 5 \times 13 \times 25 = 4875$, $\lvert s\rvert = 4875$

3.4.2. Final optimized timetable

The final optimized timetable produced by the proposed ILP model is presented in Table 8. The table reports, for each course, the assigned room, lecture day, starting period, duration (number of consecutive periods), the occupied period indices, and the corresponding lecturer. The day index follows the convention 1=Monday, 2=Tuesday, 3=Wednesday, 4=Thursday, and 5=Friday. This presentation is intended to support direct operational use by the study program, while also allowing verification of the hard constraints such as room non-overlap, lecturer non-overlap, cohort non-overlap, consecutive periods for multi-credit courses, and room-type compatibility (theory vs practicum).

Table 8: Optimized timetable satisfying hard constraints and minimizing lecturer time-slot rejection (Day: 1=Mon, 2=Tue, 3=Wed, 4=Thu, 5=Fri).

Subject	Room	Day	Start	Duration	Period	Lecturer
Basic Calculus Response	LAB 1	3	7	1	7	FLOG
Introduction to Logic and Sets	RK 12	2	11	3	11, 12, 13	MS
Number Theory	RK 12	3	10	3	10, 11, 12	HS
Analytical Geometry	RK 12	4	4	3	4, 5, 6	RBM
Calculus	RK 12	5	1	3	1, 2, 3	YOP
Physics	RK 12	5	7	2	7, 8	STPL
Religion	RK 12	5	10	2	10, 11	Rev. RS
Programming Design and Algorithms	LAB 1	1	4	3	4, 5, 6	PA
Elementary Linear Algebra	RK 11	1	13	1	13	RT
Statistics	RK 12	4	10	3	10, 11, 12	YOP
Multivariate Calculus	RK 11	5	5	3	5, 6, 7	TMS
Pancasila	RK 12	3	5	2	5, 6	SDS
Multimedia	LAB 1	2	9	2	9, 10	JTH
Real Analysis II	RK 11	3	8	3	8, 9, 10	DRS
Combinatorics	RK 11	2	4	3	4, 5, 6	DGS
Sampling Theory and Methods	RK 11	1	10	1	10	RS
Decision Support System	RK 11	2	8	3	8, 9, 10	DJAS
Mathematical Statistics	RK 11	5	10	3	10, 11, 12	RS
Entrepreneurship Mathematics Science	LAB 1	4	5	2	5, 6	DGS
Science Research Methodology	RK 11	3	5	3	5, 6, 7	DGS
Queue Theory	RK 12	4	1	3	1, 2, 3	RS
Applied Mathematics of Industry Based on Pro Deo Et Patria	RK 11	2	11	3	11, 12, 13	RS
Stochastic Process	RK 12	1	10	1	10	CS
KKN	RK 11	3	11	3	11, 12, 13	YOP
Dynamic Program	RK 11	1	6	2	6, 7	RS

The feasibility of the timetable in Table 8 is verified in the next subsection by checking each hard constraint explicitly.

3.4.3. Feasibility with respect to hard constraints

A direct inspection of Table 8 confirms that all hard constraints are satisfied. First, no clashes occur in room usage: each room hosts at most one course in the same day–period. Second, lecturer non-overlap and cohort non-overlap constraints prevent simultaneous assignments for the

same lecturer and for the same student cohort. Third, multi-credit courses occupy consecutive periods according to their required duration. Fourth, room-type compatibility is enforced: facility-required courses (G_p) are assigned exclusively to the laboratory (LAB 1), while theory-oriented courses are scheduled in regular rooms (RK 11 and RK 12). Finally, the institutional worship-time restriction on Tuesday is respected by treating periods 2 and 3 as unavailable; therefore, no course is scheduled in those periods on Tuesday.

3.4.4. Preference satisfaction and objective interpretation

Lecturers' time-slot preferences are encoded by the acceptance matrix Q_{tk} (Table 6), where $Q_{tk} = 1$ indicates an acceptable period and $Q_{tk} = 0$ indicates rejection. The optimization penalizes rejected assignments using $P_{tk} = 1 - Q_{tk}$ and minimizes

$$Z = \sum_{r \in \mathcal{R}} \sum_{d \in \mathcal{D}} \sum_{t \in \mathcal{T}} \sum_{k \in \mathcal{K}} x_{rdtk} \, P_{tk}.$$

Since $P_{tk} \in \{0, 1\}$, the objective value Z counts the number of occupied course–period assignments that fall into rejected time slots. The obtained solution yields $Z^* = 0$, which implies that *every scheduled course period is assigned only to acceptable time slots* (i.e., all occupied (t, k) pairs satisfy $Q_{tk} = 1$).

For an immediate visual validation, Fig. 2 overlays the optimized schedule (Table 8) on the acceptance matrix. In this overlay, outlined cells represent scheduled course–period selections. In the reported solution, all outlined cells fall on acceptable entries ($Q_{tk} = 1$), consistent with $Z^* = 0$.

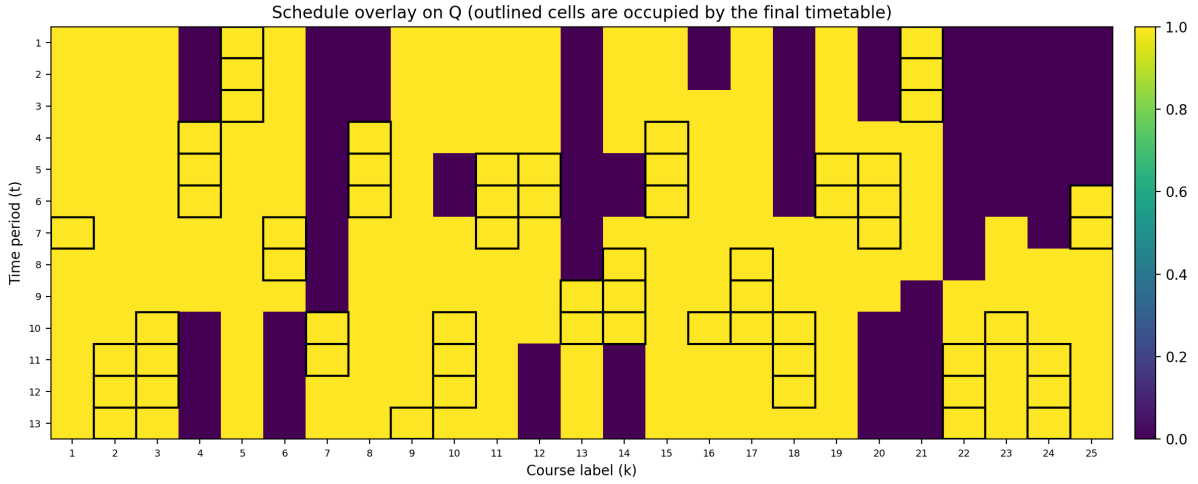

Fig. 2: Overlay of the optimized schedule on the lecturer time-slot acceptance matrix Q. Cells indicate lecturer acceptance ($Q_{tk} = 1$ acceptable, $Q_{tk} = 0$ rejected), while the outlined cells mark the time periods selected by the optimized timetable (Table 8).

3.4.5. Timetable visualization and room utilization

While Table 8 provides the detailed allocation, Fig. 3 summarizes the same solution in a room-by-day timetable layout, making it easier to verify room utilization patterns and confirm the absence of conflicts.

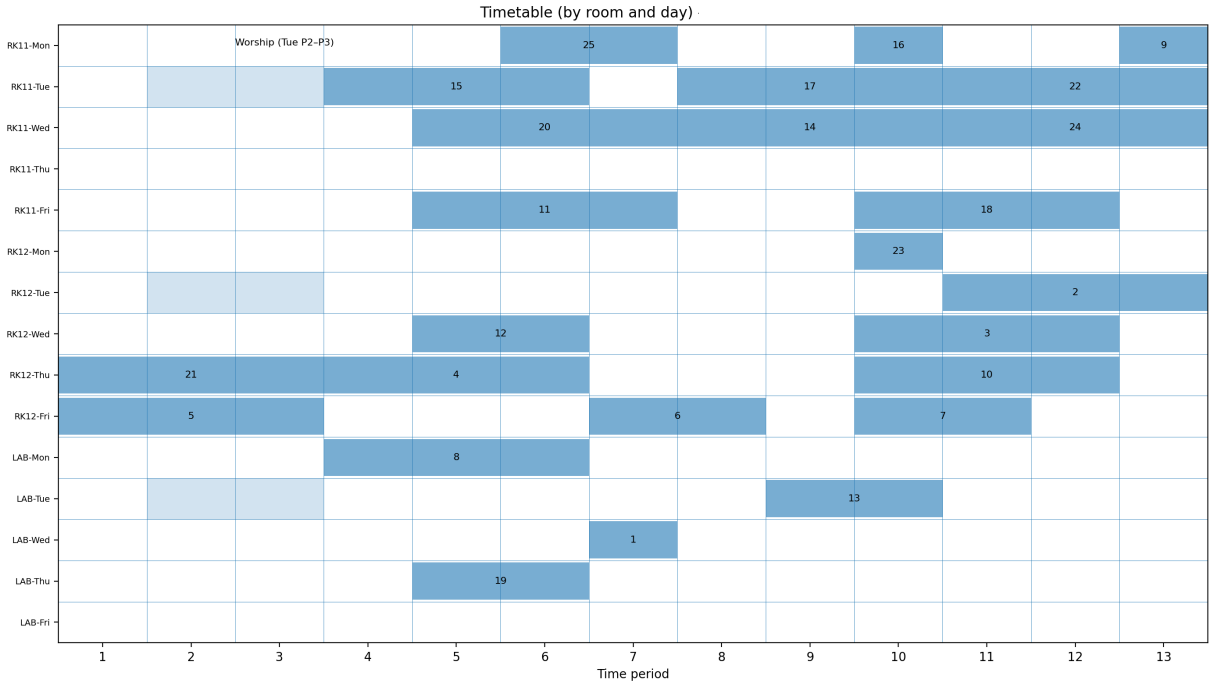

Fig. 3: Graphical timetable of the optimized schedule by room (RK 11, RK 12, and LAB 1) and day (Monday–Friday). Each block represents a scheduled course spanning its required consecutive periods.

Across three rooms, five days, and thirteen periods per day, the weekly capacity is $3 \times 5 \times 13 = 195$ period-slots. From Table 8, the timetable uses 61 period-slots, corresponding to approximately $61/195 \approx 31.3\%$ utilization. This utilization level is reasonable because the instance covers a single study program with a limited weekly teaching load relative to the available room-time capacity. Overall, the results show that the proposed ILP model can generate a feasible and operational timetable, while simultaneously achieving full satisfaction of the lecturers' binary time-slot preferences under the stated constraints.

4. Conclusion

This study proposed an Integer Linear Programming (ILP) model for lecture timetabling in the Mathematics Study Program, HKBP Nommensen University. The model integrates key constraints, including room capacity, lecturer and cohort non-overlap, consecutive periods for multi-credit courses, practicum/theory room compatibility, and institutional rules such as the Tuesday worship-time restriction.

The model was implemented in Python using PuLP and solved with the CBC solver. The solver returned an *Optimal* status, and the resulting timetable schedules all 25 courses across RK 11, RK 12, and LAB 1 without conflicts. Practicum courses are assigned to the laboratory and theory courses to regular rooms, while all hard constraints are satisfied and lecturer time-slot rejections are minimized.

Future work may extend the model by adding class-size and room-capacity constraints, using weighted penalties for soft constraints, and combining ILP with heuristic methods to improve scalability for larger timetabling instances.

CRediT Authorship Contribution Statement

Gayus Simarmata: Conceptualization, Methodology, Formal Analysis, Writing–Original Draft. **Rajainal Saragih**: Software, Data Curation, Validation, Writing–Review & Editing, Project Administration. **Anil Hakim Syofra**: Validation, Investigation, Writing–Review & Editing.

Declaration of Generative AI and AI-assisted Technologies

Generative AI tools were used during the preparation of this manuscript to assist with language refinement, structural editing, and clarity improvement of the text. Specifically, ChatGPT (OpenAI) was used as a writing support tool under the full supervision and critical review of the authors. All scientific content, mathematical formulations, data interpretation, and conclusions remain the sole responsibility of the authors.

Declaration of Competing Interest

The authors declare no competing interests.

Funding and Acknowledgments

This research received no external funding. The authors would like to acknowledge the Mathematics Study Program, Faculty of Mathematics and Natural Sciences, HKBP Nommensen University, Pematangsiantar, for providing academic data and institutional support that enabled this research to be conducted.

Data and Code Availability

The data and code supporting the findings of this study are available from the corresponding author upon reasonable request. The scheduling data are derived from internal academic records of the Mathematics Study Program, HKBP Nommensen University, and are subject to institutional confidentiality policies.

References

[1] S. Ibrahim, M. H. Abdullah, and N. A. Rahman. "A General Mathematical Model for University Courses Timetabling: Implementation to a Public University in Malaysia". In: *Malaysian Journal of Fundamental and Applied Sciences* 18.1 (2022), pp. 1–10. DOI: 10.11113/mjfas.v18n1.2408.

[2] H. Babaei, J. Karimpour, and A. Hadidi. "A Survey of Approaches for University Course Timetabling Problem". In: *Computers & Industrial Engineering* 86 (2015), pp. 43–59. DOI: 10.1016/j.cie.2014.11.010.

[3] H. Rudová, T. Müller, and K. Murray. "Complex University Course Timetabling". In: *Journal of Scheduling* 14.2 (2011), pp. 187–207. DOI: 10.1007/s10951-010-0171-3.

[4] M. H. Cruz-Rosales et al. "Metaheuristic with Cooperative Processes for the University Course Timetabling Problem". In: *Applied Sciences* 12 (2022), p. 542. DOI: 10.3390/app12020542.

[5] W. A. U. D. Perera and G. H. J. Lanel. "A Model to Optimize University Course Timetable Using Graph Coloring and Integer Linear Programming". In: *IOSR Journal of Mathematics* 12.5 (2016), pp. 13–18. DOI: 10.9790/5728-1205031318.

[6] G. A. Guzman, C. Martínez, and J. Pacheco. "An Integer Linear Programming Model for a University Timetabling Problem Considering Time Windows and Consecutive Periods". In: *Journal of Applied Operational Research* 6.3 (2015), pp. 159–167. DOI: 10.48287/2310-5070.2023.121.

[7] S. Daskalaki, T. Birbas, and E. Housos. "An Integer Programming Formulation for a Case Study in University Timetabling". In: *European Journal of Operational Research* 153.1 (2004), pp. 117–135. DOI: 10.1016/S0377-2217(03)00103-6.

[8] R. Lewis. "A Survey of Metaheuristic-Based Techniques for University Timetabling Problems". In: *OR Spectrum* 30 (2008), pp. 167–190. DOI: 10.1007/s00291-007-0097-0.

[9] Z. Lü and J. K. Hao. "Adaptive Tabu Search for Course Timetabling". In: *European Journal of Operational Research* 200.1 (2010), pp. 235–244. DOI: 10.1016/j.ejor.2008.12.007.

[10] X. Feng, Y. Lee, and I. Moon. "An Integer Program and a Hybrid Genetic Algorithm for the University Timetabling Problem". In: *Optimization Methods & Software* 32 (2017), pp. 625–649. DOI: 10.1080/10556788.2016.1233970.

[11] E. H. Kampke et al. "A Network Flow Based Construction for a GRASP-SA Algorithm to Solve the University Timetabling Problem". In: *Computational Science and Its Applications – ICCSA 2019*. Springer, 2019, pp. 215–231. DOI: 10.1007/978-3-030-24302-9_16.

[12] C. W. Fong, H. Asmuni, and B. McCollum. "A Hybrid Swarm-Based Approach to University Timetabling". In: *IEEE Transactions on Evolutionary Computation* 19 (2015), pp. 870–884. DOI: 10.1109/TEVC.2015.2411741.

[13] S. I. Hossain et al. "Optimization of University Course Scheduling Problem Using Particle Swarm Optimization with Selective Search". In: *Expert Systems with Applications* 127 (2019), pp. 9–24. DOI: 10.1016/j.eswa.2019.02.026.

[14] F. Al-Shihri and A. Al-Thoheir. "A Mixed Integer Programming Model for University Course Scheduling". In: *Journal of King Saud University – Computer and Information Sciences* 19.2 (2007), pp. 53–62. DOI: 10.1016/j.jksuci.2007.02.002.

[15] E. Rappos, R. Barták, and E. K. Burke. "A Mixed-Integer Programming Approach for Solving University Course Timetabling Problems". In: *Journal of Scheduling* 25 (2022), pp. 391–404. DOI: 10.1007/s10951-021-00715-5.

[16] A. E. Phillips, M. Ehrgott, and D. M. Ryan. "Integer Programming Methods for Large-Scale Practical Classroom Assignment Problems". In: *Computers & Operations Research* (2014). DOI: 10.1016/j.cor.2014.07.012.

[17] P. Demeester and K. Goossens. "The Examination Timetabling Problem: A New Integer Programming Approach". In: *Computers & Operations Research* 36.6 (2009), pp. 1845–1856. DOI: 10.1016/j.cor.2008.06.013.

[18] B. McCollum. "A Perspective on Bridging the Gap between Theory and Practice in University Timetabling". In: *Practice and Theory of Automated Timetabling VI*. Vol. 3867. Lecture Notes in Computer Science. Springer, 2007, pp. 3–23. DOI: 10.1007/978-3-540-77345-0_1.