computation

MDPI

Review

From Integer Programming to Machine Learning: A Technical Review on Solving University Timetabling Problems

Xin Gu [1,*,†], Muralee Krish [2,†], Shaleeza Sohail [3,‡], Sweta Thakur [2,‡], Fariza Sabrina [4,‡] and Zongwen Fan [5,‡]

[1] Lincoln Institute of Higher Education, Sydney, NSW 2000, Australia
[2] School of Information Technology, King's Own Institute, Sydney, NSW 2000, Australia; muralee.krish@koi.edu.au (M.K.); sweta.thakur@koi.edu.au (S.T.)
[3] College of Engineering, Science and Environment, The University of Newcastle, Callaghan, NSW 2308, Australia; shaleeza.sohail@newcastle.edu.au
[4] School of Engineering and Technology, Central Queensland University, Rockhampton, QLD 4701, Australia; f.sabrina@cqu.edu.au
[5] College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China; zongwen.fan@hqu.edu.cn
* Correspondence: xin.gu@lincolnau.nsw.edu.au
† These authors contributed equally to this work.
‡ These authors contributed equally to this work.

Abstract: Solving the university timetabling problem is crucial as it ensures efficient use of resources, minimises scheduling conflicts, and enhances overall productivity. This paper presents a comprehensive review of university timetabling problems using integer programming algorithms. This study explores various integer programming techniques and their effectiveness in optimising complex scheduling requirements in higher education institutions. We analysed 95 integer programming-based models developed for solving university timetabling problems, covering relevant research from 1990 to 2023. The goal is to provide insights into the evolution of these algorithms and their impact on improving university scheduling. We identify that the implementation rate of models using integer programming is 98%, which is much higher than 34% implementation rates using meta-heuristics algorithms from the existing review. The integer programming models are analysed by the problem types, solutions, tools, and datasets. For three types of timetabling problems including course timetabling, class timetabling, and exam timetabling, we dive deeper into the commercial solvers CPLEX (47), Gurobi (11), Lingo (5), Open Solver (4), C++ GLPK (4), AIMMS (2), GAMS (2), XPRESS (2), CELCAT (1), AMPL (1), and Google OR-Tools CP-SAT (1) and identify that CPLEX is the most frequently used integer programming solver. We explored the uses of machine learning algorithms and the hybrid solutions of combining the integer programming and machine learning algorithms in higher education timetabling solutions. We also identify areas for future work, which includes an emphasis on using integer programming algorithms in other industrial areas, and using machine learning models for university timetabling to allow data-driven solutions.

Keywords: integer programming; university timetabling; soft constraints and hard constraints; CPLEX

check for updates

Academic Editors: Demos T. Tsahalis and Francesco Cauteruccio

Received: 18 October 2024
Revised: 15 December 2024
Accepted: 17 December 2024
Published: 3 January 2025

Citation: Gu, X.; Krish, M.; Sohail, S.; Thakur, S.; Sabrina, F.; Fan, Z. From Integer Programming to Machine Learning: A Technical Review on Solving University Timetabling Problems. *Computation* **2025**, *13*, 10. https://doi.org/10.3390/computation13010010

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The task of university timetabling involves the complicated process of assigning courses to time slots and classrooms while adhering to a number of constraints, including faculty availability, room capacities, and student preferences [1]. In higher education

institutions, this endeavour presents a complex challenge due to the multifaceted relationship of various constraints and requirements. Efficiently scheduling courses, classrooms, and resources while accommodating diverse student preferences and faculty availability necessitates solutions of a sophisticated nature. As academic institutions expand their offerings and struggle with increasing scheduling complexities, the demand for systematic and optimised timetabling solutions has become a necessity.

Traditionally, manual timetabling approaches have been employed to tackle these challenges. However, the modern scale and intricacy of academic institutions have rendered such methods inadequate. In response to this growing gap, the integration of computational methods, notably integer programming, has emerged as a central focus of both research and practice. Integer programming offers a promising mathematical paradigm for addressing the intricate scheduling dilemmas inherent in university timetabling. Over the years, integer programming has established itself as a powerful mathematical approach for resolving timetabling problems by providing near-optimal or optimal solutions [2,3].

This paper presents a comprehensive review of the role and application of integer programming techniques within the context of university timetabling. By synthesising the existing literature and research findings, our work includes the following:

1. Identifying Timetabling Problems: We categorise and define the array of timetabling challenges within higher education that have been effectively addressed through the utilisation of integer programming techniques. This spans diverse scenarios such as course timetabling, classroom timetabling, lecturer timetabling, student timetabling, examination timetabling and so on.
2. Examining Methodologies: We explore the methodologies and strategies employed to formulate university timetabling problems as integer programming models. Our focus is on how these models encapsulate the complicated relationship of constraints and objectives unique to academic scheduling contexts.
3. Evaluating Solver Tools: We undertake a thorough review of the software tools, libraries, and frameworks employed to implement integer programming solutions for university timetabling problems.
4. Analysing Dataset Characteristics: We investigate the datasets commonly utilised in university timetabling studies that leverage integer programming.
5. Investigating Machine Learning Algorithms: We investigate the uses of machine learning algorithms in timetabling solutions.
6. Exploring Hybrid Solutions: We explore the possibilities to combine integer programming and machine learning algorithms together to solve timetabling problems.

In summary, this paper seeks to highlight the role of integer programming in developing the approach to university timetabling challenges. By bringing together the existing knowledge and exploring the details of problem formulation, solution methodologies, solver tools, and dataset characteristics, we aim to provide a comprehensive overview that stimulates further innovations and advancements in the realm of academic scheduling.

The research questions underpinning our exploration are as follows:

- RQ. 1: What are the existing timetabling problems in higher education that have been effectively addressed through integer programming?
- RQ. 2: How is integer programming employed to solve these timetabling problems?
- RQ. 3: What tools, including software libraries and frameworks, are utilised in tackling these timetabling problems using integer programming?
- RQ. 4: What are the defining characteristics of the datasets employed in testing and analysing these timetabling problems?
- RQ. 5: What are the machine learning algorithms used in timetabling problems if there is any?

- RQ. 6: Is it possible to combine integer programming and machine learning algorithms together to solve timetabling problems?

By conducting a comprehensive investigation into these research questions, this paper aims to contribute to a deeper understanding of the role of integer programming in addressing timetabling challenges within higher education. Our exploration of the problems addressed, methodologies employed, solver tools used, and dataset characteristics will culminate in a comprehensive overview of the current state of the field. Through this, we aspire to lay the foundation for further advancements in this critical domain of study.

The contributions of this paper are listed below:

- This paper presents a comprehensive review of the existing literature, highlighting key insights on integer programming-based solutions for university timetabling problems.
- This paper provides a detailed analysis of the effectiveness of existing approaches in addressing complex scheduling challenges.
- This paper provides in-depth insights and a thorough analysis that highlights the effectiveness of integer programming compared to heuristic and meta-heuristic approaches for solving university timetabling problems.
- This paper provides the analysis of using machine learning in higher education timetabling solutions.

The outline of the rest of this paper is as follows: Section 2 discusses the existing review papers, Section 3 presents the methodology used in this paper, Section 4 discusses the findings, and Section 5 concludes the paper and presents future work.

2. Existing Reviews

In the past two decades, a massive amount of publications have been generated to tackle the university timetabling problems. Consequently, quite a few reviews and surveys have been published to synthesise and contextualise the work already performed in this space. A brief overview of these reviews and surveys are presented in chronological order in this section.

Qu et al. [4] surveyed search methodologies and automated approaches for examination timetabling. Through 65 publications, the identified approaches were graph heuristics, constraint-based techniques, local search-based techniques (including tabu search, simulated annealing, and other local search-based techniques), population-based algorithms (including evolutionary algorithms, memetic algorithms, ant algorithms), multi-criteria techniques, and hyper-heuristics. However, integer programming [5] was only mentioned once as one of the techniques used for timetabling systems.

Pillay [6] conducted a review on the hyper-heuristics approach for university examination timetabling, university course timetabling, and school timetabling. This work primarily focuses on hyper-heuristics as a general solution approach for educational timetabling problems, including university examination, course, and school timetabling. On the other hand, our work explores the effectiveness of usage of integer programming techniques for solving university timetabling problems.

Aziz and Aizam [7] conducted a brief review on the features of university course timetabling problems over 39 studies. They identified the complications in timetabling mainly due to the variation of constraints, and further discussed five types of constraints: completeness, conflict of resources, workload, availability of resources, and meeting patterns.

Stephen [8] summarised the iterative improvements methods used in approaching the optimisation problem for university timetabling. The methods included simulated annealing and the great deluge algorithm, tabu search, evolutionary algorithms, mimetic al-

gorithms, ant colony optimisation, and hybrid approaches. However, integer programming was not mentioned in this review.

Hosny [9] reviewed meta-heuristic approaches for handling university timetabling problems and analysed 11 case studies from Middle Eastern Universities. The review focused on the hard constraints and soft constraints on four types of timetabling including examination, course, staff, and project. Meta-heuristic approaches rely on the quality of the initial solution for creating an effective final solution and those solutions are not explainable.

Bashab et al. [10] investigated into one specific algorithm: meta-heuristic algorithms for handling university timetabling problems. They adopted a systematic mapping methodology to study 131 publications that were using meta-heuristics algorithms. In total, 66% of these publications were only solution proposals and 34% were published with implementation.

Alghamdi and Alsubait [1] conducted a review over 16 relevant publications from the aspects of techniques, tools, advantages, and disadvantages. Two of these publications proposed integer programming solutions using various solver packages such as BARON. The review concluded that this problem has been approached through heuristic techniques and meta-heuristics.

Chen et al. [11] conducted a survey of university course timetabling problems over eleven universities. Various solution methods were identified including SA, TS, clustering and colour mapping, GA, IP, clustering, VND, HH (TS + VNS), HGA, MIP, and Hybrid (VNS + TS). Fifteen constraints including hard constraints and soft constraints were summarised from these eleven universities.

Tan et al. [12] conducted a survey on the optimisation methodologies in school timetabling problems. In this work, all algorithms were classified into six types: mathematical optimisation algorithms, meta-heuristic algorithms, graph colouring algorithms, meta-heuristics approaches, hyper-heuristic approaches, and hybrid approaches. Integer programming was categorised under mathematical optimisation algorithms in this survey. In this survey, six integer programming solutions from 2014 to 2020 were reviewed.

Zhu, Li, and Li [13] conducted a survey on computational intelligence in educational timetabling. In this survey, the computational intelligent approaches were classified into heuristic approaches, hybrid approaches, fuzzy logic approaches, and distributed multi-agent systems. Integer programming was adopted in resolving course timetabling as the heuristic approach. As a fuzzy logic approach, a fuzzy integer linear programming was employed to tackle an examination timetabling problem.

Kaur et al. [14] reviewed meta-heuristic techniques for tackling university course timetabling problems. Techniques to solve timetabling problems were classified into four types: operations research-based techniques (graph colouring, integer linear programming, constraint satisfaction programming), meta-heuristic techniques (case-based reasoning approach, single-solution-based techniques, population-based techniques), intelligent novel approaches (hybrid approaches, fuzzy theory-based approaches, artificial intelligence-based approaches, and clustering algorithms), and multi-agent based approaches. One integer programming solution was included in the operations research-based techniques.

Bashab et al. [15] investigated the optimisation techniques in university timetabling problems through 155 academic publications. They summarised the hard and soft constraints for examination timetabling and course timetabling, ten types of methodologies, benchmarks, and open issues. The ten methodologies studied were sequential methods, cluster methods, constraint-based methods, meta-heuristic methods, generalised search, hybrid evolutionary algorithms, multi-criteria approaches, case-based reasoning techniques, hyper heuristics, and adaptive approaches. The algorithms used in the methodologies were explained in detail. A diagram illustrating the hierarchy of the university timetabling

approaches was proposed. However, no integer programming was mentioned in this review.

The key topics of these review studies are listed in Table 1.

Table 1. Key Topics of Existing Review Studies.

Paper	Key Topics
Qu et al. [4]	Search methodologies and automated approaches
Pillay [6]	Hyper-heuristics approach
Aziz and Aizam [7]	The features and constraints
Stephen [8]	Iterative improvements methods
Hosny [9]	Meta-heuristic approaches
Bashab et al. [10]	Meta-heuristic algorithms
Alghamdi and Alsubait [1]	Techniques, tools, advantages, and disadvantages
Chen et al. [11]	Various solution methods and constraints analysis
Tan et al. [12]	Optimisation methodologies
Zhu, Li, and Li [13]	Computational intelligence
Kaur et al. [14]	Meta-heuristic techniques
Bashab et al. [15]	Ten types of optimisation techniques

The existing review work provides a picture of evolution in the technical approaches in solving university timetabling problems. The techniques in the reviews start from search mythologies and heuristics, moving to meta-heuristics and computational intelligence. Integer programming has always been an important technique in this space. Nevertheless, there is no review conducted on the integer programming approach and the tools used in tackling university timetabling problems. Moreover, there is no review on adopting machine learning algorithms in solving university timetabling problems. Therefore, our review aims to gain a deeper understanding of integer programming and machine learning-based solutions relevant to timetabling problems at universities.

In Section 3, our review methodology is documented in detail. The results of this review, and corresponding answers to the research questions, are provided in Section 4. Section 5 summarises the findings and provides a discussion of opportunities of integer programming and machine learning in university timetabling and future work.

3. Methodology

To achieve the objectives of our review, the relevant university timetabling publications were collected from a bibliographical databases. Among all bibliographic data sources, Scopus, Web of Science, and Google Scholar are recognised as the major three bibliographic databases. Studies indicate that the coverage of Web of Science has been declining in recent years. Google Scholar's coverage is so broad that it includes academic lecture slides [16]. Scopus has been proven having the widest coverage in peer-reviewed scholarly journals and publications among bibliographic databases [16,17]. Therefore, Scopus is adopted as the only bibliographic data source for this review.

There are two steps in the literature search to serve the objectives of this review. The first step focuses on integer programming applied to university timetabling, while the second step explores the use of machine learning in university timetabling.

3.1. Integer Programming-Related Literature

To investigate the integer programming applied to university timetabling, three key components were designed and used in the search query. Firstly, all published work needs to be related to timetabling. Secondly, those published studies need to be in the university space. Thirdly, the published studies need to be related to integer programming or linear

programming. Based on the search keywords identified in the three components, the Boolean search string is constructed: TITLE-ABS-KEY((“timetabl*” AND “universit*”) AND (“integer programming” OR “linear programming”)).

The parameters of the search for integer programming solutions are listed in Table 2.

Table 2. Bibliography Data Source and Parameters of Search I.

Data Source:	**Scopus**
Search By:	TITLE-ABS-KEY
Search Terms:	timetabl* AND universit* AND (“integer programming” OR “linear programming”)
Article Type:	Journal articles, conference papers, working paper, book section
Search On:	Title, Abstract, Keywords
Time Frame:	1990 to 2023
Search Date:	2 May 2023

Through the designed Scopus search, 154 records have been returned and collected. The downloaded results include publishing years, authors, titles, abstracts, and keywords. To determine whether a paper is relevant, the paper needed to match the following inclusion criteria: the topic needed to be related to timetabling, university, and integer programming/linear programming; the articles needed to be in full text; and the articles needed to be published in English. The initial screening and mapping reviewed the abstracts. After the abstract screening, a set of 141 articles were identified as relevant and 13 articles were identified as irrelevant. The final assessment work required the context review of the full text. We could not locate 12 articles in full text. Two sets of duplicate papers from the search results were presented with the same article name and the same authors’ names but in different years. Duplicate papers were considered as only one. One paper from the search results was not written in English. After the full text review, another 31 papers were identified as irrelevant to the integer programming solutions to university timetabling problems. As a result, 95 articles are included in this review. The data collection and screening process are presented in Figure 1.

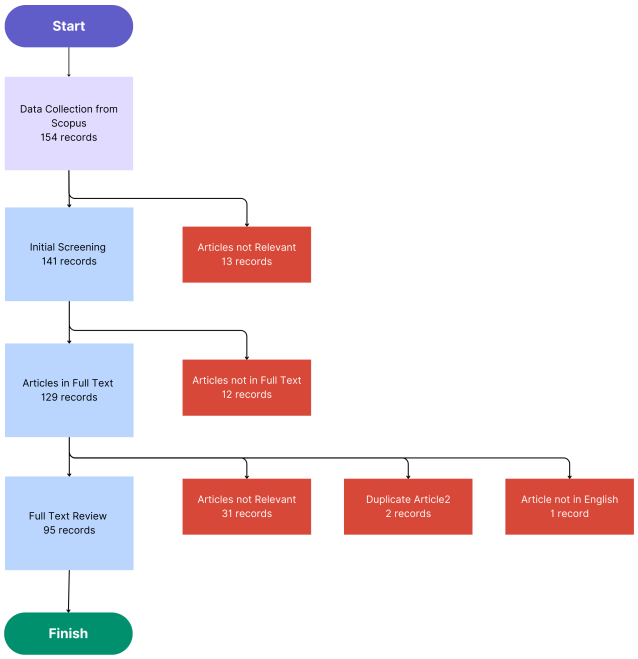

Figure 1. Data Collection and Screening Process.

In this review, integer programming, linear programming, and integer linear programming are interchangeable words.

3.2. Machine Learning-Related Literature

To explore how the machine learning algorithms are applied to university timetabling, again three key components were designed and used in the search query. Firstly, all published work needs to be related to timetabling. Secondly, those published studies need to be in the university space. Thirdly, the published studies need to be related to machine learning. Based on the search keywords identified in the three components, the Boolean search string is constructed: TITLE-ABS-KEY((“timetabl*” AND “universit*”) AND (“machine learning”)).

The parameters of the search for machine learning solutions are listed in Table 3.

Table 3. Bibliography Data Source and Parameters of Search II.

Data Source:	**Scopus**
Search By:	TITLE-ABS-KEY
Search Terms:	timetabl* AND universit* AND (“Machine Learning”)
Article Type:	Journal articles, conference papers, working paper, book section
Search On:	Title, Abstract, Keywords
Time Frame:	1990 to 2024
Search Date:	8 November 2024

Through the designed Scopus search, 45 records have been returned and collected. After the full text review, 6 papers are identified as relevant to the machine learning solutions to university timetabling problems. In this review, only the articles that meet the search criteria are considered relevant. As a result, these 6 articles are included in this review.

4. Results

In this section, we present the results of the review based on the 95 relevant articles that were collected, screened, and assessed from Scopus bibliographic database.

4.1. Research Question 1

The first research question is “RQ. 1: What are the existing timetabling problems in higher education that have been effectively addressed through integer programming?”

A timetable is a table of details that shows when occurrences in advance are arranged to take place for a specific time. Several ways are conveyed as follows in the literature [18]:

Ref. [19] defined timetable as, “Timetabling is the allocation, subject to constraints of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible a set of desirable objectives.”

Ref. [20] stated it as, “The timetabling problem consists in scheduling a sequence of lectures between teachers and students in a prefixed period of time (typically a week), satisfying a set of constraints of various types.”

“A timetable is a placement of a set of meetings in time. A meeting is a combination of resources (e.g., rooms, people and items of equipment), some of which may be specified by the problem, and some of which must be allocated as part of the solution” [21].

From the reviewed literature, the studied timetabling problems in the higher education context can be largely divided into three categories: course timetabling, classroom timetabling, and exam timetabling. These are discussed in the following section.

4.1.1. Course Timetabling

This is also referred as university course timetabling or curriculum-based course timetabling. It consists of the weekly planning of the lectures for a number of courses offered by the university within a restricted number of rooms and time schedule without overlapping and avoiding conflict withthe academic curriculum. It turns into a difficult scenario whenever there are unpredictable changes in course enrolments or limited accessibility to the rooms, due to running numerous classes simultaneously. In this section, we have covered various aspects of timetabling through different studies.

Ref. [22] has defined the course timetabling as a subsection of the planning problem. The universities and educational institutions must allocate the lecture to the lecturers and the respective rooms to the lecturers in different time slots in this issue. The goal underlying the timetabling problem is that the student should be able to select different subjects from the course they are enrolled in to create a timetable that fits their preferences and should be acceptable.

Ref. [23] has stated that the university course scheduling problem is the problem of designating days and hours to the courses, taking the limitations into account such as the availability of the lecturers, the positions of the lecture venues, and the direction of the educational institutions.

Ref. [24] has focussed on the university timetabling. They studied the traditional approach to recognise the practical problems related to timetables. Here, the objective is to assign lectures to rooms and time slots and outline a better timetable for students and teachers. This area has acquired a lot of acknowledgement as the problem endures, and it differs between countries and various stages of education.

As we know, each course consists of several lectures and each day consists of various time slots. A period is a term used for a distinctive day and a set time slot. Ref. [25] has identified that to plot a university course timetable, the following points must be considered: No two courses may be preplanned in parallel for a curriculum, which is defined as a set of courses. To arrange any lecture for a period, the room should provide sufficient space to conduct lectures. Two courses should not take place at identical times in the same room, and the same teacher cannot teach two courses at the same period.

4.1.2. Classroom Timetabling

In this category, the center of attention is the classroom. The objective is to find the most suitable allocation of the classroom as per the availability of where the courses are conducted, considering there is no issue with the assignment of the teacher. The capacity of the classroom, accessibility on a particular day, and avoiding overlapping of the classroom are the major factors while considering the proximity of classrooms to minimise the changes which may occur in a day-to-day allocation. Classroom timetabling includes classroom timetabling, lecturer timetabling, and student timetabling. Classroom timetabling refers to the scheduling part, where classes are assigned a time and a room (if required) [26]. Lecturer timetabling aspires to reduce the overall dissatisfaction and partiality costs related to assigning classes to faculty members [27]. Teacher timetabling is based on entertaining teachers' preferences regarding teaching days of the week or classrooms for designated courses [2]. The rules to fulfil the requirements include organising lectures, assessing different features and appeal for different lecturers, and taking into consideration sessions with successive time periods and/or reoccurrence of the same course in different sessions. The preferred subjects are then allocated to the lecturers and associated with a set of acceptable rooms.

Student Timetabling According to [26], organising students according to their course enrolment and the curriculum structure where the students must be enroled in specific

classes is another feature to consider. It is obvious that this problem arises when courses and classes with a substantial number of numerous teaching groups are scheduled concurrently [28].

4.1.3. Exam Timetabling

This is an unavoidable issue for educational institutions. Each institution has its own definite limitations; however, the principal structure is the same, where exams must be allocated to time slots and classrooms. In several institutions, the problem is resolved manually, but as the availability is restricted, it becomes more strenuous when the students' numbers are becoming larger every year. It requires skill to set exams into appropriate time slots and classrooms, taking constraints into account along with individual inclinations. Some universities have the capability to establish software themselves, but some still deliver the timetable by constructing it manually because they lack automatic software support [29]. This issue appears at the end of every trimester or semester, and it is quite challenging for the planner, hence the associated supporting tools are of utmost importance. According to this study, to create an examination timetable several rules must be satisfied and some of them are as follows: each exam must be designated to a particular day and slot; there must be only one exam in a classroom for a slot; instructors may have more than one course and they may ask to have their examinations at the same time and they also may choose specific days or slots for exams; if there are significant numbers of students enroled in any two courses, then their exams must be in different time slots and their exams must be one slot apart, i.e., there should be 90 min between two consecutive exams; and, in contrast, some courses' exams must be held simultaneously [29].

All the different aspects of timetabling emphasis on course timetabling, classroom timetabling, and exam timetabling are covered in this analysis. A visualised pie chart is below to demonstrate the distribution of the timetabling categories: course, classroom, and exam timetabling (Figure 2).

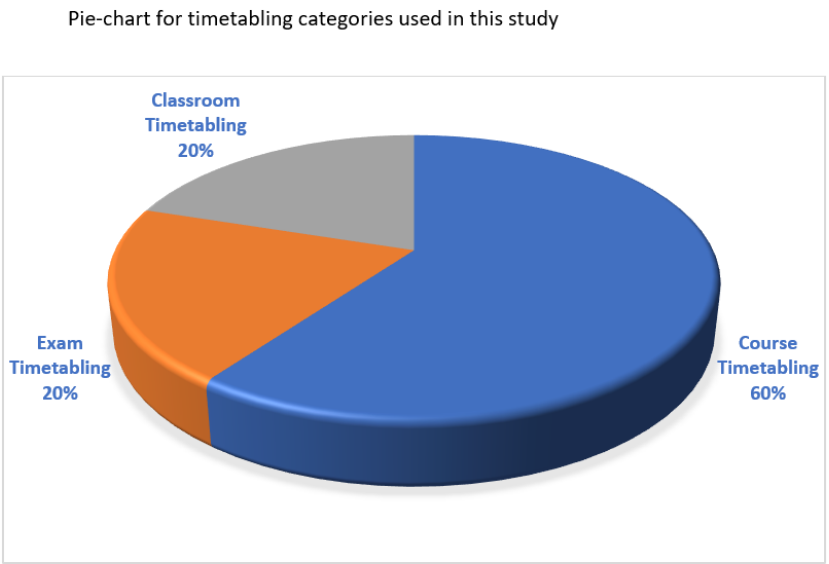

Figure 2. Timetabling problems categories.

Figure 2 shows various types of problems associated with timetabling. The three main categories are represented: course timetabling, classroom timetabling, and exam timetabling. These were studied through 95 reviewed articles in this study.

4.2. Research Question 2

The second research question is phrased as, "How is integer programming employed to solve these timetabling problems?". This research question is answered based on the algorithms used for solving timetabling problems from the literature reviewed in RQ1.

Researchers have used various algorithms to solve timetabling problems in higher education institutes; however, this paper mainly reviewed the literature considering the integer linear programming algorithm (ILP). A very old mathematical subject in the name of operational research has developed theories of integer linear programming, mixed-integer linear programming, and integer linear programming with heuristic or meta-heuristic approaches. Earlier applications of these theories were production planning and scheduling, including service and vehicle scheduling in transportation networks, territorial portioning, telecommunication networks, and cellular networks. In 1973, a mathematician E.A.Akkoyunlu [30] applied integer linear programming theory to write an algorithm to solve university timetabling problems. His algorithm was used in American universities in the 1970s and many obtained an optimal solution for solving timetabling issues.

This worked as a trigger for research in this field as many researchers from different countries started to create timetables for their departments, schools, faculties, or universities using integer linear programming (ILP). In the 95 reviewed articles, researchers used mixed-integer linear programming (MILP), integer linear programming with heuristic (ILPH) or meta-heuristic approaches, or mixed-integer linear programming with heuristic (MILPH) approaches. In some cases, the computation and time requirements increase exponentially with the increase in data. To avoid this, researchers split the mathematical equations into two phases to find an optimal solution.

Figure 3 gives a comparison of each linear programming model used in all the reviewed papers. The figure shows that ILP is very popular, which is understandable as many variables used are integer variables in timetabling algorithms with linear programming. We identified that 90% of our reviewed papers used variables that take values zero or one. These models were described as binary integer linear programming (BILP).

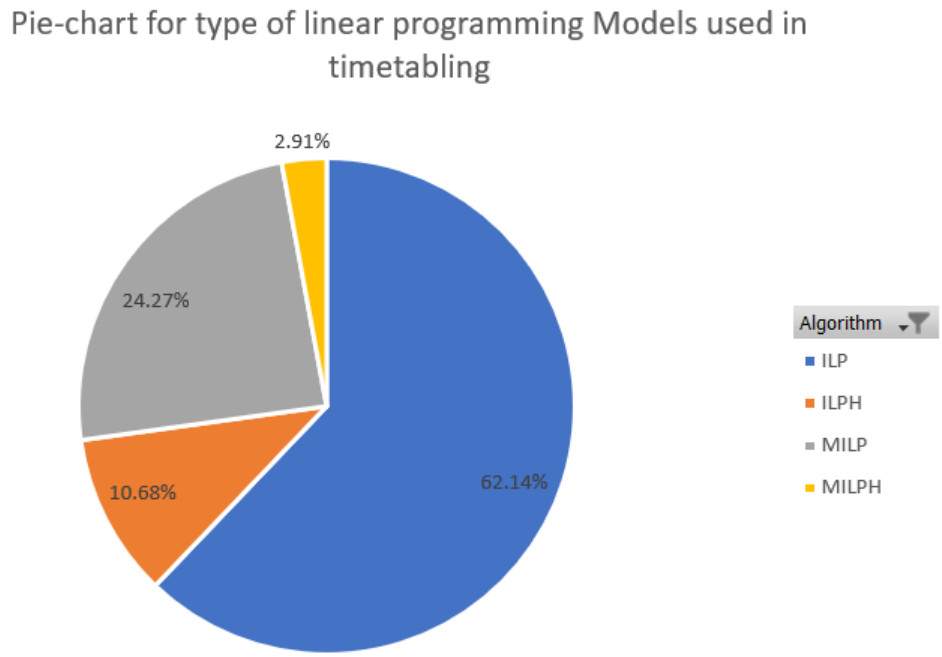

Figure 3. Pie Chart of Linear Programming Models Used.

Further in our comparison, Figure 4 shows the types of linear programming models used for particular purposes of timetabling, namely, classroom, course, and exam. Only three timetabling purposes were considered as all other categories are subparts of these three main categories. For example, the lecturer timetable comes under the course timetable. Further to this, we consider classroom timetables as course timetable. The figure identifies that mostly course timetabling was performed with ILP and very few researchers considered MILPH for any type of timetabling. MILPH was used when a large number of subjects were considered, which was very rare, as shown in Figure 4.

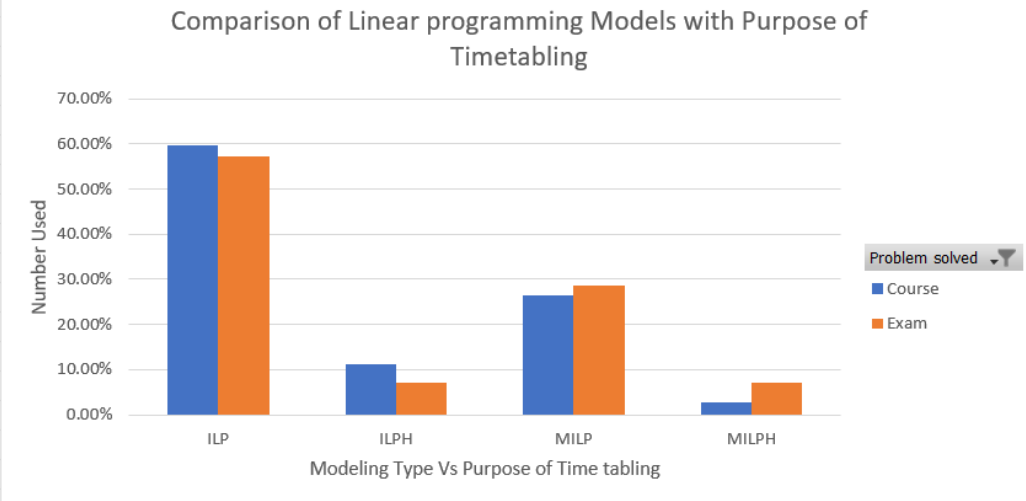

Figure 4. Comparison of Linear Programming Models with Purpose of Timetabling.

In the case of university timetabling problems, when we set up a linear model, almost all of the variables, which we may have to introduce, take integer values. In our review papers, those who use ILPH, MILP, or MILPH for university timetabling and had not reached the optimal solution instead obtained either approximate, feasible, or good approximate solutions. The heuristic or meta-heuristic approaches, especially, need many iterative mathematical calculations and tend to have approximate or heuristic solutions. In many cases, the solutions are very hard to interpret. This evidence shows that integer linear programming is a powerful algorithm to obtain optimal solutions for timetabling problems for universities.

The general strategy for writing a linear programming algorithm is summarised in Figure 5, based on the papers reviewed for this work. The details of the steps involved in the linear programming algorithm are provided in the following subsections. For improved readability, some steps are combined in the discussion below.

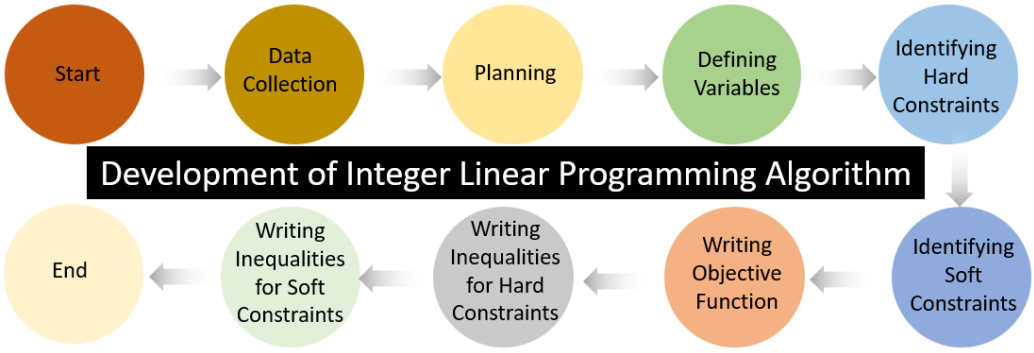

Figure 5. Flow Chart for Developing Integer Linear Programming Algorithm.

4.2.1. Data Collection

The very first step for writing an algorithm for university timetabling using linear programming is to collect all the information needed to complete the task. The collection of information varies according to universities and the types of timetables to create. A list of the most common information to be collected is given below, as considered in most of the reviewed papers for course timetabling:

- University, faculty, and departmental policies regarding timetabling;
- List of courses;
- List of lecture groups for each course;
- List of tutorial groups for each lecture group;
- Number of rooms available;
- Capacity of each room;
- Students' preferred time for a class;
- Lecturers and tutors' preferred time slots;
- List of maximum number of hours staff can work.

4.2.2. Planning and Defining Variables

After data collection, a thoughtful plan needs to be created to write an appropriate and efficient linear programming model. This leads to the definition of variables in notation form and to write the objective functions and constraint expressions. Careful consideration should be taken in finding notations for any variables as in some cases [31], more than 100,000 variables may be declared. When the timetable is developed for different semesters, the number of variables increases [32], where the largest number of variables is 1,010,160 and the smallest is 296,500. It is interesting to see that authors define notations for indices, parameters, and sets in addition to variables [23].

4.2.3. Hard and Soft Constraints

Constraints are an important part of linear programming models. In scheduling problems, there are two types of constraints. Hard constraints need to be satisfied to obtain an optimal solution and soft constraints can be violated, if required. However, these constraints must be satisfied as much as possible in the process of obtaining controversy-free schedules. These two classes of constraints differ according to institutions, semesters, and type of timetable to create. In some cases, a hard constraint becomes a soft constraint and vice versa. With more constraints, more processing is required, which can take lots of time to produce output; mathematical simplification can be used to reduce the number of constraints. Common hard and soft constraints are given below.

Hard Constraints:

- No two subjects can be assigned at the same time and same room.
- No two subjects can be allocated to a teacher at the same time.
- Tutorial for a group should come after the lecture class of the subject for that group in a week.
- Every subject has to be allocated a time slot and a room.
- Student number should not exceed the capacity of the room allocated.
- University, faculty, school, and department policies.

Soft Constraints:

- Teacher time and room preferences.
- Popular student time slot.
- More than three subjects for a student in a day.
- Closer rooms for teachers and students between next-hour classes.

4.2.4. Writing Objective Functions and Inequalities for All Types of Constraints

Solving a mathematical problem via linear programming involves an objective function that needs to be optimised with the use of many inequalities considering the same variables in the objective function. These inequalities come from hard and soft constraints. There can be more than one objective function to optimise. The following are some examples of writing objective functions:

- Maximise students' and teachers' preferences;
- Minimise students' and teachers' travel distance;
- Maximise to satisfy institution policies.

4.3. Scalability in Integer Linear Programming

We have identified that the ILP algorithm is a powerful tool to solve timetabling problems in universities. However, their scalability can be a significant issue when dealing with larger and complex scenarios.nFor an example, in the university timetabling problems, each course, room, and time slot can be represented by a variable, leading to a large number of variables.

Challenges in Scalability We have seen that number of variables increases exponentially. In addition, we will have an increase in the number constraints. For example, conflicts between courses, respecting room capacities, and faculty causing an increase in the number of constraints. Many ILP solvers struggle with the problem that they have a large number of variables and constraints, especially when the problem involves many possible combinations. Larger problems might lead to infeasible or suboptimal solutions. The need for exact optimisation may not be practical due to time complexity.

Approaches to Improve Scalability Decompose the larger-scale problem into smaller subproblems. For example, the university timetabling problem can be divided into each faculty, school, or department timetable problem. Another way is to consider very important variables, and other variables can be added iteratively as needed. In some cases, relaxing some constraints temporarily can lead to solutions faster, after which theses constraints are gradually reintroduced to redefine the timetable. Machine learning can be integrated with ILP to predict a promising schedule or help in constraint propagation, which could reduce the complexity of the search problem. For instance, AI can be used to identify patterns in existing timetables that can then be leveraged to guide the search process for new solutions.

4.4. Research Question 3

Based on the review of RQ1 and RQ2, integer programming has been proven an effective mathematical optimisation approach that deals with university timetabling problems where the constraints and decision variables are integer values. The third research question is "What tools, including software libraries and frameworks, are utilised in tackling these timetabling problems using integer programming?". In this section, 95 existing studies are reviewed on how various tools and algorithms have been developed to tackle the challenging timetabling optimisation problems efficiently over the years.

4.4.1. Implementation Rate

Out of these 95 publications, 2 of them present the proposed conceptual solutions without any accompanying implementations. In 93 out of 95 publications (98%), they reported implementations using integer programming to provide university timetabling optimisation solutions. From the review results from Bashab et al. [10], only 34% of 131 publications have accomplished implementation using meta-heuristics algorithms. The implementation rate is determined by whether any results are published in the publications.

The evidence shows that the implementation rates are higher using integer programming algorithms than using meta-heuristics ones in tackling university timetabling problems.

Although the meta-heuristics approach has many advantages such as being more suitable for large and complex instances, faster processing speed to find solutions, and more flexible in handling various problem types [9,10], the comparison results indicate that integer programming could be a more approachable and achievable solution than meta-heuristics in approaching timetabling problems.

4.4.2. Overview of Tools

All solutions proposed from 95 publications are visually distributed to a bar plot in Figure 6.

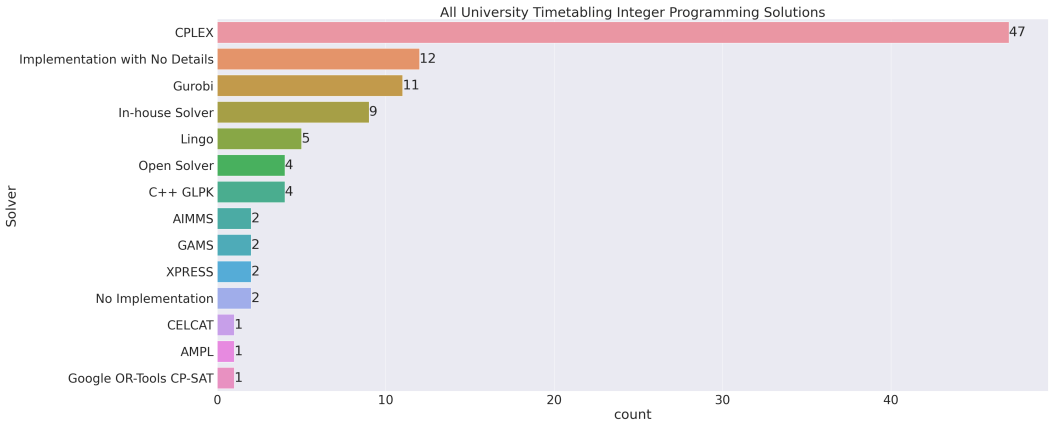

Figure 6. All University Timetabling Integer Programming Solutions.

Among the pool of 95 publications, 2 publications introduce only conceptual models and 12 publications showcase results without delving into implementation details. Among 81 publications, 8 publications offer dual solutions using distinct approaches. Within 89 solutions, there are 9 solutions that feature proprietary in-house solutions, and 80 solutions adopt commercial solvers. The commercial solvers presented in this review are including CPLEX (47), Gurobi (11), Lingo (5), Open SOlver (4), C++ GLPK (4), AIMMS (2), GAMS (2), XPRESS (2), CELCAT (1), AMPL (1), and Google OR-Tools CP-SAT (1). The number of applications are written in the brackets after the solver names.

To investigate the development of various solvers in this particular research area across the review period, all solutions are plotted on a bar chart based on the year in Figure 7.

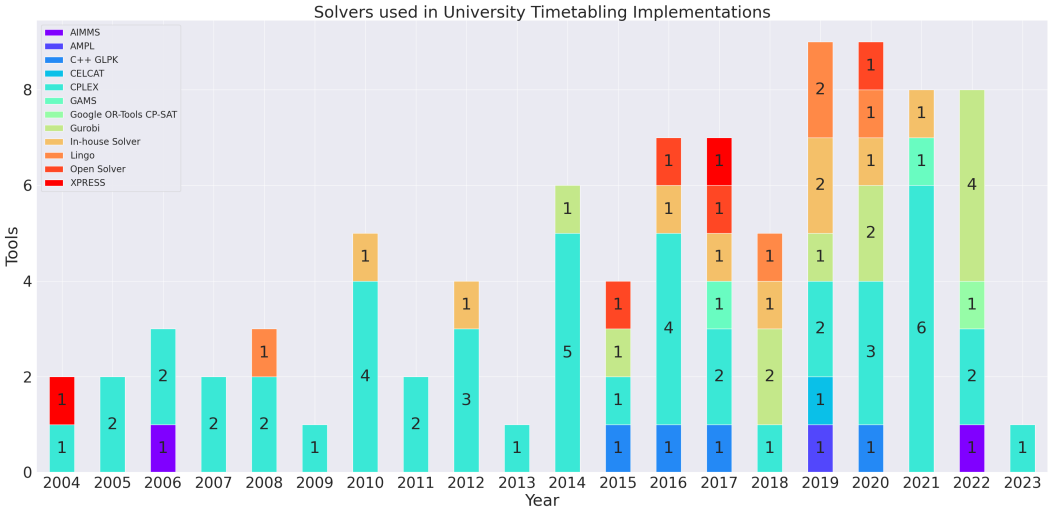

Figure 7. Solvers Used in University Timetabling Implementations.

4.4.3. Tools

The tools and their applications are explained in the following subsections.

- CPLEX

From Figure 6, it is obvious that CPLEX is the most frequently used integer programming solver in approaching university timetabling problems in this review. To investigate deeper into the the usage of this particular solver, the number of CPLEX implementations is counted by year and visualised in Figure 7.

CPLEX is a commercial optimisation software package developed by IBM. It is specifically designed to solve complex optimisation problems, including linear programming, mixed-integer programming, quadratic programming, and other related types of mathematical optimisation [33]. Due to its robustness, speed, and versatility, it is reported that CPLEX is widely used by researchers, analysts, and practitioners to solve challenging optimisation problems in diverse fields [34]. From 2004 to 2023 across 19 years, CPLEX has been updated to various versions. In this review, 47 implementations (Figure 8) adopted versions from CPLEX 5.1 [2,3] to CPLEX 12.8 [35] and CPLEX 12.9 [36]. Industrial software CPLEX can efficiently handle formulations with thousands of constraints and variables [37]. Based on our review, the total number of variables and constraints solved by CPLEX has reached more than 100,000 [38]. This integer programming solver has definitely gained popularity and success in addressing university timetabling optimisation problems.

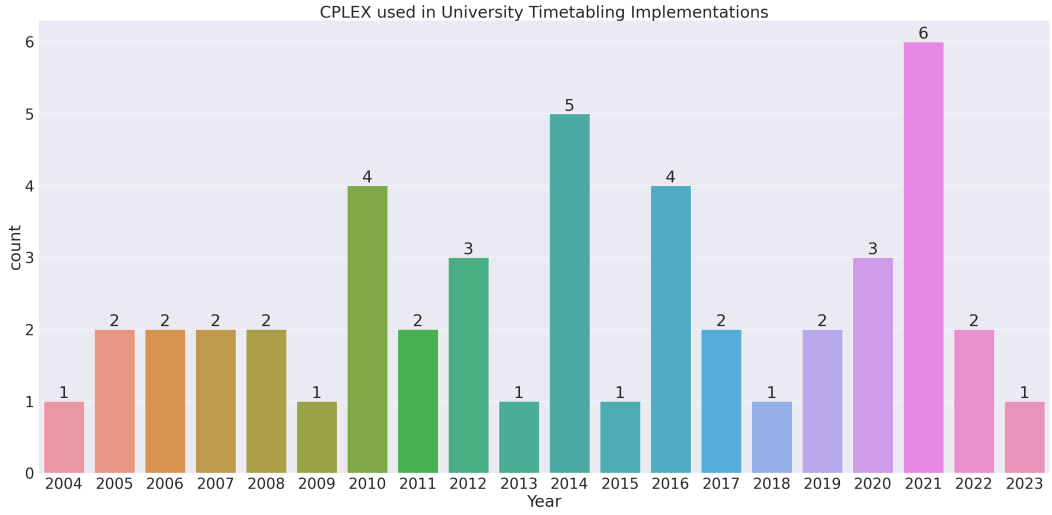

Figure 8. CPLEX Used in University Timetabling Implementations.

- Gurobi

Gurobi is widely regarded as a premier optimisation tool for providing robust and efficient solutions for the optimisation problems [39]. It is widely used in academia and industry to address complex optimisation challenges. Gurobi offers efficient solvers for linear programming and mixed-integer programming problems, as well as solvers for quadratic programming and nonlinear programming problems. Additionally, Gurobi includes global optimisation algorithms that aim to find global optima for non-convex optimisation problems. Gurobi offers APIs and interfaces for various programming languages, including Python, C++, Java, and C# in .NET.

Gurobi is ranked as the second most frequently used commercial solver in untangling university timetabling problems. Among all publications that we reviewed, 11 of them used Gurobi for implementation [26,29,40–48].

- Lingo

Developed by LINDO Systems, Lingo is a software tool that provides a user-friendly environment with modelling capabilities, and powerful algorithms for solving optimisation problems, including linear programming, integer programming, quadratic programming, global optimisation, and so on [49]. It is commonly used by organisations and researchers to address various optimisation challenges. In this review, we found that five publications used Lingo for their implementation work [50–54].

- Open Solver

Provided within Microsoft Excel, open solver is an add-in tool that provides optimisation capabilities various types of problems, such as linear programming, integer programming, nonlinear programming,and quadratic programming [55]. In this review, there are four applications developed using Open Solver [56–59]. The solver tool is commonly used to solve optimisation problems without the need for specialised optimisation software. It has an easy-to-use graphical user interface through Microsoft Excel.

- C++ GLPK

GNU Linear Programming Kit (GLPK) [60] is an open-source software package for solving large-scale linear programming (LP), mixed-integer programming (MIP), and other related problems. It provides a C/C++ library. For defining optimisation problems, it provides a modelling language called GNU Math Programming Language (GMPL), a subset of AMPL language. The GLPK package includes six main components: primal and dual simplex methods, primal–dual interior-point method, branch-and-cut method, translator for GNU MathProg, application program interface (API), and stand-alone LP/MIP solver.

The implementations using C++ GLPK in this review include four applications [61–64]. However, three out of four implementations conducted a comparison and present the parallel results from CPLEX and C++ GLPK for the same university timetabling problems.

- AIMMS

Advanced Integrated Multidimensional Modelling Software (AIMMS) is an optimisation modelling platform with a user-friendly interface designed to model and solve complex decision-making problems [65]. There are only two AIMMS implementations presented in this review [66,67]. The highlights of this software are not only about solving the optimisation problems using advanced algorithms, but also on visualising the results. AIMMS offers a comprehensive modelling environment that allows users to create optimisation models using a high-level modelling language. This solver supports linear programming, mixed-integer programming, nonlinear programming, and constraint programming. This software can interface with various optimisation solvers, but it does not provide APIs to interface with various programming languages.

- GAMS

General Algebraic Modelling System (GAMS) is a high-level modelling and optimisation software designed to facilitate the formulation and solution of large-scale mathematical programming and optimisation problems [68]. In this review, GAMS has been employed in two cases of implementation [32,69]. This optimisation software provides a user-friendly environment for modelling complex decision-making problems and finding optimal solutions, where objective functions, decision variables, and constraints can be defined in a readable way. GAMS supports various solvers for linear programming, mixed-integer programming, nonlinear programming, and other optimisation techniques. Besides its visualised interface for development, GAMS can be integrated with external modelling languages, including Python, MATLAB, Excel, and so on.

- XPRESS

Developed by the Fair Isaac Corporation (FICO), XPRESS is a mathematical optimisation software suite equipped with a wide range of optimisation techniques and tools for solving complex mathematical programming and optimisation problems [70]. Two XPRESS application implementations are presented in this review [71,72]. XPRESS has strong capabilities in solving problems in linear programming, mixed-integer programming, quadratic programming, nonlinear programming, global optimisation, constraint programming, and meta-heuristics. The interfaces provided by XPRESS APIs integrate to various programming languages, including C++, Java, .NET (C#), and Python.

- AMPL

A Mathematical Programming Language (AMPL) is a high-level modelling language and environment designed for formulating, solving, and analysing mathematical optimisation problems [73]. AMPL offers a powerful modelling language used to describe optimisation problems. It provides a user-friendly platform to describe various optimisation problems, including linear programming, integer programming, nonlinear programming, mixed-integer programming, quadratic programming, and constraint programming. AMPL is widely used in academia and industry to model and solve complex decision-making problems. AMPL can be interfaced with various optimisation solvers so the end users can choose different solvers for experiments. However, it does not support any APIs to programming languages.

In this review, we found that only one application had been developed by adopting the AMPL solver [74]. However, it is worth mentioning that there are two publications that used AMPL as the model platform [75,76].

- CELCAT

CELCAT is a commercial software solution designed to assist in managing and optimising the scheduling and timetabling processes for educational institutions and organisations [77]. This software takes into account various constraints including room availability, instructor preferences, student groups, and so on. The highlighted feature of CELCAT lies in the automated scheduling in timetabling, exam scheduling, and resource management.

Abdalla et al. adopted CELCAT as the solver to tackle their university timetabling problems [78]. This is the only scholarly publication using a commercial off-shelf solver to untangle the university timetabling problem. In reality, it is believed that there are many similar implementations.

- Google OR-Tools CP-SAT

The Google OR-Tools library, which includes the CP-SAT Solver, was first released in September 2013. This tool is a powerful constraint programming solver as part of the Operations Research Tools (OR-Tools) open-source software suite. It is designed to solve combinatorial optimisation problems by finding feasible solutions that satisfy a set of constraints [79]. This is not a new IP solver. It was not developed for commercial purposes. Google OR-Tools CP-SAT had not gained much popularity. However, its first implementation in resolving university timetabling optimisation problems was published in 2022 [40].

- In-house Solvers

In-house solvers refers to custom-built optimisation tools developed specifically for addressing university timetabling problems, often using general-purpose programming languages like Java, Python, or C#. The first application in this review was published with an in-house solver. There is a total of nine in-house solver implementations from the most recent 13 years. In this review, the programming languages used for in-house solvers

include Java [58,71,80–82], MATLAB [83], C# [24], and Python [84,85]. The solver names and the implementation years are presented in Table 4.

Table 4. In-house Solvers.

Year	Solver	Published Application
2021	Python	[85]
2020	MATLAB	[83]
2019	Python	[84]
2019	Java	[82]
2018	C#	[24]
2017	Java	[71]
2016	Java	[58]
2012	Java	[81]
2010	Java	[80]

No Java built-in integer programming libraries are mentioned in the in-house implementations in this review. Java does not provide any built-in library specifically tailored for integer programming or mixed-integer programming. Like Java, C# does not provide any built-in integer programming libraries. All implementations in Java and C# in this review could have used their own versatile math libraries to perform mathematical calculations to perform the basic optimisation tasks. Alternatively, commercial software suites such as Gurobi and XPRESS could have been adopted to integrate with Java or C# applications to provide the optimal university timetabling solutions.

MATLAB's optimisation toolbox provides functions like linprog [86] for linear programming. Though there is a lack of insights from the publication [83], MATLAB implementation could also be using a commercial solver for the implementation.

PuLP [87] is a high-level modelling library that allows the users to create integer programming solutions using Python expressions. Two Python implementations [84,85] could build on top of the PuLP library or the integration with external commercial solvers.

Interestingly, there is a trend in in-house solvers shifting from Java to Python/MATLAB. The main drive behind this phenomenon could be that Python and MATLAB are equipped with built-in libraries to support integer programming for optimisation problems.

This review covers ten types of tools that have been adopted to provide integer programming solutions for university timetabling problems. They are CPLEX, Gurobi, Lingo, Open Solver, C++ GLPK, AIMMS, GAMS, XPRESS, AMPL, and Google OR-Tools CP-SAT. This review also presents two software libraries: PuLP from Python, and linprog from MATLAB. Based on our analysis, it can be concluded that optimisation solvers have reached a level of maturity and they are able to effectively tackle real-world problems.

4.4.4. Performance Comparisons

In this review, performance comparisons cover the comparisons over solution quality, time efficiency, scalability, and robustness for the major IP solvers. Solution quality shows how close the solutions are to the optimal or best-known solutions. Time efficiency shows how fast the solver runs in finding a solution, especially for large instances. Scalability is to measure how well the solver handles larger problem sizes or more complex constraints. And robustness is for how well the solver performs across different types of timetabling problems. The comparison results are presented in Table 5.

Table 5. Comparison of Tools Used in University Timetabling Optimisation.

Solver	Cases	Solution Quality	Time Efficiency	Scalability	Robustness
CPLEX	47	High	Fast	Very High	Very High
Gurobi	11	High	Very Fast	High	Very High
Lingo	5	Moderate	Moderate	Moderate	Moderate
Open Solver	4	Low-Moderate	Slow	Low	Low
C++ GLPK	4	Moderate	Moderate-Slow	Moderate	Moderate
AIMMS	2	High	Moderate	High	High
GAMS	2	High	Moderate	High	High
XPRESS	2	High	Fast	Moderate-High	High
AMPL	1	High	Moderate-Slow	High	High
Google OR-Tools	1	Moderate	Moderate-Fast	Low-Moderate	Moderate
CELCAT	1	High	Fast	High	Very High
In-house Solvers	9	Varies	Varies	Varies	Varies

In university timetabling optimisation, CPLEX, which appeared in 47 publications, is the most frequently used solver. It is known for its high-quality solutions and time efficiency. Studies have shown that CPLEX is able to handle problems with more than 100,000 variables in under 5 min [31]. Its robustness across different types of constraints including room assignments and exam scheduling is well documented, which makes it the go-to solver for large-scale problems. In terms of solution quality, CPLEX consistently produces near-optimal solutions, with deviations typically under 1%. Scalability is also a strong suit, as CPLEX handles large datasets with ease, though the cost of the commercial license may be a limiting factor.

Gurobi, used in 11 publications, also offers similar time efficiency and solution quality, often outperforming CPLEX in terms of raw processing speed. Gurobi is reported to solve university timetabling problems involving 50 rooms and 200 courses in under 2 min, while consistently maintaining a solution quality that is within 1% of the optimal solution [43]. However, Gurobi also comes with a higher licensing cost price tag.

In contrast, Open Solver is an open-source tool integrated with Microsoft Excel, which makes it a more accessible option. This solver appeared four times in the publications. Due to the limitations of Excel, it tends to be slower for large instances, with processing times significantly higher than those of CPLEX or Gurobi. Studies such as Marmolejo et al. demonstrate that Open Solver struggled with problems involving more than 50 courses or 5 rooms [58], making it less scalable for larger timetabling problems. Additionally, while it is a free tool, its solution quality often falls short compared to commercial solvers.

4.5. Research Question 4

The proposed timetabling approaches in the literature are tested using real-world data, competition datasets, and/or simulated data. It is interesting to observe that numerous research works did not mention the source of data used for the evaluation of the proposed techniques, as shown in Figure 9. Furthermore, a majority of the reviewed literature, around 61%, considered their dataset or data collected from other relevant sources, which signifies the importance of relevant data for customisable solutions. Competition datasets ITC 2019 [88] and ITC 2007 [89] are also used to gauge the performance of the timetabling algorithms.

From the reviewed papers that clearly mention the data used for testing the proposed approaches, a significant number of researchers have opted to utilise real-world university data, as shown in Figure 10. The choice of using real-world university datasets provides a unique ability to encompass distinctive constraints arising from university policies and regulations. These additional constraints, alongside the standard limitations

regarding resource availability and sequential arrangement, contribute to the intricate nature of these datasets. The size and complexity of these data can vary to a large degree as some researchers have used a smaller subset to test the initial design of the proposed techniques [90]. On the other hand, some techniques are tested with huge datasets spanning multiple semesters [5,38].

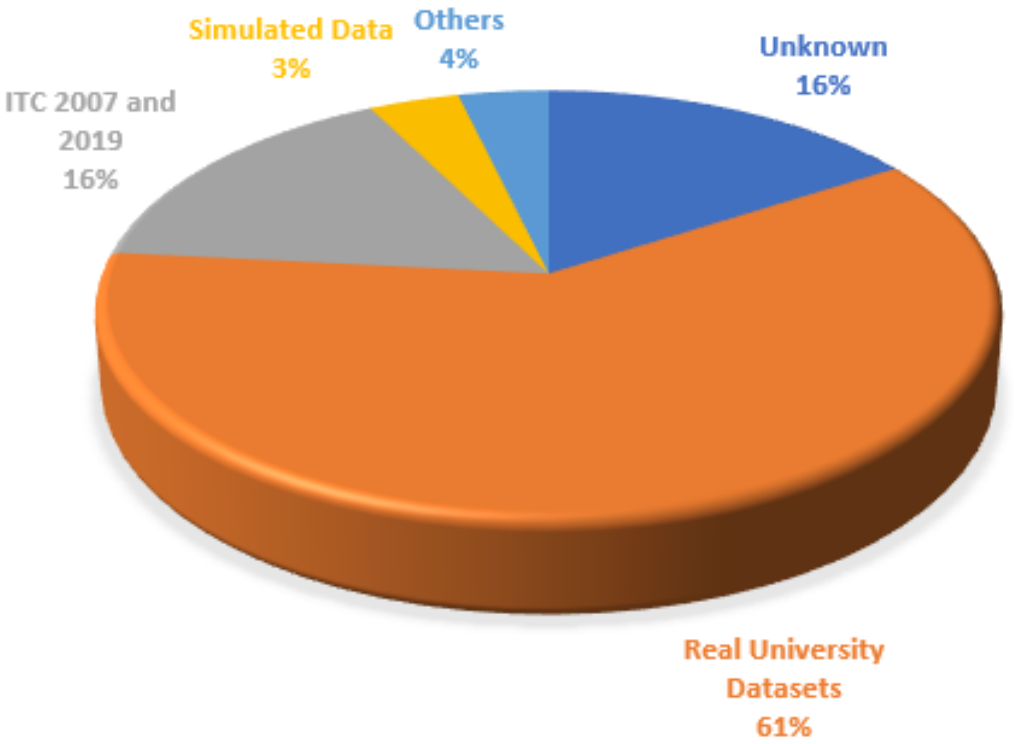

Figure 9. Types of Data Used for Evaluation, All Instances Considered.

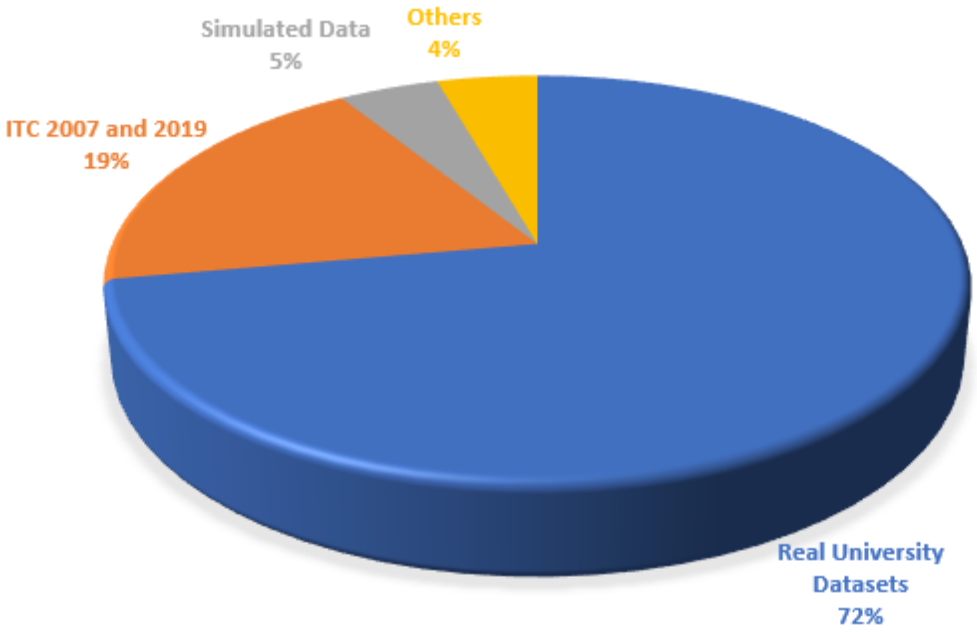

Figure 10. University Timetabling Implementations: Types of Data Used for Evaluation, Only Known Instances Considered.

Several competition datasets have been created for university timetabling challenges like Unitime [89] and ITC2019 [88]. The International Timetabling Competition (ITC) 2007 [89] and ITC 2019 [88] datasets are most commonly used by researchers in this domain, as inferred from the literature review. Competition datasets provide a standardised option to evaluate the performance and robustness of proposed approaches. Well-established benchmark instances in addition to realistic complex scenarios included in these datasets provide a perfect platform for fair comparison of efficiency, quality, and scalability. These datasets also provide numerous options to test proposed methodologies, for example, ITC 2019 includes data from ten different universities in eight countries on five continents [26].

Simulated data allow for controlled experimentation while limiting the irregularities and unexpected challenges of the real world. For university timetabling problems, this involves generating artificial instances that mimic real-world scenarios in a controlled manner. In the reviewed literature, simulated data are used by researchers and practitioners to test algorithms, optimise strategies, and evaluate early-stage solutions specifically when scalability, controlled variation, and reproducibility are the focus. Simulated data can be very helpful when used to compare and benchmark different algorithms and optimisation techniques for solving university timetabling problems. These simulations provide a controlled environment to compare the performance of various solutions and assess how well they handle different constraints and preferences. The simulated dataset can vary in size and complexity; [62] considered a simulated data with 33 teachers, 20 classes, 40 groups, and 23 courses.

4.6. Research Question 5

The six selected articles provide a picture of the recent advancements in using machine learning techniques for timetabling problems in higher education. Each article explores different machine learning applications to enhance the accuracy, efficiency, and adaptability of scheduling algorithms, showcasing both practical implementations and theoretical improvements.

Algorithm Selection and Optimisation: Several studies tackle the challenge of selecting the best algorithm for specific timetabling scenarios. Van Bulck et al. discuss a machine learning-based system that chooses optimal algorithms based on sports scheduling constraints, thus matching problem instances with the most effective algorithms and improving scheduling reliability [91]. Similarly, De Coster et al. focus on curriculum-based timetabling, where instance space analysis combined with machine learning helps to automate algorithm selection, resulting in a more adaptive and accurate scheduling system [92].

The development of a university schedule generator by Chougule et al. illustrates the potential of machine learning in balancing resources like educator availability, classroom allocation, and student requirements. This approach applies AI to navigate complex scheduling challenges, ultimately enabling efficient timetable generation within educational institutions [93].

The study by Feutrier et al. addresses the challenge of enhancing artificial timetabling instances to reflect real-world behavior more accurately. By implementing a feasibility prediction model trained on synthetic data, the authors achieve improved performance in timetabling solutions, bridging the gap between generated and real-world datasets [94].

Enhanced Simulated Annealing for Course Timetabling: Goh et al. introduce a novel adaptation of simulated annealing, integrating reinforcement learning to adjust search strategies dynamically. This approach, tested on post-enrolment course timetabling problems, improves solution quality by reducing soft constraint violations, thus meeting diverse institutional requirements more effectively [95].

Automated Parameter Selection: Finally, Alefragis et al. describe a machine learning-driven system for parameter tuning in scheduling algorithms, drastically reducing the need for manual adjustment. By learning from previously solved instances, the model adapts algorithm parameters automatically, optimising both development time and scheduling quality [96].

Collectively, these studies demonstrate the transformative role of ML in timetabling. By refining algorithm selection, instance generation, and optimisation processes, they enable more adaptive, efficient, and feasible scheduling solutions that align with the unique demands of higher education timetabling.

4.7. Research Question 6

Machine learning can be integrated with integer programming to address complex timetabling problems. This hybrid approach leverages the strengths of both methods. Integer programming provides rigor and precision in optimisation, while machine learning is used for insights and prediction. In the integer programming solutions, machine learning algorithms can be used for various steps, including instance selection and algorithm prediction, parameter turning, constraint relaxation and feasibility prediction, reduced search space for integer programming, and post-processing and solution refinement.

Instance Selection and Algorithm Prediction: Machine learning can be used to predict which IP solver or algorithm settings works best for specific timetabling instances. For example, if different problem types are anticipated (e.g., different combinations of constraints), machine learning can guide which IP methods or solver parameters will yield optimal results. This can reduce computation time and improve solution quality.

Parameter Tuning: Machine learning models can be trained to optimise parameters for the IP formulation, such as penalty weights for constraints or branching priorities in a mixed-integer programming context. This automation of parameter tuning, as seen in studies like that by Alefragis et al., allows IP methods to perform at their best without requiring extensive manual adjustment [96].

Constraint Relaxation and Feasibility Prediction: Machine learning can be used to identify infeasible instances or suggest constraint relaxations before integer programming is applied, helping to generate only feasible solutions. For example, Feutrier et al. used ML to predict instance feasibility, which could streamline the IP process by focusing on solvable constraints [94].

Reduced Search Space for integer programming: Machine learning, specifically deep learning or reinforcement learning models, can analyse data from previous scheduling instances and reduce the search space for integer programming by suggesting promising variable bounds or eliminating unlikely variable assignments. This can make large-scale integer programming problems more manageable by focusing on the most relevant decision variables.

Post-processing and Solution Refinement: After an integer programming solution is generated, machine learning algorithms can assist in refining the schedule by making adjustments that improve soft constraints, such as minimising scheduling conflicts or accommodating preference-based adjustments. Goh et al. integrated machine learning into simulated annealing for greater efficiency [95].

Machine learning provides insights, heuristic guidance, and parameter optimisation, while integer programming ensures that solutions are mathematically optimal, feasible, and adhere to the strict constraints inherent in timetabling. By combining machine learning and integer programming, it is possible to achieve a robust, efficient, and adaptable timetabling solution.

5. Conclusions

This paper presents a comprehensive overview of how integer programming plays a pivotal role in addressing timetabling challenges within higher education, encompassing the examination of existing problems, solution methodologies, solver tools, and dataset characteristics. The insights provided by this research will help further progress and enhance readers' understanding in this crucial field of study.

Scopus is adopted as the bibliographic data source to extract the relevant literature. In total, 95 articles are selected as the relevant literature for integer programming solutions and 6 articles are identified as the relevant literature for machine learning solutions. These 101 articles are assessed and analysed for these research objectives.

From the analysis of our review, integer programming gives a concrete optimal controversy-free timetable for any higher education. To produce the desired results, there must be a need for mathematical and computer skills. However, if a computer program is developed with these skills, then a software developer can make appropriate changes to run the program for a future need.

Through 95 integer programming-relevant articles, it can also be concluded that integer programming is an effective approach in solving university timetabling problems using various tools. This review includes a total of 10 types of commercial solvers, encompassing CPLEX (47), Gurobi (11), Lingo (5), Open Solver (4), C++ GLPK (4), AIMMS (2), GAMS (2), XPRESS (2), CELCAT(1), AMPL (1), and Google OR-Tools CP-SAT (1). CPLEX is reviewed as the most frequently used solver with 47 implementations in this review. CPLEX has proven its robustness, speed, and versatility to handle more than 100,000 constraints. This solver is rated as the most effective IP solver in the university timetabling domain.

The analysis of six machine learning-related papers showed that machine learning algorithms can be integrated with integer programming. Machine learning techniques can be used for instance selection and algorithm prediction, parameter turning, constraint relaxation and feasibility prediction, reduced search space for integer programming, and post-processing and solution refinement. This hybrid approach highlights the strengths of both methods.

Based on our review and the existing review papers in university timetabling solutions, it can be concluded that integer programming provides much higher implementation rates than heuristic or meta-heuristic approaches within academic publications or experiments, and integration with machine learning algorithms can achieve a mathematically optimal timetabling solutions.

Many papers in this review explore alternative methods beyond traditional integer and linear programming, heuristic, and meta-heuristic approaches. Future research could incorporate machine learning and AI to enhance university timetabling solutions. Additionally, applying integer programming techniques combined with machine learning algorithms in broader research areas beyond university scheduling could be investigated as future work.

This review is limited within academic publications. The industry-based real-world implementations for university timetabling solutions, which have not been academically published, are not included in this review.

Author Contributions: F.S. contributed to the introduction. X.G. contributed to the existing reviews and methodology. Z.F. contributed to the methodology section. X.G., M.K., S.T. and S.S. contributed to the results sections. X.G. and S.S. contributed to the conclusion section. X.G. and M.K., the lead authors, contributed equally to this work. The remaining authors also made equal contributions. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This review paper does not contain original data, as it is a comprehensive analysis and synthesis of existing literature and research studies. All data sources and references used in this paper are cited within the text and listed in the references section.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Alghamdi, H.; Alsubait, T.; Alhakami, H.; Baz, A. A review of optimization algorithms for university timetable scheduling. *Eng. Technol. Appl. Sci. Res.* **2020**, *10*, 6410–6417. [CrossRef]
2. Daskalaki, S.; Birbas, T.; Housos, E. An integer programming formulation for a case study in university timetabling. *Eur. J. Oper. Res.* **2004**, *153*, 117–135. [CrossRef]
3. Daskalaki, S.; Birbas, T. Efficient solutions for a university timetabling problem through integer programming. *Eur. J. Oper. Res.* **2005**, *160*, 106–120. [CrossRef]
4. Qu, R.; Burke, E.; McCollum, B.; Merlot, L.T.; Lee, S.Y. *A Survey of Search Methodologies and Automated Approaches for Examination Timetabling*; Computer Science Technical Report No. NOTTCS-TR-2006-4; University of Nottingham: Nottingham, UK, 2006.
5. Dimopoulou, M.; Miliotis, P. Implementation of a university course and examination timetabling system. *Eur. J. Oper. Res.* **2001**, *130*, 202–213. [CrossRef]
6. Pillay, N. A review of hyper-heuristics for educational timetabling. *Ann. Oper. Res.* **2016**, *239*, 3–38. [CrossRef]
7. Aziz, N.L.A.; Aizam, N.A.H. A brief review on the features of university course timetabling problem. *AIP Conf. Proc.* **2018**, *2016*, 020001.
8. Stephen, E.A. A critical review on the optimization methods in solving exam timetabling and scheduling. *Int. J. Mech. Eng. Technol.* **2018**, *9*, 416–428.
9. Hosny, M. Metaheuristic approaches for solving university timetabling problems: A review and case studies from Middle Eastern Universities. In Proceedings of the Information Systems and Technologies to Support Learning: Proceedings of EMENA-ISTL 2018 2, Fez, Morocco, 25–27 October 2018; Springer: Cham, Switzerland, 2019; pp. 10–20.
10. Bashab, A.; Ibrahim, A.O.; AbedElgabar, E.E.; Ismail, M.A.; Elsafi, A.; Ahmed, A.; Abraham, A. A systematic mapping study on solving university timetabling problems using meta-heuristic algorithms. *Neural Comput. Appl.* **2020**, *32*, 17397–17432. [CrossRef]
11. Chen, M.C.; Goh, S.L.; Sabar, N.R.; Kendall, G. A survey of university course timetabling problem: Perspectives, trends and opportunities. *IEEE Access* **2021**, *9*, 106515–106529. [CrossRef]
12. Tan, J.S.; Goh, S.L.; Kendall, G.; Sabar, N.R. A survey of the state-of-the-art of optimisation methodologies in school timetabling problems. *Expert Syst. Appl.* **2021**, *165*, 113943. [CrossRef]
13. Zhu, K.; Li, L.D.; Li, M. A survey of computational intelligence in educational timetabling. *Int. J. Mach. Learn. Comput.* **2021**, *11*, 40–47. [CrossRef]
14. Kaur, M.; Saini, S. A review of metaheuristic techniques for solving university course timetabling problem. In Proceedings of the Advances in Information Communication Technology and Computing: Proceedings of AICTC 2019, Bikaner, India, 8–9 November 2019; pp. 19–25.
15. Bashab, A.; Ibrahim, A.O.; Hashem, I.A.T.; Aggarwal, K.; Mukhlif, F.; Ghaleb, F.A.; Abdelmaboud, A. Optimization Techniques in University Timetabling Problem: Constraints, Methodologies, Benchmarks, and Open Issues. *Comput. Mater. Contin.* **2023**, *74*, 6461–6484. [CrossRef]
16. Gu, X.; Blackmore, K. Characterisation of academic journals in the digital age. *Scientometrics* **2017**, *110*, 1333–1350. [CrossRef]
17. Jacsó, P. Google Scholar: The pros and the cons. *Online Inf. Rev.* **2005**, *29*, 208–214. [CrossRef]
18. Aizam, N.A.H.; Caccetta, L. Computational model for timetabling problem. *Numer. Algebr. Control Optim.* **2014**, *4*, 269–285. [CrossRef]
19. Wren, A. Scheduling, timetabling and rostering—A special relationship? In Proceedings of the International Conference on the Practice and Theory of Automated Timetabling, Edinburgh, UK, 29 August–1 September 1995; Springer: Berlin/Heidelberg, Germany, 1995; pp. 46–75.
20. Schaerf, A. A survey of automated timetabling. *Artif. Intell. Rev.* **1999**, *13*, 87–127. [CrossRef]
21. Burke, E.; Jackson, K.; Kingston, J.H.; Weare, R. Automated university timetabling: The state of the art. *Comput. J.* **1997**, *40*, 565–571. [CrossRef]
22. Mirrazavi, S.K.; Mardle, S.J.; Tamiz, M. A two-phase multiple objective approach to university timetabling utilising optimisation and evolutionary solution methodologies. *J. Oper. Res. Soc.* **2003**, *54*, 1155–1166. [CrossRef]

23. Yavuz, S.; Demirci, Z.; Güven, B.; Kuleli, Z.; Yılmazer, B.; Eliiyi, D.T. A Course Timetabling Problem for Classroom Usage Minimization. In Proceedings of the The International Symposium for Production Research, Antalya, Türkiye, 6–8 October 2022; Springer: Cham, Switzerland, 2022; pp. 543–553.
24. Lindahl, M.; Mason, A.J.; Stidsen, T.; Sørensen, M. A strategic view of University timetabling. *Eur. J. Oper. Res.* **2018**, *266*, 35–45. [CrossRef]
25. Lach, G.; Lübbecke, M.E. Optimal university course timetables and the partial transversal polytope. In Proceedings of the Experimental Algorithms: 7th International Workshop, WEA 2008 Provincetown, MA, USA, 30 May–1 June 2008; Proceedings 7; Springer: Berlin/Heidelberg, Germany, 2008; pp. 235–248.
26. Holm, D.S.; Mikkelsen, R.Ø.; Sørensen, M.; Stidsen, T.J. A graph-based MIP formulation of the International Timetabling Competition 2019. *J. Sched.* **2022**, *25*, 405–428. [CrossRef]
27. Al-Yakoob, S.M.; Sherali, H.D. Mathematical programming models and algorithms for a class–faculty assignment problem. *Eur. J. Oper. Res.* **2006**, *173*, 488–507. [CrossRef]
28. Heitmann, H.; Brüggemann, W. Preference-based assignment of university students to multiple teaching groups. *OR Spectr.* **2014**, *36*, 607–629. [CrossRef]
29. Güler, M.G.; Gecici, E. A spreadsheet-based decision support system for examination timetabling. *Turk. J. Electr. Eng. Comput. Sci.* **2020**, *28*, 1584–1598. [CrossRef]
30. Akkoyunlu, E.A. A Linear Algorithm for Computing the Optimum University Timetable. *Comput. J.* **1973**, *16*, 347–350. [CrossRef]
31. Schimmelpfeng, K.; Helber, S. Application of a real-world university-course timetabling model solved by integer programming. *Spectr.* **2007**, *29*, 783–803. [CrossRef]
32. Cateldo, A.; Ferror, J.C.; Miranda, J.; Rey, P.A.; Sauré, A. An integer programming approach to curriculum-based examination timetabling. *Ann. Oper. Res.* **2017**, *258*, 369–393. [CrossRef]
33. IBM CPLEX Optimization Studio—CPLEX Optimizer. Available online: https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-optimizer (accessed on 21 August 2023).
34. Bliek1ú, C.; Bonami, P.; Lodi, A. Solving mixed-integer quadratic programming problems with IBM-CPLEX: A progress report. In Proceedings of the Twenty-Sixth RAMP Symposium, Tokyo, Japan, 16–17 October 2014; pp. 16–17.
35. Arias-Osorio, J.; Mora-Esquivel, A. A solution to the university course timetabling problem using a hybrid method based on genetic algorithms. *Dyna* **2020**, *87*, 47–56. [CrossRef]
36. Zaulir, Z.M.; Aziz, N.L.A.; Aizam, N.A.H. A General Mathematical Model for University Courses Timetabling: Implementation to a Public University in Malaysia. *Malays. J. Fundam. Appl. Sci.* **2022**, *18*, 82–94. [CrossRef]
37. Al-Yakoob, S.M.; Sherali, H.D.; Al-Jazzaf, M. A mixed-integer mathematical modeling approach to exam timetabling. *Comput. Manag. Sci.* **2010**, *7*, 19–46. [CrossRef]
38. Shimazaki, S.; Sakakibara, K.; Matsumoto, T. Iterative optimization techniques using man–machine interaction for university timetabling problems. *SpringerPlus* **2015**, *4*, 1–11. [CrossRef] [PubMed]
39. Gurobi Optimization, LLC. Gurobi Optimization. Available online: https://www.gurobi.com/ (accessed on 21 August 2023).
40. Dimitsas, A.; Nastos, V.; Gogos, C.; Valouxis, C. An exact based approach for the Post Enrollment Course Timetabling Problem. In Proceedings of the 26th Pan-Hellenic Conference on Informatics, Athens, Greece, 25–27 November 2022; pp. 77–82.
41. Mikkelsen, R.Ø.; Holm, D.S. A parallelized matheuristic for the International Timetabling Competition 2019. *J. Sched.* **2022**, *25*, 429–452. [CrossRef]
42. Rappos, E.; Thiémard, E.; Robert, S.; Hêche, J.F. A mixed-integer programming approach for solving university course timetabling problems. *J. Sched.* **2022**, *25*, 391–404. [CrossRef]
43. Urbán Rivero, L.E.; Benítez Escárcega, M.R.; Velasco, J. An Integer Linear Programming Model for a Case Study in Classroom Assignment Problem. *Comput. Y Sist.* **2020**, *24*, 97–104. [CrossRef]
44. Bagger, N.C.F.; Kristiansen, S.; Sørensen, M.; Stidsen, T.R. Flow formulations for curriculum-based course timetabling. *Ann. Oper. Res.* **2019**, *280*, 121–150. [CrossRef]
45. Bagger, N.C.F.; Sørensen, M.; Stidsen, T.R. Benders' decomposition for curriculum-based course timetabling. *Comput. Oper. Res.* **2018**, *91*, 178–189. [CrossRef]
46. Ásgeirsson, E.; Gunnarsdóttir, Þ. An Integer Programming Formulation for the Music School Timetabling Problem. In Proceedings of the 12th International Conference on the Practice and Theory of Automated Timetabling (PATAT-2018), Vienna, Austria, 28–31 August 2018; Volume 2831, pp. 171–182.
47. Phillips, A.E.; Waterer, H.; Ehrgott, M.; Ryan, D.M. Integer programming methods for large-scale practical classroom assignment problems. *Comput. Oper. Res.* **2015**, *53*, 42–53. [CrossRef]
48. Phillips, A.E.; Walker, C.G.; Ehrgott, M.; Ryan, D.M. Integer programming for minimal perturbation problems in university course timetabling. *Ann. Oper. Res.* **2017**, *252*, 283–304. [CrossRef]
49. Lingo. Available online: https://www.lindo.com/ (accessed on 21 August 2023).

50. Bakir, M.A.; Aksop, C. A 0-1 integer programming approach to a university timetabling problem. *Hacet. J. Math. Stat.* **2008**, *37*, 41–55.
51. Malik, B.B.; Nordin, S.Z. Mathematical model for timetabling problem in maximizing the preference level. *AIP Conf. Proc.* **2018**, *1974*, 020037
52. Budiningsih, E.; Rosyidi, C.N.; Jauhari, W.A. An optimization model of undergraduate thesis examination scheduling in department of industrial engineering, Sebelas Maret University. *AIP Conf. Proc.* **2019**, *2097*, 030040.
53. Rosyidi, C.N.; Budiningsih, E.; Jauhari, W.A. Scheduling of undergraduate thesis examination: A case study in Industrial Engineering Department of Universitas Sebelas Maret. *J. Ind. Eng. Int.* **2019**, *15*, 209–221. [CrossRef]
54. Al-Hawari, F.; Al-Ashi, M.; Abawi, F.; Alouneh, S. A practical three-phase ILP approach for solving the examination timetabling problem. *Int. Trans. Oper. Res.* **2020**, *27*, 924–944. [CrossRef]
55. OpenSolver. Available online: https://opensolver.org/ (accessed on 21 August 2023).
56. Marmolejo-Saucedo, J.A.; Rodriguez-Aguilar, R. A timetabling application for the assignment of school classrooms. In Proceedings of the Intelligent Computing and Optimization: Proceedings of the 2nd International Conference on Intelligent Computing and Optimization 2019 (ICO 2019), Koh Samui, Thailand, 3–4 October 2019; Springer: Cham, Switzerland, 2020; pp. 1–10.
57. Prabodanie, R. An integer programming model for a complex university timetabling problem: A case study. *Ind. Eng. Manag. Syst.* **2017**, *16*, 141–153. [CrossRef]
58. Marmolejo, J.; Aranzabal, I.; Sanchez, S.; Tellaeche, K. Development of a tool for university timetabling using an integrated spreadsheet. In Proceedings of the International Conference Practice Theory Auto-Mated Timetabling, Udine, Italy, 23–26 August 2016; Volume 11.
59. Kassa, B.A. Implementing a class-scheduling system at the College of Business and Economics of Bahir Dar University, Ethiopia. *Interfaces* **2015**, *45*, 203–215. [CrossRef]
60. GNU Linear Programming Kit (GLPK). Available online: https://www.gnu.org/software/glpk/ (accessed on 21 August 2023).
61. Arbaoui, T.; Boufflet, J.P.; Moukrim, A. A matheuristic for exam timetabling. *IFAC-PapersOnLine* **2016**, *49*, 1289–1294. [CrossRef]
62. Ribić, S.; Turčinhožić, R.; Muratović-Ribić, A. Modelling constraints in school timetabling using integer linear programming. In Proceedings of the 2015 XXV International Conference on Information, Communication and Automation Technologies (ICAT), Sarajevo, Bosnia and Herzegovina, 29–31 October 2015; pp. 1–6.
63. Bucco, G.B.; Bornia-Poulsen, C.J.; Bandeira, D.L. Development of a linear programming model for the University Course Timetabling Problem. *Gest Ao Produç Ao* **2017**, *24*, 40–49. [CrossRef]
64. Al-Matouq, A.A. A Convex LASSO Framework for Linear Timetabling. *IEEE Access* **2020**, *8*, 167475–167489. [CrossRef]
65. AIMMS Optimization Software. Available online: https://www.aimms.com/ (accessed on 21 August 2023).
66. Cimen, M.; Belbağ, S.; Soysal, M.; Sel, Ç. Invigilators Assignment in Practical Examination Timetabling Problems. *Int. J. Ind. Eng.* **2022**, *29*, 351.
67. MirHassani, S. A computational approach to enhancing course timetabling with integer programming. *Appl. Math. Comput.* **2006**, *175*, 814–822. [CrossRef]
68. General Algebraic Modeling System (GAMS). Available online: https://www.gams.com/ (accessed on 21 August 2023).
69. Suárez-Rodríguez, J.; Piña, J.C.; Malagón-Alvarado, L.; Blanco, V.; Correa, M.; De La Rosa, L.; Lopera, M.; Valderrama, J.; Vega-Mejía, C.A. An Optimization Model for University Course Timetabling. A Colombian Case Study. In Proceedings of the International Conference of Production Research–Americas, Bahía Blanca, Argentina, 9–11 December 2020; Springer: Cham, Switzerland, 2020; pp. 33–46.
70. FICO Xpress Optimization Documentation. Available online: https://www.fico.com/fico-xpress-optimization/docs/latest/overview.html (accessed on 21 August 2023).
71. Feng, X.; Lee, Y.; Moon, I. An integer program and a hybrid genetic algorithm for the university timetabling problem. *Optim. Methods Softw.* **2017**, *32*, 625–649. [CrossRef]
72. Dimopoulou, M.; Miliotis, P. An automated university course timetabling system developed in a distributed environment: A case study. *Eur. J. Oper. Res.* **2004**, *153*, 136–147. [CrossRef]
73. AMPL Modeling Language. Available online: https://ampl.com/ (accessed on 21 August 2023).
74. Oladejo, N.; Abolarinwa, A.; Salawu, S.; Bamiro, O.; Lukman, A.; Bukari, H. Application of optimization principles in classroom allocation using linear programming. *Int. J. Mech. Eng. Technol. (IJMET)* **2019**, *10*, 874–885.
75. Harrabi, O.; Siala, J.C. A New Sum Coloring-based Integer Linear Programming for Solving the University Course Timetabling Problem. In Proceedings of the 2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA), Tunis, Tunisia, 6–8 February 2020; pp. 1–2.
76. Sarin, S.C.; Wang, Y.; Varadarajan, A. A university-timetabling problem and its solution using Benders' partitioning—A case study. *J. Sched.* **2010**, *13*, 131–141. [CrossRef]
77. Celcat. Available online: https://www.celcat.com/ (accessed on 21 August 2023).

78. Abdalla, M.H.; Obit, J.H.; Alfred, R.; Bolongkikit, J. Performance Comparison of Sequential and Cooperative Integer Programming Search Methodologies in Solving Curriculum-Based University Course Timetabling Problems (CB-UCT). In Proceedings of the Computational Science and Technology: 5th ICCST 2018, Kota Kinabalu, Malaysia, 29–30 August 2018; Springer: Singapore, 2019; pp. 145–154.
79. Google OR-Tools CP-SAT Solver Documentation. Available online: https://developers.google.com/optimization/cp/cp_solver (accessed on 21 August 2023).
80. Gora, W.; Lach, G.; Lübbe, J.; Pfeiffer, O.; Zorn, E.; Jeschke, S. Management and optimal distribution of large student numbers. In Proceedings of the IEEE EDUCON 2010 Conference, Madrid, Spain, 14–16 April 2010; pp. 1891–1896.
81. Gogos, C.; Alefragis, P.; Housos, E. An improved multi-staged algorithmic process for the solution of the examination timetabling problem. *Ann. Oper. Res.* **2012**, *194*, 203–221. [CrossRef]
82. Lemos, A.; Melo, F.S.; Monteiro, P.T.; Lynce, I. Room usage optimization in timetabling: A case study at Universidade de Lisboa. *Oper. Res. Perspect.* **2019**, *6*, 100092. [CrossRef]
83. Palma, C.D.; Bornhardt, P. Considering section balance in an integer optimization model for the curriculum-based course timetabling problem. *Mathematics* **2020**, *8*, 1763. [CrossRef]
84. Samiuddin, J.; Haq, M.A. A novel two-stage optimization scheme for solving university class scheduling problem using binary integer linear programming. *Oper. Manag. Res.* **2019**, *12*, 173–181. [CrossRef]
85. Güler, M.G.; Geçici, E.; Köroğlu, T.; Becit, E. A web-based decision support system for examination timetabling. *Expert Syst. Appl.* **2021**, *183*, 115363. [CrossRef]
86. MATLAB linprog. Available online: https://www.mathworks.com/help/optim/ug/linprog.html (accessed on 21 August 2023).
87. Python PuLP. Available online: https://coin-or.github.io/pulp/ (accessed on 21 August 2023).
88. ITC2019. Available online: https://www.itc2019.org/home (accessed on 21 August 2023).
89. ITC2007. Available online: https://www.unitime.org/itc2007/ (accessed on 21 August 2023).
90. Algethami, H.; Laesanklang, W. A mathematical model for course timetabling problem with faculty-course assignment constraints. *IEEE Access* **2021**, *9*, 111666–111682. [CrossRef]
91. Van Bulck, D.; Goossens, D.; Clarner, J.P.; Dimitsas, A.; Fonseca, G.H.; Lamas-Fernandez, C.; Lester, M.M.; Pedersen, J.; Phillips, A.E.; Rosati, R.M. Which algorithm to select in sports timetabling? *Eur. J. Oper. Res.* **2024**, *318*, 575–591. [CrossRef]
92. De Coster, A.; Musliu, N.; Schaerf, A.; Schoisswohl, J.; Smith-Miles, K. Algorithm selection and instance space analysis for curriculum-based course timetabling. *J. Sched.* **2022**, *25*, 35–58. [CrossRef]
93. Chougule, S.; Ghuge, K.; Kote, A.; Ramteke, A. University schedule generator. *AIP Conf. Proc.* **2024**, *3156*, 040002.
94. Feutrier, T.; Veerapen, N.; Kessaci, M.E. Improving the Relevance of Artificial Instances for Curriculum-Based Course Timetabling through Feasibility Prediction. In Proceedings of the Companion Conference on Genetic and Evolutionary Computation, Lisbon, Portugal, 15–19 July 2023; pp. 203–206.
95. Goh, S.L.; Kendall, G.; Sabar, N.R. Simulated annealing with improved reheating and learning for the post enrolment course timetabling problem. *J. Oper. Res. Soc.* **2019**, *70*, 873–888. [CrossRef]
96. Alefragis, P.; Sofos, C. Automated parameter selection of scheduling algorithms using machine learning techniques. In Proceedings of the 21st Pan-Hellenic Conference on Informatics, Larissa, Greece, 28–30 September 2017; pp. 1–5.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.