



Faculty of Computer Science and Information Technology

**Variable Neighborhood Descent and Whale Optimization Algorithm for
Examination Timetabling Problems at Universiti Malaysia Sarawak**

Emily Siew Sing Kiang

**Doctor of Philosophy
2025**

Variable Neighborhood Descent and Whale Optimization Algorithm for
Examination Timetabling Problems at Universiti Malaysia Sarawak

Emily Siew Sing Kiang

A thesis submitted

In fulfillment of the requirements for the degree of Doctor of Philosophy

(Artificial Intelligence)

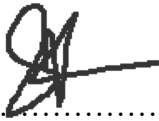
Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2025

DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Malaysia Sarawak. Except where due acknowledgements have been made, the work is that of the author alone. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



.....

Signature

Name: Emily Siew Sing Kiang

Matric No.: 22010236

Faculty of Computer Science and Information Technology

Universiti Malaysia Sarawak

Date : 07 May 2025

ACKNOWLEDGEMENT

I am profoundly grateful to God for blessing me with the strength and grace needed to successfully navigate and complete this challenging journey.

I would like to take this opportunity to express my heartfelt gratitude to my supervisor, Associate Professor Dr. Sze Sah Nah, and my co-supervisor, Associate Professor Dr. Goh Say Leng, for their invaluable guidance, encouragement, and unwavering support throughout my PhD journey.

To my family, thank you for your indirect but meaningful support in helping me see this work through to completion. My deepest thanks go to all my friends for their unwavering support and encouragement.

Lastly, I sincerely appreciate the Faculty of Computer Science and Information Technology, Centre for Graduate Studies, and the management of Universiti Malaysia Sarawak for their invaluable support and guidance in completing my studies.

ABSTRACT

The examination timetabling problem, which involves the allocation of exams to limited resources such as timeslots and rooms, while satisfying constraints. While the existing literature predominantly addresses uncapacitated or capacitated formulations of this problem, this study seeks to bridge the gap between theoretical research and practical application. Specifically, it investigates a real-world examination timetabling problem at Universiti Malaysia Sarawak, proposing a unified framework that incorporates both uncapacitated and capacitated formulations. A mathematical model grounded in real-world data was developed to capture the latest operational requirements. The model employs a two-level structure, where the first level uses standard soft constraints as the objective function to evaluate solution quality, while the second level dynamically adapts to faculty-specific preferences. A constructive algorithm was developed to generate an initial feasible solution, which was subsequently refined using two primary approaches to evaluate their efficiency: Iterative Threshold Pipe Variable Neighborhood Descent (IT-PVND), and a modified Whale Optimization Algorithm (WOA). The IT-PVND adjusts acceptance criteria using a simple iteration parameter, permitting the acceptance of improved solutions and those below a predefined threshold for a set number of iterations to prevent premature convergence to local optima. The original WOA is modified by replacing the equations designed for continuous problem domains with local search methods, enhancing its adaptability to discrete optimization problems. Both IT-PVND and the modified WOA achieved measurable efficiency gains over proprietary software and manual scheduling when applied to real-world data. They also proved competitive against established methods in the literature when validated on the Toronto problem, demonstrating robustness and effectiveness across diverse instances. Furthermore, the research provided practical insights by comparing centralized

and decentralized scheduling strategies, offering universities evidence-based guidance on balancing efficiency and space utilization. Future studies could expand the scope of datasets to validate the centralized framework at the full institutional level rather than only at selected faculties. Moreover, the hybrid WOA introduced here can be further extended — and complemented with other metaheuristics — to address broader domains of scheduling and resource allocation beyond exam timetabling.

Keywords: Examination timetabling, variable neighborhood descent, whale optimization algorithm, optimization, university timetabling

Penurunan Pemboleh Ubah Kejiranan dan Algoritma Pengoptimuman Paus untuk Masalah Penjadualan Peperiksaan di di Universiti Malaysia Sarawak

ABSTRAK

Masalah penjadualan peperiksaan melibatkan pengagihan peperiksaan kepada sumber terhad seperti slot masa dan bilik, sambil memenuhi kekangan-kekangan tertentu. Walaupun kajian sedia ada banyak tertumpu pada formulasi masalah ini sama ada tidak berkapasiti atau berkapasiti, kajian ini berusaha untuk merapatkan jurang di antara penyelidikan teori dan aplikasi praktikal. Secara khusus, kajian ini mengkaji masalah penjadualan peperiksaan dunia sebenar di Universiti Malaysia Sarawak dengan mencadangkan rangka kerja bersepadu yang menggabungkan kedua-dua formulasi tidak berkapasiti dan berkapasiti. Satu model matematik berasaskan data dunia sebenar telah dibangunkan untuk menangkap keperluan operasi terkini. Model ini menggunakan struktur dua peringkat, di mana peringkat pertama menggunakan kekangan lembut standard sebagai fungsi objektif untuk menilai kualiti penyelesaian, manakala peringkat kedua menyesuaikan diri secara dinamik mengikut keutamaan khusus fakulti. Satu algoritma konstruktif telah dibangunkan untuk menjana penyelesaian awal yang boleh dilaksanakan, yang kemudiannya diperhalusi menggunakan dua pendekatan utama bagi menilai kecekapan masing-masing: Penurunan Pemboleh Ubah Kejiranan berasaskan Paip dengan Ambang Iteratif (IT-PVND), dan Algoritma Pengoptimuman Paus (WOA) yang diubah suai. IT-PVND melaraskan kriteria penerimaan menggunakan parameter lelaran yang mudah, membolehkan penerimaan penyelesaian yang lebih baik serta yang berada di bawah ambang yang telah ditetapkan untuk jumlah lelaran tertentu bagi mengelakkan penumpuan awal kepada optima tempatan. Algoritma WOA asal telah diubah suai dengan menggantikan persamaan yang direka untuk domain masalah berterusan dengan kaedah pencarian setempat, meningkatkan

kebolehadaptasiannya terhadap masalah pengoptimuman diskret. Kedua-dua IT-PVND dan WOA yang diubah suai telah mencapai peningkatan kecekapan yang ketara berbanding perisian proprietari dan penjadualan manual apabila digunakan pada data dunia sebenar. Ia juga terbukti berdaya saing berbanding kaedah sedia ada dalam literatur apabila disahkan pada masalah Toronto, menunjukkan keteguhan dan keberkesanan merentasi pelbagai situasi. Tambahan pula, kajian ini memberikan pandangan praktikal dengan membandingkan strategi penjadualan berpusat dan terdesentralisasi, sekali gus menawarkan panduan berasaskan bukti kepada universiti dalam mengimbangi kecekapan dan penggunaan ruang. Kajian masa hadapan boleh memperluaskan skop set data untuk mengesahkan rangka kerja berpusat di peringkat institusi penuh dan bukan hanya pada fakulti terpilih. Selain itu, WOA hibrid yang diperkenalkan di sini boleh dikembangkan lagi — serta dilengkapi dengan metaheuristik lain — untuk menangani domain penjadualan dan peruntukan sumber yang lebih luas melangkaui penjadualan peperiksaan.

Kata kunci: *Penjadualan peperiksaan, penurunan pemboleh ubah kejituan, algoritma pengoptimuman paus, pengoptimuman, penjadualan universiti*

TABLE OF CONTENTS

	Page
DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
ABSTRAK	v
TABLE OF CONTENTS	vii
LIST OF TABLES	xiii
LIST OF FIGURES	xv
LIST OF ABBREVIATIONS	xviii
CHAPTER 1: INTRODUCTION	1
1.1 Background	1
1.2 Problem Statement	3
1.3 Objectives	5
1.4 Research Scope	6
1.5 Research Significance	7
1.6 Thesis Overview	9
CHAPTER 2: LITERATURE REVIEW	11
2.1 Overview	11

2.2	Examination Timetabling Variants	12
2.2.1	Uncapacitated Variant	12
2.2.2	Capacitated Variant	14
2.2.2.1	Room-Capacitated	14
2.2.2.2	Slot-Capacitated	16
2.3	Real-World ETP Constraints	18
2.4	Solution Approaches in ETP	20
2.4.1	Mathematical Optimization	21
2.4.2	Heuristics	24
2.4.3	Metaheuristics	25
2.4.3.1	Population-based	25
2.4.3.2	Single-solution-based	27
2.4.4	Hyper-heuristics	30
2.4.5	Hybrid Approaches	31
2.5	Current Trends of Solution Approaches	33
2.6	State-of-the-Art Methods for Benchmark Problems	40
2.6.1	Toronto Benchmark	41
2.6.2	ITC 2007 Track 1 Benchmark	42
2.6.3	Yeditepe Benchmark	43
2.7	Metaheuristic Methods Considered for Application to ETP	44

2.7.1	Variable Neighborhood Descent	45
2.7.2	Whale Optimization Algorithm	47
2.7.2.1	Exploitation Phase – Encircling Prey and Spiral Updating Position	51
2.7.2.2	Exploration Phase – Searching for Prey	53
2.7.2.3	Original WOA	54
2.8	Research Gaps and Opportunities in ETP	55
2.9	Conclusion	59
CHAPTER 3: A CASE STUDY ON THE UNIMAS EXAMINATION TIMETABLING PROBLEM: MODELING AND DATA SPECIFICATION		61
3.1	Overview	61
3.2	UNIMAS Background and Exam Timetabling Process	61
3.3	Final Semester Examination Timetabling Flow	63
3.4	Problem Formulation	65
3.4.1	Hard Constraints	68
3.4.2	Soft Constraints	69
3.4.3	Objective Functions	71
3.4.4	Extended Soft Constraints	72
3.5	Datasets	72
3.5.1	Semester 1 of 2022/2023	77
3.5.2	Semester 2 of 2023/2024	78

3.5.3	Benchmark Dataset	78
3.6	Research Methodology	79
3.7	Conclusion	81
CHAPTER 4: VARIABLE NEIGHBORHOOD DESCENT APPROACH FOR		
UNIMAS EXAM TIMETABLING PROBLEM		82
4.1	Overview	82
4.2	Preprocessing Stage	83
4.3	Constructive Algorithm	84
4.4	Variable Neighborhood Descent	86
4.4.1	Neighborhood Structures	86
4.4.2	Variable Neighborhood Descent Algorithm	90
4.4.3	Room Split Balancer	94
4.5	Experimental Result	95
4.5.1	Experimental Setup	95
4.5.2	Comparison of PVND with Proprietary Software and Manual Method	97
4.5.3	Extended Soft Constraints	99
4.5.3.1	Shorten Exam Session	99
4.5.3.2	Exam Room Sharing	101
4.5.3.3	Balance Room Allocation for Split Exam	102
4.6	Statistical Analysis of Neighborhood Structures	106

4.7	Comparative Analysis of Centralized and Decentralized Timetabling	113
4.7.1	Centralized vs Decentralized without Reallocation	116
4.7.2	Centralized vs Decentralized with Reallocation	117
4.8	Comparative Analysis of PVND, BVND, and IT-PVND	119
4.8.1	Iterative Threshold based Pipe Variable Neighborhood Descent	119
4.8.2	Basic Sequential Variable Neighborhood Descent	121
4.8.3	Comparison of PVND, BVND, and IT-PVND	122
4.9	Conclusion	124
CHAPTER 5: DISCRETE WHALE OPTIMIZATION ALGORITHM FOR EXAMINATION TIMETABLING		127
5.1	Introduction	127
5.2	Modified Discrete WOA Algorithms	128
5.3	Experiment Results for WOA Models	137
5.4	Conclusion	141
CHAPTER 6: ADDRESSING TORONTO BENCHMARK PROBLEM		143
6.1	Introduction	143
6.1.1	Neighborhood Structure	144
6.1.2	Perturbation Procedures	146
6.2	Experiment Setup	148
6.3	Results and Discussion	149
6.3.1	Comparison of Results with Population-Based Methods	150

6.3.2	Comparison of Results with Other Metaheuristic Methods	153
6.4	Conclusion	157
CHAPTER 7: CONCLUSION		159
7.1	Research Summary	159
7.2	Research Contributions	162
7.2.1	Contributions to Theory	164
7.2.2	Contributions to Applications	164
7.3	Future Work	165
REFERENCES		167
APPENDICES		186

LIST OF TABLES

	Page
Table 2.1 The Characteristics of Toronto Datasets	13
Table 2.2 The Characteristics of ITC 2007 Exam Track Datasets	16
Table 2.3 The Characteristics of Yeditepe Datasets	17
Table 2.4 Methodologies and Datasets in ETP (2015 – 2024)	34
Table 2.5 Counts of Categories across Datasets	40
Table 2.6 Best Scores of State-of-the-Art Methods for Toronto Benchmark	41
Table 2.7 Best Scores of State-of-the-Art Methods for ITC 2007 Exam Track	42
Table 2.8 Best Scores of State-of-the-Art Methods for Yeditepe Benchmark	44
Table 2.9 Symbols for WOA Equations	50
Table 3.1 Notations Employed in for the Problem Formulation of UNIMAS ETP	66
Table 3.2 The Characteristics of Problem Instances from Real-World Dataset	74
Table 4.1 Improvement Pattern for Real-World Datasets	96
Table 4.2 Comparison Results for Studied Datasets	98
Table 4.3 Comparison Timeslots Used and Runtime for Studied Datasets	100
Table 4.4 Result on Reduced Timeslot and Objective Value Comparison	101
Table 4.5 Comparison of JFI with and without Balancing	103
Table 4.6 Descriptive Statistics for Studied Instances	107
Table 4.7 Tests of Normality	109

Table 4.8	Test of Homogeneity	110
Table 4.9	Dunnett T3 Post Hoc for FEB23 and FCSIT24	111
Table 4.10	Turkey HSD Post Hoc for FCSHD23 and FEB24	112
Table 4.11	Comparison of Centralized and Decentralized without Reallocation	117
Table 4.12	Comparison of Centralized and Decentralized with Reallocation	118
Table 4.13	Descriptive Result for PVND, BVND, and IT-PVND	123
Table 5.1	Results Obtained from WOARB-1, WOARB-2, and WOAIB	138
Table 5.2	Results of the Comparisons Among the Five Algorithms	140
Table 6.1	Statistical Summary of Results from IT-PVND and WOAIB	149
Table 6.2	Comparisons to Population-Based Methods	151
Table 6.3	Compared Population-Based Methods	152
Table 6.4	Rank the Proposed Methods against Population-Based Methods	152
Table 6.5	Comparisons to Other Metaheuristic Methods	154
Table 6.6	Compared Metaheuristic Methods	155
Table 6.7	Rank the Proposed Methods against Other Metaheuristic Methods	155

LIST OF FIGURES

	Page
Figure 2.1 Real-world ETP Constraints	19
Figure 2.2 Categories of ETP Solution Methodologies	37
Figure 2.3 Distribution of the ETP Studies by Publication Year	38
Figure 2.4: Distribution of the ETP Studies by Categories	38
Figure 2.5 (a) Distribution of Datasets (b) Distribution of Categories by Datasets	39
Figure 2.6 Pseudocode of Sequential VND	47
Figure 2.7 Pseudocode of Original WOA	54
Figure 3.1 Flowchart of Final Examination Timetable Preparation	65
Figure 3.2 Network Representation for FEB23	75
Figure 3.3 Network Representation for FCSHD24	76
Figure 3.4 Network Representation for FEB24	76
Figure 3.5 Network Representation for FCSIT24	77
Figure 4.1 Constructive Algorithm	85
Figure 4.2 Execution of the Move Exam with Same Rooms	87
Figure 4.3 Execution of the Move Exam with Reduced Room	87
Figure 4.4 Execution of the Swap Exams with Same Room	87
Figure 4.5 Execution of Swap Exams with Reduced Room	88
Figure 4.6 Execution of Kick Exam with Same Room	88

Figure 4.7	Execution of Kick Exam with Reduced Room	88
Figure 4.8	Execution of Shift Room	89
Figure 4.9	Execution of Compact Room	89
Figure 4.10	Execution of Downsize Room with Same Number of Rooms	90
Figure 4.11	Execution of Downsize Room with Reduced Number of Rooms	90
Figure 4.12	Procedure of PVND	93
Figure 4.13	Procedure of the Local Search	93
Figure 4.14	Procedure of the Balancer	95
Figure 4.15	Convergence Curves of PVND for Real-World Datasets	97
Figure 4.16	Shared vs. Non-Shared Room Settings	102
Figure 4.17	Balanced vs. Unbalanced Room Split for FEB23	104
Figure 4.18	Balanced vs. Unbalanced Room Split for FCSHD23	105
Figure 4.19	Balanced vs. Unbalanced Room Split for FEB24	105
Figure 4.20	Balanced vs. Unbalanced Room Split for FCSIT24	106
Figure 4.21	Comparison of Room Type Occupancy	115
Figure 4.22	Procedure of the IT-PVND	120
Figure 4.23	Local Search using Threshold Acceptance Criterion	121
Figure 4.24	Procedure of BVND	122
Figure 5.1	Procedure of the First Descent Method	129
Figure 5.2	Procedure of the First Descent below Threshold Value Method	129

Figure 5.3	Procedure of the PVND (Hansen et al., 2017)	130
Figure 5.4	Procedure of the WOARB-1	132
Figure 5.5	The Framework of the WOARB-1 Algorithm	133
Figure 5.6	Procedure of the WOARB-2	134
Figure 5.7	The Framework of the WOARB-2 Algorithm	135
Figure 5.8	Procedure of the WOAIB	136
Figure 5.9	The Framework of the WOAIB Algorithm	137
Figure 5.10	Box Plot using the Objective Function Value for Instances	141
Figure 6.1	Shaking Procedure	147
Figure 6.2	Probabilistic Perturbation Procedure	147

LIST OF ABBREVIATIONS

CGS	Centre for Graduate Studies
UNIMAS	Universiti Malaysia Sarawak
VND	Variable Neighborhood Descent
WOA	Whale Optimization Algorithm
ETP	Examination Timetabling Problem
PVND	Pipe Variable Neighborhood Descent
IT-PVND	Iterative Threshold with Pipe Variable Neighborhood Descent

CHAPTER 1

INTRODUCTION

1.1 Background

Scheduling spans various domains: education, sports, transportation, nurse scheduling, and more. This broad applicability leads to a dispersed focus in research, where studies tend to be problem-specific. Scheduling involves allocating resources to objects in space-time while adhering to constraints to minimize the total cost of the utilized resources (Wren, 1996). Timetabling, a specialized form of scheduling, involves allocating resources to objects within a space-time framework to satisfy a set of organizational objectives as closely as possible, referred to as soft constraints, which vary across institutions based on managerial decisions.

Among the vast array of scheduling challenges, educational timetabling is one of the most extensively explored topics due to the demanding nature of periodically preparing schedules in educational institutions, whether on a semester, annual, or seasonal basis. In certain instances, the involvement of multiple individuals in the scheduling process necessitates dedicating thousands of hours to this endeavor. The educational timetabling problem can be broadly categorized into three primary categories: school timetabling, course timetabling, and examination timetabling (Schaerf, 1999).

This work focuses on the examination timetabling problem (ETP). The ETP is one of numerous challenging real-world combinatorial optimization problems, proven NP-hard (Cooper & Kingston, 1996). It has attracted researchers for over five decades, sparking numerous innovative approaches to tackle the conundrum. ETP aims to efficiently assign

fixed examinations to designated periods and/or rooms subject to predefined hard and soft constraints (Qu et al., 2009). Satisfying hard constraints is mandatory to ensure the feasibility of the schedule, whereas adhering to soft constraints, although not mandatory, enhances the quality of the timetable. The primary goal of a scheduler is to construct a feasible timetable that achieves a high standard of quality.

The ETP can be categorized as uncapacitated or capacitated, depending on whether room capacity is considered (Lei et al., 2018). In the uncapacitated variant, room capacities are disregarded, whereas the capacitated variant must account for hard constraints. Specifically, the number of students assigned to a given room during a scheduled period must not exceed the room's capacity (Pillay & Banzhaf, 2009; Leite et al., 2019). Due to the impact of COVID-19 and the implementation of social distancing measures, some institutions have transitioned specific courses from physical to online examinations (Zhu et al., 2022), resulting in changes to the timetabling constraints.

This study investigates real-world ETP at UNIMAS, incorporating practical constraints considerations. Exam timetabling operates under a decentralized approach, with each faculty independently scheduling its examinations. While a decentralized approach allows each faculty to tailor the examination schedule to its specific needs, it also introduces complexities such as exam durations and resource allocation variations. This work aims to develop a unified strategy that addresses the diverse requirements of different faculties at UNIMAS. This approach considers variations in events, resources, student numbers, course offerings, and examination room availability, optimizing exam scheduling while accommodating the unique needs of each faculty.

Metaheuristic approaches have gained significant attention for solving combinatorial optimization problems such as timetabling and scheduling. The Whale Optimization Algorithm (WOA), inspired by the bubble-net feeding strategy of humpback whales, is widely used for complex optimization tasks. Its ability to balance exploration and exploitation makes it particularly suitable for examination timetabling, where numerous constraints and large search spaces pose significant challenges. WOA's adaptive mechanism ensures effective convergence toward high-quality solutions while preserving simplicity. This study applies WOA to real-world examination timetabling, leveraging its efficiency and adaptability to constraint-rich environments.

Numerous mechanisms have been implemented in literature to address ETPs (Alzaqebah & Abdullah, 2015; Leite et al., 2018; Tilahun, 2019; Mandal et al., 2020; Al-Betar, 2020; Bellio et al., 2021). The technique that motivates the adoption of an effective local search neighborhood structure for the proposed algorithm within the examination timetabling search space is derived from these studies. A modified WOA that harnesses this local search neighborhood structure is proposed to effectively address feasibility and optimality problems. By better representing real-world complexities in the examination timetabling process, the proposed approaches aim to be more practical and preferable in real-world scenarios compared to existing models prevalent in scientific literature.

1.2 Problem Statement

This study focuses on a real-world ETP at UNIMAS, addressing the following issues:

- i. Lack of a Mathematical Model for Measuring Timetable Efficiency

As the number of students and courses has increased in recent years, while room availability and resources remain unchanged, exam scheduling has become more complex due to constraints such as preventing exam clashes, ensuring fair spacing between exams, and optimizing room utilization. These challenges can lead to student fatigue, underutilization of rooms, and suboptimal exam timetables. A mathematical model is needed to formalize the exam scheduling process; however, there is currently no comprehensive mathematical model available for measuring the quality of the exam schedule at UNIMAS.

ii. Limitations of Conventional Scheduling Methods in Exam Timetabling

The current practice of exam timetabling for core subjects at UNIMAS is decentralized, with faculty-specific exam timetabling. Traditionally, faculty exam timetabling follows a capacitated variant, which involves assigning exams to specific time slots and rooms. However, due to the post-pandemic shift and the adoption of blended learning, some faculties' exam timetabling now incorporates both variants: the uncapacitated variant for online exams, which are assigned to time slots, and the capacitated variant for physical exams, which are assigned to both time slots and rooms. This has become a challenge, requiring the need to address both variants together, as most studies have typically tackled them separately.

iii. Constraints Addressing Faculty-Specific Needs

Another issue lies in the utilization of room resources. Due to the limited availability of large and medium-sized rooms, it is necessary to split exams across multiple rooms. Such a split-room arrangement presents two challenges: first, the potential allocation of rooms across different buildings or areas, which complicates exam administration; and second, an imbalance in the split, which can result in confusion among students assigned to rooms with

fewer candidates, particularly concerning the exam location. This necessitates a solution to ensure a balanced split, with all split exam rooms for a given exam located within the same building or area.

Moreover, faculties may have conflicting room-sharing preferences: some faculties prefer sharing rooms to optimize space and resources, while others prioritize independence. Therefore, in addition to complying with university-level soft constraints, the timetabling must also account for faculty preferences. These include (a) minimizing the overall examination period, (b) accommodating room-sharing preferences, and (c) balancing the students across split rooms.

1.3 Objectives

The objectives of this thesis are defined as follows:

- i. To formulate a mathematical model for the UNIMAS examination timetabling problem.
- ii. To develop a unified approach that leverages Variable Neighborhood Descent and Whale Optimization Algorithm to integrate both uncapacitated and capacitated variants of the real-world faculty exam timetabling problem.
- iii. To conduct a statistical analysis of the proposed algorithms and to compare their performance against proprietary software and a manual method, validated using a benchmark problem.

1.4 Research Scope

The scope of the research is outlined from three distinct perspectives: data, algorithms, and solution evaluation.

The first perspective pertains to the data. This study addresses the faculty ETP within a real-world context at UNIMAS. The examination modes offered in UNIMAS are categorized into two types: online exams conducted via the eLEAP platform and physical exams held in examination rooms or halls. The exam schedule is divided into two slots per day: morning and afternoon. The courses are categorized into four types: Generic Language Course Examination, General Subjects Examination, University Elective Courses, and Core Courses. Faculty timetabling will only involve Core Courses, while the other types will be centrally scheduled in reserved time slots.

To address the diverse faculty preferences and ensure the robustness of the analysis, the study utilized four real-world datasets: two from the Faculty of Economics and Business, one from the Faculty of Cognitive Sciences and Human Development, and one from the Faculty of Computer Science and Information Technology. Additionally, to evaluate the validity and efficiency of the proposed algorithms, the Toronto (Carter) benchmark dataset was employed as a testbed. This dataset was selected due to its high complexity, the diversity of instances it represents, its widespread use in research over the past decade, and its resemblance to the subset of constraints found in the real-world timetabling problem examined in this study.

The second perspective pertains to the algorithmic approach. The ETP is classified as an NP-hard problem, implying that it cannot be solved exactly within polynomial time due to the exponential growth of both the problem size and its inherent complexity. While exact

algorithms are guaranteed to yield optimal solutions, they are only suitable for problems of a relatively small scale. Consequently, this study adopts multi-stage algorithms, as they offer the potential for obtaining sufficiently good solutions within an acceptable computational time. In the first stage, a constructive algorithm was developed to identify a feasible solution. In the second stage, two optimization algorithms, the VND approach and the modified WOA, were employed to enhance the solution, with their efficiency compared.

The third perspective of this study focuses on solution evaluation. Since examination timetabling is managed independently by each faculty, individual preferences may vary while still adhering to standard constraints. However, only the standard soft constraints are evaluated in the objective function, while faculty-specific preferences from selected faculties are also considered. To evaluate the effectiveness of the proposed algorithms, the generated timetables were benchmarked against the officially released timetable using objective function values. Additionally, statistical analysis was conducted to evaluate the performance of the proposed algorithms and the neighborhood structures implemented in this study. Further comparisons were made between the generated timetables and those produced by other VND variations and alternative WOA models. The Toronto benchmark dataset was used to further assess the effectiveness of the proposed algorithms.

1.5 Research Significance

The university ETP plays a crucial role in academic administration, directly impacting students, faculty, and institutional efficiency. Unlike synthetic benchmark datasets or other real-world timetabling problems that typically handle them separately, focusing on either uncapacitated or capacitated variants alone, this study unifies the scheduling of both uncapacitated and capacitated examinations within a single framework.

Considering the diversity of exam room resources—where medium- and large-sized shared rooms are available only in a few randomly distributed slots, while small-sized rooms are accessible across all time slots—strategic allocation is crucial, as suboptimal arrangements may lead to delays in the overall examination schedule. The limited availability of medium- and large-sized shared spaces may lead to greater fragmentation of large exam groups, forcing them to be split across multiple rooms. Additionally, it makes prioritizing large exams in earlier time slots increasingly challenging.

Exam timetabling constraints are typically considered either at the university or faculty level. In contrast, this study adopts a two-fold approach: university-level constraints are used as standard soft constraints, while also incorporating faculty-level extended constraints to ensure that the generated timetables better align with the distinct needs of each faculty.

This research seeks to tackle the aforementioned challenges by implementing a well-formulated approach and benchmarking its results against proprietary software and manual scheduling methods. The findings offer valuable insights into the effectiveness of the proposed solution and highlight its potential benefits in comparison to existing methods. By incorporating real-world considerations, the research enhances the practicality and applicability of the approach, making it more relevant to actual university ETP.

Through this research, by implementing a properly formulated approach to address the aforementioned problems and challenges, benchmarking results are compared against proprietary software and manual scheduling methods. The research provides valuable insights into its effectiveness and potential advantages over current solutions. These real-world considerations enhance the practicality and applicability of the proposed approach,

making it more relevant to actual university ETP and advancing research in university timetabling.

1.6 Thesis Overview

This thesis comprises seven chapters, each addressing distinct aspects of the research. The introductory chapter outlines the study's background, problem statements, and objectives. The structure of the remaining chapters is outlined as follows:

Chapter 2 examines the ETP by presenting its variants, datasets, and constraints as reported in scientific literature over the past decade. It also reviews current solution methodologies and emerging trends in examination timetabling.

Chapter 3 details the constraints and objective functions, provides a mathematical model of the problem and describes the datasets used throughout the thesis.

Chapter 4 introduces a pipe-based VND, a metaheuristic developed that incorporates a local search process to refine initial solutions produced by a constructive algorithm. The proposed VND employs multi-neighborhood structures to achieve higher-quality solutions compared to UNIMAS proprietary software and manual methods. Additionally, an iterative threshold with pipe-based variable neighborhood descent (IT-PVND) is explored, and statistical analyses are conducted to evaluate neighborhood structures. The chapter also compares centralized and decentralized timetabling approaches regarding their effect on soft constraints optimization.

In Chapter 5, a recent popular metaheuristic, WOA, is modified and designed to adapt its application from a continuous domain to a discrete domain, addressing the same ETP as in Chapter 4. This modified method replaces the standard equations of the original WOA

with local search methods to balance exploration and exploitation. Three WOA models have been evaluated and compared among themselves and with other VND approaches, offering valuable insights into the performance of the various methods.

Chapter 6 discusses the implementation of the IT-PVND and modified WOA methods on the most widely studied Toronto benchmark datasets. The experiments evaluate the applicability of these methods to similar benchmark problems. The results demonstrate that the proposed methods yield competitive outcomes when compared to literature benchmarks, albeit with more extended computational time requirements.

Chapter 7 summarizes the findings and suggests possible avenues for future research in the field of examination timetabling.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

The work presented in this chapter has been published in the IEEE Open Access (Siew et al., 2024). This chapter is structured into nine sections. This section offers a summary of the background on examination timetabling. Section 2.2 explores examination timetabling variants, and Section 2.3 addresses real-world ETP constraints. Section 2.4 presents solution approaches to the problem, Section 2.5 discusses current trends of solution approaches, Section 2.6 elaborates on the state-of-the-art methods and outcomes of benchmark problems, and Section 2.7 concludes the chapter.

Timetabling, a combinatorial optimization problem studied extensively for over four decades in fields such as operations research, constitutes a prominent example of challenging combinatorial optimization problems and involves the mathematical pursuit of finding optimal arrangements, groupings, orderings, or selections of finite discrete objects (Lawler, 2001). The timetabling problem, as defined by Burke et al. (2004), is described as “A timetabling problem is a problem with four parameters: T , a finite set of times; R , a finite set of resources; M , a finite set of meetings; and C , a finite set of constraints. The problem is to assign times and resources to the meetings to satisfy the constraints as far as possible”.

Educational timetabling can be broadly categorized into three primary types: examination timetabling, school timetabling, and course timetabling (Schaerf, 1999). Each involves assigning events (e.g., classes, exams, lectures, tutorials, labs) to limited resources (e.g., periods, classrooms, rooms, and halls) while adhering to a predefined set of constraints.

Although certain similarities may exist, successfully solving one type of problem using a specific method does not ensure its effectiveness for another. Therefore, each problem type has been approached independently, with individual instances often varying significantly in dimensions and constraints.

Examination timetabling problem (ETP) is classified as NP-complete (Karp, 1972; Cooper & Kingston, 1996), indicating that no known polynomial-time exact algorithm exists to find an optimal solution for these problems. ETP was defined by Qu et al. (2009) as “assigning a set of exams into a limited number of timeslots (periods) and rooms (of certain capacity) subject to a set of constraints”. The ETP is classified into two main variants: uncapacitated and capacitated. The uncapacitated variant imposes no room capacity constraints, assuming that sufficient physical space will always exist for all students. In contrast, the capacitated variant considers room/ slot capacity per period.

2.2 Examination Timetabling Variants

The constraints, objective functions, and datasets for each exam timetabling variant are detailed in the following subsections.

2.2.1 Uncapacitated Variant

The uncapacitated variant was introduced by Carter et al. (1996) and utilises the Toronto (Carter) dataset, which comprises 13 real-world instances primarily from Canadian universities. Table 2.1 presents the characteristics of the Toronto version 1 dataset.

- i. Constraints: Consider a hard constraint ensuring no conflicting exams and a soft constraint aimed at maximizing the spread of students.

- ii. Objective functions: Minimize total proximity costs per student depending on the time interval between conflicting exams: 16 (no gap), 8 (1 interval), 4 (2 intervals), 2 (3 intervals), and 1 (4 intervals).
- iii. Datasets: The data is accessible through <http://www.cs.nott.ac.uk/~pszrq/data.htm>. Data is provided in plain text across two files: one listing exams and the other detailing student enrollments. The primary input is the Boolean enrollment matrix, which records each student-exam pair.
- iv. Description: The uncapacitated variant is a simplified model that intentionally excludes all side constraints for broader applicability across cases, as noted by Carter et al. (1996). Despite its simplicity, it remains a focus of research due to its challenging benchmarks, which, to the best of current knowledge, have not been solved to proven optimality to date.

Table 2.1: The Characteristics of Toronto Datasets

Instances	Exams	Students	Enrolments	Conflict Density	Periods
CAR91	682	16925	56877	0.13	35
CAR92	543	18419	55522	0.14	32
EAR83	190	1125	8109	0.27	24
HEC92	81	2823	10632	0.42	18
KFU93	461	5349	25118	0.06	20
LSE91	381	2726	10918	0.06	18
PUR93	2419	30032	120681	0.03	42
RYE92	486	11483	45051	0.07	23

Table 2.1 continued

STA83	139	611	5751	0.14	13
TRE92	261	4360	14901	0.18	23
UTA92	622	21266	58979	0.13	35
UTE92	184	2750	11793	0.08	10
YOR83	181	941	6034	0.29	21

2.2.2 Capacitated Variant

Capacitated variants can be further divided into room-capacitated and slot-capacitated types. Specifically, the room-capacitated ETP limits the number of students that can be accommodated in a room during a given period. At the same time, the slot-capacitated ETP applies to capacity constraints for each period.

2.2.2.1 Room-Capacitated

The 2nd International Timetabling Competition (ITC 2007) Track 1 exam timetabling provided 12 instances from British universities, in accordance with the competition regulations. Table 2.2 presents the characteristics of the ITC 2007 exam track dataset.

- i. Constraints: There are five hard constraints—exam conflicts, room capacity, period length, precedence, and room exclusivity—and seven soft constraints: consecutive exams, mixed durations, undesired periods, spread, frontloading, undesired rooms, and two exams in a day.

- ii. Objective functions: Minimize the penalty for violating the seven terms related to the soft constraints. The weights assigned to soft constraints differ across cases and are specified within each instance's input file.
- iii. Datasets: The data is accessible through <https://www.unitime.org/ITC2007/>. The file format, created for the ITC 2007 competition, is a text-only single file. The time frame is divided into periods corresponding to a specific day. These periods have defined lengths (in minutes) and may incur penalties if exams are scheduled during them. Similarly, rooms have capacities and may incur penalties if they are overused. Each exam has the required duration, is compatible only with periods of sufficient length, and may need a dedicated room or can be shared with others. Additionally, a set of enrolled students is provided for each exam, and some exams have precedence rules specifying their relative scheduling constraints.
- iv. Description: This variant incorporates novel real-world features, particularly those relevant to British universities. They include objective weights within the instance rather than relying on fixed values across all scenarios. As McCollum et al. (2007) noted, this approach acknowledges that different institutions have varying priorities, encouraging the creation of robust solvers instead of those overly tuned to a single set of weights.

Table 2.2: The Characteristics of ITC 2007 Exam Track Datasets

Instances	Exams	Students	Enrolments	Conflict Density	Periods	Rooms	Capacities
EXAM1	607	7891	32380	0.05	54	7	802
EXAM2	870	12743	37379	0.01	40	49	4076
EXAM3	934	16439	61150	0.03	36	48	5212
EXAM4	273	5045	21740	0.15	21	1	1200
EXAM5	1018	9253	34196	0.01	42	3	2395
EXAM6	242	7909	18466	0.06	16	8	2050
EXAM7	1096	14676	45493	0.02	80	15	2530
EXAM8	598	7718	31374	0.05	80	8	922
EXAM9	169	655	2532	0.08	25	3	170
EXAM10	214	1577	7853	0.05	32	48	1914
EXAM11	934	16439	61150	0.03	26	40	4924
EXAM12	78	1653	3685	0.18	12	50	1525

2.2.2.2 Slot-Capacitated

The Faculty of Engineering at Yeditepe University dataset encompasses real-world problems spanning eight semesters over three years (Özcan & Ersoy, 2005). These instances are comparatively small, containing no more than 210 exams, and are based on a more straightforward formulation than ITC 2007.

- i. Constraints: There are two hard constraints —exam conflicts and slot capacity —and one soft constraint —exam spread.

- ii. Objective functions: Minimize instances of students having two consecutive exams on the same day.
- iii. Datasets: Table 2.3 displays the characteristics of the Yeditepe dataset. The data is accessible through <https://people.cs.nott.ac.uk/pszajp/timetabling/exam/yeditepe-data.zip>. Parkes and Özcan (2010) have converted from their initial format to the ITC 2007 format, but the focus is on maximum seating capacity for each period instead of room capacity.
- iv. Description: These instances have been studied less than the Toronto and ITC 2007 exam track benchmarks because they pose fewer challenges.

Table 2.3: The Characteristics of Yeditepe Datasets

Instances	Exams	Students	Enrolments	Conflict Density	Periods	Capacities
yue20011	126	559	3486	0.18	6	450
yue20012	141	591	3708	0.18	6	450
yue20013	26	234	447	0.25	2	150
yue20021	162	826	5755	0.18	7	550
yue20022	182	869	5687	0.17	7	550
yue20023	38	420	790	0.2	2	150
yue20031	174	1125	6714	0.15	6	550
yue20032	210	1185	6833	0.14	6	550

The benchmark instances for the first two variants, Toronto and ITC 2007, remain highly challenging even after being published over 20 years ago. These benchmark instances

remain widely and actively studied, with some emphasizing narrowing the optimality gap and minimizing the time required to achieve feasible solutions. At the same time, others also aim to reduce the penalty for constraint violations. The domain of examination timetabling is characterized by continuous evolution, driving rapid progress in both theoretical frameworks and practical implementations. With the availability of public benchmark datasets, researchers are striving to contextualize the timetabling problem within real-world scenarios. Over the last decade, there has been an increased focus on studies that explore real-case scenarios with diverse constraints from various educational institutions, as detailed in the following section.

2.3 Real-World ETP Constraints

Figure 2.1 categorizes the primary constraints into constraints related to exams, periods, rooms, and invigilators, derived from real-world problems over the last decade. It outlines their occurrence across various real-world scenarios documented in the literature. These constraints are further categorized as either hard or soft, depending on the specific requirements or preferences of the respective institutions. Invigilator-related constraints have garnered comparatively limited attention. As far as is known, such constraints have yet to be integrated into any benchmark dataset. This omission can be attributed to the diverse and highly variable nature of these constraints across universities, reflecting each institution's unique challenges and preferences.

The top-priority hard constraints in Figure 2.1 primarily revolve around avoiding exam clashes, restricting students to taking only one exam at a time, enforcing room capacity limits, and allowing one exam to be held per room at any given time. The high frequency of

room-related constraints for real-world problems emphasizes the predominance of capacitated variants and underscores the vital role of room-related constraints in research.

Constraints
Exam related constraints
Each exam is allocated a clash-free timeslot. (Kasm et al., 2018; Zhu et al., 2022; Carlsson et al., 2023; Bassimir and Wanka, 2024; Modirghorasani and Hoseinpour, 2024; Ozturk et al., 2024)
Only one exam can be taken by a student at any given time. (Kahar and Kendall, 2015; Cavdur and Kose, 2016; Hawari et al., 2020; Güler and Geçici, 2020; Zhu et al., 2022; Bazari et al., 2023; Modirghorasani and Hoseinpour, 2024; Ozturk et al., 2024)
Each course's exams should be scheduled only once and within a single time slot. (Woumans et al., 2016; Cavdur and Kose, 2016; Cataldo et al., 2017; Keskin et al., 2018; Hawari et al., 2020; Güler et al., 2021; Godwin, 2022; Bazari et al., 2023; Bassimir and Wanka, 2024)
Sets of exams are mandated to be held at the same timeslot. (Keskin et al., 2018; Kasm et al., 2018; Güler and Geçici, 2020; Güler et al., 2021)
The exam can only be assigned to an available room or timeslot. (Müller, 2016; Güler and Geçici, 2020; Carlsson et al., 2023)
No consecutive exam timeslots (Müller, 2016; Cataldo et al., 2017; Kasm et al., 2018; Hawari et al., 2020; Güler and Geçici, 2020; Güler et al., 2021; Bazari et al., 2023; Godwin, 2022)
Period related constraint
Exam duration must not exceed the designated timeslot. (Kasm et al., 2018; Zhu et al., 2022)
Minimize the total number of days on which exams are held. (Kasm et al., 2018)
All exams should be scheduled within the specified exam session. (Kasm et al., 2018; Hawari et al., 2020)
Enforces the assignment of exams to specified or preferred slots or days. (Kasm et al., 2018; Güler and Geçici, 2020; Hawari et al., 2020; Güler et al., 2021; Carlsson et al., 2023; Ozturk et al., 2024)
Spread student exams evenly across the exam session. (Kahar and Kendall, 2015; Woumans et al., 2016; Cavdur and Kose, 2016; Bazari et al., 2023; Bassimir and Wanka, 2024; Modirghorasani and Hoseinpour, 2024; Ozturk et al., 2024)
Consider timeslot precedence restrictions, e.g., scheduling Exam B after Exam A. (Carlsson et al., 2023)
Daily exam scheduling is restricted to user-identified exam slots. (Cavdur and Kose, 2016; Kasm et al., 2018; Zhu et al., 2022)
Courses in each semester should have different timeslots assigned for their exams. (Cavdur and Kose, 2016; Cataldo et al., 2017)
Exams for large classes should avoid early morning slots and be scheduled in the first week. (Ozturk et al., 2024)
Room related constraints
The total number of students taking the exam remains within the limits of the capacity of the room(s). (Kahar and Kendall, 2015; Müller, 2016; Woumans et al., 2016; Cataldo et al., 2017; Keskin et al., 2018; Kasm et al., 2018; Güler and Geçici, 2020; Hawari et al., 2020; Güler et al., 2021; Godwin, 2022; Carlsson et al., 2023; Bazari et al., 2023; Bassimir and Wanka, 2024; Ozturk et al., 2024)
Certain exams must be scheduled in the preferred room or specific room type. (Müller, 2016; Güler and Geçici, 2020; Hawari et al., 2020; Güler et al., 2021; Carlsson et al., 2023)
Each room accommodates only a single examination. No room sharing allowed. (Kahar and Kendall, 2015; Müller, 2016; Cataldo et al., 2017; Kasm et al., 2018; Hawari et al., 2020; Güler and Geçici, 2020; Bazari et al., 2023; Bassimir and Wanka, 2024)
A room cannot be used during timeslots in which it is unavailable. (Müller, 2016; Güler and Geçici, 2020)
Same-exam rooms should be located proximate to each other. (Kahar and Kendall, 2015; Müller, 2016; Cataldo et al., 2017; Keskin et al., 2018)
An exam can be split across multiple rooms. (Kahar and Kendall, 2015; Godwin, 2022; Bassimir and Wanka, 2024)
Maximize utilization of the room capacity. (Müller, 2016; Godwin, 2022)
The limit for splitting an exam into multiple rooms is not exceeded. (Müller, 2016; Hawari et al., 2020; Güler et al., 2021)
Invigilator related constraints
Invigilators cannot have overlapping duties per timeslot. (Çimen et al., 2022)
Invigilator workloads should be balanced across all invigilators. (Çimen et al., 2022)
Invigilators must not have more than two consecutive duties. (Çimen et al., 2022)
Invigilators cannot be assigned additional exams during the timeslots when they are responsible for invigilator duties. (Çimen et al., 2022)

Figure 2.1: Real-world ETP Constraints

The exam timetabling community is increasingly adopting rich formulations that incorporate a variety of constraints, as highlighted in Figure 2.1, moving away from simplified approaches. However, some formulations become excessively complex due to an overload of constraints. While this shift represents a positive advancement, the excessive focus on meeting specific constraints for individual universities or faculties can reduce the general applicability of methods, given the wide variety of constraints across institutions.

Avoiding overcomplicated objective function formulations is crucial for providing practical tools for comparison, ensuring a balanced approach that fosters robust progress in the field. In the real-world case, the aim is to balance the complexity of detailed formulations with the clarity of more straightforward models. To achieve this, a generalized objective function that captures the essence of real-world ETP under investigation was developed, enabling practical applicability while supporting meaningful comparisons.

2.4 Solution Approaches in ETP

An early survey by Carter (1986) examined the practical applications of graph coloring heuristics since the 1960s. Carter and Laporte (1996) extended the survey by categorizing algorithms into cluster methods, sequential methods, generalized search strategies, and constraint-based approaches. At that time, most algorithms addressed only the basic timetabling problem with simple constraints. Qu et al. (2009) conducted a comprehensive study spanning 1995 to 2008, examining key research trends, benchmarks, algorithmic methods, and achievements within the field. They presented state-of-the-art methodologies focusing on hybridization and the latest trends related to neighborhood structures. A comprehensive review of solution methodologies employed in ETPs, focusing exclusively on high-quality journal publications from 2012 to 2023 (Siew et al., 2024).

The solution methodologies for ETP, as in the literature from 2015 to 2024, were classified into five distinct categories:

- i. Mathematical Optimization (MO) (Yang, 2008): also known as mathematical programming, it optimizes mathematical models and includes methods like constraint programming, mixed-integer programming, goal programming, and linear programming.
- ii. Heuristics (HE) (Martí et al., 2018): finds high-quality solutions efficiently without guaranteeing optimality, including techniques like constructive heuristics, graph coloring, and decomposition approach.
- iii. Metaheuristics (Hussain et al., 2019): are generic or high-level strategies that integrate underlying heuristics, utilizing intelligent methods to strike a balance between exploring and exploiting the search space while incorporating learning techniques to guide the process.
- iv. Hyper-heuristic (HH) (Burke et al., 2013): operates within a search space consisting of low-level heuristics or heuristic elements.
- v. Hybrid Approaches (HA) (Blum et al., 2011): characterized by a high level of integration in this literature, achieved either by embedding a metaheuristic algorithm within a search tool or combining multiple metaheuristic algorithms into a cohesive framework.

2.4.1 Mathematical Optimization

Arbaoui et al. (2015) presented a mixed-integer mathematical model, which was tested on the ITC 2007 and Yeditepe benchmarks using CPLEX 12.5. Preprocessing utilized a

conflict graph to address implicit constraints, incorporating cliques and a dual-feasible function to improve the model, resulting in superior performance across most instances in relation to the original model. They (Arbaoui et al., 2019) then introduced a compact mixed-integer programming (MIP) model with equations proportional to exam counts, which significantly improved lower bounds and outperformed other formulations.

Woumans et al. (2016) developed two-column generation models for multi-version exam scheduling, tested with the KU Leuven campus real-world dataset and the Toronto benchmark. Both employed heuristics and binary integer programming approaches for exam version assignments. Model 1 excelled in spread cost in two Toronto instances, and Model 2 focused on minimizing total costs, solving only one within the time limit.

Cataldo et al. (2017) addressed a curriculum-based ETP for Universidad Diego Portales, where the number of conflicts depended on the curriculum configurations. Three mathematical programming models were developed in GAMS with CPLEX, with the first assigning timeslots to course groups, the second to individual courses, and the third allocating rooms to exams derived from the second model outcomes. The schedules outperformed manual processes and reduced conflicts.

Cavdur & Kose (2016) proposed a binary Goal Programming-based approach, integrating fuzzy logic, to address the ETP for the Industrial Engineering Department at Uludağ University. Fuzzy criticality levels for exams were assigned based on class credits, success ratios, and class types. The fuzzy criticality levels and pre-processed data were input into the Goal Programming model, followed by a simple room and proctor assignment algorithm. The model outperformed the human-generated schedule in evaluated objectives.

Keskin et al. (2018) developed a two-stage heuristic MIP model for addressing ETP in the Faculty of Engineering at Pamukkale University, where exam time is determined first and then classrooms and students are allocated to exams. The method produced faster and more feasible solutions than commercial solvers, shortening the exam period.

Güler & Gecici (2020) designed a decision support system that integrates a spreadsheet and an MIP model using Excel and Gurobi for Yıldız Technical University. They (Güler et al., 2021) also introduced two MIP models, utilising a decomposition approach, for scheduling exams and supervisors at a vocational school. The MIP models were implemented in a web-based decision support system, enabling end-users to efficiently generate timetables using the CBC open-source solver.

Al-Hawari et al. (2020) developed a three-phase Integer Linear Programming (ILP) for ETP at the German Jordanian University and Toronto benchmark, decomposing scheduling into three phases: timeslot allocation, day assignment, and room allocation. CPLEX solver was employed to compute solutions based on ILP formulations for every phase. The modular approach facilitated the creation of feasible timetables in standard formats more quickly than manual methods.

Godwin (2022) employed a linear 0–1 integer mathematical model in AMPL with CPLEX for a business school ETP, introducing two surrogacy-based methods for multi-objective optimization. One method evaluated the surrogacy levels among these objective functions to identify potential substitutes for multi-objective problems, while the other generated multiple solutions to compare quality and computational efficiency. The faculty-oriented and room-oriented models optimized quickly and served as effective surrogates but were less effective for student-oriented objectives.

Bazari et al. (2023) implemented an ILP model using CPLEX in the Management Department at Ferdowsi University. Its innovative search tree traversal in CPLEX efficiently discovers feasible solutions compared to depending only on the branch-and-cut approach. The model outperformed manual timetables with optimal room utilization and balanced exam distributions.

Çalik et al. (2024) introduced binary integer programming formulations with bi-objective strategies to address a real-world exam location problem for geographically distributed examinations. Using CPLEX 22.1, their method optimizes both location costs and participant allocation. Results show that minimizing travel distance under budget constraints produces high-quality solutions, though it leads to large Pareto sets and high computational demands for larger instances.

2.4.2 Heuristics

Abou et al. (2019) proposed a constructive heuristic incorporating a color graphing algorithm to solve the ETP at the Masdar Institute and Toronto benchmark problem. They introduced a constraint limiting the number of exams a student can be assigned within a specified period. The heuristic, influenced by a parameter for high-degree nodes, improved scheduling efficiency by prioritizing large-size courses. It focused on identifying the color group with the most non-conflicting vertices rather than minimizing the number of groups. Experimental results demonstrated its effectiveness, providing solutions that were optimal or close to optimal in terms of computational time and scheduling duration for both datasets.

Alkallak et al. (2022) proposed a Sudoku-based heuristic approach utilizing a Sudoku grid and Magic Square to assign exams to time slots and rooms, effectively eliminating the constraint of scheduling two exams for a student on the same day. Their method was

benchmarked against a basic heuristic algorithm, demonstrating superior performance of the intelligent algorithm over the classical approach in terms of computational efficiency.

2.4.3 Metaheuristics

In the design and application of metaheuristic algorithms, two opposing yet complementary criteria are to be taken into account: exploring the search space, which ensures diversification, and exploiting the best-found solutions, which promotes intensification. Based on these characteristics, metaheuristics can be broadly categorized into two categories: population-based and single-solution-based algorithms. Population-based methods emphasize exploration by simultaneously evolving a group of candidate solutions, as seen in approaches inspired by evolutionary principles and swarm intelligence. In contrast, single-solution-based methods focus on intensively improving a single candidate solution.

2.4.3.1 Population-based

i. Memetic Algorithm

Lei et al. (2018) developed a multiobjective evolutionary algorithm with a decomposition-based memetic algorithm to tackle the Toronto ETP. The algorithm used double hybrid initialization for diverse populations, followed by crossover and mutation. Bi-directed and period-supplement were introduced as local search operators for solution distribution enhancement. Results showed its superior performance over other random and hyper-heuristic methods.

Leite et al. (2018) proposed a cellular memetic algorithm that combines a cellular evolutionary algorithm and threshold acceptance to address the Toronto and ITC 2007 benchmarks. The algorithm optimized solutions using threshold acceptance, Kempe chain

neighborhoods, and a combination of mutation and crossover operators. The proposed algorithm outperformed previous methods, achieving the smallest relative deviation in multiple cases of both datasets.

ii. Prey-Predator Algorithm

Tilahun (2019) introduced an extended Prey-Predator (PPA) and experimented with it on six instances of the Toronto benchmark. The algorithm categorizes solutions into best prey, predator, and ordinary prey. The best prey exploited the neighborhood while the predator explored the search space. Both exploration and exploitation influenced ordinary prey. Statistical comparisons showed comparable performance to seven existing methods.

iii. Artificial Bee Colony

Zhu et al. (2022) designed a modified Artificial Bee Colony (ABC) algorithm to address a real-world online ETP. They used a conflict table to reduce extensive raw data into a smaller search space. The algorithm adjusted the neighbor search to limit over-exploration and included a parameter to prevent over-exploitation. Experimental results showed that the algorithm outperformed constraint programming.

iv. Firefly Algorithm

Nand et al. (2022) developed a discrete firefly algorithm for addressing the Toronto ETP, incorporating preference parameters and a stepping-ahead mechanism. The algorithm's three phases involved solution generation, improvement, and refinement. The stepping-ahead mechanism used preference parameters to optimize and explore the solution space, identifying superior solutions. The results performed similarly to 10 other algorithms on 5 out of 12 instances. In 2024, Nand et al. proposed a modified discrete Firefly Algorithm that

combines Kempe Chain-based neighborhoods with refined parameter tuning to enhance search performance. The algorithm was evaluated on seven Toronto benchmark datasets and a real-world HEI dataset, achieving high-quality solutions while ensuring convergence.

v. Genetic Algorithm

Ozturk et al. (2024) presented a web-based decision support system (DSS) powered by a Genetic Algorithm for addressing ETP at the Engineering Faculty of Eskisehir Technical University, considering COVID-19 constraints, including infection prevention and contact tracing. The model minimized pandemic-related violations, such as reduced occupancy and student interaction, while the DSS optimally scheduled eight departments, prioritizing social distancing.

2.4.3.2 Single-solution-based

i. Tabu Search

Amaral & Pais (2016) introduced a Tabu Search approach to address a multi-criteria ETP for the Toronto benchmark. To enhance automation, they proposed two key features: a fuzzy rule-based inference system for determining how long elements remain in the tabu list and the multi-criteria ranking system for flexible preference modeling, which facilitated a smoother trade-off among the objective function values.

ii. Simulated Annealing

Battistutta et al. (2017) introduced a Simulated Annealing (SA) method for the ITC 2007 benchmark, incorporating non-feasible solutions managed by penalty mechanisms. A two-step parameter tuning process identified key parameters and linked their values to instance features using a regression model. Their approach outperformed five ITC 2007 finalists

overall, achieving the best results in two instances and highlighting the value of well-tuned methods for generalization.

Leite et al. (2018) proposed FastSA, an SA variant with automated cooling rate computation, combining a saturated degree graph coloring-based construction phase with an optimization using the SA approach. Tested on ITC 2007 datasets, FastSA reduced evaluations and computation time compared to standard SA, ranking third among five algorithms while improving one instance.

Bellio et al. (2021) developed a two-stage SA method for addressing Toronto ETP. Stage 1 resolved exam conflicts to produce a feasible timetable, and Stage 2 optimized exam distances utilizing six neighborhood relations: Move, Swap, Kick, Double Kick, Swap Period, and Kempe Chain. Their approach was competitive and achieved the best-known results in 10 out of 13 instances.

Bulck et al. (2023) introduced an SA algorithm with six neighborhoods: Move, Swap, Kempe, Shake, PathRelinking, and Kick. Its parameters were fine-tuned using the irace package. Ablation analysis highlighted the significance of Move and Swap, whereas computational costs constrained the effectiveness of Kempe. The algorithm outperformed ITC 2007 competition finalists in 11 of 12 instances and achieved five optimal Yeditepe solutions.

Carlsson et al. (2023) developed and compared three methods —enhanced SA, constraint programming, and two MIP models — for addressing a real-world ETP. The extended SA consistently outperformed the standard SA in all instances and surpassed most MIP approaches, while constraint programming demonstrated weaker performance, and MIP encountered issues in certain instances.

Bassimir & Wanka (2024) presented a multi-objective SA model for curricula-based exam scheduling at Friedrich-Alexander University, addressing uncertainties from unknown student numbers and resulting conflicts. They utilized Kempe-chain techniques, greedy room assignment, and two robust measures—one minimizing elective student conflicts and another handling year-to-year variations. Results show these measures improve robustness with minimal impact on quality, reducing the need for rescheduling once precise data is available.

iii. Great Deluge

Kahar & Kendall (2015) developed a modified Great Deluge (GD) algorithm for a capacitated ETP at UMP, incorporating a dynamic parameter to adjust the decay rate, boundary, and target value throughout the search process. This approach outperformed UMP's proprietary software and original Dueck GD by producing high-quality solutions that met all constraints.

Burke & Bykov (2016) introduced Flex-Deluge, an enhanced GD variant with a flexible acceptance condition, tested on the Toronto and ITC 2007 benchmarks. They employed two techniques to enhance search performance: slowing the uphill move and tailoring the algorithm to problem-specific characteristics. Experimental results showed that adaptive Flex-Deluge algorithms outperformed GD, Hill Climbing, and fixed Flex-Deluge, achieving nine top results in the ITC 2007 benchmark and nine of the top 16 in the Toronto benchmark.

iv. Hill Climbing

Burke & Bykov (2017) enhanced Late Acceptance Hill Climbing (LAHC) with a greedy rule and improved history recording. Tested on the ITC 2007 benchmarks, it outperformed

basic LAHC, SA, threshold acceptance, and Great Deluge, especially on larger-sized problems. Bykov and Petrovic (2016) developed the Step Counting Hill Climbing (SCHC) algorithm for exam scheduling, using the Saturated Degree heuristic to generate an initial solution and iteratively refining it with various moves. Three variants were created: SCHC-all, which considers all moves; SCHC-ACP, which considers only accepted moves; and SCHC-imp, which counts only improving moves. SCHC-ACP outperformed LAHC, SA, GD, and other SCHC variants on ITC 2007 benchmarks, surpassing previous results in eight instances.

2.4.4 Hyper-heuristics

Sabar et al. (2015) developed a hyper-heuristic framework based on gene expression programming for addressing the ITC 2007 ETP. It combined perturbative low-level heuristics with dynamic multi-armed bandit-based heuristic selection and an automated move acceptance criterion using gene expression programming. This method outperformed the competition winner and post-competition approaches in four of eight instances.

Qu et al. (2015) proposed an Estimation Distribution Algorithm (EDA) based hyper-heuristic framework for addressing the Toronto ETP. Their high-level search used a basic EDA to guide heuristic selection, evaluating low-level heuristics via probability distributions during solution construction. Despite not incorporating backtracking or local refinement, this approach yielded promising results, matching the top performance among the ten methods.

Muklason et al. (2017) introduced a three-stage, multi-objective hyper-heuristic approach for improving fairness and standard objectives for Toronto, ITC 2007, and Yeditepe benchmarks. Using squeaky wheel optimization and a selection hyper-heuristic

integrated with GD and reinforcement learning, the framework demonstrated effectiveness on three benchmarks, matching the performance of established methods.

Hao et al. (2021) presented a hyper-heuristic framework that integrates evolutionary multitasking with graph heuristics for high-level search and low-level heuristics. This framework outperformed single-task hyper-heuristic methods in experiments on the Toronto benchmark.

Pillay & Özcan (2019) proposed a generative construction hyper-heuristic method to automate low-level construction heuristics, focusing on arithmetic heuristics using genetic programming (AHH) and hierarchical heuristics (HHH). AHH outperformed HHH and graph-coloring heuristics on the Toronto and ITC 2007 benchmarks. Heuristic effectiveness has been shown to be problem-dependent within educational timetabling contexts.

Mweshi & Pillay (2021) expanded the hyper-heuristic framework to create perturbative heuristics through grammatical evolution, incorporating additional conditional constructs, solution space data, and an expanded range of action syntax. Their approach demonstrated the effectiveness and versatility of the newly generated heuristics, outperforming existing methods on the ITC 2007 exam timetabling benchmark.

2.4.5 Hybrid Approaches

Alzaqebah & Abdullah (2015) hybridized a bee colony optimization with LAHC local search and self-adaptive disruptive selection strategy, producing results that rivaled the best-performing methods on the Toronto benchmark and replicated the success of the top five winners in the ITC 2007 benchmark. The results demonstrated the higher effectiveness of

the proposed hybrid approach compared to the basic bee algorithm, random and tournament selection, and SA-based local search.

Fong et al. (2015) hybridized the ABC algorithm with the GD algorithm for improved local exploitation and a global optimal model derived from particle swarm optimization to improve global exploration. Evaluations on the Toronto benchmark exceeded the performance of the standard ABC algorithm and maintained competitiveness with leading-edge approaches.

Li et al. (2015) introduced an Evolutionary Ruin and Stochastic Rebuild approach, validated on the Toronto benchmark. The method involved four phases: solution decomposition, evolutionary ruin, stochastic recreation, and solution acceptance. Though this approach outperformed graph-based hyper-heuristic algorithms, it was less effective than local search algorithms, according to experimental results.

Müller (2016) developed a multi-phase search algorithm for the real-world ETP at Purdue University. The algorithm utilized an iterative forward search method for solution construction, followed by hill climbing and GD for solution improvement and stabilization of bound variations. Results confirmed the effectiveness of the two-hour limit, with UniTime enabling interactive, solver-guided adjustments.

Lei et al. (2017) proposed a memetic algorithm that uses an adaptive coevolutionary strategy to select the Toronto benchmark search space. The memetic algorithm employed an evolutionary search approach in two domains: a heuristic space for global exploration and a solution space for local refinement, utilising adaptive coevolution to dynamically select the search space. Their approach ranked in the top five across 12 instances when evaluated against 11 competing algorithms.

Aldeeb et al. (2022) introduced a hybrid intelligent water drops (IWD) algorithm for the Toronto ETP, involving six phases and a local search optimizer using Move, Swap, and Kempe Chain structures for solution enhancement. The hybrid IWD outperformed nine swarm-based methods in three datasets and achieved the best results in one dataset compared to ten metaheuristics approaches.

Modirkhorasani & Hoseinpour (2024) proposed a hybrid algorithm that integrates genetic and SA techniques using the decentralized approach to address real-world ETP at a University. Empirical evaluations on 18 diverse instances demonstrate that the hybrid algorithm consistently outperforms both standalone methods and commercial solvers in optimization effectiveness. While the genetic algorithm excels in speed, the hybrid approach delivers consistently superior outcomes across all instances, demonstrating its effectiveness and robustness.

2.5 Current Trends of Solution Approaches

Table 2.4 summarizes the methodologies and datasets used in ETP research from 2015 to 2024, while Figure 2.2 illustrates the categorization of ETP solution methods. As shown in Table 2.4, there has been a noticeable shift in research focus from a predominant emphasis on categories heuristics (HE) and hyper-heuristics (HH) towards an increasing inclination towards categories population-based Metaheuristics (MH-P), single-solution-based Metaheuristic (MH-S), Mathematical Optimization (MO), and Hybrid Approach (HA) from 2015 to 2024.

Table 2.4: Methodologies and Datasets in ETP (2015 – 2024)

Year	Methodology (Reference)	Category	Dataset
2015	Hyper-heuristics + Multi-Armed-Bandit + Gene Expression Programming (Sabar et al., 2015)	HH	ITC 2007
	Hyper-heuristics (Qu et al., 2015)	HH	Toronto
	Bee Colony Optimization + Simulated Annealing + Late Acceptance Hill Climbing (Alzaqebah & Abdullah, 2015)	HA	Toronto, ITC 2007
	Artificial Bee Colony + Nelder-Mead + Great Deluge (Fong et al., 2015)	HA	Toronto
	Evolutionary Ruin + Stochastic Rebuild (Li et al., 2015)	HA	Toronto
	Great Deluge (Kahar & Kendall, 2015)	MH-S	Real-world
	Mixed Integer Programming (Arbaoui et al., 2015)	MO	ITC 2007, Yeditepe
2016	Step Counting Hill Climbing (Bykov & Petrovic, 2016)	MH-S	ITC 2007
	Tabu Search (Amaral & Pais, 2016)	MH-S	Toronto
	Flex-Deluge (Burke & Bykov, 2016)	MH-S	Toronto, ITC 2007
	Integer Programming + Column Generation (Woumans et al., 2016)	MO	Real-world, Toronto
	Goal Programming (Cavdur & Kose, 2016)	MO	Real-world
	Iterative Forward Search + Hill Climbing + Great Deluge (Müller, 2016)	HA	Real-world

Table 2.4 continued

2017	Simulated Annealing (Battistutta et al., 2017)	MH-S	ITC 2007
	Late Acceptance Hill Climbing (Burke & Bykov, 2017)	MH-S	ITC 2007
	Selection Hyper-heuristics + Great Deluge (Muklason et al., 2017)	HH	Toronto, ITC 2007, Yeditepe
	Adaptive Coevolutionary + Memetic Algorithm (Lei et al., 2017)	HA	Toronto
	Mixed Integer Programming (Cataldo et al., 2017)	MO	Real-world
2018	Memetic Algorithm (Lei et al., 2018)	MH-P	Toronto
	Cellular Memetic Algorithm (Leite et al., 2018)	MH-P	Toronto, ITC 2007
	Mixed Integer Programming (Keskin et al., 2018)	MO	Real-world
2019	Arithmetic Hyper-heuristics + Hierarchical Hyper-heuristics (Pillay & Özcan, 2019)	HH	Toronto, ITC 2007
	Simulated Annealing (Leite et al., 2019)	MH-S	ITC 2007
	Prey-Predator Algorithm (Tilahun, 2019)	MH-P	Toronto
	Mixed Integer Programming (Arbaoui et al., 2019)	MO	ITC 2007, Yeditepe
	Graph Coloring Heuristics (Kasm et al., 2019)	HE	Real-world, Toronto
2020	Mixed Integer Programming (Güler & Geçici, 2020)	MO	Real-world
	Integer Linear Programming (Hawari et al., 2020)	MO	Real-world, Toronto

Table 2.4 continued

2021	Grammatical Evolution Hyper-heuristics (Mweshi & Pillay, 2021)	HH	ITC 2007
	Simulated Annealing (Bellio et al., 2021)	MH-S	Toronto
	Mixed Integer Programming (Güler et al., 2021)	MO	Real-world
	Selective Hyper-heuristics (Hao et al., 2021)	HH	Toronto
2022	Intelligent Water Drops + Local Search Optimizer (Aldeeb et al., 2022)	HA	Toronto
	Stepping Ahead Firefly Algorithm (Nand et al., 2022)	MH-P	Toronto
	Integer Linear Programming (Godwin, 2022)	MO	Real-world
	Artificial Bee Colony (Zhu et al., 2022)	MH-P	Real-world
	Sudoku-based Heuristic (Alkallak et al., 2022)	HE	Real-world
2023	Simulated Annealing (Bulck et al., 2023)	MH-S	ITC 2007, Yeditepe
	Integer Linear Programming (Bazari et al., 2023)	MO	Real-world
	Constraint Programming, Mixed Integer Programming, and Simulated Annealing (Carlsson et al., 2023)	MO MH-S	Real-world
2024	Genetic Algorithm (Ozturk et al., 2024)	MH-P	Real-world
	Genetic + Simulated Annealing (Modirkhorasani & Hoseinpour, 2024)	HA	Real-world
	Simulated Annealing (Bassimir & Wanka, 2024)	MH-S	Real-world
	Stepping Ahead Firefly Algorithm (Nand et al., 2024)	MH-P	Real-world, Toronto
	Binary Integer Programming (Çalık et al., 2024)	MO	Real-world

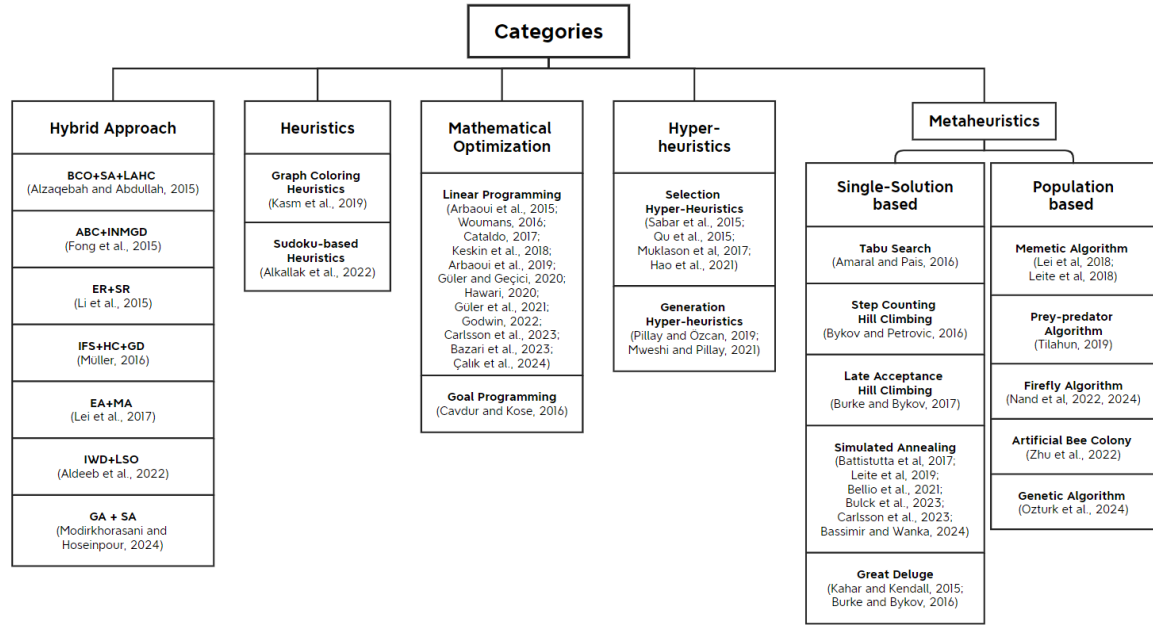


Figure 2.2: Categories of ETP Solution Methodologies

The distribution of the papers studied in the literature, based on publication year, is shown in Figure 2.3. The distribution by categories is presented in Figure 2.4, with both figures offering percentage and numerical perspectives. Figure 2.3 shows a fluctuating trend in publication numbers, with 2015 having the highest number of papers (15.56%), followed by 2016 (13.33%). Years 2024, 2022, 2019, and 2017 each account for 11.11%, while 2020 has the lowest share at 4.44%, indicating relatively lower and variable counts. Figure 2.4 shows that the Metaheuristics (MH) category has the highest percentage, accounting for 36.21% of the papers. The mathematical optimization (MO) category follows with 29.31%, while the heuristics category has the least at 5.17%.

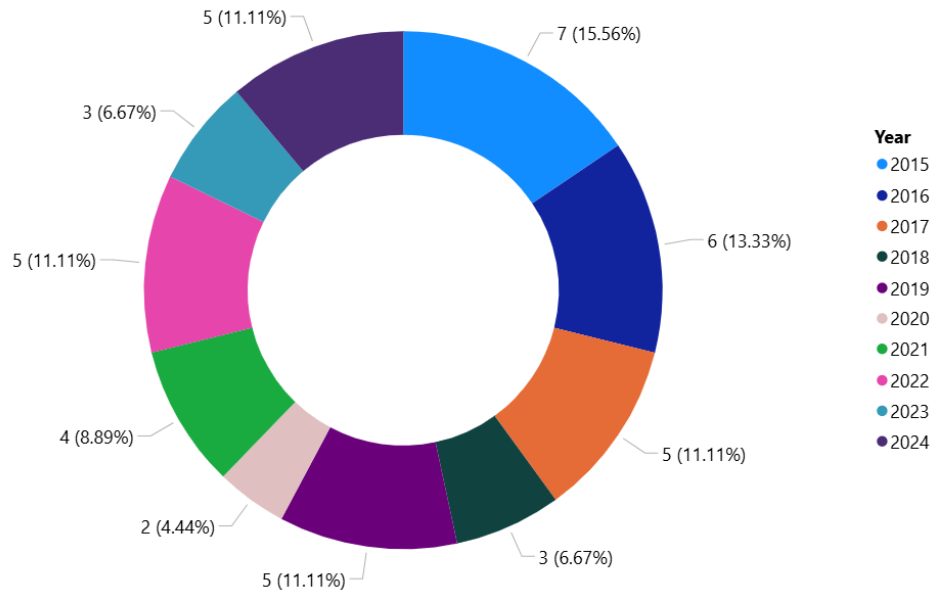


Figure 2.3: Distribution of the ETP Studies by Publication Year

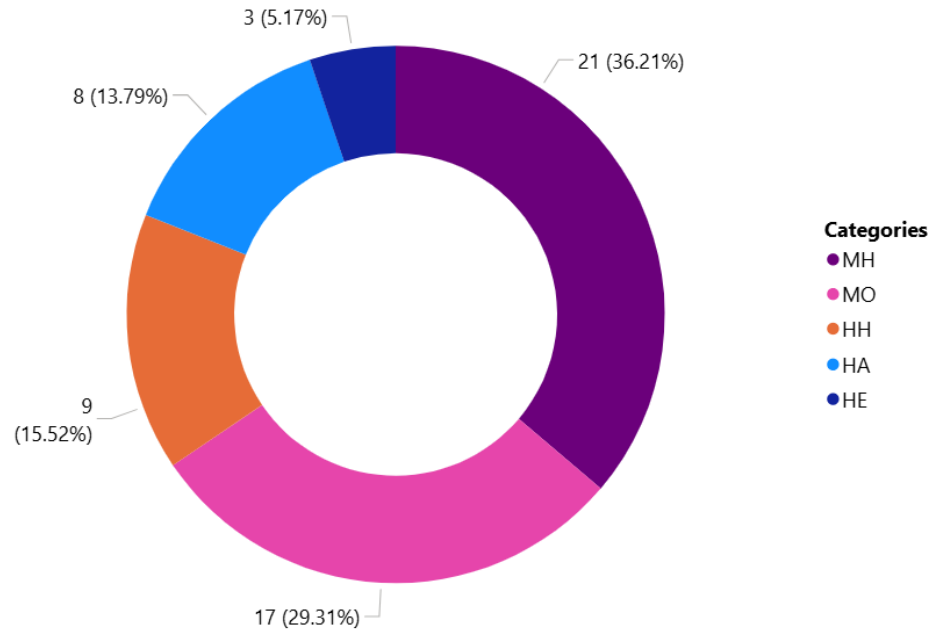


Figure 2.4: Distribution of the ETP Studies by Categories

Figure 2.5 (a) illustrates the distribution of studied datasets, with other real-world datasets being the most examined, followed by the Toronto benchmark and the ITC 2007 benchmark. Figure 2.5 (b) shows the distribution of categories used for each dataset. The

most studied benchmark dataset is Toronto, which predominantly uses the hybrid approach, followed by the population-based metaheuristic and single-solution-based metaheuristic. In the real-world datasets, which mainly consist of smaller-scale data, the dominant category is Mathematical Optimization. In contrast, the ITC 2007 dataset primarily focuses on single-solution-based metaheuristics.

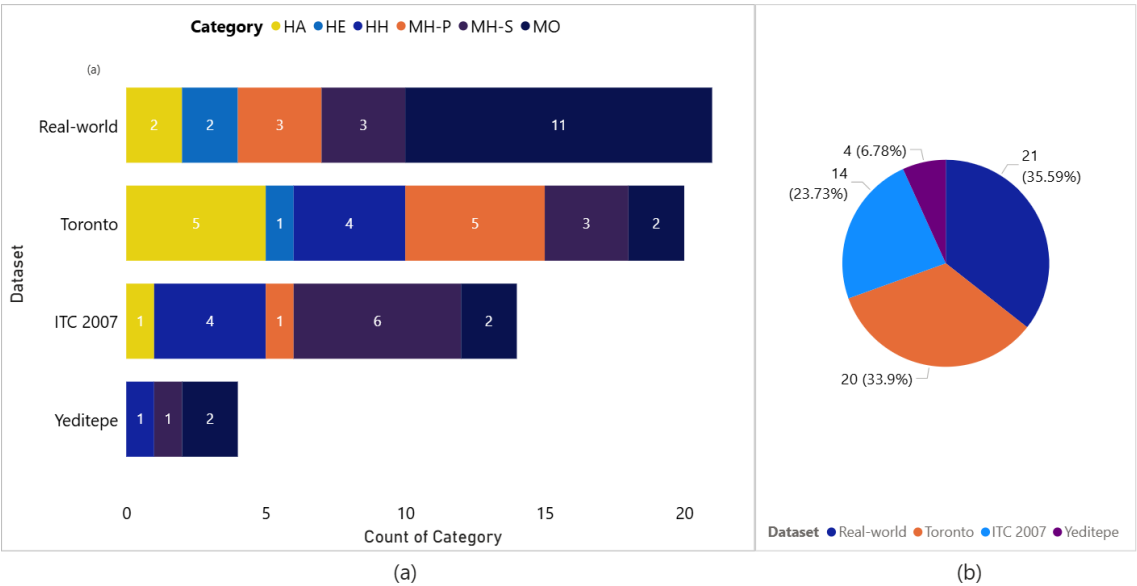


Figure 2.5: (a) Distribution of Datasets (b) Distribution of Categories by Datasets

Table 2.5 presents the count of categories across different datasets, along with their total counts. The dataset distribution shows that real-world cases have the highest total count (21), where Mathematical Optimization is the most widely applied method. The Toronto datasets (20) dominate by Metaheuristics and Hybrid Approach, while Mathematical Optimization and Heuristics are less utilized. The ITC 2007 datasets, with a total of 14, are primarily led by single-solution-based metaheuristics. In contrast, the Yeditepe dataset has the smallest total (4), indicating minimal representation across categories. Overall, Metaheuristics (21) and Mathematical Optimization (17) are the most prominent categories across all datasets, contributing significantly to 59 entries.

From this literature, it is evident that the majority of current studies on ETPs primarily focus on benchmark problems (64.4%), with the Toronto benchmark being the most extensively studied (33.9%). This work focuses on studying contemporary real-world problems at a university to bridge the gap between theoretical benchmarks and practical application. To achieve this, solution approaches tailored to real-world scenarios were developed. Furthermore, the proposed solution approaches are validated by testing the Toronto benchmark problem, ensuring their applicability across instances with diverse characteristics.

Table 2.5: Counts of Categories across Datasets

Datasets	Categories						Total
	MO	HE	MH-S	MH-P	HA	HH	
Real-World	11	2	3	3	2	-	21
Toronto	2	1	3	5	5	4	20
ITC 2007	2	-	6	1	1	4	14
Yeditepe	2	-	1	-	-	1	4
Total	17	3	13	9	8	9	59

2.6 State-of-the-Art Methods for Benchmark Problems

This section presents the outcomes for each formulation considered “state-of-the-art,” referring to those that achieved the best scores for the instances introduced in Section 2.3. For specific variants, the runtime is also considered where relevant. The optimal results for each instance, as shown in the table, are marked in bold.

2.6.1 Toronto Benchmark

Table 2.6 presents the state-of-the-art methods with the best scores for addressing the Toronto benchmark problem. The top scores are shown in bold. The multi-neighborhood SA approach by Bellio et al. (2021) usually achieves leading results. Additionally, the cellular memetic method by Leite et al. (2018) and the flex-deluge approach by Burke & Bykov (2016) achieved the best outcomes in specific instances. However, performance in this benchmark problem is highly time-sensitive, with longer runtimes producing better results. Therefore, fair comparisons are difficult due to the literature's varying processor speeds and runtime conditions.

Table 2.6: Best Scores of State-of-the-Art Methods for Toronto Benchmark

Instances	Burke & Bykov (2016)	Leite et al. (2018)	Bellio et al. (2021)
CAR-S-91	4.32	4.31	4.24
CAR-F-92	3.67	3.68	3.64
EAR-F-83	32.62	32.48	32.42
HEC-S-92	10.06	10.03	10.03
KFU-S-93	12.80	12.81	12.81
LSE-F-91	9.78	9.78	9.78
PUR-S-93	3.88	4.14	4.00
RYE-F-92	7.91	7.89	7.84
STA-F-83	157.03	157.03	157.03
TRE-S-92	7.64	7.66	7.59
UTA-S-92	2.98	3.01	2.95
UTE-S-92	24.78	24.80	24.76

Table 2.6 continued

YOR-F-83	34.71	34.45	34.40
Category	MH-S	MH-P	MH-S
Methods	Flex-Deluge	Memetic Algorithm	Simulated Annealing
Time [hours] (Min, Max, Mean)	4.6 5.7 5.1	24 48 31.4	26.2 52.2 34.7

2.6.2 ITC 2007 Track 1 Benchmark

Table 2.7 presents the state-of-the-art methods with the best scores for addressing the ITC 2007 Exam Track benchmark problem. The top scores are shown in bold. The competition's top results were primarily achieved through Bykov & Petrovic's (2016) step-counting hill climbing method and Burke & Bykov's (2016) flex-deluge approach, which dominated most instances. Gogos et al.'s (2010) parallel distributed scatter search method excelled in three instances. To ensure comparability, the running time closely adheres to the time limit specified by the ITC 2007 benchmarking tool.

Table 2.7: Best Scores of State-of-the-Art Methods for ITC 2007 Exam Track

Instances	Bykov & Petrovic (2016)	Burke & Bykov (2016)	Gogos et al. (2010)
EXAM 1	3647	3691	4128
EXAM 2	385	385	380
EXAM 3	7487	7359	7769
EXAM 4	11779	11329	13103
EXAM 5	2447	2482	2513

Table 2.7 continued

EXAM 6	25210	25265	25330
EXAM 7	3563	3608	3537
EXAM 8	6614	6818	7087
EXAM 9	924	902	913
EXAM 10	12931	12900	13053
EXAM 11	23784	22875	24369
EXAM 12	5097	5107	5095
Category	MH-S	MH-S	MH-P
Methods	Step Counting Hill Climbing	Flex-Deluge	Scatter Search

2.6.3 Yeditepe Benchmark

Table 2.8 presents state-of-the-art methods with the best scores for addressing the Yeditepe benchmark problem. The best scores are displayed in bold. The state-of-the-art method for this dataset is the multi-neighborhood SA, introduced by Bulck et al. (2023). Both the hyper-heuristics method by Muklason et al. (2017) and the MIP approach by Arbaoui et al. (2019) deliver the best results in the same two instances, matching the highest score achieved by Bulck et al. (2023).

Table 2.8: Best Scores of State-of-the-Art Methods for Yeditepe Benchmark

Instances	Muklason et al. (2017)	Arbaoui et al. (2019)	Bulck et al. (2023)
yue20011	56	73	47
yue20012	122	162	97
yue20013	29	29	29
yue20021	76	125	46
yue20022	162	365	119
yue20023	56	56	56
yue20031	143	255	79
yue20032	434	550	296
Category	HH	MO	MH-S
Methods	Hyper-heuristics + Great Deluge	Mixed Integer Programming	Simulated Annealing

2.7 Metaheuristic Methods Considered for Application to ETP

Metaheuristics, being the most widely employed approach in examination timetabling over the past decade, are high-level strategies that guide search agents to iteratively enhance solutions by exploring random elements and possibilities. While they may not guarantee the absolute best solution, they often produce high-quality results within a reasonable computational time. Metaheuristic algorithms primarily differ in how they balance exploration (global search) and exploitation (local refinement near promising solutions). This section offers an overview of the metaheuristic methods considered for application to ETP, including prior work and algorithmic design.

2.7.1 Variable Neighborhood Descent

Variable Neighborhood Descent (VND), introduced by Mladenović and Hansen (1997), is a single-solution-based metaheuristic which aims to discover high-quality solutions by systematically changing neighborhood structures. VND explores a given neighborhood in the search process until it reaches a local optimum. It then deterministically switches to a different, previously unexamined neighborhood structure to proceed with the search. This systematic change in neighborhoods helps avoid the search getting stuck in local optima. Since different neighborhood structures often lead to distinct local optima, leveraging multiple structures can significantly enhance the likelihood of finding higher-quality solutions.

The concept of VND is based on the following key insights: a solution that is a local optimum in one neighborhood structure may not be locally optimal in a different one, while a global optimum must be locally optimal across all neighborhood structures. Therefore, VND systematically explores the solution space using multiple neighborhood structures—applied in sequential, nested, or mixed strategies (Duarte et al., 2018). VND variants can be distinguished based on the order of neighborhoods, the choice of search strategies (first improvement vs. best improvement), and the move strategy, which dictates how the search proceeds when an improvement is found in a neighborhood.

VND has been effectively adopted as a local search replacement within metaheuristics, giving rise to hybrid algorithms and forming a variant of Variable Neighborhood Search (VNS). VND is the simplest variant of VNS. Unlike traditional VNS-based algorithms, the VND does not utilize a shake strategy. Instead, it transitions from one neighbourhood definition to the next in a fixed, pre-defined order that remains unchanged throughout the

algorithm's execution. This contrasts with other VNS-based approaches, where the next neighbourhood to explore is determined dynamically during the algorithm's progress, based on the shake strategy.

VND has been widely integrated into the improvement phase or employed as a local search method in educational timetabling. It has been hybridized with other approaches for course timetabling, including the VNS-TS method by Vianna et al. (2020) and the combination of heuristic orderings with VND proposed by Chen et al. (2021). VND has also been adopted as a base algorithm in several studies, such as the iterative VND by Soria et al. (2017), the VND-based search algorithm by Borchani et al. (2017), and the application of VND as a local search method by Teixeira et al. (2019) for school timetabling problems.

In examination timetabling, Ayob et al. (2006) proposed an iterative restart VNS, a descent-ascent heuristic that utilizes neighborhood structures in the improvement phase, and applied it to the Toronto benchmark ETP, yielding promising results. Burke et al. (2010) investigated different variants of VNS and introduced a hybrid GA that intelligently selects the most appropriate neighborhoods within the VNS framework. Their method demonstrated highly competitive solution quality on the same benchmark problems, albeit with significantly higher computational time.

Figure 2.6 outlines the pseudocode for the sequential VND algorithm. An initial feasible solution, denoted as X_{best} , is first generated—either randomly or through a specific heuristic algorithm. While $k \leq k_{max}$, a local search is conducted within the current neighborhood structure N_k , followed by a transition to the next neighborhood based on a predefined sequence. This structured progression guides the exploration of the solution space. If the

optimal solution identified in the current neighborhood improves upon X_{best} , it replaces it and serves as the starting point for the next iteration.

Procedure Sequential Variable Neighborhood Descent (VND)

```

1:  $X_{best} \leftarrow \text{GENERATEINITIALSOLUTION}()$ 
2:  $k \leftarrow 1$ 
3: while  $k \leq k_{max}$  do
4:    $X' \leftarrow \text{argmin}_{N_k}(X_{best})$ 
5:    $k \leftarrow k + 1$ 
6:   if  $Z(X') < Z(X_{best})$  then
7:      $X_{best} \leftarrow X'$ 
8:   end if
9: end while
10: return  $X_{best}$ 

```

Figure 2.6: Pseudocode of Sequential VND

The VND algorithm's key advantage is its ability to ensure minimal or near-minimal solutions within each neighborhood, improving solution quality. By strategically choosing neighborhoods and switching between them, VND enhances efficiency and helps escape local optima. The effectiveness of VND is largely influenced by the ordering and selection of neighborhoods, making the design of the neighborhood structure a critical component. VND consistently demonstrates strong performance and is predominantly applied in single-objective optimization contexts, where it typically produces a single optimal solution (Yu et al., 2020). The straightforward structure of VND, combined with the potential for efficient neighborhood definitions, makes it an approach worth exploring and implementing in the context of exam timetabling.

2.7.2 Whale Optimization Algorithm

The Whale Optimization Algorithm (WOA) is a swarm intelligence metaheuristic biologically inspired technique developed by Mirjalili and Lewis (2016), which considers

the group hunting behavior of humpback whales. These whales are considered highly intelligent and often target prey near the ocean surface, such as krill or small fish. Their unique hunting technique, known as bubble-net feeding, involves swimming around the prey while releasing bubbles to trap them. This behavior, particularly the circular or ‘9’-shaped bubble patterns, forms the basis of the WOA’s optimization strategy.

To adapt WOA for broader applications, various variants have been proposed that modify the distance and position equations, tune internal parameters, introduce new mechanisms for initializing solutions, convert the problem space from continuous to discrete, or hybridize WOA with other methods to enhance its performance. Enhancing its three search strategies—encircling the prey, searching for prey, and spiral bubble-net attacking—greatly affected its performance, serving as the primary motivation behind most of these advancements.

Various discrete variants of the WOA have been explored, involving the incorporation of new equations to transform the search space into discrete or binary representations. Successful studies have demonstrated the effectiveness of the WOA in solving discrete optimization problems, with most variants developed through algorithmic improvement and hybridization techniques.

A hybrid WOA was proposed by Rana et al. (2022) to address the virtual machine scheduling problem in cloud computing by incorporating Differential Evolution as a local search optimizer within the WOA framework to enhance solution diversity and improve local search performance. The results of the experiments indicated that the approach consistently reduced makespan and improved cost efficiency over existing algorithms.

A hybrid WOA for the Flexible Job Shop Scheduling Problem was developed by Yang et al. (2022), incorporating a nonlinear convergence factor to balance the dynamics between global and local searches. Furthermore, a multi-neighborhood structure was introduced to enhance the search process from multiple directions. Tested on seven benchmarks, the hybrid WOA showed superior scheduling efficiency. Zhang and Gu (2023) proposed a discrete WOA for solving the No-Wait Flow Shop Scheduling Problem, integrating Parallel Neighbourhood Search and Serial Neighborhood Search for strengthening the algorithm's global and local search capabilities, respectively.

An improved WOA was introduced by Abdel et al. (2019) to address both single and multidimensional 0–1 knapsack problems by integrating Local Search and Levy flights to balance exploration and exploitation. Likewise, Li et al. (2020) proposed a discrete WOA based on a transfer V-shaped function for the knapsack problem.

Xu et al. (2018) proposed a Memetic WOA for optimization by integrating a chaotic local search mechanism to enhance its exploitation capabilities. The algorithm outperforms its competitors in terms of convergence speed and accuracy, as demonstrated by tests on 48 benchmark functions. Hussien et al. (2020) proposed an adaptation of the WOA to handle binary optimization problems by introducing S-shaped and V-shaped transfer functions that map the continuous search space to a binary one for discrete optimization problems. Experimental results demonstrate that the proposed WOA outperforms state-of-the-art metaheuristics in both accuracy and speed.

For University Timetabling, Austero et al. (2020) presented an enhanced WOA to tackle the course timetabling problem by integrating crossover operator, mutation operator, and neighborhood swapping. The enhanced WOA outperformed classical WOA in generating

optimal timetables across different dataset sizes. Premananda et al. (2020) proposed a Hybrid WOA to solve the ITC 2019 course timetabling problem by utilizing neighborhood structures such as mutation, swap, and greedy mutation in WOA's exploitation and exploration phases. LAHC is activated after stagnation to further enhance the search by exploring improved solutions. The approach achieved the 7th rank out of 13, matching the best-known result in the competition.

As reported in WOA (Mirjalili & Lewis, 2016), the algorithm was originally designed for single-objective, continuous optimization problems. Its mathematical formulation includes equations for calculating the distance between whales and their prey, as well as determining the optimal position of the whales using a set of internal parameters. Table 2.9 presents a set of symbol definitions utilized within equations executed within the search process in the original WOA algorithm.

Table 2.9: Symbols for WOA Equations

Symbols	Description
t	The current iteration variable.
X	The current position vector.
X^*	The current best position vector.
A	The coefficient vector used to control the step size and direction of movement, determining the region where a whale can be located relative to the prey's position.
C	The coefficient vector to introduce randomness for diversifying the search.
a	Elements in a parameter linearly from 2 to 0 throughout the iterations in both exploration and exploitation phases.
r	Randomly generated value with a uniform distribution in the interval $[0,1]$.
$Max_Iteration$	The maximum allowed iterations.

Table 2.9 continued

D	The distance between the i -th whale and the current best solution.
b	A constant that defines the shape of the logarithmic spiral.
l	A random number within the range $[-1,1]$.
X_{rand}	A random whale selected from the population.

WOA consists of two phases: exploitation and exploration. In the exploitation phase, the algorithm mimics the whales' bubble-net hunting strategy by simulating circular and spiral movements around prey, while the exploration phase models random search behavior. The encircling prey and spiral bubble-net attacking methods were implemented in the exploitation phase of the algorithm. The exploration phase involves a random search for prey. The mathematical formulation for each phase is outlined in the next section.

2.7.2.1 Exploitation Phase – Encircling Prey and Spiral Updating Position

i. Encircling Prey

To hunt prey, humpback whales exhibit encircling behavior. Since the exact position of the optimum is unknown, the best current solution is assumed to represent the target prey. Once this best agent is identified, the other agents update their positions accordingly to move toward it, as modeled by Equation 2.1 and Equation 2.2. According to Equation 2.2, the search agents update their positions based on the location of the current best-known position.

The coefficient vectors A and C , derived from Equation 2.3 and Equation 2.4 are used to refine the search area and identify the whale's position relative to its prey. Various positions around the current best agent can be explored by adjusting the values of the A and C coefficient vectors. The shrinking encircling behavior is controlled by gradually reducing

the value of a in Equation 2.3 and Equation 2.5, which narrows the range of A to $[-a, a]$. The incorporation of a random vector r introduces stochasticity, allowing the search agent to sample positions within a region defined by the current and best solutions. Consequently, Equation 2.2 enables each search agent to update its position in the vicinity of the best-known solution, effectively modeling the encircling behavior observed in prey-hunting whales.

$$D = |C \cdot X^*(t) - X(t)| \quad \text{Equation 2.1}$$

$$X(t+1) = X^*(t) - A \cdot D \quad \text{Equation 2.2}$$

$$A = 2ar - a \quad \text{Equation 2.3}$$

$$C = 2r \quad \text{Equation 2.4}$$

$$a = 2 - t \left(\frac{2}{Max_Iteration} \right) \quad \text{Equation 2.5}$$

ii. Spiral Updating Position

In this mechanism, the initial step involves calculating the distance between the whale and its prey. Subsequently, a spiral path is generated to replicate the humpback whale's movement in a helical shape, which can be represented by Equation 2.6. To simulate humpback whales' simultaneous circular and spiral movements around prey, a 50% probability is assigned to select between the mechanisms of shrinking encircling prey and the spiral updating mechanism to model their respective behaviors. The search agent's

position is updated using Equation 2.2 when $p < 0.5$, applying the mechanisms of shrinking encircling prey, and Equation 2.6 when $p \geq 0.5$, simulating the spiral- updating mechanism.

$$X(t + 1) = D' \cdot \exp(bl) \cdot \cos(2\pi l) + X^*(t) \quad \text{Equation 2.6}$$

2.7.2.2 Exploration Phase – Searching for Prey

Instead of updating the positions of search agents solely based on the position of the current best agent, a random search agent is used to guide the search process for exploration. Vector A with random values greater than 1 or less than -1 , is used to encourage the search agent to move away from the current best solution. This mechanism is mathematically represented by Equation 2.7 and Equation 2.8. When $|A| > 1$, the mechanism emphasizes exploration, enabling WOA to conduct a broader global search across the solution space.

$$D = |C \cdot X_{rand} - X| \quad \text{Equation 2.7}$$

$$X(t + 1) = |X_{rand} - A \cdot D| \quad \text{Equation 2.8}$$

2.7.2.3 Original WOA

Figure 2.7 outlines the pseudocode for the original WOA algorithm. The algorithm begins by creating a random initial population and evaluating the fitness values of each. Once the best solution is identified, the algorithm iterates through the subsequent steps until a termination condition is fulfilled. Once the best solution is identified, the algorithm iteratively performs four main steps until a stopping criterion is reached: (1) the coefficient vectors a , A , C and l are calculated and updated; (2) a random value is generated for p , which determines how the position of a solution is updated using either Equation 2.2, Equation 2.8, or Equation 2.6; (3) constraining solutions within the defined search space; (4) the best solution identified is returned as an approximation of the global optimum.

Procedure Original Whale Optimization Algorithm	
1:	Generate Initial Population X_i ($i = 1, 2, \dots, n$)
2:	Calculate the fitness of each solution
3:	$X^* \leftarrow$ the best search agent
4:	while $t < Max_Iteration$ do
5:	for each solution do
6:	Update a , A , C , l , and p
7:	if $p < 0.5$ then
8:	if $ A < 1$ then
9:	Update the position of the current solution by Eq. (2.2)
10:	else
11:	Select a random search agent
12:	Update the position of the current search agent by Eq. (2.8)
13:	end if
14:	else
15:	Update the position of the current search agent by Eq. (2.6)
16:	end if
17:	end for
18:	Check if any solution goes beyond the search space and amend it
19:	Calculate the fitness of each solution
20:	Update X^* if there is a better solution
21:	$t \leftarrow t + 1$
22:	end while
23:	return X^*

Figure 2.7: Pseudocode of Original WOA

2.8 Research Gaps and Opportunities in ETP

i. Datasets in Examination Timetabling

Due to the distinct policies and regulatory requirements of individual institutions, developing a universal algorithm to solve all ETPs is nearly impossible. Benchmark ETPs are typically simplified for methodological comparisons, whereas real-world scenarios demand practical, institution-specific solutions that address diverse constraints. Although the ITC 2007 exam track benchmark offers more realistic constraints and a complex formulation compared to other benchmarks, it is based on data from 2007 and therefore does not adequately reflect current operational requirements or real-world ETP complexities. Thus, incorporating studies based on current data that reflect contemporary academic environments and evolving operational requirements is essential for achieving accurate and practical evaluations.

Over the past decade, higher education has undergone a major shift, with 98% of universities adopting online classes (UpSkillWise, 2025) and leading institutions successfully conducting remote examinations during the COVID-19 pandemic (Pokhrel & Chhetri, 2021; Gorgani & Shabani, 2021). Even in the post-pandemic era, this trend persists—Monash University, for example, now delivers around 30% of its exams as e-assessments, creating new flexibility. A hybrid model combining online and paper-based exams creates new opportunities but also introduces evolving timetabling challenges, necessitating fresh perspectives and innovative methods. (Chanthadavong, 2019).

Furthermore, most real-world examination timetabling models in the literature tend to focus narrowly on the faculty or departmental level, frequently overlooking broader, institution-wide dynamics. This highlights a research opportunity in developing integrated

approaches that address cross-departmental or cross-faculty challenges to enhance institutional management efficiency. These gaps underscore the need for real-world datasets that accurately represent contemporary institutional requirements and practical constraints (Muklason et al., 2017). This study adopts an applied research focus to address gaps in meeting current institutional requirements and aligning with real-world needs, while exploring research opportunities through a comparative analysis of centralized and decentralized scheduling approaches.

ii. Metaheuristic Parameters and Neighborhood Structures

Based on existing literature, metaheuristics are generally considered state-of-the-art methods in most benchmark problems. However, the algorithm's efficiency and effectiveness in producing high-quality solutions for ETPs largely depend on the precise tuning of its parameters (Nand et al., 2024) and problem-specific parameter settings (Petrovic & Burke, 2004), which limit their applicability to diverse instances. Despite advancements, excessive parameter tuning makes metaheuristics particularly challenging and time-consuming for non-experts, such as timetable officers, to implement effectively, leading many universities to continue relying on manual scheduling methods.

One of the broadly used metaheuristic, local search methods, heavily relies on the selection of neighborhood structure and the defined search space. Many implementations struggle due to overly simplistic neighborhood designs. Recent state-of-the-art techniques (Sabar et al., 2015; Bykov and S. Petrovic, 2016; Burke & Bykov, 2016; Leite et al., 2018; Bellio et al., 2021). Bulck et al. (2023) demonstrated improved performance by incorporating sophisticated neighborhood structures, especially during the improvement phase. Notably, Bellio et al. (2021) achieved optimal results using a combination of

neighborhoods based on various atomic moves and proposed exploring dynamic neighborhood probabilities via online learning.

To enhance local intensification, adaptation mechanisms that dynamically select neighborhood structures and manage the search process have been recommended (Abdullah & Alzaqebah, 2013; Alzaqebah & Abdullah, 2015). Researchers have suggested that future research could benefit from developing neighborhood operators alongside Kempe-chain operators (Leite et al., 2019; Nand et al., 2024), exploring alternative variations like shuffling periods instead of relocating exams, and adapting neighborhoods for weakly connected subgraphs (Bulck et al., 2023).

Given the limitations of complex parameter tuning in many metaheuristics, this study adopts Variable Neighborhood Descent (VND) for its simplicity and robustness. Specifically, VND is selected over other variants due to its deterministic nature and ability to systematically explore multiple neighborhood structures without requiring heavy parameter calibration. Moreover, the effective design of neighborhood structures has proven to significantly enhance solution quality and adapt to various search spaces, reinforcing VND as a justified choice for this work.

iii. Hybrid and Innovative Approach for Examination Timetabling

Although metaheuristic algorithms have demonstrated success across various research domains, their application to university ETPs remains limited. Notably, several algorithms—such as Symbiotic Organisms Search (Cheng & Prayogo, 2014), Sine–Cosine Algorithm (Mirjalili, 2016), Moth-Flame Optimisation (Mirjalili, 2015), Whale Optimisation Algorithm (WOA) (Mirjalili & Lewis, 2016), and Elitist Self-Adaptive Step-Size Search (Azad & Hasançebi, 2014)—have yet to be explored in this context. Future research should

investigate these contemporary algorithms for ETPs, systematically evaluating and comparing their performance against existing methods. This would help identify their respective strengths and limitations, contributing to a robust reference framework for subsequent studies (Bashab et al., 2020).

Each algorithm, including the most successful, has inherent weaknesses in its search strategy, leading to the continuous development of numerous optimization algorithms. Hybridizing existing algorithms rather than developing new ones offers a practical approach to improving search strategies by addressing their limitations. Population-based techniques explore multiple regions of the search space simultaneously but often struggle to identify local optima (Fesanghary et al., 2009). In contrast, local search-based techniques efficiently handle large-scale problems and quickly converge with local optima (Burke & Bykov, 2017), though they lack a comprehensive search of the entire space.

Local search algorithms, when hybridized with population-based methods (Betar et al., 2014) have shown considerable effectiveness in tackling complex real-world scenarios, with research consistently highlighting their enhanced problem-solving capabilities (Blum & Roli, 2003). Exploring alternative hybrid approaches (Al-Betar et al., 2014; Leite et al., 2018; Muklason et al., 2017) offers considerable research opportunities to enhance algorithmic performance (Bashab et al., 2020), as combining methods can mitigate individual weaknesses and leverage their strengths.

A notable research gap exists in the application of hybrid approaches to exam timetabling, particularly the absence of studies utilizing a recent promising metaheuristic technique, the WOA. Although originally designed for continuous optimization problems, WOA has been successfully adapted for various discrete domains due to its strong

exploration–exploitation balance and minimal parameter requirements. Compared to other metaheuristics, its simpler structure allows adaptation to hybridization methods, particularly in the context of exam timetabling, where the application of WOA remains unexplored. Hybridizing WOA with local search methods leverages the strengths of both approaches, enabling quicker and more precise convergence to optimize solution quality. Despite not being widely used for university timetabling in recent literature, its properties have high potential to suit real-world problems' constraints.

2.9 Conclusion

This chapter introduces the educational timetabling problem, with a focus on examination timetabling. The specific constraints associated with examination timetabling vary across institutions. According to the literature, the Toronto benchmark has been the most extensively studied variant over the past decade, followed by other real-world problems. The ITC 2007 and Yeditepe benchmarks are also notable variants. Despite its long-standing presence, the Toronto benchmark remains a challenging problem due to its inherent characteristics.

There is a growing trend towards studying more real-world problems, which is a positive development. However, including diverse constraints often leads to complex objective functions, making generalization and comparative studies difficult. This underscores the need for a more generalized objective function that can accommodate diverse constraints while enabling meaningful comparisons across studies.

Various solution approaches, particularly metaheuristics, have been extensively applied to address the benchmark ETPs. Nonetheless, the effectiveness of metaheuristics is heavily dependent on parameter tuning (Petrovic & Burke, 2004), which can be a challenge for non-

experts, such as timetabling officers. This research primarily employs VND alongside the recently introduced WOA, which features minimal parameter requirements while demonstrating proven effectiveness in combinatorial optimization and a balance between exploration and exploitation, to address the real-world ETP at UNIMAS. Further details of these proposed methods will be discussed in the following chapters.

CHAPTER 3

A CASE STUDY ON THE UNIMAS EXAMINATION TIMETABLING PROBLEM: MODELING AND DATA SPECIFICATION

3.1 Overview

This chapter studies a real-world ETP at UNIMAS, introducing constraints at the university level that have not been formally modeled before. These constraints apply generally across all faculties, with complexity arising from additional preferences from certain faculties. A key contribution of this work is the formal definition of a mathematical model for the UNIMAS ETP. Section 3.2 provides an overview of UNIMAS and its exam timetabling process, while Section 3.3 outlines the current preparation process for final semester examination timetabling. Section 3.4 presents the problem formulation, mathematical model, and specific constraints for the ETP at UNIMAS. The datasets used in this study are discussed in Section 3.5, with concluding remarks presented in Section 3.6.

3.2 UNIMAS Background and Exam Timetabling Process

Established on December 24, 1992, Universiti Malaysia Sarawak (UNIMAS) is Malaysia's eighth university in Kuching, Sarawak. UNIMAS operates from two branches, with its main campus in Kota Samarahan, and another branch in Sibul. As of 2024, UNIMAS comprises ten faculties, serving a total of 13,380 students. These faculties encompass a range of disciplines, including the Faculty of Economics and Business (FEB), Faculty of Social Sciences and Humanities (FSSH), Faculty of Cognitive Sciences and Human Development (FCSHD), Faculty of Engineering (FE), Faculty of Resource Science and Technology (FRST), Faculty of Built Environment (FBE), Faculty of Medicine and Health Sciences (FMHS), Faculty of Language and Communication (FELC), Faculty of Applied and Creative

Arts (FACA), and Faculty of Computer Science and Information Technology (FCSIT). UNIMAS offers a variety of academic programs, including Foundation, Bachelor's, Master's, and PhD programs, with options for both full-time and part-time enrollments. All students enrolled in courses with semester-end examinations must adhere to a unified examination timetable.

UNIMAS employs a dual scheduling approach, combining centralization and decentralization to achieve optimal efficiency in the ETP. The centralized approach facilitated by the Pre-Graduate Studies Division (BPPs) for scheduling Generic Language, General Studies or *Mata Pelajaran Umum* (MPU), and Elective courses. This centralized coordination ensures consistency and coherence across the university for these common courses. On the other hand, the scheduling of courses specific to each faculty's program is decentralized and overseen by the *Jawatankuasa Teknikal Peperiksaan Fakulti* (JTPF) within individual faculties. This decentralization allows faculties to tailor their schedules according to their unique requirements, considering factors such as faculty-specific constraints, faculty resources, and program-specific needs.

The UNIMAS ETP presents various challenges, encompassing constraints associated with limited available space, logistics, and human resources. Each faculty advises against scheduling examinations for other courses during the implementation period of Generic Language, Generic MPU, and Elective courses. There are two examination spaces: shared rooms and rooms within the faculties. BPPs are responsible for providing guidance and information regarding examination locations and periods to each faculty.

The JTPF is responsible for optimizing the utilization of examination slots and rooms according to allocated quotas, with the rotation of slot usage aligned with the days designated

for their respective faculties. Nevertheless, the rate of utilization for shared rooms, assigned to each faculty, may vary due to factors such as fluctuating student numbers, course group sizes, and resource availability at the respective faculties. Resource disparities may arise due to the diversity in room availability and the varying number of scheduled exams per faculty, leading to differential exam durations for distinct faculties.

While most faculties' JTPFs manually generate their respective examination timetables, a few use the FESS proprietary software (Sze et al., 2017) developed by FCSIT. The FESS software employs a two-stage heuristic algorithm: the first stage groups courses to reduce problem size and avoid clashes, while the second stage allocates each course to the best-fit room to maximize utilization and minimize student sectioning. The software has successfully generated the examination timetable; however, it is only capable of evaluating the quality of the solutions it produces at the faculty level. The limitation stems from an incomplete problem formulation, precisely, the absence of a suitable underlying mathematical model to evaluate how well the generated timetable performs. Accordingly, a key objective of this research is to formulate and develop a mathematical model tailored explicitly for UNIMAS ETP.

3.3 Final Semester Examination Timetabling Flow

Figure 3.1 illustrates the flowchart outlining the process of preparing the final semester examination timetable. The processes involved in timetabling examinations include:

- i. Each faculty receives information about examination locations and times from the BPPs or the Centre for Graduate Studies (CGS). A list of information is compiled, and the JTPF initiates the intricate task of drafting the examination timetable for the specific semester.

- ii. The drafted timetable is then submitted to the Deputy Dean of Undergraduates for review. Upon confirmation, the timetable is distributed to all students, prominently displayed on notice boards, and made accessible on the faculty's website for student reference.
- iii. In cases where amendments are necessary, the JTPF addresses student objections, adjusts the draft as needed, and revisits the drafting phase if necessary. After implementing any modifications, the final examination schedule is published to the students.
- iv. The process concludes with compiling lists for exam invigilators, and emails are dispatched to inform lecturers.

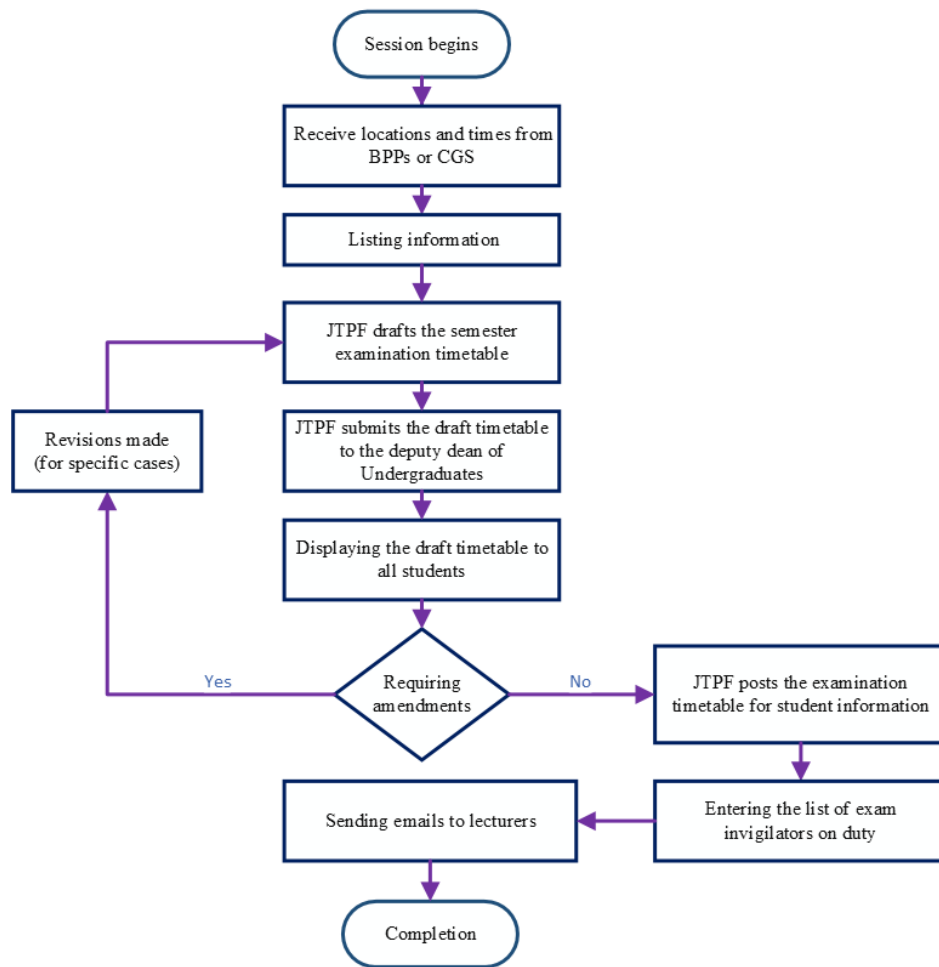


Figure 3.1: Flowchart of Final Examination Timetable Preparation

The process occurs each semester, as students enrol in distinct courses every term. Consequently, the examination timetable applies only to the specific semester in which it is published. As a practical measure, the examination timetabling procedure typically commences several months before the examination period.

3.4 Problem Formulation

ETP is a combinatorial optimization problem. Examinations must be scheduled within a set number of periods and rooms. There are several hard constraints that must be met, and soft constraints that should be minimized (McCollum et al., 2010). These constraints can differ between institutions. Examples and explanations can be found in Burke et al. (1996).

At UNIMAS, the ETP has its own standard constraints. These apply to all faculty exam timetables at the university. Table 3.1 lists notations and variable definitions that formally describe these constraints.

Table 3.1: Notations Employed in for the Problem Formulation of UNIMAS ETP

Notation	Description
N	The number of examinations, $i, j \in \{1, 2, \dots, N\}$, where $(i \neq j)$.
T	The number of timeslots, $k \in \{1, 2, \dots, T\}$.
R	The number of rooms, $r, u \in \{1, 2, \dots, R\}$, where $(r \neq u)$
D	The total duration of the examination period (in days).
A	The number of distinct areas where rooms are located, $a, b \in \{1, 2, \dots, A\}$, where $(a \neq b)$.
R_k	The number of rooms available at timeslot k .
v_r	The total capacity for room r .
u_r	The total unused capacity for room r .
m_i	The number of rooms where exam i has been split.
S	The number of students, $s \in \{1, 2, \dots, S\}$.
s_i	The number of students enrolled in exam i .
x_{ik}	A decision variable indicating the assignment of exam i to timeslot k .
y_{ir}	A decision variable indicating the assignment of exam i to room r .
b_{ia}	A decision variable indicating the assignment of exam i to area a .
c_{ij}	A conflict matrix, where each element c_{ij} , $i, j \in \{1, 2, \dots, N\}$, represents the number of students who are simultaneously taking both exams i and j , $(i \neq j)$.

The model utilizes four binary variables as specified by:

x_{ik} : whether exam i can be scheduled to timeslot k .

$$x_{ik} = \begin{cases} 1, & \text{if exam } i \text{ is assigned to timeslot } k, \\ 0, & \text{otherwise.} \end{cases}$$

y_{ir} : whether exam i can be assigned to room r .

$$y_{ir} = \begin{cases} 1, & \text{if exam } i \text{ is assigned to room } r, \\ 0, & \text{otherwise.} \end{cases}$$

b_{ia} : whether exam i can be assigned room in area a .

$$b_{ia} = \begin{cases} 1, & \text{if exam } i \text{ is assigned to room in area } a, \\ 0, & \text{otherwise.} \end{cases}$$

c_{ij} : whether exam i conflicts with exam j (e.g. share common student).

$$c_{ij} = \begin{cases} 1, & \text{if exam } i \text{ and } j \text{ are in conflict,} \\ 0, & \text{otherwise.} \end{cases}$$

The following are the constraints specific to the UNIMAS ETP, which align with the standard constraints typically applied across all faculties within the university. The notations and variable definitions outlined in Table 3.1 indicate the hard constraint (section 3.4.1) and soft constraint (section 3.4.2) in the UNIMAS ETP, followed by the objective function (section 3.4.3). Finally, extended soft constraints (section 3.4.4) are given special consideration on a case-by-case basis.

3.4.1 Hard Constraints

HC1: Each exam must be allocated to only one timeslot. HC1 mandates that each exam be allocated to only one available timeslot, as specified in Equation 3.1.

$$\sum_{k=1}^T x_{ik} = 1 \quad \forall i \in \{1, \dots, N\} \quad \text{Equation 3.1}$$

HC2: No student is allowed to take two examinations simultaneously. HC2 enforces a conflict-free requirement, wherein if examinations i and j are scheduled in timeslot k , the stipulation is that the number of students sitting for both examinations i and j must be 0, i.e., $c_{ij} = 0$. This stringent constraint is expressed in Equation 3.2.

$$\sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^T x_{ik} x_{jk} c_{ij} = 0 \quad \text{Equation 3.2}$$

HC3: The total number of students assigned to a room does not surpass its capacity. HC3 ensures that the number of students assigned to a room remains less than or equal to its capacity, as Equation 3.3 indicates.

$$s_i \leq \sum_{r=1}^R y_{ik} v_r \quad \forall i \in \{1, \dots, N\} \quad \text{Equation 3.3}$$

HC4: The timeslots used must not exceed the duration of the planned exam session. HC4 ensures that the number of time slots utilized does not exceed the number of days in the planned exam session, with two time slots allocated per day, as indicated in Equation 3.4.

$$\sum_{k=1}^T x_k \leq 2D \quad \text{Equation 3.4}$$

3.4.2 Soft Constraints

SC1: Minimize the number of rooms used to reduce room wastage.

The cost of the count of rooms used in each period k , denoted by r_k , and summing these counts across all timeslots during the examination period in SC1, denoted as z_1 , is illustrated in Equation 3.5, along with Equation 3.6.

$$z_1 = \sum_{k=1}^T r_k \quad \text{Equation 3.5}$$

where

$$\sum_{i=1}^N x_{ik} = r_k \quad \text{Equation 3.6}$$

SC2: Spread each student's exams evenly over the exam period.

The proximity cost in SC2, denoted as z_2 , pertaining to the distribution of exams throughout the exam period, is depicted in Equation 3.7. Equation 3.8 represents the weight of the penalty incurred when two exams, i and j , are scheduled with a gap of $|k_j - k_i|$ timeslots between them. The proximity values utilized in this context (16, 8, 4, 2, and 1) correspond to the formula's logic, where a larger gap between exams results in a decreased penalty. The proximity values introduced by Carter et al. (1996) have been widely adopted in other research studies, including those by Muklason et al. (2017), Kahar & Kendall (2010), Ayob et al. (2007), and Burke et al. (2004).

$$z_2 = \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^N c_{ij} w_{|k_j - k_i|}}{S} \quad \text{Equation 3.7}$$

where

$$w_{|k_j-k_i|} = \begin{cases} 2^{5-|k_j-k_i|} & \text{if } 1 \leq |k_j - k_i| \leq 5 \\ 0 & \text{otherwise} \end{cases} \quad \text{Equation 3.8}$$

SC3: Minimize the number of rooms for a split examination; allocate rooms in the same area (building or campus) when possible.

The splitting cost in SC3, denoted as z_3 , is illustrated in Equation 3.9. Equation 3.10 defines m_i , representing the number of rooms to which exam i has been assigned, while Equation 3.11 defines g_i , representing the number of areas to which exam i has been assigned. In cases where an exam is divided into multiple rooms, a penalty cost is assigned. Additionally, allocating an exam to different areas incurs a penalty of 1 for each additional area beyond the first. This penalty reflects the undesirability of splitting exams across multiple areas due to the increased complexity or inefficiency it introduces. For each additional area beyond the first, a penalty of 1 is added.

$$z_3 = \sum_{i=1}^N [(m_i - 1) + (g_i - 1)] \quad \text{Equation 3.9}$$

where

$$m_i = \sum_{r=1}^R y_{ir} \quad \forall i \in \{1, \dots, N\} \quad \text{Equation 3.10}$$

$$g_i = \sum_{r=1}^R b_{ia} \quad \forall i \in \{1, \dots, N\} \quad \text{Equation 3.11}$$

SC4: Schedule exams within designated timeslots.

The undesired slot cost for scheduling exams outside their designated timeslot $\bar{k}$, denoted as z_4 , is illustrated in Equation 3.12.

$$z_4 \sum_{i=1}^N (1 - x_{i\bar{k}}) \quad \text{Equation 3.12}$$

3.4.3 Objective Functions

According to the mathematical expressions given, the total soft conflict penalty cost for a solution can be formulated using Equation 3.13. X denotes a solution to a single-objective ETP. The objective is to minimize the total cost, denoted as $z(X)$, in a feasible solution X . The penalty coefficients for the soft constraints, labelled as α_1 , α_2 , α_3 , and α_4 , are assigned values of 1, 1, 1, and 2, respectively. The penalty coefficient for α_4 carries a higher penalty compared to other constraints as it requires directly affects the faculty and structure of the exam schedule, which leads to the concept of constraint prioritization, where different weights are assigned to certain constraints, giving them greater importance.

Prioritizing adherence to designated timeslots restricts the flexibility to make improvements or adjustments for individual exams, limiting the options for rescheduling or reallocating resources. Consequently, this restriction amplifies the penalties associated with other soft constraints. In general evaluation, the objective function will solely consider standard soft constraints. ETP is framed as a minimization problem, with a lower objective function value indicating a higher-quality solution.

$$z(X) = \alpha_1 \cdot z_1 + \alpha_2 \cdot z_2 + \alpha_3 \cdot z_3 + \alpha_4 \cdot z_4 \quad \text{Equation 3.13}$$

3.4.4 Extended Soft Constraints

The extended soft constraints are intended to accommodate the specific preferences of individual faculties or exam officers in faculty exam timetabling, which vary on a case-by-case basis. Incorporating extended soft constraints enables greater flexibility in the scheduling process, ensuring that it aligns more effectively with faculty-specific demands and results in a more efficient and satisfactory timetable. The extended soft constraints are as follows:

- i. SE1: Faculty Exam Session Flexibility– Faculties can shorten their exam sessions if they adhere to the overall duration allocated by the university for exams.
- ii. SE2: No Room Sharing – Some faculties prohibit multiple exams from sharing a room, utilizing it as a unary resource. However, this policy does not apply if the exams are combined.
- iii. SE3: Balance Room Allocation for Split Exam – Ensuring fair distribution of students across rooms for split exams.

3.5 Datasets

The investigations utilized real-world datasets from two semesters across two faculties at UNIMAS:

- i. Semester 1 of 2022/2023 (S1-2022/23): Faculty of Economics and Business (FEB23) and Faculty of Cognitive Sciences and Human Development (FCSHD23).

- ii. Semester 2 of 2023/2024 (S2-2023/24): Faculty of Economics and Business (FEB24) and Faculty of Computer Science and Information Technology (FCSIT24).

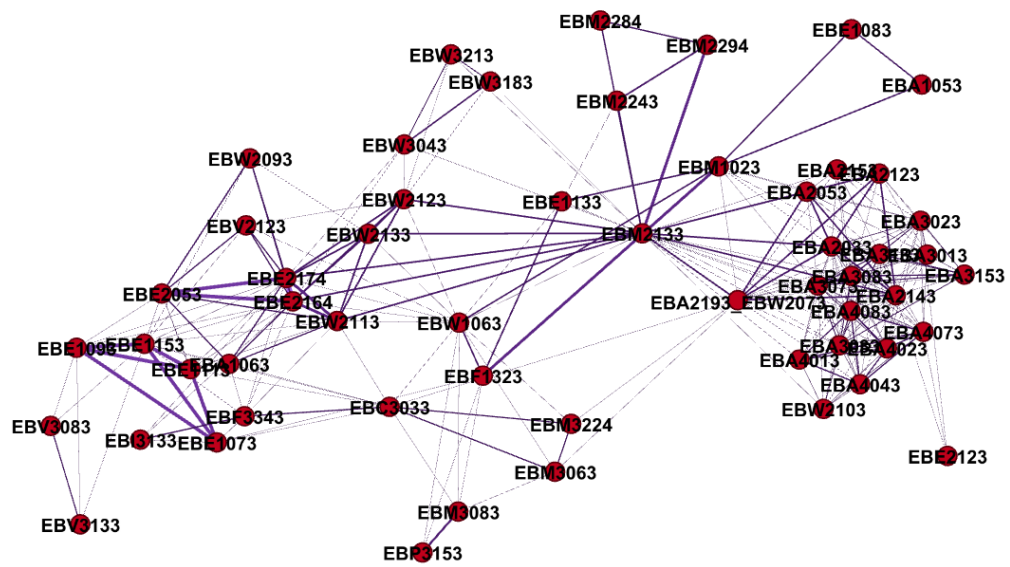
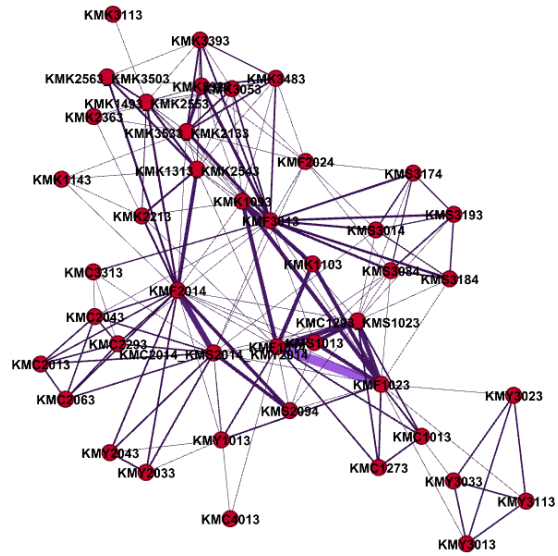
Table 3.2 summarizes the real-world dataset features, including the number of exams, number of students, enrolments (student–exam registrations), conflict density, examination types, room sharing, and room availability. The room availability count specifies the distribution of available rooms across three size categories—small (<100), medium (100–150), and large (>150)—for each instance. In the context of room resources, two types of rooms are available: (1) small-sized faculty rooms, available in every timeslot, and (2) medium to large-sized shared rooms. The latter's availability varies and is allocated an event-proportionate ratio among ten faculties, as determined by the BPPs based on the number of exams and student size. All four instances are characterised by a predominance of small rooms, a scarcity of medium-sized rooms, and very limited large rooms. This distribution suggests timetabling is constrained by the scarcity of larger spaces.

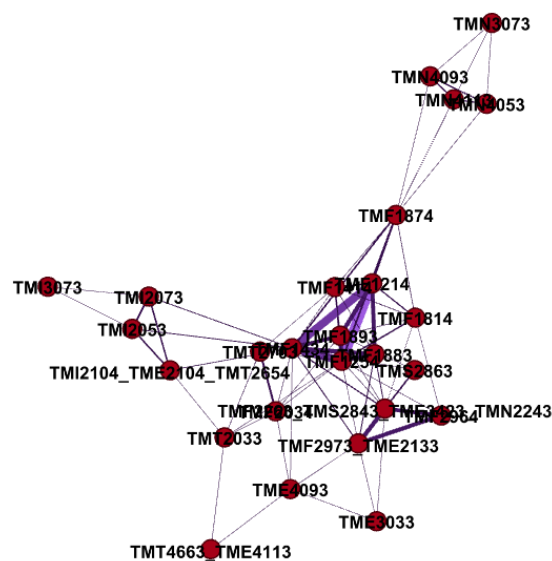
The scheduling involves two available timeslots daily throughout the exam session in allocated rooms. Some instances include both online and physical exam types. Online exams are uncapacitated and administered via the eLEAP system, while physical exams are capacitated and require room management. Among all the faculties at UNIMAS, it is noteworthy that FEB and FCSHD—ranking as the faculties with the largest and second-largest student populations, respectively—present a particular challenge. Numerous courses in both faculties have substantial enrollments, each exceeding 400 students, which surpasses the maximum room capacity of 300 for final exams. As a result, it is necessary to split these examinations and conduct them across multiple venues.

Table 3.2: The Characteristics of Problem Instances from Real-World Dataset

	Semester 1 of 2022/2023		Semester 2 of 2023/2024	
	FEB23	FCSHD23	FEB24	FCSIT24
Share Room	Yes	No	Yes	Yes
Exam Type	Online & Physical	Physical	Physical	Online & Physical
Exam	40	52	55	31
Student	1,921	1,734	2,011	1,026
Enrolment	6,026	6,399	7,035	2,837
Conflict Density	0.251	0.144	0.182	0.234
Room Availability Count	Small: 114 Medium: 24 Large: 10	Small: 112 Medium: 25 Large: 14	Small: 65 Medium: 24 Large: 13	Small: 120 Medium: 15 Large: 7

In four instances, the number of exams ranges from 31 to 55, with the corresponding student enrolments varying from 1,026 to 2,011. The total enrolment across the instances spans from 2,837 to 7,035 students, reflecting the diversity in the scale and complexity of each dataset. Table 3.2 suggests that FEB24 has the highest enrolment, while instance FCSIT24 has the fewest exams and enrolments. These instances also vary in their exam types and shared room preferences. Instances FEB23 and FCSIT24 involve both online and physical exams, with shared rooms not allowed. Instance FCSHD23 consists solely of physical exams and prohibits room sharing, whereas instance FEB24, although also limited to physical exams, permits room sharing.





3.5.1 Semester 1 of 2022/2023

In session S1-2022/23, FEB23 has scheduled 40 exams involving 1,921 students and 6,026 enrolments. The adjacency matrix for FEB is 40x40, and the system's graph displays 196 edges, indicating a conflict density of 0.251, which accounts for 25.1% of all possible conflicts between the exams. Conversely, the FSCHD has 52 exams, engaging 1,734 students and 6,399 enrolments. The adjacency matrix for FSCHD23 is 52x52, and the system's graph exhibits 184 edges, representing a conflict density of 0.144, which means that 14.4% of all possible conflicts between the exams are present. While FEB aims to minimize room usage by allowing different exams to share rooms, FCSHD prefers not to share rooms for any exams. Both faculties generated their exam timetables using FESS proprietary software.

3.5.2 Semester 2 of 2023/2024

In session S2-2023/24, FEB24 has scheduled 55 exams involving 2,011 students and 7,035 enrolments. The adjacency matrix for FEB24 is 55x55, and the system's graph displays 540 edges, indicating a conflict density of 0.182, meaning that 18.2% of all possible conflicts between the exams are present. Conversely, the FCSIT24 has 31 exams, engaging 1,026 students and totaling 2,837 enrolments. The adjacency matrix for FCSIT24 is 31x31, and the system's graph exhibits 152 edges, representing a conflict density of 0.234, which means that 23.4% of all possible conflicts between the exams are present. FEB and FCSIT aim to minimize room usage by sharing different exams in rooms. FEB generated its exam timetables using the FESS software, whereas FCSIT's timetables were created manually.

3.5.3 Benchmark Dataset

Carter et al. (1996) provided benchmark examination timetabling instances from real-world datasets, commonly referred to as the Toronto dataset. This dataset represents one of the most thoroughly examined problem instances over the past decade, owing to its complexity and the challenges posed by its diverse instances, each characterized by unique attributes, as discussed in Chapter 2. Toronto benchmark shares the same hard constraints on exam clashes and soft constraints on exam spread as our formulation. Additionally, the datasets, ranging from 100 to over 500 exams and generating thousands of possible timeslot combinations, present significant computational challenges, making it well-suited for evaluating diverse timetabling scenarios and providing high complexity for robust testing. As a result, the proposed algorithm was evaluated using the Toronto dataset.

3.6 Research Methodology

The research methodology in this study is outlined as follows:

Step 1: Data Collection and Requirement Analysis

This step involves gathering data such as exam course lists, student enrollments, guidelines for slot and room allocation, and the official timetable from recent semesters. An interview with the examination administrator was conducted, and preferences stated in email correspondence from the administrator were reviewed to capture requirements, ensuring that all relevant constraints, preferences, and institutional policies are understood.

Step 2: Requirement Analysis and Problem Formulation

The gathered data and elicited requirements are thoroughly analyzed to identify hard constraints, soft constraints, optimization goals, and priorities for timetabling. Following this, the problem is mathematically formulated by defining variables, constraints, and an objective function to quantify soft constraint violations, establishing a structured basis for guiding the algorithm's development.

Step 3: Algorithm Development

The algorithms were developed and executed in Java within the IntelliJ IDEA integrated development environment. A prototype of the constructive algorithm is developed based on hard constraints. This phase emphasizes rapid prototyping, allowing early testing and validation before refining the algorithms further. The multi-neighborhood local search algorithms were developed to optimize timetables.

Step 4: Iterative Testing and Algorithm Refinement

The algorithm undergoes rigorous testing in multiple iterations, initially using real datasets from UNIMAS to evaluate its performance in a practical setting against the objective function defined in Step 2. The results from each test are analyzed to identify conflicts or inefficiencies, and the algorithms are refined accordingly for continuous improvement.

Step 5: Result Compilation and Comparative Analysis

Generated timetables are compiled and analyzed based on metrics such as computational time and objective function values. A comparative analysis is conducted to evaluate the algorithm's effectiveness and identify areas for improvement by comparing its results with those of existing proprietary software and manual scheduling methods.

Step 6: Algorithm Validation

The top-performing algorithm is selected and adapted to assess its validity and effectiveness by applying it to the Toronto benchmark problem, evaluating its performance against established objective functions, and comparing the results with those of other approaches in the literature.

Step 7: Reporting

The final step involves compiling and documenting research findings in a thesis report, which includes detailed discussions of the mathematical model, implemented solutions, compiled results, and evaluations.

3.7 Conclusion

This chapter presents a summary of the background of the UNIMAS ETP, offering a description of the examination constraints. The ETP has been elucidated through the formulation of mathematical models. Given the decentralized faculty timetabling practices at UNIMAS, the constraints are further categorized into two levels: university-level constraints, which are adhered to by all faculties, and faculty-specific constraints, which are tailored to the faculty or exam officer's preferences. The objective function is designed to ensure generality and facilitate comparison by including university-level soft constraints to optimize the timetable. Conversely, faculty-specific constraints that are contradictory to the objective function are excluded from its evaluation but are still considered within the faculty timetabling process.

This work contributes by introducing and systematically documenting a new examination dataset that incorporates extended constraints, distinguishing it from existing benchmark or real-world problem datasets. Since these constraints had not previously been formalized at UNIMAS in a mathematically structured manner, this study further contributes by developing a mathematical model to formally define the UNIMAS ETP. Two instances from faculties with distinct characteristics were collected per semester, resulting in a total of four instances across two semesters. These instances represent the diverse constraints specified in the problem. The real-world dataset underwent thorough preprocessing and data cleaning to ensure consistency and accuracy.

CHAPTER 4

VARIABLE NEIGHBORHOOD DESCENT APPROACH FOR UNIMAS EXAM TIMETABLING PROBLEM

4.1 Overview

Based on the literature reviewed in Chapter 2, metaheuristics have become increasingly prominent for solving ETP. These methods have shown significant advantages when combined with well-designed neighborhood structures. Therefore, this study focuses on developing an efficient metaheuristic emphasizing proper neighborhood structures. Specifically, an improved Variable Neighborhood Descent (VND) approach was introduced to optimize the objective function defined in the previous chapter while satisfying the extended soft constraints for a real-world ETP.

This section provides background on the faculty exam timetabling selected for this study. Section 4.2 addresses the preprocessing stage, while Section 4.3 discusses the constructive algorithm employed to generate the initial solution. Section 4.4 provides a detailed description of the proposed VND approach. Section 4.5 presents experimental results and analysis based on real-world datasets, followed by a statistical analysis of neighborhood structure in Section 4.6. A comparison of centralized and decentralized timetabling with post-scheduling shared room reallocation is discussed in Section 4.7. Section 4.8 compares the VND algorithm, its variants, and improved versions, along with the experimental results, while Section 4.9 summarizes the study's key findings.

In the field of examination timetabling, both uncapacitated and capacitated ETPs are studied, with capacitated variants generally considered more complex due to the need to

account for room-related capacities. Most real-world exam timetabling challenges focus on capacitated variants because they better reflect the intricacies of actual scheduling scenarios.

Besides satisfying hard constraints, the examination timetabling at UNIMAS must optimize the solution based on an objective function that considers exam spread, room usage, room allocation, and preferred time slots. ETP at UNIMAS is further complicated by the need to manage an exam timetable that incorporates online (uncapacitated) exams and physical (capacitated) exams within a single timetable. Additionally, room-sharing preferences vary across faculties at UNIMAS; some prohibit it, while others encourage it to minimize resource wastage, complicating the situation further.

All these challenges were addressed, the objective function was optimized within the proposed solution method, and experiments were conducted to compare its effectiveness with the current methods employed by individual faculties, whether they utilize proprietary software or manual techniques. Firstly, a constructive algorithm was employed to generate a feasible solution by prioritizing the scheduling of online exams, followed by physical exams, and then refining this solution using the VND approach.

4.2 Preprocessing Stage

The preprocessing stage involves ordering exams using the Largest Enrollment First (LE) strategy to ensure the efficient allocation of space, and then sorting resources, including total area and room capacity, in non-decreasing order. During scheduling, the allocation of exams to rooms impacts the remaining capacity, necessitating dynamic updates to the order of balancing capacity for both areas and rooms. Sorting in non-decreasing order aligns with a “best fit” strategy by ensuring that areas with smaller room counts are considered first, maximizing utilization of available resources.

4.3 Constructive Algorithm

The constructive algorithm is employed to create an initial feasible solution. The goal of this algorithm is to satisfy all hard constraints (HC1-HC4), soft constraints (SC4) related to priority exam slot preferences, and the extended soft constraint (SE1) of minimizing the number of timeslots utilized. The construction algorithm unfolds in two phases:

As illustrated in Figure 4.1, the constructive algorithm first sorts *OnlineExams* and *PhysicalExams* using Largest Enrolment *LE* heuristic and arranges rooms *R* in non-decreasing order of capacity. In Phase I, each *OE_exam[i]* is assigned to a feasible timeslot if available; otherwise, it is added to *unscheduledList*. In Phase II, for each *PE_exam[j]*, a list of feasible timeslot–room options *Options* is generated; if multiple options exist, the one with minimum *ResourceCost* is selected. If no feasible allocation exists, the exam is moved to *unscheduledList*.

The algorithm initially focuses on the timeslots available during the first week of the exam period to minimize the number of timeslots utilized. If the available timeslots cannot accommodate all exams, the algorithm extends the scheduling by incrementally adding one day with each attempt until all exams are successfully assigned. This approach prioritizes assigning exams to available single rooms for efficient room allocation, thereby minimizing the overall number of rooms utilized. In cases where single rooms are unavailable, the approach utilizes split rooms to minimize the number of splits and the need for cross-area allocations.

Algorithm 4.1 Constructive Algorithm

```

1: Input: OnlineExam, PhysicalExam
2: Output: Initial feasible solution
3: Pre-processing Step:
4: Sort online exams OnlineExams in OExams[] based on LE
5: Sort physical exams PhysicalExams in PExams[] based on LE
6: Sort rooms R in non-decreasing order of total room capacity

7: Phase I - Assigning Online Exams to Timeslots:
8:  $i \leftarrow 0$ 
9: while  $i < \text{size}(\text{OExams})$  do
10:   Schedule OExams[ $i$ ] to a feasible timeslot
11:   Pop OExams[ $i$ ] from OExams
12:   if OExams[ $i$ ] cannot be scheduled then
13:     Add OExams[ $i$ ] to unscheduledList
14:   end if
15: end while

16: Phase I - Assigning Physical Exams to Timeslots and Rooms:
17:  $j \leftarrow 0$ 
18: while  $j < \text{size}(\text{PExams})$  do
19:   Find list Options of feasible timeslots for PExams[ $j$ ]
20:   for each  $t \in \text{Options}$  do
21:     if there exists one or more rooms  $r \subseteq R$  available then
22:       Set  $\text{ResourceCost}[t] \leftarrow \text{RoomCost} + \text{SplitCost} + \text{AreaCost}$ 
23:     end if
24:   end for
25:   Schedule PExams[ $j$ ] in  $t$  and  $r$  with the minimum ResourceCost
26:   Pop PExams[ $j$ ] from PExams
27:   if PExams[ $j$ ] cannot be scheduled then
28:     Add PExams[ $j$ ] to unscheduledList
29:   end if
30: end while

31: Phase II - Repair Infeasible Solution:
32:  $k \leftarrow 0$ 
33: while  $k < \text{size}(\text{unscheduledList})$  do
34:   Apply ejection and insertion moves to unscheduledList[ $k$ ]
35:   if unscheduledList[ $k$ ] can be scheduled then
36:     Pop unscheduledList[ $k$ ] from unscheduledList
37:   end if
38: end while

```

Figure 4.1: Constructive Algorithm

In these experiments, a feasible solution is not assured for minor cases, necessitating Phase II. If exams cannot be scheduled due to conflicts that violate constraints, the conflicting exams are kept in the *unscheduledList*. Phase II addresses infeasibility and ensures that all unscheduled exams are scheduled. A repair mechanism utilizing the neighborhoods of Insertion Move and Ejection Move has been carried out as follows:

- i. Insertion Move denoted as $IM(e, p, r)$, takes as attribute one exam, $e \in E$,

one period, $p \in P$, and one room $r \in R$. This move involves inserting an exam e from the *unscheduledList* into a random period, p and available room, r , without causing any conflict violations.

Preconditions: At least one unscheduled exam, e is in the *unscheduledList*.

- ii. Ejection Move (Ding, 2019), denoted by $EM(e)$, takes as attribute one exam $e \in E$. The move involves removing exam e from the *unscheduledList*.

Preconditions: If $IM(e, p, r)$ is successful.

Both neighborhoods proved successful in ensuring feasibility in Phase II; however, they fell short in minimizing the objective function value.

4.4 Variable Neighborhood Descent

The solution method begins with the final state of the constructive algorithm and focuses on refining the solution within the feasible region to achieve optimal results, considering other soft constraints (SC1–SC3) and any applicable extended soft constraints.

4.4.1 Neighborhood Structures

The neighborhood structure, which outlines permissible moves between solutions, is the fundamental mechanism of local search. Six neighborhoods were introduced, focusing exclusively on exploring the feasible portion of the search space.

- N1. Move Exam, denoted by $SM(e, p, r)$, takes as attribute one exam $e \in E$ and one period $p \in P$. It involves assigning e to a randomly selected period p and room(s) r , with $p(e) \neq p$. The Move Exam using the same room is

presented in Figure 4.2, while Move Exam with a reduced number of rooms is shown in Figure 4.3.

Conditions: Occurs if the size of $r \leq r(e)$.

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1	Exam 1	Exam 1		
TIMESLOT 2				
TIMESLOT 3	Exam 1	Exam 1		
TIMESLOT 4				

Figure 4.2: Execution of the Move Exam with Same Rooms

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1	Exam 1	Exam 1		
TIMESLOT 2				
TIMESLOT 3			Exam 1	
TIMESLOT 4				

Figure 4.3: Execution of the Move Exam with Reduced Room

N2. Swap Exams, denoted by $SE(e_i, e_j)$, takes as attributes two exams $e_i, e_j \in E$ where $e_i \neq e_j$. It involves swapping e_i within period $p(e_i)$ and rooms $r(e_i)$ with e_j within period $p(e_j)$ and rooms $r(e_j)$. The Swap Exams using the same room is presented in Figure 4.4, while Swap Exams between rooms is presented in Figure 4.5.

Conditions: Occurs if the size of the selected $r \leq r(e)$.

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1				
TIMESLOT 2				
TIMESLOT 3				
TIMESLOT 4				

Figure 4.4: Execution of the Swap Exams with Same Room

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1	Exam 2	Exam 1	Exam 1	
TIMESLOT 2				
TIMESLOT 3		Exam 2		Exam 1
TIMESLOT 4				

Figure 4.5: Execution of Swap Exams with Reduced Room

N3. Kick Exam (Di Gaspero, 2002), denoted by $KE(e_i, e_j, p)$, takes as attributes two exams $e_i, e_j \in E$ where $e_i \neq e_j$ and one period $p \in P$ with $p(e_i), p(e_j) \neq p$. It involves assigning e_i to $p(e_j)$, and assigning e_j to a randomly selected period p . The Kick Exam with the same room is shown in Figure 4.6, and the Kick Exam with a reduced room is shown in Figure 4.7.

Conditions: Occurs if the size of the selected $r \leq r(e)$.

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1	Exam 1			
TIMESLOT 2	Exam 1		Exam 2	
TIMESLOT 3			Exam 2	
TIMESLOT 4				

Figure 4.6: Execution of Kick Exam with Same Room

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1	Exam 1	Exam 1		
TIMESLOT 2			Exam 1	Exam 2
TIMESLOT 3				Exam 2
TIMESLOT 4				

Figure 4.7: Execution of Kick Exam with Reduced Room

N4. Shift Room, denoted by $SR(e, r)$, takes as attribute one exam $e \in E$ and one or more rooms $r \in R$. It involves assigning e to a randomly selected

room or set of rooms r , considering only rooms in the same area. The Shift Room is illustrated in Figure 4.8.

Conditions: Occurs if capacity of $r < r(e)$, and $p(r) \neq p(r(e))$.

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1		Exam 1		
TIMESLOT 2				
TIMESLOT 3	Exam 1	Exam 1		
TIMESLOT 4				



Figure 4.8: Execution of Shift Room

N5. Compact Room, denoted by $CR(e, r)$, takes as attribute one exam $e \in E$ and one or more rooms $r \in R$. It involves assigning e in to a randomly selected r within the same period and non-empty room(s) in the same area. The Compact Room is shown in Figure 4.9.

Conditions: The room sharing is allowed. Occurs if $p(r) = p(r(e))$ and capacity of $r < r(e)$.

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1				
TIMESLOT 2				
TIMESLOT 3	Exam 1	Exam 1		Exam 1 Exam 2
TIMESLOT 4				



Figure 4.9: Execution of Compact Room

N6. Downsize Room, denoted by $DR(e, r)$, takes as attribute one exam $e \in E$, and one or more rooms $r \in R$. It involves assigning e in to a randomly selected r within the same period and smaller sized empty room(s). The Downsize Room with the same number of rooms used is shown in Figure 4.10, while the Downsize Room with fewer room usage is presented in

Figure 4.11.

Conditions: Sharing a room is prohibited. Occurs if $p(r) = p(r(e))$ and the capacity of $r < r(e)$.

	ROOM 1 (100)	ROOM 2 (50)	ROOM 3	ROOM 4
TIMESLOT 1	Exam 1 (45)	Exam 1 (45)		
TIMESLOT 2				
TIMESLOT 3				
TIMESLOT 4				

Figure 4.10: Execution of Downsize Room with Same Number of Rooms

	ROOM 1	ROOM 2	ROOM 3	ROOM 4
TIMESLOT 1	Exam 1	Exam 1		Exam 1
TIMESLOT 2				
TIMESLOT 3				
TIMESLOT 4				

Figure 4.11: Execution of Downsize Room with Reduced Number of Rooms

For ease of reference, the neighborhoods associated with the six moves above are symbolized as N_1, N_2, N_3, N_4, N_5 , and N_6 , respectively. $N_1 \sim N_3$ are similar to the neighborhood proposed by Bellio (2021) but are adapted here to include room constraints. $N_4 \sim N_6$ correspond to room-related moves, while $N_1 \sim N_3$ pertain to both period-related moves and room-related moves. Due to the simplification of constraints in $N_2 \sim N_5$, the search space of these neighborhoods is slightly smaller than that of N_1 and N_6 . Consequently, the search in neighborhoods $N_2 \sim N_5$ is much faster than that in N_1 and N_6 .

4.4.2 Variable Neighborhood Descent Algorithm

VND is a metaheuristic systematically modifies the neighborhood structure to find optimal or near-optimal solutions. VND and its variants have proven to be highly effective

in solving various challenges within combinatorial and global optimization (Mladenovic & Hansen, 2017).

Each VND consists of two main components. The first component of the VND involves a local search strategy that explores the neighborhood structure $N(X)$ surrounding the current solution X during each iteration. This phase can employ the best improvement (steepest descent) or first descent (first descent) search strategy. Steepest descent explores the entire neighborhood $N(X)$ at each step, offering a thorough search but at a high computational cost. In contrast, the first descent heuristic makes a move immediately upon identifying a descent direction, trading off thoroughness for faster, less computationally intensive progress. The local search process begins by selecting an initial solution X , then identifying a descent direction within the neighborhood $N(X)$, and proceeding toward a solution in $N(X)$ that yields a lower objective function value along that direction. The search terminates if no descent direction exists; otherwise, the process is iterated.

The second component focuses on modifying the neighborhood structure, wherein the next neighborhood to explore is chosen, and an acceptance or rejection criterion is used to evaluate the newly found solution. VND leverages the principle that a solution reaching local optimality across several neighborhood structures has a higher chance of approximating the global optimum than one optimized within just a single neighborhood. The neighborhood $N(X)$ of X is explored thoroughly at each step. More specifically, the VND procedure explores several neighborhood structures either sequentially or proceeds through various steps (Hansen & Mladenovic, 2017), utilizing different rules for choosing the following neighborhood structure to improve the incumbent solution, such as:

- i. Basic Sequential Neighborhood Change Step: If the current solution is

improved within a given neighborhood structure, the search proceeds from the first neighborhood structure following the predefined sequence of the new incumbent solution. If no improvement is found, the search moves on to the next neighborhood structure in the established order.

- ii. **Cyclic Neighborhood Change Step:** The search progresses to the following neighborhood structure in the list, whether or not an improvement is found in the current neighborhood.
- iii. **Pipe Neighborhood Change Step:** If the current solution is improved within a particular neighborhood, the exploration in that neighborhood continues.
- iv. **Skewed Neighborhood Change Step:** New incumbent solutions are accepted if they either enhance or deteriorate the current solution. This approach facilitates the exploration of valleys beyond the incumbent solution. A trial solution is assessed by considering its objective values relative to the incumbent solution and measuring the distance between them.

Among all neighborhoods changing rules, the pipe method achieves the fastest speed while delivering competitive solution quality. Consequently, it is adopted in this work (Hansen & Mladenovic, 2017). Thus, the Pipe Variable Neighborhood Descent (PVND) algorithm, which employs the pipe neighborhood change step is developed for addressing the ETP, as shown in Figure 4.12. The statement $\text{LocalSearch}(X, N_k)$ implies that local search is conducted within the current neighborhood structure N_k , using X as the starting solution. Figure 4.13 illustrates the local search step. A first-improvement strategy is employed across all neighborhoods, meaning that the corresponding move is carried out whenever an improved solution is identified within a given neighborhood, as outlined in Section 4.4.1.

The loop systematically explores neighborhood structures in sequence: for each neighborhood N_k , it applies LocalSearch to generate a new solution X' ; if X' improves the objective function value with $Z(X') < Z(X)$, the current solution X is updated to X' and the search continue intensifying in the same neighborhood, otherwise the search advance to the next neighborhood with $k \leftarrow k + 1$. This process repeats iteratively until the maximum iteration limit, $MaxIter$ is reached.

Algorithm 4.2 Pipe Variable Neighborhood Descent (PVND)

```

1: Input: Neighborhood structures  $N$ , initial solution  $X$ 
2: Output: Best solution  $X$  found
3:  $k \leftarrow 1$ 
4:  $iter \leftarrow 0$ 
5: while  $iter < MaxIter$  do
6:   while  $k \leq k_{max}$  do
7:      $X' \leftarrow \text{LOCALSEARCH}(X, N_k)$ 
8:     if  $Z(X') < Z(X)$  then
9:        $X \leftarrow X'$ 
10:    else
11:       $k \leftarrow k + 1$ 
12:    end if
13:  end while
14:   $iter \leftarrow iter + 1$ 
15: end while
16:  $X \leftarrow \text{BALANCER}(X)$ 
17: return  $X$ 

```

Figure 4.12: Procedure of PVND

Algorithm 4.3 LOCALSEARCH(X, N_k)

```

1: Generate a random permutation  $queue$  of all the exams
2:  $j \leftarrow 0$ 
3: while  $j < exam\_size$  and time limit not reached do
4:    $X' \leftarrow$  applying neighborhood  $N_k(X)$  to  $queue_j$ 
5:   if  $Z(X') < Z(X)$  then
6:      $X \leftarrow X'$ 
7:     break
8:   end if
9:    $j \leftarrow j + 1$ 
10: end while
11: return  $X$ 

```

Figure 4.13: Procedure of the Local Search

Two different sets of neighborhood structures are utilized based on whether sharing is allowed. When room sharing is permitted, the algorithm explores neighborhoods $N_1 \sim N_5$, whereas when sharing is prohibited, it resorts to neighborhoods $N_1 \sim N_4$ and N_6 . Moves that violate hard constraints are rejected, while those adhering to specific soft constraints aim to improve elements related to other violations. The algorithm continually evaluates the objective function $Z(X)$ to identify the incumbent solution and update the current solution accordingly. They are rejected if they result in a deterioration of the objective function value.

4.4.3 Room Split Balancer

To fulfil the extended soft constraint SE3, which ensures balanced room allocation for split exams, distributing students evenly across rooms helps prevent overcrowding and underutilization. Currently, the splitting method follows a sequential approach. For example, with 205 students split across rooms A, B, and C, rooms A and B hold 100 students, leaving only 5 in room C. This uneven distribution can be confusing, as the small group in room C often struggles to locate the correct room, leading to delays. Students who arrive at the wrong room must be redirected, wasting valuable time and increasing stress during the exam process.

To address this, the exam administration enforces SE3 for balanced student distribution. Similar work by Laporte and Desroches (1984) utilized a room allocation subroutine where each room hosts a single exam. In this approach, larger exams are split across multiple rooms, ensuring even student distribution using a room balancer, but only when room sharing is prohibited. In contrast, a balancer that accommodates shared and non-shared exams is integrated for more effective distribution.

The Room Split Balancer Algorithm, as outlined in Figure 4.14, is designed explicitly for exams allocated in split rooms. It is executed after the completion of the PVND algorithm, and the objective function value remains unaffected. The reallocation for each split room is determined by multiplying the room's capacity by the ratio of its total capacity to the overall total capacity of all rooms, scaled by the exam size. Any remaining residuals are allocated to rooms with higher capacity.

Algorithm 4.4 BALANCER(X)

```

1: for each exam do
2:   if number of rooms allocated for the exam > 2 then
3:      $remaining\_balance \leftarrow examSize$ 
4:      $totalCapacity \leftarrow$  calculate the total capacity of  $rleRoom \triangleright rleRoom$ 
       is the rooms used by the exam
5:     for each room in  $rleRoom$  do
6:        $newAllocation \leftarrow roomCapacity / totalCapacity \times examSize$ 
7:       update room allocation for exam
8:     end for
9:     Allocate residual to room with highest capacity
10:   end if
11: end for
12: return  $X$ 

```

Figure 4.14: Procedure of the Balancer

4.5 Experimental Result

The proposed algorithms were implemented on the real-world dataset described in Chapter 3. The timetable for that semester was generated using either proprietary software or a manual method. The performance of PVND is therefore evaluated in comparison with these existing methods.

4.5.1 Experimental Setup

The proposed algorithms were implemented using Java in IntelliJ IDEA. All experiments were conducted on a Windows 11 laptop equipped with an Intel(R) Core (TM)

i7 CPU at 2.80 GHz and 16 GB of RAM. A preliminary test was conducted using the proposed algorithm, which was executed 30 times on each instance, resulting in the graph shown in Figure 4.15. This graph illustrates the objective function value over each iteration. Based on the convergence patterns observed in the graph, all the instances converged at iteration 8 or 9. The improvement pattern for each instance in Table 4.1 reveals that iterations 10-15 exhibited complete stagnation, with zero total improvement and zero average improvement per iteration, confirming that no further optimization occurred beyond iteration 9. Based on the graph in Figure 4.15 and the improvement pattern in Table 4.1, the algorithm converged at iterations 9–10; hence, 10 iterations were used.

Table 4.1: Improvement Pattern for Real-World Datasets

Instances	Iteration Range	Total Improvement	Average Value per Iteration	Pattern
FEB23	1–3	9.514	4.757	Rapid decline
	4–6	0.554	0.185	Slow decline
	7–9	3.727	1.242	Optimization
	10–15	0.000	0.000	Stagnation
FCSHD23	1–3	18.864	9.432	Rapid decline
	4–6	7.551	2.517	Slow decline
	7–9	5.866	1.955	Optimization
	10–15	0.000	0.000	Stagnation
FEB24	1–3	19.258	9.629	Rapid decline
	4–6	0.249	0.083	Slow decline
	7–9	1.041	0.347	Optimization
	10–15	0.000	0.000	Stagnation

Table 4.1 continued

FCSIT24	1–3	28.838	14.419	Rapid decline
	4–6	1.078	0.359	Slow decline
	7–9	1.000	0.333	Optimization
	10–15	0.000	0.000	Stagnation

**Figure 4.15:** Convergence Curves of PVND for Real-World Datasets

4.5.2 Comparison of PVND with Proprietary Software and Manual Method

Some faculties, such as FEB and FCSHD, utilize FESS proprietary software, and FCSIT schedules are manually done. The FESS proprietary software (Sze et al., 2017) employs a two-stage heuristic: first, it clusters courses for concurrent examination and then allocates the clusters to specific timeslots and rooms. The solution from the proposed approach was compared with the existing solutions generated by proprietary software or manual methods using the objective function outlined in Chapter 3.

The time slots used for all instances were standardized to align with those generated by proprietary software or manual method, ensuring fair comparison. PVND was applied using the initial solution produced by the constructive algorithm. Table 4.2 shows the objective function value and a detailed breakdown of the best and mean objective values for each soft constraint (SC1–SC4). It compares proprietary software and manual methods with PVND across four instances: FEB23, FCSHD23, FEB24, and FCSIT24.

The initial objective values for the four instances studied, generated using the constructive algorithm, were 126.1 for FEB23, 191.1 for FCSHD23, 156.9 for FEB24, and 74.1 for FCSIT24. Although the constructive algorithm produced feasible solutions, the objective values remained significantly higher compared to both the current solutions and those derived from the PVND method. The experimental results in Table 4.2 show that PVND consistently achieves a lower objective function value and lower mean objective values across most soft constraints (SC1 to SC4) compared to FESS. Though excelling in certain soft constraints led to higher objective values in others, PVND minimized the objective function value across all instances.

Table 4.2: Comparison Results for Studied Datasets

SC	Objective value	Instances							
		FEB23		FCSHD23		FEB24		FCSIT24	
		FESS	PVND	FESS	PVND	FESS	PVND	Manual	PVND
SC1	Best	58	59	78	72	74	65	27	23
	Mean	-	61.0	-	75.2	-	66.9	-	25.8
SC2	Best	18.0	21.2	19.7	12.6	23.6	22.7	18.6	4.7
	Mean	-	23.2	-	16.1	-	24.2	-	6.9

Table 4.2 continued

SC3	Best	40	31	32	22	25	26	10	7
	Mean	-	31.5	-	25.6	-	27.5	-	7
SC4	Best	16	6	4	6	26	16	0	0
	Mean	-	6	-	6	-	16	-	0
Total	Best	132.0	118.5	133.7	114.4	148.6	132.1	55.6	36.8
	Mean	-	123.3	-	126.8	-	134.7	-	39.7

Bold indicates the best objective function value. SC = Soft Constraint

For FEB23, the resulting solution of 118.5 represents a 10% enhancement compared to the proprietary software's solution of 132.0. Similarly, for FCSHD23, the solution improved to 114.4, demonstrating a 14% reduction from 133.7. In FEB24, the method achieved a solution of 132.1, 11% better than the 148.6 provided by the proprietary software. Lastly, for FCSIT24, the solution achieved a score of 36.8, representing a 33% improvement over the 55.6 score obtained by the manual method. Using PVND, notable improvements were achieved compared to existing results.

4.5.3 Extended Soft Constraints

Experiments are conducted to evaluate the proposed methods for enforcing extended soft constraints. Each extended soft constraint is further discussed in the following subsections, demonstrating their effectiveness in enforcement for selected instances.

4.5.3.1 Shorten Exam Session

Apart from optimizing the objective function value, all the studied faculties aim to satisfy the SE1 on Flexible Exam Session constraint. Certain faculty exam administrators desire to shorten the duration of their exam sessions instead of strictly adhering to the exam

timeframe determined by the University. This adjustment enables students to complete their exams sooner while allowing lecturers to allocate their time more effectively for grading and providing feedback. However, implementing this adjustment is challenging due to limited room constraints.

The experiment involved applying the constructive algorithm to all the studied instances to evaluate the effectiveness of the proposed approach in achieving the SE1 constraint by scheduling exams using minimum timeslots. Table 4.3 compares timeslots used in proprietary software or manual methods and the constructive algorithm for all these instances. Constructive algorithm has proven its ability to achieve SE1 by minimizing the overall exam session for all datasets. While comparing with the existing solution, it was observed that the timeslots required for FEB23 and FEB24 remain the same. In contrast, the timeslots for FCSHD and FCSIT can be further reduced to 13 and 8, respectively.

Table 4.3: Comparison Timeslots Used and Runtime for Studied Datasets

Instances	FESS / Manual	Constructive Algorithm	
	Timeslots	Timeslots	Runtime (s)
FEB23	10	10	0.20
FCSHD23	14	13*	1.66
FEB24	12	12	0.32
FCSIT24	14	8*	0.73

* indicates the timeslots that are fewer than existing methods

Table 4.4 presents the objective function values of the FESS/Manual and PVND methods for FCSHD23 and FCSIT24, using reduced timeslots. For FCSHD23, the PVND solution resulted in a higher objective value of 140.0 with a reduced timeslot of 13, which

was not accepted as the new solution. However, for FCSIT24, the PVND method achieved a lower objective value of 53.0 while reducing the timeslot to 8, which was adopted as the new solution in the subsequent experiment.

Table 4.4: Result on Reduced Timeslot and Objective Value Comparison

Instances	FESS / Manual		PVND	
	Objective value	Timeslot	Objective value	Reduced Timeslot
FCSHD23	133.7	14	140.0	13
FCSIT24	55.6	14	53.0	8

4.5.3.2 Exam Room Sharing

In contrast to other faculties that permit room sharing for multiple exams to reduce room usage, FCSHD enforces the extended soft constraint, SE2, which prohibits room sharing. In the constructive algorithm, this constraint is not taken into consideration. Therefore, in the PVND approach, neighborhood structure of N_4 and N_6 are employed for the FCSHD23 instance to eliminate any shared exams within a single room as a unary resource. These neighborhood structures have effectively removed all shared exams and addressed faculty preferences by satisfying the extended soft constraints.

To assess the impact of including SE2 constraints, a comparative analysis was conducted on the objective function values for the inclusion and exclusion of shared room preference for the FCSHD23 instance. Figure 4.16 presents box plots comparing the distribution of objective function values derived from experiments with shared and non-

shared room settings. The figure clearly illustrates the performance variations, emphasizing how including SE2 constraints can lead to a degradation in solution quality.

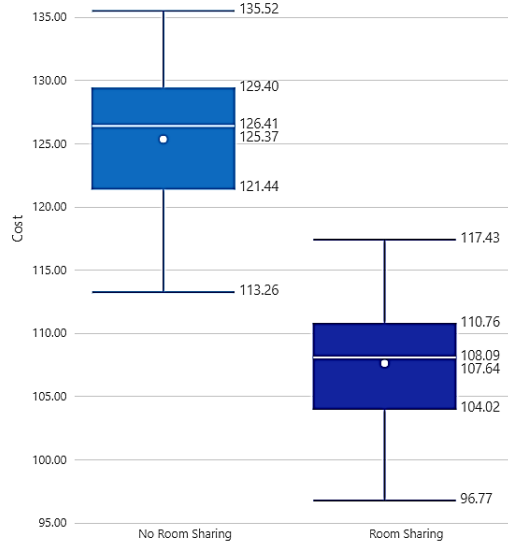


Figure 4.16: Shared vs. Non-Shared Room Settings

4.5.3.3 Balance Room Allocation for Split Exam

In the post-scheduling phase, for exams split across multiple rooms as requested by the exam officer, a balancer is applied to ensure a fair distribution of students based on room capacities. This balancing does not affect the objective function value. Jain's Fairness Index (*JFI*) (Jain et al., 1984), a widely used measure of fairness in resource allocation, is applied. Let $usedCap_{ir}$ represents the used capacity and $totalCap_{ir}$ represents the total capacity of exam i in rooms r .

The *JFI* falls within the range of $[0, 1]$. Its interpretation is intuitive: a *JFI* equal to 1 signifies that the solution represents a completely fair allocation, while a *JFI* equal to 0 indicates a completely unfair allocation. The intention is to improve fairness by maximizing

the *JFI*, which reflects a more equitable distribution of resources. Exams that are not split are excluded from the calculation.

The throughput of an exam, denoted as TP_i , is defined in Equation 4.1. The *JFI* over the total number of split exams, n_e , is defined in Equation 4.2.

$$TP_i = \sum_{r=1}^m \frac{usedCap_{ir}}{totalCap_{ir}}, \quad i = 1, 2, \dots, n_e \quad \text{Equation 4.1}$$

$$JFI = \frac{(\sum_{i=1}^{n_e} TP_i)^2}{n_e \cdot \sum_{i=1}^{n_e} (TP_i)^2} \quad \text{Equation 4.2}$$

Table 4.5 shows an improvement in the *JFI* values after balancing the student distribution across multiple rooms for split exams. A noticeable increase in *JFI* with percentage improvement varies between instances, ranging from a modest 3% to a significant 13%, indicating that the balancing process had a varying but consistently positive effect on fairness, as measured by *JFI*. Overall, all instances exhibit improved balance and fairness, with the FCSHD23 instance showing strong effects, suggesting that the non-sharing room setting contributes more significantly compared to the sharing case.

Table 4.5: Comparison of JFI with and without Balancing

Instances	JFI		%Difference
	Without Balancing	With Balancing	
FEB23	0.894	0.923	3%
FCSHD23	0.881	0.999	13%
FEB24	0.894	0.930	4%
FCSIT24	0.845	0.876	3%

The experiment was conducted across all instances, focusing on balancing split exams in both shared and non-shared room settings. Figures 4.17 – 4.20 visually present the distribution of capacity usage by room size for both balanced and unbalanced solutions across four instances. As illustrated in the bar charts, the balanced solutions demonstrate a more even distribution of capacity usage across rooms than the unbalanced solution, underscoring the positive impact of balancing measures.

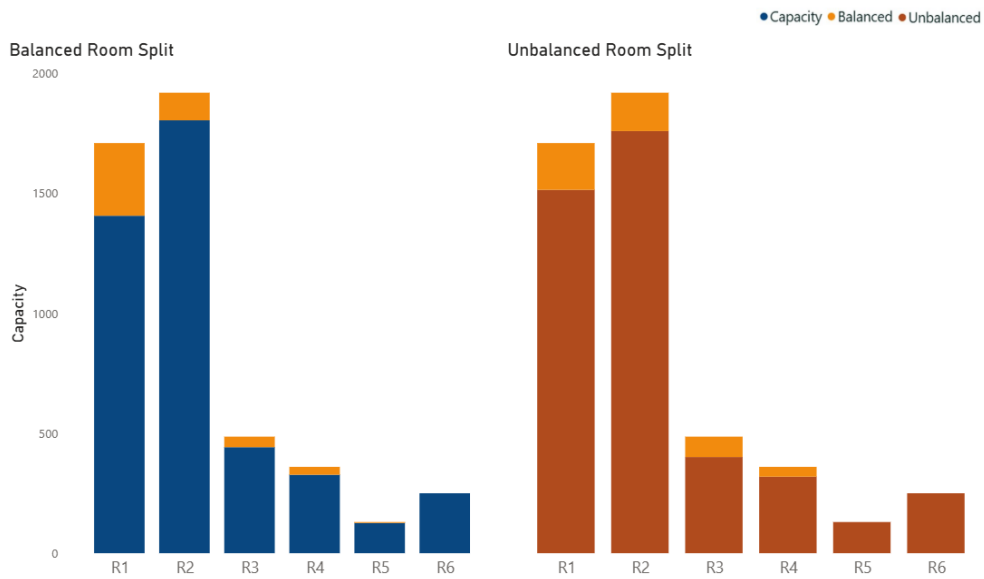


Figure 4.17: Balanced vs. Unbalanced Room Split for FEB23

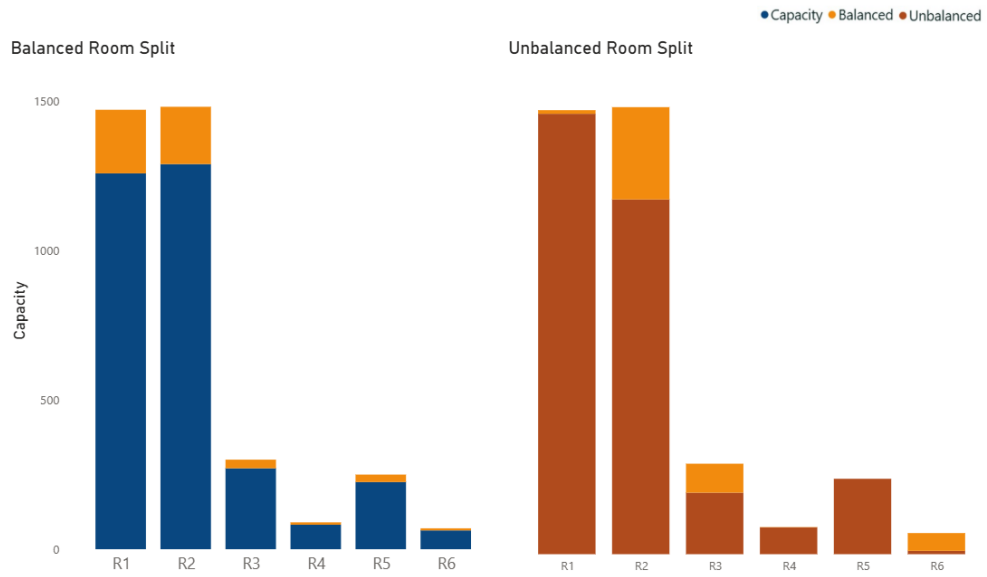


Figure 4.18: Balanced vs. Unbalanced Room Split for FCSHD23

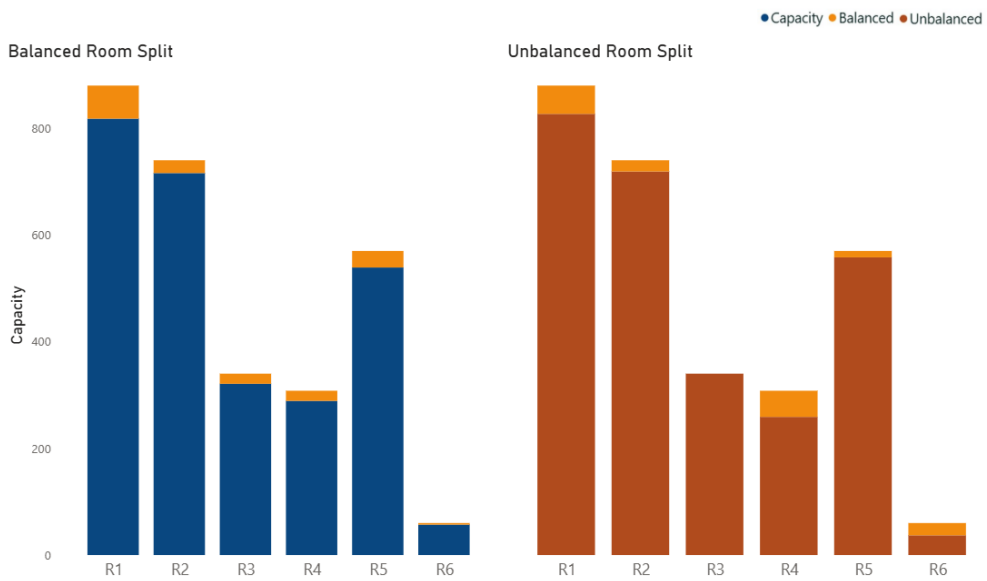


Figure 4.19: Balanced vs. Unbalanced Room Split for FEB24

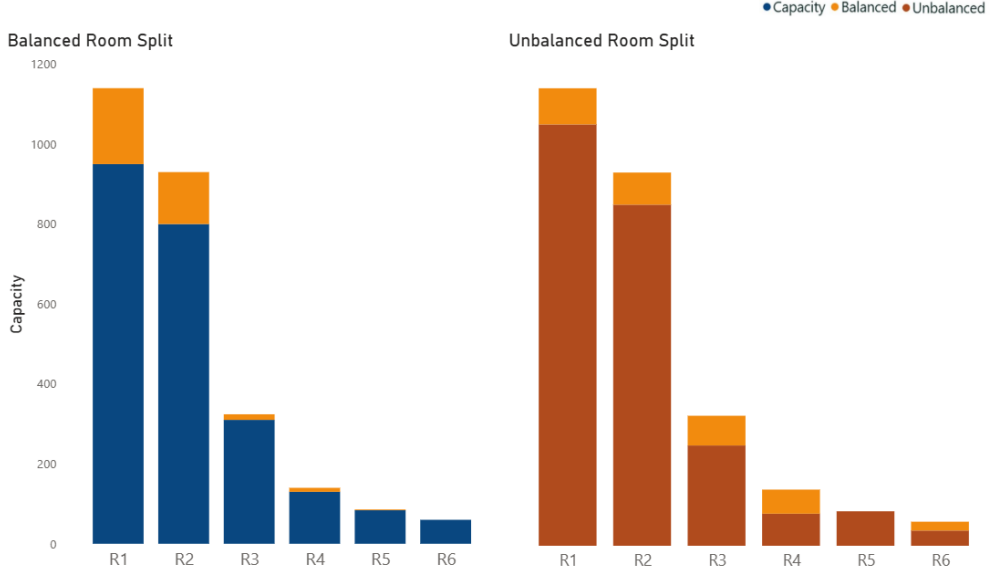


Figure 4.20: Balanced vs. Unbalanced Room Split for FCSIT24

4.6 Statistical Analysis of Neighborhood Structures

A set of multi-neighborhood search strategies was designed to enhance local search performance, aiming to ensure these neighborhood structures are fit for use in ETP. NS_1 , NS_2 , and NS_3 are neighborhood structures that involve both period-related and room-related moves, significantly impacting search space optimization. A statistical analysis was conducted to compare the neighborhood structures employed in the experiments and to assess whether there were any statistical differences. This comparison aims to identify any significant differences between using neighborhoods composite of $NS_1 \cdot NS_2 \cdot NS_3$ and the single or dual neighborhoods.

Selecting appropriate neighborhood structures is crucial in guiding the search toward high-quality solutions (Thompson and Dowsland, 1998). Timetables are generated using a PVND method using the initial random starting solution. Thirty timetable solutions are produced for each employed neighborhood composite, with the corresponding objective

function values recorded. The descriptive statistics of the five employed methods for 30 runs are given in Table 4.6 below. All statistical analyses are conducted using SPSS 29 version.

For FEB23, the combined structures $N_1 \cdot N_2 \cdot N_3$, yield the lowest mean value at 125.45, followed by $N_1 \cdot N_3$ at 126.53, while N_2 shows the highest mean value at 139.9. For FCSHD23, N_2 has the worst mean at 149.47, while the combined structures generally produce lower mean, especially for $N_1 \cdot N_3$ at 122.7, followed by $N_1 \cdot N_2 \cdot N_3$ at 124.49. For FEB24, N_2 exhibits the highest mean at 155.11, in contrast to $N_1 \cdot N_3$, which shows the lowest mean at 133.64, followed by N_1 at 133.83. For FCSIT24, N_2 has the highest mean of 75.37, while the neighborhood structures N_3 , at 51.07, followed by $N_1 \cdot N_3$ at 51.76, exhibit lowest mean values.

Table 4.6: Descriptive Statistics for Studied Instances

Neighborhood Structure	N	Mean	Std. Dev.	Min	Max
FEB23					
N_1	30	127.1097	2.27597	123.03	132.25
N_2	30	139.9057	1.38235	136.33	143.02
N_3	30	128.0087	2.47482	122.69	132.88
$N_1 \cdot N_2$	30	127.2267	2.45305	119.75	133.02
$N_1 \cdot N_3$	30	126.5343	2.39285	119.12	131.93
$N_2 \cdot N_3$	30	128.6877	3.77325	121.17	135.69
$N_1 \cdot N_2 \cdot N_3$	30	125.4523	2.34113	118.38	130.45
FCSHD23					
N_1	30	125.6840	5.41722	114.77	135.91

Table 4.6 continued

N_2	30	149.4779	5.76685	135.13	162.79
N_3	30	126.0160	7.44707	111.59	137.30
$N_1 \cdot N_2$	30	125.1139	6.55008	113.29	137.35
$N_1 \cdot N_3$	30	122.7077	4.89187	113.71	130.55
$N_2 \cdot N_3$	30	125.1400	5.14704	116.22	135.89
$N_1 \cdot N_2 \cdot N_3$	30	124.4990	5.40114	114.29	133.76
FEB24					
N_1	30	133.8390	1.88817	130.02	137.38
N_2	30	155.1139	2.15564	150.82	159.50
N_3	30	138.0642	2.54018	134.22	143.00
$N_1 \cdot N_2$	30	134.3153	1.39825	132.09	137.45
$N_1 \cdot N_3$	30	133.6437	1.49645	130.73	136.45
$N_2 \cdot N_3$	30	137.6800	1.74451	134.59	140.96
$N_1 \cdot N_2 \cdot N_3$	30	134.1333	1.53763	131.67	137.18
FCSIT24					
N_1	30	52.5630	1.43829	50.11	55.50
N_2	30	75.3789	3.47705	68.01	81.17
N_3	30	51.0783	1.60290	48.79	53.82
$N_1 \cdot N_2$	30	52.6547	1.48166	49.37	54.96
$N_1 \cdot N_3$	30	51.7680	1.46742	49.56	54.52
$N_2 \cdot N_3$	30	51.9420	2.21037	48.65	56.13
$N_1 \cdot N_2 \cdot N_3$	30	51.8833	1.61094	49.16	55.97

Std. Dev. = Standard Deviation; Min = Minimum; Max = Maximum

Overall, N_2 , the neighborhood structure that swaps exams performs poorly when used individually due to the varying availability of shared rooms across timeslots. However, it proves effective when integrated with other neighborhood structures. Conversely, N_1 and N_3 consistently perform better, individually and in combination with other structures.

Statistical analysis begins with a normality test using the Shapiro-Wilk method and proceeds with the appropriate statistical test based on the results of the normality assessment. The normality test is conducted using Shapiro-Wilk, where the null hypothesis, H_0 posits that the samples are normally distributed, while the alternative hypothesis, H_1 asserts that the samples are not normally distributed. H_0 is rejected when $p < 0.05$ and accept it when $p \geq 0.05$. According to Table 4.7, based on the Shapiro-Wilk normality test, all tested instances exhibit $p \geq 0.05$ significance threshold, therefore the normality is satisfied for all instances.

Table 4.7: Tests of Normality

Neighborhood Structure	Shapiro-Wilk			
	FEB23	FCSHD23	FEB24	FCSIT24
N_1	0.835	0.828	0.722	0.261
N_2	0.087	0.662	0.379	0.525
N_3	0.534	0.226	0.050	0.072
$N_1 \cdot N_2$	0.091	0.560	0.086	0.280
$N_1 \cdot N_3$	0.121	0.240	0.524	0.116
$N_2 \cdot N_3$	0.283	0.562	0.410	0.187
$N_1 \cdot N_2 \cdot N_3$	0.166	0.160	0.414	0.731

Since all instances demonstrate a normal distribution, the ANOVA test is employed on the mean values of the objective function, followed by post-hoc analysis, to identify any significant differences. Before conducting the ANOVA test, a homogeneity test is performed, with the results shown in Table 4.8.

Table 4.8: Test of Homogeneity

Instances	Levene Statistic	df1	df2	Significance
FEB23	4.735	6	203	<0.001
FCSHD23	1.313	6	203	0.253
FEB24	1.912	6	203	0.080
FCSIT24	8.382	6	203	<0.001

Based on the Levene statistics, the variances for FEB23 and FCSIT24 are not assumed to be equal at the 0.05 significance level, indicating a violation of the homogeneity of variances assumption. Consequently, the Dunnett T3 post hoc test is employed for multiple comparisons, as shown in Table 4.9. In contrast, FCSHD23 and FEB24 have significant values of 0.253 and 0.080, respectively, which do not indicate a significant violation of this assumption; therefore, a standard ANOVA test can be applied.

ANOVA results indicate a significant difference in objective value between the groups ($F(6, 203) = 76.585, p < 0.001$) for FCSHD23 and ($F(6, 203) = 515.949, p < 0.001$) for FEB24. Therefore, a post hoc test is performed to identify which specific group pairs show significant differences. The post hoc Tukey's HSD (honestly significant difference) test conducted for FCSHD23 and FEB24 is presented in Table 4.10. Symbol * indicates that the mean difference is statistically significant at the 0.05 level.

The statistical test results indicate significant differences (reject H_0) with p-values of 0.05. The findings in Tables 4.9 and 4.10 reveal significant differences among the factors analyzed for the instances studied. In all the instances, a strongly significant difference (reject H_0) is observed for N_2 , with a p-value less than 0.05. In FEB23 and FEB24, N_3 and $N_2 \cdot N_3$ also exhibit significant differences (reject H_0) with p-values less than 0.05, underscoring their influence on the objective function. The negative values for these factors further indicate their detrimental effects.

Table 4.9: Dunnett T3 Post Hoc for FEB23 and FCSIT24

(I) Method	(J) Method	Mean Difference		Sig.	95% Confidence Interval	
		(I-J)	Std. Error		Lower Bound	Upper Bound
FEB23						
$N_1 \cdot N_2 \cdot N_3$	N_1	-1.65733	0.59612	0.137	-3.5412	0.2265
	N_2	-14.45333*	0.49638	0.000	-16.0372	-12.8694
	N_3	-2.55633*	0.62198	0.003	-4.5221	-0.5906
	$N_1 \cdot N_2$	-1.77433	0.61909	0.110	-3.7309	0.1822
	$N_1 \cdot N_3$	-1.08200	0.61119	0.802	-3.0134	0.8494
	$N_2 \cdot N_3$	-3.23533*	0.81073	0.005	-5.8184	-0.6523
FCSIT24						
$N_1 \cdot N_2 \cdot N_3$	N_1	-0.67967	0.39428	0.832	-1.9263	0.5669
	N_2	-23.49560*	0.69964	0.000	-25.7453	-21.2459
	N_3	0.80500	0.41491	0.675	-0.5061	2.1161
	$N_1 \cdot N_2$	-0.77133	0.39960	0.683	-2.0345	0.4918
	$N_1 \cdot N_3$	0.11533	0.39785	1.000	-1.1423	1.3730
	$N_2 \cdot N_3$	-0.05867	0.49936	1.000	-1.6428	1.5255

Std. Error = Standard Error; Sig. = Significance Level

Table 4.10: Turkey HSD Post Hoc for FCSHD23 and FEB24

(I) Index	(J) Index	Mean Difference		Sig.	95% Confidence Interval	
		(I-J)	Std. Error		Lower Bound	Upper Bound
FCSHD23						
$N_1 \cdot N_2 \cdot N_3$	N_1	-1.18500	1.51362	0.986	-5.6928	3.3228
	N_2	-24.97893*	1.51362	0.000	-29.4867	-20.4711
	N_3	-1.51700	1.51362	0.953	-6.0248	2.9908
	$N_1 \cdot N_2$	-0.61493	1.51362	1.000	-5.1227	3.8929
	$N_1 \cdot N_3$	1.79133	1.51362	0.900	-2.7165	6.2991
	$N_2 \cdot N_3$	-0.64100	1.51362	1.000	-5.1488	3.8668
FEB24						
$N_1 \cdot N_2 \cdot N_3$	N_1	0.29430	0.48074	0.996	-1.1374	1.7260
	N_2	-20.98057*	0.48074	0.000	-22.4123	-19.5489
	N_3	-3.93083*	0.48074	0.000	-5.3625	-2.4991
	$N_1 \cdot N_2$	-0.18200	0.48074	1.000	-1.6137	1.2497
	$N_1 \cdot N_3$	0.48967	0.48074	0.949	-0.9420	1.9214
	$N_2 \cdot N_3$	-3.54667*	0.48074	0.000	-4.9784	-2.1150

Std. Error = Standard Error; Sig. = Significance Level

In conclusion, the results for all instances indicate that N_2 significantly influences the objective function, considered the worst neighborhood move, as it barely improves the initial objective value. In contrast, regardless of whether N_1 is used individually or in combination

with other neighborhoods, such as $N_1 \cdot N_2$, and $N_1 \cdot N_3$, or even $N_1 \cdot N_2 \cdot N_3$ approach demonstrates a positive influence on the objective function value.

Although results for certain instances could also be approached effectively using a single effective neighborhood structure such as N_1 , it is notable that using multi-neighborhood is intuitive. This is due to the complementary nature of neighborhoods, which target distinct features of the current state, while each neighborhood independently leaves the search space unconnected. Additionally, experiments have shown that the sequence of these neighborhoods does not significantly affect the results, and they can therefore be applied in any order without impacting the effectiveness of the search. This work offers complementary insights into the use of multi-neighborhood metaheuristics.

4.7 Comparative Analysis of Centralized and Decentralized Timetabling

The PVND approach is designed to prioritize large-capacity exam rooms first, mostly shared hall resources. As shown in Figure 4.21, all large-size rooms were fully occupied before transitioning to medium-size rooms across all instances. The rationale behind this approach is twofold: first, it minimizes the need to split exams across small-sized rooms, which are all faculty rooms, thus maintaining cohesion for larger enrolments; second, it reduces the overall number of invigilators required by limiting the number of rooms in use for any given exam. This strategy enhances resource efficiency and ensures optimal room allocation.

However, large-size shared halls have limited resources, and their availability at each timeslot is not fixed. The assignment of shared halls to each faculty is carried out by BPPs using a fair allocation method based on the number of enrollments or exam size, as discussed in Chapter 3. The fairness of the BPP's allocation of shared rooms among faculties is being

considered due to the unequal faculty room resources, varying exam enrollment, and conflict density across different faculties.

To address the concern, a comparative analysis is conducted between the current decentralized faculty timetabling and centralized timetabling. This analysis utilizes the PVND approach and the dataset from Semester 2-2023/24 for FEB24 and FCSIT24 instances, as the gap between the timeslots used is larger, which indicates more residual resources available for insights. For a measurable and fair comparison, all resources in each timeslot remain unchanged, and all the solutions will be evaluated using the same objective function structure to assess whether the solution's quality can be improved and if resource allocation can be optimized.

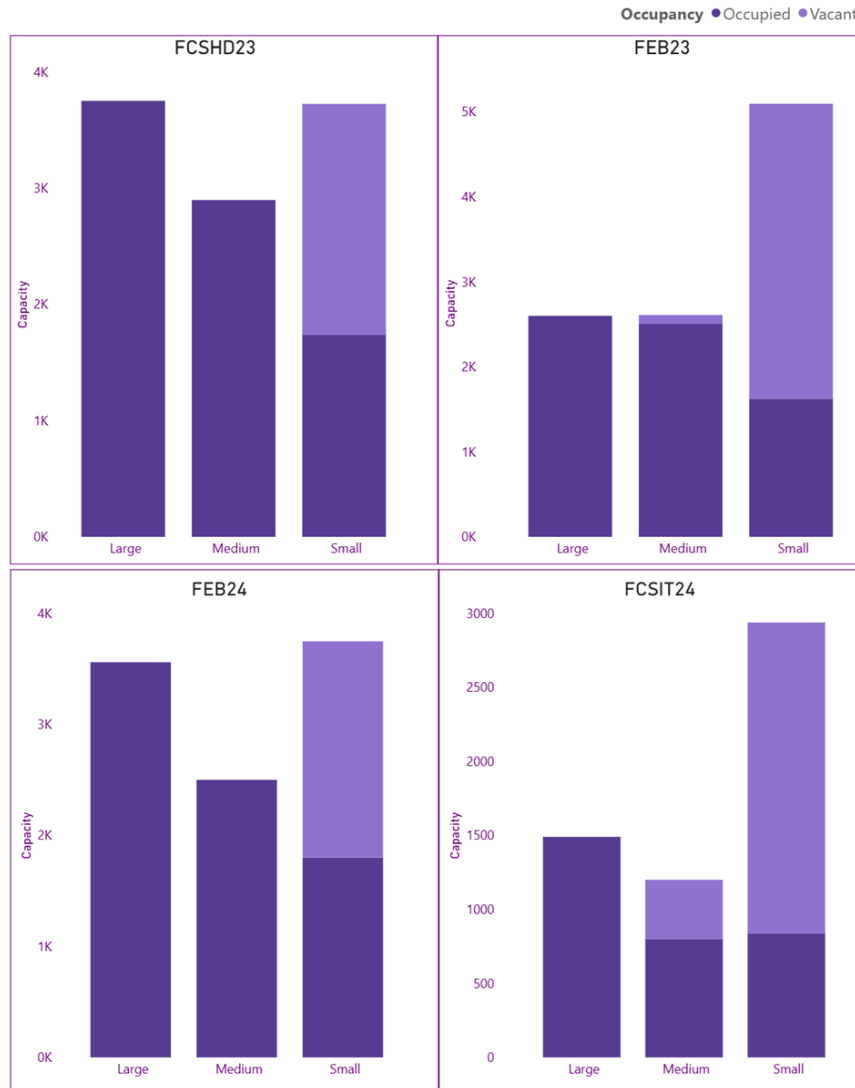


Figure 4.21: Comparison of Room Type Occupancy

The comparison analysis compares the current decentralized system with centralized scheduling and two alternative decentralized configurations. This analysis contrasts the following approaches:

- i. Centralized with standardized timeslots, without room reallocation: Uses a standardized 12-timeslot configuration, ensuring that both instances (FEB24 and FCSIT24) have the same end-of-exam session.

- ii. Decentralized with standardized timeslots, without room reallocation: Each faculty uses a standardized 12-timeslot configuration without post-scheduling room reallocation.
- iii. Decentralized with minimized timeslots and room reallocation: Each faculty uses the minimum number of timeslots but applies post-scheduling room reallocation.

In the experimental setting, the centralized solution will be measured using the same objective function value for FEB24 and FCSIT24 instances undergoing 30 runs. Then, the analysis will evaluate if a statistically significant difference exists between these approaches.

4.7.1 Centralized vs Decentralized without Reallocation

The comparison of centralized and decentralized solutions using a standardized configuration of 12 timeslots for each instance, with no room reallocation, is presented in Table 4.11. When comparing the objective values of centralized solutions and the total of decentralized solutions across all aspects (SC1 to SC4) for both instances, it is evident that decentralized solutions incur higher overall objective values. Specifically, the total objective value for decentralized solutions is 179.3, while the total for centralized is 144.5, representing an improvement of approximately 19.4%. DEC consistently shows higher values across each scenario, with the most significant differences in spread (SC2) and slot preference (SC4).

Table 4.11: Comparison of Centralized and Decentralized without Reallocation

Instance	SC1		SC2		SC3		SC4		Total		p-value
	CEN	DEC	CEN	DEC	CEN	DEC	CEN	DEC	CEN	DEC	
FEB24	–	65.6	–	24.6	–	26.1	–	16.0	–	132.3	–
FCSIT24	–	26.4	–	12.6	–	7.8	–	0.0	–	46.8	–
Total	86.0	92.0	19.5	37.2	29.1	33.9	9.8	16.0	144.5	179.3	<0.001

Bold indicates the best objective value; CEN = Centralized; DEC = Decentralized

4.7.2 Centralized vs Decentralized with Reallocation

A post-scheduling room allocation is carried out to adjust the unoccupied shared resources from faculties whose timetables end earlier to those whose timetables extend later, which can only be done once all faculty timetables are finalized. For Semester 2-2023/24, FCSIT24 utilized minimum timeslots of 8, whereas FEB24 used 12 timeslots. As a result, shared rooms from timeslots 9 to 12 were reallocated from FCSIT24 to FEB24. After reallocating shared rooms from FCSIT24 to FEB24, the objective value improved significantly, with the best objective value decreasing from 132.3 to 120.6, resulting in an 8.8% $((132.3 - 120.6)/132.3 * 100)$ improvement. These adjustments demonstrate the positive impact of reallocating unused shared resources on resource usage and objective value efficiency.

Table 4.12 compares centralized and decentralized solutions with minimized timeslots and room reallocation. The decentralized approach incurs significantly higher objective values in SC2 on spread and SC4 on the preferred slot. In contrast, for SC1 and SC3, which focus on minimizing room usage and minimizing splits, the objective value for the

centralized approach is higher, indicating that the decentralized approach is more efficient in resource allocation.

In comparing their total objective function values, it is evident that the centralized approach performs better overall, with a lower total value (144.5) than the decentralized approach (172.0), which indicates a reduction in the total objective value of approximately 15.9%, calculated as $(172.0 - 144.5) / 172.0 \times 100$. This enhancement in the centralized approach underscores its effectiveness in improvements in the quality of the solution.

Table 4.12: Comparison of Centralized and Decentralized with Reallocation

Instance	SC1		SC2		SC3		SC4		Total	
	CEN	DEC	CEN	DEC	CEN	DEC	CEN	DEC	CEN	DEC
FEB24	–	59.1	–	25.1	–	16.0	–	20.4	–	120.6
FCSIT24	–	23.2	–	18.8	–	9.4	–	0	–	51.4
Total	86.0	82.3	19.5	43.9	29.1	25.4	9.8	20.4	144.5	172.0

Bold indicates the best objective value; CEN = Centralized; DEC = Decentralized

Preliminary results indicate that centralized scheduling, which employs standardized timeslots without room reallocation, outperforms decentralized scheduling. Centralized scheduling achieves superior exam spread distribution and slot preference fulfillment, reflecting strong temporal optimization. In contrast, a decentralized configuration with minimized timeslots and room reallocation optimizes room usage and room or area splits, demonstrating superior space utilization and resource efficiency. Such decentralized configuration is recommended if resource management is prioritized. Alternatively, a centralized approach may be preferable when human-centric considerations dominate.

Overall, all three alternatives outperform the current decentralized baseline in terms of their objective function values. However, since complete datasets were unavailable for all faculties, only a limited number of faculties could be included, and the experiment could not be conducted in its entirety. However, even with datasets from only two faculties, the results already demonstrate the benefits of centralized scheduling, highlighting the value of further research in future experiments.

4.8 Comparative Analysis of PVND, BVND, and IT-PVND

Besides proposing the PVND approach, which must outperform existing methods on the studied dataset, an Iterative Threshold Pipe Variable Neighborhood Descent (IT-PVND) is proposed and designed. IT-PVND extends the standard PVND by allowing the acceptance of non-improving solutions that are better than certain thresholds as new incumbent solutions, in contrast to the standard PVND, which admits only improving solutions. Additionally, the Basic Sequential Variable Neighborhood Descent (BVND) is developed to validate the proposed approaches and assess their effectiveness.

4.8.1 Iterative Threshold based Pipe Variable Neighborhood Descent

The proposed PVND algorithm in Section 4.4.2 employs a first-improvement acceptance criterion, which updates the solution as soon as a better candidate is found. A modified version of PVND, called IT-PVND, has been developed, which uses objective value difference as the acceptance criterion to evaluate the impact of different acceptance criteria on algorithm performance. As shown in Figure 4.22, the algorithm (Line 3–4) embeds a control parameter, referred to as the objective value threshold ($cMax$) and one input parameter, referred to as the iteration count (L), into its process.

IT-PVND relies on the $cMax$ parameter to guide the acceptance rate of moves. It is initialized to the initial solution objective function value. The algorithm (Lines 16–18) increments the iteration count in steps of L , and once every L iteration (i.e., $cMax$ modulo L equals 0), the $cMax$ is updated to the current objective value. Throughout the search, the algorithm always accepts both improvements and sideways moves as long as the objective function values are lower than $cMax$, as illustrated in Figure 4.23, which depicts the local search procedure.

Algorithm 4.5 Iterative Threshold based Pipe Variable Neighborhood Descent (IT-PVND)

```

1: Input: Neighborhood structures  $N$ , initial solution  $X$ 
2: Output: Best solution  $X$  found
3: Set the Iteration Count  $L$ 
4:  $cMax \leftarrow Z(X)$ 
5:  $k \leftarrow 1$ 
6:  $iter \leftarrow 0$ 
7: while  $iter < MaxIter$  do
8:   while  $k \leq k_{max}$  do
9:      $X' \leftarrow \text{LOCALSEARCH\_CD}(X, N_k, cMax)$ 
10:    if  $Z(X') < Z(X)$  then
11:       $X \leftarrow X'$ 
12:    else
13:       $k \leftarrow k + 1$ 
14:    end if
15:  end while
16:  if  $iter \bmod L = 0$  then
17:     $cMax \leftarrow Z(X)$ 
18:  end if
19:   $iter \leftarrow iter + 1$ 
20: end while
21:  $X \leftarrow \text{BALANCER}(X)$ 
22: return  $X$ 

```

Figure 4.22: Procedure of the IT-PVND

Algorithm 4.6 LOCALSEARCH_CD($X, N_k, cMax$)

```
1: Generate a random permutation queue of all the exams
2:  $j \leftarrow 0$ 
3: while  $j < exam\_size$  and time limit not reached do
4:    $X' \leftarrow$  applying neighborhood  $N_k(X)$  to  $queue_j$ 
5:   if  $Z(X') < Z(X)$  or  $Z(X') < cMax$  then
6:      $X \leftarrow X'$ 
7:     break
8:   end if
9:    $j \leftarrow j + 1$ 
10: end while
11: return  $X$ 
```

Figure 4.23: Local Search using Threshold Acceptance Criterion

The main idea of IT-PVND is to maintain the given objective value threshold over L iterations for multiple neighborhood structures, rather than just one. The search process permits accepting solutions that are worse than the current best, as long as they fall below a certain objective value threshold. This approach provides a more flexible acceptance criterion, leading to a slower reduction in the current cost and requiring a longer time to achieve convergence. This addition facilitates a more exploratory search behavior, avoiding early convergence to local optima.

4.8.2 Basic Sequential Variable Neighborhood Descent

The BVND method is a widely used metaheuristic that conducts a local search by systematically altering neighborhood structures to explore the search space. In each iteration, a neighborhood structure is selected in a predefined order. If the local search finds a candidate solution that improves the current one, it is accepted, and the search restarts with the first neighborhood structure. If not, the search moves on to the next neighborhood structure, continuing until all neighborhoods have been explored.

The primary advantage of BVND over conventional local search methods lies in its ability to escape local minima by transitioning between different neighborhood structures. BVND operates on the principle that a local minimum in one neighborhood structure might not correspond to a local minimum in a different one. To compare its effectiveness with the PVND algorithm, a BVND algorithm is developed for addressing ETP that utilizes the same set of neighborhood structures, as illustrated in Figure 4.24.

Algorithm 4.7 Basic Sequential Variable Neighborhood Descent (BVND)

```

1: Input: Neighborhood structures  $N$ , initial solution  $X$ 
2: Output: Best solution  $X$  found
3:  $k \leftarrow 1$ 
4:  $iter \leftarrow 0$ 
5: while  $iter < MaxIter$  do
6:   while  $k \leq k_{max}$  do
7:      $X' \leftarrow \text{LOCALSEARCH}(X, N_k)$ 
8:     if  $Z(X') < Z(X)$  then
9:        $X \leftarrow X'$ 
10:       $k \leftarrow 1$ 
11:     else
12:        $k \leftarrow k + 1$ 
13:     end if
14:   end while
15:    $iter \leftarrow iter + 1$ 
16: end while
17:  $X \leftarrow \text{BALANCER}(X)$ 
18: return  $X$ 

```

Figure 4.24: Procedure of BVND

4.8.3 Comparison of PVND, BVND, and IT-PVND

Table 4.13 compares the performance of PVND, BVND, and IT-PVND across all instances, presenting the corresponding objective function values. Each algorithm executed 30 runs over five iterations for FCSIT24 and ten iterations for the other datasets. Compared to PVND, IT-PVND consistently demonstrates a trend toward lower mean values across the instances analyzed. For the FEB23 instance, IT-PVND reduces the mean value by 4.2%, while BVND provides a mean that is 2.4% lower than PVND. For the FCSHD23 instance,

IT-PVND achieves a mean value of 3.9% lower than VND, whereas BVND's value is 2.3% higher than PVND. For the FEB24 instance, IT-PVND exhibits a 1.7% reduction in mean value, while BVND remains slightly lower than PVND. For the FCSIT24 instance, IT-PVND lowers the mean value by 2.8%, while BVND's mean value is 1.1% higher than PVND.

Table 4.13: Descriptive Result for PVND, BVND, and IT-PVND

Instance	PVND			IT-PVND			BVND		
	Best	Mean	Std. Dev.	Best	Mean	Std. Dev.	Best	Mean	Std. Dev.
FEB23	123.9	128.9	2.80	118.3	123.5	2.60	121.9	125.8	3.18
FCSHD23	115.1	126.8	5.85	112.4	121.9	5.09	118.6	129.7	6.21
FEB24	132.1	134.7	1.23	130.3	132.4	1.21	131.8	134.6	1.77
FCSIT24	50.1	53.0	1.47	49.4	51.5	1.41	50.4	53.6	1.96
Average	105.3	110.9	2.84	102.6	107.3	2.6	105.7	110.9	3.3

Std. Dev. = Standard Deviation; Bold values indicate the best value.

When comparing the best objective function values of the three approaches, IT-PVND consistently offers the lowest best values and means in all cases, with comparatively lower standard deviations across the instances analyzed. In contrast, BVND exhibited the least favorable outcomes overall, with higher variability and elevated mean values. Notably, BVND performed the poorest in FCSHD23 with a room-sharing setting.

The performance of the BVND and PVND algorithms is nearly identical; Their inability to reach different local optima systematically contributes to their relatively poorer results. Although all algorithms utilize the same neighborhood structure, the IT-PVND approach,

with the current objective function value serving as the acceptance condition for several consecutive iterations, has been proven to enhance overall performance and outperform those of PVND and BVND in all instances.

4.9 Conclusion

A PVND algorithm is developed that incorporates a pipe neighborhood change. Six period-related and room-related neighborhood structures have been designed to operate under room-sharing and non-sharing settings, effectively accommodating faculty room-sharing preferences. The novelty of this PVND algorithm lies in its ability to schedule both uncapacitated (online) and capacitated (physical) exams within a unified framework, satisfying all standard constraints.

PVND is implemented to solve real-world ETP for two faculties across two semesters, improving objective function values of the solution, delivering higher-quality results than those produced by proprietary software or manual approaches. Additionally, the extended soft constraint on minimizing exam session durations was validated against the rigid constraint of Toronto benchmarks. It also efficiently facilitates a balanced split of exam constraints in shared and non-shared settings.

Statistical analysis was conducted to investigate differences between single and composite neighborhood structures. Some neighborhood structures performed better individually but mostly excelled in combination, affecting the quality of the returned solution. A detailed analysis using the Dunnett T3 post hoc test and Tukey HSD post hoc test reveals that only the use of N_2 leads to significant differences (rejecting H_0) for the smaller instance, while N_2 , N_3 , and their composite ($N_2 \cdot N_3$) show significant differences (rejecting H_0) for the larger instance. However, the inclusion of N_1 does not result in

significant differences, though it mostly maintains the objective function's quality. Therefore, it is recommended to incorporate all neighborhood structures, as each contributes unique strengths that enhance the algorithm's ability to explore the search space.

The comparison between centralized and decentralized approaches revealed a significant difference in objective function values, with centralized timetabling demonstrating superior objective value efficiency and resource utilization, which raises scrutiny over BPP's claim of fairness in room allocation among faculties for a decentralized approach. However, the experiment faced limitations, as only two out of ten faculties were available, preventing a comprehensive analysis. Future research can focus on incorporating complete datasets from all faculties to explore the benefits of centralized approaches thoroughly.

The IT-PVND algorithm, a modified version of the PVND algorithm, yields high-quality solutions that outperform both the PVND and BVND algorithms. It utilizes a straightforward parameter, the iterative objective value threshold, which promotes a more exploratory search behavior to identify practical solutions, although it requires a slightly longer run time. Modifying the acceptance criterion increases the effectiveness of the search and helps mitigate the effects of premature convergence.

The key contributions of this chapter are summarized below:

- i. Preliminary testing revealed that even a standard PVND employing a simple local search can yield competitive results, outperforming proprietary software and manual methods.
- ii. The IT-PVND demonstrated superior performance compared to the BVND

and PVND approaches. This capability enables IT-PVND to avoid premature convergence to suboptimal solutions effectively.

- iii. A comparative study was conducted to evaluate decentralized and centralized timetabling approaches for the real-world ETP.

Literature studies demonstrate that hybridizing single and population-based approaches can yield competitive performance in achieving desired solutions (Blum & Roli, 2003). This serves as an impetus for investigating the use of the hybridization method on the studied problem. Given the effectiveness of the multi-neighborhood structures and the PVND variations, the hybridization of PVND variations is further explored with another population-based method in the following chapter.

CHAPTER 5

DISCRETE WHALE OPTIMIZATION ALGORITHM FOR EXAMINATION TIMETABLING

5.1 Introduction

The Whale Optimization Algorithm (WOA), proposed by Mirjalili and Lewis (2016), is a swarm-based metaheuristic inspired by the foraging behavior of humpback whales, which create spiral-shaped bubble nets to encircle and trap prey near the water's surface. WOA is well known for its straightforward concept, ease of coding implementation, minimal dependency on initial solutions, and overall effectiveness (Gharehchopogh & Gholizadeh, 2019).

Known for its simplicity, minimal parameter tuning, WOA incorporates a logarithmic spiral function to enhance search capabilities, particularly in the border areas of the search space. Due to its advantages, the WOA has gained widespread adoption across various domains, with studies demonstrating that it offers competitive or superior performance compared to many existing optimization techniques (Mirjalili & Lewis, 2016).

Leveraging its multi-segment local search capability, this study develops a discrete version of the approach to efficiently solve the ETP, a promising method yet to be explored in the existing literature and offers valuable opportunities for further investigation. Exploitation in the original WOA is driven by distance-based calculations between each search agent and the best solution found so far. In this study, local search methods—specifically VND—are employed to explore the neighborhood of the best-known solution, with the expectation of achieving improved outcomes. This framework aims to strengthen WOA's exploitation capabilities while preserving its exploratory power.

Since the WOA was originally designed for continuous optimization, its evolutionary process depends on the continuous updating of individual positions, rendering it unsuitable for direct application to discrete timetabling problems. To overcome this limitation, discrete WOA models that embed local search methods are proposed for addressing real-world ETP, as detailed in Section 5.2. Section 5.3 discusses the result of comparing these WOA models, together with the PVND variations discussed in Chapter 4, using the same datasets. Finally, section 5.4 concludes with key insights.

The research presented in this chapter was published in the *International Journal of Advanced Computer Science and Applications* (Siew et al., 2025).

5.2 Modified Discrete WOA Algorithms

In the original WOA algorithm, whales update their positions within a continuous domain using specific equations, which makes it unsuitable for discrete problems. To adapt the WOA for discrete or binary problems, local search mechanisms were employed (Abdel et al., 2019; Xu et al., 2018; Rana et al., 2022), and neighborhood search techniques were incorporated to enhance the search capabilities (Zhang & Gu, 2023; Austero et al., 2020; Premananda et al., 2020). Motivated by these studies, local search methods that enable the algorithm to function directly within a discrete search space are adopted. For the discrete update procedure, three local search methods—first descent method (LS_FD), first descent below a threshold value method (LS_FDT), and PVND method—are applied, as illustrated in Figures 5.1, 5.2, and 5.3.

Algorithm 5.1 Local Search using First Descent

```
1: function LS_FD( $X, N$ )
2:    $j \leftarrow 0$ 
3:   Generate a random permutation  $queue$  of all the exams
4:   while  $j < exam\_size$  and time limit not reached do
5:     Select a random neighborhood  $N_r \in N$ 
6:      $X' \leftarrow$  applying  $N_r(X)$  to  $queue_j$ 
7:     if  $Z(X') < Z(X)$  then
8:        $X \leftarrow X'$ 
9:       break
10:    end if
11:     $j \leftarrow j + 1$ 
12:  end while
13:  return  $X$ 
14: end function
```

Figure 5.1: Procedure of the First Descent Method

Algorithm 5.2 Local Search using First Descent Below Threshold Value

```
1: function LS_FDT( $X, N, vT$ )
2:   Generate a random permutation  $queue$  of all the exams
3:    $j \leftarrow 0$ 
4:   while  $j < exam\_size$  and time limit not reached do
5:     Select a random neighborhood  $N_r \in N$ 
6:      $X' \leftarrow$  applying neighborhood  $N_r(X)$  to  $queue_j$ 
7:     if  $Z(X') < Z(X)$  or  $Z(X') < vT$  then
8:        $X \leftarrow X'$ 
9:       break
10:    end if
11:     $j \leftarrow j + 1$ 
12:  end while
13:  return  $X$ 
14: end function
```

Figure 5.2: Procedure of the First Descent below Threshold Value Method

Algorithm 5.3 Pipe Variable Neighborhood Descent (PVND)

```
1: function PVND( $X, N$ )
2:    $k \leftarrow 1$ 
3:   while  $k \leq k_{max}$  do
4:      $X' \leftarrow \text{LOCALSEARCH}(X, N_k)$ 
5:     if  $Z(X') < Z(X)$  then
6:        $X \leftarrow X'$ 
7:     else
8:        $k \leftarrow k + 1$ 
9:     end if
10:  end while
11:  return  $X$ 
12: end function
```

Figure 5.3: Procedure of the PVND (Hansen et al., 2017)

For LS_FD, a local search method operates by exploring a set of neighborhood structure, N , around the current incumbent solution, X , at each iteration. N_r represents the randomly selected move from the neighborhood set N . Beginning with X solution, the search iteratively evaluates the neighborhood and selects a solution X' that improves upon X , if such a solution exists. The selected solution X' then becomes the new incumbent solution. The LS_FDT strategy is similar to LS_FD, except with different acceptance criteria, promoting exploration by accepting a solution X' either that improves or beyond an iterative updated threshold value, vT .

The modified discrete WOA follows these steps:

- i. Step 1: Set a predefined population of whales and generate initial solutions randomly using a constructive algorithm.
- ii. Step 2: Assess the objective value of each whale and identify the one with the lowest objective value.
- iii. Step 3: To refine the solutions for each whale, perform the discrete update

procedure using local search methods.

- iv. Step 4: Evaluate if the maximum number of iterations has been reached. If so, proceed to Step 5; otherwise, return to Step 2.
- v. Step 5: End the process and output the final best solution.

The proposed approach integrates local search methods into the population-based WOA through three distinct models: (1) the Random-based WOA Model 1 (WOARB-1), (2) the Random-based WOA Model 2 (WOARB-2), and (3) the Iterative-based WOA (WOAIB), as illustrated in Figures 5.4, 5.5, and 5.6.

The first two models, WOARB-1 and WOARB-2, use the standard WOA mechanism, where the search behavior—whether to intensify around the best solution or explore alternative regions—is probabilistically determined by the parameter $p \in [0,1]$ at each iteration. Third model, WOAIB adopts a sequential split based on iteration count. During the initial half of the total iterations, the algorithm balances exploration and exploitation, allowing adaptive search around and beyond the current best solution. In the second half, it shifts to focused exploitation, refining the search within high-quality regions to improve convergence precision.

As illustrated in Figure 5.4, the WOARB-1 model employs a randomly generated probability $p \in [0, 1]$ when performing search updates. If $p < 0.5$, the algorithm evaluates the size of A . If $|A| < 1$, it enters the exploitation phase by applying the LS_FD procedure (Line 10), replacing Equation 2.2 from the original WOA. It applied to guide the search by greedily selecting the first improving solution. Otherwise, if $|A| \geq 1$, it shifts the LS_FDT procedure (Lines 12-15), replacing Equation 2.8, to accept worse solutions within a

predefined threshold, allowing the algorithm to avoid local optima and continue exploring the solution space. If $p \geq 0.5$, the algorithm updates the solution using PVND procedure (Line 18), replacing Equation 2.6 from the original WOA, systematically explores various neighborhood structures to identify the best solution. The framework is depicted in Figure 5.5.

Algorithm 5.4 Random-based Whale Optimization Algorithm (WOARB-1)

```

1: Generate Initial Population  $X_i$  ( $i = 1, 2, 3$ )
2: Calculate the fitness of each solution
3:  $X^* \leftarrow$  the best search agent
4:  $vT \leftarrow Z(X)$ 
5: while  $t < Max\_Iteration$  do
6:   for each solution do
7:     Update  $a$ ,  $A$ , and  $p$ 
8:     if  $p < 0.5$  then
9:       if  $|A| < 1$  then
10:         $X \leftarrow LS\_FD(X, N)$ 
11:       else
12:         $X \leftarrow LS\_FDT(X, N, vT)$ 
13:        if  $t \bmod L == 0$  then
14:           $vT \leftarrow Z(X)$ 
15:        end if
16:       end if
17:     else
18:        $X \leftarrow PVND(X, N)$ 
19:     end if
20:   end for
21:   Calculate the fitness of each solution
22:   Update  $X^*$  if there is a better solution
23:    $t \leftarrow t + 1$ 
24: end while
25: return  $X^*$ 

```

Figure 5.4: Procedure of the WOARB-1

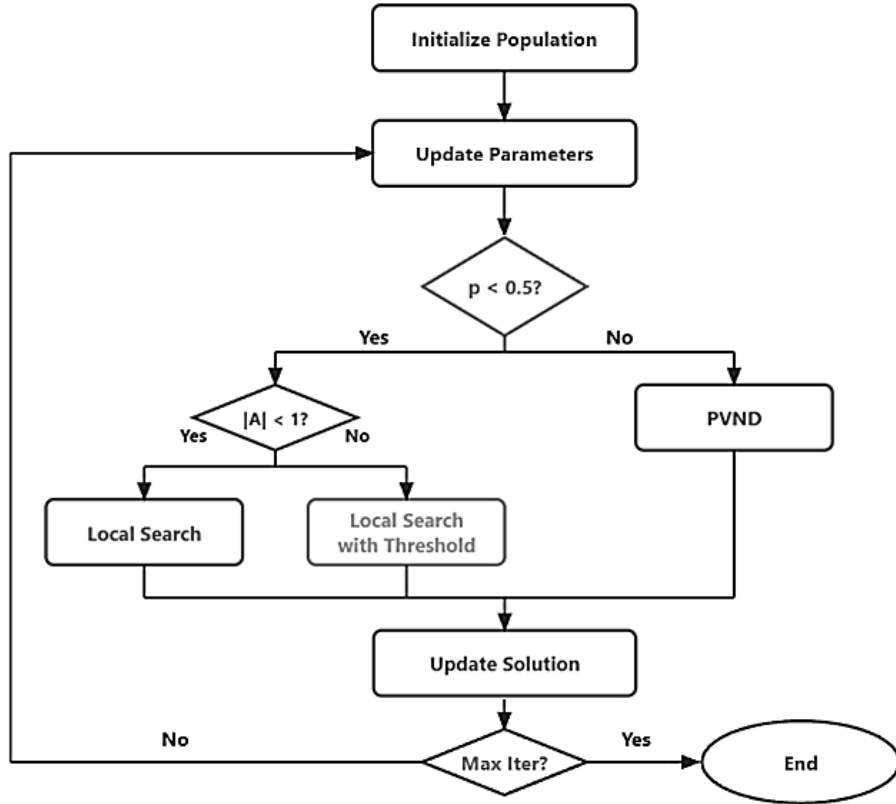


Figure 5.5: The Framework of the WOARB-1 Algorithm

The algorithm of WOARB-2 model, illustrated in Figure 5.6, is also guided by a randomly generated probability $p \in [0, 1]$ during the search updates. When $p < 0.5$, LS_FD procedure (Line 12) is applied if $|A| < 1$ which substitutes Equation 2.2 from the original algorithm, and LS_FDT procedure (Line 14-17) is used to replace Equation 2.8 when $|A| \geq 1$. However, when $p \geq 0.5$, the algorithm employs IT-PVND procedure (Lines 20-23) which takes the place of Equation 2.6 from the original algorithm to enhance exploitation. The corresponding framework is shown in Figure 5.7.

Algorithm 5.5 Random-based Whale Optimization Algorithm (WOARB-2)

```
1: Generate Initial Population  $X_i$  ( $i = 1, 2, 3$ )
2: Calculate the fitness of each solution
3: Initialize  $L_1, L_2$ 
4:  $X^* \leftarrow$  the best search agent
5:  $vT \leftarrow Z(X)$ 
6:  $cMax \leftarrow Z(X)$ 
7: while  $t < Max\_Iteration$  do
8:   for each solution do
9:     Update  $a, A$ , and  $p$ 
10:    if  $p < 0.5$  then
11:      if  $|A| < 1$  then
12:         $X \leftarrow LS\_FD(X, N)$ 
13:      else
14:         $X \leftarrow LS\_FDT(X, N, vT)$ 
15:        if  $t \bmod L_1 == 0$  then
16:           $vT \leftarrow Z(X)$ 
17:        end if
18:      end if
19:    else
20:       $X \leftarrow IT\_PVND(X, N, cMax)$ 
21:      if  $t \bmod L_2 == 0$  then
22:         $cMax \leftarrow Z(X)$ 
23:      end if
24:    end if
25:  end for
26:  Calculate the fitness of each solution
27:  Update  $X^*$  if there is a better solution
28:   $t \leftarrow t + 1$ 
29: end while
30: return  $X^*$ 
```

Figure 5.6: Procedure of the WOARB-2

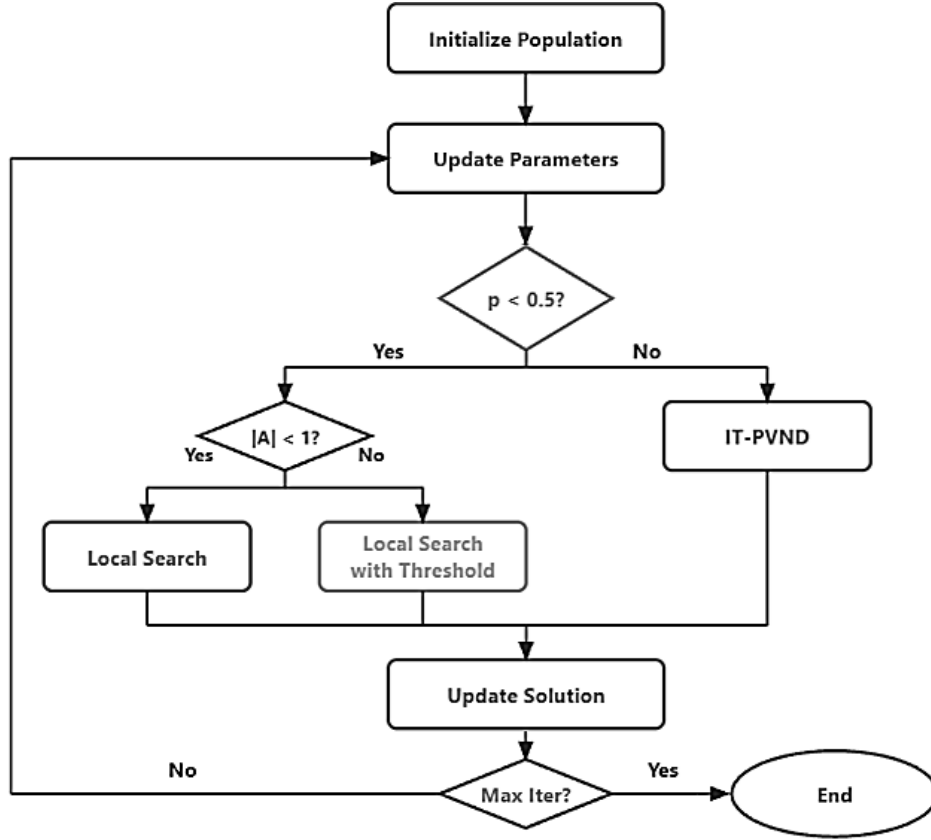


Figure 5.7: The Framework of the WOARB-2 Algorithm

The WOARB model and its corresponding algorithm, presented in Figure 5.8, eliminate the dependency on probability p for controlling behavioral phases. Instead, it divides the optimization process into two equal temporal phases based on the current iteration count. During the first half of the iterations, the algorithm focuses on both exploitation and exploration: it applies the LS_FD procedure (Line 12) when $|A| < 1$, which substitutes Equation 2.2 from the original algorithm, and switches to the LS_FDT procedure (Lines 14-17), replacing Equation 2.8, when $|A| \geq 1$. In the second half of the iterations, it employs the IT-PVND procedure (Lines 20-23), replacing Equation 2.6, to prioritize the exploitation phase, concentrating search efforts on intensifying around the current best solution. The overall framework is depicted in Figure 5.9.

Algorithm 5.6 Iteration-based Whale Optimization Algorithm (WOAIB)

```
1: Generate Initial Population  $X_i$  ( $i = 1, 2, 3$ )
2: Calculate the fitness of each solution
3: Initialize  $L_1, L_2$ 
4:  $X^* \leftarrow$  the best search agent
5:  $vT \leftarrow Z(X)$ 
6:  $cMax \leftarrow Z(X)$ 
7: while  $t < Max\_Iteration$  do
8:   for each solution do
9:     Update  $a, A$ , and  $p$ 
10:    if  $iter < 0.5 \times MaxIter$  then
11:      if  $|A| < 1$  then
12:         $X \leftarrow LS\_FD(X, N)$ 
13:      else
14:         $X \leftarrow LS\_FDT(X, N, vT)$ 
15:        if  $t \bmod L_1 == 0$  then
16:           $vT \leftarrow Z(X)$ 
17:        end if
18:      end if
19:    else
20:       $X \leftarrow IT\_PVND(X, N, cMax)$ 
21:      if  $t \bmod L_2 == 0$  then
22:         $cMax \leftarrow Z(X)$ 
23:      end if
24:    end if
25:  end for
26:  Calculate the fitness of each solution
27:  Update  $X^*$  if there is a better solution
28:   $t \leftarrow t + 1$ 
29: end while
30: return  $X^*$ 
```

Figure 5.8: Procedure of the WOAIB

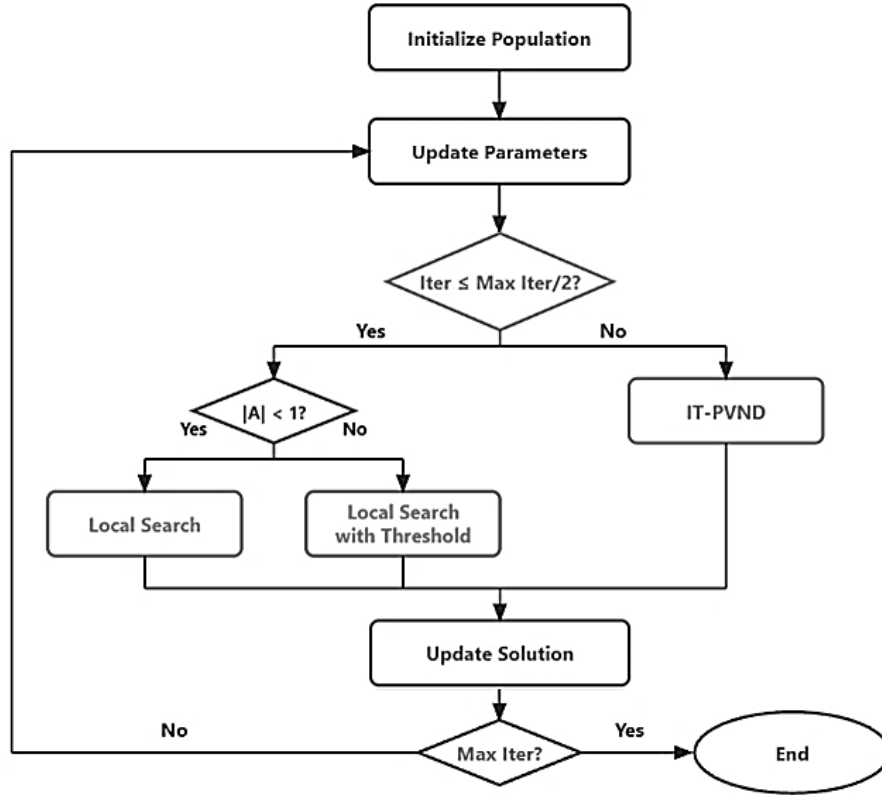


Figure 5.9: The Framework of the WOAIB Algorithm

5.3 Experiment Results for WOA Models

To assess the effectiveness of the three WOA models, these algorithms are implemented using Java, utilizing the same computer resources as those employed in the previous chapter. The same dataset as in Chapter 4 is used for experimentation, and the objective values are calculated over 30 independent runs. The results from each WOA method are presented in Table 5.1, comparing the mean, best, worst, and standard deviation values.

The objective values in Table 5.1 indicate that the iteration-based WOAIB method outperforms than the random probability WOARB-1 and WOARB-2 methods. WOAIB demonstrated the best average performance with mean values of 122.5 for FEB23, 118.7 for FCSHD23, 132.0 for FEB24, and 51.2 for FCSIT24. In contrast, WOARB-1 shows the worst average performance in all instances.

Table 5.1: Results Obtained from WOARB-1, WOARB-2, and WOAIB

Instance	N	Algorithm	Best	Mean	Worst	Std. Dev.
FEB23	30	WOARB-1	121.9	125.3	129.9	2.114
		WOARB-2	120.7	123.7	127.9	1.529
		WOAIB	117.4	122.5	126.3	2.366
FCSHD23	30	WOARB-1	113.5	125.0	139.4	6.338
		WOARB-2	113.5	123.3	133.8	5.707
		WOAIB	110.2	118.7	125.9	4.282
FEB24	30	WOARB-1	130.3	132.9	135.7	1.211
		WOARB-2	131.0	132.9	135.3	1.046
		WOAIB	129.9	132.0	134.0	1.139
FCSIT24	30	WOARB-1	49.8	52.9	57.7	2.090
		WOARB-2	49.0	52.0	54.7	1.404
		WOAIB	48.7	51.2	53.7	1.275

Bold values indicate the best objective values; Std. Dev. = Standard Deviation

The standard deviations indicate slight differences in solution consistency, with WOARB-2 achieving the best results in two instances, FEB23 and FEB24, while WOAIB outperformed in the remaining two, FCSHD23 and FCSIT24. This suggests that both algorithms perform reliably with varying variability. Moreover, the results confirm that the IT-PVND method offers greater advantages than the PVND method.

Table 5.2 presents the optimal objective values and the average running time of the five proposed models, including the WOA-based models and the two PVND approaches introduced in Chapter 4. The results are gathered over 30 independent runs, each with 10

iterations. The mean objective values for the five approaches are PVND at 105.3, IT-PVND at 102.6, WOARB-1 at 103.9, WOARB-2 at 103.6, and WOAIB at 101.6. The ordering of these mean values from minimum to maximum, $WOAIB < IT-PVND < WOARB-2 < WOARB-1 < PVND$.

The mean computational times for the algorithms are as follows: PVND (35s), IT-PVND (43s), WOARB-1 (40s), WOARB-2 (46s), and WOAB (41s). Execution times across algorithms show minimal variation, with minor differences that do not significantly impact practical usability. While PVND and WOARB-1 achieve the fastest speeds, they sacrifice solution quality compared to other approaches. WOAIB achieves the best quality overall, while IT-PVND secures second place in accuracy and remains competitively fast. This trade-off between quality and time highlights that WOAIB and IT-PVND offer the most practical balance for real-world applications.

Figure 5.10 presents the box plots for the four instances generated using Tableau, a business intelligence and analytics software. WOAIB demonstrates a faster convergence towards optimal solutions, as indicated by its lower box plot height compared to the other algorithms in most instances. The lower mean, median, and distribution values observed in the box plots, consistent with the statistical analysis in Table 5.2, confirm that WOAIB consistently outperforms all other methods. In contrast, PVND emerges as the poorest performer, characterised by higher median values and greater variability, rendering it the least effective approach. IT-PVND ranks second, showing respectable performance, though not as optimal as WOAIB.

Table 5.2: Results of the Comparisons Among the Five Algorithms

Instance	Algorithm	Optimal Solution	Median	Average Time (s)
FEB23	PVND	123.9	128.4	29
	IT-PVND	118.3	123.7	29
	WOARB-1	121.9	124.0	25
	WOARB-2	120.7	124.6	26
	WOAIB	117.4	122.8	27
FCSHD23	PVND	115.1	125.7	62
	IT-PVND	112.4	121.0	78
	WOARB-1	113.5	122.4	76
	WOARB-2	113.5	124.5	82
	WOAIB	110.2	119.7	74
FEB24	PVND	132.1	134.8	38
	IT-PVND	130.3	132.6	49
	WOARB-1	130.3	132.9	45
	WOARB-2	131.0	132.9	55
	WOAIB	129.9	132.0	45
FCSIT24	PVND	50.1	53.0	10
	IT-PVND	49.4	51.5	15
	WOARB-1	49.8	52.2	13
	WOARB-2	49.0	52.9	22
	WOAIB	48.7	51.1	17

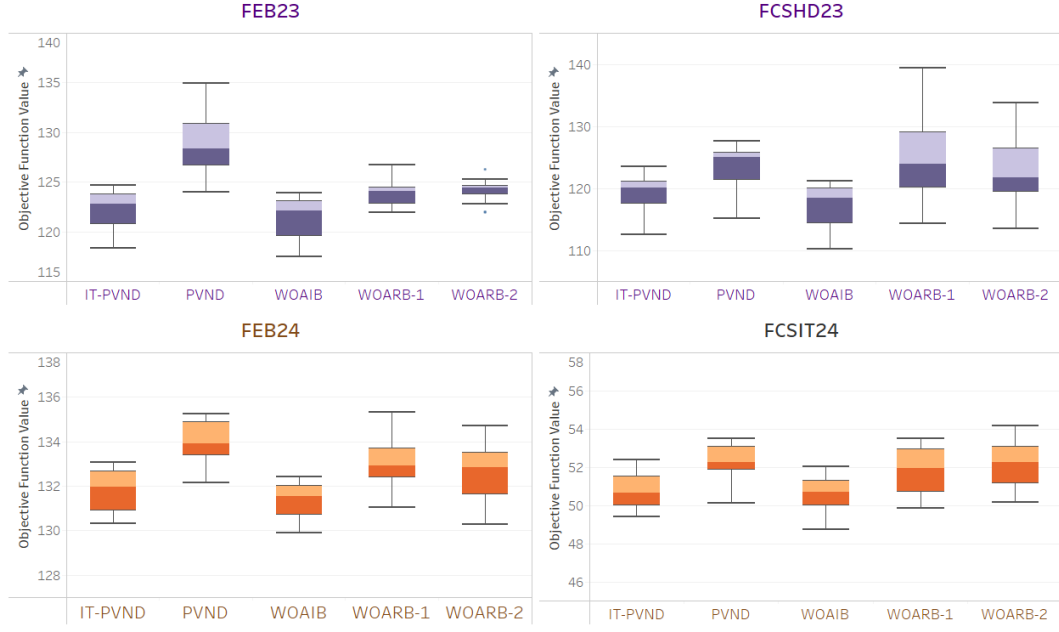


Figure 5.10: Box Plot using the Objective Function Value for Instances

Integrating local search with WOA improves convergence behavior by strengthening the algorithm's exploitation ability, enabling it to improve solutions more effectively. WOARB-2 provides a baseline improvement, while WOAIB demonstrates superior performance. The incorporation of a threshold mechanism and periodic updates in WOAIB enhances exploration while maintaining diversity in the search space, resulting in a more balanced exploration–exploitation trade-off.

5.4 Conclusion

The contribution in this chapter includes presenting a modified discrete WOA approach incorporating simple local search methods to replace the original equation, enabling the algorithm to address the ETP effectively. Three local search methods are employed: first descent method, first descent with a dynamically changing threshold method, and PVND/IT-PVND methods. The first descent method and PVND/IT-PVND are employed in the exploitation phase to refine the solutions. In contrast, the first descent method with a

dynamically changing threshold is used in exploration, updating during the search to accept solutions with objective values below the threshold.

Three modified WOA models are proposed: WOARB-1, WOARB-2, and WOAIB. WOARB-1 and WOARB-2 utilize a random probability mechanism, whereas WOAIB implements a structured approach by partitioning the iterations. The performance of the proposed algorithms was evaluated and compared based on their objective values, with the ranking determined as follows: $WOAIB < IT-PVND < WOARB-2 < WOARB-1 < PVND$. This ranking confirms that WOAIB is the most effective algorithm, followed by IT-PVND.

In conclusion, the modified WOA has demonstrated its effectiveness when suitably adapted to real-world ETP. Additionally, since exploration in WOA is governed by the position updates of agents relative to randomly selected solutions, incorporating an alternative mechanism, such as IT-PVND, enhances the algorithm's exploration. Notably, the iterative threshold parameter allows weaker solutions to persist longer in the search process, thereby promoting greater diversity in the population and improving overall search performance. Overall, it balances exploration and exploitation while maintaining acceptable computational efficiency, underscoring its potential for addressing complex combinatorial optimization challenges. Consequently, the WOAIB and IT-PVND approaches are selected for further experimentation in the subsequent chapter to validate their effectiveness using the most widely recognized benchmark dataset for the ETP.

CHAPTER 6

ADDRESSING TORONTO BENCHMARK PROBLEM

6.1 Introduction

In this chapter, the implementation and experimentation of the proposed algorithms, specifically IT-PVND and WOAIB, are presented on the Toronto benchmark datasets. This experiment evaluates whether the proposed algorithms can address the Toronto benchmark problem as effectively as they did with the UNIMAS ETP. The Toronto benchmark problem was introduced to provide researchers with a standardized real-world ETP for testing their algorithms. The chapter is structured in the following way: Section 6.1 describes the neighborhood structure and adjustment of the proposed algorithms for addressing the Toronto ETP. Section 6.2 outlines the experimental setup, while Section 6.3 presents the results and discussion of the proposed methods. Finally, Section 6.4 summarizes the conclusion.

The Toronto dataset, introduced by Carter et al. (1996), represents the uncapacitated variant of the ETP, comprising 13 real-world cases from universities, primarily located in Canada. The datasets vary significantly in conflict density and size, resulting in high problem complexity. Toronto ETP remains a widely studied benchmark in research, primarily due to its challenging instances; to date, only one instance has been solved to prove optimality (Ceschia et al., 2023).

Toronto version I was selected for the study as it is most widely used in literature over the last decade and is challenging as a dataset of real-world instances. Also, the constraints employed in the Toronto benchmark are identical to those in the UNIMAS exam problem

studied in Chapter 3, with the hard constraint (HC2) ensuring feasibility by preventing conflicting exams from being scheduled in the same timeslot and the soft constraint (SC2) aiming to minimize the proximity cost. The objective function is associated with SC2, the spread between exams that share common students, as outlined in Equation 3.7 in chapter 3.

Since the proposed algorithms, WOAIB and IT-PVND, handle both uncapacitated and capacitated exams within a unified framework, several minor adjustments have been made to adapt them to the Toronto problem. These modifications primarily involve removing room constraints and accounting for considerations related to room sharing and splitting. Additionally, for the uncapacitated Toronto problem, an additional set of neighborhood structures was incorporated, and perturbation procedures were utilized within the existing algorithms.

6.1.1 Neighborhood Structure

Ten neighborhoods were designed and utilized in both the WOAIB and IT-PVND algorithms. All these neighborhoods are considered only within the feasible region, each of which will be elaborated upon below.

- i. Swap Exams, denoted by $NS_1(e_i, e_j)$, takes as attributes two exams $e_i, e_j \in E$, $e_i \neq e_j$. The move consists of swapping the periods of e_i and e_j , where $p(e_i) \neq p(e_j)$.
- ii. Double Swap Exams, denoted by $NS_2(e_i, e_j, e_k)$, takes as attributes three exams $e_i, e_j, e_k \in E$, $e_i \neq e_j$ and $e_j \neq e_k$. The move involves swapping e_i in one period, denoted by $p(e_i)$, with e_j and e_k , in period denoted by $p(e_j)$, where $p(e_i) \neq p(e_j)$ and $p(e_j) = p(e_k)$.

- iii. Kick Exams, denoted by $NS_3(e_i, e_j, p)$, takes as attributes two exams $e_i, e_j \in E$, $e_i \neq e_j$ and one period $p \in P$. The move involves assigning e_i to $p(e_j)$, and assigning e_j to a randomly selected period p , where $p(e_i) \neq p(e_j) \neq p$.
- iv. Double Kick Exams, denoted by $NS_4(e_i, e_j, e_k, p)$, takes as attributes three exams $e_i, e_j, e_k \in E$, $e_i \neq e_j \neq e_k$. The move involves assigning e_i to $p(e_j)$, assigning e_j to $p(e_k)$, and assigning e_k to a randomly selected period p , where $p(e_i) \neq p(e_j) \neq p(e_k) \neq p$.
- v. Move Exam, denoted by $NS_5(e, p)$, takes as attribute one exam $e \in E$, and one period $p \in P$. The move involves assigning e to a randomly selected period p , where $p(e) \neq p$.
- vi. Shift Periods, denoted by NS_6 , where the move involves cyclically shifting the exams scheduled in all periods.
- vii. Random Kempe Chain Move, denoted by $S_1(e, p)$, takes as attribute one exam $e \in E$, and one period $p \in P$. The move involves assigning e to a randomly selected period p , with $p(e) \neq p$. All exams in period p that conflict with e are moved to $p(e)$, and vice versa, until no more conflicts exist.
- viii. Kempe Chain Move, denoted by $DS_1(p)$, takes as attribute one period $p \in P$. The move involves iterating through all available periods, attempting to swap a chain of conflicting exams between the selected period p and other period. The process halts once a successful move is found.

- ix. Random Swap Periods, denoted by $S_2(p_i, p_j)$, takes as attributes two periods $p_i, p_j \in P$ and $p_i \neq p_j$. The move involves assigning all exams in p_i to p_j and vice versa.
- x. Swap Periods, denoted by $DS_2(p)$, takes as attribute one period $p \in P$. The move involves iterating through all available periods, attempting to swap all exams between the selected period, p and other period. The process halts once a successful swap is found.

The NS_4 neighborhood builds upon the concept of the NS_3 neighborhood but features a longer chain. Introduced by González-Velarde and Laguna (2002) for graph coloring, it combines the NS_1 , NS_3 , and NS_4 neighborhoods under the term “Ejection Chains.” The neighborhood structures NS_1 through NS_6 are utilized for the local search in both the IT-PVND and WOAIB algorithms, while S_1 , S_2 , DS_1 , and DS_2 are employed for perturbation.

The proposed IT-PVND approach, along with IT-PVND methods within the WOAIB model, consists of two key components: a local search method and a neighborhood change mechanism. During the improvement phase, local search methods are systematically applied to refine a given solution, while the neighborhood change step facilitates exploration within the solution space. Additionally, two perturbation procedures are introduced to enhance the algorithm’s capability to explore different regions of the search space and avoid premature convergence.

6.1.2 Perturbation Procedures

In this work, two distinct perturbation procedures were implemented: a random shake procedure and a probabilistic perturbation procedure, both designed to help escape local

optima. The algorithm for shaking procedure is presented in Figure 6.1, while the algorithm for probabilistic perturbation procedure is shown in Figure 6.2.

Algorithm 6.1 Shaking Procedure

```

1: function SHAKING( $X$ )
2:   Set  $N = \{S_1, S_2\}$ 
3:   Select a random neighborhood  $N_r \in N$ 
4:    $X' \leftarrow$  Apply  $N_r$  to  $X$ 
5:   if  $Z(X') < (Z(X)$  and move sequence  $\notin$  Tabu List then
6:      $X \leftarrow X'$ 
7:     Insert move sequence in Tabu List
8:     return  $X$ 
9:   end if
10: end function

```

Figure 6.1: Shaking Procedure

Algorithm 6.2 Perturbation Procedure

```

1: function PERTURBATION( $X, p = 0.0$ )
2:    $\triangleright p$ : Perturbation probability for accepting worse solutions
3:   Set  $N = \{DS_1, DS_2\}$ 
4:   for  $S_i \in N$  do
5:      $X' \leftarrow$  Apply  $S_i$  to  $X$ 
6:     if  $Z(X') < (Z(X) \cdot p + Z(X))$  then
7:        $X \leftarrow X'$ 
8:       Insert move sequence in Tabu List
9:       return  $X$ 
10:    end if
11:  end for
12:  return  $X$ 
13: end function

```

Figure 6.2: Probabilistic Perturbation Procedure

The shaking procedure is executed in every iteration of the algorithm. In contrast, the probabilistic perturbation procedure, which features a larger search region, is applied if no improvement is observed to escape local optima. Both procedures utilize local search but differ in their neighborhood structures they apply; the shaking procedure uses S_1 and S_2 , while the probabilistic perturbation procedure employs DS_1 and DS_2 .

In probabilistic perturbation procedure, a new candidate solution X' is accepted if it satisfies the condition $Z(X') < Z(X) \cdot p + Z(X)$, where $Z(X)$ represents the objective value of the current solution, and p represents the perturbation probability allowing the acceptance of worse solutions. The default value for the perturbation probability, p , is set to 0.0, meaning no worse solutions are accepted. However, if a local optimum is encountered, the parameter p is adjusted to accept worse solutions, thereby facilitating an escape from the local optimum.

Occasionally, accepting solutions inferior to the current solution for specific iterations below a threshold value can encounter a challenge known as cycling, where the algorithm becomes trapped in a repetitive sequence of solutions. To prevent cycling back to previously explored solutions, a Tabu List is implemented in both procedures to record previously visited moves, thereby preventing the algorithm from revisiting the same sequence. The Tabu List operates on a First In, First Out (FIFO) protocol, meaning that the next solution with the earliest recorded date is overwritten.

6.2 Experiment Setup

In this study, the IT-PVND algorithm from Chapter 4 and the WOAIB algorithm from Chapter 5 were implemented with adjustments discussed in the preceding section. Each experiment for both IT-PVND and WOAIB algorithms was conducted 5 times, with the number of iterations ranging from 2,000 to 10,000, determined by the size of the exam or terminating upon reaching 30 local optima, while not exceeding 400 hours (Woumans et al., 2016). Fewer and longer runs of the algorithm were performed to avoid numerous short runs, as this strategy has been shown to be ineffective for specific algorithms (Burke et al., 2004; Burke & Bykov, 2016). In real-world scenarios, running the exam scheduling process for

several hours is generally acceptable, as the timetable is often created months or weeks in advance, making the time required for its construction non-critical.

6.3 Results and Discussion

Table 6.1 compares the best, mean, and worst results achieved by two proposed algorithms, IT-PVND and WOAIB, across 12 Toronto instances. The PUR93 instance is excluded from the results as it could not be successfully executed on the experimental machine due to its size, memory demands, and computational costs (Sharma, 2020). Both algorithms performed equally well for STA83, while IT-PVND slightly outperformed WOAIB in CAR91 and UTE92. In all other instances, WOAIB outperformed IT-PVND by achieving lower best values. WOAIB consistently achieved better or similar results compared to IT-PVND across most instances.

Table 6.1: Statistical Summary of Results from IT-PVND and WOAIB

Instance	IT-PVND			WOAIB		
	Best	Mean	Worst	Best	Mean	Worst
CAR91	5.29	5.37	5.49	5.5	5.73	6.34
CAR92	4.33	4.45	4.61	4.17	4.23	4.26
EAR83	34.81	35.03	35.46	33.12	33.72	34.58
HEC92	10.58	10.71	10.95	10.43	10.59	10.7
KFU93	13.87	14.29	15.13	13.39	13.73	14.04
LSE91	11.14	11.29	11.45	10.8	10.94	11.06
RYE92	8.95	9.07	9.21	8.73	8.91	9.08
STA83	157.03	157.16	157.28	157.03	157.09	157.16
TRE92	8.48	8.66	8.91	8.2	8.448	8.94

Table 6.1 continued

UTA92	3.60	3.63	3.66	3.48	3.568	3.64
UTE92	24.95	25.49	26.42	24.97	25.33	25.62
YOR83	37.15	37.64	38.39	35.62	36.14	36.56

Bold values indicate the best objective function values.

6.3.1 Comparison of Results with Population-Based Methods

As one of the proposed algorithms, WOAIB is population-based; therefore, its performance is compared with that of other algorithms in the same category. This section evaluates the proposed algorithms by comparing their performance with ten population-based methods from the literature, using the Toronto datasets as detailed in Table 6.2. A “-” indicates instances where no feasible solution was generated, and the bold values represent the best solutions for each instance. The columns A–J in Table 6.2 correspond to references from literature that utilize population-based methods, as detailed in Table 6.3.

Table 6.4 compares the ranking of the proposed methods against population-based methods. The difference ratio formula is defined in Equation 6.1.

$$\text{Difference Ratio} = \frac{|Best\ Result\ Cited - Our\ Best\ Result|}{Best\ Result\ Cited} \quad \text{Equation 6.1}$$

According to Table 6.2, WOAIB algorithm performed better than other population-based approaches, achieving the best results in four instances (EAR83, KFU93, YOR83, and STA83). Similarly, the IT-PVND algorithm yielded competitive results, outperforming others in two instances of the Toronto dataset (STA83 and UTE92). Although WOAIB ranked lower at 10th out of 12 in CAR91, it consistently achieved a strong overall ranking among population-based methods, as summarized in Table 6.4.

Table 6.2: Comparisons to Population-Based Methods

Instance	A	B	C	D	E	F	G	H	I	J	IT-PVND	WOAIB
CAR91	5.2	5.4	3.2	5.7	5.4	4.8	5.4	5.0	5.2	–	5.3	5.5
CAR92	4.3	4.8	4.7	4.8	4.6	3.9	4.5	4.2	4.4	5.9	4.3	4.2
EAR83	36.8	37.5	35.7	36.8	38.6	33.6	36.3	34.1	34.7	–	34.8	33.1
HEC92	11.1	11.2	10.7	11.3	11.2	10.6	10.7	10.3	10.2	11.5	10.6	10.4
KFU93	14.5	15.1	14.5	15.0	14.9	13.4	15.2	13.9	13.5	–	13.9	13.4
LSE91	11.3	12.1	10.8	12.1	11.7	10.9	11.9	11.0	11.3	–	11.1	10.8
RYE92	9.8	-	10.0	10.2	9.8	8.8	8.6	9.2	8.7	–	9.0	8.7
STA83	157.3	157.5	157.1	157.2	157.2	157.1	158.2	157.0	157.0	–	157.0	157.0
TRE92	8.6	9.2	8.5	8.8	9.0	7.9	8.9	8.4	8.5	8.0	8.5	8.2
UTA92	3.5	3.9	3.5	3.8	3.7	3.3	3.6	3.4	3.6	3.6	3.6	3.5
UTE92	26.4	27.6	25.9	27.7	26.9	25.4	25.3	28.8	25.0	–	25.0	25.0
YOR83	39.4	40.9	38.7	39.6	39.3	35.7	40.3	36.5	36.2	–	37.2	35.6

Table 6.3: Compared Population-Based Methods

Column	Reference	Approach
A	Eley (2007)	Ant Algorithm
B	Alzaqebah and Abdullah (2011)	Artificial Bee Colony
C	Ahandani et al. (2012)	Particle Swarm Optimization
D	Turabieh and Abdullah (2011)	Electromagnetic-Like Mechanism and Particle Swarm Optimization
E	Bolaji et al. (2012)	Artificial Bee Colony
F	Abdullah and Alzaqebah (2013)	Self-Adaptive Bees Algorithm
G	Abed and Tang (2012)	Genetic Algorithm
H	Bolaji et al. (2015)	Artificial Bee Colony
I	Nand et al. (2022)	Firefly Algorithm
J	Tilahun (2019)	Prey Predator Algorithm

Table 6.4: Rank the Proposed Methods against Population-Based Methods

Instance	IT-PVND	WOAIB	Best Result Obtained	Best Result Cited	Difference Ratio (%)
CAR91	6	10	5.29	3.22	64
CAR92	5	2	4.17	3.94	6
EAR83	5	1	33.12	33.61	1
HEC92	5	3	10.43	10.20	2
KFU93	4	1	13.39	13.44	0
LSE91	5	2	10.80	10.76	0

Table 6.4 continued

RYE92	5	2	8.73	8.60	2
STA83	1	1	157.03	157.03	0
TRE92	6	3	8.20	7.94	3
UTA92	8	3	3.48	3.27	6
UTE92	1	2	24.95	25.00	0
YOR83	5	1	35.62	35.74	0

6.3.2 Comparison of Results with Other Metaheuristic Methods

Metaheuristics have gained prominence due to their ability to generate superior solutions compared to those produced by sequential heuristics alone. In university ETP, an initial solution is typically derived using a suitable heuristic, followed by refinement through metaheuristic methods. The proposed algorithms are evaluated by comparing their performance with that of other metaheuristic methods, as detailed in Table 6.5. Columns A–J in Table 6.5 refer to literature that uses other metaheuristic methods, as detailed in Table 6.6. Column G in Table 6.5 shows the state-of-the-art solutions for instances CAR91, CAR92, EAR83, HEC92, KFU93, LSE91, RYE92, STA83, TRE92 and UTA92, whereas Column A presents the state-of-the-art solutions for instances KFU93, LSE91 and STA83.

Table 6.5: Comparisons to Other Metaheuristic Methods

Instance	A	B	C	D	E	F	G	H	I	J	IT-PVND	WOAIB
CAR91	4.3	4.8	4.9	5.1	5.2	5.2	4.2	5.7	6.2	6.4	5.3	5.5
CAR92	3.7	3.9	4.2	4.3	4.3	4.4	3.6	–	5.2	5.4	4.3	4.2
EAR83	32.6	33.4	35.9	35.1	35.8	34.9	32.4	39.4	45.7	39.8	34.8	33.1
HEC92	10.1	10.5	11.5	10.6	11.2	10.3	10.0	10.9	12.4	11.2	10.6	10.4
KFU93	12.8	13.7	14.4	13.5	14.5	13.5	12.8	–	18.0	16.0	13.9	13.4
LSE91	9.8	10.3	10.9	10.5	10.9	10.2	9.8	12.6	15.5	13.7	11.1	10.8
RYE92	7.9	–	8.8	–	–	9.8	7.8	–	–	12.1	9.0	8.7
STA83	157.0	157.1	157.8	157.3	157.2	159.2	157.0	157.4	160.8	157.2	157.0	157.0
TRE92	7.6	7.9	8.4	8.4	8.5	8.4	7.6	–	10.0	9.5	8.5	8.2
UTA92	3.0	3.1	3.4	3.5	3.4	3.6	3.0	4.1	4.2	4.3	3.6	3.5
UTE92	24.8	25.3	27.2	25.1	26.7	26.0	24.8	–	27.8	27.3	25.0	25.0
YOR83	34.7	36.1	39.3	37.4	39.5	36.2	34.4	39.7	41.0	40.5	37.2	35.6

Table 6.7 compares the rankings of the proposed methods with those of the other methods. As summarized in Table 6.7, the ranking highlights the strong performance of IT-PVND and WOAIB in specific instances. Notably, both approaches achieved the top rank for STA83. Additionally, WOAIB consistently performed well, securing high ranks across several instances—such as EAR83, KFU93, RYE92, and YOR83—demonstrating its competitiveness. While IT-PVND excelled in certain instances, such as UTE92, its overall performance was less consistent, ranking lower in other cases.

Table 6.6: Compared Metaheuristic Methods

Column	Reference	Approach
A	Burke and Bykov (2016)	Great Deluge
B	Fong et al. (2015)	Hybrid Swarm-based Approach
C	Pillay and Banzhaf (2010)	Hyper-heuristic
D	Merlot et al. (2003)	Constraint Programming, Simulated Annealing, and Hill Climbing
E	Burke et al. (2014)	Heuristics
F	Abdullah et al. (2007)	Tabu Search
G	Bellio et al. (2021)	Simulated Annealing
H	Di Gaspero (2002)	Tabu Search
I	Di Gaspero and Schaerf (2001)	Tabu Search
J	Lei et al. (2014)	Genetic Algorithm and Simulated Annealing

Table 6.7: Rank the Proposed Methods against Other Metaheuristic Methods

Instance	IT-PVND	WOAIB	Best Result Obtained	Best Result Cited	Difference Ratio (%)
CAR91	8	9	5.29	4.24	25
CAR92	8	4	4.17	3.64	15
EAR83	5	3	33.12	32.42	2
HEC92	6	4	10.43	10.03	4
KFU93	7	3	13.39	12.80	5

Table 6.7 continued

LSE91	9	6	10.80	9.78	10
RYE92	5	3	8.73	7.84	11
STA83	1	1	157.03	157.03	0
TRE92	8	4	8.20	7.59	8
UTA92	9	6	3.48	2.95	18
UTE92	3	4	24.95	24.76	1
YOR83	6	3	35.62	34.40	4

The proposed algorithms were assessed in comparison to 20 approaches from the literature, as detailed in Tables 6.5 and 6.7. The findings demonstrate that the proposed WOAIB algorithm outperforms many metaheuristic approaches reported in the literature, showcasing its competitive performance compared to existing methods. Notably, the WOAIB algorithm demonstrates more consistent performance on most instances; its relative underperformance on CAR91 can be attributed to the instance's large size and complex distribution of student enrollments, which create tightly coupled exam clusters. The algorithm still performs well on other large instances such as CAR92 and UTA, indicating that the difficulty with CAR91 arises from its unique structural characteristics rather than a general limitation on large-scale problems.

Overall, the results indicate that the algorithm is capable of efficiently generating high-quality solutions while maintaining adaptability across diverse instance sizes and conflict structures. Although the proposed algorithms do not surpass the best-known results, the findings indicate that the proposed algorithms, particularly the WOAIB algorithm, perform

competitively in most instances. At the same time, IT-PVND achieves results comparable to those reported in the literature.

6.4 Conclusion

The IT-PVND and WOAIB algorithms were employed in this chapter to demonstrate their ability to generate feasible solutions while validating their effectiveness on the Toronto benchmark problem. Both algorithms involved minor modifications to remove the allocation of exams to rooms, as room constraints were not considered in this problem. Additionally, the algorithms were enhanced by incorporating two perturbation procedures and designing seven new neighborhood structures.

An experiment consisting of five independent runs on each Toronto instance was conducted. The results, compared against 20 published approaches—including 10 population-based methods and 10 other metaheuristic methods—demonstrated that the proposed IT-PVND and WOAIB algorithms achieved competitive rankings among state-of-the-art methods. According to the experimental results on 12 Toronto instances, the proposed WOA algorithm outperformed IT-PVND in most cases and demonstrated competitive performance compared to methods from the literature.

Experimental results validate these proposed algorithms against a benchmark dataset, demonstrating their effectiveness for tackling ETP and serving as preliminary proof of concept. Although both algorithms were initially designed to address real-world ETP, they also perform effectively on the Toronto benchmark datasets with minor modifications. The findings highlight the potential of these algorithms in addressing the Toronto ETP, offering the timetabling community alternative solution approaches. Given that the WOAIB

algorithm yields competitive results, further exploration of the WOA metaheuristic in educational timetabling is worthwhile

CHAPTER 7

CONCLUSION

7.1 Research Summary

This chapter summarizes the work presented in this thesis. Section 7.1 summarizes the key findings of the overall research, while Section 7.2 outlines the primary contributions of this work. Lastly, Section 7.3 discusses recommendations for future research.

This study pursued three objectives. First, it aimed to formulate a mathematical model for the UNIMAS examination timetabling problem. Second, it sought to develop a unified approach that leverages Variable Neighborhood Descent and Whale Optimization Algorithm. This approach integrates both uncapacitated and capacitated variants of the real-world faculty exam timetabling problem. Third, it conducted a statistical analysis of the proposed algorithms. It also compared their performance against proprietary software and a manual method, validated using a benchmark problem.

A systematic review of the leading publications from the last decade provided the foundation for understanding the methods and current trends that have shaped this field. A mathematical model grounded in practical institutional data at UNIMAS that incorporates the latest operational constraints to evaluate timetable efficiency. The standard soft constraints, generally applicable to all faculties, including exam spread, slot preference, reduced room usage, and minimized exam splits, are embedded into a single objective function to evaluate the quality of the solution in a general way. The consideration of constraints differs from others in the literature, as most centralized timetabling approaches apply standard soft constraints universally across the entire university. In contrast, the

decentralized faculty timetabling approach also considers extended soft constraints that adapt according to the specific preferences of each faculty, which may vary significantly from one another. The extended soft constraints include minimizing the exam period, accommodating room-sharing preferences, and balancing a split exam.

Second, a unified approach that leverages Variable Neighborhood Descent (VND) that integrates both uncapacitated and capacitated variants was designed and evaluated. The proposed VND with six tailored neighborhood structures successfully addressed all constraints while significantly improving solution quality: an 8% gain in room utilization, 23% better exam spread, 20% improvement in room/area split, and 39% higher slot preference fulfilment. Collectively, these enhancements resulted in an overall 15% improvement in the objective function and efficiency gains across diverse dataset groups (10%, 14%, 11%, and 33% for FEB23, FSCHD23, FEB24, and FCSIT24, respectively). Leveraging the same multi-neighborhoods, variants and extensions, including Base VND, Pipe VND, and Iterative Threshold Pipe VND (IT-PVND) consistently outperformed proprietary software and manual practices, with improvements of 10.0%, 10.4%, and 12.7% respectively.

Building upon the improvements, the research introduced a novel hybridization of Whale Optimization Algorithm (WOA) with local search and VND, marking the first application of this technique to the ETP. WOA was initially designed for continuous problems but has been modified to address the discrete exam timetabling problem. Three models of modified discrete WOA have been developed, including the Random-based WOA models (WOARB-1 and WOARB-2), which rely on probability-driven randomization, and the Iterative-based WOA (WOAIB), which refines solutions through iterative local search.

WOARB-1, WOARB-2, and WOAIB delivered performance improvements of 11.6%, 11.9%, and 13.6%, respectively, over existing solutions. Experimental results comparing the WOA models with other VND variants demonstrate that WOA achieves superior outcomes and remarkable consistency across all datasets, with IT-PVND emerging as the second-best performer. These findings have proven the potential of WOA algorithms to deliver efficient and reliable solutions in timetabling optimization.

Third, the study demonstrated the advantages of centralized versus decentralized scheduling approaches. A centralized scheduling with standardized timeslots yielded a 29.3% improvement in objective value compared with current decentralized methods. or decentralized configurations, two variants were tested: (a) standardized timeslots without post-scheduled room reallocation, which improved by 12.3%, and (b) minimized timeslots with post-scheduled room reallocation, which improved by 15.8%. Overall, the findings indicate that centralized scheduling delivers stronger performance in exam spread distribution and slot preference allocation while maintaining high efficiency, whereas decentralized approach (b) demonstrates superior space utilization.

Finally, validation against Toronto benchmark problems confirmed the robustness of the proposed WOAIB and IT-PVND. When compared to 20 metaheuristic approaches, including state-of-the-art methods from literature, the algorithms demonstrated competitive performance, achieving top ranks in several instances. Notably, WOAIB secured 3rd place in four instances, while IT-PVND ranked 3rd in one instance, underscoring the effectiveness of the methods across diverse scenarios.

However, this study has two limitations. First, while the proposed algorithms performed competitively, they did not consistently surpass benchmark methods across all instances,

specifically on the largest instance, indicating a need for further refinement. Second, due to limited datasets, the centralized scheduling framework was tested only at the faculty level rather than the full institutional scale. These constraints underscore the need for broader datasets and continued algorithmic development to enhance both robustness and generalizability.

Overall, this research delivers a unified and validated framework for the exam timetabling problem, integrating real-world operational requirements with metaheuristic techniques. The findings not only advance theoretical understanding but also offer practical tools for universities, enabling more efficient, equitable, and scalable scheduling practices.

7.2 Research Contributions

The overall research contributions are outlined as follows:

- i. Conduct a systematic review of high-quality research solution methodologies in the examination timetabling domain over the past decade, which includes analyzing current trends and providing valuable insights into ongoing advancements in the field (see Chapter 2).
- ii. Develop a mathematical model addressing the UNIMAS exam timetabling problem, incorporating both university-level standard constraints and faculty-level constraints that have not previously been documented in scientific literature. The model is designed to closely represent real-world ETP, ensuring greater generality and facilitating comparability in future research efforts (see Chapter 3).
- iii. Implement a VND metaheuristic approach to address the UNIMAS ETP,

integrating both uncapacitated and capacitated variants within a unified framework. The goal is to optimize constraints at both levels while outperforming UNIMAS proprietary software and manual methods. The proposed approach generates high-quality solutions within reasonable computational times, outperforming both proprietary and manual systems. Furthermore, the use of an iterative threshold parameter as an acceptance criterion in IT-PVND is investigated, which helps prevent premature convergence to local optima. When the IT-PVND was validated and tested using real-world data to assess its applicability, it demonstrated superior solution quality to other VND variants, ensuring its robustness and practical usability across diverse educational environments (see Chapter 4).

- iv. Design and develop a discrete WOA algorithm, initially designed for continuous problems and not previously utilized for exam timetabling, for addressing UNIMAS ETP. Experimental results highlight the effectiveness of the proposed approach, which substitutes traditional equations designed for continuous problems with local search methods tailored explicitly for discrete ETP. The discrete WOA algorithm produced high-quality solutions for the UNIMAS ETP, demonstrating improved performance compared to other VND approaches. It enhanced scheduling efficiency and resource utilization, contributing to better educational resource management (see Chapter 5), and produced competitive outcomes in certain instances, underscoring its effectiveness relative to other established methods for the Toronto benchmark problem (see Chapter 6).

- v. Conduct a comprehensive statistical analysis to evaluate the significance of various factors, including neighborhood structures, balancing exam splits, and centralized versus decentralized approaches (see Chapter 4).

These research contributions are summarized under theoretical significance and practical applicability, as follows.

7.2.1 Contributions to Theory

- i. Development of a mathematical model that integrates institution-wide real-world constraints and captures the latest operational requirements under realistic conditions.
- ii. Introduction of hybrid metaheuristics — including IT-PVND and WOA hybrids — that extend methodological insights into multi-neighborhood design and hybridization in optimization.

7.2.2 Contributions to Applications

- i. Delivery of efficient, higher-quality timetabling solutions that outperform existing proprietary software and manual methods, with measurable improvement in room utilization, exam spread, room and area split, and slot preference fulfilment.
- ii. Provision of decision-support insights for a university through a comparative analysis of centralized versus decentralized scheduling approaches, offering clear guidance on trade-offs between efficiency and space utilization.

- iii. Novel application of the WOA to the exam timetabling problem, demonstrating effective results on real-world data and competitive performance in most Toronto benchmark instances, compared with existing methods in the literature.

7.3 Future Work

The research findings highlight several potential avenues for future studies, as outlined below.

The objective function is currently formulated at the first level, while the second level includes faculty constraints only as extended soft constraints due to limited dataset availability. Future research could build more complex bi-level optimization models that incorporate all faculty constraints and accommodate the preferences of various stakeholders, which may involve distinct objective functions for general constraints applied across all faculties at the first level, as established in this study, and dynamic adaptations at the second level. Additionally, dynamic priority mechanisms could enhance adaptability and efficiency by allowing exam agents to adjust their priorities.

Future research could explore the application of dynamic neighborhood probabilities, potentially derived through an online learning mechanism, and extend the proposed approach to other real-world instances or benchmark problems, such as the ITC-2007 exam track and Yeditepe problems. Additionally, the proposed methodology may find applicability in analogous problems within other timetabling or scheduling domains. Furthermore, future investigations may focus on the dynamic variations in examination timetabling, including fluctuations in exam dates, potential venues, and other relevant factors that may evolve.

The proposed approaches exhibit significant potential for hybridizing WOA with other local search methods, presenting further exploration and refinement opportunities. Moreover, the potential for combining various heuristic or metaheuristic frameworks into a unified system could further enhance solution optimization, providing flexibility and adaptability to diverse constraints and conditions. Future research could investigate applying more recent and promising metaheuristic techniques, exploring innovative approaches to address the problem and expanding the range of potential solutions within the educational timetabling domain.

REFERENCES

- Abdullah, S., Ahmadi, S., Burke, E. K., Dror, M., & McCollum, B. (2007). A tabu-based large neighborhood search methodology for the capacitated examination timetabling problem. *Journal of the Operational Research Society*, 58(11), 1494-1502. <https://doi.org/10.1057/palgrave.jors.2602258>
- Abdullah, S., & Alzaqebah, M. (2013). A hybrid self-adaptive bees algorithm for examination timetabling problems. *Applied Soft Computing*, 13(8), 3608-3620. <https://doi.org/10.1016/j.asoc.2013.04.010>
- Abdel-Basset, M., El-Shahat, D., & Sangaiah, A. K. (2019). A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem. *International Journal of Machine Learning and Cybernetics*, 10, 495-514. <https://doi.org/10.1007/s13042-017-0731-3>
- Abed, M. H., & Tang, A. Y. (2013). Hybridizing genetic algorithm and record-to-record travel algorithm for solving uncapacitated examination timetabling problem. *Electronic Journal of Computer Science and Information Technology*, 4(1).
- Abou Kasm, O., Mohandes, B., Diabat, A., & El Khatib, S. (2019). Exam timetabling with allowable conflicts within a time window. *Computers & Industrial Engineering*, 127, 263-273. <https://doi.org/10.1016/j.cie.2018.11.037>
- Ahandani, M. A., Baghmisheh, M. T. V., Zadeh, M. A. B., & Ghaemi, S. (2012). Hybrid particle swarm optimization transplanted into a hyper-heuristic structure for solving examination timetabling problem. *Swarm and Evolutionary Computation*, 7, 21-34. <https://doi.org/10.1016/j.swevo.2012.06.004>

- Al-Betar, M. A., Khader, A. T., & Doush, I. A. (2014). Memetic techniques for examination timetabling. *Annals of Operations Research*, 218, 23-50.
<https://doi.org/10.1007/s10479-013-1500-7>
- Al-Betar, M. A. (2021). A β -hill climbing optimizer for examination timetabling problem. *Journal of Ambient Intelligence and Humanized Computing*, 12(1), 653-666.
<https://doi.org/10.1007/s12652-020-02047-2>
- Aldeeb, B. A., Al-Betar, M. A., Norwawi, N. M., Alissa, K. A., Alsmadi, M. K., Hazaymeh, A. A., & Alzaqebah, M. (2022). Hybrid intelligent water Drops algorithm for examination timetabling problem. *Journal of King Saud University-Computer and Information Sciences*, 34(8), 4847-4859.
<https://doi.org/10.1016/j.jksuci.2021.06.016>
- Al-Hawari, F., Al-Ashi, M., Abawi, F., & Alounch, S. (2020). A practical three-phase ILP approach for solving the examination timetabling problem. *International Transactions in Operational Research*, 27(2), 924-944.
<https://doi.org/10.1111/itor.12471>
- Alkallak, I. N., Shaban, R. Z., & Alnema, Y. H. S. (2022). Intelligent university timetable scheduling system using sudoku grid with magic square. *Bulletin of Electrical Engineering and Informatics*, 11(3), 1526-1534.
<https://doi.org/10.11591/eei.v11i3.3694>
- Alzaqebah, M., & Abdullah, S. (2011). Hybrid artificial bee colony search algorithm based on disruptive selection for examination timetabling problems. In *International Conference on Combinatorial Optimization and Applications* (pp. 31-45). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-22616-8_3

- Alzaqebah, M., & Abdullah, S. (2015). Hybrid bee colony optimization for examination timetabling problems. *Computers & Operations Research*, 54, 142-154. <https://doi.org/10.1016/j.cor.2014.09.005>
- Amaral, P., & Pais, T. C. (2016). Compromise ratio with weighting functions in a Tabu Search multi-criteria approach to examination timetabling. *Computers & Operations Research*, 72, 160-174. <https://doi.org/10.1016/j.cor.2016.02.012>
- Arbaoui, T., Boufflet, J.-P., & Moukrim, A. (2015). Preprocessing and an improved MIP model for examination timetabling. *Annals of Operations Research*, 229(1), 19-40. <https://doi.org/10.1007/s10479-015-1832-6>
- Arbaoui, T., Boufflet, J.-P., & Moukrim, A. (2019). Lower bounds and compact mathematical formulations for spacing soft constraints for university examination timetabling problems. *Computers & Operations Research*, 106, 133-142. <https://doi.org/10.1016/j.cor.2019.02.013>
- Austero, L. D., Sison, A. M., Matias, J. B., & Medina, R. P. (2022, May). Solving course timetabling problem using Whale Optimization Algorithm. In *2022 8th International Conference on Information Technology Trends (ITT)* (pp. 160-166). IEEE. <https://doi.org/10.1109/ITT56123.2022.9863951>
- Ayob, M., Burke, E. K., & Kendall, G. (2006). An iterative re-start variable neighbourhood search for the examination timetabling problem. *Practice and Theory of Automated Timetabling*. LNCS, 1153, 336-344.
- Ayob, M., Abdullah, S., & Malik, A. (2007). A practical examination timetabling problem at the Universiti Kebangsaan Malaysia. *International Journal of Computer Science and Network Security*, 7(9), 198-204.

- Azad, S. K., & Hasançebi, O. G. U. Z. H. A. N. (2014). An elitist self-adaptive step-size search for structural design optimization. *Applied Soft Computing*, 19, 226-235. <https://doi.org/10.1016/j.asoc.2014.02.017>
- Bassimir, B., & Wanka, R. (2024). On the computation of robust examination timetables: methods and experimental results. *Journal of Scheduling*, 1-23. <https://doi.org/10.1007/s10951-024-00815-y>
- Battistutta, M., Schaerf, A., & Urli, T. (2017). Feature-based tuning of single-stage simulated annealing for examination timetabling. *Annals of Operations Research*, 252, 239-254. <https://doi.org/10.1007/s10479-015-2061-8>
- Bashab, A., Ibrahim, A. O., Abed, Elgabar, E. E., Ismail, M. A., Elsafi, A., Ahmed, A., & Abraham, A. (2020). A systematic mapping study on solving university timetabling problems using meta-heuristic algorithms. *Neural Computing and Applications*, 32, 17397-17432. <https://doi.org/10.1007/s00521-020-05110-3>
- Bazari, S., Pooya, A., Soleimani Fard, O., & Roozkhosh, P. (2023). Modeling and solving the problem of scheduling university exams in terms of new constraints on the conflicts of professors' exams and the concurrence of exams with common questions. *Opsearch*, 60(2), 877-915. <https://doi.org/10.1007/s12597-023-00638-z>
- Bellio, R., Ceschia, S., Di Gaspero, L., & Schaerf, A. (2021). Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling. *Computers & Operations Research*, 132, 105300. <https://doi.org/10.1016/j.cor.2021.105300>
- Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)*, 35(3), 268-308. <https://doi.org/10.1145/937503.937505>

- Blum, C., Puchinger, J., Raidl, G. R., & Roli, A. (2011). Hybrid metaheuristics in combinatorial optimization: A survey. *Applied Soft Computing*, 11(6), 4135-4151. <https://doi.org/10.1016/j.asoc.2011.02.032>
- Bolaji, A. L. a., Khader, A. T., Al-Betar, M. A., Awadallah, M., & Thomas, J. J. (2012). The effect of neighborhood structures on examination timetabling with artificial bee colony. *9th International Conference on the Practice and Theories of Automated Timetabling (PATAT 2012)*.
- Bolaji, A. L. a., Khader, A. T., Al-Betar, M. A., & Awadallah, M. A. (2015). A hybrid nature-inspired artificial bee colony algorithm for uncapacitated examination timetabling problems. *Journal of Intelligent Systems*, 24(1), 37-54. <https://doi.org/10.1515/jisys-2014-0002>
- Borchani, R., Elloumi, A., & Masmoudi, M. (2017). Variable neighborhood descent search based algorithms for course timetabling problem: Application to a Tunisian University. *Electronic Notes in Discrete Mathematics*, 58, 119-126. <https://doi.org/10.1016/j.endm.2017.03.016>
- Burke, E., Bykov, Y., Newall, J., & Petrovic, S. (2004). A time-predefined local search approach to exam timetabling problems. *IIE Transactions*, 36(6), 509-528. <https://doi.org/10.1080/07408170490438410>
- Burke, E. K., Eckersley, A. J., McCollum, B., Petrovic, S., & Qu, R. (2010). Hybrid variable neighbourhood approaches to university exam timetabling. *European Journal of Operational Research*, 206(1), 46-53. <https://doi.org/10.1016/j.ejor.2010.01.044>
- Burke, E., Elliman, D., Ford, P., & Weare, R. (1996). Examination timetabling in British universities: A survey. *Practice and Theory of Automated Timetabling: First*

International Conference Edinburgh, UK, August 29–September 1, 1995 Selected Papers 1.

- Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., & Qu, R. (2013). Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, 64(12), 1695-1724. <https://doi.org/10.1057/jors.2013.71>
- Burke, E. K., Qu, R., & Soghier, A. (2014). Adaptive selection of heuristics for improving exam timetables. *Annals of Operations Research*, 218, 129-145. <https://doi.org/10.1007/s10479-012-1140-3>
- Burke, E. K., & Bykov, Y. (2016). An adaptive flex-deluge approach to university exam timetabling. *INFORMS Journal on Computing*, 28(4), 781-794. <https://doi.org/10.1287/ijoc.2015.0680>
- Burke, E. K., & Bykov, Y. (2017). The late acceptance hill-climbing heuristic. *European Journal of Operational Research*, 258(1), 70-78. <https://doi.org/10.1016/j.ejor.2016.07.012>
- Bykov, Y., & Petrovic, S. (2016). A step counting hill climbing algorithm applied to university examination timetabling. *Journal of Scheduling*, 19, 479-492. <https://doi.org/10.1007/s10951-016-0469-x>
- Çalık, H., Wauters, T., & Vanden Berghe, G. (2024). The exam location problem: mathematical formulations and variants. *Computers & Operations Research*, 161, 106438. <https://doi.org/10.1016/j.cor.2023.106438>
- Carlsson, M., Ceschia, S., Di Gaspero, L., Mikkelsen, R. Ø., Schaerf, A., & Stidsen, T. J. R. (2023). Exact and metaheuristic methods for a real-world examination timetabling problem. *Journal of Scheduling*, 26(4), 353-367. <https://doi.org/10.1007/s10951-023-00778-6>

- Carter, M. W. (1986). OR practice—a survey of practical applications of examination timetabling algorithms. *Operations Research*, 34(2), 193-202. <https://doi.org/10.1287/opre.34.2.193>
- Carter, M.W., Laporte, G. (1996). Recent developments in practical examination timetabling. In Burke, E., Ross, P. (eds) *Practice and Theory of Automated Timetabling. PATAT 1995*. Lecture Notes in Computer Science, vol 1153. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-61794-9_49
- Carter, M. W., Laporte, G., & Lee, S. Y. (1996). Examination timetabling: Algorithmic strategies and applications. *Journal of the Operational Research Society*, 47(3), 373-383. <https://doi.org/10.1057/jors.1996.37>
- Cataldo, A., Ferrer, J.-C., Miranda, J., Rey, P. A., & Sauré, A. (2017). An integer programming approach to curriculum-based examination timetabling. *Annals of Operations Research*, 258, 369-393. <https://doi.org/10.1007/s10479-016-2321-2>
- Cavdur, F., & Kose, M. (2016). A fuzzy logic and binary-goal programming-based approach for solving the exam timetabling problem to create a balanced-exam schedule. *International Journal of Fuzzy Systems*, 18, 119-129. <https://doi.org/10.1007/s40815-015-0046-z>
- Ceschia, S., Di Gaspero, L., & Schaerf, A. (2023). Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research*, 308(1), 1-18. <https://doi.org/10.1016/j.ejor.2022.07.011>
- Chanthadavong, A. (2019). *Digital exams to help Monash University save AU\$7 million*. [online] Available at: <https://www.zdnet.com/article/digital-exams-to-help-monash-university-save-au7-million/> [Accessed on 8 September 2025]

- Cheng, M. Y., & Prayogo, D. (2014). Symbiotic organisms search: a new metaheuristic optimization algorithm. *Computers & Structures*, 139, 98-112. <https://doi.org/10.1016/j.compstruc.2014.03.007>
- Chen, M. C., Goh, S. L., & Lau, S. P. (2021). A Hybrid of Heuristic Orderings and Variable Neighbourhood Descent for a Real Life University Course Timetabling Problem. *International Journal of Systematic Innovation*, 6(5), 1-10. [https://doi.org/10.6977.IJoSI.202109_6\(5\).0001](https://doi.org/10.6977.IJoSI.202109_6(5).0001)
- Çimen, M., Belbağ, S., Soysal, M., & Sel, Ç. (2022). Invigilators assignment in practical examination timetabling problems. *International Journal of Industrial Engineering*, 29(3). <https://doi.org/10.23055/ijietap.2022.29.3.6943>
- Cooper, T. B., & Kingston, J. H. (1996). The complexity of timetable construction problems. *Practice and Theory of Automated Timetabling: First International Conference Edinburgh, UK, August 29–September 1, 1995 Selected Papers* 1.
- Di Gaspero, L. (2002). Recolour, shake and kick: A recipe for the examination timetabling problem. In *Proceedings of the Fourth International Conference on the Practice and Theory of Automated Timetabling*, Gent, Belgium.
- Di Gaspero, L., & Schaerf, A. (2001). Tabu search techniques for examination timetabling. *Practice and Theory of Automated Timetabling III: Third International Conference, PATAT 2000 Konstanz, Germany, August 16–18, 2000 Selected Papers* 3.
- Ding, J., Shen, L., Lü, Z., & Peng, B. (2019). Parallel machine scheduling with completion-time-based criteria and sequence-dependent deterioration. *Computers & Operations Research*, 103, 35-45. <https://doi.org/10.1016/j.cor.2018.10.016>
- Duarte, A., Sánchez-Oro, J., Mladenović, N., Todosijević, R. (2018). Variable Neighborhood Descent. In: Martí, R., Pardalos, P., Resende, M. (Eds.), *Handbook of*

- Heuristics*. Springer International Publishing. https://doi.org/10.1007/978-3-319-07124-4_9
- Dueck, G. (1993). New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, 104(1), 86-92. <https://doi.org/10.1006/jcph.1993.1010>
- Eley, M. (2006). Ant algorithms for the exam timetabling problem. *International Conference on the Practice and Theory of Automated Timetabling*.
- Fesanghary, M., Damangir, E., & Soleimani, I. (2009). Design optimization of shell and tube heat exchangers using global sensitivity analysis and harmony search algorithm. *Applied Thermal Engineering*, 29(5-6), 1026-1031. <https://doi.org/10.1016/j.applthermaleng.2008.05.018>
- Fong, C. W., Asmuni, H., & McCollum, B. (2015). A hybrid swarm-based approach to university timetabling. *IEEE Transactions on Evolutionary Computation*, 19(6), 870-884. <https://doi.org/10.1109/TEVC.2015.2411741>
- Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey: Whale Optimization Algorithm and its applications. *Swarm and Evolutionary Computation*, 48, 1-24. <https://doi.org/10.1016/j.swevo.2019.03.004>
- Godwin, T. (2022). Obtaining quality business school examination timetable under heterogeneous elective selections through surrogacy. *Opsearch*, 59(3), 1055-1093. <https://doi.org/10.1007/s12597-022-00590-4>
- Gogos, C., Goulas, G., Alefragis, P., Kolonias, V., & Housos, E. (2010). Distributed scatter search for the examination timetabling problem. *PATAT 2010*, 211

- González-Velarde, J. L., & Laguna, M. (2002). Tabu search with simple ejection chains for coloring graphs. *Annals of Operations Research*, 117, 165-174.
<https://doi.org/10.1023/A:1021573507189>
- Gorgani, H. H., & Shabani, S. (2021). Online exams and the COVID-19 pandemic: a hybrid modified FMEA, QFD, and k-means approach to enhance fairness. *SN applied sciences*, 3(10), 818. <https://doi.org/10.1007/s42452-021-04805-z>
- Güler, M. G., & Gecici, E. (2020). A spreadsheet-based decision support system for examination timetabling. *Turkish Journal of Electrical Engineering and Computer Sciences*, 28(3), 1584-1598. <https://doi.org/10.3906/elk-1909-14>
- Güler, M. G., Geçici, E., Köroğlu, T., & Becit, E. (2021). A web-based decision support system for examination timetabling. *Expert Systems with Applications*, 183, 115363. <https://doi.org/10.1016/j.eswa.2021.115363>
- Hansen, P., Mladenovic, N., Brimberg, J., & Pérez, J. M. (2003). Variable neighborhood search. *Handbook of Metaheuristics*. Kluwer, Dordrecht, The Netherlands.
- Hansen, P., Mladenović, N., Todosijević, R., & Hanafi, S. (2017). Variable neighborhood search: basics and variants. *EURO Journal on Computational Optimization*, 5(3), 423-454. <https://doi.org/10.1007/s13675-016-0075-x>
- Hao, X., Qu, R., & Liu, J. (2021). A unified framework of graph-based evolutionary multitasking hyper-heuristic. *IEEE Transactions on Evolutionary Computation*, 25(1), 35-47. <https://doi.org/10.1109/TEVC.2020.2991717>
- UpSkillWise. (2025). Online learning statistics. UpSkillWise.
<https://upskillwise.com/online-learning-statistics/>

- Hussain, K., Mohd Salleh, M. N., Cheng, S., & Shi, Y. (2019). Metaheuristic research: a comprehensive survey. *Artificial Intelligence Review*, 52(4), 2191-2233. <https://doi.org/10.1007/s10462-017-9605-z>
- Jain, R. K., Chiu, D.-M. W., & Hawe, W. R. (1984). A quantitative measure of fairness and discrimination for resource allocation in shared computer system. Eastern Research Laboratory, *Digital Equipment Corporation, Hudson, MA*, 21(1), 2022-2023.
- Kahar, M. N. M., & Kendall, G. (2010). The examination timetabling problem at Universiti Malaysia Pahang: Comparison of a constructive heuristic with an existing software solution. *European Journal of Operational Research*, 207(2), 557-565. <https://doi.org/10.1016/j.ejor.2010.04.011>
- Kahar, M., & Kendall, G. (2015). A great deluge algorithm for a real-world examination timetabling problem. *Journal of the Operational Research Society*, 66(1), 116-133. <https://doi.org/10.1057/jors.2012.169>
- Karp, R.M. (1972). Reducibility among Combinatorial Problems. In Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds) *Complexity of Computer Computations*. The IBM Research Symposia Series. Springer, Boston, MA. https://doi.org/10.1007/978-1-4684-2001-2_9
- Keskin, M. E., Döyen, A., Akyer, H., & Güler, M. G. (2018). Examination timetabling problem with scarce resources: a case study. *European Journal of Industrial Engineering*, 12(6), 855-874. <https://doi.org/10.1504/EJIE.2018.10016933>
- Laporte, G., & Desroches, S. (1984). Examination timetabling by computer. *Computers & Operations Research*, 11(4), 351-360. [https://doi.org/10.1016/0305-0548\(84\)90036-](https://doi.org/10.1016/0305-0548(84)90036-4)

- Lawler, E. L. (2001). Combinatorial optimization: networks and matroids. Courier Corporation.
- Lei, Y., Gong, M., Jiao, L., Li, W., Zuo, Y., & Cai, Q. (2014). A double evolutionary pool memetic algorithm for examination timetabling problems. *Mathematical Problems in Engineering*, 2014(1), 867645. <https://doi.org/10.1155/2014/867645>
- Lei, Y., Gong, M., Jiao, L., Shi, J., & Zhou, Y. (2017). An adaptive coevolutionary memetic algorithm for examination timetabling problems. *International Journal of Bio-Inspired Computation*, 10(4), 248-257. <https://doi.org/10.1504/IJBIC.2017.087918>
- Lei, Y., Shi, J., & Yan, Z. (2018). A memetic algorithm based on MOEA/D for the examination timetabling problem. *Soft Computing*, 22, 1511-1523. <https://doi.org/10.1007/s00500-017-2886-y>
- Leite, N., Fernandes, C. M., Melicio, F., & Rosa, A. C. (2018). A cellular memetic algorithm for the examination timetabling problem. *Computers & Operations Research*, 94, 118-138. <https://doi.org/10.1016/j.cor.2018.02.009>
- Leite, N., Melicio, F., & Rosa, A. C. (2019). A fast simulated annealing algorithm for the examination timetabling problem. *Expert Systems with Applications*, 122, 137-151. <https://doi.org/10.1016/j.eswa.2018.12.048>
- Li, J., Bai, R., Shen, Y., & Qu, R. (2015). Search with evolutionary ruin and stochastic rebuild: A theoretic framework and a case study on exam timetabling. *European Journal of Operational Research*, 242(3), 798-806. <https://doi.org/10.1016/j.ejor.2014.11.002>
- Li, Ya, He, Y., Liu, X., Guo, X., & Li, Z. (2020). A novel discrete whale optimization algorithm for solving knapsack problems. *Applied Intelligence*, 50(10), 3350-3366. <https://doi.org/10.1007/s10489-020-01722-3>

- Mahmood, S., Bawany, N. Z., & Tanweer, M. R. (2023). A comprehensive survey of whale optimization algorithm: modifications and classification. *Indonesian Journal of Electrical Engineering and Computer Science*, 29(2), 899. <https://doi.org/10.11591/ijeecs.v29.i2.pp899-910>
- Mandal, A. K., Kahar, M. N. M., & Kendall, G. (2020). Addressing examination timetabling problem using a partial exams approach in constructive and improvement. *Computation*, 8(2), 46. <https://doi.org/10.3390/computation8020046>
- Martí, R., Pardalos, P. M., & Resende, M. G. (2018). Handbook of heuristics. *Springer Publishing Company, Incorporated*. <https://doi.org/10.1007/978-3-319-07124-4>
- McCollum, B., McMullan, P., Burke, E. K., Parkes, A. J., & Qu, R. (2007). *The second international timetabling competition: Examination timetabling track*. Technical Report QUB/IEEE/Tech/ITC2007/-Exam/v4. 0/17, Queen's University, Belfast.
- McCollum, B., Schaerf, A., Paechter, B., McMullan, P., Lewis, R., Parkes, A. J., Gaspero, L. D., Qu, R., & Burke, E. K. (2010). Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing*, 22(1), 120-130. <https://doi.org/10.1287/ijoc.1090.0320>
- Merlot, L. T., Boland, N., Hughes, B. D., & Stuckey, P. J. (2003). A hybrid algorithm for the examination timetabling problem. *Practice and Theory of Automated Timetabling IV: 4th International Conference, PATAT 2002*, Gent, Belgium, August 21-23, 2002. Selected Revised Papers 4.
- Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in engineering software*, 95, 51-67. <https://doi.org/10.1016/j.advengsoft.2016.01.008>

- Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249. <https://doi.org/10.1016/j.knosys.2015.07.006>
- Mirjalili, S. (2016). SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-based systems*, 96, 120-133. <https://doi.org/10.1016/j.knosys.2015.12.022>
- Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11), 1097-1100. [https://doi.org/10.1016/S0305-0548\(97\)00031-2](https://doi.org/10.1016/S0305-0548(97)00031-2)
- Mladenović, N., Sifaleras, A., & Sörensen, K. (2017). Editorial to the Special Cluster on Variable Neighborhood Search, Variants and Recent Applications. *International Transactions in Operational Research*, 24(3). <https://doi.org/10.1111/itor.12369>
- Modirghorasani, A., & Hoseinpour, P. (2024). Decentralized exam timetabling: A solution for conducting exams during pandemics. *Socio-Economic Planning Sciences*, 92, 101802. <https://doi.org/10.1016/j.seps.2024.101802>
- Muklason, A., Parkes, A. J., Özcan, E., McCollum, B., & McMullan, P. (2017). Fairness in examination timetabling: Student preferences and extended formulations. *Applied Soft Computing*, 55, 302-318. <https://doi.org/10.1016/j.asoc.2017.01.026>
- Müller, T. (2016). Real-life examination timetabling. *Journal of Scheduling*, 19, 257-270. <https://doi.org/10.1007/s10951-014-0391-z>
- Mweshi, G., & Pillay, N. (2021). An improved grammatical evolution approach for generating perturbative heuristics to solve combinatorial optimization problems. *Expert Systems with Applications*, 165, 113853. <https://doi.org/10.1016/j.eswa.2020.113853>

- Nand, R., Sharma, B., & Chaudhary, K. (2022). An introduction of preference based stepping ahead firefly algorithm for the uncapacitated examination timetabling. *PeerJ Computer Science*, 8, e1068. <https://doi.org/10.7717/peerj-cs.1068>
- Nand, R., Reddy, E., Chaudhary, K., & Sharma, B. (2024). Preference-Based Stepping Ahead Firefly Algorithm for Solving Real-World Uncapacitated Examination Timetabling Problem. *IEEE Access*, 12, 24685-24699. <https://doi.org/10.1109/ACCESS.2024.3365734>
- Ozcan, E., & Ersoy, E. (2005). Final exam scheduler-FES. *2005 IEEE Congress On Evolutionary Computation*.
- Ozturk, Z. K., Gundogan, H. S., Mumyalmaz, E., & Kececiloglu, T. (2024). Exam scheduling under pandemic conditions: A mathematical model and decision support system. *Technological Forecasting and Social Change*, 208, 123687. <https://doi.org/10.1016/j.techfore.2024.123687>
- Parkes, A. J., & Ozcan, E. (2010). Properties of Yeditepe examination timetabling benchmark instances. In *Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling*.
- Petrovic, S., & Burke, E. K. (2004). University Timetabling.
- Pillay, N., & Banzhaf, W. (2009). A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. *European Journal of Operational Research*, 197(2), 482-491. <https://doi.org/10.1016/j.ejor.2008.07.023>
- Pillay, N., & Banzhaf, W. (2010). An informed genetic algorithm for the examination timetabling problem. *Applied Soft Computing*, 10(2), 457-467. <https://doi.org/10.1016/j.asoc.2009.08.011>

- Pillay, N., & Özcan, E. (2019). Automated generation of constructive ordering heuristics for educational timetabling. *Annals of Operations Research*, 275, 181-208. <https://doi.org/10.1007/s10479-017-2625-x>
- Pokhrel S, Chhetri R (2021) A literature review on impact of COVID-19 pandemic on teaching and learning. *Higher education for the future*, 8(1), 133-141. <https://doi.org/10.1177/2347631120983481>
- Premananda, I. G. A., Tjahyanto, A., & Muklason, A. (2022). Hybrid whale optimization algorithm for solving timetabling problems of ITC 2019. 2022 *IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)*.
- Qu, R., Burke, E. K., McCollum, B., Merlot, L. T., & Lee, S. Y. (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal Of Scheduling*, 12, 55-89. <https://doi.org/10.1007/s10951-008-0077-5>
- Qu, R., Pham, N., Bai, R., & Kendall, G. (2015). Hybridising heuristics within an estimation distribution algorithm for examination timetabling. *Applied Intelligence*, 42, 679-693. <https://doi.org/10.1007/s10489-014-0615-0>
- Rana, N., Abd Latiff, M. S., Abdulhamid, S. i. M., & Misra, S. (2022). A hybrid whale optimization algorithm with differential evolution optimization for multi-objective virtual machine scheduling in cloud computing. *Engineering Optimization*, 54(12), 1999-2016. <https://doi.org/10.1080/0305215X.2021.1969560>
- Sabar, N. R., Ayob, M., Kendall, G., & Qu, R. (2012). A honey-bee mating optimization algorithm for educational timetabling problems. *European Journal of Operational Research*, 216(3), 533-543. <https://doi.org/10.1016/j.ejor.2011.08.006>
- Sabar, N. R., Ayob, M., Kendall, G., & Qu, R. (2015). A dynamic multiarmed bandit-gene expression programming hyper-heuristic for combinatorial optimization problems.

- IEEE Transactions on Cybernetics*, 45(2), 217-228.
<https://doi.org/10.1109/TCYB.2014.2323936>
- Schaerf, A. (1999). A survey of automated timetabling. *Artificial Intelligence Review*, 13, 87-127. <https://doi.org/10.1023/A:1006576209967>
- Sharma, A. (2020). A Constraint Driven Solution Model for Discrete Domains with a Case Study of Exam Timetabling Problems. *ArXiv*, abs/2002.03102.
<https://doi.org/10.48550/arXiv.2002.03102>
- Siew, E. S. K., Sze, S.N., Goh, S. L., Kendall, G., Sabar, N. R., & Abdullah, S. (2024). A Survey of solution methodologies for exam timetabling problems. *IEEE Access*, 12, 41479-41498
- Soria-Alcaraz, J. A., Ochoa, G., Sotelo-Figueroa, M. A., Carpio, M., & Puga, H. (2017). Iterated VND versus hyper-heuristics: Effective and general approaches to course timetabling. *Nature-Inspired Design of Hybrid Intelligent Systems*, 687-700.
https://doi.org/10.1007/978-3-319-47054-2_45
- Sze, S.-N., Phang, M.-H., & Chiew, K.-L. (2017). Real-life faculty examination timetabling to utilise room used. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 9(3-11), 51-54
- Thompson, J. M., & Dowsland, K. A. (1998). A robust simulated annealing based examination timetabling system. *Computers & Operations Research*, 25(7-8), 637-648. [https://doi.org/10.1016/S0305-0548\(97\)00101-9](https://doi.org/10.1016/S0305-0548(97)00101-9)
- Tilahun, S. L. (2019). Prey-predator algorithm for discrete problems: a case for examination timetabling problem. *Turkish Journal of Electrical Engineering and Computer Sciences*, 27(2), 950-960. <https://doi.org/10.3906/elk-1809-175>

- Turabieh, H., & Abdullah, S. (2011). An integrated hybrid approach to the examination timetabling problem. *Omega*, 39(6), 598-607.
<https://doi.org/10.1016/j.omega.2010.12.005>
- Van Bulck, D., Goossens, D., & Schaerf, A. (2023). Multi-neighborhood simulated annealing for the ITC-2007 capacitated examination timetabling problem. *Journal of Scheduling*, 1-16. <https://doi.org/10.1007/s10951-023-00799-1>
- Vianna, D. S., Martins, C. B., Lima, T. J., Vianna, M. D. F. D., & Meza, E. B. M. (2020). Hybrid VNS-TS heuristics for university course timetabling problem. *Brazilian Journal of Operations & Production Management*, 17(2), 1-20.
<https://doi.org/10.14488/BJOPM.2020.014>
- Woumans, G., De Boeck, L., Beliën, J., & Creemers, S. (2016). A column generation approach for solving the examination-timetabling problem. *European Journal of Operational Research*, 253(1), 178-194. <https://doi.org/10.1016/j.ejor.2016.01.046>
- Wren, A. (1995). Scheduling, Timetabling and Rostering - A Special Relationship? *International Conference on the Practice and Theory of Automated Timetabling*.
- Xu, Z., Yu, Y., Yachi, H., Ji, J., Todo, Y., & Gao, S. (2018). A novel memetic whale optimization algorithm for optimization. In *International Conference on Swarm Intelligence* (pp. 384-396). Cham: Springer International Publishing.
https://doi.org/10.1007/978-3-319-93815-8_37
- Yang, X. S. (2008). *Introduction to mathematical optimization: From linear programming to metaheuristics*. Cambridge International Science Publishing.
- Yang, W., Su, J., Yao, Y., Yang, Z., & Yuan, Y. (2022). A novel hybrid whale optimization algorithm for flexible job-shop scheduling problem. *Machines*, 10(8), 618.
<https://doi.org/10.3390/machines10080618>

- Yu L. Lan, F. Liu, W. W. Y. Ng, J. Zhang and M. Gui. (2020). Decomposition Based Multi-Objective Variable Neighborhood Descent Algorithm for Logistics Dispatching. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5), 826-839. <https://doi.org/10.1109/TETCI.2020.3002228>
- Zhang, J., Hong, L., & Liu, Q. (2020). An improved whale optimization algorithm for the traveling salesman problem. *Symmetry*, 13(1), 48. <https://doi.org/10.3390/sym13010048>
- Zhang, S., & Gu, X. (2023). A discrete whale optimization algorithm for the no-wait flow shop scheduling problem. *Measurement and Control*, 56(9-10), 1764-1779. <https://doi.org/10.1177/00202940231180622>
- Zhu, K., Li, L. D., & Li, M. (2022). Developing an online examination timetabling system using artificial bee colony algorithm in higher education. *Broadband Communications, Networks, and Systems: 12th EAI International Conference, BROADNETS 2021, Virtual Event, October 28–29, 2021, Proceedings* 12.

APPENDICES

Appendix A: Journal Publications

1. **Siew, E. S. K.**, Sze, S. N., Goh, S. L., Kendall, G., Sabar, N. R., & Abdullah, S. (2024). A survey of solution methodologies for exam timetabling problems. *IEEE Access*, 12, 41479-41498. <https://doi.org/10.1109/ACCESS.2024.3378054>
2. **Siew, E. S. K.**, Sze, S. N., Goh, S. L. (2025). Optimizing decentralized exam timetabling with a discrete whale optimization algorithm. *International Journal of Advanced Computer Science and Applications*, 16(1), 257-265. <https://doi.org/10.14569/IJACSA.2025.0160125>
3. **Siew E. S. K.**, Sze S. N., Goh S. L., Hossin M., Chiew K. L. (2025). Multi-neighborhood local search with room split balancer for exam timetabling: A case study. *Mathematical Modeling and Computing*, 12(1), 144–157. <https://doi.org/10.23939/mmc2025.01.144>