



Faculty of Computer Science and Information Technology

***TEACHER-SCHOOL ASSIGNMENTS WITH SUBJECT AND LOCATION
CONSIDERATIONS***

Vincent Choo Jia Xian

Bachelor of Computer Science with Honours

(Computational Science)

2025

***TEACHER-SCHOOL ASSIGNMENTS WITH SUBJECT AND LOCATION
CONSIDERATIONS***

VINCENT CHOO JIA XIAN

This project is submitted in partial fulfilment of
the requirements for the degree of Bachelor of Computer Science with Honours
(Computational Science)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2025

***PENEMPATAN GURU KE SEKOLAH BERDASARKAN PERTIMBANGAN
SUBJEK DAN LOKASI***

VINCENT CHOO JIA XIAN

Projek ini merupakan salah satu keperluan untuk Izajah
Sarjana Muda Sains Komputer dengan Kepujian

Fakulti Sains Komputer dan Teknologi Maklumat

UNIVERSITI MALAYSIA SARAWAK

2025

DECLARATION

I hereby declare that the project is my original work. I have not copied from any other student's work or from any other sources except due to reference or acknowledgement is not made explicitly in the text, nor has any part had been written for me by another person.



(Vincent Choo Jia Xian, 81448)

16th June 2025

Faculty of Computer Science and

Information Technology

Universiti Malaysia Sarawak

ACKNOWLEDGEMENTS

I would like to extend my deepest gratitude to everyone who contributed or assisted to the completion of the Final Year Project. First of all, I would like to acknowledge and thank my supportive supervisor, Dr. Sze San Nah for giving me the opportunity to work under her supervision. Her invaluable advice and encouragement have helped me through the FYP. Secondly, I want to thank my FYP examiner, Dr. Stephanie Chua for her helpful feedback and comments on my FYP. Her valuable feedback has helped me improve the overall quality of my FYP. Additionally, I want to thank Prof. Wang Yin Chai, our FYP coordinator, for his patience in addressing students' questions and providing FYP guidelines. Furthermore, I am appreciative for the assistance provided by my academic institution, Universiti Malaysia Sarawak. I also want to thank all of my lecturers for giving me with the necessary skills as well as knowledge for this FYP. Finally, I want to thank my family and friends for their encouragement, guidance, support, and understanding during my FYP. This report could not have been created without their invaluable efforts.

TABLE OF CONTENTS

DECLARATION	i
ACKNOWLEDGEMENTS	ii
TABLE OF CONTENTS	iii
LIST OF TABLES	vi
LIST OF FIGURES	vii
LIST OF CODE SNIPPETS	ix
ABSTRACT	x
ABSTRAK	xi
CHAPTER 1: INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	1
1.3 Scope	2
1.4 Objectives	3
1.5 Methodology	3
1.6 Significance of Project	5
1.7 Project Schedule	6
1.8 Expected Outcome	9
1.9 Project Outline	10
CHAPTER 2: LITERATURE REVIEW	11
2.1 Introduction	11
2.1.1 Assignment Problem	11
2.2 Review on Existing Allocation Algorithms	12
2.2.1 Greedy Algorithm	12
2.2.2 Priority Rule-Based Algorithm	13
2.2.3 Ultimate Hungarian Algorithm	14
2.3 Review on Similar Existing Systems	14
2.3.1 Excel Solver	15
2.3.2 OpenSolver	19
2.3.3 AtoZmath.com	23
2.4 Comparison of the Existing System	27
2.5 Evaluation Metrics for Teacher-School Allocation	28
2.5.1 F1-Score	28
2.5.2 Preferences Satisfaction Rate	29
2.6 Conclusion	29

CHAPTER 3: REQUIREMENT ANALYSIS AND DESIGN.....	31
3.1 Introduction	31
3.2 Problem Description.....	31
3.2.1 Teachers.....	31
3.2.2 Schools.....	32
3.3 Modelling the Teacher-School Allocation Problem.....	34
3.3.1 Constraints.....	34
3.3.2 Objective Function	34
3.4 Methodology.....	34
3.4.1 Algorithm Design	35
3.4.2 Implementation	38
3.5 Possible Issues	39
3.6 Conclusion	39
CHAPTER 4: IMPLEMENTATION AND TESTING.....	40
4.1 Introduction	40
4.2 System Requirements and Specification.....	40
4.2.1 Software Modules.....	40
4.2.2 Hardware Requirement.....	41
4.3 Input Data	41
4.3.1 Teacher Data	41
4.3.2 School Data	42
4.3.3 Subject Data.....	43
4.4 Output Result.....	44
4.5 Processes of Coding Algorithm.....	45
4.6 Conclusion	53
CHAPTER 5: EVALUATION	54
5.1 Introduction	54
5.2 Simulated Data.....	55
5.3 Results.....	56
5.3.1 Scenario 1.....	56
5.3.2 Scenario 2.....	59
5.3.3 Scenario 3.....	61
5.4 Discussion	66
5.5 Conclusion	67
CHAPTER 6: CONCLUSION AND FUTURE WORK.....	68
6.1 Introduction	68
6.2 Conclusion	68

6.3	Future Works	68
	References	70

LIST OF TABLES

Table 2.1: Functions in Excel Solver.	17
Table 2.2: Functions in OpenSolver.	21
Table 2.3: Functions in AtoZmath.com.	26
Table 2.4: Comparison of the Features of Three Existing Systems.	27
Table 4.1: Software Requirements.	40
Table 4.2: Hardware Requirements.	41
Table 4.3: Table of Teacher Data	42
Table 4.4: Table of School Data	43
Table 4.5: Table of Subject Data	44
Table 4.6: Table of Assignment.	45
Table 5.1: Assignment Effectiveness Across Datasets for Scenario 1.	56
Table 5.2: Assignment Effectiveness Across Datasets for Scenario 2	59
Table 5.3: Assignment Effectiveness Across Datasets for Scenario 3 with 20 subjects	62
Table 5.4: Assignment Effectiveness Across Datasets for Scenario 3 with 30 subjects	64

LIST OF FIGURES

Figure 1.1: Waterfall Methodology Stages from Waseem (2023).	4
Figure 1.2: The project schedule of FYP 1.	6
Figure 1.3: Gantt Chart of FYP 1.....	7
Figure 1.4: Gantt Chart of FYP 1.....	7
Figure 1.5: Project Schedule of FYP 2	8
Figure 1.6: Gantt Chart of FYP 2.....	8
Figure 2.1: Overview of Excel Solver	15
Figure 2.2: 3 Options of Solving Method	16
Figure 2.3: Solver Results.....	16
Figure 2.4: Add Constraints	17
Figure 2.5: Overview of OpenSolver.....	19
Figure 2.6: AutoModel in OpenSolver.....	20
Figure 2.7: Solver Engine in OpenSolver	20
Figure 2.8: OpenSolver's button and menu in Excel's Data Ribbon	20
Figure 2.9: OpenSolver's menu for more advanced options	21
Figure 2.10: Dashboard of AtoZmath.com	24
Figure 2.11: Input Table	24
Figure 2.12: Examples of Operation Research Calculators	25
Figure 2.13: Crew Assignment Problem Calculator	25
Figure 2.14: Step by step solution.....	25
Figure 3.1: Flowchart of the Proposed Algorithm	37
Figure 3.2: Allocation Results.....	38
Figure 4.1: Input Teacher Data	42
Figure 4.2: Input School Data.....	43
Figure 4.3: Input Subject Data	43
Figure 4.4: Interface for Assignment	44
Figure 4.5: Assignment Analysis	45
Figure 4.6: Sorted teacher list	45
Figure 4.7: Sorted school list	46
Figure 4.8: Assignment by high-priority subject	48
Figure 4.9: Assignment by low-priority subject	49
Figure 4.10: Assignment by same-category subject	50

Figure 5.1: Teacher Data Interface.....	54
Figure 5.2: Subject Data Interface	54
Figure 5.3: School Data Interface	55

LIST OF CODE SNIPPETS

Code Snippet 4.1: Code to retrieve teacher data.....	46
Code Snippet 4.2: Code for sort teacher	46
Code Snippet 4.3: Code for sort school	47
Code Snippet 4.4: Code for calling High-Priority Subjects function	48
Code Snippet 4.5: Code for High-Priority Subjects function	48
Code Snippet 4.6: Code for calling Low-Priority Subjects function	49
Code Snippet 4.7: Code for calling Same-Category Subjects function	50
Code Snippet 4.8: Code for create unallocated record	50
Code Snippet 4.9: Code for Haversine formula	50
Code Snippet 4.10: Code for create assignment records	51
Code Snippet 4.11: Code for finding matched schools.....	51
Code Snippet 4.12: Code for finding nearest schools.....	52
Code Snippet 4.13: Code for finding nearest schools.....	52

ABSTRACT

Equitable distribution of teachers to schools is a significant challenge in education systems particularly in rural and underprivileged areas which shortage of teachers require a balance teacher preferences and school demands. This project proposes a Teacher-School Assignment Algorithm that optimizes teacher assignments according to subject and geographical preferences. The algorithm applies greedy approach to assign teachers to schools by prioritizing exact matches before using the category-based fallback mechanism. The effectiveness of the assignment algorithm was evaluated through simulated scenarios with different number of teachers, subjects and location preferences constraints. The assignment results generated by the algorithm show high assignment rates with high subject preference matches and moderate location matches among the scenarios. Besides, minimal school with no assigned teachers shows that the teachers were evenly distributed across the schools.

ABSTRAK

Pengagihan guru secara saksama ke sekolah-sekolah adalah cabaran yang signifikan dalam sistem pendidikan terutamanya di kawasan luar bandar dan kurang bernasib baik di mana kekurangan guru memerlukan keseimbangan antara keutamaan guru dan permintaan sekolah. Projek ini mencadangkan Algoritma Penugasan Guru-Sekolah yang mengoptimumkan penugasan guru mengikut keutamaan subjek dan geografi. Algoritma ini menggunakan pendekatan tamak untuk menugaskan guru ke sekolah dengan mengutamakan padanan tepat sebelum menggunakan mekanisme sandaran berasaskan kategori. Keberkesanan algoritma penugasan telah dinilai melalui senario simulasi dengan bilangan guru, subjek dan kekangan keutamaan lokasi yang berbeza. Keputusan penugasan yang dihasilkan oleh algoritma menunjukkan kadar penugasan yang tinggi dengan padanan keutamaan subjek yang tinggi dan padanan lokasi yang sederhana di antara senario-senario tersebut. Selain itu, sekolah yang minim dengan tiada guru yang ditugaskan menunjukkan bahawa guru-guru tersebut telah diedarkan secara merata di seluruh sekolah.

CHAPTER 1: INTRODUCTION

1.1 Introduction

Education systems around the world face continual issues in ensuring fair access to excellent instruction, particularly in rural and underprivileged areas. The problem of teacher shortages and uneven distribution is widely known nowadays which some schools unable to match their staffing demands with teacher expertise, particularly in specialised subjects like mathematics and sciences. This will not only affect the quality of education at the school but also affect the students in their studies.

A web-based simulation, Teacher Allocation Simulation is developed to help the schools by automating the process of allocating teachers to schools which ensuring that their expertise matches the needs of each school. According to Kraft et al. (2020), utilising computational approaches in resource allocation enhances efficiency while also reducing the possibility of human error, which is useful when facing complicated staffing needs in education. The computational techniques such as Greedy algorithm is used in the system to optimally assign teachers based on their subject specialization and other constraints. The system allows the admin to minimize manual decision making on the assignment of teachers and input the school information and automatically generate optimal allocation that consider different constraints. This system significantly improves and balances the allocation of teacher to schools which enhance the educational outcomes.

1.2 Problem Statement

The efficient allocation of teachers to schools, particularly in rural and underprivileged areas, is an ongoing concern in worldwide education systems. This issue evolved from several causes, including insufficient alignment between teacher qualifications and school staffing requirements which frequently results in an imbalance in the distribution of experienced teachers. According to Luschei and Chudgar (2015), they highlighted that the shortages of

teachers are often caused by inequitable distribution of teachers. Schools in underprivileged or rural regions are particularly affected by these shortages, as educators are typically hesitant to transfer to these places, preferring to stay closer to major cities with greater infrastructure and resources. Besides, the geographical preferences of teachers are another factor that affect the allocation. Many teachers prefer to work closer to their hometowns for reasons such as family ties, insight with the local culture, and better work-life balance. However, this preference causes shortages in regions without a sufficient local teaching workforce and leaving certain schools continuously understaffed and significantly affecting kids' educational experiences in those areas. Furthermore, inefficient manual processes for assigning teachers can also causes the shortages of teachers in certain regions. Traditional method requires extensive paperwork, limited data integration and some consideration of factors like specialisation or regional demand which resulting in inconsistent and often ineffective teacher allocation.

To address these issues, an effective allocation mechanism is required to optimise teacher assignments while taking into account essential aspects such as specialisation, school requirements, and geographic limits (Dillon & Malick, 2020). A systematic and automated approach using a heuristic-based Greedy Algorithm can efficiently streamline the allocation process by associating each teacher with a school based on their expertise and proximity to the school. This approach focusses on making the best possible assignment at each stage. Thus, this approach will improve educational performance across varied regions.

1.3 Scope

The scope of the project is considered by few aspects:

1. Model Constraints

- Teacher Specialization - Match teachers to schools based on subject expertise.
- Geographical Location - Consider the geographical origin of teachers to assign them closer to their preferred areas.

- School Prioritization - Prioritize the schools at the rural areas.
- Teacher Preferences - Take into account teacher preferences to enhance satisfaction and minimize reallocation requests.

2. Focus Area and Data Size

- Region - Target the schools in the underserved and rural areas at Sarawak.
- Simulated Data - The model uses a dataset of 50 teachers, 20 subjects, and 20 schools to simulate real allocation issues.

3. Solution Technique

- Heuristic Algorithm – Greedy Algorithm is used to make optimal and efficient decisions about allocation.

1.4 Objectives

This project aims to develop a teacher allocation web-based system which can optimize the assignment of teachers and the recruitment process for teachers. The objectives are:

- To simulate Teacher-School Assignments with Subject and Location Considerations.
- To develop a Heuristic Algorithm for Teacher-School Assignment.
- To evaluate the accuracy of subject assignment using F1-score and geographic preference satisfaction using preference satisfaction rate.

1.5 Methodology

The methodology used in developing the teacher allocation web-based system is waterfall methodology. The waterfall methodology is structured sequentially like a waterfall's flow where the operation progress for one phase to the next phases without return to the previous phases (Waseem, 2023). This means that each development phase starts only when the previous one is completed (Senarath, 2021). This methodology is particularly effective for short-term tasks like final year projects because of its straightforward and linear approach. The Waterfall approach divides software development into phases with each phase providing input

for the next (Senarath, 2021). Minimal customers involvement in waterfall methodology because of the project will only begin after the customer's requirements are clearly stated (Waseem, 2023). There are five phases in waterfall methodology which are requirements, design, implementation, verification and maintenance.

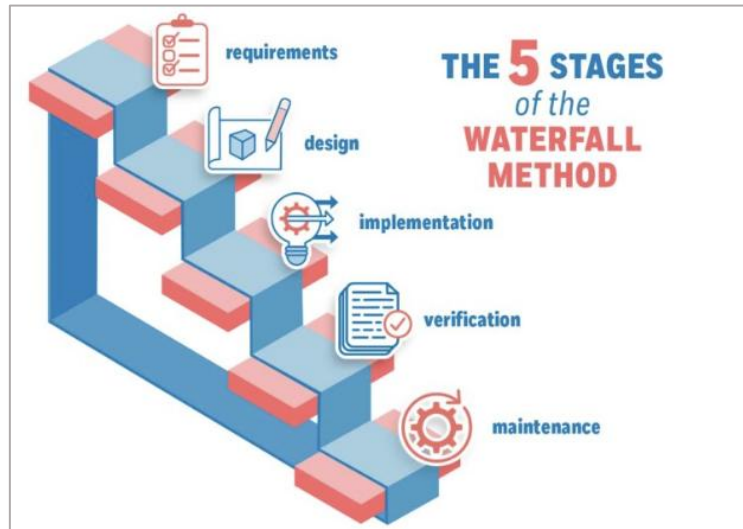


Figure 1.1: Waterfall Methodology Stages from Waseem (2023).

During the requirement analysis phase, the primary goal will be to identify and collect the datasets required for the Teacher Allocation Simulation. This includes obtaining teacher and school data, such as topic specializations, school capacity, and geographic locations, to aid in the allocation process. These datasets will be analysed to ensure that they are comprehensive and relevant to properly addressing the allocation problem. The requirement phase is crucial in the early stages of system development because it develops a firm understand of the project's aims and objectives, providing the groundwork for succeeding phases.

For the design phase, the major goal will be to identify the core issues in the teacher-school allocation process and create an appropriate algorithm to address them. This includes developing a detailed foundation for the allocation algorithm which ensuring that it is consistent with constraints such as subject demand, teacher preferences, and school capacity. The tools used to develop the allocation algorithm and the simulation will also be briefly discussed in this phase.

In the implementation phase involves the actual coding and development the system based on the previous design determined in the design phase. The front-end and back-end of the teacher allocation simulation will be started in this phase. The front-end development will be focus on the user interface of the system while the back-end development will focus on the implementing of the allocation algorithm and the database functionalities of the system. Testing will also be conducted along with the system development to identify and solve any issues.

During the verification phase, the algorithm will undergo extensive testing to ensure that the accuracy of the assignment is as expected. The accuracy of the algorithm must be tested with different scenarios and any issues encountered during the testing will be refined in this phase. Thus, this phase is important to ensure the algorithm is well developed without any errors or issues.

Last in maintenance phase, it involves addressing any problems, delivering updates or improvements based on the user feedback. This phase will focus on system maintenance, improving the system functionalities and ensuing the system meets the user satisfactions. Thus, this phase is important after the system is post-deployment.

1.6 Significance of Project

The implementation of a web-based simulation for teacher allocation has benefited both schools and teachers by resolving significant problems in the equitable allocation of teachers, particularly in underserved and remote areas. The system matches teachers' qualifications and subject expertise to school-specific requirements, eliminating administrative overhead and assuring equitable resource distribution. The overall quality of education can also be improved which ensure that the students in the school receive proper guidance. For teacher, the system takes into account subject specialisation, personal preferences, and geographical limits which resulting in higher job satisfaction and fewer reallocation requests. In short, the development

of the system is important to reduce the teacher shortages and boost the educational equity which will benefit students' educational performance and school operational efficiency.

1.7 Project Schedule

WBS	Task	Start Date	End Date	Duration
1	Final Year Project 1	10-Oct-2024	25-Jan-2025	108d
1.1	FYP Brief Proposal	10-Oct-2024	20-Oct-2024	11d
1.1.1	Identify Project Title	10-Oct-2024	14-Oct-2024	5d
1.1.2	Identify Problem Statement	15-Oct-2024	16-Oct-2024	2d
1.1.3	Define Objectives and Project Outcomes	17-Oct-2024	18-Oct-2024	2d
1.1.4	Identify Basic Skill Requirements	19-Oct-2024	20-Oct-2024	2d
	Milestone 1: Brief Proposal is completed	20-Oct-2024	20-Oct-2024	0
1.2	FYP Feedback and Comment from Reviewer and Examiner	21-Oct-2024	28-Oct-2024	8d
1.2.1	Refine Brief Proposal	21-Oct-2024	28-Oct-2024	8d
	Milestone 2: Brief Proposal is refined	28-Oct-2024	28-Oct-2024	0
1.3	FYP Full Proposal	29-Oct-2024	14-Nov-2024	17d
1.3.1	Refine Problem Statement and Objectives	29-Oct-2024	30-Oct-2024	2d
1.3.2	Elaborate Project Scope	31-Oct-2024	3-Nov-2024	4d
1.3.3	Identify Methodology	4-Nov-2024	9-Nov-2024	6d
1.3.4	Create Project Schedule	10-Nov-2024	14-Nov-2024	5d
	Milestone 3: Full Proposal is completed	14-Nov-2024	14-Nov-2024	0
1.4	FYP Chapter 1	15-Nov-2024	21-Nov-2024	7d
1.4.1	Define Project Report Outline	15-Nov-2024	21-Nov-2024	7d
	Milestone 4: Chapter 1 is completed	21-Nov-2024	21-Nov-2024	0
1.5	FYP Chapter 2	22-Nov-2024	13-Dec-2024	22d
1.5.1	Identify Relevant Literature	22-Nov-2024	25-Nov-2024	4d
1.5.2	Review Literature	26-Nov-2024	1-Dec-2024	6d
1.5.3	Review Software and Hardware	2-Dec-2024	6-Dec-2024	5d
1.5.4	Collect Detailed Requirement	7-Dec-2024	13-Dec-2024	7d
	Milestone 5: Chapter 2 is completed	13-Dec-2024	13-Dec-2024	0
1.6	FYP Chapter 3	14-Dec-2024	5-Jan-2025	23d
1.6.1	Design System Architecture	14-Dec-2024	17-Dec-2024	4d
1.6.2	Design System Diagram	18-Dec-2024	22-Dec-2024	5d
1.6.3	Plan Database Design	23-Dec-2024	27-Dec-2024	5d
1.6.4	Design User Interface	28-Dec-2024	5-Jan-2025	9d
	Milestone 6: Chapter 3 is completed	5-Jan-2025	5-Jan-2025	0
1.7	FYP 1 Final Report	6-Jan-2025	17-Jan-2025	12d
1.7.1	Compilation and Final Check of FYP 1	6-Jan-2025	17-Jan-2025	12d
	Milestone 7: FYP 1 Final Report is submitted	17-Jan-2025	17-Jan-2025	0
1.8	FYP Presentation	24-Jan-2025	25-Jan-2025	2d
	Milestone 8: FYP Presentation is completed	25-Jan-2025	25-Jan-2025	0

Figure 1.2: The project schedule of FYP 1.

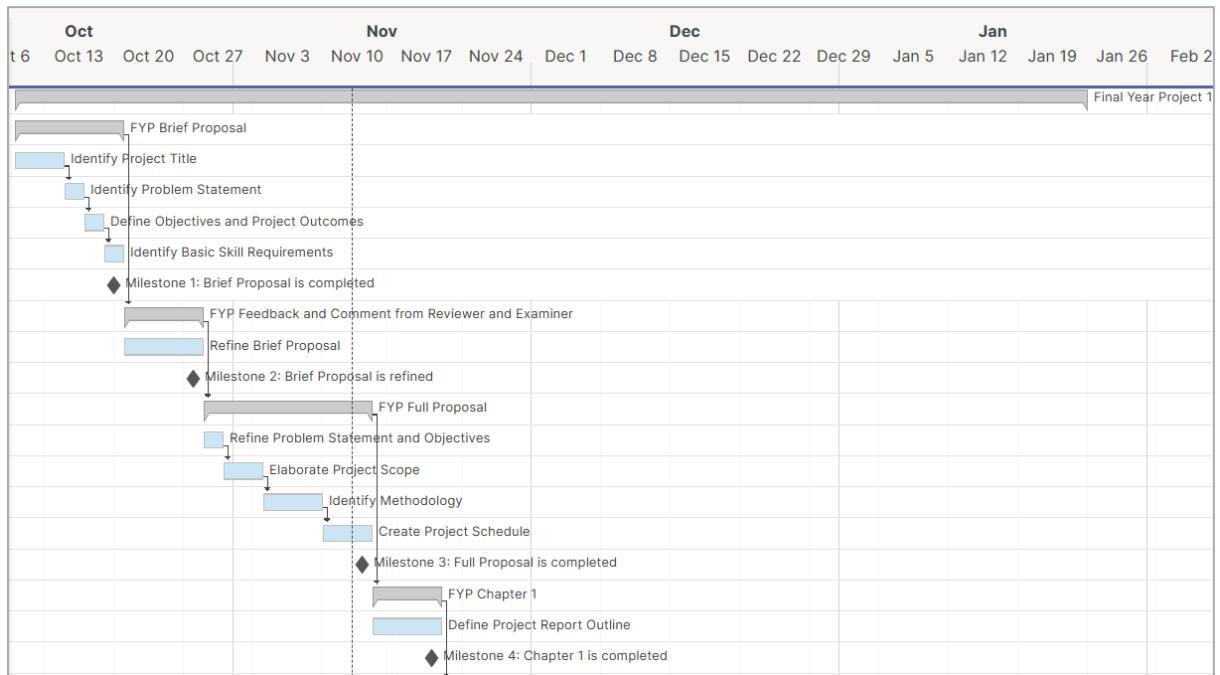


Figure 1.3: Gantt Chart of FYP 1.

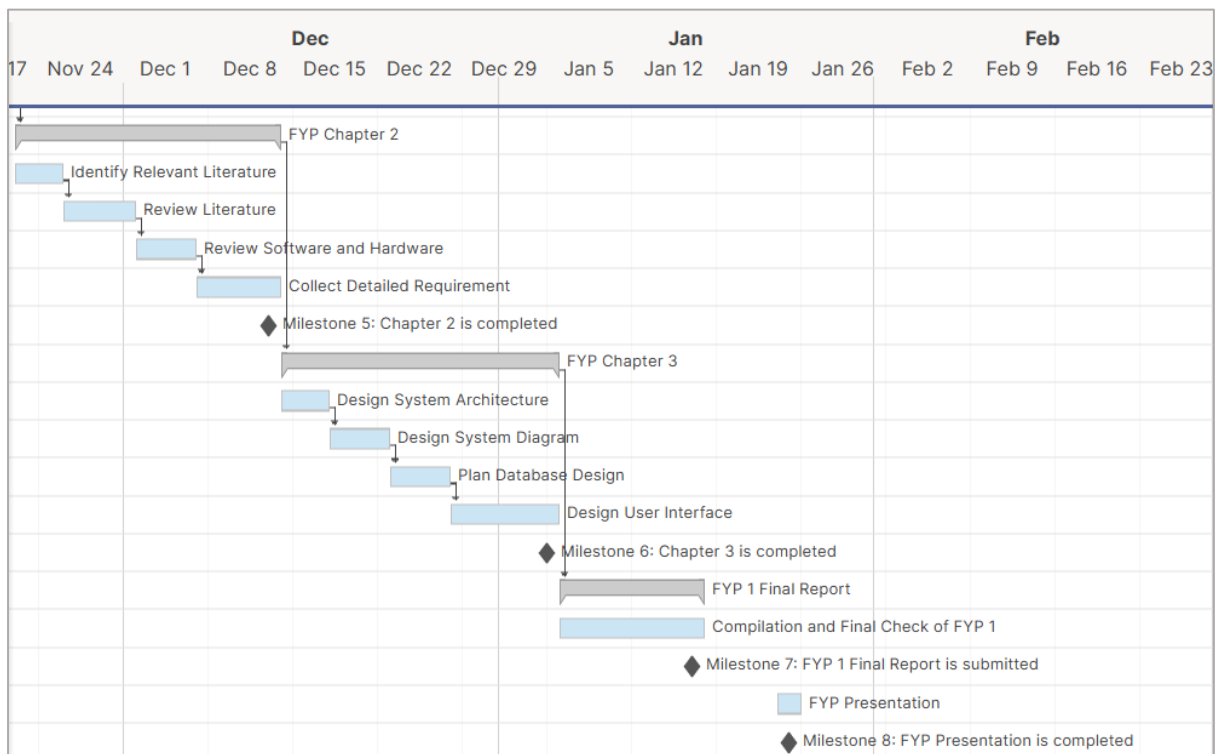


Figure 1.4: Gantt Chart of FYP 1.

	ACTIVITIES	START	CD	DUE
	Final Year Project 2:	24/Mar	101d	02/Jul
1	✓ Revised Final Year Project 1	24/Mar	9d	01/Apr
2	✓ Create Gantt Chart for FYP 2	24/Mar	9d	01/Apr
3	✓ Milestone 1: Revised Final Year Project 1 submitted	01/Apr	1d	01/Apr
	✓ FYP Chapter 4	02/Apr	44d	15/May
5	✓ Development of Teacher-School Assignment Simulation	02/Apr	39d	10/May
6	✓ Draft Report For FYP Chapter 4	11/May	5d	15/May
7	✓ Milestone 2: Draft FYP Chapter 4 submitted	15/May	1d	15/May
	✓ Draft FYP Chapter 5, 6 and Abstract	16/May	16d	31/May
9	✓ Chapter 5	16/May	7d	22/May
10	✓ Chapter 6	23/May	7d	29/May
11	✓ Abstract	30/May	2d	31/May
12	✓ Milestone 3: Draft FYP Chapter 5, 6 and Abstract submitted	31/May	1d	31/May
	✓ Draft FYP 2 Full Report	01/Jun	10d	10/Jun
14	✓ Compile Chapter 1-6	01/Jun	10d	10/Jun
15	✓ Milestone 4: Draft FYP 2 Full Report submitted	10/Jun	1d	10/Jun
	✓ Submission of FYP 2 report	11/Jun	13d	23/Jun
17	✓ Revised FYP 2 Report	11/Jun	13d	23/Jun
18	✓ Milestone 5: FYP 2 Report submitted	23/Jun	1d	23/Jun
	✓ FYP Symposium	01/Jul	2d	02/Jul
20	✓ Milestone 6: FYP Symposium completed	02/Jul	1d	02/Jul

Figure 1.5: Project Schedule of FYP 2

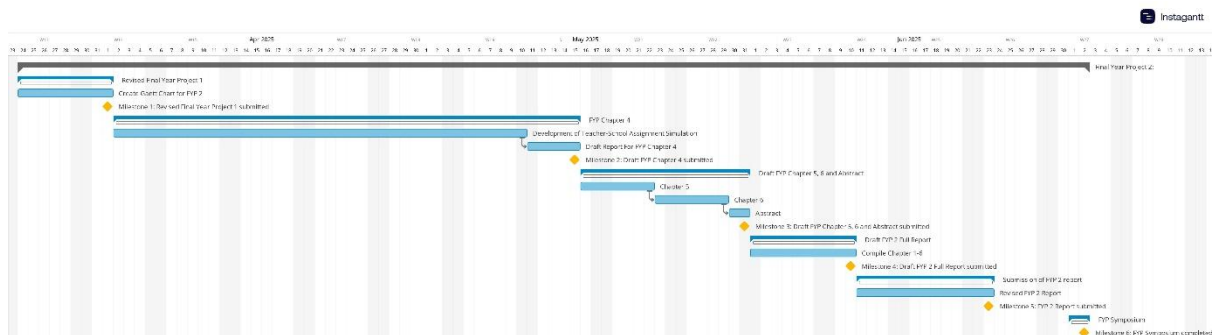


Figure 1.6: Gantt Chart of FYP 2

1.8 Expected Outcome

The expected outcomes of this project are as follows:

- The teacher allocation system can automate the allocation of teacher to eliminate the manual effort needed to allocate teacher.
- The teacher allocation system can ensure that every school have achieved optimal distribution of teachers.
- User-friendly and interactive interface of the system can ease the user while using the system.

1.9 Project Outline

There are six chapters consist in this FYP report. Chapter 1 gives an overview of the project which including the introduction of project, the problem statement, the project scope, the project objectives, the methodology, the significant of the project, the project schedule, the expected outcomes and the project outline.

Chapter 2 includes a literature review and a comparison of related online resources such as research papers on the related existing allocation approaches and comparable web-based applications.

Chapter 3 covers the methodology used to solve the teacher-school allocation problem. This chapter identifies the required datasets and designs a solution to the teacher-school allocation problem. The tools used for developing the design will also be discussed in this chapter.

Chapter 4 describes the design's implementation through allocation algorithm design and the system design. This chapter will cover every step of the project's development.

Chapter 5 covers all the evaluation testing take placed after the allocation algorithm is proposed to make sure the teacher-school assignment reached optimal solution. Every evaluation of the algorithm will be recorded in this chapter.

Chapter 6 discusses the project's conclusion and makes recommendations for further work on the proposed system.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter will focus on a review of existing allocation algorithm and systems relevant to the proposed system. Three relevant systems or research articles will be chosen from online resources and studied and analysed to further understand the concepts, technologies, approaches, and implementations. The information acquired will aid in determining the proposed system's potential requirements, enhancements, and limitations based on a comparison to the selected systems.

2.1.1 Assignment Problem

Assignment problem is essential in numerous sectors globally due to its practical significance and have been solved utilising a variety of ways. In education sector, assignment problem is categorized in two main category which is timetabling and allocation problem. The focus of this project is the teacher-school allocation problem which had been an issue for the school in underserved or rural areas. Allocation problems within the education sector frequently involve several constraints that must be addressed to maintain equity and efficiency. The key constraints that affect the allocation problem include teacher specialization and geographical constraints.

Teacher specialization is an important aspect in the teacher-school allocation process as it guarantees that students receive education from educators with knowledge in the subject areas they teach. According to studies (UNESCO, 2021), assigning teachers to fields outside of their specialization, sometimes known as "out-of-field teaching," can have a negative influence on student results, especially in complicated subjects like mathematics and science. Specialization not only increases the quality of instruction but also connects with national education goals to increase student achievement and eliminate learning gaps. Systems that emphasise teacher specialization in allocation models understand the need of matching teacher

qualifications to the subject requirements in order to sustain academic standards and promote equitable education delivery (OECD, 2019).

Geographical constraints play an important factor in teacher allocation systems, particularly in underserved or rural areas. These constraints take into factors such as the teacher's location, distance to the school, and availability of transportation which might influence a teacher's willingness and capability to work in specific schools. Teacher allocation system with geographical constraints will help to reduce the inequities by matching teachers not only to schools in need but also consider the teachers' location. This is crucial for increase the teachers job satisfaction.

2.2 Review on Existing Allocation Algorithms

Before selecting algorithms, it is important to thoroughly study and review existing algorithm to determine the greatest fit for the proposed system. Furthermore, creative designs can be taken from the studied solutions to generate excellent concepts for the proposed system.

2.2.1 Greedy Algorithm

The Greedy Algorithm is an essential technique in optimization and problem solving that involves making a sequence of locally optimal decisions at each stage with the goal of attaining a global result. This algorithm emphasises efficiency and simplicity by evaluating the optimal option immediately without taking into account the long-term effects of future decisions. The approach includes iteratively choosing the most promising candidate based on a predetermined criterion, such as maximizing utility or minimizing cost. According the studies Ali (2021), greedy algorithm is proposed to solve the student allocation problem that occur during internship program. The algorithm optimizes the overall distance and time needed for teacher visits. The proposed approach included the following key steps:

1. Initialize the distance matrix and capacity constraints for each university.

2. Calculate the cost of allocating each student to each university which taking into account constraints such as teacher-student ratios.
3. Students are distributed to universities in order of increasing expense until capacity has been reached.
4. The solutions are revised to reduce travel distances and ensure that all students are assigned to a university.

2.2.2 Priority Rule-Based Algorithm

The Priority Rule-Based Algorithm is a simple heuristic solution for solving assignment and scheduling problems. This algorithm operates by establishing a series of priority rules or criteria that control resource allocation. In resource allocation tasks, entities such as workers, machines, or educators are assigned according to their priority ranking which may be influenced by characteristics such as experience, availability, workload, or job urgency. The algorithm's simplicity comes from its reliance on established rules which make it computationally effective and quick to execute, particularly for problems with reasonable limitations. However, this approach may not always give the optimal solution. Alipour et al. (2020) have developed a Priority Rule-Based Heuristic Algorithm to solve the on-line order batching problem in an order picking warehouse. This algorithm ranks orders from customers according to predefined priority rules and allocates resources like batches and pickers. The author hopes to reduce the maximum completion time of all the batches. The proposed approach included the following key steps:

1. Define rules for batching orders and allocating resources.
2. Set threshold for the incoming orders and limit the sizes.
3. Sort the incoming orders according on the chosen priority rule and group them into batches based on the predefined batching parameters.
4. Allocate the highest-priority batch to an available resource.

5. Determine the best route for each batch.
6. Continue until all the orders are processed and stop.

2.2.3 Ultimate Hungarian Algorithm

The Ultimate Hungarian Algorithm is an optimized version of Hungarian Algorithm which solves assignment problems in polynomial time. These problems include distributing resources like workers and equipment to reduces or increases a particular cost or benefit which is often represented by a cost matrix. Chen et al. (2019) has proposed the ultimate Hungarian Algorithm to maximise task assignment in cases when several workers are assigned with varied tasks, each with differing efficiency. This algorithm developed uses a bipartite graph model, which one set of vertices representing workers and the other set representing jobs. The edges between these sets are weighted according to how efficient each worker is at each task. The proposed approach included the following key steps:

1. Label every node in one subset based on the maximum weight of the edges that connect them.
2. Create an initial matching M with edges marked with maximum weights and the algorithm terminated if the matching is matched.
3. Search for augmenting path from any unpaired vertices.
4. Change edges in M with those along the path that are not already in M after identifying an augmenting path.
5. Increase vertex labelling if no augmenting path is found.
6. Repeat steps 3 to 5 until an ideal fit is found.

2.3 Review on Similar Existing Systems

This section will review and analyse three relevant systems. The sources for the literature review will be "Excel Solver", "OpenSolver", and "AtoZmath.com" found on the Internet. The discussion will focus on the relevant features of the selected systems in

comparison to the proposed system. More information will be delivered in the following subsections.

2.3.1 Excel Solver

Excel Solver is an advanced add-in for Microsoft Excel which helps the user find the best solutions to mathematical and engineering problems. It's a flexible approach that could be applied to a variety of optimization problems including those in finance, operations research, and engineering. Below will show some of the Excel Solver interface:

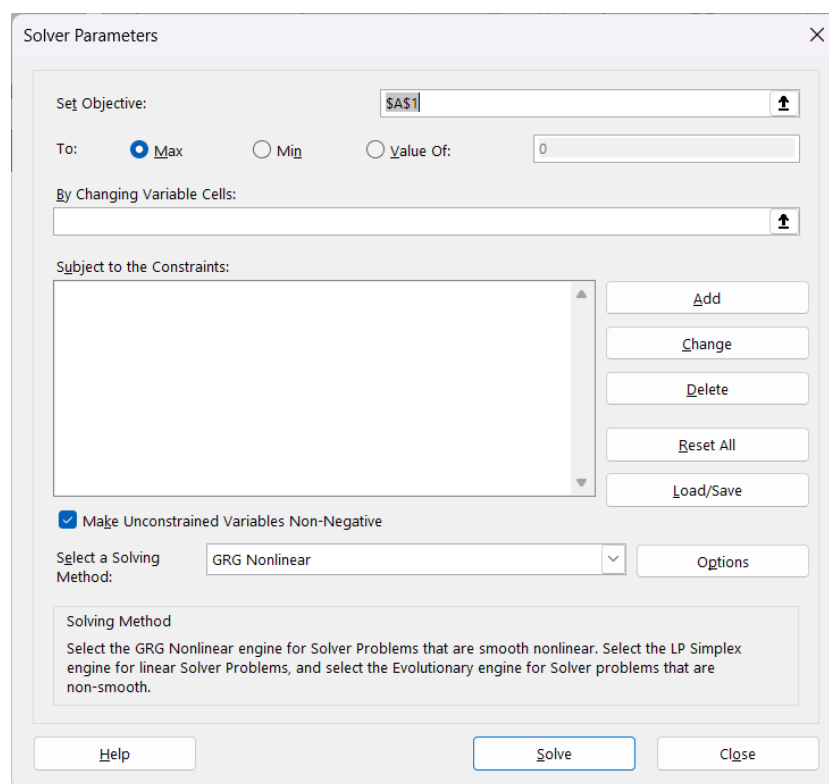


Figure 2.1: Overview of Excel Solver

Solver Parameters

Set Objective:

To: ☒ Max ☐ Min ☐ Value Of:

By Changing Variable Cells:

Subject to the Constraints:

☒ Make Unconstrained Variables Non-Negative

Select a Solving Method:

Solving Method: GRG Nonlinear, Simplex LP, Evolutionary

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Buttons: Add, Change, Delete, Reset All, Load/Save, Options, Help, Solve, Close

Figure 2.2: 3 Options of Solving Method

Solver Results

Solver found a solution. All Constraints and optimality conditions are satisfied.

☒ Keep Solver Solution ☐ Restore Original Values

☐ Return to Solver Parameters Dialog ☐ Outline Reports

Reports: Answer, Sensitivity, Limits

OK Cancel Save Scenario...

Solver found a solution. All Constraints and optimality conditions are satisfied.

When the GRG engine is used, Solver has found at least a local optimal solution. When Simplex LP is used, this means Solver has found a global optimal solution.

Figure 2.3: Solver Results

Figure 2.4: Add Constraints

There are some functions in Excel Solver will be listed in the table below:

Table 2.1: Functions in Excel Solver.

Functions	Descriptions
Set Objective	Figure 2.1 show the set objective function in the first row. The user can set their objective function of their problem by specifying the cell that contain the formula for the objective function. The user can select whether to maximize, minimize, or set the objective to a certain value.
Decision Variables	The user can select the cells in the spreadsheet that represent the variables the user wishes to modify to determine the best solution. These variables can be either continuous, integer, or binary.
Constraints	The user can define the constraints of the solution by clicking “Add” button shown in Figure 2.1 and it will redirect the user to Figure 2.4 which allow the user input the constraints.

	The user can add multiple constraints to the problem.
Solving Methods	Figure 2.2 show 3 options of solving method that can be chosen for the user. The user can choose the suitable solving method based on the nature of the problem. GRG Nonlinear is suitable for smooth nonlinear problems while Evolutionary is suitable for non-smooth nonlinear problems. LP Simplex can be used for linear programming problems.
Sensitivity Analysis	After the user solved the problem, user may conduct sensitivity analysis report that shown in Figure 2.3 to observe how different input parameters affect the ideal solution. This allows the users to make more precise decision on the problem.

Overall, Excel Solver is an excellent tool for resolving optimization issues within Microsoft Excel. Its user-friendly design and integration with Excel's familiar environment make it suitable for a wide range of users. Besides, Excel Solver also a useful tool for the user who lack of significant programming or optimisation knowledge to use it to solve optimization problem. However, its capabilities are restricted by its dependence on iterative techniques which can be computationally demanding for large tasks. Excel Solver can also have difficulty with large-scale problems and may not perform as well as specialized optimization software.

2.3.2 OpenSolver

OpenSolver is an effective open-source add-in for Microsoft Excel that expands the functionality of the built-in Solver tool. It offers a more efficient and reliable solution for linear and integer programming problems. It can tackle larger and more complex problems than the normal Excel Solver because of the use of the COIN-OR CBC solver. This makes it an invaluable resource for academics, analysts, and students who need to address optimization problems in the familiar Excel environment.

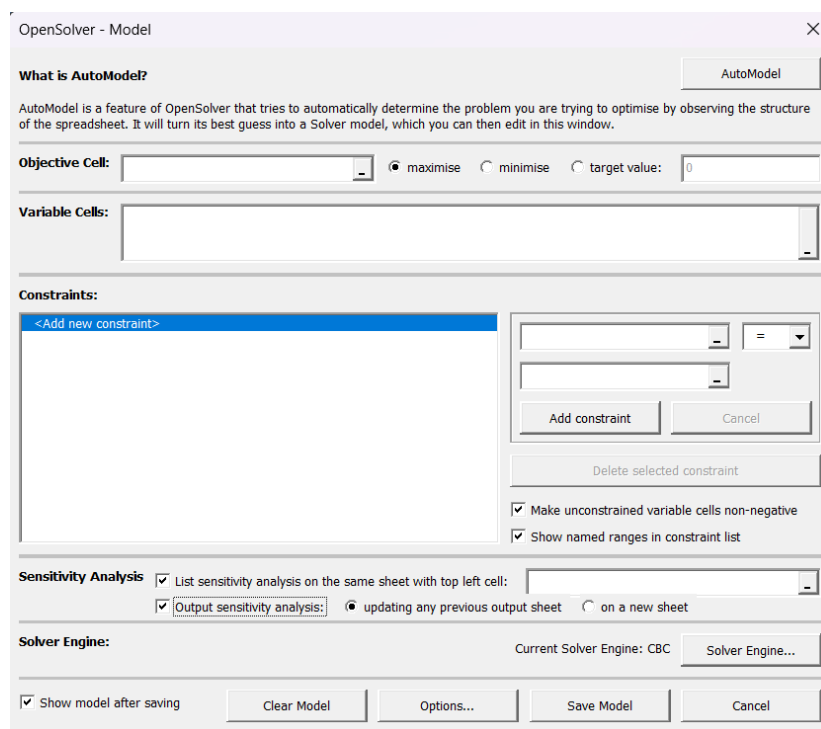


Figure 2.5: Overview of OpenSolver

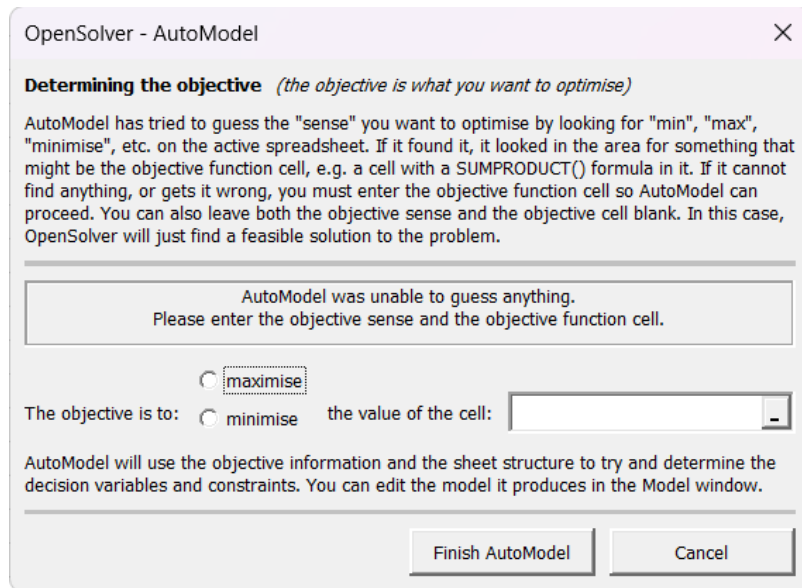


Figure 2.6: AutoModel in OpenSolver

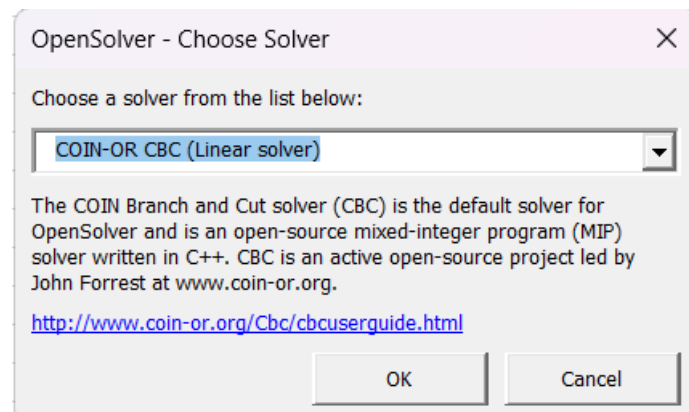


Figure 2.7: Solver Engine in OpenSolver

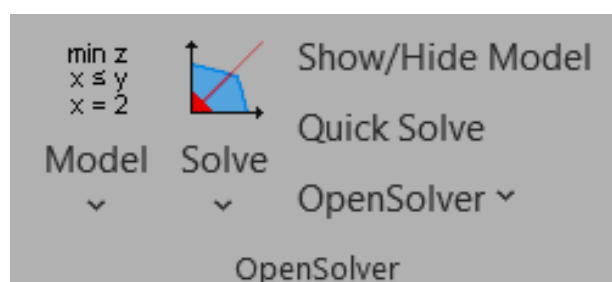


Figure 2.8: OpenSolver's button and menu in Excel's Data Ribbon

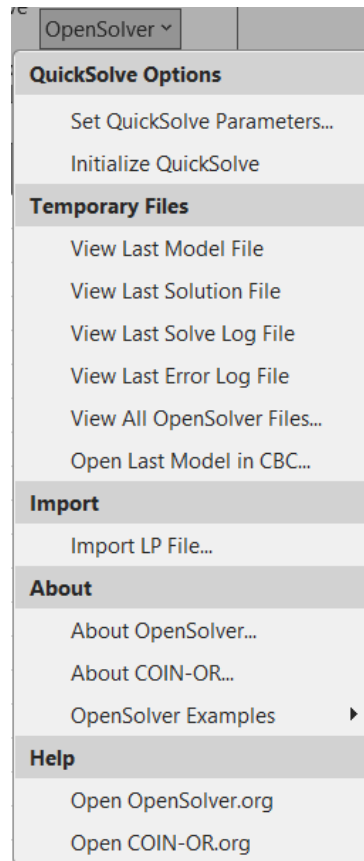


Figure 2.9: OpenSolver's menu for more advanced options

There are some functions in OpenSolver will be listed in the table below:

Table 2.2: Functions in OpenSolver.

Functions	Descriptions
AutoModel	Figure 2.6 show the AutoModel features in OpenSolver after the user click "AutoModel" button shown in Figure 2.5. The modal will use the objective information and the sheets enter by the user to determine the decision variables and constraints. This feature runs automatically and generates the Solver model.
Set Objective	Figure 2.5 show the set objective function in the first row. The user can set their objective

	function of their problem by specifying the cell that contain the formula for the objective function. The user can select whether to maximize, minimize, or set the objective to a target value.
Decision Variables	The user can select the cells in the spreadsheet that represent the variables the user wishes to modify to determine the best solution. These variables can be either continuous, integer, or binary.
Constraints	The user can define the constraints of the solution by inputting the constraints and clicking “Add Constraint” button shown in Figure 2.5. The user can add multiple constraints to the problem. The user can also delete the selected constraint.
Solver Engine	Figure 2.7 show the options of solver engine that can be chosen for the user. The user can choose the suitable solving method based on the nature of the problem. All the solver engine has detailed information so that the user can know the use of the solver engine.
Sensitivity Analysis	The user can choose to list the sensitivity analysis on the same sheet or output the sensitivity analysis in a new sheet or previous

	sheet which shown in Figure 2.5. This allows the user to observe how different input parameters affect the ideal solution. This allows the users to make more precise decision on the problem.
Quick Solve	Figure 2.8 show the Quick Solve feature in OpenSolver. The user can use the Quick Solve to solve the problem when repeated solving the same problem. This can shorten the time of resolving the models.

In summary, OpenSolver offers numerous features which ease the users to solve the assignment problem. OpenSolver provides the user with a powerful solver engine which is the COIN-OR CBC solver. This enables the user to handle complicated optimization problems efficiently. Besides, OpenSolver does not set artificial limits on the size of problem to solve which means that user can set many constraints for the problem. This allows the user to handle complicated optimization problems. On the other hands, the limitation of OpenSolver lies on the steeper learning curve needed for utilise the tool. It requires the user to have a basic understanding of the optimization concepts and Excel formula before using it. Furthermore, OpenSolver has limited documentation and community support for the users. This will cause problem for the users when they encounter issues with the tool and require troubleshooting.

2.3.3 AtoZmath.com

AtoZmath.com is an interactive online platform that helps users solve mathematical questions in a variety of fields which including algebra, calculus, statistics, and operations research. The website provides a variety of tools and calculators with step-by-step solutions to

problems. This resulting it is an excellent resource for students, instructors, and professionals. One of its most notable aspects is its operation research section, which provides solvers for the assignment problem that employ approaches such as the Hungarian algorithm for both minimization and maximization situations. AtoZmath.com assists users in improving their understanding and solving mathematics and optimisation problems more efficiently.

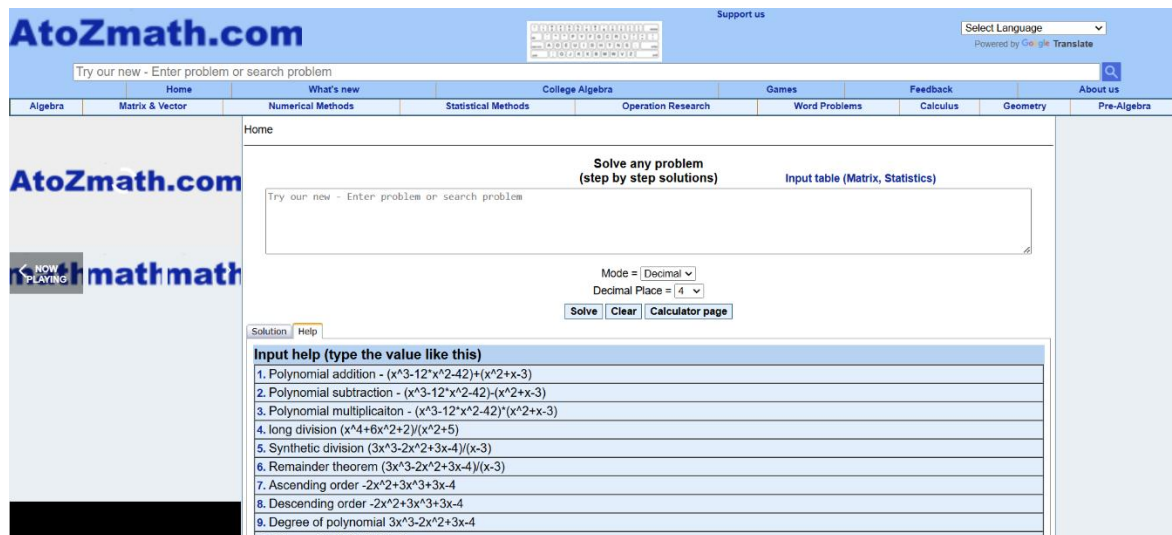


Figure 2.10: Dashboard of AtoZmath.com

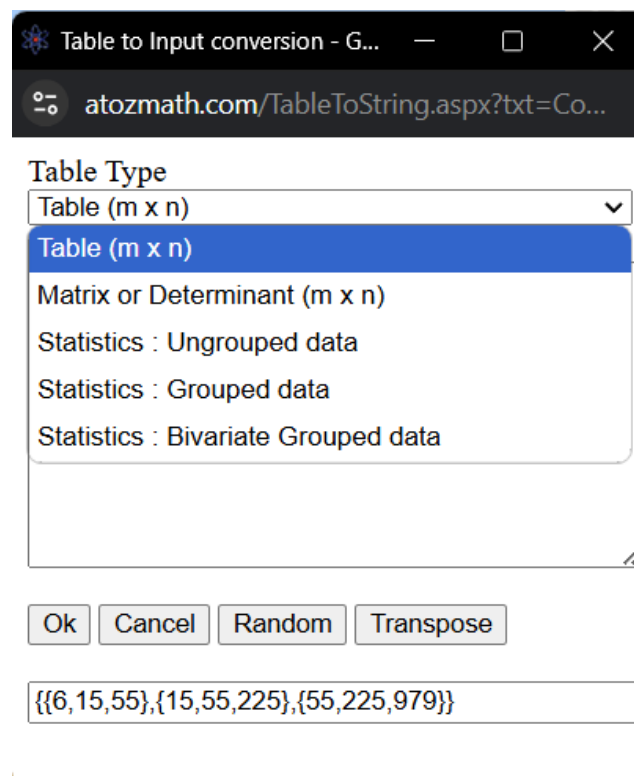


Figure 2.11: Input Table

AtoZmath.com

Try our new - Enter problem or search problem

Home | What's new | College Algebra | Games | Feedback | About us

Algebra | Matrix & Vector | Numerical Methods | Statistical Methods | Operation Research | Word Problems | Calculus | Geometry | Pre-Algebra

Home > Operation Research calculators

AtoZmath.com - Homework help (with all solution steps)

Educational Level: Secondary school, High school and College
 Program Purpose: Provide step by step solutions of your problems using online calculators (online solvers)
 Problem Source: Your textbook, etc

Operation Research Calculators (examples)

1. Assignment problem
 - 1.1 Assignment problem (Using Hungarian method-2)
 - 1.2 Assignment problem (Using Hungarian method-1)
 - 2.1 Travelling salesman problem using Hungarian method
 - 2.2 Travelling salesman problem using branch and bound (penalty) method
 - 2.3 Travelling salesman problem using branch and bound method
 - 2.4 Travelling salesman problem using nearest neighbor method
 - 2.5 Travelling salesman problem using diagonal completion method
 3. Crew assignment problem
2. Simplex method (Solve linear programming problem using)
 0. Formulate linear programming model examples
 1. Graphical method
 2. Simplex method (BigM method)
 3. Two-Phase method
 4. Primal to dual conversion
 5. Dual Simplex method
 6. Integer Simplex method (Gomory's cutting plane method)
 7. Branch and Bound method
 8. 0-1 Integer programming problem
 9. Revised Simplex method

Figure 2.12: Examples of Operation Research Calculators

AtoZmath.com

Home | What's new | College Algebra | Games | Feedback | About us

Algebra | Matrix & Vector | Numerical Methods | Statistical Methods | Operation Research | Word Problems | Calculus | Geometry | Pre-Algebra

Home > Operation Research calculators > Crew assignment problem calculator

Algorithm and examples

Crew assignment problem

Type your data, for separator you can use space or tab

a	06.00	12.00	1	05.30	11.30
b	07.30	13.30	2	09.00	15.00
c	11.30	17.30	3	15.00	21.00
d	19.00	01.00	4	18.30	00.30
e	00.30	06.30	5	00.00	06.00

OR

Rows: 5

City-1: City-2:

Delhi - Mumbai Mumbai - Delhi

Flight No	Departure	Arrival	Flight No	Departure	Arrival
a	06.00	12.00	1	05.30	11.30
b	07.30	13.30	2	09.00	15.00
c	11.30	17.30	3	15.00	21.00
d	19.00	01.00	4	18.30	00.30
e	00.30	06.30	5	00.00	06.00

Minimum layover of hours between flights:

[Solution](#) [Help](#)

Figure 2.13: Crew Assignment Problem Calculator

Solution

Solution will be displayed step by step (In 4 parts)

Solution

Crew assignment problem

Delhi - Mumbai			Mumbai - Delhi		
Flight No	Departure	Arrival	Flight No	Departure	Arrival
a	06.00	12.00	1	05.30	11.30
b	07.30	13.30	2	09.00	15.00
c	11.30	17.30	3	15.00	21.00
d	19.00	01.00	4	18.30	00.30
e	00.30	06.30	5	00.00	06.00

Minimum layover of hours between flights : 4

Solution:
 To determine optimal assignments, first we calculate layover times from the above time table.
 Calculating values for table 1 (layover time)

1st Row

Cell-1 Arrival time (Mumbai) = 12:00, Departure time (Mumbai) = 05:30, Waiting time (difference) = 17.5
 Cell-2 Arrival time (Mumbai) = 12:00, Departure time (Mumbai) = 09:00, Waiting time (difference) = 21
 Cell-3 Arrival time (Mumbai) = 12:00, Departure time (Mumbai) = 15:00, Waiting time (difference) = 27
 Cell-4 Arrival time (Mumbai) = 12:00, Departure time (Mumbai) = 18:30, Waiting time (difference) = 6.5
 Cell-5 Arrival time (Mumbai) = 12:00, Departure time (Mumbai) = 00:00, Waiting time (difference) = 12

Figure 2.14: Step by step solution

There are some functions in AtoZmath.com will be listed in the table below:

Table 2.3: Functions in AtoZmath.com.

Functions	Descriptions
Dashboard	Figure 2.10 show the dashboard of the AtoZmath.com. The user can enter their problem or search related problem in the text area. The user can also input the table by selecting the types of tables shown in Figure 2.11.
Different problem example	Figure 2.12 illustrate the different types of problem calculators in Operation Research section. The types of problem include assignment problem, simplex method, transportation problem, linear programming tools and others.
Step by step Solution	Figure 2.14 show the step-by-step solution of the problem entered. After the user clicked “Find” button shown in Figure 2.13, the system will show the step-by-step solution for the problem.

Thus, AtoZmath.com is a useful tool for addressing a variety of mathematics and optimization problems. Its comprehensive tools such as those for tackling assignment problems and developing transportation models, are very useful in operations research. Besides, the step-by-step solution also one of the strengths of the platform. The detailed explanation of the

solution will let the user have more understanding on the problem. However, one important limitation is the lack of customization options which may make it difficult for users to adapt the tools to very particular or advanced problem scenarios. This causes their flexibility for the professional use is reduced.

2.4 Comparison of the Existing System

The review and investigation of existing systems might give insights into the features and functions that could be included in the proposed system. The table below compares the features of existing systems.

Table 2.4: Comparison of the Features of Three Existing Systems.

Systems Features	Existing Systems		
	Excel Solver	OpenSolver	AtoZmath.com
Dashboard	✗	✗	✓
Add Constraints	✓	✓	✓
Sensitivity Analysis	✓	✓	✗
Solver	✓	✓	✗
Customization			
Multi-Objective	✓	✓	✗
Optimization			
Error Handling	✓	✓	✗
Step-by-Step	✗	✗	✓
Explanation			

Examples and Templates	✗	✓	✓
---------------------------	---	---	---

Based on Table 2.4, OpenSolver appears to be the best tool among the three existing tools for solving optimization problems. This is because OpenSolver provides numerous significant features that are essential for addressing complicated optimization issues, including the capacity to handle a larger number of variables and constraints, integration with the COIN-OR CBC solver for improved efficiency, and support for a variety of optimization techniques. Although Excel Solver is user-friendly, it has constraints in managing large-scale issues and may lack the efficiency of OpenSolver for complicated optimization tasks. Lastly, AtoZmath.com is a useful tool for learning and practicing mathematical issues other than solving large scale of optimization problems. This is because of the lack of necessary features that used to solve the complex optimization problem.

2.5 Evaluation Metrics for Teacher-School Allocation

To evaluate the accuracy of teacher-school allocation, different types of evaluation metrics have been proposed by the previous studies. These metrics help to evaluate the accuracy of subject assignment and the geographical preferences satisfaction for the model.

2.5.1 F1-Score

The F1 score is a crucial evaluation metric used in classification tasks to assess a model's performance. It integrates precision and recalls into a singular metric. It offers equilibrium between these two measurements and is often used in classification, resource allocation, and optimization issues such as teacher-school assignments. The formula of F1-score is as follow:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

From the formula above, the Precision is defined as the ratio of correct positive predictions to all positive estimations (Chen & Xu, 2024). Recall indicates how effectively a model can detect true positive cases. It can be expressed as the number of true positive predictions divided by the total number of real positive cases. The F1-score, with values ranging from 0 to 1 is a helpful metric for measuring performance, particularly in circumstances of unequal class distribution or varied effects of false positives and false negatives on job performance. According to Chen and Xu (2024), the authors used F1-score as one of the evaluation metrics to evaluate each algorithm that used for automatically grading students' assignments. This provides the authors a valuable insight of the effectiveness of each algorithm in enhancing the assessment processes. Thus, the F1-score is suitable to use as the evaluation metric for the subject specialization assignment in teacher-school allocation.

2.5.2 Preferences Satisfaction Rate

Preference Satisfaction Rate (PSR) is a metric that determines how effectively an allocation system matches individuals' preferences. It is widely used in workforce allocation, educational planning, job matching, and resource distribution to assess whether individuals are assigned to their chosen options which including regions, job roles, or institutions. PSR is defined as the percentage of people who acquire assignments that match their stated preferences. A high PSR implies a greater match between preferences and actual assignments whereas a lower PSR indicates many individuals were assigned to non-preferred options. The formula of Preference Satisfaction Rate (PSR) is as follow:

$$PSR = \frac{\text{Individuals Assigned to Preferred Options}}{\text{Total Assigned Individuals}} \times 100\%$$

2.6 Conclusion

Chapter 2 delivered a review of the literature on existing allocation algorithm and systems to inform the design and deployment of teacher allocation web-based system. The

analysis consisted of three notable platforms, Excel Solver, OpenSolver, and AtoZmath.com. Each platform was carefully evaluated for its features, strengths, and weaknesses.

CHAPTER 3: REQUIREMENT ANALYSIS AND DESIGN

3.1 Introduction

After reviewing several research on assignment methods from the previous chapter, this chapter will focus on the method that will be used to develop a simulation to solve the teacher-school allocation problem. It is important to choose an appropriate method in ensuring the project is completed successfully and on time. The problem will be identified carefully and the allocation algorithm is developed to solve the allocation problem in this chapter. The possible issues faced for the solution will also be discussed and followed by the discussion on the tools used in the implementation phase.

3.2 Problem Description

The teacher-school allocation problem relies on the challenge of assigning teachers to schools in a way that meets the educational goals of the schools while also considering the teachers' preferences. The issue is particularly serious in rural areas of Sarawak where teacher shortages, geographical limits, and poor infrastructure make it difficult to ensure appropriate staffing. The issue refers to teachers who possessing unique qualities such as subject expertise, location preferences and schools which have particular needs including subject demand and recruitment capacity. The goal is to create an optimal allocation approach that aligns teacher preferences with school needs which ensuring that underprivileged schools obtain the necessary teaching resources while remaining fair and efficient in the allocation process.

3.2.1 Teachers

Teachers play an important part in the allocation simulation since they are the major resource for meeting the educational needs of schools. Each teacher is defined by unique attributes such as subject specialization and geographic preferences. These characteristics influence the allocation process which ensuring that teachers are assigned to schools that match their skills and preferences.

Subject specialization of a teacher is a critical aspect in the allocation process. It is important to ensure the subject expertise of the assigned teachers align to the specific subject demand of the schools. This can reduce the risk of mismatch of subject specialization in the schools and improve the educational level of the schools. Besides, matching subject specialization with school demands improves resource utilisation and ensures that available teaching skill is efficiently distributed across all subjects.

Geographical preferences of teachers are another aspect that need to be considered in the allocation process. Teachers frequently express preferences for urban or rural postings based on criteria such as closeness to home, family needs, and availability to facilities. This causes the schools in rural area faced the shortage of teachers due to the location. Thus, it is important for the teachers to state their preferable distance between their house and the schools for the allocation process. Thus, this can make sure that the allocation process assigns teachers based on the teacher preferences.

Alternative methods are critical for dealing with situations in which a teacher cannot be assigned to their first preference because of constraints such as school capacity, subject mismatches, or geographic preferences. One option to solve this problem is to reallocate the teacher to nearby schools that closely align with their preferences. Besides, reconsideration of the subject preferences of the teachers can be another way to solve the problem. For example, the teacher could be allocated the subject in related field with their preferences. Thus, these methods will solve the problem and ensure the optimal assignment reached.

3.2.2 Schools

The teacher-school allocation problem focuses on the rural schools in Sarawak. The schools considered in this problem are in the area of Siburan, Bau, Lundu, Asajaya, Serian and other rural areas. Schools in these areas often faced issues such as teacher shortages and limited

resources because geographical location of the schools. To solve the allocation problem, there are few aspects of the schools need to be considered which are the subject demand by the schools, location of the school, and capacity of the school.

Subject demand is an important issue for the schools in rural area of Sarawak. Subject such as Mathematics, Science, English and other specialised field are always lack of qualified teachers with specialized knowledge in these fields. Thus, these subjects must be prioritizing in the allocation process in order to solve issue of the schools. The assigned teachers must have the specific subject specialization which align with the school's subject demand. By clearly identifying subject demand, schools guarantee that the allocation process is in line with their educational objectives and fulfils essential teaching tasks.

Schools are classified depending on their geographical regions, such as rural, urban, or semi-urban locations, and each presents unique obstacles in the allocation process. Rural schools frequently have crucial challenges in attracting teachers due to reasons such as insufficient infrastructure, remote locations, and accessibility limitations, which may prevent teachers from accepting assignments. Thus, the distance between the school and the teacher location preferences will be considered and the nearest distance will be first chosen during the allocation process.

The school's capacity refers to the number of teachers that can be assigned to a specific subject for the schools. This capacity is influenced by factors such as the number of classrooms, student enrolment in each topic, and the total need for teaching professionals in a specific subject area. It is important for the allocation process to ensure that teacher assignments are in line with the school's actual needs, avoiding overstaffing in specific subjects and providing a fair allocation of teaching resources across all subject areas.

3.3 Modelling the Teacher-School Allocation Problem

After studying on the teacher-school allocation problem from the previous section, this section will discuss the constraints and objective function used to solve the teacher-school allocation problem.

3.3.1 Constraints

Constraints are critical in the Teacher-School Assignment Problem as they specify the constraints and conditions that must be met when assigning teachers to schools. Constraints can be categorised as:

Hard Constraints:

- A teacher can only be assigned to one subject at a time.
- A school can only receive teacher with subject specialize that match with the subject demand of the school.
- Number of assigned teachers for a school cannot exceed the capacity of the school.

Soft Constraints:

- Teachers can specify the subject that they prefer to teach.
- Teachers can specify the ranges of school that they prefer to assigned to.
- Schools can set the number of teachers required for the schools.
- Schools can specify the subjects needed by the schools.

3.3.2 Objective Function

Optimize the assignment of teachers to schools while fulfilling all the constraints and maximizing specific criteria.

3.4 Methodology

This section will explain and design the algorithm used to solve the allocation problem and followed by the tools that used to develop the simulation and the interface for the excepted

output solution. The detailed steps of the algorithm will also be showed and discussed in the section.

3.4.1 Algorithm Design

The Algorithm Design section focusses on developing the logical and procedural steps required to effectively solve the teacher-school allocation problem. A heuristic algorithm, Greedy Algorithm is used to solve the teacher-school allocation problem. There are a few constraints such as subject specialization, geographical preferences, and school capacity will be considered while designing the algorithm. The main steps of the algorithm are showed as follow:

Step 1: Input the teacher data, school data and constraints

- Input teacher data such as name, subject specialization and location preferences.
- Input school data like name, subject demand, capacity and location
- Determine the constraints of allocation.

Step 2: Sort the teachers in a list according to a priority criterion

- Teachers with fewer preferences options will be assign first to reduce the risk of unassigned teachers.
- Teachers with high demand subjects such as Mathematics, English will also be assigned first.

Step 3: Sort the schools in a list based on school demand.

- Schools with high demand of subjects will be processed first.

Step 4: Iterate the teacher list for allocation until optimal assignment found

1. Find matching schools

- Determine schools that match with the teacher's subject specialization and preferences.
- Filter out schools that do not meet these criteria.

2. Check school capacity

- Make sure the number of assigned teachers does not exceed the school's maximum capacity.

3. Assign teacher

- Assign the teacher to the highest-priority school that meets the criteria.

4. Update the school capacity

- Reduce the school's remaining capacity to reflect the new allocation.

Step 5: Handle unallocated teachers

- Mark the unallocated teachers as unallocated.
- Match again with the closest preferences.

Step 6: Display the assignment results

- A list of teacher-school assignment will be displayed.

Figure 3.1 shows the flowchart of the proposed algorithm used to solve the teacher-school allocation problem.

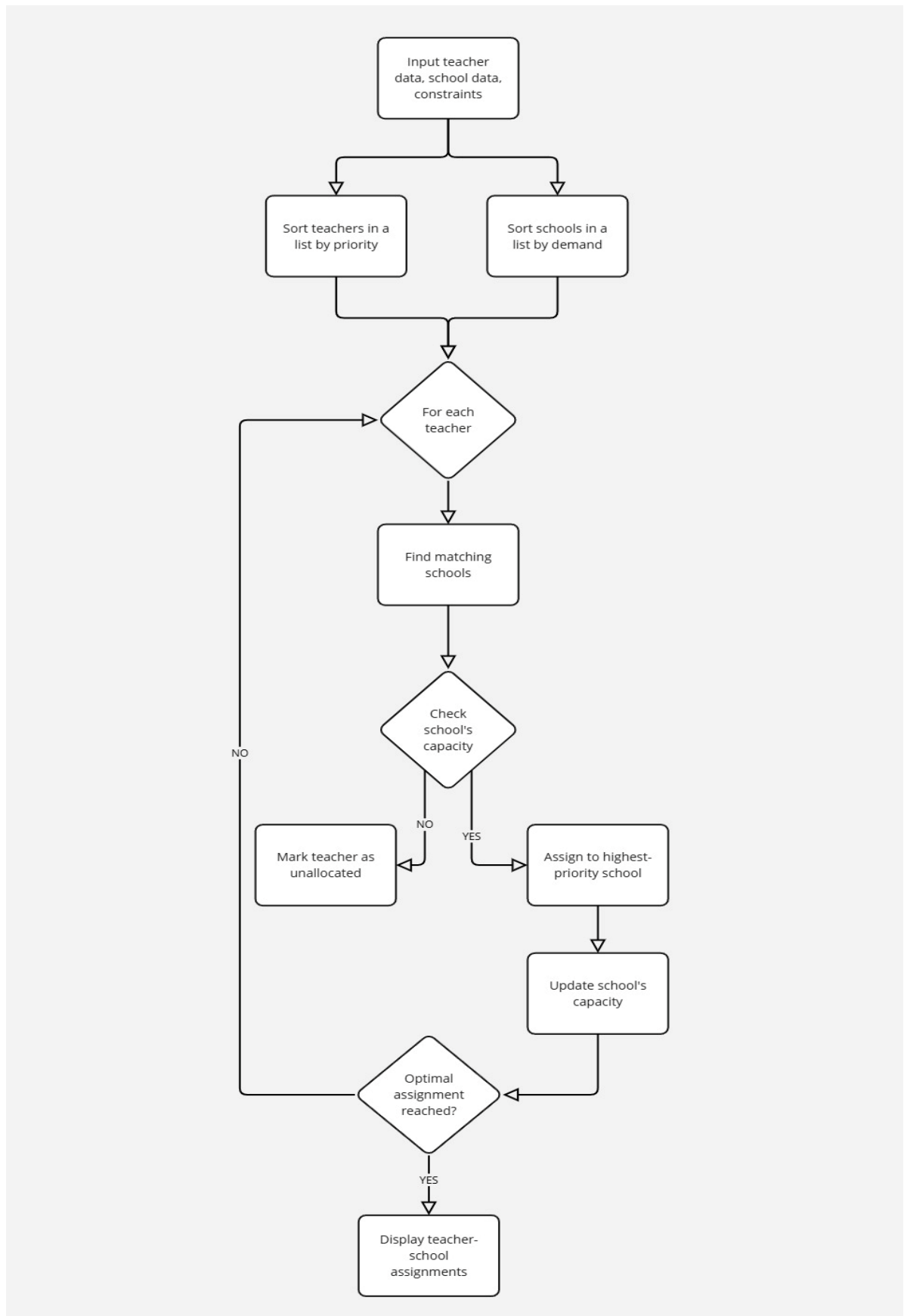


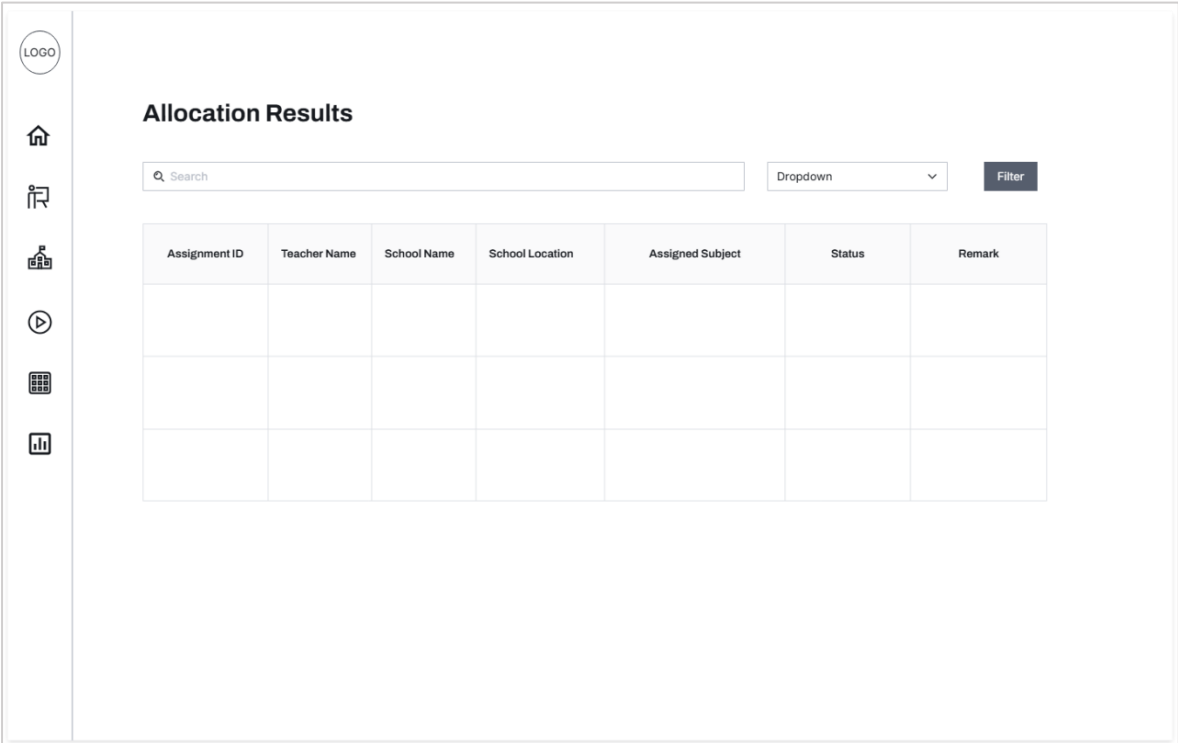
Figure 3.1: Flowchart of the Proposed Algorithm

3.4.2 Implementation

PHP programming languages and MySQL database will be used for the implementation of the teacher-school allocation simulation. PHP is suitable for creating algorithms and managing the logic required to handle teacher and school data. Besides, PHP allows for smooth interaction with Laravel which provides a user-friendly interface for data entry, simulation execution, and result presentation.

MySQL database is utilised to store and manage information on teachers, schools, subjects, and allocation results. It preserves data integrity and enables quick querying of big datasets which making it ideal for the allocation process. The data stored in the database will then be used for the allocation process which the algorithm will retrieve the data from the database and use to find the optimal solution for teacher-school assignment.

The interface of the output is shown below:



The screenshot displays a web application interface titled "Allocation Results". On the left is a vertical sidebar with a "LOGO" at the top and several icons below it. The main content area features a search bar with a magnifying glass icon and the text "Search", a "Dropdown" menu, and a "Filter" button. Below these elements is a table with the following structure:

Assignment ID	Teacher Name	School Name	School Location	Assigned Subject	Status	Remark

Figure 3.2: Allocation Results

3.5 Possible Issues

There will be several possible issues faced during the allocation process. One of the possible issues may arise is the solution may not reach optimal because of the performance and efficiency of the chosen algorithm. Besides, the reliance on accurate and comprehensive data for teachers and schools will be another possible issue. Any errors or missing information could result in inefficient allocations. Moreover, constraints such as geographic preferences and school capacity limitations may lead to certain teachers being unassigned. Lastly, the algorithm may suffer with scalability if the number of teachers and schools increases rapidly, thereby affecting the system's performance and efficiency. Thus, the optimal solution heavily depends on how these issues been solved.

3.6 Conclusion

In conclusion, this chapter has presented a detailed discussion on the methodology used to solve the teacher-school allocation problem. The incorporation of specific constraints, such as subject specialization, geographic preferences, and school capacity which makes sure that the solution is both practical and applicable to real-world situations. The chapter also outlined the allocation design from the allocation problem for further implementation in next chapter.

CHAPTER 4: IMPLEMENTATION AND TESTING

4.1 Introduction

Chapter 4 will explore and present the implementation of the Teacher-School Assignment Simulation using the algorithm proposed in Chapter 3. The Teacher-School Assignment Simulation was implemented and tested using Microsoft Visual Studio Code. Besides, this chapter will also include the information about the simulation's input data. This chapter will also provide a detailed description of how the algorithm operates during the simulation. Finally, this chapter ends with a conclusion.

4.2 System Requirements and Specification

This section will clearly capture and describe the requirements for software and hardware. Considering a suitable software and hardware is critical to ensuring that the simulation runs well.

4.2.1 Software Modules

The Teacher-School Assignment Simulation depends on a specific software environment to guarantee solid development and implementation. Microsoft Visual Studio Code, a flexible source code editor that supports PHP, HTML, CSS, and JavaScript is been used for the development of the algorithm. Besides, the backend is created with Laravel 11, which provides an organised and efficient platform for handling the assignment process and data exchanges. MySQL database is used to store teacher, school, subject, and assignment information which ensuring the flexibility and reliability for multiple simulation datasets. The software requirement needed for the implementation of the simulation is outlined below.

Table 4.1: Software Requirements.

Software	Specifications
Operating System	Windows 10 or higher
Web Browser	Google Chrome
Source Code Editor	Microsoft Visual Studio Code
Programming Languages	PHP, HTML, CSS, JavaScript

Programming Framework	Laravel 11
Database	MySQL
User Interface	Materialize CSS template

4.2.2 Hardware Requirement

The simulation is run on a laptop with an Intel or AMD dual-core CPU to meet the processing demands of the Laravel-based backend and JavaScript-driven frontend. A minimum of 8 GB of RAM is necessary for multitasking and smooth functioning of Microsoft Visual Studio Code, MySQL, and web browsers. A screen resolution of 1280x768 or greater is required for clear visualisation of the interactive map and user interface. Lastly, a steady and high-speed internet connection is crucial for obtaining OpenStreetMap tiles which enabling smooth map rendering and data exchanges during simulations. The hardware requirement needed for the implementation of the simulation is outlined below.

Table 4.2: Hardware Requirements.

Hardware	Specifications
Laptop Processor	Intel or AMD Dual core
Memory (RAM)	8 GB or higher
Screen Resolution	1280x768 or higher
Internet Connection	Stable and high-speed Internet Access

4.3 Input Data

The Teacher-School Assignment Simulation requires several input data to run the assignment. This section will display the input data needed for the proposed algorithm. The teacher, school and subject data will be shown below.

4.3.1 Teacher Data

The teacher data will consist of the data such as teacher id, name, address, latitude, longitude subject preference, location preference and dataset name. User can either submit the teacher data

through filling the submit form or upload a csv file. Below show the interface of the teacher data and the table to store the teacher data.

Teacher

Name

Address

Subject Preference
Select Subject

Location Preference
Select Location

Save Edit

Or Upload CSV File

Upload CSV File
Choose File No file chosen

Upload

Filter by Dataset:
All

ID	NAME	SUBJECT PREFERENCE	LOCATION PREFERENCE	ACTION
1	Heather Ng	Pendidikan Moral, Sejarah, Bah...	40km or more	Delete
2	Regina Chan	Biology, Accounting, Physics,...	20-30km	Delete
3	Donald bin Azlan	Pendidikan Islam, Economics, E...	10-20km	Delete
4	Ryan Lau	Pendidikan Islam, Sejarah, Acc...	40km or more	Delete
5	Ashley anak Ridzuan	Business Studies, ICT, Physics...	40km or more	Delete
6	Maria Bala	Pendidikan Islam, Economics, B...	10-20km	Delete
7	Joe Teo	Pendidikan Jasmani & Kesihatan...	30-40km	Delete
8	Barbara Ng	Bahasa Melayu, Pendidikan Isla...	40km or more	Delete
9	Carlos Lim	Business Studies, Accounting,...	30-40km	Delete

Figure 4.1: Input Teacher Data

Table 4.3: Table of Teacher Data

ID	Name	Address	Latitude	Longitude	Subject Preference	Location Preference	Dataset Name

4.3.2 School Data

The school data will consist of the data such as school id, name, address, latitude, longitude and subject demand and capacity. User can either submit the school data through filling the submit form or upload a csv file. The data of the schools will be shown using a map. Below show the interface of the school data and the table to store the school data.

Schools

School Name

School Address

Subject Demand
Select Subject Demand

Subject Capacity

Save

Or Upload CSV File

Upload CSV File
Choose File No file chosen

Upload

Figure 4.2: Input School Data

Table 4.4: Table of School Data

ID	Name	Address	Latitude	Longitude	Subject Demand & Capacity

4.3.3 Subject Data

The subject data will consist of the data such as subject id, subject name, subject priority, and subject category. User can either submit the subject data through filling the submit form or upload a csv file. Below show the interface of the subject data and the table to store the subject data.

Subjects

Subject Name

Subject Priority
Select Subject Priority

Subject Category
Select Subject Category

Save

Or Upload CSV File

Upload CSV File
Choose File No file chosen

Upload

ID	SUBJECT NAME	SUBJECT PRIORITY	SUBJECT CATEGORY	ACTION
1	Bahasa Melayu	High	Language	Delete
2	English	High	Language	Delete
3	Mathematics	High	Core Academic	Delete
4	Science	High	Core Academic	Delete
5	Sejarah	High	Core Academic	Delete
6	Additional Mathematics	High	Core Academic	Delete
7	Physics	High	Science	Delete
8	Chemistry	High	Science	Delete
9	Biology	High	Science	Delete
10	Pendidikan Moral	Low	Ethics & Social Studies	Delete

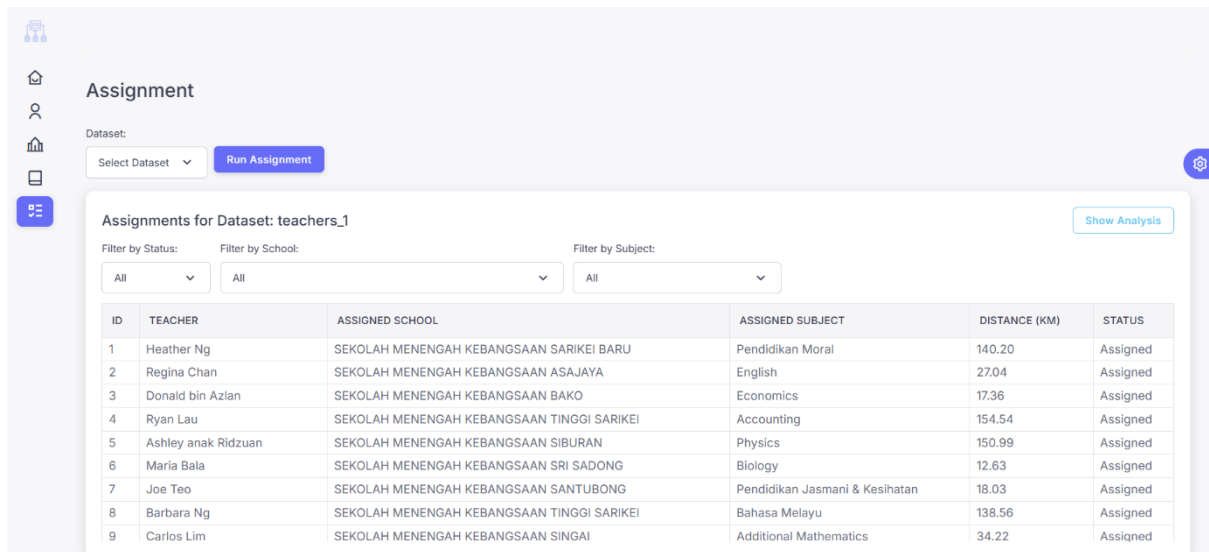
Figure 4.3: Input Subject Data

Table 4.5: Table of Subject Data

ID	Subject Name	Subject Priority	Subject Category

4.4 Output Result

After inserting all the necessary data that required for teacher-school assignment, the result of the assignment will store in a table that consist of assignment id, assigned teacher, assigned school, assigned subject, distance between teacher and school, and assignment status. Users can run the assignment by selecting the dataset and the result will display in a table. Users are able to find the specific result by filtering based on status, school or subject. Lastly, users can also view the assignment analysis by clicking the “Show Analysis”. The assignment analysis will consist of the information about total assignment, total assigned teacher, total unassigned teacher, assignment that within teachers’ location preference and subject preference, and school with no assigned teacher.



Assignment

Dataset: Select Dataset Run Assignment

Assignments for Dataset: teachers_1 Show Analysis

Filter by Status: All Filter by School: All Filter by Subject: All

ID	TEACHER	ASSIGNED SCHOOL	ASSIGNED SUBJECT	DISTANCE (KM)	STATUS
1	Heather Ng	SEKOLAH MENENGAH KEBANGSAAN SARIKEI BARU	Pendidikan Moral	140.20	Assigned
2	Regina Chan	SEKOLAH MENENGAH KEBANGSAAN ASAJAYA	English	27.04	Assigned
3	Donald bin Azlan	SEKOLAH MENENGAH KEBANGSAAN BAKO	Economics	17.36	Assigned
4	Ryan Lau	SEKOLAH MENENGAH KEBANGSAAN TINGGI SARIKEI	Accounting	154.54	Assigned
5	Ashley anak Ridzuan	SEKOLAH MENENGAH KEBANGSAAN SIBURAN	Physics	150.99	Assigned
6	María Bala	SEKOLAH MENENGAH KEBANGSAAN SRI SADONG	Biology	12.63	Assigned
7	Joe Teo	SEKOLAH MENENGAH KEBANGSAAN SANTUBONG	Pendidikan Jasmani & Kesehatan	18.03	Assigned
8	Barbara Ng	SEKOLAH MENENGAH KEBANGSAAN TINGGI SARIKEI	Bahasa Melayu	138.56	Assigned
9	Carlos Lim	SEKOLAH MENENGAH KEBANGSAAN SINGAI	Additional Mathematics	34.22	Assigned

Figure 4.4: Interface for Assignment

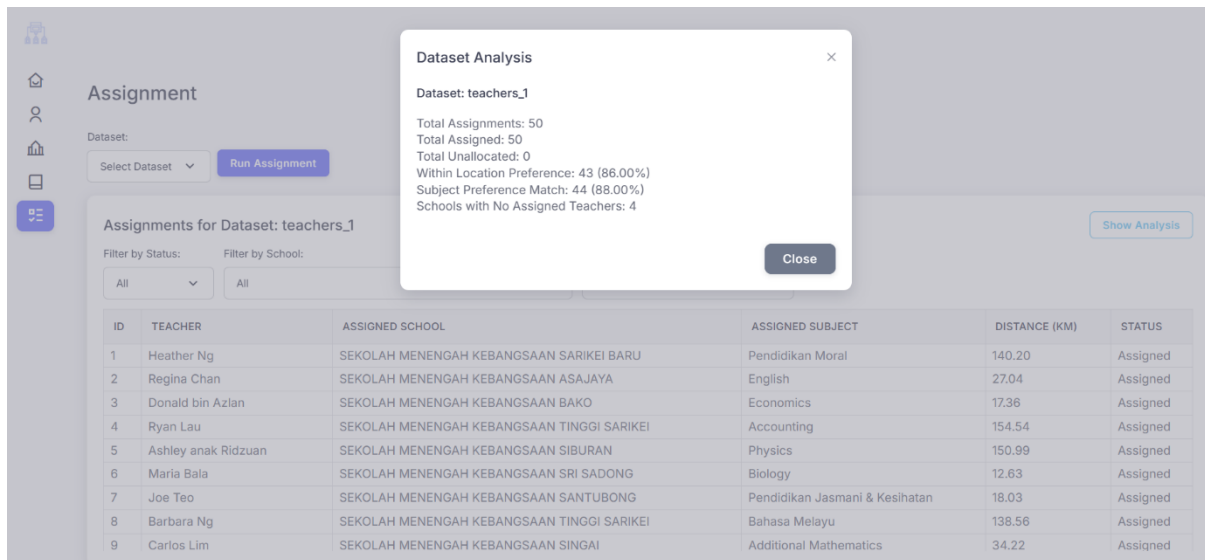


Figure 4.5: Assignment Analysis

Table 4.6: Table of Assignment

ID	Teacher ID	School ID	Subject ID	Distance	Status

4.5 Processes of Coding Algorithm

After the user inserted the data in the database that required to run the assignment, the algorithm will first retrieve the teacher data with valid coordinates and dataset name. Then, the algorithm will sort the teacher data based on the subject priority from high to low. Teachers with high-priority subject preferences will be prioritized first in the assignment to ensure that the assignment reaches optimal. Teacher with high-priority subject will be given score of 1 and teacher with low-priority subject will be given score of 2 to differentiate them. Below shows the illustration figure (Figure 4.6) and code in PHP (Code Snippet 4.1 & 4.2).

Teacher		Subject		Sorted Teacher List	
ID	Subject Preference	Name	Subject Priority	ID	Subject Preference
1	Geography	Mathematics	High	2	Mathematics
2	Mathematics	Geography	Low	1	Geography

Figure 4.6: Sorted teacher list

```
$query = Teacher::whereNotNull(columns: ['latitude', 'longitude']);
$query->where(column: 'dataset_name', operator: $selectedDataset);
$teachersForAssignment = $query->get();
$teachers = $this->sortTeacher(teachersCollection: $teachersForAssignment);
$schools = $this->sortSchool();
```

Code Snippet 4.1: Code to retrieve teacher data

```
private function sortTeacher($teachersCollection): mixed
{
    $subjectPriorities = Subject::pluck(column: 'subject_priority', key: 'subject_name');
    return $teachersCollection->sortBy(function ($teacher) use ($subjectPriorities): int {
        $preferences = array_map(callback: 'trim', array: explode(separator: ',', string: $teacher->subject_preference));
        $priorityScores = array_map(callback: function ($pref) use ($subjectPriorities): int {
            return match ($subjectPriorities[$pref] ?? 'Unknown') {
                'High' => 1,
                'Low' => 2,
                default => 3,
            };
        }, array: $preferences);
        $highestPriority = !empty($priorityScores) ? min($priorityScores) : 3;
        return $highestPriority;
    })->values();
}
```

Code Snippet 4.2: Code for sort teacher

After the sorting of teacher data, the algorithm will sort the school list based on the subject priority and the total subject capacity. School with high-priority subject demand and higher total subject capacity will be prioritized first in the assignment. For example, school 2 has subject demand of Mathematics and Science with capacity of 2 respectively since both are high-priority subject and the total subject capacity is 4 then school 2 will be prioritized first compared to other schools. Below shows the illustration figure (Figure 4.7) and code in PHP (Code Snippet 4.3).

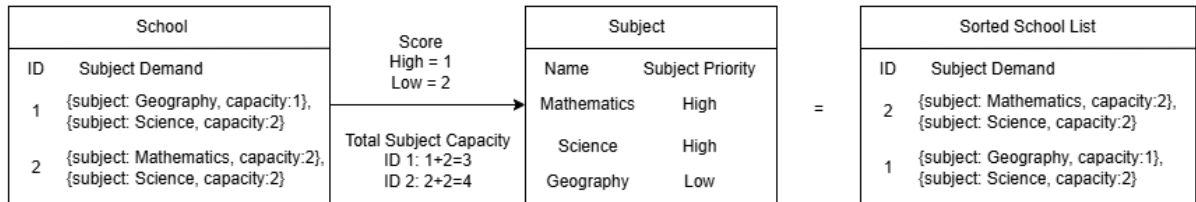


Figure 4.7: Sorted school list

```

private function sortSchool(): Collection
{
    $subjectPriorities = Subject::pluck(column: 'subject_priority', key: 'subject_name');
    return School::whereNotNull(columns: ['latitude', 'longitude'])
        ->get()
        ->map(callback: function (TModel $school): TModel {
            $school->demands = $school->subject_demand ?? [];
            return $school;
        })
        ->sortBy(callback: function (TModel $school) use ($subjectPriorities): int {
            $highestPriorityScore = 3;
            foreach ($school->demands as $demand) {
                $priority = $subjectPriorities[$demand['subject']] ?? 'Unknown';
                $score = match ($priority) {
                    'High' => 1,
                    'Low' => 2,
                    default => 3,
                };
                $highestPriorityScore = min(value: $highestPriorityScore, values: $score);
            }
            return $highestPriorityScore;
        })
        ->sortByDesc(callback: function (TModel $school): float|int {
            $totalCapacity = 0;
            foreach ($school->demands as $demand) {
                $totalCapacity += $demand['capacity'];
            }
            return $totalCapacity;
        });
}

```

Code Snippet 4.3: Code for sort school

After done sorting teacher and school data, the algorithm will then start the assignment phases. The algorithm will assign each teacher based on three prioritized phases which are High-Priority Subjects, Low-Priority Subjects and Same-Category Subjects. For high-priority subjects' phase, the algorithm will first find teachers with high-priority preferred subject and calls the findMatchSchools() function to find all schools with available for the subject. Then, selected schools will be filtered again by the algorithm which the algorithm will call findNearestSchool() function to select the closest school within the teacher's location preference. The distance between the school and teacher will be calculated using the Haversine formula. If the school is within the teacher's location preference, then the algorithm will create an assignment record. Below shows the illustration figure (Figure 4.8) and code in PHP (Code Snippet 4.4 & 4.5).

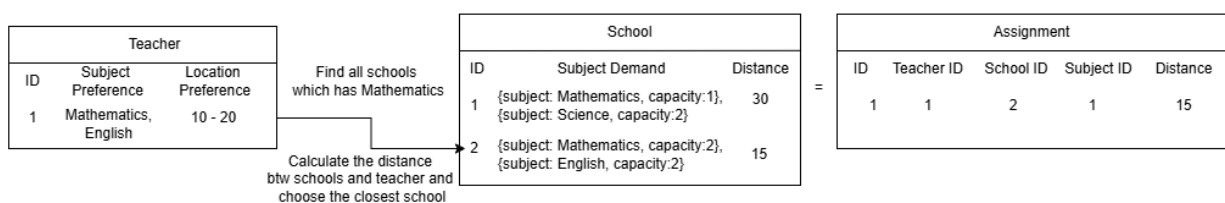


Figure 4.8: Assignment by high-priority subject

```
foreach ($preferences as $subjectPreference) {
    if ($this->attemptAssignmentByPriority([
        teacher: $teacher,
        subjectPreference: $subjectPreference,
        priority: 'High', schools: $schools,
        dataset: $selectedDataset])) {
        $assigned = true;
        break;
    }
}
```

Code Snippet 4.4: Code for calling High-Priority Subjects function

```
private function attemptAssignmentByPriority(Teacher $teacher, string $subjectPreference, string $priority, Collection $schools, ?string $dataset) {
    $subjectPriorities = Subject::pluck(column: 'subject_priority', key: 'subject_name');
    $teacherName = $teacher->name;
    if (($subjectPriorities[$subjectPreference] ?? null) === $priority) {
        $matchingSchools = $this->findMatchSchools(subjectPreference: $subjectPreference, schools: $schools, checkCategory: false, ...null);
        if ($matchingSchools->isEmpty()) {
            $nearest = $this->findNearestSchool(teacher: $teacher, schools: $matchingSchools);
            if ($nearest) {
                $assignedSchool = $nearest['school'];
                $distance = $nearest['distance'];
                $demands = collect(value: $assignedSchool->subject_demand);
                $assignedDemand = $demands->firstWhere(key: 'subject', operator: $subjectPreference);
                $subjectId = Subject::where(column: 'subject_name', operator: $subjectPreference)->value(column: 'subject_id');
                $canAssignPreferred = true;
                if ($assignedDemand['subject'] === 'Pendidikan Islam') {
                    if (stripos(haystack: $teacherName, needle: 'bin') === false && stripos(haystack: $teacherName, needle: 'Mohd') === false) {
                        $canAssignPreferred = false;
                    }
                } elseif ($assignedDemand['subject'] === 'Bahasa Cina') {
                    if (stripos(haystack: $teacherName, needle: 'bin') !== false || stripos(haystack: $teacherName, needle: 'Mohd') !== false) {
                        $canAssignPreferred = false;
                    }
                }
            }
        }
        $assignedCount = Assignment::where(column: 'school_id', operator: $assignedSchool->school_id)
            ->where(column: 'subject_id', operator: $subjectId)
            ->where(column: 'status', operator: 'Assigned')
            ->where(column: 'dataset', operator: $dataset)
            ->count();
        if ($assignedDemand && $canAssignPreferred && $assignedCount < $assignedDemand['capacity']) {
            $this->createRecords(teacher: $teacher, school: $assignedSchool, subjectId: $subjectId, distance: $distance, dataset: $dataset);
            return true;
        }
    }
}
```

Code Snippet 4.5: Code for High-Priority Subjects function

If the teacher does not assigned school in high-priority subject phase, the algorithm will then go to the next phase which is the low-priority subject phase. The algorithm will try to assign the teacher based on the low-priority subject in teacher's subject preferences. The flow of the assignment will be similar to the assignment in high-priority subject phase. Below shows the illustration figure (Figure 4.9) and the code in PHP (Code Snippet 4.6).

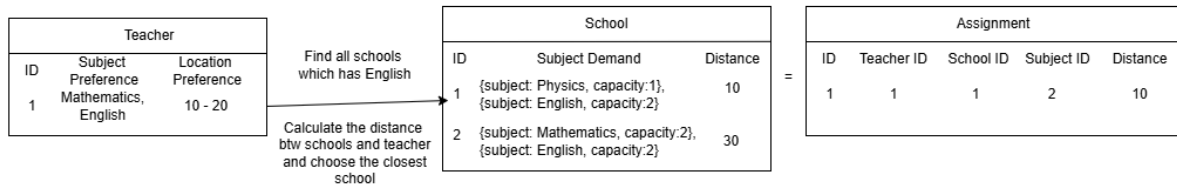


Figure 4.9: Assignment by low-priority subject

```
// Try Low priority subjects if still unassigned
if (!$assigned) {
    foreach ($preferences as $subjectPreference) {
        if ($this->attemptAssignmentByPriority(
            teacher: $teacher,
            subjectPreference: $subjectPreference,
            priority: 'Low',
            schools: $schools,
            dataset: $selectedDataset)) {
            $assigned = true;
            break;
        }
    }
}
```

Code Snippet 4.6: Code for calling Low-Priority Subjects function

If the teacher still does not assigned school in low-priority subject phase, the algorithm will then go to the next phase which is the same-category subject phase. The algorithm will try to assign a teacher to a school for either their preferred subject or another subject in the same category if the first two phases failed. For example, Teacher 1 has subject preference of Science but failing to assign in high-priority and low-priority phases due to location preference mismatches. The algorithm will find the subject that has same category as Science in subject table like Physics then it will filter out all the schools with subject demand of Physics and assign the teacher to the nearest school with the subject, Physics. Below shows the illustration figure (Figure 4.10) and the code in PHP (Code Snippet 4.7).

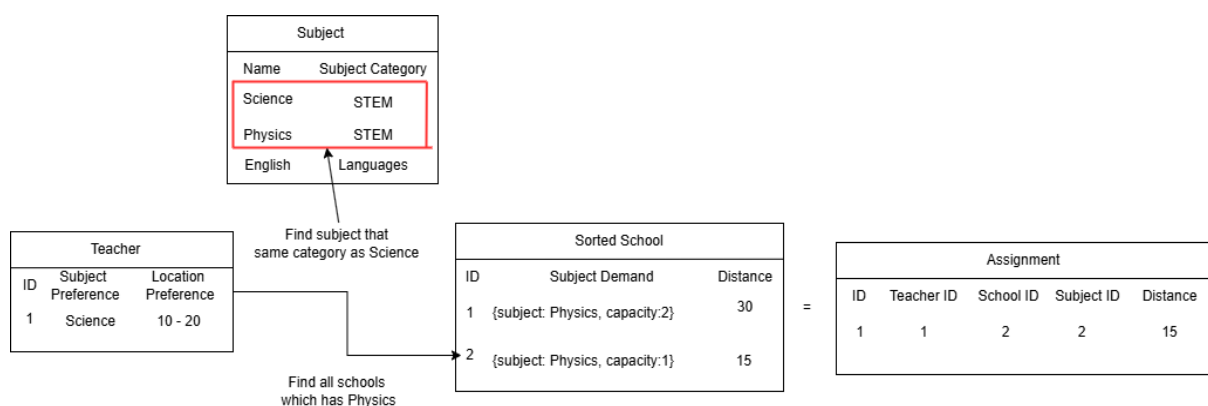


Figure 4.10: Assignment by same-category subject

```
// Try same category subjects if still unassigned
if (!$assigned) {
    foreach ($preferences as $subjectPreference) {
        if ($this->attemptAssignmentByCategory([
            teacher: $teacher,
            subjectPreference: $subjectPreference,
            schools: $schools,
            dataset: $selectedDataset])) {
            $assigned = true;
            break;
        }
    }
}
```

Code Snippet 4.7: Code for calling Same-Category Subjects function

Finally, if all three phases failed, the algorithm will create an unallocated record and saved in the database. The assignment status will be marked as Unallocated. Below shows the code in PHP (Code Snippet 4.8).

```
// If not assigned after all attempts
if (!$assigned) {
    Assignment::create(attributes: [
        'teacher_id' => $teacher->teacher_id,
        'school_id' => null,
        'subject_id' => null,
        'status' => 'Unallocated',
        'dataset' => $selectedDataset
    ]);
}
```

Code Snippet 4.8: Code for create unallocated record

Below shows some helper function used in the algorithm.

```
private function calculateDistance($lat1, $lon1, $lat2, $lon2): float
{
    $earthRadius = 6371;
    $dLat = deg2rad(num: $lat2 - $lat1);
    $dLon = deg2rad(num: $lon2 - $lon1);
    $a = sin(num: $dLat / 2) * sin(num: $dLat / 2) +
        cos(num: deg2rad(num: $lat1)) * cos(num: deg2rad(num: $lat2)) *
        sin(num: $dLon / 2) * sin(num: $dLon / 2);
    $c = 2 * atan2(y: sqrt(num: $a), x: sqrt(num: 1 - $a));
    return $earthRadius * $c;
}
```

Code Snippet 4.9: Code for Haversine formula

```
private function createRecords(Teacher $teacher, School $school, int $subjectId, float $distance, ?string $dataset = null): void
{
    Assignment::create(attributes: [
        'teacher_id' => $teacher->teacher_id,
        'school_id' => $school->school_id,
        'subject_id' => $subjectId,
        'distance' => $distance,
        'status' => 'Assigned',
        'assigned_at' => now(),
        'dataset' => $dataset
    ]);
}
```

Code Snippet 4.10: Code for create assignment records

```
private function findMatchSchools($subjectPreference, $schools, $checkCategory = false, $teacherSubjectCategory = null, $dataset = null): mi
{
    $subjectCategories = Subject::pluck(column: 'subject_category', key: 'subject_name');

    return $schools->filter(function ($school) use ($subjectPreference, $checkCategory, $teacherSubjectCategory, $subjectCategories, $dataset) {
        foreach ($school->demands as $demand) {
            $schoolSubjectCategory = $subjectCategories[$demand['subject']] ?? null;
            $subjectId = Subject::where(column: 'subject_name', operator: $demand['subject'])->value(column: 'subject_id');

            // Check remaining capacity for this subject in the current dataset
            $assignedCount = Assignment::where(column: 'school_id', operator: $school->school_id)
                ->where(column: 'subject_id', operator: $subjectId)
                ->where(column: 'status', operator: 'Assigned')
                ->where(column: 'dataset', operator: $dataset)
                ->count();

            $hasCapacity = $demand['capacity'] > $assignedCount;

            if ($hasCapacity) {
                if (!$checkCategory) {
                    if ($demand['subject'] === $subjectPreference) {
                        return true;
                    }
                } else {
                    if ($schoolSubjectCategory === $teacherSubjectCategory) {
                        return true;
                    }
                }
            }
        }
        return false;
    });
}
```

Code Snippet 4.11: Code for finding matched schools

```

private function findNearestSchool($teacher, $schools, $strict = true): array|null
{
    $nearestSchool = null;
    $minDistance = PHP_INT_MAX;
    $defaultMinMoreDistance = config(key: 'app.default_min_more_distance', default: 40);

    if (str_contains(haystack: $teacher->location_preference, needle: 'or more')) {
        $minPreferredDistance = $defaultMinMoreDistance;
        $maxPreferredDistance = PHP_INT_MAX;
    } else {
        $range = explode(separator: '-', string: str_replace(search: 'km', replace: '', subject: $teacher->location_preference));
        if (count(value: $range) === 2 && is_numeric(value: trim(string: $range[0])) && is_numeric(value: trim(string: $range[1]))) {
            $minPreferredDistance = (float) trim(string: $range[0]);
            $maxPreferredDistance = (float) trim(string: $range[1]);
        } else {
            return null;
        }
    }

    foreach ($schools as $school) {
        $distance = $this->calculateDistance(
            lat1: $teacher->latitude,
            lon1: $teacher->longitude,
            lat2: $school->latitude,
            lon2: $school->longitude
        );

        if ($distance >= $minPreferredDistance && $distance <= $maxPreferredDistance && $distance < $minDistance) {
            $minDistance = $distance;
            $nearestSchool = $school;
            break;
        }
    }
}

```

Code Snippet 4.12: Code for finding nearest schools

```

if (!$nearestSchool && !$strict) {
    foreach ($schools as $school) {
        $distance = $this->calculateDistance(
            lat1: $teacher->latitude,
            lon1: $teacher->longitude,
            lat2: $school->latitude,
            lon2: $school->longitude
        );

        if ($distance < $minDistance) {
            $minDistance = $distance;
            $nearestSchool = $school;
        }
    }
}

return $nearestSchool ? [
    'school' => $nearestSchool,
    'distance' => $minDistance
] : null;
}

```

Code Snippet 4.13: Code for finding nearest schools

4.6 Conclusion

In short, this chapter outlined the software requirements and specification that required for running the assignment like the data for teachers, schools and subjects. A greedy algorithm also being implemented for that assigns teachers to schools based on subject preferences and location constraints. The implementation of the algorithm is then explained step-by-step with illustration figure and code snippet that show how the algorithm flow. The implemented algorithm has successfully assigned teacher to school based on teachers' subject preferences and location preferences. Lastly, the assignment result will be evaluated in the next chapter.

CHAPTER 5: EVALUATION

5.1 Introduction

This chapter will evaluate and discuss about the performance of the simulation and the assignment result generated. The main concern of the simulation is the effectiveness of the assignment algorithm in obtaining the optimal assignment. Different scenarios will be carried out to examine the effectiveness of the assignment algorithm. The parameters that set in Chapter 1 are used for running the simulation include the number of schools which fixed at 20, the number of teacher (50 teacher) and the number of subjects is 20. Each school has three subject demands with capacity 1-3, and each teacher has randomized subject and location preferences. The teacher and subject data are show in table while the school data is show in a map. Below shows how the data for each parameter being display in the simulation.

The screenshot shows the 'Teacher' data interface. On the left is a form with fields for Name, Address, Subject Preference (a dropdown), and Location Preference (a dropdown). Below the form are 'Save' and 'Edit' buttons. Further down is a section for uploading a CSV file with 'Upload CSV File', 'Choose File', and 'Upload' buttons. On the right is a table of teachers with columns: ID, NAME, SUBJECT PREFERENCE, LOCATION PREFERENCE, and ACTION. The table contains 9 rows of data, each with a 'Delete' button in the ACTION column.

ID	NAME	SUBJECT PREFERENCE	LOCATION PREFERENCE	ACTION
1	Heather Ng	Pendidikan Moral, Sejarah, Bah...	40km or more	Delete
2	Regina Chan	Biology, Accounting, Physics,...	20-30km	Delete
3	Donald bin Azlan	Pendidikan Islam, Economics, E...	10-20km	Delete
4	Ryan Lau	Pendidikan Islam, Sejarah, Acc...	40km or more	Delete
5	Ashley anak Ridzuan	Business Studies, ICT, Physics...	40km or more	Delete
6	Maria Bala	Pendidikan Islam, Economics, B...	10-20km	Delete
7	Joe Teo	Pendidikan Jasmani & Kesihatan...	30-40km	Delete
8	Barbara Ng	Bahasa Melayu, Pendidikan Isla...	40km or more	Delete
9	Carlos Lim	Business Studies, Accounting,...	30-40km	Delete

Figure 5.1: Teacher Data Interface

The screenshot shows the 'Subjects' data interface. On the left is a form with fields for Subject Name, Subject Priority (a dropdown), and Subject Category (a dropdown). Below the form is a 'Save' button. Further down is a section for uploading a CSV file with 'Upload CSV File', 'Choose File', and 'Upload' buttons. On the right is a table of subjects with columns: ID, SUBJECT NAME, SUBJECT PRIORITY, SUBJECT CATEGORY, and ACTION. The table contains 10 rows of data, each with a 'Delete' button in the ACTION column.

ID	SUBJECT NAME	SUBJECT PRIORITY	SUBJECT CATEGORY	ACTION
1	Bahasa Melayu	High	Language	Delete
2	English	High	Language	Delete
3	Mathematics	High	Core Academic	Delete
4	Science	High	Core Academic	Delete
5	Sejarah	High	Core Academic	Delete
6	Additional Mathematics	High	Core Academic	Delete
7	Physics	High	Science	Delete
8	Chemistry	High	Science	Delete
9	Biology	High	Science	Delete
10	Pendidikan Moral	Low	Ethics & Social Studies	Delete

Figure 5.2: Subject Data Interface

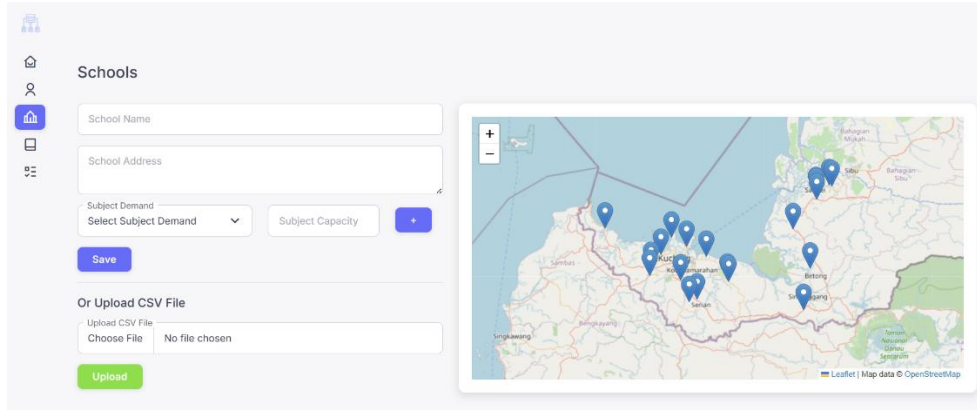


Figure 5.3: School Data Interface

5.2 Simulated Data

To examine the effectiveness of the assignment algorithm, different scenarios of the teacher-school assignment will be carried out to test the ability of the algorithm to handle different conditions. The parameter such as number of schools is fixed for all the three scenarios with 20 schools in Sarawak. Below shows three different scenario that will be carried out and the data used for each scenario.

Scenario 1: Balanced Teacher Supply with Moderate Subject Pool

The data used for Scenario 1 are 20 schools, 50 teachers and 20 subjects. There will be 10 datasets used which each with randomized subject and location preferences. For the 20 schools, each school will have 3 subject demands which the subject capacity of each subject is ranging from 1 to 3. For the 50 teachers, each teacher will have 5 subject preferences.

Scenario 2: Teacher Shortages

The data used for Scenario 2 are 20 schools, 15 teachers and 20 subjects. There will be 10 datasets used which each with randomized subject and location preferences. For the 20 schools, each school will have 3 subject demands which the subject capacity of each subject is ranging from 1 to 3. For the 15 teachers, each teacher will have 3 subject preferences.

Scenario 3: Large Teacher Supply and Subjects

The data used for Scenario 3 are 20 schools, 80 teachers, 20 subjects and 30 subjects. There will be 10 datasets used which each with randomized subject and location preferences. For the 20 schools, each school will have 3 subject demands which the subject capacity of each subject is ranging from 1 to 4. For the 80 teachers, each teacher will have 5 subject preferences.

5.3 Results

5.3.1 Scenario 1

This section will assess how well the greedy algorithm assigns teachers to schools which concentrating on its ability to optimise assignments while meeting constraints in Scenario 1. Scenario 1 illustrates a balanced teacher supply for a moderate subject pool. The key metrics used to analyse the effectiveness of the proposed assignment algorithm includes the total number of assigned teachers, total number of unassigned teachers, percentage of assignments within location preferences, percentage of subject preferences matches and the school with no assigned teachers. Below shows analysis of the assignment result generated by the proposed algorithm based on Scenario 1.

Table 5.1: Assignment Effectiveness Across Datasets for Scenario 1

Datasets	Assigned Teacher	Unassigned Teacher	Location Preferences (%)	Subject Preferences (%)	School with No Assigned Teachers
Dataset_1	50	0	60	82	1
Dataset_2	50	0	54	76	2
Dataset_3	50	0	64	97	2
Dataset_4	50	0	58	76	2
Dataset_5	50	0	62	82	4
Dataset_6	50	0	64	78	3
Dataset_7	50	0	62	84	4

Dataset_8	50	0	74	78	1
Dataset_9	50	0	68	78	1
Dataset_10	50	0	62	80	1

Figures below show the interface of the analysis.

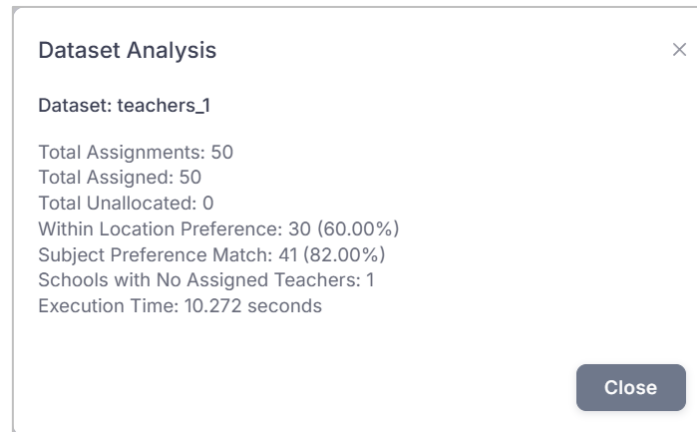


Figure 5.4: Analysis Interface

Table 5.1 presents a complete evaluation of the effectiveness of the proposed assignment algorithm over 10 datasets which each having 50 teachers and randomized subject and location preferences. Based on the table, all datasets obtained a 100% assignment rate with 50 teachers allocated and none unassigned. This illustrates the algorithm's efficiency in guaranteeing complete teacher allocation which likely because of the fallback mechanism of category-based assignments in Phase 3. The algorithm will try to assign the teacher who failed in the first two phases, high-priority subjects and low-priority subjects assignment using the fallback mechanism of category-based assignments which allow the teacher to be assigned to the subject that has the same subject category in their subject preferences. Thus, this fallback mechanism will ensure the proposed algorithm to obtain optimal assignment.

The main concern that may affect the effectiveness of the proposed assignment algorithm is the teachers' location preferences. There are few options that can be chosen for the location preferences which including 5-10km, 10-20km, 20-30km, 30-40km and 40km or more.

Based on the analysis from the 10 datasets, the percentage of assignment within teachers' location preferences is ranging from 54% in dataset_2 to 74% in dataset_8. The lower percentage in dataset_2 is because of the stricter location preferences like 5-10km or 10-20km were often unmet and the fallback mechanism will relax the constraints to ensure assignments. The algorithm will try to assign the teacher to the second nearest school that met the subject preferences. On the other hand, the high percentage in dataset_8 indicates that it has more compatible location preferences which the dataset has a greater number of assignments within teachers' location preferences. Thus, the teachers' location preferences will likely influence the assignment to reach optimal.

Furthermore, teachers' subject preferences will also affect the effectiveness of the proposed assignment algorithm. The percentage of subject preference matches assesses the efficacy of the assignment algorithm in matching teachers assigned subjects with their stated preferences. From the 10 datasets, the percentage of subject preference matches varied from 76% in dataset_2 and dataset_4 to a high of 97% in dataset_3. High match percentage indicate that the algorithm has successfully found the exact matches between teacher's subject preferences and school demands in early phases, Phase 1 and Phases 2. For example, dataset_3 achieved a 97% match percentage which indicating an optimal distribution of teacher preferences linked with school demands. It contains more exact subject preferences that aligned with the school high-priority subject demands which may increase the percentage of subject preference matches.

Lastly, the number of schools with no assigned teachers is another concern on the effectiveness of the assignment algorithm. Based on Table 5.1, the number of schools with no assigned teachers is ranging from 1 in dataset_1, dataset_8, dataset_9 and dataset_10 to 4 in dataset_5 and dataset_7. This behaviour aligns with the greedy algorithm's local optimisation approach which prioritises allocating each teacher to the nearest suitable school rather than

revisiting previous assignments to balance school distribution. Although there will be some schools have no assigned teacher but each teacher has been assigned to a school based on the proposed assignment algorithm.

5.3.2 Scenario 2

This section will assess how well the greedy algorithm assigns teachers to schools which concentrating on its ability to optimise assignments while meeting constraints in Scenario 2. Scenario 2 illustrates the teacher shortage with high subject diversity where 15 teachers and 20 subjects are used. The key metrics used to analyse the effectiveness of the proposed assignment algorithm includes the total number of assigned teachers, total number of unassigned teachers, percentage of assignments within location preferences, percentage of subject preferences matches and the school with no assigned teachers. Below shows analysis of the assignment result generated by the proposed algorithm based on Scenario 2.

Table 5.2: Assignment Effectiveness Across Datasets for Scenario 2

Datasets	Assigned Teacher	Unassigned Teacher	Location Preferences (%)	Subject Preferences (%)	School with No Assigned Teachers
Dataset_1	15	0	66.67	66.67	9
Dataset_2	15	0	60.00	86.67	10
Dataset_3	15	0	80.00	86.67	12
Dataset_4	15	0	60.00	93.33	10
Dataset_5	15	0	71.43	71.43	10
Dataset_6	15	0	66.67	80.00	9
Dataset_7	15	0	66.67	73.33	10
Dataset_8	15	0	60.00	100.00	11
Dataset_9	15	0	60.00	86.67	8

Dataset_10	15	0	86.67	100.00	11
------------	----	---	-------	--------	----

Table 5.2 presents a complete evaluation of the effectiveness of the proposed assignment algorithm over 10 datasets which each having 15 teachers with randomized subject and location preferences. Based on the table, all datasets obtained a 100% assignment rate with 15 teachers allocated and none unassigned. The excess of available slots around 120 slots where each school has 3 subject demands with average of 2 subject capacity ensures that all 15 teachers can be assigned.

The primary factor that may affect the effectiveness of the proposed assignment algorithm is the teachers' location preferences. Based on the analysis from the 10 datasets, the percentage of assignment within teachers' location preferences is ranging from 60% in dataset_2, dataset_4, dataset_8 and dataset_9 to 86.67% in dataset_10. The 60% of assignment within teachers' location preferences means that there 9 out of 15 teachers that are assigned with their preferred ranges. The other may affected by the strict location preferences such as 5-10km that the algorithm cannot found the school that has the subject preferences within the range. This caused the algorithm to expand more wider ranges to ensure that the teacher is assigned. The 86.67% in dataset_10 shows that it had better alignment between teacher locations and nearby schools. Thus, the teachers' location preferences will likely influence the assignment to reach optimal.

Furthermore, teachers' subject preferences will also affect the effectiveness of the proposed assignment algorithm. The percentage of subject preference matches assesses the efficacy of the assignment algorithm in matching teachers assigned subjects with their preferred preferences. From the 10 datasets, the percentage of subject preference matches varied from 66.67% in dataset_1 to 100% in dataset_8 and dataset_10. All 15 teachers in dataset_8 and dataset_10 have been assigned to the school that perfect matches with their subject preferences.

This reflects that the algorithm has been executed correctly which assigning teachers in Phase 1 and 2 without using the fallback mechanism, Phase 3. However, the low percentage in dataset_1 shows that there are some teacher's subject preferences that do not aligned with available school's subject demand which causes the algorithm requires Phase 3 to assign teachers to schools based on the same subject category.

Finally, the number of schools with no assigned teachers is another concern on the effectiveness of the assignment algorithm. Based on Table 5.2, the number of schools with no assigned teachers is ranging from 8 in dataset_9 to 12 in dataset_3. The limited subject preferences with around 45 preferences from 15 teachers which each with 3 preferences compared to the excess school's subject demands with around 120 slots will cause a high number of schools with no assigned teachers. For example, there is still 12 schools with no assigned teachers in dataset_3 but all 15 teachers were assigned to schools with high location and subject preferences, 80% and 86.67%. This reflects that the 15 teachers each with 3 preferences are not enough to cover up the 20 schools which each with 3 subject demands.

5.3.3 Scenario 3

This section will assess how well the greedy algorithm assigns teachers to schools which concentrating on its ability to optimise assignments while meeting constraints in Scenario 3. Scenario 3 demonstrates the large teacher supply and subjects for an assignment problem in real world. The data used to simulate the scenario include 80 teachers with 20 subjects and 80 teachers with 30 subjects. The key metrics used to analyse the effectiveness of the proposed assignment algorithm includes the total number of assigned teachers, total number of unassigned teachers, percentage of assignments within location preferences, percentage of subject preferences matches and the school with no assigned teachers. Below shows analysis of the assignment result generated by the proposed algorithm based on Scenario 3 with 20 subjects.

Table 5.3: Assignment Effectiveness Across Datasets for Scenario 3 with 20 subjects

Datasets	Assigned Teacher	Unassigned Teacher	Location Preferences (%)	Subject Preferences (%)	School with No Assigned Teachers
Dataset_1	80	0	66.25	83.75	0
Dataset_2	80	0	65.00	85.00	0
Dataset_3	80	0	51.25	73.75	0
Dataset_4	80	0	60.00	76.25	1
Dataset_5	80	0	48.75	80.00	0
Dataset_6	80	0	52.50	81.25	0
Dataset_7	80	0	53.75	76.25	0
Dataset_8	80	0	61.25	82.50	0
Dataset_9	80	0	56.25	85.00	0
Dataset_10	80	0	61.25	85.00	0

Table 5.3 presents a complete evaluation of the effectiveness of the proposed assignment algorithm over 10 datasets which each having 80 teachers and randomized subject preferences from 20 subjects and location preferences. Based on the table, all 80 teachers in each dataset had been assigned and none unassigned. The 120 slots of school's subject demands provide more than enough capacity to assign all the 80 teachers which there may still have some available slots after assignment done. Each teacher with 5 subject preferences increases the match opportunities which ensuring all the teachers find a slot.

The main factor that may affect the effectiveness of the proposed assignment algorithm is the teachers' location preferences. Based on the analysis from the 10 datasets, the percentage of assignment within teachers' location preferences is ranging from 48.75% in dataset_5 to 66.25% in dataset_1. The low percentage in dataset_5 is because of stricter location preferences

like 5-10km which causing the algorithm to execute Phase 3 to lose the location constraints. For instance, there are 17 teachers with location preferences of 5-10km which causes only 39 teachers that the algorithm found the school matched with the teachers' preferred ranges and 41 teachers are assigned to school by the algorithm under Phase 3 where the schools that assigned to them are not in their preferred ranges.

Furthermore, teachers' subject preferences will also affect the effectiveness of the proposed assignment algorithm. From the 10 datasets, the percentage of subject preference matches varied from 73.75% in dataset_3 to 85.00% in dataset_2, dataset_9 and dataset_10. The low percentage of subject preference matches in dataset_3 shows that the 80 teachers were competing for the 120 available slots which increases the mismatch risks and lower down the subject matches. It also reflects there are 21 teachers out of 80 teachers need Phase 3 so that they can be assigned to a school.

Lastly, the number of schools with no assigned teachers is another concern on the effectiveness of the assignment algorithm. Based on Table 5.4, there is only dataset_4 with one school with no assigned teachers. The ratio of 80 teachers to 20 schools each with 5 subject preferences ensures that there is possibility for most of the schools to get assigned with teachers. For example, all the teachers in dataset_5 are assigned to a school although the location preferences matches are lower. However, some of the school that with no assigned teacher is due to early exhaustion of teachers. For example, all 80 teachers in dataset_4 have been assigned to 19 schools and leaving 1 school with no assigned teachers.

Below shows analysis of the assignment result generated by the proposed algorithm based on Scenario 3 with 30 subjects.

Table 5.4: Assignment Effectiveness Across Datasets for Scenario 3 with 30 subjects

Datasets	Assigned Teacher	Unassigned Teacher	Location Preferences (%)	Subject Preferences (%)	School with No Assigned Teachers
Dataset_1	80	0	67.50	75.00	0
Dataset_2	80	0	61.25	73.75	0
Dataset_3	80	0	65.00	73.75	0
Dataset_4	80	0	66.25	72.50	0
Dataset_5	80	0	51.25	63.75	1
Dataset_6	80	0	62.50	73.75	0
Dataset_7	80	0	58.75	70.00	0
Dataset_8	80	0	48.75	62.50	0
Dataset_9	80	0	55.50	73.75	0
Dataset_10	80	0	66.25	72.50	0

Table 5.4 presents a complete evaluation of the effectiveness of the proposed assignment algorithm over 10 datasets which each having 80 teachers and randomized subject preferences from 30 subjects and location preferences. Based on the table, all 80 teachers in each dataset had been assigned and none unassigned. The 120 slots of school's subject demands provide more than enough capacity to assign all the 80 teachers which there may still have some available slots after assignment done. The three phases in the algorithm also ensure that the assignment to reach optimal assignment.

The main factor that may affect the effectiveness of the proposed assignment algorithm is the teachers' location preferences. Based on the analysis from the 10 datasets, the percentage of assignment within teachers' location preferences is ranging from 48.75% in dataset_8 to 67.50% in dataset_1. The low percentage in dataset_8 indicates that the algorithm was

struggled in finding matches within the teachers' preferred ranges in Phase 1 and 2 and need the use of Phase 3 to loosen the constraints to ensure the assignment. For example, there are only 39 teachers that the algorithm found the school matched with the teachers' preferred ranges and 41 teachers are assigned to school by the algorithm under Phase 3 where the schools that assigned to them are not in their preferred ranges.

Furthermore, teachers' subject preferences will also affect the effectiveness of the proposed assignment algorithm. From the 10 datasets, the percentage of subject preference matches varied from 62.50% in dataset_8 to 75% in dataset_1. The low percentage of subject preference matches in dataset_8 shows that the 80 teachers were competing for the 120 available slots which increases the mismatch risks and lower down the subject matches. It also reflects that most of the teachers' subject preferences in dataset_8 are not aligned with the school subject demands during the assignment which only 50 teachers have been assigned to school with their preferred subjects.

Lastly, the number of schools with no assigned teachers is another concern on the effectiveness of the assignment algorithm. Based on Table 5.4, the number of schools with no assigned teachers is ranging from 0 to 1. The ratio of 80 teachers to 20 schools each with 5 subject preferences ensures that there is possibility for most of the schools to get assigned with teachers. For example, all the teachers in dataset_8 are assigned to a school although the location and subject preferences matches are lower. However, some of the school that with no assigned teacher is due to early exhaustion of teachers. For example, all 80 teachers in dataset_5 have been assigned to 19 schools and leaving 1 school with no assigned teachers.

5.4 Discussion

The greedy approach that implemented in the Teacher-School Assignment Algorithm shows its strengths and limits in Scenarios 1, 2, and 3 where each with different teacher counts, subject numbers, and preference demands. The key advantage lies on its efficiency in obtaining high assignment rates as the approach focus on prioritizes local optimal solution in Phase 1 and 2 before initiating a Phase 3 fallback. This benefit Scenario 1 where the excess slot capacity with moderate teacher number for efficient allocation of teachers. In Scenario 2, the algorithm's capability to optimise limited preferences ensures complete allocation even with a low teacher-to-school ratio. Besides, Scenario 3 with greater preference choices and slot availability also boost by the algorithm's efficiency in guaranteeing that all teachers are assigned.

However, the greedy algorithm's local optimisation method presents significant limitations that vary with each scenario especially Scenario 2. Its tendency to prioritise immediate best matches frequently results in suboptimal global consequences, such as inconsistent school coverage. It should be perfect matches for the subject and location preferences in Scenario 2 with the excess slot availability but because of the algorithm's inability to revisit earlier decisions causing it cannot be achieved. Furthermore, the algorithm's sensitivity to strict constraints which affects preference satisfaction differently across scenarios. This drawback mainly affected Scenario 1 and 3 where it lowers the location preference accuracy for both scenarios. Overall, the algorithm succeeds at assignment efficiency but it needed changes such as wider preferences to improve equality and satisfaction.

5.5 Conclusion

This chapter's evaluation showed the effectiveness of the proposed greedy-based teacher-school assignment algorithm in three simulated scenarios like balanced teacher supply, teacher shortage, and extended subject pool. In every scenario, the approach had a 100% assignment rate which demonstrating its strength and flexibility under a number of constraints. The implementation of subject category-based fallback matching was essential in optimising assignments in the absence of exact matches. Subject preference alignment remained strong in limited-subject scenarios but decreased significantly as the subject range expanded. Strict location preference reduced the percentages of assignment within teacher's preferred distance. In short, the results prove the effectiveness of the algorithm in assigning teachers fairly.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Introduction

This chapter provides the final thoughts on the development and implementation of the Teacher-School Assignment Simulation. Besides, suggestions for future enhancements and research directions were outlined in the end of the project.

6.2 Conclusion

In conclusion, a Teacher-School Assignment Algorithm has been successfully developed throughout this project. The assignment algorithm implemented with Greedy algorithm has showed efficient for fast decision-making and produced excellent outcomes in assigning a large number of teachers to right schools. Through various simulation with different scenarios, the algorithm has obtained high assignment rates by successfully balancing teacher preferences and school demands. The algorithm was also able to reduce the number of unassigned teachers in the rural areas. The three phase that implemented in the algorithm which prioritize the exact subject matches before using the fallback mechanism has proved to be efficient for multiple constraints like vary teacher-to-school ratio and subject. The results showed that the increase of the number of teachers enhanced school converge. Moreover, high number of subject preferences will increase the subject preference matches which allow the teachers to be assigned to their preferred schools. In short, the implementation of the assignment algorithm was essential for automating the teacher assignments which can streamlines the teacher assignment process.

6.3 Future Works

There are several enhancements can be done for future research and development to enhance the algorithm's efficiency and accuracy. First, implementing dynamic demand changes based on real-time teacher preference information should improve subject and location matching while also resolving mismatches discovered in high-competition scenarios. For

example, the algorithm can dynamically adjust the assignment strategy based on the different importance levels of the teacher preferences.

The second enhancement could include the integration of more advanced optimization approaches Genetic Algorithms or Simulated Annealing which could help to solve the Greedy algorithm's constraint of generating only locally optimal solutions. These approaches may provide greater global solutions by considering a greater range of allocation choices particularly when balancing several competing challenges.

Lastly, the accuracy of the geographical data also one of the enhancements that can be considered for future development. This is because the accuracy of the geographical data will affect the effectiveness of the algorithm in making sure the location of the teachers and schools are correctly stated. Thus, integrating real-time geographical data using paid map services like Google Maps compared to free open-source map services like OpenStreetMap which will allow the algorithm to calculate the distances between teachers and schools more accurately.

References

- Ali, A.Y. (2021). Greedy algorithm for solving student allocation problem in internship program: a case study. *International journal of research in industrial engineering*, 10(2), 155-164.
- Alipour, M., Zare Mehrjdrdi, Y., & Mostafaeipour, A. (2020). A rule-based heuristic algorithm for on-line order batching and scheduling in an order picking warehouse with multiple pickers. *RAIRO - Operations Research*, 54(1), 101–107.
<https://doi.org/10.1051/ro/2018069>
- Chen, D., & Xu, F. (2024). Design and Implementation of Machine Learning Algorithms in Automatic Grading of Students' Assignments. *J. Electrical Systems*, 20(3s), 899–919.
<https://doi.org/10.52783/jes.1388>
- Chen, F., Chen, N., Mao, H., & Hu, H. (2019). The Application of Bipartite Matching in Assignment Problem. *ArXiv (Cornell University)*.
<https://doi.org/10.48550/arxiv.1902.00256>
- Dillon, E., & Malick, S. (2020). *Teacher turnover and access to effective teachers in the School District of Philadelphia* (REL 2020–037). U.S. Department of Education, Institute of Education Sciences, National Center for Education Evaluation and Regional Assistance, Regional Educational Laboratory Mid-Atlantic. <http://ies.ed.gov/ncee/edlabs>.
- Kraft, M. A., Brunner, E. J., Dougherty, S. M., & Schwegman, D. J. (2020). Teacher accountability reforms and the supply and quality of new teachers. *Journal of Public Economics*, 188, 104212. <https://doi.org/10.1016/j.jpubeco.2020.104212>
- Luschei, T. F., & Chudgar, A. (2015). Evolution of policies on teacher deployment to disadvantaged areas. *Background Paper Prepared for the Education for All Global Monitoring Report 2015, Education for All 2000-2015: Achievements and Challenges*.

- OECD (2019). Education at a Glance 2019: OECD Indicators, OECD Publishing, Paris,
<https://doi.org/10.1787/f8d7880d-en>.
- Senarath, U. (2021). Waterfall Methodology, Prototyping and Agile Development. *Waterfall Methodology, Prototyping and Agile Development*.
<https://doi.org/10.13140/RG.2.2.17918.72001>
- UNESCO. (2021). *Global Education Monitoring Report*. UNESCO Publishing.
- Waseem, A. (2023). *Waterfall Methodology: History, Principles, Stages & More*. Management Library. <https://management.org/waterfall-methodology>