Optimal Scheduling of Engineering Maintenance Work

A Report Submitted

In Partial Fulfilment of the Requirements

For the Degree of

Master of Science

in

Artificial Intelligence with Business Strategy

by

Varughese Varughese Chakkittadathu [210195482]

to the

College of Engineering and Physical Sciences and Aston Business School

Aston University

Birmingham, UK

September 2022

1. Abstract

The primary focus of this project involves the scheduling of both planned and reactive maintenance work for a facilities management company called Arcus FM. This MSc dissertation is related to the Knowledge Transfer Partnership (KTP) hosted by the University (supervised by Dr Felipe Campelo and Dr Anikó Ekárt). The aim of this dissertation is to improve the productivity of the partnering company by proposing modifications to schedules previously created (within the KTP project) when unexpected disruptions to the schedule occur, such as new jobs being requested. A suite of local search algorithms is proposed to be investigated so that the effects of such disruptions are minimized. The project also provides an opportunity to gain an insight about computational intelligence methods, the scheduling problem as well as the travelling salesman problem and their implementation in a real-life scenario. A scheduling algorithm was designed using these concepts as its foundation and tested first using synthetic data and then on real historical data which was obtained following an in-person requirements gathering interview with a KTP associate which was conducted after getting the necessary ethical approval from the university. The results from these tests were analysed for gauging the reliability and usability of the proposed scheduling algorithm. This was followed by the submission of a report to the KTP associate which contained evaluation data and recommendations for the partnering company based on the results. Positive comments were made regarding the study, and the insights appeared beneficial for the KTP project's future development.

2. Acknowledgements

Words cannot express how grateful I am to Dr Anikó Ekárt and Dr Felipe Campelo from the College of Engineering and Physical Sciences and Dr Ali Radfard from the Aston Business School for their invaluable supervision, support, and tutelage during the course of my MSc degree. They have profoundly influenced me with their passion, authenticity, and commitment. This endeavour would not have been possible without the support and feedback from Dr Arezoo Vejdanparast to whom I extend my deepest gratitude. Last but not least, I would like to sincerely thank my family and colleagues for their unconditional support. I would also like to convey my appreciation to Aston University for giving me this wonderful opportunity to develop both academically and personally.

CONTENTS

3. Introduction

The primary focus of this dissertation project is finding a solution for a complex optimisation problem involving the scheduling of both planned and reactive engineering maintenance work for the facilities management company, Arcus FM, thereby improving overall productivity and distribution of workload for the company's employees in a manner that accounts for unexpected jobs or emergencies.

Arcus FM is an award-winning facilities management company which has been active in the FM business for over a decade. It is a company driven by over 4,500 employees and offers reactive and maintenance services such as mechanical, electrical, HVAC, refrigeration, drainage etc. which is available twenty-four seven, the entire year. They have a diverse client base which includes household names such as Sainsbury's, Coop, Boots, Travelodge, and EVRI, and offers them a broad selection of hard and soft FM services that can be blended and packaged to meet the various demands of each organisation. The company also has a keen interest in keeping up with the latest developments in technology and augmenting them into their various services. Using the Knowledge Transfer Partnership with Aston University, the company aims to further enhance the services that they currently offer, and this dissertation project intends to contribute to the development of a job scheduler that is being designed for the company.

The dissertation report is divided into five main sections, which begins with a requirement analysis to clearly define the objectives of the dissertation, followed by a literature review of the various theories, concepts and tools required to realize the identified objectives. A methodology section is dedicated to describing the overall design of the solution along with its implementation. This is followed by an evaluation of the proposed solution which gauges its reliability and usability. Finally, a project management section provides details on how the project was executed and briefly elaborates the original plan and how it turned out in the end.

4. Requirement Analysis

The project aims to optimize the allocation of jobs for the company when disruptions occur in their schedule by making modifications to the existing schedule, which may result in the rescheduling of existing job allocations or the addition of new jobs to the schedule.

Requirement Analysis is a necessary process that is used to understand the needs and expectations of the proposed solution for a problem. Hass et al. (2007) emphasises that the first step for a requirement analysis is identifying the key stakeholders and end users of the project. A Stakeholder Analysis Matrix (Mendelow,1991) is created to chart out all the main stakeholders in the project and the level of influence and interest that each of them possesses:

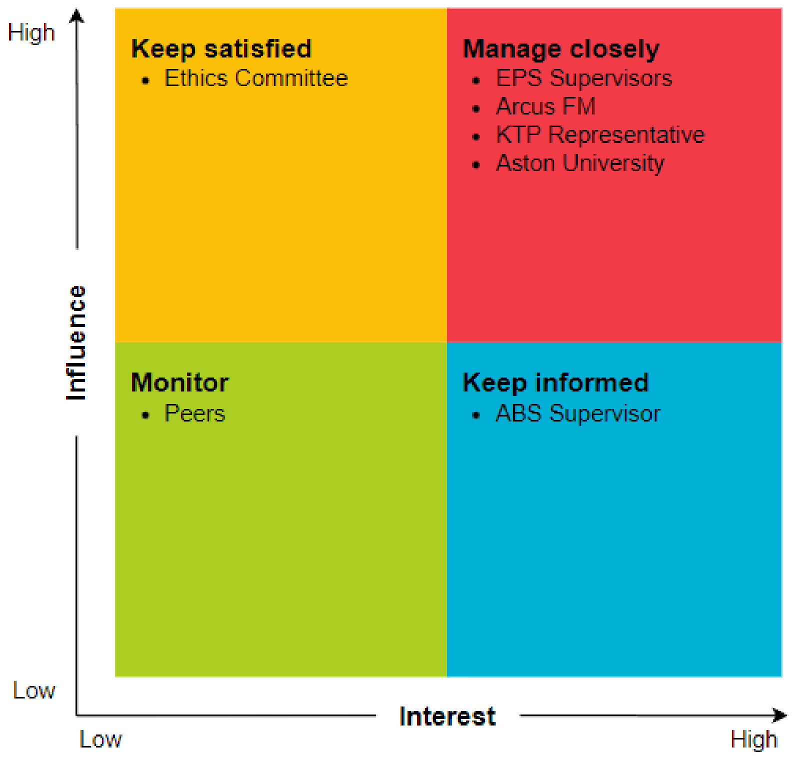


Figure 1: Stakeholder Analysis Matrix

After analysing the Mendelow Matrix, it is evident that the primary requirements come from the facilities management company, Arcus FM, making it an industrial project, but at the same time, since the project also involves the investigation of computational intelligence theories to aid with the development of the final product, it can be labelled as a research project as well.

An example scenario has been created to showcase why the problem is important and identify potential commercial and economic applications both at a high level and in detail. The example can also help in logically deriving the requirements of the project and clearly define what the final product will be.

This example scenario attempts to emulate the day-to-day operations of the facilities management company and extract the various requirements of the dissertation project. All the data of the engineers, jobs and their locations are synthetic.

Consider a facilities management company that has contracted four engineers to perform ten jobs over various locations in the West Midlands. Each job requires a specific skill and a set amount of time to complete. The total amount of time set for a day of work is 8 hours. For the sake of simplicity, only the following four skills would be considered, with each engineer having two skills each: Mechanical, Electrical, HVAC and Drainage.

Let the four Engineers be labelled: {E_1, E_2, E_3, E_4} and the ten jobs be labelled: {J_1, J_2, J_3, J_4, J_5, J_6, J_7, J_8, J_9, J_{10}} each having different skill requirements and processing times. Let the four Engineers have the following skills and are currently at these given locations:

Engineers	Skills	Location
E_1	Mechanical, Electrical	B21 9RJ
E_2	Electrical, HVAC	B68 0LH
E_3	Mechanical, Drainage	B11 1LZ
E_4	Mechanical, HVAC	B25 8RN

Let the ten jobs have the following skill requirements and are at these locations:

Jobs	Skill Required	Location	Time Required
J_1	Electrical	DY11 5SW	120 Mins
J_2	Electrical	WV4 6ED	160 Mins
J_3	Electrical	B79 7PB	120 Mins
J_4	Mechanical	B90 8AT	120 Mins
J_5	Mechanical	B46 1AN	120 Mins
J_6	Mechanical	B98 9EY	120 Mins
J_7	HVAC	NN1 5BD	240 Mins
J_8	Drainage	B96 6BD	120 Mins
J_9	Electrical	WV7 3BW	60 Mins
J_{10}	Mechanical	CV9 1LQ	90 Mins

The map below shows the initial locations of the various engineers as well as the jobs they are assigned to complete for the day:

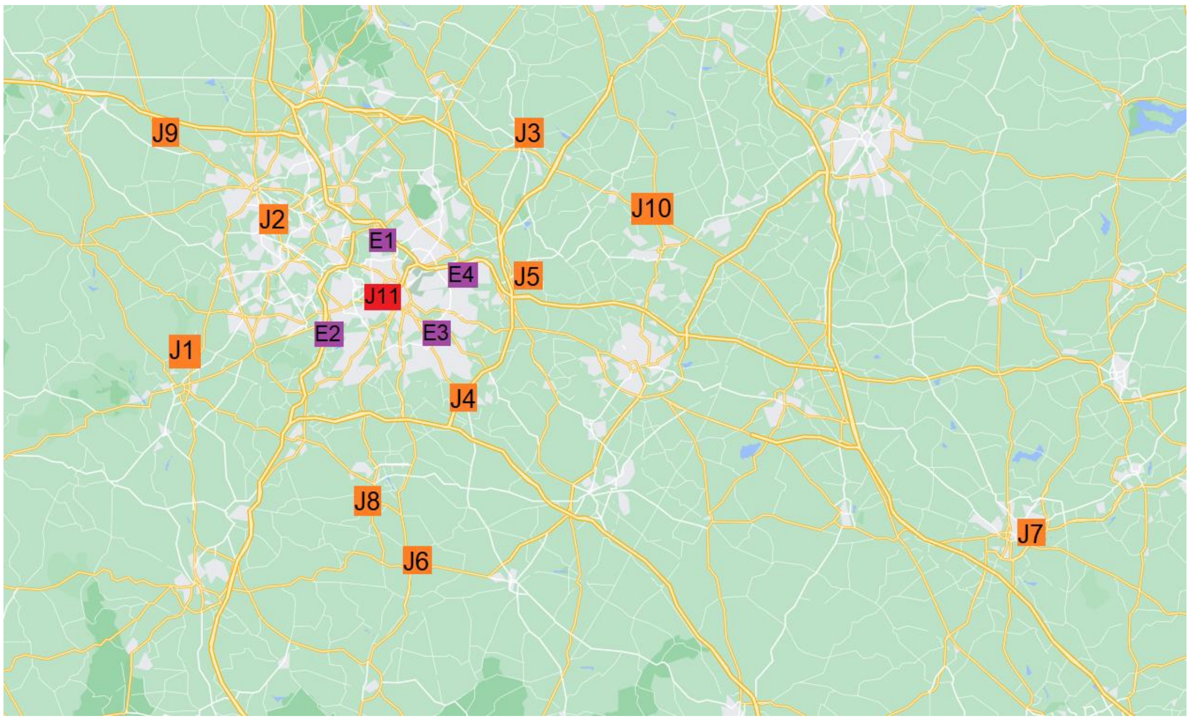


Figure 2: Map of West Midlands showing the location of various facilities and the available engineers

The location labelled as J_{11} has been marked as an unexpected job and will be used as the main component of this example.

Some relevant distance time matrix data for each engineer is as follows:

Engineer 1				Engineer 2			
From	**To**	**Time**	**Distance**	**From**	**To**	**Time**	**Distance**
Home	J_3	32 min	35.1 km	Home	J_2	20 Min	16.3 km
J_3	J_{10}	16 min	16.2 km	J_2	J_9	19 Min	15.8 km
J_{10}	J_5	20 min	19.0 km	J_9	J_1	41 min	38.6 km
J_5	Home	22 min	23.3 km	J_1	Home	27 min	26.6 km
Engineer 3				**Engineer 4**			
From	**To**	**Time**	**Distance**	**From**	**To**	**Time**	**Distance**
Home	J_8	32 min	30.5 km	Home	J_7	62 min	84.9 km
J_8	J_6	12 min	10.8 km	J_7	Home	63 min	84.7 km
J_6	J_4	15 min	17.6 km				
J_4	Home	22 min	13.4 km				

Using the data mentioned above, the following itinerary is created:

Engineers	Jobs Assigned	Total Time Required
E_1	Home -> J_3 -> J_{10} -> J_5	440 Minutes
E_2	Home -> J_2 -> J_9 -> J_1	430 Minutes
E_3	Home -> J_8 -> J_6 -> J_4	430 Minutes
E_4	Home -> J_7	370 Minutes

The schedule for each engineer starts and ends at his or her home location, and the total time required to complete the given schedule is calculated by adding up the total travel time between each location and the time required to complete each job assigned to them.

Now an unexpected job, J_{11}, is introduced and needs to be assigned a qualified engineer:

Jobs	Skill Required	Location	Time Required
J_{11}	HVAC	B16 9HN	120 Mins

The following job can be inserted during two scenarios, one during the beginning of the day as an unscheduled job or another as an emergency job during the schedule. For this example, the latter scenario will be considered.

The current positions of the four engineers are: Engineer E_1 performing job J_3 at address B79 7PB, Engineer E_2 performing job J_2 at address WV4 6ED, Engineer E_3 performing job J_8 at address B96 6BD and Engineer E_4 performing job J_7 at address NN1 5BD.

The engineers eligible for the job are E_2 and E_4, but with the current schedule, the new job cannot be assigned to E_2 since the total time required to complete all the jobs would exceed 8 hours, and in the case of Engineer E_4, the large distance between the jobs would make the proposed schedule not viable.

A possible solution can be found by moving jobs between engineers, which could enable a qualified engineer to take care of the emergency while keeping his or her work time below 8 hours.

After reallocating the jobs in the schedules of each engineer, the following itinerary is created, where the jobs highlighted in blue are reallocated jobs and the job highlighted in red is the job which has come in as an emergency.

Engineers	Jobs Assigned	Total Time Required
E_1	Home -> J_3 -> J_{10} -> J_1	440 Minutes
E_2	Home -> J_2 -> J_{11} -> J_9	430 Minutes
E_3	Home -> J_8 -> J_5 -> J_6	430 Minutes
E_4	Home -> J_7 -> J_4	370 Minutes

The newly created itinerary has enabled the engineers to complete all their assigned tasks successfully and handle the emergency within their allotted work time. This was achieved by taking account of all the engineer's skill levels and current locations and reallocating existing work among them.

Identifying Requirements

According to an IEEE article from Tang, Han, and Chen (2004), every project would have a wide range of requirements that it should fulfil, and they need to be grouped accordingly to avoid any confusion during the later stages of the project. Using the example illustrated above as a reference, the requirements are categorized into three groups:

1) Functional Requirements:
    - Design a solution that can optimize a premade schedule of a facilities management company during the following events:
        - A new job arriving before the beginning of a workday
        - A new job arriving during the workday
2) Technical Requirements:
    - The designed solution should comply with a set of constraints specified by the company:
        - The total time taken by an engineer to complete an assigned schedule should be less than or equal to a set time limit
        - An engineer can be assigned a job only if he or she has the necessary skills to perform it
3) Operational Requirements:
    - The designed solution requires a database of the engineers, the jobs that they will be assigned and the daily schedule.
    - A method to calculate the distance between jobs and engineers
    - A method to search and assign jobs to engineers such that the final schedule generated is viable and optimal

Feasibility

Modelling an optimisation problem for a real-life scenario such as the facilities management company and creating an optimal itinerary involving a lot of complex elements such as the skill level and location of the company’s employees and setup time for jobs along with addressing unexpected jobs or emergencies would be challenging. But this challenge can be tackled by using theories from computational intelligence methods, the scheduling problem, and the travelling salesman problem. But it is worth noting that this is an NP-hard problem where finding the most optimal solutions would be very difficult due to the large number of combinations possible between various job assignments for each engineer. The solution will be designed while considering such parameters and the constraints set by the KTP company.

5. Literature Review

Using the requirement analysis as a reference, the main academic disciplines involved in this dissertation are deduced:

- Operations Research
- Artificial Intelligence
- Business Strategy

The primary objective of this dissertation is the optimal scheduling of engineering maintenance jobs, and this can be achieved using theories in Operations Research that are used to solve problems involving scheduling and constraint satisfaction.

Constraint Satisfaction Problems are researched in both Operations Research and Artificial Intelligence, as many other theories in these disciplines are heavily reliant on this topic. The two other concepts used to find a solution for this complex optimisation problem is the Travelling Salesman Problem and the implementation of Local Search Heuristics, which is part of Artificial Intelligence Research.

When it comes to Business Strategy, the primary themes that can be discussed are Human Resource Management: how organisations schedule and allocate work to their employees and the importance of employees conforming to the work assigned to them and the possible ways artificial intelligence can be applied to these sections of business strategy.

This literature review performs an initial exploration of the various theories mentioned above, followed by a critical analysis of the previous academic and practical works in the relevant areas which would help in the subsequent design and implementation of a solution for the dissertation project.

5.1 The Scheduling Problem

According to Hoos and Stützle (2004), Scheduling Problems can be defined as the mapping of a set of jobs, J: = {J_1, J_2,.., J_n} to a set of machines, M: = {M_1, M_2,.., M_n}, such that they conform to a range of feasibility constraints and optimisation objectives. This is an important class of combinatorial problems where most of the real-life scenarios are NP-hard and difficult to solve, especially when optimal solutions are desired.

From the requirement analysis, it is evident that the Arcus FM scheduling problem comes under the category of scheduling problems called Project Scheduling or the Coordination Problem, where the primary concern is the planning of activities that must adhere to a set of resource constraints such as manpower, money, equipment, etc.) (Bellman, Esogbue, and Nabeshima, 2014).

In this variation of the scheduling problem, for j jobs and m machines or, in this case, engineers, the processing time and travel time to each job are known, each engineer can perform only one job at a time, and the engineers are working in parallel, and the primary objective is to keep the makespan- the total length of the schedule to a minimum or below a set time limit.

To aid in the design of an optimal scheduling algorithm, a variety of previous academic works were consulted:

Since the basic scenario of the problem being tackled is optimising a schedule in response to an unexpected event, an attempt was made to gain inspiration from a Greedy Randomised Adaptive Search Procedure based algorithm by Cildoz, Mallor and Mateo (2021) for solving an emergency room physician scheduling problem. The proposed algorithm included the use of metaheuristics, linear programming and network flow optimisation that searches for good but assorted solutions and scans the solution space where it is most promising. The literature highlighted the effectiveness of local search heuristics in solving complex np-hard problems, and this led to an analysis of a local search algorithm for job shop scheduling problems with sequence-dependent setup times by Choi and Choi (2002). The algorithm was designed to generate scheduling for various manufacturing environments and projects with jobs that have predefined setup times in a multi-machine environment.

The literature also provides an extensive computational study of the proposed local search algorithm in various job shop environments. Further research was conducted on this topic by reviewing a global-local neighbourhood search algorithm and tabu search for a flexible job shop scheduling problem (Serna et al., 2021). The concepts discussed in the literature proved to be well beyond the scope of the dissertation, but useful methods for the intelligent exploration of the neighbourhood structures for local search algorithms were found which had the potential to be integrated into the scheduling algorithm for Arcus FM.

Finally, an academic paper by Mosayebi, Sodhi and Wettergren (2021) on a Travelling Salesman Problem with Job Times provided the required motivation for the design and implementation of the solution that would be proposed. The paper presents a variant of the Travelling Salesman Problem that involves performing various jobs at different locations. The objective was to find a sequence of locations that a worker would need to visit to minimize the maximum completion time of the assigned jobs. The paper also cited the work done by Tang, Miller-Hooks, and Tomastik (2007) on a scheduling problem which involved assigning maintenance engineers throughout a large area and generating an optimized schedule using local search heuristics such as Tabu Search.

The analysis of these academic works led to the conclusion that the scheduling algorithm requires a complex amalgamation of theories such as the constraint satisfaction problem, the travelling salesman problem and local search heuristics to achieve the objectives set by the KTP company. A deeper review of each of these theories is conducted in the coming sections to better integrate these concepts into the scheduling algorithm.

5.2 The Constraint Satisfaction Problem

The process of finding a solution that is a set of values which satisfies all the constraints for a particular set of problem variables is called constraint satisfaction (Rossi, Van Beek, and Walsh, 2006). The success of this project is heavily reliant on the scheduling algorithm meeting the various constraints set by the facilities management company.

The two primary constraints that would need to be passed for a successful schedule to be assigned to an engineer would be:

1) Time: The total time taken for an engineer to complete a schedule should be less than the set time limit.
2) Skill: A job can be assigned to an engineer only if he or she has the necessary skill to perform it.

A paper by De Bruecker et al. (2015) provides in-depth data on skill-related workforce planning problems and develops realistic and applicable mathematical solutions. The literature proposes a variety of techniques ranging from heuristic to exact solution approaches. The literature also discusses the managerial aspects of integrating skills into such scheduling problems.

On the theoretical front, works from Frühwirth and Abdennadher (2003) and Rossi, Van Beek, and Walsh (2006) on constraint programming were consulted, and their theories were integrated well into the designed algorithm, helping it to make appropriate job assignments to qualified engineers.

5.3 The Travelling Salesman Problem

The project, in essence, is an optimisation problem where the primary objective is to create optimal schedules for engineers with minimal total time. One of the best ways to reduce total time would be by minimising the travel time for each engineer by assigning them the right combination of jobs that they are qualified for and performing them in an intelligent order. Theories discussed in the Travelling Salesman Problem can be used to achieve this desired effect.

The aim of the Travelling Salesman problem is to find a route across a collection of locations that can travel through all of them and back to the original starting point in minimal time, or in a more formal sense, it is the process of finding a Hamiltonian circuit with the lowest cost in an edge-weighted graph. William J. Cook’s book, “In Pursuit of the Travelling Salesman” (2015), provided the necessary research material for applying the concept in the scheduling algorithm’s design.

An online guide by Google Developers (2019) presents a possible method of solving the Travelling Salesman Problem along with its implementation, but after conducting due research, it has been concluded that relying on a routing engine to provide a solution would be better over a conventional algorithm.

There were several API solutions available online to choose from, such as the Google Maps API, Amazon’s AWS, Microsoft’s Azure etc., but they all come with a premium price tag and, therefore, are not considered. An online article from gis-op.com (Nolde, 2020) suggests a few open-source routing engines that had the potential to be used in the scheduling algorithm.

Open-source routing engines provide an enhanced level of flexibility to their users by giving them the ability to host their own servers on their own terms with full transparency. They are freely available and can be tweaked according to the user’s preferences. The routing engines considered for the scheduling algorithm all use data from OpenStreetMap (OSM), the world’s biggest open database on the entire planet with over 7.6 billion nodes and 200 million road segments. The next section provides an overview of four open-source routing engines, along with their advantages and disadvantages:

Openrouteservice (ORS):

Openrouteservice is an open-source routing engine which has been operating for over fifteen years. It provides a wide variety of global geospatial services such as providing directions, isochrones, time-distance matrices, geocoding, elevation, and trip optimization. It is written in Java and is easy to set up.

Advantages	Disadvantages
• Has a variety of travel profiles • Offers essential routing services • Has good performance • Provides a moderate level of routing flexibility • Host various other routing solutions	• High RAM requirements • Customisations can be done only in Java • Low level of community involvement and troubleshooting • Not available offline

Graphhopper:

Graphhopper provides open-source routing solutions through its libraries and servers, which are written in Java and can also host a routing API over HTTP. It can be run using a dedicated server, desktops, mobiles, or even single-board computers like Raspberry Pi. It mainly focuses on commercial solutions for highly complex vehicle routing problems, but the engine is available to use by the public as well.

Advantages	Disadvantages
• Provides a variety of travel profiles • Fast route planning and optimization • Scalable architecture • Resource efficient • Easy integration • High community involvement • Responsive troubleshooting • Multiplatform and available offline	• Moderate-High RAM requirements • Custom profiles can be created only using Java source code with custom rules • Some features are closed source such as their matrix API • The free tier offers very little compared to their premium subscription services.

Valhalla:

Valhalla is an open-source routing software with high versatility and scalability, capable of producing routes with high levels of quality. This is made possible using the software's powerful routing engine called THOR (Tiled, Hierarchical Open Routing). Most of the key features in the project have names based on Norse mythology, which made its analysis quite interesting. It is used by Tesla for its in-car navigation systems due to its high efficiency and flexibility.

Advantages	Disadvantages
• Highly versatile travel profiles • Offers high levels of functionality • Highly active development community • Low RAM requirements • Multiplatform and available offline	• Slightly slower at producing solutions • Not easy to create custom profiles • Highly difficult to set up due to poor documentation

Open-Source Routing Machine (OSRM):

OSRM is a high-performance routing engine written purely in C++ and generates the shortest paths possible in road networks in a matter of a few milliseconds, even for routes spanning thousands of kilometres. It is highly customisable and can produce routes based on real-time traffic data. Even though it offers half the functionality of the other routing engines discussed earlier, it closes that gap by using its superior performance and customizability.

Advantages	Disadvantages
• High-Performance Routing • Provides essential travel profiles • Produces results extremely fast • Highly customisable • Easy to set up • Used in several other open-source projects • Available offline	• High RAM requirements • Low activity on feature development • Maintained and developed by a small, dedicated team

After carefully testing and analysing these routing engines, it came down to choosing between Valhalla and OSRM. The decision was made to use OSRM as the scheduling algorithm's routing engine due to its high performance and ease of set-up. Further details on the routing engine are available in the design section, and instructions on how to set up a local server are available in the appendix.

5.4 Local Search Heuristics

A conventional method for searching for a solution from a given search space is through the use of systematic search algorithms, but when it comes to cases such as the Arcus FM scheduling problem, the search spaces that have to be considered are too big for these algorithms and may not produce meaningful results. Implementing local search heuristics would be the best choice for these problems if there is a high possibility that a solution exists. These methods begin with a solution which may or may not be feasible and try to improve that solution locally. The search is complete when a stopping criterion is met but there is no assurance over the quality of the found solution (Poole and Mackworth, 2017).

Google's OR-Tools open-source library (Acrogenesis.com, 2015), David Poole and Alan Mackworth's book on Artificial Intelligence (2017) and a book on Stochastic Local Search by Holger H. Hoos and Thomas Stützle (2004) provide the relevant theory needed to apply local search methods on the scheduling algorithm. The referenced literature provides a deep insight into various local search metaheuristics along with their basic implementation.

Another rich source of research material was found in the documentation for Optaplanner, an open-source constraint solver that is capable of optimizing planning problems (docs.optaplanner.org, 2022). The literature provides research material on local search concepts and how to define a neighbourhood for a given solution, and methods to explore it. Furthermore, the project supervisors, Dr Anikó Ekárt and Dr Felipe Campelo have recommended referring to two articles: "Analysis of Stochastic Local Search Methods for the Unrelated Parallel Machine Scheduling Problem" by Santos et al. (2019) and "Statistical Investigation of Relative Utility of Neighbourhood Structures in Combinatorial Problems" by Maravilha, Mayra-Pereira, and Campelo (2019) and using the neighbourhood structures or operators described in them to explore the solution space. An analysis was conducted on these six neighbourhood structures, and suitable structures were implemented into the final scheduling algorithm for the project.

5.5 Business Strategy

As concluded earlier, the dissertation is an industrial project therefore, components of business strategy must always be considered. The main goal of job scheduling in an organisation is to assign tasks to its workers in a way that fully utilises their contracted hours and complete as many jobs as possible. To make this a reality, an organisation must fully exploit both its managerial prowess as well as the current IT-based solutions at its disposal.

A literature review on staff scheduling by Van den Bergh et al. (2013) was a great starting point for compiling information for the project's business plan. The report gives a general overview of scheduling practices in several industries and notes the current trends in the industrial space. Ruiz-Torres et al. (2019) analyse the relationship between an employee's performance and their degree of satisfaction by using a multi-criteria model that assigns tasks to employees based on their preferences. Further review was done on the topic by referring to a book by Knights, Willmott, and Brewis (2007) about organizational behaviour and management and a book by Dessler (2013) on Human Resource Management. The former provided theoretical concepts and methods that could improve the performance of employees in an organisation by assessing and making changes to their work environment, while the latter is considered one of the most read books when it comes to HR and provided great insight on how to manage a workforce.

The importance of digital human resource management tools was analysed by Strohmeier (2020) through an article in a popular German journal for human resource management. The literature makes a great case for why every organisation needs to digitise employee management and job scheduling. An article available on Wiley Online Library (Charlwood and Guenole, 2022) describes scenarios where Artificial Intelligence revolutionises the practice of organizing work and how it is imperative to consider ethics and fairness while such AI solutions are developed. Further research was conducted by reviewing a book by Pathak et al. (2022) which discusses the role of Artificial Intelligence in Organisational Transformation. These sources clearly show that the future of human resource management would be heavily reliant on the adoption of IT-based solutions to adequately respond to the growing needs and complexity of the operations in a company.

5.6 Currently Available Commercial Solutions

This section conducts a critical appraisal of the currently available commercial solutions that may be capable of providing an answer to the optimization problem that is being tackled by the dissertation project. These range from full-fledged software packages to software that provides just the main component required to arrive at a solution:

(i) Optaplanner:

Formerly known as Drool's Planner, Optaplanner is a powerful open-source AI constraint solver. It runs on a lightweight and embeddable planning engine. It is purely written in Java and is also compatible with other JVM languages such as Scala and Kotlin. It has a wide range of use cases, such as vehicle routing, employee rostering, job shop and maintenance scheduling, task assignment etc., and is more than capable of providing optimal solutions for the project. It also has a python wrapper called OptaPy, but as of now, it does not have the same level of performance as the original product.

(ii) Vehicle Routing Open-Source Optimization Machine:

Often abbreviated as the 'VROOM' project, this software package is an open-source optimization engine written purely in C++ and capable of providing fast and optimal solutions to various vehicle routing problems such as TSP, capacitated vehicle routing, pickup and delivery problems with time windows, etc. It adds further functionality by considering constraints such as skills and priorities of various tasks and can be fully integrated with routing engines like OSRM, Valhalla and openrouteservice. The project should be capable of generating results for the problem being researched in this dissertation.

(iii) Google Optimization Tools:

This software suite, also known as OR-Tools, provides solutions for a variety of combinatorial optimization problems such as constraint satisfaction, mixed integer programming, linear programming etc. The suite is open-source and written in C++, but wrappers are available in several languages like Java, Python and C#. Although it does not provide a direct solution, the suite can be tailored to cooperate with additional elements like a routing engine to produce an optimal solution. The package also provides unique features which are available only through a premium subscription.

(iv) LocalSolver:

LocalSolver is a closed-source, commercial optimizer that offers intricate APIs that are available in many languages, including Python, Java, C++ etc. It is widely utilised by several conglomerates like Sony, Bosch, Fujitsu, and Macy's and is very scalable. The company which has developed the optimizer asserts that it is the world's fastest optimization solver for a variety of problems, including vehicle flexible job shop scheduling, workforce scheduling, vehicle routing, and production scheduling. This is a great option to consider if the KTP client has the budget, as this commercial solution provides great results for the optimisation problem that is being researched.

The next section provides details about the business strategy that was formulated using the various theories and concepts discussed in the literature review. This is followed by an analysis of the client company, Arcus FM to determine whether the proposed strategy would be feasible.

6. Formulation of Business Strategy

Job scheduling is primarily the responsibility of the human resources division of a company, and it is the duty of the managers in the organization to ensure that the employees follow their assigned schedules and complete them on time. This is crucial in facilities management companies like Arcus FM since maintenance jobs are often time-sensitive, and clients may suffer significant damage to their businesses if deadlines are missed. These damages may stem from the malfunction of machinery that is crucial to the operation of the company or from the loss of resources due to inadequate response from the maintenance company. Every job has special requirements and can only be carried out by workers who are competent to do it, but there may be situations where such qualified individuals are not available.

The KTP project is responsible for designing a job scheduler which encompasses all these parameters to create optimal schedules. The objective of the dissertation is to provide a solution for the reactive maintenance services offered by the company and optimize the schedules when unforeseen events occur. The scheduler must be in line with the requirements and constraints set by the collaborating company, Arcus FM.

Proposed Strategy:

The KTP company can enhance its ability to schedule jobs which would enable it to complete more tasks per day and increase its responsiveness to unexpected situations that may arise during regular business operations. This can be realised by incorporating the job scheduler with the company’s existing infrastructure.

A business strategy framework called McKinsey’s 7S (Peters and Waterman, 1984) was used to determine whether the business strategy that was proposed is feasible.

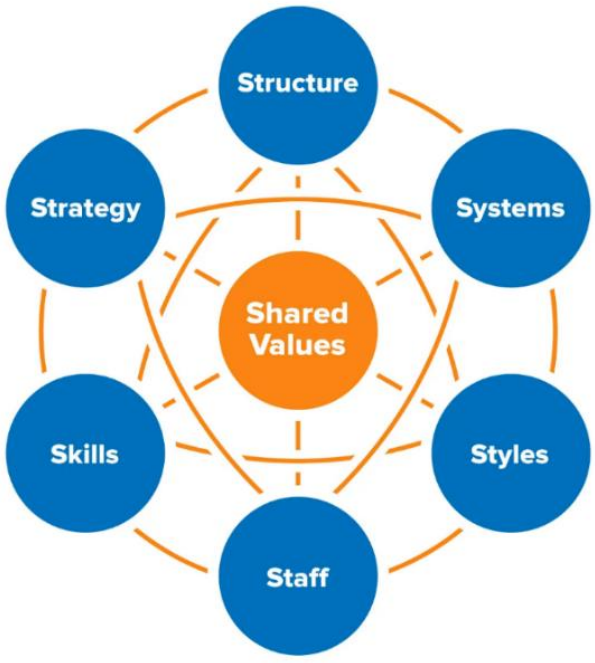


Figure 3: McKinsey’s 7S (Mindtools, 2021)

The data for each segment of the 7-S model was obtained by referring to the website hosted for Arcus FM (arcusfm.com).

Styles:

The company aims to promote a highly skilled and diverse workforce and achieve growth in a sustainable manner. The company employs cutting-edge, market-leading technology and a wealth of sector-specific expertise to deliver facilities management solutions that match its clients' shifting needs.

Skills:

Arcus FM offers a wide range of facilities management services ranging from Electrical, Mechanical, HVAC, Refrigeration, Drainage, and Sanitation to even energy auditing and management. The company uses an innovative ‘Internet-of-Things’ platform called Helix which enables it to monitor customers' assets, spot problems early, and act before they cause disruptions to the client’s operations.

Systems:

The company has specific divisions for each facilities management service that they offer. They also have an inhouse research and development team and their own supplier compliance management and safety management divisions. This has pushed it to become a formidable entity in the FM business space.

Structure:

The company follows the basic functional organizational structure where the company departmentalizes based on various job functions such as separate departments for marketing, sales, customer support, R and D, etc. This boosts accountability, increases stability and efficiency. This structure also establishes a chain of command that helps with communication between departments.

Staff:

The company has over 4,500 employees, and over 1,640 of them are technical engineers who are qualified to perform various FM services the company offers.

Strategy:

The company uses the latest technology available to provide an extensive array of reactive and maintenance services all day throughout the year, all of which may be integrated and packaged to meet the various demands of each business.

Shared Values:

The company's ultimate goal is to further expand its services and presence throughout the rest of the country and become the market leader when it comes to facilities management. They plan to make this a reality by using the latest technology available in the market and pursuing social missions such as being eco-friendly and promoting social harmony to increase their reputation, a valuable corporate asset.

The 7-S model has seven segments that can be grouped into ‘hard’ segments, which consist of Strategy, Structure and Systems which can be influenced directly by management and ‘soft’ segments, which consist of Style, Staff, Skills, and Shared Values that primarily depends on the company culture.

After analysing the 7-S model for Arcus FM, the shared values are in line with the various segments mentioned in the model and improving the job scheduling capabilities of the company can provide a significant competitive advantage to the company. The company also has the required infrastructure and personnel to implement the proposed strategy. The KTP project can create optimal schedules for the company, and the company would be able to execute those schedules using their highly skilled staff workers and managers. Emergencies such as unexpected jobs where client equipment needs immediate repair or complications in an ongoing task can be detected using the company’s IoT platform and be integrated into the job scheduler, which would be able to make alterations to the job schedules accordingly.

By combining the company’s existing infrastructure and integrating the Job scheduler into its array of HR tools, the company can gain a competitive advantage in terms of efficiency and customer satisfaction. Optimising the distribution of workload would improve the overall well-being of the organisation’s employees and the satisfaction they get from their work. The job scheduler can make this possible by making sure that a single employee doesn’t get overloaded with most of the work. By removing the human element during job scheduling, various cognitive biases can be overcome, and jobs can be assigned to the most suitable employees. Customer Satisfaction can also be improved by following this strategy; if more jobs can be done in a day, it would mean more clients have been served. Having the ability to tend to a client on short notice is a great asset to the company. The job scheduler can assist in accommodating such unforeseen emergencies in a way that will not jeopardise the existing appointments for an engineer.

7. Methodology

The industrial project involves the creation of an algorithm; Therefore, the methodology section is split into two subsections: design and implementation. The design section reports on the various components used in the algorithm and provides justification on why they are being used. It also provides an overview of the general structure of the algorithm. The implementation section describes how all the components mentioned in the design section are used in the realisation of the scheduling algorithm along with relevant pseudocode.

7.1 Algorithm Design

The literature review has already given a brief overview of the various theoretical concepts being used in the project, and a general structure of the scheduling algorithm is described below:

```
Algorithm 1: General Structure

   Input: Engineer Data (ID, Location, Skills)
          Job Data (ID, Location, Time Required)
          Schedule (Engineer ID, Jobs, Total Work Time)
          New Job Data (ID, Location, Time Required)
1  Set Constraints
2  Run Local Search Algorithm:
3      Generate Neighborhood
4      Search Neighborhood for an optimal solution
5      If an optimal schedule is found which passes the constraints:
6          Update schedule
   Output: Modified Schedules
```

The pseudocode provides the skeletal structure of the algorithm, and the upcoming sections expand on each element.

Inputs:

The scheduling algorithm first accepts data consisting of details about the employees in the facilities management company, such as their employee IDs, their locations, and the various skills each of them possesses, followed by data regarding the jobs being scheduled which includes the job IDs, the location in which it must be performed, and the skill required to perform the job. The algorithm also requires the schedules of all the engineers, which shows the jobs assigned to each of them and the order in which they must be executed. All the data will be fed in the form of CSV files.

Setting Constraints:

As mentioned earlier in the literature review, the scheduling algorithm can consider a generated solution only if it passes a set of constraints:

1) Time Limit: The total amount of time taken to complete a schedule which includes the travel time between each job location, starting from the engineer's home and back, along with the time taken to perform each job, should be less than a set limit (ideally 8 hours or 480 minutes).
2) Skill: The facilities management company has a roster of engineers who are qualified to perform a certain range of jobs, so the algorithm should only assign a job to an engineer who possesses the necessary skill required to perform it.

These constraints are enforced while the local search algorithm searches for possible solutions in the neighbourhood.

Local Search Algorithm:

The scheduling algorithm makes use of local search heuristics over classical search methods due to its ability to search for solutions over large spaces. This is made possible by starting with an initial solution which may or may not be feasible and improving it by iteratively taking steps such that a better or feasible solution is reached (Poole and Mackworth, 2017). One of the best candidates among the many local search algorithms that can be used in the scheduling algorithm is a variation of the simple hill-climbing algorithm called the Steepest-Ascent hill-climbing algorithm.

The hill climbing algorithm is a popular algorithm used in Artificial Intelligence which starts from a base solution and moves up continuously until an optimal solution is reached. It is based on the greedy approach, which enables the algorithm to reach local minima or maxima, but it also means that the solutions generated would be sub-optimal in many cases. Therefore, the Steepest Ascent Hill Climbing method is more appropriate in this scenario as modifications are being made on a preoptimized schedule, and this method first examines the neighbouring nodes in a neighbourhood and chooses the node which is closest to an optimal solution (Hoos and Stützle, 2004). The hill climbing algorithm has two primary components:

1) Neighbourhood Generating Function:

The neighbourhood generating function is used to create feasible solutions for the algorithm; this is made possible using the input data and generating new schedules for the engineers while adhering to the set constraints. First, the function takes a list of jobs which an engineer is qualified for and sends the addresses to a routing engine which returns the best possible route along with details such as the trip duration. The total time required for the schedule is then calculated by adding up the trip duration, and the time it would take to complete each job that has been considered. If the total time is less than the set limit, then the schedule is considered a solution. A pseudocode of the function is described below:

Algorithm 2: Neighborhood Generating Function

```
   Input: Engineers, Jobs, Schedules Data
1  Generate schedule consisting of eligible jobs for Engineer
2  Send job address list to routing engine:
3      Convert the address list into latitude and longitude
4      Generate optimal route
5      Return route and trip details
6  Calculate the total time required for the schedule
7  If the schedule passes all set constraints
8      Return Schedule
```

The Neighbourhood Generating Function is reliant on two main components for producing viable schedules for the engineers:

(i) Open-Source Routing Engine (OSRM):

The neighbourhood generating function uses an open-source routing engine for finding routes between the various job locations. The routing engine can produce routes as well as optimise them using a built-in travelling salesman problem solver. A local server hosting the engine has been set up, which is using pre-processed OpenStreetMap data of the United Kingdom. The function sends HTTP requests to the local server after converting the given addresses to latitude and longitude. The engine then performs the necessary calculations and sends back results in the form of JSON files which consist of the route and trip data. This data is then used to calculate the total time required for the schedule and check whether it is feasible.

(ii) Geocoder (Geopy):

The geocoder is responsible for converting the addresses into latitude and longitude values before sending them to the routing engine. The scheduling algorithm uses geopy, a client designed for Python, to provide access to many popular geocoding web services. The geocoding client provides a separate class for each geocoding service and can be accessed by importing them as libraries. The scheduling algorithm uses the geocoding services provided by Nominatim and ArcGIS.

Nominatim is an open-source geocoder which uses OpenStreetMap data to find geographical coordinates, while ArcGIS is a powerful pythonic library that offers a variety of GIS functionalities such as geocoding, mapping, routing, querying etc. Both the geocoders are used extensively along with OSRM to provide routing capabilities to the scheduling algorithm.

2) Neighbourhood Operators:

After generating a neighbourhood, the algorithm requires a tool to explore it for neighbours that could provide a better solution or, in this case, a more optimal schedule. This can be done by making modifications to an engineer's schedule by reallocating jobs within or between the schedules of other engineers.

Research articles from Santos et al. (2019) and Maravilha, Mayra-Pereira, and Campelo (2019) propose using six different neighbourhood functions or operators to explore the search space. Each of the operators is analysed to determine which of them is qualified to be used in the scheduling algorithm:

(i) Shift Operator:

The shift operator creates a new neighbour by moving the position of a job in an engineer's schedule to another position on the same schedule. The target position for the moved job is selected greedily where the total time taken to complete the schedule is the least. It has a complexity of O(n^2).

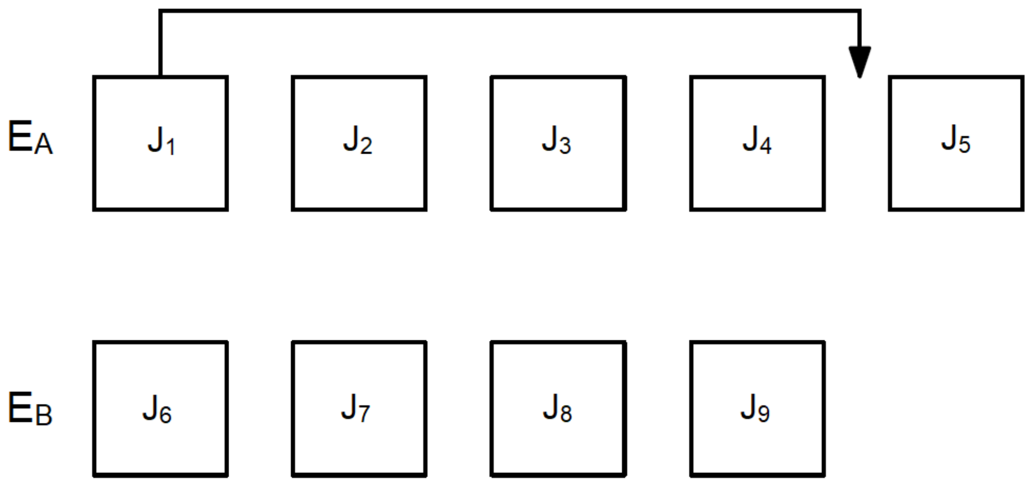


Figure 4: Shift Operator (Maravilha et al., 2019)

(ii) Switch Operator:

The switch operator generates a new neighbour by switching the order of two jobs in an engineer's schedule. The first job is selected randomly, and the second one is greedily chosen such that the switch would reduce the makespan. It has a complexity of O(n^2).

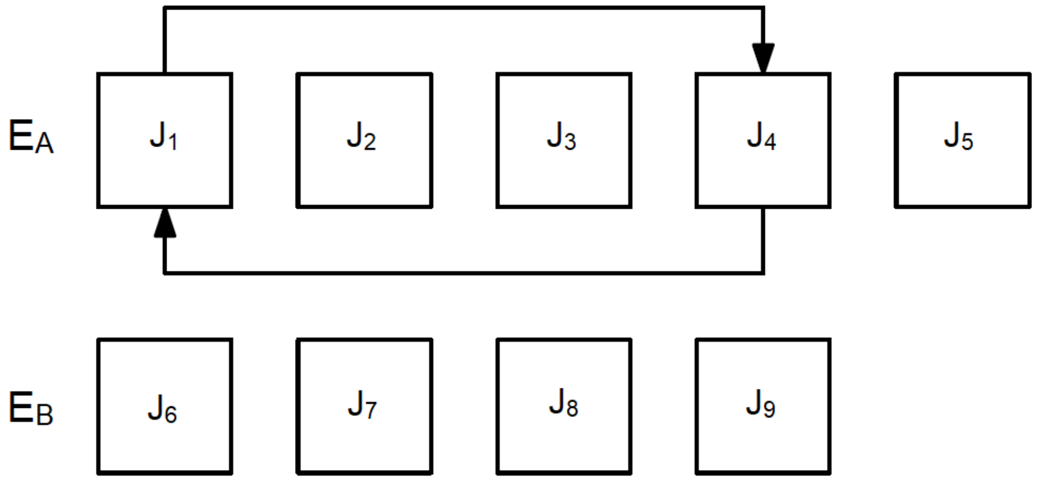


Figure 5: Switch Operator (Maravilha et al., 2019)

(iii) Two-Shift Operator:

The two-shift operator is a variation of the shift operator where the shift operation is done twice on the same engineer's schedule. It has a complexity of O(n^4).

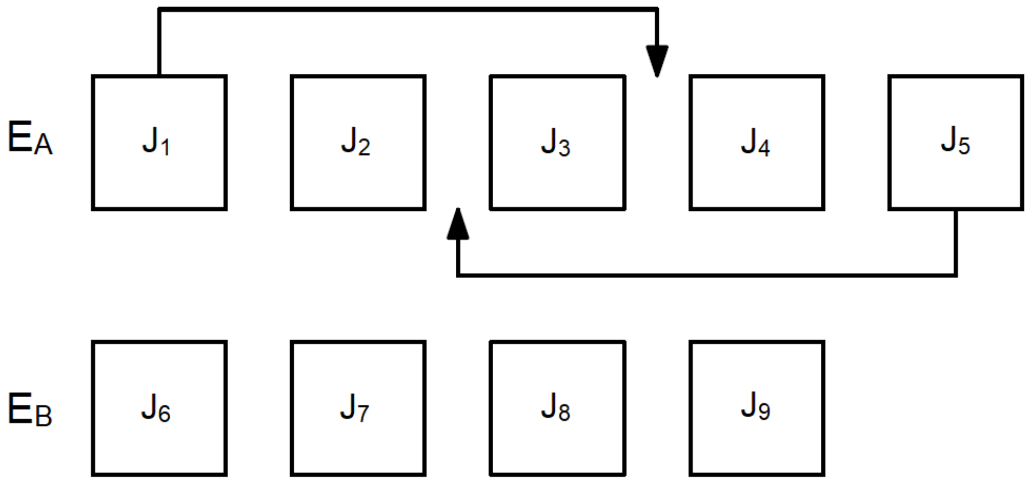


Figure 6: Two-Shift Operator (Maravilha et al., 2019)

(iv) Swap Operator:

The swap operator generates a new neighbour by swapping two jobs between two engineers' schedules. The jobs are chosen and placed randomly unless a greedy approach is used. It has a complexity of O(n^4).

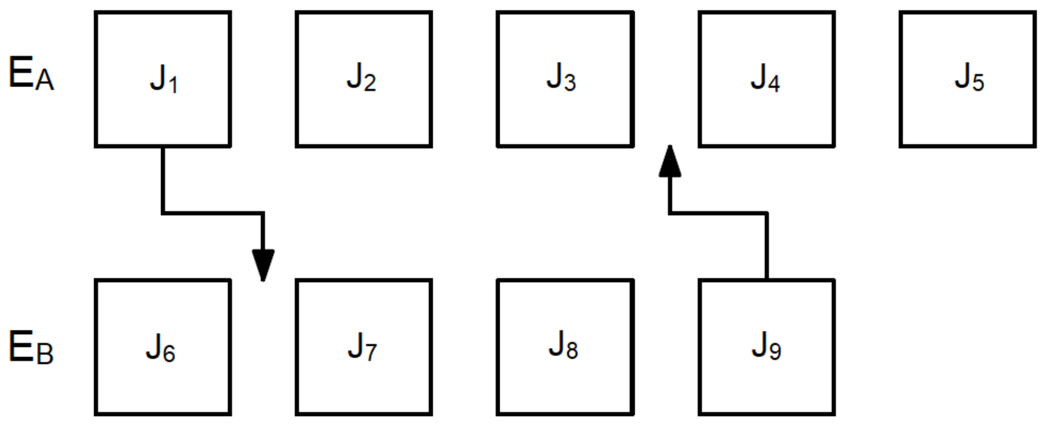


Figure 7: Swap Operator (Maravilha et al., 2019)

(v) Direct Swap Operator:

The direct swap operator is like the swap operator, but instead of placing the swapped jobs in a more optimal position, the jobs are placed in the position of the swapped job. It has a complexity of O(n^2).

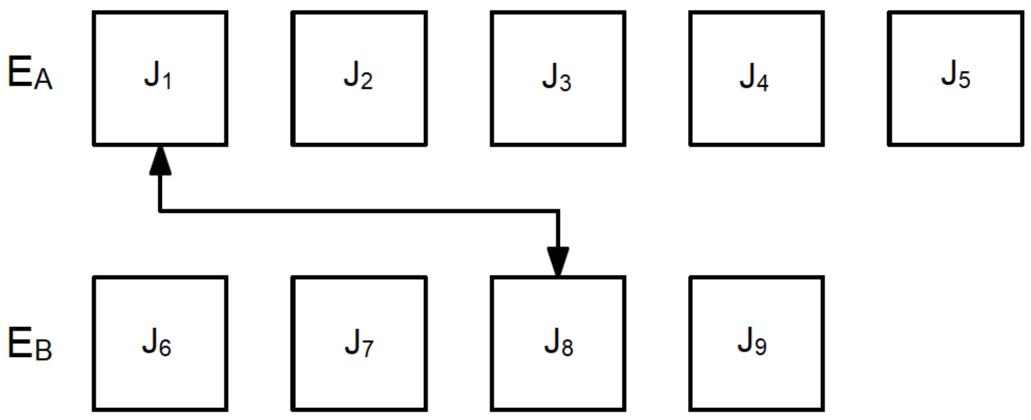


Figure 8: Direct Swap Operator (Maravilha et al., 2019)

(vi) Task Move Operator:

The task move operator creates a neighbour by choosing a random job from an engineer's schedule and assigning it to another engineer's schedule while making sure the job is placed in the best position that minimizes the total time taken to complete the schedule. It has a complexity of O(n^2).

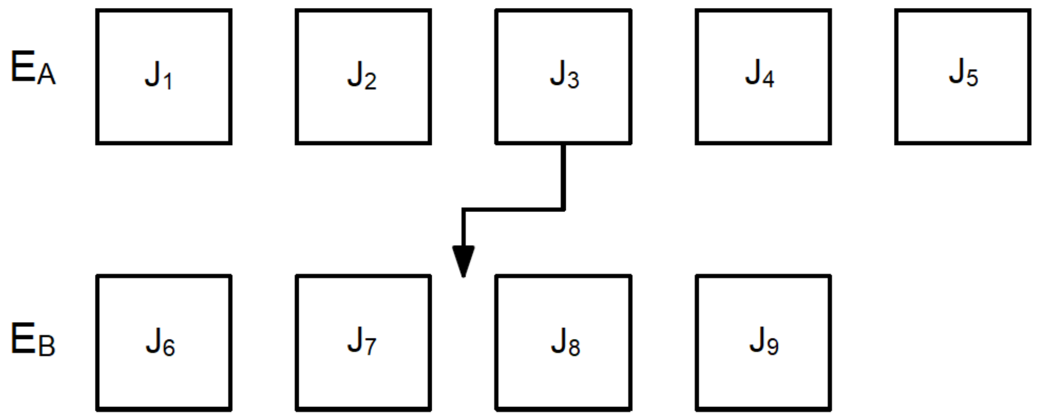


Figure 9: Task Move Operator (Maravilha et al., 2019)

The six operators can be split into two groups, intra-engineer operators, which consists of the shift, two-shift and switch operators and inter-engineer operators, which consists of the task move, swap and direct swap operators. Both groups play an important role in exploring the search space for an optimal solution, but some operators mentioned above may not perform as well as others in the scheduling algorithm.

Intra-Engineer Operators:

To determine the best intra-engineer operator for the scheduling algorithm, A brief comparison is made between the shift and switch operator, where an engineer's schedule consisting of nine job locations receives a tenth job, and one level of shifting or switching is conducted in each instance. A small-scale implementation of this scenario was coded in Python with all the components mentioned above, and the results were recorded.

The set of job addresses are locations across the West Midlands, and the schedule is set in a way where the engineer starts and comes back to the same location. The schedule is as follows:

```
['DY11 5SW', 'WV4 6ED', 'B79 7PB', 'B90 8AT', 'B46 1AN', 'B98
9EY', 'B96 6BD', 'WV7 3BW', 'CV9 1LQ', 'DY11 5SW']
```

A new job in address B16 9HN needs to be included in the schedule.

After performing the shift operation, the following results are obtained:

```
Best Schedule:
['DY11 5SW', 'WV4 6ED', 'B79 7PB', 'B90 8AT', 'B46 1AN', 'B98
9EY', 'B96 6BD', 'WV7 3BW', 'CV9 1LQ', 'B16 9HN', 'DY11 5SW']

Total Time Required for Schedule: 321.9583333333333 minutes
Execution time: 60.40881943702698 seconds
CPU Execution time: 2.828125 seconds
```

Similarly, the following results were obtained after performing the switch operation:

```
Best Schedule:
['DY11 5SW', 'WV4 6ED', 'WV7 3BW', 'B90 8AT', 'B46 1AN', 'B98
9EY', 'B96 6BD', 'B79 7PB', 'CV9 1LQ', 'B16 9HN', 'DY11 5SW']

Total Time Required for Schedule: 284.22833333333335 minutes
Execution time: 302.30359649658203 seconds
CPU Execution time: 15.59375 seconds
```

The analysis showed that the switch operator could provide better and more varied solutions, but it takes a significantly longer time to generate these solutions compared to the shift operator due to the greater number of possible neighbours that it creates. The primary problem with these operators is that they are not capable of creating an optimal schedule without the presence of the right implementation.

Therefore, intra-engineer operations were performed using the help of the Travelling Salesman Problem solver present in the OSRM routing engine.

The TSP solver plugin in OSRM provides a Hamiltonian circuit of the addresses that have been fed using a greedy heuristic called the farthest insertion algorithm. This method is used when there are more than ten input locations. Otherwise, a brute force approach is taken to find a solution. A basic implementation of the operation was created, and the results are as follows:

```
Best Schedule:
['DY11 5SW', 'WV4 6ED', 'WV7 3BW', 'B16 9HN', 'CV9 1LQ', 'B79
7PB', 'B46 1AN', 'B90 8AT', 'B98 9EY', 'B96 6BD', 'DY11 5SW']

Total Time Required for Schedule: 227.965 minutes
Execution time: 4.666219234466553 seconds
CPU Execution time: 0.140625 seconds
```

The results obtained using the help of a TSP solver provide far better results than both the shift and switch operators. The schedule obtained takes a lower time to complete, and the solution was generated in the span of a few seconds.

This analysis showed that the scheduling algorithm would be more efficient if the intra-engineer operations were performed intelligently with the help of the ORSM routing engine. The only downside to this approach is that it highly deviates from the originally created schedule but provides better results at very high speeds.

The two-shift operator was not analysed further as it was just a variation of the shift operator which could create more possible neighbours with no guarantee of finding an optimal solution. Furthermore, its high complexity makes it even less attractive an option to be considered.

Inter-Engineer Operators:

Inter-Engineer Operators perform operations between two schedules to create new solutions, an analysis was conducted between the task move, swap, and direct swap operators using the synthetic data described in the example scenario of the requirement analysis. Since the operations are between two engineers, the time limit to complete a schedule was set to 480 minutes and skill constraints were put into effect.

The following schedules were assigned to each engineer, and a new job J_{11} needs to be assigned a qualified engineer:

```
Initial Schedule:
------------------------------------
E.ID = 1
Schedule = ['3', '10', '5']
Total Time = 360.7 minutes
------------------------------------
E.ID = 2
Schedule = ['2', '9', '1']
Total Time = 447.545 minutes
------------------------------------
E.ID = 3
Schedule = ['8', '6', '4']
Total Time = 442.005 minutes
------------------------------------
E.ID = 4
Schedule = ['7']
Total Time = 375.03666666666663 minutes
```

Statistical data about the longest and shortest schedule and the total time of all the schedules are given below:

```
Maximum Time: 447.545 minutes
Minimum Time: 360.7 minutes
Range: 86.84500000000003 minutes
Total Time: 1625.2866666666666 minutes
```

The Task Move operator was applied on the given schedule, and the following results were obtained:

```
Final Schedule:
------------------------------------
E.ID = 1
Schedule = ['9', '3', '10', '5']
Total Time = 460.76666666666665 minutes
------------------------------------
E.ID = 2
Schedule = ['1', '2', '11']
Total Time = 448.8 minutes
------------------------------------
E.ID = 3
Schedule = ['8', '6', '4']
Total Time = 442.005 minutes
------------------------------------
E.ID = 4
Schedule = ['7']
Total Time = 375.03666666666663 minutes

Maximum Time: 461.8233333333333 minutes
Minimum Time: 375.03666666666663 minutes
Range: 86.78666666666669 minutes
Total Time: 1730.5933333333335 minutes

Execution time: 51.159278869628906 seconds
CPU Execution time: 0.953125 seconds
```

Similarly, the Swap and Direct Swap operators were tested using the same data, but they were not able to generate feasible results. Therefore, the data required alteration either by reducing the time of all the jobs by ten percent or by relaxing the time limit constraint. The choice was made to reduce the job times by ten percent, and these are the initial stats which were recorded:

```
Maximum Time: 417.545 minutes
Minimum Time: 345.03666666666663 minutes
Range: 72.50833333333338 minutes
Total Time: 1510.2866666666666 minutes
```

The Swap operator was applied to the altered data, and these were the results:

```
Final Schedule:
------------------------------------
E.ID = 1
Schedule = ['11', '5', '10', '9']
Total Time = 468.2866666666663 minutes
------------------------------------
E.ID = 2
Schedule = ['1', '2', '3']
Total Time = 436.7816666666664 minutes
------------------------------------
E.ID = 3
Schedule = ['8', '6', '4']
Total Time = 402.005 minutes
------------------------------------
E.ID = 4
Schedule = ['7']
Total Time = 345.0366666666663 minutes


Maximum Time: 469.475 minutes
Minimum Time: 345.0366666666663 minutes
Range: 124.4383333333339 minutes
Total Time: 1655.8016666666665 minutes

Execution time: 36.141122341156006 seconds
CPU Execution time: 0.765625 seconds
```

Similarly, the Direct Swap operator was applied to the schedules, and the result is as follows:

```
Final Schedule:
------------------------------------
E.ID = 1
Schedule = ['9', '10', '5', '11']
Total Time = 469.7766666666664 minutes
------------------------------------
E.ID = 2
Schedule = ['2', '3', '1']
Total Time = 455.75 minutes
------------------------------------
E.ID = 3
Schedule = ['8', '6', '4']
Total Time = 402.005 minutes
------------------------------------
E.ID = 4
Schedule = ['7']
Total Time = 345.0366666666663 minutes
```

```
Maximum Time: 469.77666666666664 minutes
Minimum Time: 345.03666666666663 minutes
Range: 124.74000000000001 minutes
Total Time: 1672.5683333333334 minutes

Execution time: 38.14116835594177 seconds
CPU Execution time: 0.84375 seconds
```

The analysis showed that out of the three operators, only the Task Move operator could produce feasible results without needing any alterations to the data being fed into the algorithm. The Task Move and Swap operators had the TSP Shift operator integrated along with them to make sure that the most optimal placements were given to the relocated jobs. Whereas in the case of the Direct Swap operator, the job locations were just swapped between the schedules, and that does not ensure optimal placement, which makes it a poor choice for the scheduling algorithm. This can be seen by comparing the results of the Swap and Direct Swap operators where similar jobs were assigned to both schedules, but their ordering was different, resulting in different times required to complete the schedule.

Another major flaw with the Swap and Direct Swap operators is that they create bigger disruptions in the schedules compared to the Task Move operator, as two jobs are getting reallocated at a time. There may also be a possibility that the operators may skip possible solutions which may need only one job to be reallocated. Moreover, the Swap operator can be implemented using the Task Move operator by executing it twice between two schedules. This leads to the conclusion that the only inter-engineer operator which would be required is the Task Move operator. All the code that was used to conduct these tests, along with their results, have been included in the dissertation submission.

OUTPUT:

The scheduling algorithm uses the various components mentioned in the previous subsections to generate optimal schedules for the engineers. The algorithm takes data about the engineers, jobs and initial schedules and passes them into the local search algorithm where a neighbourhood is generated, and this neighbourhood is explored using the TSP Shift and Task Move operators, which searches for an optimal solution which matches all the constraints set by the algorithm and returns output schedules.

7.2 Implementation

Using the various components and theories mentioned in the previous sections, a working model of the scheduling algorithm was created. The following pseudocode is a high-level overview of its implementation:

Algorithm 3: Scheduling Algorithm

```
   Input: Engineer Data (ID, Location, Skills)
          Job Data (ID, Location, Time Required)
          Schedule (Engineer ID, Jobs, Total Work Time)
          New Job Data (ID, Location, Time Required)
BEGIN:
  Find qualified engineers for new job and create a list
  Run TSP Shift Operator:
      Insert Job into a qualified engineer
      Calculate total work time for the engineer
      If total work time < Set time limit (480 Minutes):
          Update schedule
          END: Job successfully assigned
      Else:
          Move to next qualified engineer
  If Job not assigned:
      Run Task Move Operator:
          Insert Job into qualified engineer [1]
          Calculate total work time for engineer [1]
          If total work time > Set time limit (480 Minutes):
              Select a job from the engineer's [1] schedule
              Find qualified engineers for the job
              Insert job into a qualified engineer's [2] schedule
              Calculate total work time for engineer [2]
              If total work time < Set time limit (480 Minutes):
                  Update schedule
                  END: Job successfully assigned
              Else:
                  Move to the next qualified engineer [2]
              Search with every job in engineer [1]
          If job not assigned:
              Remove new job from engineer [1]
              Add the new job to the next qualified engineer
   Output: Modified Schedules
```

The scheduling algorithm can be divided into two stages. The first stage [From **1** to **10**] begins by attempting to insert the new job into a qualified engineer and optimising the schedule for the new addition using the TSP solver plugin from the OSRM routing engine. This stage involves making changes to a single engineer’s schedule and creates the least amount of disruption. If this results in a failure where no schedules are found that passes all the set constraints, the algorithm moves to the next stage which tries task move operations on the schedules.

The second stage [From **11** to **27**] involves operations between a pair of engineer schedules to find an optimal allocation for the new job. Since changes are made in the schedules of two engineers, the level of disruption in the original schedule increases. The schedule provided to the algorithm is an optimized one, so it would be best if a minimal number of changes are made to it. Therefore, the task move operation is done only at one level, and various combinations of jobs are checked between two schedules. If a feasible solution is not reached, then another pair of schedules are considered until an optimal solution is found or all combinations have been checked.

The scheduling algorithm can be implemented as a systematic search, but it will not be feasible when dealing with large datasets. Therefore, elements of local search can be implemented into the algorithm by introducing stochasticity while picking the engineer to insert the new job in stage one or while picking the pair of engineers as well the jobs to perform task move operations in stage two.

When the new job is inserted into an engineer’s schedule, it is called a random initialization and a neighbourhood is generated, the TSP plugin of the routing engine searches for the best solution within that neighbourhood and returns it. If the proposed schedule passes all the constraints, it becomes the best solution. Random restarts can be implemented to explore other parts of the search space by generating a new neighbourhood by following the same procedure with another engineer. If the newly generated schedule has a lower work time, then it can be chosen as the new best solution, this process of minimization is called greedy descent. This can be implemented in both stage one as well as stage two.

The local search algorithm can be used for optimization purposes by running it for a certain amount of time and returning the best solution found. But this can only guarantee a local optimum. A global optimum cannot be found until all other possible solutions are systematically searched in every part of the search space. This can be regarded as a key weakness of the algorithm. A choice must be made whether the algorithm should just find a solution which passes all the set constraints or the best solution among all the other possible solutions at the cost of a longer computational time. The optimization process can be sped up by reducing the search space through the use of the constraints set for determining what qualifies as a solution. An example that can be considered is using skill as a constraint significantly reduces the search space of the algorithm by limiting the search within the schedules of only qualified engineers.

The scheduler is meant to be used during two instances:

1) Inserting a new job before the beginning of the workday:

This can be done by following the implementation procedure mentioned above by providing the algorithm with the original schedule and the new job which needs to be inserted.

2) Inserting a new job during the workday:

This scenario can be tackled in a similar way as instance 1. The locations of the engineers are first tracked. This will usually be the location of the job site they are currently at. Fleet tracking systems like Verizon Connect, Samsara or Jobber can be used to get accurate estimates of engineer locations. Once such data has been recorded, the completed jobs are removed from the schedule and the ongoing jobs become the starting locations for the engineers. The new job is then inserted into an engineer’s schedule depending on the urgency of the new job in a similar fashion as instance 1.

8. Testing Methods

The scheduling algorithm was initially tested using the synthetic data created for analysing the neighbourhood operators for the algorithm. This testing was done for two cases: The shift stage and the task move stage. To implement this, the test data was altered in a way that the scheduling algorithm must move on to the next stage to find a solution. This was done to make sure that the algorithm functions as intended.

Further testing was done on real life data which was procured after conducting an online interview with a KTP associate of the project. The interview was done after getting the required ethical approval from the university and all rules and regulations set by the ethics committee were followed when it came to the storage and use of the data that was provided. The interview involved asking the KTP associate a series of questions about the daily operations of the company, the data given by the KTP associate that will be the input of this dissertation and the requirements for the local search algorithm

The KTP associate provided a set of four schedules which were generated using their complex algorithms. It consisted of a five-day schedule which included five different schedules for 110 engineers and 4175 refrigeration jobs. The other three samples were on a smaller scale which included engineers and jobs with multiple skills.

The algorithm could have been tested by simply inserting a new job into each schedule, but that result would not be a reliable way to gauge the usability of the scheduler. Therefore, a testing method was devised where ten percent of the jobs assigned in each schedule were removed and reinserted using the scheduling algorithm. The constraints set for the schedules were very relaxed as every job in every schedule had a duration of one hour and with no serious time limits. This was evident from the fact that there were engineers who had no jobs assigned to them while some had work lasting up to fourteen hours. So, the primary feature that was being compared was the total driving time for each engineer in the schedules.

Day one of the five-day itinerary was the first sample to be tested. It consisted of over 600 jobs distributed unevenly to 110 engineers. All the engineers and jobs required the same skill called ‘refrigeration’. An attempt was made to remove 60 jobs and reinsert them back into the schedule, but the analysis was met with a bottleneck. The routing engine and geocoder were unable to process the large number of requests that were being made for this analysis. The servers used for these calculations are free and open-sourced, but they are set with a request limit for each connection to prevent abuse. Therefore, the analysis was reduced to 30 engineers and around 150 jobs.

Fifteen random jobs were removed from the schedule and reinserted and the bar graph below shows the difference between the driving times of each engineer before the jobs were removed and after they have been reintroduced into the schedules.

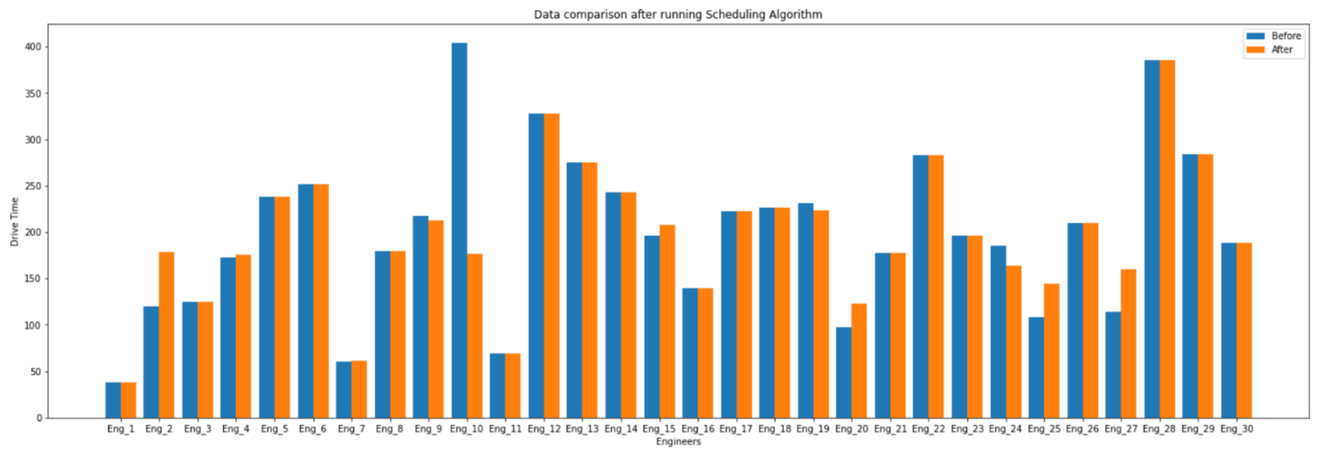


Figure 10: Drive Times of Engineers for Day 1 in 5-day Schedules (S1)

Similarly, the same test was done on the other three sample schedules. These samples were of smaller scales and the routing engine and geocoder had no trouble accommodating them. The main difference between the previous schedule and the three in question was that these schedules had jobs with a wide variety of skill requirements and many of the engineers assigned to them possessed more than one skill. The first out of the three schedules that were considered consisted of 35 engineers who were assigned 97 jobs. The distribution of jobs between the engineers was very skewed where some engineers had no jobs assigned to them while some were assigned over seven.

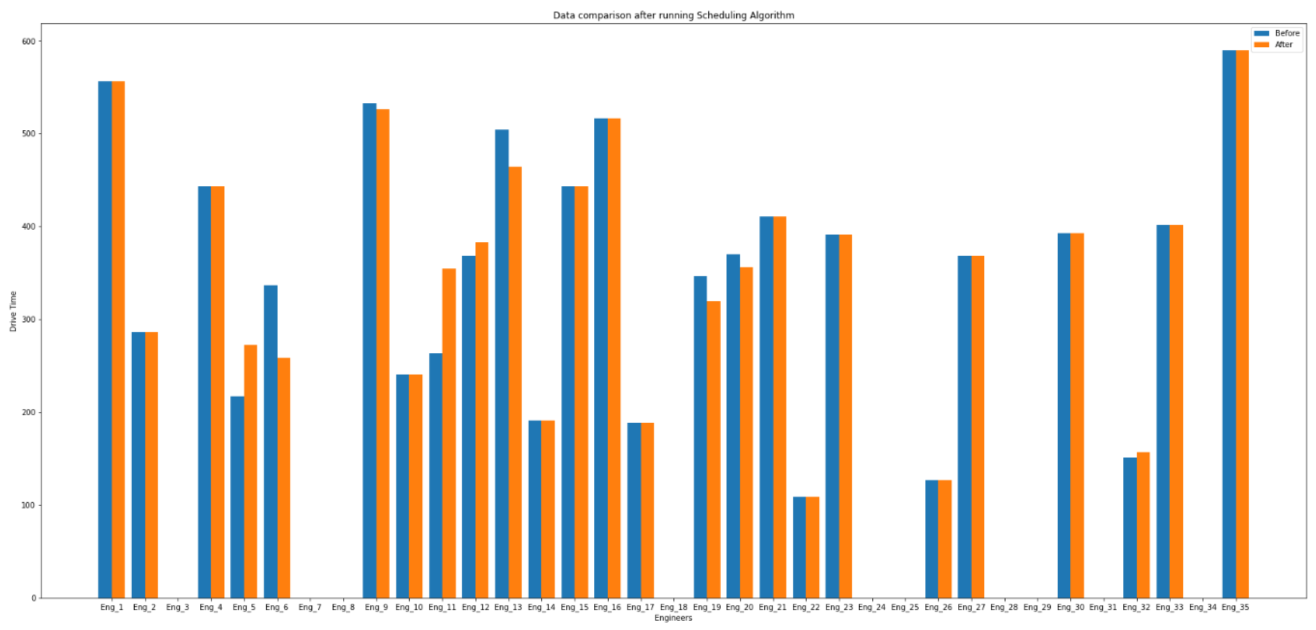


Figure 11: Drive Times of 35 Engineers assigned 97 Jobs (S2)

The bar graph above conveys the result obtained after ten random jobs were removed from the schedule and reassigned using the scheduling algorithm.

The analysis was conducted on the other two schedules as well, one with 15 engineers and 59 jobs and the other with 24 engineers and 38 jobs. 6 jobs were removed from the former and 4 from the latter. These jobs were then reinserted, and the results are as follows:

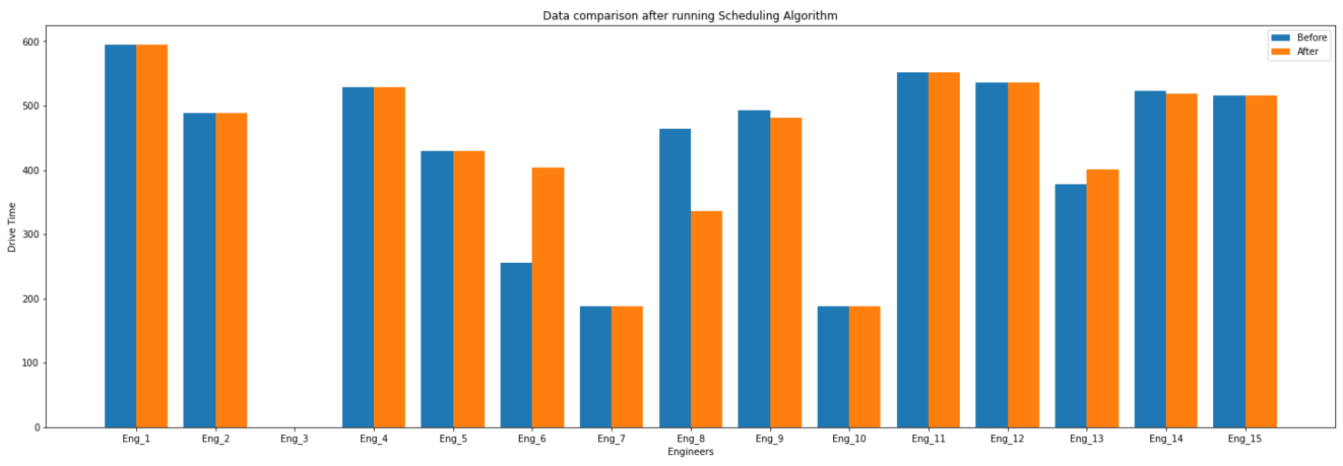


Figure 12: Drive Times of 15 Engineers assigned 59 Jobs (S3)

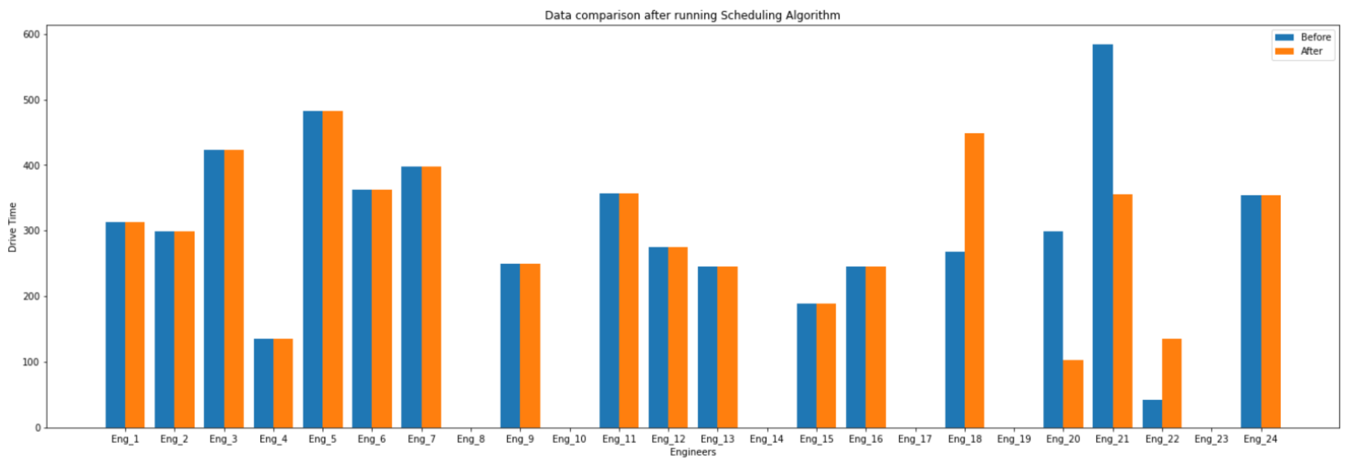


Figure 13: Drive Times of 24 Engineers assigned 38 Jobs (S4)

The code used to conduct the analysis along with their outputs have been submitted in the form of Jupyter notebooks. The results of the analysis are discussed in depth in the evaluation section.

9. Results and Evaluation

This section conducts a realistic evaluation of the results obtained from the tests on the scheduling algorithm to gauge its reliability and usability. Recommendations for the company were extracted from the evaluation, and feedback was obtained from the KTP associate on its usefulness. This is followed by an assessment of the methodology used to develop the scheduling algorithm and a brief analysis of its business value and the ethical implications it may have on the stakeholders of the project.

9.1 Analysis of Test Results

The implementation section has briefly described the testing methods that were used to analyse the effectiveness of the scheduling algorithm. The bar graphs that were generated as a result show that the proposed scheduling algorithm is feasible. The general trend that can be seen from all four test cases where ten percent of the assigned jobs were randomly removed and reassigned, is that the driving times of many engineers are very similar.

The schedules that were used in the test were preoptimized and the role of the proposed scheduling algorithm is to insert a new job into them. If the driving times of engineers were similar before and after the test was conducted, then that means that the scheduler was able to reassign the job into a position which was considered optimal. The scheduler has also been able to significantly reduce the driving times of some engineers by assigning the jobs to more appropriate engineers. This can be seen in the drive times for engineer 10 in Schedule 1 (Figure 10 (S1)) and engineer 21 in Schedule 4 (Figure 13 (S4)).

The scheduler was able to evenly distribute the jobs such that the total drive times of the schedules were more or less equal to the original schedule. But the overall drive time for some engineers was increased by assigning them new jobs so that there would be a significant reduction in the drive times of other engineers. A bar graph of the total drive times of all the engineer (S1 to S4) before and after the test was conducted, is as follows:

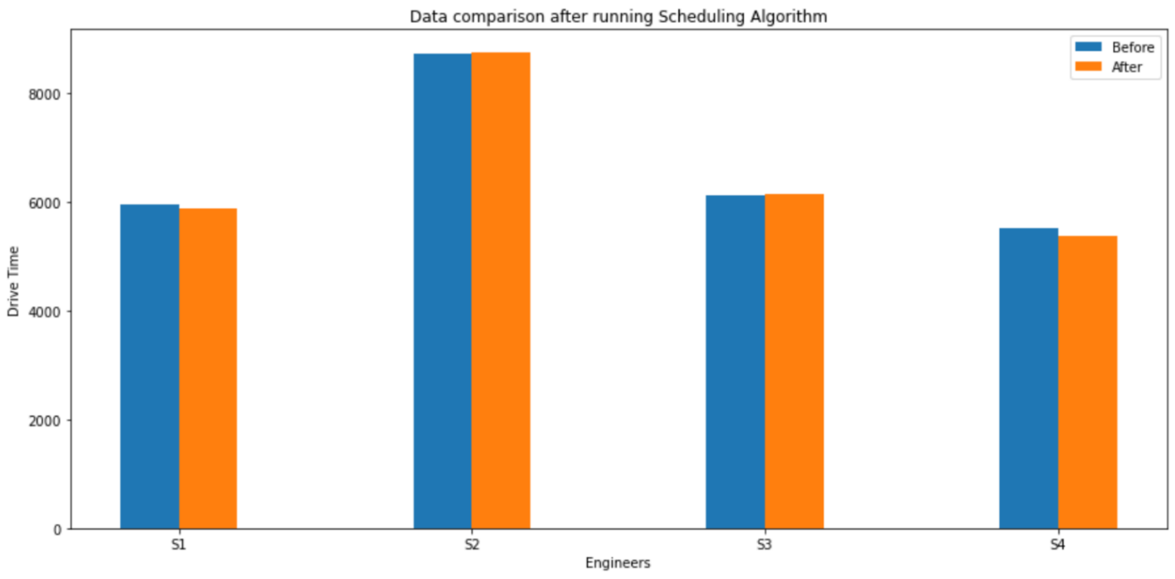


Figure 14: Total Drive Times the Schedules (S1 to S4)

These results were obtained due to the various constraints that were set on the scheduling algorithm when the removed jobs were reinserted into the schedule such as maximum number of jobs that can be assigned or avoiding engineers without any jobs assigned to them. A test was conducted on schedule 4 (S4) without any constraints and 50 percent of the jobs were removed from it and reassigned.

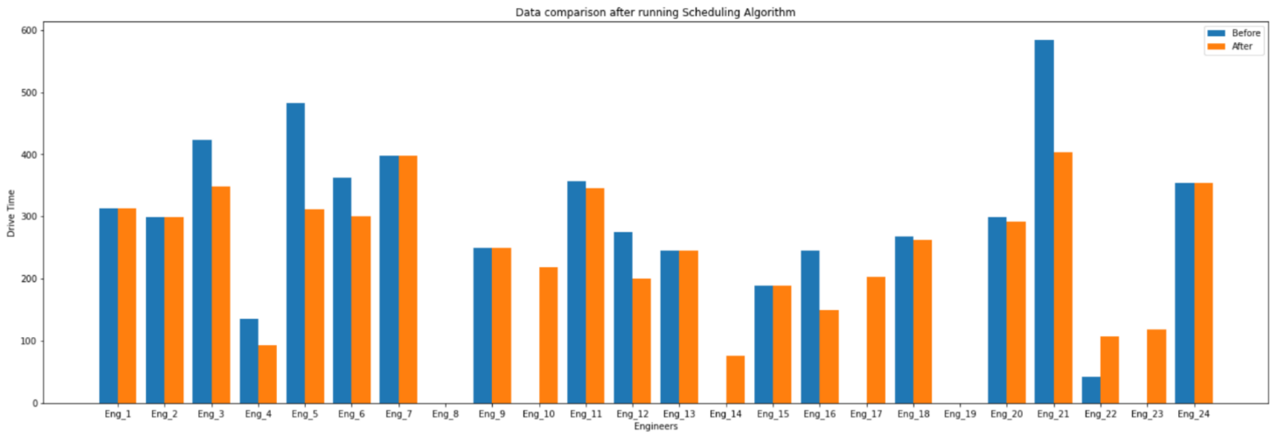


Figure 15: Drive Times of 24 Engineers assigned 38 Jobs (S4) [Without Constraints]

The resulting bar graph shows that the jobs were assigned to some engineers who initially didn’t have any jobs in their schedules, and this has significantly reduced the travel times for some engineers, but this schedule may not be reasonable as it may be better to have some engineers on standby in case of an emergency than having them all work at the same time.

9.2 Recommendation

Based on the results from the evaluation, the best approach for inserting a new job into a preoptimized schedule is to use the first stage of the scheduling algorithm where the new job is inserted into a schedule and a TSP solver is run on it to find the best combination. Then a greedy descent approach can be taken which searches for a better solution by performing the same procedure on another schedule. If a schedule is found with a lower total time, then it becomes the new solution. The search is performed for all the other assignments until a global optimum is reached.

9.3 Usability and Reliability

This approach would result in the least number of changes made in the original schedule and a solution will always be obtained unless hard constraints such as time limits or skill requirements are placed. But such constraints can be useful for the scheduling algorithm as the search space would be reduced and a global optimum can be found faster.

The primary weakness of this approach can be seen in the analysis that was conducted on the schedule S1 with 110 engineers and 600 jobs. Since no hard constraints such as skill requirements or time limits were specified, the algorithm must consider a large search space and a global optimum cannot be reached without systematically searching every possible assignment, making it very time-consuming (Poole and Mackworth, 2017).

9.4 Client Feedback

The data collected from the evaluation of the scheduling algorithm was documented and a report was submitted (available in supplementary submissions) to the KTP associate, Dr Arezoo Vejdanparast, along with recommendations for the company. The feedback from Dr Arezoo was positive and the observations provided by the study were viewed to be useful for the future development of the KTP project. She also advised on improving the recommended approach through the addition of a constraint in the scheduling algorithm which places deadlines on jobs being assigned to the engineers. But this change could not be added to the scheduling algorithm under the given time scale.

9.5 Assessment of Methodology

The project followed the basic structure of a quantitative research which began by analysing the required theories and concepts which came into play during the realisation of the solution. This was followed by a conceptualisation stage where several variations of pseudocode were proposed which was critical in making sure that the scheduling algorithm met all the specified objectives of the project. Each variation of the pseudocode was iteratively refined, and the final version was converted into source code. The resulting algorithm was first tested on synthetic data followed by actual historical data from the KTP company. This was done after getting the due ethical approval from the university. The results of the tests were then analysed to gauge the reliability and usability of the algorithm and then documented.

The project methodology also involved elements of the strategic management process such as setting goals or objectives, analysis, strategy formation, implementation, and monitoring of progress (Johnson et al., 2020). This was practised from the early stages of the project where regular meetings were set with the project supervisors from both the college of engineering and physical science as well as the school of business. These meetings were essential for the proposal of ideas and solving problems that were encountered during development. Strategic management also helped in effectively allocating tasks to be completed and responding to unforeseen delays.

From the computer science perspective, the project would not have become a reality without the use of various open-sourced tools which were freely available online. They can be considered the backbone for driving digital innovation and development. Both the routing engine as well as the geocoder used in the scheduling algorithm are open-source software. They are very transparent, and the source code is available via platforms such as GitHub, making it possible for anyone to understand how the software works and the various functions it performs.

From a more technical standpoint, the actual development of the source code was completed smoothly thanks to the adoption of modular programming. Even though this programming principle has been around for over six decades, it is still invaluable to the modern software engineer. The scheduling algorithm, which was completely coded in Python, followed this process where the program was divided into smaller sub-programs which were easier to manage. This separation made it simpler to test, implement, or develop the various functions that the scheduling algorithm uses. The merits of this programming methodology became clear when the scheduling algorithm was validated using actual historical data. The only change that needed to be made to the program was the pre-processing section and the rest of the code was reused. The program was able to further embrace modularity by using libraries and APIs which help hide the core of the code and only expose the parts that developers need to utilise them.

9.6 Analysis of Business Value

This section conducts an analysis of the potential business values that the project deliverables provide to the main stakeholder, Arcus FM. A business strategy framework developed by the Boston Consulting Group called the BCG Growth Share Matrix (Reeves, Moose, and Venema, 2014) was used to determine whether the project would help in accelerating the company’s growth and have a competitive edge over rival organizations. The business strategy formulation section gave a brief insight into the various strengths and capabilities the company currently possess.

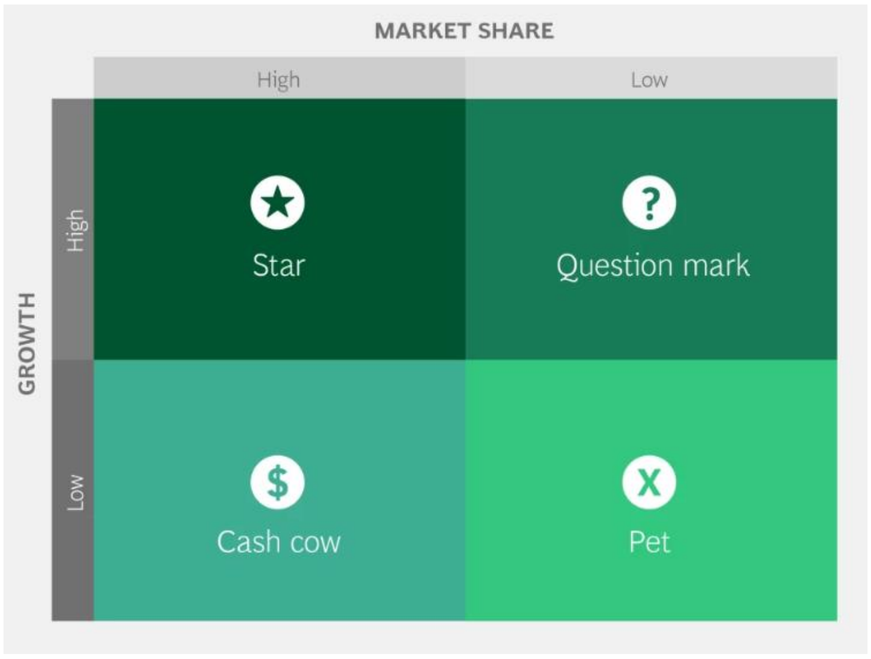


Figure 16: BCG Growth-Share Matrix (bcg.com)

The McKinsey's 7S analysis showed that the company has a keen interest in enhancing its IT-based capabilities to set itself apart from the competition. The addition of the job scheduler designed by the KTP program if integrated along with the company's existing infrastructure could enhance the services that it currently offers. If the job scheduler could work alongside the company's IoT platform Helix, it would place the project deliverables in the Star quadrant of the matrix as it would provide the company with the ability to better serve customers as well as improve overall productivity through optimized job allocations for the company's employees and effectively respond to emergency situations. Elements in the star quadrant usually generate the same amount of capital as it takes to maintain it. This is due to the initial costs of setting up such an integration. But if executed correctly, this implementation can turn into a cash cow which would generate more capital than it consumes and push the company to become a market leader. The incorporation of the most recent technologies into the architecture of the business may also aid in retaining current investors and luring in new ones. Overall, the project deliverables provide a great amount of value to the company and investing in it would be profitable.

9.7 Risk Assessment and Ethical Implications

The project has four main ethical standards that need to be upheld: (a) Minimizing risk, (b) Obtaining informed consent, (c) Protecting data confidentiality and (d) Avoiding deceptive practices (Dissertation.laerd.com, 2021).

The key objective of the dissertation is to modify a travelling maintenance worker's itinerary in response to emergency tasks. This requires allocating a predetermined route in order to create an ideal timetable. There may be a chance that the scheduler assigns an engineer a potentially dangerous travel path. This should be prevented by adding hard constraints on the routing engine which stops it from generating such routes and guarantee the safety of the user following the itinerary.

When risk is considered from a technical perspective, there may exist the possibility that a component of the scheduling algorithm fails to function as intended. An example of such a scenario is when the routing engine used in the project fails to calculate the distance between two points. The only way to tackle this problem would be to adopt an alternative method to calculate the value such as using a haversine function to get a rough estimate of the distance when the routing engine fails.

When it comes to the actual dissertation, any activity related to the scheduling algorithm along with its implementation should be done only after getting the informed approval of the project supervisors and the other major stakeholders. The data provided by the partnering company must be protected against accidental, illegal, or unauthorized access and misuse. This can be ensured by getting ethical approval from the university’s ethics committee before handling company data and adhering to the highest ethical standards possible. Another major ethical responsibility is maintaining academic integrity by citing and providing credit to every source that has been utilised in the dissertation and for the creation of the scheduling algorithm.

The main ethical implication that such an IT-based solution which uses Artificial Intelligence would have on society is how it could change the way companies manage human resources. This theme has been briefly discussed in the literature review section for business strategy and its formulation. More and more firms are taking a keen interest in adopting AI-based programs for activities such as screening job applications for new employees, administrative responsibilities like job allocation, report generation, monitoring employees, resource management and even marketing (Charlwood and Guenole, 2022). In the case of the scheduling algorithm designed in the dissertation, it uses a rudimentary element of Artificial Intelligence which can only recommend an optimal schedule for a given collection of jobs, but it can be improved through the augmentation of functionalities such as predictive analytics and cut out managers from the equation when it comes to assigning jobs to workers and eliminate scheduling biases (Pathak et al., 2022). A possible example that can be considered is how an AI system decides whom to assign a task from a group of equally qualified employees. The system can make this decision by extracting information from an existing database about employee performance and assigning the job to the best-rated employee. But this is where the ethical implication arises, the way jobs would be assigned would depend on the kind of data being fed into the AI system. If an excellent employee happens to underperform due to some issues in his or her private life, it wouldn't matter to the AI system, to it, the employee's performance rating has reduced, and this makes him or her a less favoured option to consider while assigning tasks. This can be applied to other scenarios such as screening candidates for jobs where an AI may favour a particular set of people while ignoring other qualified candidates just because its decisions are based on the training data provided to it.

To mitigate such bias caused in the decision-making of AI, it is imperative that companies that use AI systems stay up to date on this rapidly evolving field of study. Organisations can have internal "red teams" or conduct external audits to ensure that their AI systems are fair. Another way to mitigate bias is to run the AI system alongside human decision-makers where the system can provide a list of recommendations that an actual human can choose from. Finally, the best way to improve an AI system's degree of fairness is to provide it with data that is ethical and transparent while embedding ethics into the AI system's core set of directives (Harvard Business Review, 2019).

10. Project Management

The project initially followed a Waterfall methodology where all the important requirements of the project were clarified and plans to accommodate these requirements were presented. Regular meetings were held between the supervisors and these plans were scrutinized thoroughly. Once the project plan was approved, a logical design of the project was presented in the form of a pseudocode and the project's implementation was initiated after the pseudocode was authorised. This was done in parallel to conducting research on various theories and concepts necessary to implement the proposed plans. The following Gantt chart shows the initial project plan that was proposed:

Tasks	May	June	July	August
Research on topics relevant to the project	02/05/22			
Submission of dissertation proposal	03/05/22			
Further research on the local search and scheduling problem	31/05/22			
Creation of an initial framework		16/06/22		
Submission of Ethics Self-Declaration and Project Definition Form		17/06/22		
Creation of a basic working model		30/06/22		
Model evaluated by dissertation supervisor			05/07/22	
Ethical approval application submission			05/07/22	
Identifying issues and bugs in the model			10/07/22	
Improvements on current working model			15/07/22	
New model submitted for approval			20/07/22	
Company Interview for gathering data			30/07/22	
New model gets tested using real life data			31/07/22	
Final version of optimization model created				15/08/22
Company consultation				22/08/22
Dissertation report created and submitted				31/08/22

The project timeline was designed to accommodate any kind of delays by attempting to complete all tasks a month before the actual project submission date. Everything which was done related to the project was recorded on Trello, a Kanban-style list-making programme that was created by Trello Enterprise, an Atlassian subsidiary. Details of the Trello board are available in the appendix section.

While the research and development of the project went relatively smoothly, issues during the application for ethical approval from the university that was necessary to arrange a meeting with the KTP associate and use the data she provided to test out the solution developed for the dissertation severely hampered progress. The delays began from the early stages of the application, filling out an application form and getting it approved by all three supervisors took longer than anticipated resulting in the application being submitted on the 25th of July instead of the 5th. This was followed by further delays while processing the application as it required amendments due to discrepancies in its contents, and the slow response from the faculty responsible for processing the application as he was on his annual leave. The application was approved on the 7th of September which was much later than originally planned.

During the wait for the ethical approval, a solution and methodology for the scheduling problem were designed and it was approved by the supervisor. By the end of July, a basic model of the solution was produced, however, it had a serious flaw since the routing engine wasn't operating as intended. The engine was not able to create routes for various combinations of addresses and these results were essential for generating a solution. The issue was identified and fixed within a few days and the redesigned model could generate accurate solutions when tested on synthetic data. All that was required was the real historical data from the company which was procured on the 21st of September. Before getting the dataset, an online interview was conducted on the 12th of September with the KTP associate where a series of questions were asked to clarify the various objectives and constraints set by the company while scheduling jobs for the engineers and gain an understanding of their daily operations. A deadline extension was requested due to the short amount of time left for testing the scheduling algorithm with the new data and documenting the results. But thanks to the way the program was developed, pre-processing the data, and integrating it with the scheduling algorithm was faster than anticipated.

The evaluation data obtained after testing the scheduling algorithm on real historical schedules was documented, and a report containing recommendations for the company was submitted to the KTP associate on the 25th of September and the final dissertation report which included the associate's feedback was submitted along with the code that was written, on the 30th of September. The following table describes the actual timeline of the project:

Description	Date
Project formally begins through online meeting with EPS project supervisors	26/05/22
Research for generating a solution begins	27/05/22
Requirement analysis was completed, and a basic model was presented	09/06/22
Ethics and project definition form was filled out and submitted	17/06/22
Logical example was designed, and development of pseudocode begins	23/06/22
Started filling out application for ethical approval	07/07/22
First Meeting with ABS supervisor	22/07/22
Application for ethical approval was submitted	25/07/22
Approval of proposed pseudocode and beginning of actual development	28/07/22
First working model presented	12/08/22
Debugging of the model was completed	26/08/22
Working model using different operators was presented	30/08/22
Ethical approval application was approved	07/09/22
Final Meeting with ABS supervisor	09/09/22
Meeting with KTP associate	12/09/22
Final code of the scheduling algorithm was developed	15/09/22
Application for deadline extension was submitted	20/09/22
Company data received from KTP associate	21/09/22
Application for deadline extension was approved	22/09/22
Scheduling algorithm was tested out on real historical data	22/09/22
Report containing evaluation data and recommendations submitted	25/09/22
Feedback from KTP associate	26/09/22
Final meeting with EPS supervisor	28/09/22
Dissertation report completed	29/09/22
Dissertation submitted along with code for the scheduling algorithm	30/09/22

The project could not be completed in the original timeframe due to the various complications which occurred along the way, but all the main objectives of the project were achieved, and the report was submitted only three days later than the original deadline.

11. Conclusion

The dissertation successfully designed a scheduling algorithm that can optimize the schedules of engineers working for the facilities management company, Arcus FM when new jobs require assignment before the start of the workday or during it. The various literature sources that were consulted, and the tools used to realise this algorithm were reviewed. A business strategy was formulated and checked whether it would be feasible for the company. The design and implementation of the scheduling algorithm were described followed by an analysis and evaluation of its testing methods and their results. A report that consisted of recommendations for the company along with an implementation plan was submitted to the KTP associate from whom real historical test data was initially procured through an online meeting. The report proposed a local search algorithm with a greedy descent approach and the overall feedback was positive. The primary limitation of the approach is the long time it would take to find the global optimum for a large dataset, but this can be countered by reducing the search space of the algorithm by introducing constraints for generating solutions. A possible way to improve the existing solution is by considering a new constraint which places deadlines on jobs which would make the optimisation even more complex. To take a step further, the scheduling algorithm can use predictive analytics to determine the best candidate among a set of qualified engineers, but that is beyond the scope of this research.

REFERENCES

Acrogenesis.com, 2015, 6.4. What is Local Search (LS)? — OR-Tools User's Manual, [online] Available at: https://acrogenesis.com/or-tools/documentation/user_manual/manual/ls/local_search.html

Arcusfm.com, 2022, *Technology and innovation | Arcus FM*, [online] Available at: <https://www.arcusfm.com/services/technology-and-innovation-services/>

Bellman, R., Esogbue, A.O. and Nabeshima, I., 2014, Mathematical aspects of scheduling and applications: modern applied mathematics and computer science (Vol. 4), Elsevier.

Charlwood, A. and Guenole, N., 2022, Can HR adapt to the paradoxes of artificial intelligence?, Human Resource Management Journal.

Choi, I.C. and Choi, D.S., 2002. A local search algorithm for job shop scheduling problems with alternative operations and sequence-dependent setups, Computers & Industrial Engineering, 42(1), pp.43-58.

Cildoz, M., Mallor, F. and Mateo, P.M., 2021. A GRASP-based algorithm for solving the emergency room physician scheduling problem. Applied Soft Computing, 103, p.107151.

Cook, W.J., 2015, In Pursuit of the Traveling Salesman. Princeton, N.J: Princeton University Press.

De Bruecker, P., Van den Bergh, J., Beliën, J. and Demeulemeester, E., 2015, Workforce planning incorporating skills: State of the art, European Journal of Operational Research, 243(1), pp.1-16.

Dessler, G., 2013, Fundamentals of human resource management, Pearson.

Dissertation.laerd.com, 2021, Principles of research ethics | Lærd Dissertation. [online] Available at: <https://dissertation.laerd.com/principles-of-research-ethics.php>.

docs.optaplanner.org, 2022, Chapter 9. Local search. [online] Available at: https://docs.optaplanner.org/6.0.0.CR5/optaplanner-docs/html/localSearch.html

Frühwirth, T. and Abdennadher, S., 2003, Essentials of constraint programming. Springer Science & Business Media.

Google Developers, 2019, Traveling Salesman Problem | OR-Tools | Google Developers, [online] Available at: https://developers.google.com/optimization/routing/tsp. [Accessed: 03 September 2022].

Harvard Business Review, 2019, What Do We Do About the Biases in AI?. [online] Available at: <https://hbr.org/2019/10/what-do-we-do-about-the-biases-in-ai>.

Hass, K.B., PMP, K.B.H., Wessels, D.J., PMP, D.J.W. and Brennan, K., 2007, Getting it right: business requirement analysis tools and techniques, Berrett-Koehler Publishers.

Hoos, H.H. and Stützle, T., 2004, Stochastic local search: Foundations and applications, Elsevier.

Johnson, G., Whittington, R., Regnér, P., Angwin, D. and Scholes, K., 2020. Exploring strategy. Pearson UK.

Knights, D., Willmott, H. and Brewis, J., 2007, Introducing organizational behaviour and management, London: Thomson.

Maravilha, A.L., Mayra-Pereira, L., and Campelo, F., 2019, Statistical Investigation of Relative Utility of Neighbourhood Structures in Combinatorial Problems

Mendelow, A., 1991, Stakeholder mapping, In: Proceedings of the 2nd International Conference on Information Systems, Cambridge.

Mindtools, 2021. The McKinsey 7-S Framework Ensuring That All Parts of Your Organization Work in Harmony. [online] Mindtools.com. Available at: https://www.mindtools.com/pages/article/newSTR_91.htm.

Mosayebi, M., Sodhi, M. and Wettergren, T.A., 2021, The traveling salesman problem with job-times (TSPJ), Computers & Operations Research, 129, p.105226.

Nolde, N., 2020, Open-Source Routing Engines and Algorithms - An Overview» GIS • OPS. [online], Available at: https://gis-ops.com/open-source-routing-engines-and-algorithms-an-overview. [Accessed: 04 September 2022].

Pathak, S., Jain, A., Gupta, S., Balamurugan, S., Sharma, S. and Duggal, S. eds., 2022, Impact of Artificial Intelligence on Organizational Transformation, John Wiley & Sons.

Peters, T.J., and Waterman, R.H., 1984, In search of excellence, Nursing Administration Quarterly, 8(3), pp.85-86.

Poole, D.L. and Mackworth, A.K., 2017, Artificial Intelligence: foundations of computational agents, Cambridge University Press.

Reeves, M., Moose, S. and Venema, T., 2014, The growth-share matrix, BCG–The Boston Consulting Group.

Rossi, F., Van Beek, P. and Walsh, T. eds., 2006, Handbook of constraint programming, Elsevier.

Ruiz-Torres, A.J., Ablanedo-Rosas, J.H., Mukhopadhyay, S. and Paletta, G., 2019, Scheduling workers: A multi-criteria model considering their satisfaction, Computers & Industrial Engineering, 128, pp.747-754.

Santos, H.G., Toffolo, T.A., Silva, C.L. and Vanden Berghe, G., 2019, Analysis of stochastic local search methods for the unrelated parallel machine scheduling problem. International Transactions in Operational Research, 26(2), pp.707-724.

Serna, N.J.E., Seck-Tuoh-Mora, J.C., Medina-Marin, J., Hernandez-Romero, N., Barragan-Vite, I. and Armenta, J.R.C., 2021, A global-local neighbourhood search algorithm and tabu search for flexible job shop scheduling problem, PeerJ Computer Science, 7, p.e574.

Strohmeier, S., 2020, Digital human resource management: A conceptual clarification, German Journal of Human Resource Management, 34(3), pp.345-365.

Tang, A., Han, J. and Chen, P., 2004, August. A comparative analysis of architecture frameworks, In 11th Asia-Pacific software engineering conference (pp. 640-647), IEEE.

Tang, H., Miller-Hooks, E. and Tomastik, R., 2007, Scheduling technicians for planned maintenance of geographically distributed equipment, Transportation Research Part E: Logistics and Transportation Review, 43(5), pp.591-609.

Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E. and De Boeck, L., 2013, Personnel scheduling: A literature review, European journal of operational research, 226(3), pp.367-385.

BIBLIOGRAPHY

Bahnaier, W.W., 2001, Scheduling guide for program managers, DIANE Publishing.
Berndtsson, M., Hansson, J., Olsson, B. and Lundell, B., 2007, Thesis projects: a guide for students in computer science and information systems, Springer Science & Business Media

GitHub, 2022, valhalla [online] Available at: https://github.com/valhalla/valhalla

GraphHopper Directions API, 2022, GraphHopper Directions API with Route Optimization. [online] Available at: https://www.graphhopper.com/.

LocalSolver, 2022, [online] Available at: https://www.localsolver.com/

Openrouteservice, 2022, [online] Available at: https://openrouteservice.org/.

project-osrm.org, 2022, Project OSRM. [online] Available at: http://project-osrm.org/.

PyPI, 2022, haversine: Calculate the distance between 2 points on Earth, [online] Available at: https://pypi.org/project/haversine/.

vroom-project.org, 2022, VROOM - Vehicle Routing Open-source Optimization Machine. [online] Available at: http://vroom-project.org/.

www.samsara.com, 2022, Samsara. [online] Available at: https://www.samsara.com/uk/

www.verizonconnect.com, 2022, Fleet Management Software and Solutions | Verizon Connect UK. [online] Available at: https://www.verizonconnect.com/uk/

www.westga.edu,1996, Enactment Model. [online] Available at: https://www.westga.edu/~bquest/1996/model.html.
www.youtube.com, Alan Mackworth, 2013, Lecture 15 | CSP 5: Local Search. [online] Available at: https://www.youtube.com/watch?v=5U-9NGzJFZc&list=PLlkMLZIR7yeggS60wuLDg_Dntd2E7xpRC&index=9&ab_channel=AlanMackworth

www.youtube.com, Microsoft Research, 2018, Local Search. [online] Available at: https://www.youtube.com/watch?v=tYBGGRRva5o&list=PLlkMLZIR7yeggS60wuLDg_Dntd2E7xpRC&index=11&ab_channel=MicrosoftResearch

Appendix:

A.1 Setting up Open-Source Routing Engine (OSRM):

This project is heavily reliant on the results generated by an open-source routing engine. It will not be possible to run the programs submitted along with the dissertation without a local server of the engine running in the background.

The following steps will serve as a guide for setting up a local server for the routing engine on a Windows system:

The project uses an older build of the routing engine as the latest versions are only available through Docker images.

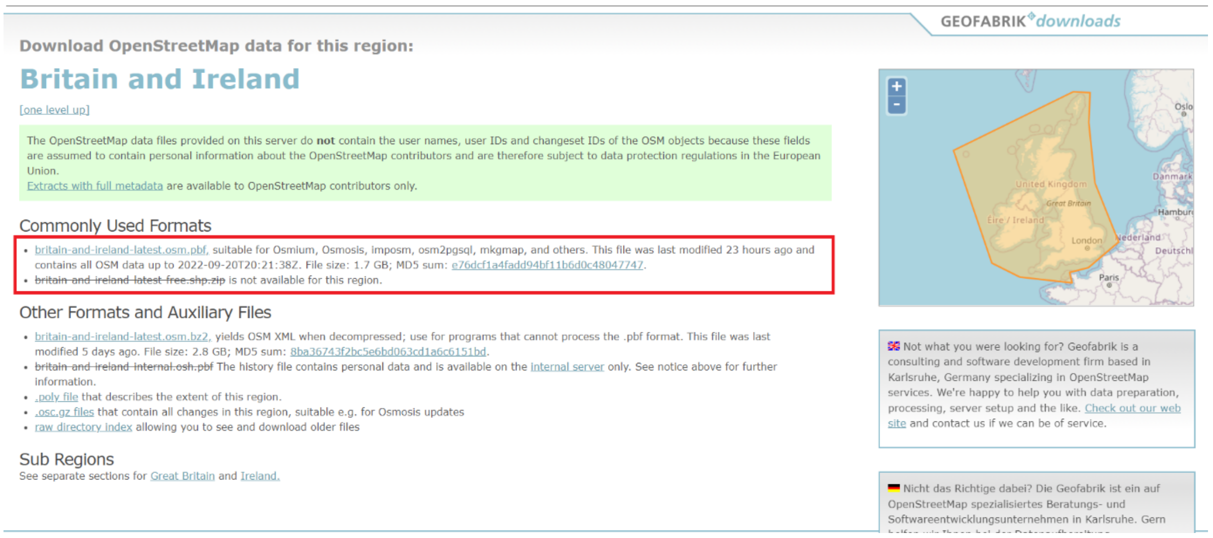
1. The build used in the project can be downloaded from the following link:

   http://build.project-osrm.org/

2. Extract the contents of the downloaded zip file into a folder

3. The project uses OpenStreetMap data of the United Kingdom. The data can be downloaded from the following link:

   https://download.geofabrik.de/europe/britain-and-ireland.html



Download OpenStreetMap data for this region:

Britain and Ireland

[one level up]

The OpenStreetMap data files provided on this server do **not** contain the user names, user IDs and changeset IDs of the OSM objects because these fields are assumed to contain personal information about the OpenStreetMap contributors and are therefore subject to data protection regulations in the European Union.
Extracts with full metadata are available to OpenStreetMap contributors only.

Commonly Used Formats

- britain-and-ireland-latest.osm.pbf, suitable for Osmium, Osmosis, imposm, osm2pgsql, mkgmap, and others. This file was last modified 23 hours ago and contains all OSM data up to 2022-09-20T20:21:38Z. File size: 1.7 GB; MD5 sum: e76dcf1a4fadd94bf11b6d0c48047747.
- ~~britain-and-ireland-latest-free.shp.zip~~ is not available for this region.

Other Formats and Auxiliary Files

- britain-and-ireland-latest.osm.bz2, yields OSM XML when decompressed; use for programs that cannot process the .pbf format. This file was last modified 5 days ago. File size: 2.8 GB; MD5 sum: 8ba36743f2bc5e6bd063cd1a6c6151bd.
- ~~britain-and-ireland-internal.osh.pbf~~ The history file contains personal data and is available on the internal server only. See notice above for further information.
- .poly file that describes the extent of this region.
- .osc.gz files that contain all changes in this region, suitable e.g. for Osmosis updates
- raw directory index allowing you to see and download older files

Sub Regions

See separate sections for Great Britain and Ireland.

Not what you were looking for? Geofabrik is a consulting and software development firm based in Karlsruhe, Germany specializing in OpenStreetMap services. We're happy to help you with data preparation, processing, server setup and the like. Check out our web site and contact us if we can be of service.

Nicht das Richtige dabei? Die Geofabrik ist ein auf OpenStreetMap spezialisiertes Beratungs- und Softwareentwicklungsunternehmen in Karlsruhe. Gern

Figure 17: Screenshot of the link above

4. Copy the britain-and-ireland-latest.osm.pbf file into the folder where the OSRM build was extracted.

5. Launch a command prompt and navigate to the OSRM build folder (cd).

The routing engine uses two pre-processing pipelines:

- Multi-Level Dijkstra (MLD)
- Contraction Hierarchies (CH)

Multi-Level Dijkstra is used if live data such as traffic or weather conditions would have to be considered while creating a route. Contraction Hierarchies does not consider such parameters while routing which allows it to generate faster responses. The initial build of the routing engine used the MLD pipeline, but it was unable to generate routes in several instances. Therefore, in the end, the CH pre-processing pipeline was used to prepare the OSM data.

6. Extract the OSM data for a travel profile set for cars using the following command:

   osrm-extract britain-and-ireland-latest.osm.pbf -p car.lua

7. Set the Contraction Hierarchies algorithm using the following command:

   osrm-contract britain-and-ireland-latest.osrm

If the following steps were followed correctly then a local server of OSRM has been successfully set up. The server can be launched by running the following command from the folder in which the server was set up:

osrm-routed britain-and-ireland-latest.osrm

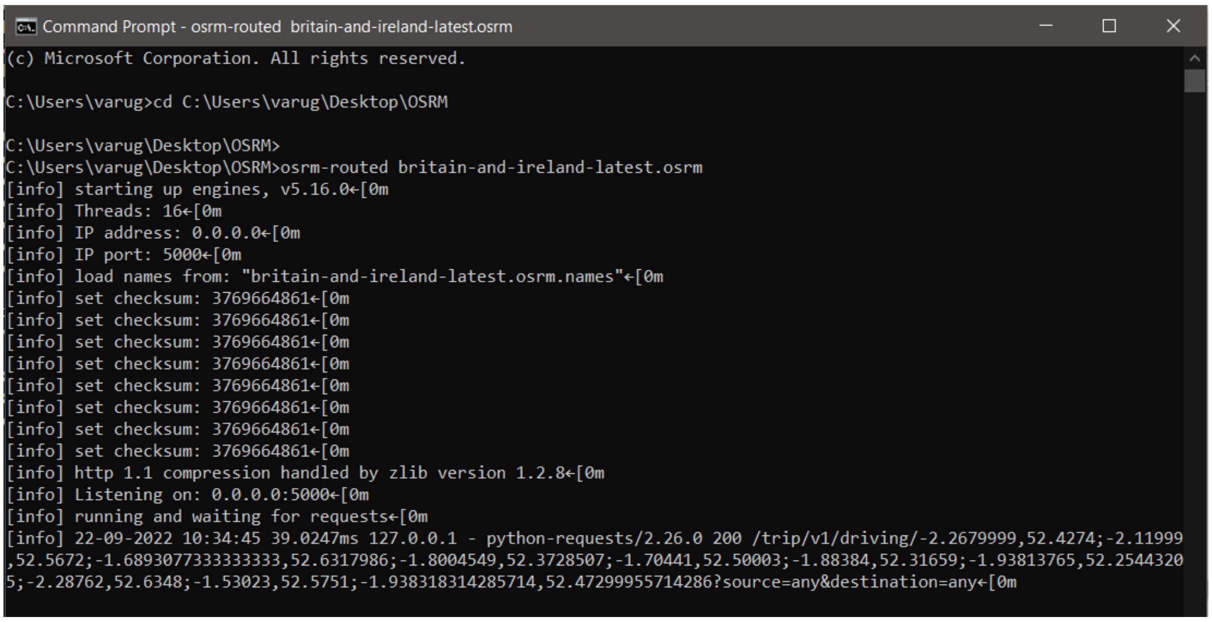
The server can be accessed through localhost IP address :127.0.0.1 in port 5000.

The following format must be used when sending requests to the server:

/{service}/{version}/{profile}/{coordinates}[.{format}]?option=value&option=value

An example request is as follows:

http://127.0.0.1:5000/trip/v1/driving/-2.2679999,52.4274;-2.11999,52.5672;-1.6893077333333333,52.6317986;-1.8004549,52.3728507;-1.70441,52.50003;-1.88384,52.31659;-1.93813765,52.25443205;-2.28762,52.6348;-1.53023,52.5751;-1.938318314285714,52.47299955714286?source=any&destination=any


Command Prompt - osrm-routed britain-and-ireland-latest.osrm

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\varug>cd C:\Users\varug\Desktop\OSRM

C:\Users\varug\Desktop\OSRM>
C:\Users\varug\Desktop\OSRM>osrm-routed britain-and-ireland-latest.osrm
[info] starting up engines, v5.16.0←[0m
[info] Threads: 16←[0m
[info] IP address: 0.0.0.0←[0m
[info] IP port: 5000←[0m
[info] load names from: "britain-and-ireland-latest.osrm.names"←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] set checksum: 3769664861←[0m
[info] http 1.1 compression handled by zlib version 1.2.8←[0m
[info] Listening on: 0.0.0.0:5000←[0m
[info] running and waiting for requests←[0m
[info] 22-09-2022 10:34:45 39.0247ms 127.0.0.1 - python-requests/2.26.0 200 /trip/v1/driving/-2.2679999,52.4274;-2.11999
,52.5672;-1.6893077333333333,52.6317986;-1.8004549,52.3728507;-1.70441,52.50003;-1.88384,52.31659;-1.93813765,52.2544320
5;-2.28762,52.6348;-1.53023,52.5751;-1.938318314285714,52.47299955714286?source=any&destination=any←[0m
```

Figure 18: OSRM Local Server taking Requests

More details are available in the OSRM documentation webpage:

http://project-osrm.org/docs/v5.24.0/api/#general-options

The latest build of the engine is available in the project's GitHub page in the form of Docker Images. The following links will have instructions on how to set up Docker and host an OSRM local server through it:

Docker Installation for Windows:

https://docs.docker.com/desktop/install/windows-install/

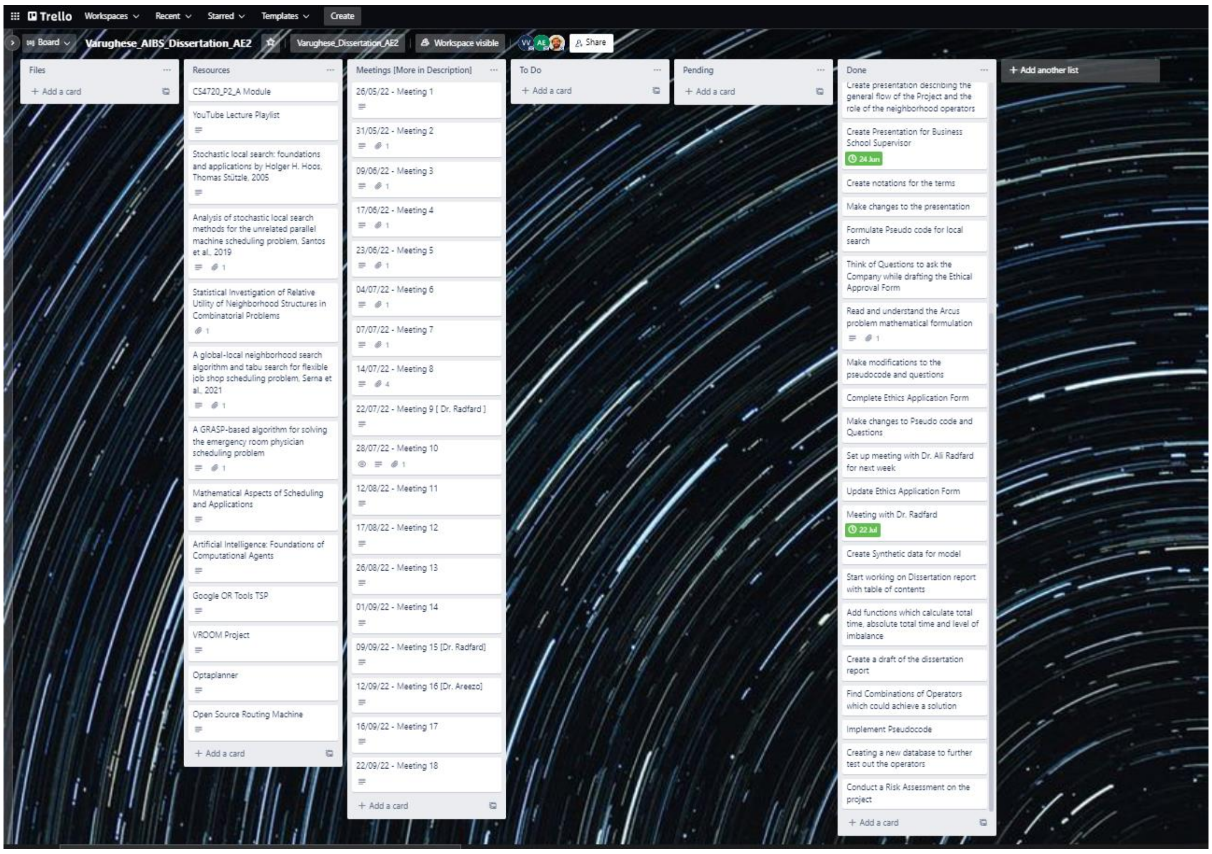
Ubuntu:

https://docs.docker.com/engine/install/ubuntu/

OSRM GitHub Page:

https://github.com/Project-OSRM/osrm-backend

A.2 Trello Board



Files
+ Add a card

Resources
- CS4720_P2_A Module
- YouTube Lecture Playlist
- Stochastic local search: foundations and applications by Holger H. Hoos, Thomas Stützle, 2005
- Analysis of stochastic local search methods for the unrelated parallel machine scheduling problem, Santos et al., 2019
- Statistical Investigation of Relative Utility of Neighborhood Structures in Combinatorial Problems
- A global-local neighborhood search algorithm and tabu search for flexible job shop scheduling problem, Serna et al., 2021
- A GRASP-based algorithm for solving the emergency room physician scheduling problem
- Mathematical Aspects of Scheduling and Applications
- Artificial Intelligence: Foundations of Computational Agents
- Google OR Tools TSP
- VROOM Project
- Optaplanner
- Open Source Routing Machine

+ Add a card

Meetings [More in Description]
- 26/05/22 - Meeting 1
- 31/05/22 - Meeting 2
- 09/06/22 - Meeting 3
- 17/06/22 - Meeting 4
- 23/06/22 - Meeting 5
- 04/07/22 - Meeting 6
- 07/07/22 - Meeting 7
- 14/07/22 - Meeting 8
- 22/07/22 - Meeting 9 [Dr. Radfard]
- 28/07/22 - Meeting 10
- 12/08/22 - Meeting 11
- 17/08/22 - Meeting 12
- 26/08/22 - Meeting 13
- 01/09/22 - Meeting 14
- 09/09/22 - Meeting 15 [Dr. Radfard]
- 12/09/22 - Meeting 16 [Dr. Areezo]
- 16/09/22 - Meeting 17
- 22/09/22 - Meeting 18

+ Add a card

To Do
+ Add a card

Pending
+ Add a card

Done
- Create presentation describing the general flow of the Project and the role of the neighborhood operators
- Create Presentation for Business School Supervisor 24 Jun
- Create notations for the terms
- Make changes to the presentation
- Formulate Pseudo code for local search
- Think of Questions to ask the Company while drafting the Ethical Approval Form
- Read and understand the Arcus problem mathematical formulation
- Make modifications to the pseudocode and questions
- Complete Ethics Application Form
- Make changes to Pseudo code and Questions
- Set up meeting with Dr. Ali Radfard for next week
- Update Ethics Application Form
- Meeting with Dr. Radfard 22 Jul
- Create Synthetic data for model
- Start working on Dissertation report with table of contents
- Add functions which calculate total time, absolute total time and level of imbalance
- Create a draft of the dissertation report
- Find Combinations of Operators which could achieve a solution
- Implement Pseudocode
- Creating a new database to further test out the operators
- Conduct a Risk Assessment on the project

+ Add a card

Figure 19: Screenshot of the Trello Board

The Trello board was extensively used to keep track of the progress made in the dissertation project and served as an objective board. Details of meetings between the project supervisors were also recorded on Trello.

The board can be accessed through the following link:

https://trello.com/invite/b/XDJVA7N7/810761d62da2854e7ecd1d36fb6380f2/varugheseaibsdissertationae2