DYNAMIC PRICING SERVICES TO MINIMISE CO_2 EMISSIONS OF DELIVERY VEHICLES

by

Yizi Zhou

under the supervision of

Dr Rupal Rana and Professor Jiyin Liu

Doctoral Thesis

Submitted in partial fulfilment of the requirements
for the award of
Doctor of Philosophy of Loughborough University
Feburary 28, 2018

Acknowledgement

First and foremost, I offer the sincerest gratitude to my supervisor Professor Jiyin Liu, without whom this work is impossible. Jiyin, who is extremely kind and smart, offered me help on research ideas, mathematical programming and guidance on algorithms, and helped me proofread my thesis even when he was very busy. I would also like to thank my supervisor Dr Rupal Rana, and she encouraged me to overcome difficulties and also offered insights into my research and proofread my thesis. One simply could not wish for a better or friendlier supervisor. I also offer enormous gratitude to my industry supervisor and a good friend Dr Anne Liret, who is a very encouraging and warm person. Anne taught me a lot of things, which are useful for my research and for my career. She introduced me to the brilliant world of software engineering and helped me to link my research to real-life problems. I would like to express my gratitude to my father, my mother and my grandma who always supported me mentally and financially. I am deeply grateful to my boyfriend Dr Zhixin Cao for his encouragement and support. His patience and smartness helped me overcome my numerous difficulties, especially mental difficulties.

Abstract

In recent years, companies delivering goods or services to customers have been under increasing legal and administrative pressure to reduce the amount of CO_2 emissions from their delivery vehicles, while the need to maximise profit remains a prime objective. In this research, we aim to apply revenue management techniques, in particular incentive/dynamic pricing to the traditional vehicle routing and scheduling problem while the objective is to reduce CO_2 emissions. With the importance of accurately estimating emissions recognised, emissions models are first reviewed in detail and a new emissions calculator is developed in Java which takes into account time-dependent travel speeds, road distance and vehicle specifications. Our main study is a problem where a company sends engineers with vehicles to customer sites to provide services. Customers request for the service at their preferred time windows and the company needs to allocate the service tasks to time windows and decide on how to schedule these tasks to their vehicles. Incentives are provided to encourage customers choosing low emissions time windows. To help the company in determining the schedules/routes and incentives, our approach solves the problem in two phases. The first phase solves time-dependent vehicle routing/scheduling models with the objective of minimising CO_2 emissions and the second phase solves a dynamic pricing model to maximise profit. For the first phase problem, new solution algorithms together with existing ones are applied and compared. For the second phase problem, we consider three different demand modelling scenarios: linear demand model, discrete choice demand model and demand model free pricing strategy. For each of the scenarios, dynamic pricing techniques are implemented and compared with fixed pricing strategies through numerical experiments. Results show that dynamic pricing leads to a reduction in CO_2 emissions and an improvement in profits.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Carbon dioxide (CO_2) is the second largest abundance greenhouse gases (GHG), and therefore its increase in the atmosphere has caused environmental concerns in two major ways. The first concern is that CO_2 emissions directly contribute to global warming. Professor Pachauri, the 2007 Nobel Peace Prize winner, said that the doubling of CO_2 concentrations in the atmosphere compared to pre-industrial levels could potentially contribute towards a global temperature rise between 1.5 °C and 4.5 °C (Harrabin, 2013). The excess CO_2 emissions from human activities have already increased ocean temperatures, melted snow and ice, raised sea levels and increased climatic extremes. According to an article on the National Geographic website (National Geographic, nd), effects and possible future effects of global warming caused by GHG emissions include ice melting, the population of Adelie penguins on Antarctica decreasing dramatically, some butterflies, foxes, and alpine plants needing to immigrate to more northern or to higher, cooler areas, increased worldwide precipitation (rain and snowfall), and bugs booms. If global warming continues at its current rate, it is anticipated that floods and droughts will happen more often, hurricanes and tsunami will become stronger, diseases will spread and ecosystems will also change. The second major concern is that CO_2 emissions could also cause an increase in ocean acidification since it dissolves in water to form carbonic acid. Ocean acid will lead to a mass extinction of some species, as 24 million tons of CO_2 emissions resulting from industrial activities are absorbed into the sea every day (BBC News, 2014).

As the effects of increases in CO_2 emissions in the atmosphere can lead to major problems, companies are now under pressure to reduce the level of CO_2 emissions associated with their operations enforced by new legislation and policies. As an outcome of the Paris climate conference (COP21) in December 2015, 195 countries all over the world agreed on the first-ever universal, legally binding global climate deal, in which the European Union (EU)'s target is to reduce emissions by at least

40% by 2030. The Kyoto Protocol, an international agreement on Climate Change, commits its Parties to work towards an internationally binding target to reduce emissions (Breidenich et al., 1998). During the second commitment period from the 1st January 2013 to the 31st December 2020, 37 countries and the European Community are required to reduce GHG emissions by at least 18% by the year 2020 in comparison to the level of 1990. The Climate Change Act of 2008 legally commits the UK to reducing emissions by at least 80% by 2050 in relation to the level of 1990 (Giddens, 2009). Among GHG, CO_2 emissions is the most concerning one as it can lead to global warming and directly threaten people's health. Emissions levels of individual companies will also influence the company's reputation and customers' perceptions.

Policies that are directly related to a company's profitability involve carbon taxation. A carbon tax is a tax levied on the carbon content of fuels, which is a form of carbon pricing and trading. It is one potential way to reduce GHG emissions, as burning fuels accounts for a large amount of total GHG emissions from human activities. Many countries such as the Organisation for Economic Co-operation and Development (OECD) countries have implemented carbon taxes, but most emissions-related taxes are levied on energy products or vehicles, rather than CO_2 emissions directly. This may be because it is hard to measure emissions simultaneously. In the UK, car holders are required to pay vehicle tax each year based on a fixed vehicle rate rather than total emissions car holders produced when they use their cars. The rate of vehicle tax for cars depends on engine size and carbon dioxide emissions rate. For example, petrol and diesel cars registered after 1st March 2001 with CO_2 emissions rate of 166 g/km-175 g/km, which are band H cars, will pay £205 annually (GOV.UK, 2017). On 1st January 2014, Mexico City imposed a tax on several types of fossil fuels, including gasoline, averaging $3 per ton of CO_2. However, the estimations of the amount of CO_2 are highly uncertain. With technology development, humans can more accurately measure the exact amount of CO_2 emissions from daily activities. A direct tax on CO_2 emissions produced can therefore be put into force in the near future. Even without the carbon tax, there is an economic cost associated with CO_2 emissions as the amount of CO_2 emissions is directly related to energy consumption and the company's goodwill. Also, congestion pricing is used in some cities where a higher price for public roads usage is applied at peak hours to encourage a more efficient use of public roads, to smooth traffic and to reduce emissions. This may encourage companies to also smooth their delivery schedules. Hence emissions cost could now be taken into consideration into the profit calculation by delivery companies.

Companies that want to reduce CO_2 emissions first need to understand where these can be emitted from. The Intergovernmental Panel on Climate Change WMO and UNEP (2007) has concluded that the main contributor to increased CO_2 levels in the atmosphere is the burning of fossil fuels. In 2016, the UK transportation accounted for 32% of all the UK greenhouse gas emissions, with around 90% of all transport emissions coming from road transportation according to Department for Business, Energy & Industrial Strategy (2016). The situation was similar in the United States, where road transportation accounted for 78% of the emissions produced by all transportation modes, and the percentage of total GHG emissions from transportation rose from 24.9% to 27.3% between 1990 and 2005 (Ohnishi, 2008). Therefore, to reduce emissions levels to meet this legislation, it is important to pay attention to the road transportation sector. In Zanni and Bristow (2010)'s study on CO_2 emissions from London transportation, they cited several regulations related to reducing emissions in the London area. The Greater London Authority launched a plan, which contained the details of policy initiatives aimed at reducing the city's carbon emissions. This plan demonstrated the necessity of a 60% reduction in CO_2 emissions in London by 2050 and identified transport as one of the key areas where significant CO_2 emissions savings must be made (Greater London Authority, 2007). In 2008, Transport for London released the London Freight Plan, which set a target of saving 0.7 million tons of CO_2 per year from the road transportation sector by the year 2025 (Transport for London, 2008).

It is important to find means to reduce CO_2 emissions in road transportation through the company's daily operations, e.g. in their vehicle routing and planning. The traditional vehicle routing problems (VRP) either in theory or in practice mainly have the objective to minimise total travel distance or travel time or the number of vehicles used. CO_2 emissions minimisation is hence a new objective to be considered. Although green logistics has gained increasing awareness from the governments and service provisioning companies, recognising that the traditional distribution and production logistics is not sustainable in the long term, challenges in such transformation are firstly the design of operationally efficient and accurate approach of CO_2 emissions calculation and secondly the smooth adoption of CO_2 awareness in planning decisions across actors (Zhou et al., 2017). Based on empirical studies, various models have been developed to more accurately estimate the amount of CO_2 emissions generated from vehicles. CO_2 emissions are affected by travel speed, travel distance, vehicle types and many other factors, where speed plays an important role, hence in literature, green vehicle routing and scheduling problem mostly consider a time-dependent travel speed setting. Then having access to data of the real-life travel speed at different times on different roads is vital for an accurate estimation of CO_2 emissions. A speed profile based on historical data is necessary for vehicle routing and scheduling planning purpose, for post analysing this profile the real-life travel speed could be read from telematics devices installed on vehicles. Also, road distance between locations and vehicle types are needed to calculate the emissions of a route. Different vehicle types can have different emissions models as there are slightly different ways to categorise them based on the fuel type, vehicle size and age. The vehicle routing problem itself is a non-deterministic polynomial-time (NP)-hard problem which is difficult to solve, and industrial applications demand a solution algorithm to solve the problem efficiently and within a reasonable time. Research has been carried out on solution algorithms such as heuristics, metaheuristics and hybrid methods, not only for green VRP but for all variants of VRP.

Reducing emissions is not only a task for companies, it is the responsibility of the customers as well. The environmental concerns among the public are increasing every year, and the demand for green consulting services and solutions are fast growing. Based on Blue & Green Tomorrow (2017), the green business sector generated £12 billion of revenue in the UK alone. However, most companies are making enough revenue and therefore have a relatively small appetite for green services or solutions. They have doubts about whether green brand perception is worth the efforts. Hemsley (2014) stated that 95% of individuals value the importance of protecting the environment when they make purchase decisions and it is often a consideration secondary to price according to a study by TNS global. Customers also think that the effectiveness of green products or services should be the same as their counterpart is less eco-friendly options. There are even customers who are willing to pay more for environmentally friendly products or services, and Laroche et al. (2001) profiled those types of customers to be females, married and with at least one child living at home. As a summary, companies should make efforts to be perceived as a green brand and this strategy could be beneficial.

Supermarkets which provide online grocery shopping and home delivery services often encounter with the problem of sustainable delivery. You may have experienced that when you check out after shopping online the website will ask you to book a delivery time window normally one-hour or two-hour in length (e.g. Tesco, Sainsbury) and some supermarkets offer a fixed price for all time windows but the majority apply differentiated or dynamic pricing to different delivery options. Take Sainsbury as an example, the delivery charge is between £1 and £7 for orders over £40 and £7 for orders under £40. In order to reduce the company's carbon footprint and save fuel, Sainsbury now provides new eco-friendly green time windows and they are marked with a green van icon. Those options are available during non-congested/off-peak hours and the prices for those options are observed to be cheaper. After customers choose this time window, they will receive an acknowledgement message in thanks for booking an environmentally friendly delivery time window. In this way, protecting the environment is a cooperation job of both the business and customers. Inspired by this idea, revenue management techniques could be applied to green delivery services which will help the business reduce CO_2

emissions by cooperating with their customers.

1.2 Aims of the research

The main aim of the research is to apply revenue management techniques, in particular incentive/dynamic pricing to the traditional vehicle routing and scheduling problem when the objective is to reduce CO_2 emissions for a field service providing company. There are other greenhouse-effect gases whose emissions also contribute to global warming. In this thesis, the focus is on CO_2 emissions only. However, the methodology may be applied to all these gases. The emissions in such cases can be measured in CO_2 equivalent units. To achieve this goal, we plan to carry out a series of approaches.

CO_2 emissions are directly related to fuel usage, so saving emissions means saving fuel cost and increasing the company's profitability. This can be any company that provides installation and maintenance services at customer sites, for example, a telecommunication company. The service is provided at the customer site by engineers, each with a vehicle. Normally, customers are required to be physically present during the service for safety and security reasons. It is similar to the grocery delivery service in that it requires customers to be physically present. Customer orders arrive dynamically through the company's booking website or telephone centre. A customer will book a time window for receiving the service. A time window is a period of time within which the service must start, e.g., a two-hour time window such as 9:00am-11:00am. We plan to schedule the service tasks and the vehicle travelling to reduce CO_2 emissions whilst maximising profits by influencing customers' choice of time windows through dynamic pricing instead of fixed price. The price will reflect the cost of the delivery option, which is calculated using the cost of CO_2 emissions, as the cost of engineer wages is fixed, the only cost varying with a delivery option is emissions cost or fuel cost. A cost value is assigned to each unit of CO_2 emissions. Environmental sustainability is a concern for all, hence the discounted delivery windows will be tagged as green/environmentally friendly windows showing the amount of emissions that can be saved by choosing such options, and this may further influence customers to choose these time windows. The problem considered here is common for services in which there are many competitors and customer retention is important. For example, this may apply to home delivery or pick-up services, as well as installation or maintenance services. This research develops a general solution framework with two stages for this type of problems. The first phase deals with the scheduling and routing decisions. The second phase makes pricing decisions using the emissions costs obtained in the first phase. This framework is applied throughout the whole research for different problem settings or scenarios.

It is important to understand how to estimate CO_2 emissions from vehicles and what are the factors that affect CO_2 emissions. There are several emissions models based on literature reviews. This thesis plans to compare those models and choose one that is relatively accurate and does not require data that is difficult to obtain. A related question can then be raised - what is the difference between travel time or travel distance objectives with the CO_2 objective? We plan to justify that there is a conflict between these traditional objectives and CO_2 objective, so that this research is different from those investigating vehicle routing problems to reduce travel time or travel distance.

As traffic conditions have a significant influence on CO_2 emissions, the amount of emissions generated per travel is modelled as being time-dependent to capture congestion patterns, as well as the travel time per travel. In order to capture the real-life traffic conditions and to build a speed profile for a geometric area, travel data of various real-life journeys are collected to compute the travel speed for each hour. We plan to build a CO_2 emissions profile for each local area of business. Data collection will be carried out for this research from a field service provider. Travel journals of engineers will be collected. In total, 121 days engineer travel journals, from 2nd July 2016 to 10th November 2016, covering five geometric areas (domains) in the UK. An hourly travel speed profile is constructed for these five domains, based on the travel speed profile we will convert this into an hourly emissions rate profile, which is new in literature.

Travel distance is another common factor that affects CO_2 emissions. Several options can be chosen to estimate this, for example, an adjusted version of straight distance between two locations, or calling some web services to get the road distance, such as Google Maps APIs. Using Google maps may be a problem for an online industrial sized problem as each web service call will cost a small amount of time, so in this case, it will be necessary to have an estimation function of road distance. In this research, we plan to provide an algorithm to estimate distance using latitude and longitude and compare the estimations with the distance provided by Google Maps, to demonstrate its accuracy. The proposed approach is to generate a data sample of road distance between pairs of locations using Google Maps APIs. Basic linear regression techniques will be applied to get a linear model to estimate distance.

Information on the vehicle types can be obtained from the company as they have the number plate and vehicle model information of all their vehicles. With this information, we will build an application to automatically map all these characteristics to translate to a suitable emissions function. With all the above functionalities (i.e., speed profile, distance estimation and vehicle type), we plan to develop a concrete CO_2 emissions calculator for vehicle routing problems, which is a stand-alone Java application.

In this thesis, the vehicle routing and scheduling problem with time-dependent travel speed and time window constraints and its variants are considered, where the objective is to generate the minimum CO_2 emissions route plans. We plan to construct new mathematical models for different scenarios and problem settings, and solve those models with the software Xpress Optimiser first. However exact method can only solve small sized problems to optimality and the computational time can be quite long. The advantages of using exact methods are to test the correctness of the mathematical formulation, and it provides global optimal solutions which will work as benchmarks for other solution methods for small test cases.

As the problem of interest is an online pricing problem, customers will not be willing to wait for minutes to get a response. This requires a fast and efficient solution algorithm. Metaheuristics methods are often applied to this type of problems. For the first phase time-dependent vehicle routing and scheduling problem with time window constraints, different metaheuristics methods will be implemented and compared on real-life test cases. They are tabu search algorithms with random neighbourhood generators, variable neighbourhood search, two variants of variable neighbourhood search (VNS) algorithms: variable neighbourhood descent (VND) and reduced variable neighbourhood search (RVNS), and simulated annealing (SA) and its variants. In this thesis, we plan to compare the performance of those metaheuristics on our test cases and for small test cases with global optimal solutions given by software solvers.

For time-dependent VRP, there is a sub-problem of the optimal start travel to the next customer time once the engineer finishes the job at the current customer site. Traditionally, vehicles will start travelling to the next customer immediately after the job finishes. For time-dependent settings, an engineer may wait at a customer site for a short while to avoid congestion period. We plan to develop a dynamic programming algorithm to solve the sub-problem and optimise the start travel time of each journey. To reduce the computational time and boost the performance of algorithms for online problem, parallel computing could be used. Modern PCs have a parallel processor structure with several computing cores that support multi-threading and task parallelism. As the computational complexity of real-life VRPs and parallel nature of the metaheuristics method apply in our case, it is beneficial to use parallel computing to boost system performance.

Several different pricing models will be considered. There are various ways to model customer demand, and one way is a linear demand model. Based on the customer's first preference, the probabilities for the customer to choose each time window are estimated by the triangular distribution. Another way is to use a discrete choice model which has been widely studied and applied in customers' choice of different transportation modes (Li, 2011). For non-linear optimisation problems, Lingo optimiser provides a generalised reduced gradient (GRG) algorithms. Newton-based algorithms and its modifications are commonly used to solve nonlinear optimisation problem, as well as searching based heuristics such as the differential evolutionary algorithm. This research plans to provide different ways to model customer's demand for online service time window booking for different real-life situations. Like the first phase model, comparisons between algorithms are investigated, and the most suitable algorithm should be fast and relatively good in solution quality. Numerical experiments are carried out to test the performance of dynamic pricing policies; a reduction in CO_2 emissions and an improvement in profits are expected. To test the performance of the models the experiments are replicated numerous times as probabilistic demand model are used, and there are uncertainties when simulating customers' choices.

We will also explore a demand model free dynamic pricing policy called incentive sharing policy for the problem when the historical information of customer purchasing behaviour is limited. In a similar booking process as before, a customer makes a request online or by phone but at least one day before the service day instead of on the same day. The routing and scheduling decisions are determined the night before or at the beginning of the service day. Instead of providing a list of available options and their corresponding prices, customers are asked to choose a preferred time window and then be asked if they would like to participate in the green delivery program, by which they will provide alternative time windows too, either on the same day or another day. Incentives will be given to customers who are flexible to participate in the green delivery option. The delivery time window will be confirmed on the day of the delivery or the night before. This is similar to real life practice. The incentives are calculated as the additional profit gained from using this approach, compared to traditional practice where customers are all served according to their initial preference. Furthermore, incentives are equally shared among customers who participate in the green delivery program and those that do not will receive service at their initial preferred time windows. For customers who participate in the program but are scheduled for service at their initial preference, incentives will not be rewarded. Simulations are implemented to measure the benefits of the incentive sharing policy for green delivery services.

1.3 Overview of the thesis

This thesis contains eight chapters. This chapter serves as an introduction of background information highlighting the importance of reducing CO_2 emissions in companies' road transportation operations, with the pressure of laws and regulations and as well as the increasing customers' perceptions of green products or services. There is a direct relationship between CO_2 emissions and fuel consumptions, unlike other types of emissions. So saving CO_2 emissions is also saving fuel and money. Studies also show that the majority of customers value the importance of environment when making purchase decisions just second to price and there are customers even willing to pay more for greener products or services. These all support the ideas of using dynamic pricing to shift customer demand when they choose delivery time window online, and reduction in emissions and improvement in profits are expected. This chapter also describes the data collection and outlines the approaches to address the problem in this thesis as well as solution methods.

In Chapter 2, literature related to this research is reviewed. It includes the research topics around green vehicle routing problems which are mainly time-dependent vehicle routing problem, dynamic programming algorithm on optimising start travel time, solution algorithms of green vehicle routing problems, CO_2 emissions models, and revenue management in time slots booking.

In Chapter 3, the critical elements of the tool for calculating CO_2 emissions are demonstrated, including speed profiles, road distance estimation algorithm and vehicles' details such as vehicle type and fuel type etc. mapping to emissions formulas. The accuracy of the emissions calculator is tested through road trips and comparing with mechanical information provided by vehicle manufacturers. In this chapter, the objectives of minimising travel time and the objectives of minimising emissions are compared, and conflicts are identified under specific scenarios. The business impact of CO_2 emissions calculation is also demonstrated.

In Chapter 4, we propose a new approach to the problem which applies low-emissions vehicle scheduling techniques with dynamic pricing to reduce CO_2 emissions and maximise profit for service delivery booking. When a customer requests for service with a preferred time window, the company will provide the customer with different service time window options and their corresponding prices. Incentives are included in the prices to influence the customer's choice to reduce CO_2 emissions. To help the company in determining the incentives, our approach solves the problem in two phases. The first phase solves time-dependent vehicle scheduling models with the objective of minimising CO_2 emissions and the second phase solves an improved dynamic pricing model to maximise profit. A linear demand model is considered in this chapter. The approach will be tested through numerical experiments using software solver.

In Chapter 5, we discuss and determine the solution algorithms for the first phase problem to be used in this research by comparing the state of art solution algorithms in literature.

In Chapter 6, we extend the problem discussed in Chapter 4 to a vehicle routing problem with real road distance. Once a new customer arrives, the system first solves a time-dependent vehicle routing problem with the objective of minimising CO_2 emissions (GTDVRP) for each time window that is available. We propose a new mixed integer linear programming (MILP) formulation of the GTDVRP. It will be solved by using a parallel variable neighbourhood search heuristic with dynamic programming procedure to optimise the start travel time. Then for each available option, the minimum additional cost for including this customer will be calculated. In the second phase, a price will be calculated for each option by solving a discrete choice pricing model based on the additional costs calculated using phase one model. The pricing model is a non-linear programming model, and the demand function follows a multinomial logit (MNL) model. The MNL model will be solved by differential evolution method. The whole booking system supports a time rolling mechanism. And the solution engine takes a parallel variable neighbourhood search algorithm. The solution framework is tested through one-month simulation experiments. By applying dynamic pricing techniques compared to the fixed price strategy, reduction in CO_2 emissions and improvement in profits are observed.

In Chapter 7, a demand model free incentive sharing pricing policy for green delivery service is proposed. This problem can be formulated as a multiple time windows VRP as in a MILP format. Two scenarios are considered, one with all time windows on the same day and the other with the preferred time window and alternatives being on different days. Two new MILP models are formulated accordingly and solved by Xpress optimiser for small test cases. Simulations are carried out for bigger size test cases using a self-adaptive simulated annealing algorithm. Benefits of this policy are measured by CO_2 emissions, total profits and other traditional performance measures such as travel distance, and travel time.

Chapter 8 gives conclusions, summarises contributions and indicates future research topics.

2 Literature review

This research aims to explore applying dynamic pricing techniques to shift customers demand towards greener delivery options, e.g. a less congested delivery time window, for online service booking. Therefore two main streams of literature need to be reviewed. The first stream is literature on green scheduling and routing problems, and models used to estimate CO_2 emissions. The second stream is literature on revenue management or dynamic pricing theory and its applications, especially in the area of online retailing or online service booking.

The rest of chapter is organised as follows. Section 2.1 reviews literature of green vehicle routing and scheduling problem, including vehicle routing problem with time windows (VRPTW), CO_2 emissions minimisation time-dependent VRPTW (TDVRPTW), multiple time windows vehicle routing problem and departure time optimisation in TDVRPTW. The state of the art solution algorithms for VRPTW are reviewed in Section 2.2. The methods to estimate CO_2 emissions from vehicles are introduced in Section 2.3. Section 2.4 reviewed articles applying revenue management techniques in vehicles delivery. Finally, concluding remarks are given in Section 2.5.

2.1 Green vehicle routing and scheduling

As regulations related to environmental issues are increasing in prevalence, the concept of Green Logistics is playing an essential role in the transportation operations of companies (Sbihi and Eglese, 2007). Initial works in this area (e.g., Apaydin and Gonullu, 2008; Ubeda et al., 2011, 2014; Li et al., 2015) have primarily focussed on reducing the total distance travelled to reduce the overall emissions. However, CO_2 emissions are not only influenced by travel distance, but by many other factors, such as road characteristics, vehicle speeds and load. In the rest of the thesis, emissions, carbon dioxide emissions and CO_2 emissions will be used interchangeably to mean the same thing unless stated otherwise. According to Eglese and Black (2010), the travel speed is an important factor that should be considered. During times of congestion, vehicles generate much more emissions than when moving at the free-flow speed. Vehicle acceleration and deceleration also increase emissions levels, and it is important to measure time-dependent travel speeds to capture congestion situations. Palmer (2007) stated that congestion period generates the highest level of CO_2 emissions.

Congestion is defined in urban mobility study by Texas Transport Institute as vehicles drive at a much slower speed than road speed limit, which is caused by heavy traffic, or narrow roadways, or fewer lanes in operation due to road construction or accident (Barth and Boriboonsomsin, 2008). In this study, we only consider the first situation, because the later ones are unpredictable and happen occasionally. In urban areas, there are usually two periods where road users experience heavy traffic and congestions on major roads, and they are called peak hours or rush hours. They are when people travel to work or back from work. Therefore, time-dependent vehicle routing problem (TDVRP) emerges. It is much like traditional vehicle routing problems (VRP) but the travelling times between nodes depend on the time of the day, such as in the cases of peak and off-peak times.

2.1.1 VRPTW and its variants

Traditional VRP focuses on the economic impacts of vehicle routes, meaning that the routes are designed to minimise total distance travelled, minimise total travel time, or minimise the number of vehicles used. The basic problems in vehicle routing class include the VRP with capacity, the VRP with time windows, the VRP with Backhauls and the VRP with pickup and delivery (for details refer to Toth and Vigo, 2014). In this thesis, we will be focusing on the VRP with time windows and its variants. The graph theoretic problem definition is as follows: Let $G = (V, A)$ be a complete graph, where $V = \{v_0, \ldots, v_n\}$ is the vertex set and $A = \{(v_i, v_j)|v_i, v_j \in V, i \neq j\}$ is the arc set. Vertices $v_1, \ldots, v_n$ represent customers, and each customer i has a non-negative demand d_i and a time interval $[a_i, b_i]$. There is also a service time s_i occurs at each customer i. v_0 represents the depot and $d_0 = 0$. Each arc in A is associated with a non-negative cost C_{ij}, which represents the cost of travelling from vertex v_i to vertex v_j. In this thesis, we study a VRP problem with known cost matrix satisfying

$$C_{ik} + C_{kj} \geq C_{ij}, \quad \forall v_i, v_j, v_k \in V \tag{2.1}$$

The cost may be related to distance, time or emission. Equation (2.1) is called the triangle inequality and required by some algorithms for VRP, such as Clarke and Wright's saving algorithm (1964). Note that the above condition is not restrictive for our problem because the indirect path via k would be chosen in case the direct path has a higher cost. There are m vehicles parking at depot v_0, and they could be identical (homogenous VRP) or different (heterogeneous VRP). The VRP consists of finding a set of m vehicle routes that has a minimum total travelling cost. At the same time, the following conditions are met: (1) Each vehicle starts and ends at the depot; (2) Each customer $v_1, \ldots, v_n$ can only be visited once by one vehicle; (3) For each customer v_i, the service starts within the time window interval $[a_i, b_i]$.

There are some variants of VRPTW in the area of green VRP literature, and these variants are developed according to real-life applications and new operational considerations. In this thesis, emphasises will be given to time-dependent VRPTW and multiple time-windows VRPTW.

2.1.2 Time-dependent VRPTW in green VRP

Most existing studies on Green VRP do not use a time-dependent setting, although they recognise the importance of speed in determining vehicles' emissions. They assumed that vehicles can travel at an emissions minimising speed, but in real-life, vehicles must move at the flow of traffic, especially in urban areas, where speeds are variants and time-dependent. It implies that urban road has a speed limit. During regular hours, traffic flow is at the rate of the speed limit, and during rush hours, the travel speed is much slower. The difference in emissions levels or fuel consumptions between using a constant speed and using a variable speed can be up to 40% (Van Woensel et al., 2001; Akcelik, 1982). With technologies development, time-varying transit times of road segments can be recorded and analysed, for example, by using remote vehicle tracking technique. In literature, some studies built up a speed or transit time lookup table/profile for each period in a day, and this is assumed the same for all road links (Figliozzi, 2010). Some studies had access to real-time road network data and had different profiles for different road links (Qian and Eglese, 2016; Ehmke et al., 2016).

The time-dependent travel speed is modelled by discretising the day into some time buckets (one-hour duration or simply morning, midday and afternoon), and then assign a unique mean speed value to each bucket. Figliozzi (2010) discretised the planning horizon into five time-intervals and associated each time interval with a mean speed value, which was artificially generated to reflect different traffic conditions. Qian and Eglese (2016) and Ehmke et al. (2016) divided a day into 24 and 15 time slots [1] respectively, and the travel speeds for each time slot for each road segment were estimated using real-life travel data, assuming a stable traffic condition in each time slot. Ichoua et al. (2003) divided a planning horizon into three time zones, morning, midday and afternoon, and arbitrarily applied a correction factor to free flow speed for each time zone to represent different traffic congestion conditions. Van Woensel et al. (2008) used a queueing model to create a function for time-dependent travel times for congestion. The method was based on the relationship between traffic flow, speeds and density under a stable traffic condition, which resulted in a speed-flow diagram following Greenshield (1935). It provided a new method for situations where time-to-time speeds data is unavailable, but traffic flows data can be easily obtained. They used real-life traffic data in Belgium. Their queueing model speeds were compared to three time-zones speeds, time-independent speeds, and the results showed that queueing model speeds best matched real-life travel speeds. Van Woensel et al. (2008) applied a tabu search heuristic to solve the TDVRP, and a 2-opt neighbourhood constructor was used. They found out that comparing to time-dependent speeds, using constant speed would produce unrealistic route solutions in a congested traffic environment.

[1]Time slot is used to describe time interval associated with time-dependent travel speeds, while time window is used to describe customers booking option.

Eglese et al. (2006) recognised that the travel time or travel speed of a journey in a road network depend on the time of the day, the day of the week and the season of the year. As there are trackable patterns of traffic congestion throughout the year, those patterns can be measured and recorded for route planning. In their research paper, they developed a time-dependent travel time database called Road Timetable^{TM} using real-world data of road network in the north west of England. Such a database will greatly benefit the study of real-life time-dependent vehicle routing and scheduling problems, as speed plays an important role in calculating emissions accurately. This inspired us to construct a similar travel speed profile and an emissions rate profile of each hour for weekdays based on the data we collected.

Figliozzi (2010) first introduced the emissions vehicle routing problem (EVRP) with time-dependent travel speeds, which typically represents real-life varying traffic conditions, hard time windows, as well as capacity constraints. A mixed integer programming model with two objectives: minimising the number of vehicles and minimising emissions cost was formulated. The amount of emissions was modelled using the MEET project method (Hickman et al., 1999). The MEET method calculated emissions based on road speeds. Time-dependency of speeds was modelled through discretising the planning horizon into five-time intervals and each time interval was associated with a mean speed value, which was artificially generated to reflect different traffic conditions, categorised as uncongested, somewhat congested and congested traffic conditions. Each congested traffic condition had two congested periods mimicking morning and evening peak hours. In the emissions cost objective function, emissions cost was calculated as an estimated market price of CO_2 emissions multiplied by the amount of CO_2 emissions. This market price may not be accurate as it is hard to predict the economic cost of emissions unless an emissions tax is officially launched. Another possible way to price emissions was shown in Franceschetti et al. (2013). They used fuel price to represent emissions price. Figliozzi (2010) applied an iterated route construction and improvement heuristic algorithm first to solve vehicles number minimisation problem and then using this fixed number of vehicles, solve emissions minimisation vehicle routing problem to minimise emissions cost. Numerical experiments were conducted on Solomon benchmark problems. Results showed the potential to significantly reduce emissions with a minimal or even null increase in other routing costs.

Jabali et al. (2012) presented a time-dependent green vehicle routing model and solved it using a tabu search procedure. The emissions per kilometre were measured using a function of speed, as provided in the MEET report (Hickman et al., 1999). These authors set up a speed and travel time profile for a planning horizon, i.e. 6:00am to 12:00am. In their model, there were two main congestion periods; morning and afternoon, and they discovered that reduction in emissions could lead to an increase in the travel time.

Maden et al. (2010) also constructed a time-dependent VRSP using real-life data for a vehicle fleet delivering electrical wholesale items in the South West of the UK. They created a road timetable that stores the shortest travel times between customers at different start times, and compared the results of using road timetable data with constant speed, and found out a 7% reduction in CO_2 emissions.

Bektaş and Laporte (2011) measured speed as a continuous variable instead of level-based values. They constructed a pollution routing problem with energy-based emissions model proposed by Barth et al. (2005). The thermal energy generated by vehicles is based on travel speeds, travel distance, vehicle types, road characteristics, vehicle gross weight (vehicle net weight plus payload) and engine efficiency. They investigated the relationship between travel speeds and fuel consumptions. In their optimisation model formulation, the travel speed depended on a road type is an average of the minimum and the maximum travel speeds allowed on that road segment. They developed an integer linear programming model with a generalised cost (emissions, fuel costs and driver's wages) minimisation objective. The constraints considered were vehicle capacity and time windows. Optimisation software CPLEX was applied to solve this problem. They found out that there are conflicts between minimisation objectives of distance, fuel costs, drivers' wages and emissions.

Many researchers found out that there are conflicts between emissions minimisation objective and traditional objectives, such as travel time minimisation and travel distance minimisation. Palmer (2007) showed through experiments a 4.8% reduction in emissions using emissions minimisation as objective compared to using travel time minimisation. However, there was a 3.8% increase in travel time. Jabali et al. (2012) showed a similar result with 11.4% reduction in emissions and 17.1% increase in travel time. Ehmke et al. (2016) showed that the reduction in emissions is proportionally larger than the increase in travel durations. Comparing to traditional objectives of VRP, minimising emissions is achievable at a relatively low expenses of other costs. Similar results can be seen in Qian and Eglese (2014). There is still a trade-off between objectives, and a solution approach is to apply multi-objective methods. Multi-objective decision making with green aspects can be seen in the research of Jemai et al. (2012) and Demir et al. (2014).

Demir et al. (2012) proposed a pollution routing problem with Barth and Boriboonsomsin (2008)'s emissions model. They solved the problem in two phases, the first phase solved a VRPTW using an adaptive large neighbourhood heuristic and the second phase solved a speed optimisation problem to determine the optimal travel speed on each road segment. The travel speed was assumed to be in the range of 20 km/h to 90 km/h. However, in real-life travel speed is not usually a decision variable, and a driver has to drive up to the speed of the travel flow for safety reasons.

Ehmke et al. (2016) studied a capacitated VRP to minimise time-dependent emissions in urban areas in Stuttgart, Germany. To capture the time-dependency of travel speed on each road, a day was divided into 24 one-hour time buckets, e.g. 6 am to 7 am. Instances were derived from real road network in Stuttgart, Germany with 230 million speed observations. This dataset was collected during the years between 2003 and 2005, and details of this dataset can be found in Lorkowski et al. (2004). When the traversing along a road segment happens at 6:30 am, the average speed observations during 6:00am to 7:00am on this road segment would be used. The emissions model used in Ehmke et al. (2016) was the Comprehensive Emissions

Model (CEM) (Barth et al., 2005). They chose this model as they wanted to study the impact of vehicle load factors on vehicle emissions levels. It is one of the advantages of CEM model over other emissions models in literature, which it counts for the impact of vehicle load on the amount of emissions. The solution method applied was LANTIME tabu search algorithm adopted from Maden et al. (2010), which was designed for solving time-dependent VRP. This algorithm requires an initial solution, which was constructed using Solomon's insertion heuristic (Solomon, 1987). Furthermore, neighbourhood constructors used in tabu search were Adapted Cross Exchange, Unsertion/Removal, One Exchange, and Swap. The cost/objective function considered was the total amount of emissions. However, when searching for the optimal solution, a solution in the neighbourhood with fewer vehicles used was always considered a superior or a better solution regardless of its emissions level. An interesting finding was there are conflicts between emissions minimisation objective and travel time minimisation objective.

Qian and Eglese (2016) studied a CO_2 emissions minimisation vehicle routing problem with time-varying travel speed. Test cases were generated on a London road map with real traffic data collected by traffic information company, ITIS Holdings. They had access to traffic speeds data for each road segment. Also, there were multiple possible road links that connect two customers' sites like in real-life, so path selection was part of the decision problem. The majority of literature in this section assumed travel speed for a certain period is constant, and it is the speed of traffic flow at that time. However, vehicle speed could be slower than current traffic flow speed. For example, if the current traffic flow speed is 100 km/h, a driver could drive at 78 km/h assuming this is the most fuel efficiency speed. Qian and Eglese (2016) treated the travel speed as a decision variable, ranging up to the current maximum traffic flow speed on the link travelled. Furthermore, a limit on waiting time at customer site was applied. They implemented a column generation based tabu search algorithm and showed that with path selection and speed optimisation, 2-3% emissions could be saved compared with always using fastest path and traffic flow speeds. However, if within a speed range emissions level is not very sensitive to speed change, setting travel speeds to traffic flow speeds will give a good solution as well. It is interesting that allowing more waiting time at the customer site will save 2-3% emissions. The computational time was more than 20 hours for 25 customer cases, and they simplified the algorithm by solving a distance minimisation VRP first, and then applied speed optimisation and path selection only at the final step. Results showed to be similar to the original solution with only five minutes solution time. It maybe still not applicable for online problems which required much shorter reaction time. This work is an extension of Qian and Eglese (2014)'s work with the same problem setting, but different solution approaches. In Qian and Eglese (2014), they proposed a time-increment-based dynamic programming method. Dynamic Programming method is widely used in time-varying travel speed vehicle routing problems. In time-varying speed road-network, departure time is a vital factor that determines the emissions of travel. It will influence the solution quality as well (Van Woensel et al., 2008).

There are other types of GVRP which do not reduce emissions through reducing fuel consumption. They studied vehicles using emissions-free energy such as electric vehicles. These types of vehicles need to recharge often. It formulates a different kind of problem considering locations of charge station. This kind of problem is not discussed in this research. To reduce road transportation emissions, researchers focus on developing a fuel-efficient engine, electric vehicles, hybrid vehicles and alternative fuels. Some of the technologies are in their early stage and won't be ready to be put into implementation shortly. Some techniques start to enter the market, for example, the electric vehicle brand - Tesla. Electronic vehicles generate zero emissions, but the price of electric cars is much higher. The majority of road users still use fossil fuel vehicles, and it takes time for these vehicles to depreciate fully. Furthermore, electric vehicle technology is premature. The earliest predicted time for electric vehicles to fully cover the market is probably 2025 (Kordic, 2016). Hence the short term solution is to reduce emissions from road transportation through creating emissions reduction routes.

2.1.3 Multiple time windows vehicle routing problem with time window constraints

In Chapter 7, we will propose an incentive sharing pricing policy for green delivery service, where customers have an initial preferred time window and several alternatives. A multiple time windows VRP problem is formulated. There is little literature on multiple time windows VRP, and this topic has not received much attention. However, real-life applications do exist, such as in the delivery of furniture or electronic products where customers occasionally provide a choice of time periods, or in long-haul transportation (Belhaiza et al., 2014). It was first introduced in Favaretto et al. (2007), and they studied a problem where each customer has one or many time windows and would be visited periodically. The objective was to minimise the total weighted time of both travelling and waiting time. A mixed integer linear programming model was formulated which is similar to traditional VRPTW formulation with additional variables related to the multiple time windows and specific constraints tackling periodic visits to customers. A set of test cases was generated based on Fisher (1994). The problem was solved by an ant colony type of metaheuristics. Belhaiza et al. (2014) also investigated VRPMTW problems similar to that in Favaretto et al. (2007) with the same objective without periodic visits, i.e., each customer is visited by one vehicle once. They extended the mathematical programming model in Favaretto et al. (2007), and solved this problem with a new hybrid variable neighbourhood tabu search heuristic. Furthermore, they proposed an interesting minimum backward time slack algorithm to adjust the departure times at each customer site under multiple time windows environment. This algorithm proved to be able to find the optimal solution to the minimum travel duration problem with multiple time windows. In VRP with time windows either single or multiple time windows, departure time will largely affect the objectives even if the sequence of the routes is the same. Hence, this raises the departure times optimisation problem.

2.1.4 Departure times optimisation

For traditional VRPTW, the departure time at a customer site is set to be the end service time at the previous customer site, but for green VRPTW, there is a problem with the optimal departure times, due to the existence of congestion period. For example, when finishing at a customer site, it is in the middle of the congested period, and the vehicle could wait at the customer site for some time and depart later. Kok et al. (2011) observed that most solution algorithms for time-dependent vehicle routing problem schedule the departure time of each journey to the earliest available time, but temporary traffic congestion makes these route plans non-optimal in the sense of minimising total travel time. They proposed a departure time optimisation problem with integer linear programming formulation and solved the problem using constructive insertion algorithm. They considered travel speed to be a piecewise linear function of time of the day and a resulting travel time function of the departure times of the day. The test case results showed a 15% reduction in total travel time.

Kim et al. (2005) also studied the optimal departure times with real-time traffic information on an urban road network in Southeast Michigan. They considered a stochastic vehicle routing problem on a road network with non-stationary travel times and formulated it as a Markov decision process. Travel speeds for congested and non-congested situations were modelled as random variables following two different normal distributions, respectively. Transition dummy variable was modelled as a random variable following a bivariate normal distribution. With real-time traffic information, cost savings and productivity improvement were experienced by optimising departure times.

Hashimoto et al. (2006) proposed a standard vehicle routing problem with soft time windows and travelling time constraints by putting constraints into the cost function. They solved the problem first by determining the optimal sequence of routes by an iterated local search algorithm with Cross, Exchange, 2-opt* and Or-opt neighbourhoods' operators and then optimised the departure times to customers by a pseudo-polynomial time algorithm of dynamic programming for convex travel time cost functions. Tested on Solomon benchmark problems showed the effectiveness of the proposed integrated algorithm.

2.2 Solution algorithms

The computational complexity of solving a VRP to optimality was proved to be NP-hard (Lenstra and Rinnooy-Kan, 1981). NP-hard problems are unlikely to be solvable in polynomial time. Exact solution methods are computational costly for industrial-sized problems (Braekers et al., 2016). For an online type of problem, where a response is needed to be given to the customer in a short time, it is more practical to use a heuristic or metaheuristic method, which generates an acceptable solution in a short time. However, the solution is not guaranteed to be optimal. In a survey paper of vehicle routing problem, metaheuristics (71.25% of articles reviewed) are used more often than exact methods and classical heuristics, furthermore simulation and real-time solution methods are rarely applied. The state of the art metaheuristics used in green vehicle routing problems are reviewed by Lin et al. (2014), including tabu search, simulated annealing, variable neighbourhood search, genetic algorithm and ant colony optimisation. Metaheuristics for time-dependent vehicle routing problem are also reviewed by Bräysy and Gendreau (2005).

2.2.1 Tabu search

Tabu search (TS) developed by Glover (1989) is based on a local search algorithm. However, local search algorithms tend to get trapped into suboptimal regions or on plateaus where many solutions fit equally. The basic idea is that a new current best solution is selected in the neighbourhood of the current solution at each iteration, even if it leads to a deterioration in the objective function. This deterioration happens only to avoid paths which have already been investigated recently. Then this method can escape from suboptimal regions. Besides this characteristic, this algorithm has a short-term memory, known as the tabu list. The tabu list stores either the reversals of recent moves or the recent complete solutions with a specific length (e.g. the recent 9 solutions, length = 9). Each time a new element is added to the end of the list it deposes an item from the top of the list. Those elements in tabu lists are tabu or forbidden to be visited again shortly to avoid short-term cycling. This forbidden status could be broken if certain conditions are met, and this is called the aspiration criterion. Typically, the feasibility of solutions is maintained during the local searches. The termination criterion is that the objective function has not been improved for the last N iterations, or some fixed number of iterations has been reached. The initial solution is typically created by insertion heuristics such as Solomon's I1 insertion heuristic (Solomon, 1987). Starting from the initial solution, improvements are made to it by applying local searches with one or more neighbourhood operators, such as 2-opt, 2-opt*, Or-opt, Swap, Exchange, Relocate and so on. The tabu search algorithm is demonstrated in a detailed graphic form in Glover (1990). The first attempt to apply tabu search to the VRP was by Willard (1989). There are improvements and changes to the classic tabu search algorithms, such as the tabu route heuristic (Gendreau et al., 1994), the Tailard tabu search algorithm (Taillard et al., 1997) and the granular tabu search algorithm (Toth and Vigo, 2003). Post-optimisation techniques are developed to incorporate with tabu search algorithm, such as GENIUS heuristics (Gendreau et al., 1992).

2.2.2 Simulated annealing

Simulated annealing (SA) is based on a randomised local search algorithm. Like TS, this method accepts moves which lead to deteriorations in the objective function with certain probabilities to avoid getting stuck in local minimum solutions. SA originates from Kirkpatrick (1984) who studied the optimisation problem in annealing in metallurgy. The physical annealing process aims at generating solids with low-energy states by gradually reducing temperature levels. The process is heating a crystalline solid and then letting it cool very slowly until it achieves its most regular possible crystal lattice configuration with the lowest energy state. In this state, it becomes free of crystal defects. Simulated annealing establishes the connection between this type of thermodynamic behaviour and the search for global minima for a discrete optimisation problem. The basic algorithm for a minimisation problem is at each step considering a random neighbouring solution of the current solution, if the energy (objective value) is less then update the current state, otherwise, accept the current solution with a probability. The probability of acceptance depends on the system temperature parameter which is decreasing as time goes on. The probability will decrease as temperature decrease in a predefined fashion. As for the later stage of searching, a worse solution is less likely to be accepted. Accepting worse solution is designed to be able to escape from local optima. The stopping criterion is either system temperature becoming very close to zero or a certain number of iterations being reached. According to Gendreau et al. (1994), it has been shown that SA asymptotically converges to the global optimum. Also, the effectiveness of this method is affected by the choice of parameters and initial temperature. It is better to involve a restart procedure after every few steps. The main limitation is that SA needs a few predefined parameters and the optimal set of parameters are different for different problems. Parameter tuning is needed for efficient application of SA.

2.2.3 Variable neighbourhood search

Variable neighbourhood search was first developed in Mladenović and Hansen (1997) and finds its success in combinatorial optimisation applications. It is a metaheuristic algorithm based on systematic changes of neighbourhoods in the search process. The ability to escape from local optimum is enhanced by starting the search in each neighbourhood from a random neighbour of the incumbent solution. The basic idea of it is to find a local minimum in a descent method in one neighbourhood, and then try to escape this local minimum by exploring a distant neighbourhood of this solution in a sequential way among the candidate neighbourhoods or in a random fashion. A jump to a new neighbourhood only happens if the local minimum found in this neighbourhood is better. The pillars of VNS is based on the observation that a local minimum found in one neighbourhood structure is not necessary a local minimum for other neighbourhood structures. A global minimum is a local minimum in the combination of all possible neighbourhood structures. Local minima found by different neighbourhoods are relatively close to each other for many problems. Unlike tabu search or simulated annealing, VNS is not a trajectory following method which accepts a worse solution. The advantages of this method are its ease to implement, and it is parameter-free, which means it could be applied to different problems without extra effort on parameter tuning. Since it was introduced, this method has developed rapidly and found success in solving combinatorial optimisation problems. There are several variants of the basic VNS, such as variable neighbourhood descent and reduced variable neighbourhood search (Hansen et al., 2010). Also, due to its parallel nature, there are several different parallel versions of VNS too.

2.2.4 Others

TS, SA and VNS are local search based metaheuristics, where each iteration is based on a single candidate solution and these metaheuristics are developed to improve local search heuristics. Genetic algorithm and ant colony algorithm are population based metaheuristics, while the search for the solution is using a population of candidate solutions.

Genetic algorithm (GA) is a population search metaheuristic that mimics biological processes of species evolution and follows the Darwinian Theory that the fittest survives. This algorithm begins with an initial population of parent solutions, and offspring solutions are produced which exhibit some of the characteristics of each parent with values of fitness. Then offspring solutions with higher fitness values are more likely to be chosen to become parent solutions in the next generation. In computer science, solutions are represented as chromosomes in the form of binary strings, but this is not necessary. Genes are those individual positions that form a chromosome. New generations of solutions are produced by genetic operators, such as crossover and/or mutation. Fitness values of those in the new generations are calculated by the values of the objective function in the VRP case. The selection rule usually prefers the lowest objective value such as the lowest travel distance.

In details, the initial step of GA is to generate the initial population either randomly (Prins, 2004) or by reasonably structured solutions, for example, by the Sweep method or the Fisher-Jaikumar method (Baker and Ayechew, 2003). Then those solutions are represented as chromosomes with trip delimiters (Baker and

Ayechew, 2003) or without trip delimiters (Prins, 2004). The benefits of eliminating trip delimiters are demonstrated by Prins (2004). Then a crossover procedure takes place, such as 2-point crossover or OX, LOX for instances without trip delimiters. Mutate offspring when necessary or when to accelerate the convergence of GA. The next step is to compute fitness and choose next parent solutions based on that. The stopping criteria can be based on the number of generations or the computing time and so on.

Genetic Algorithms are also well developed in solving vehicle routing problems, but they rely on the quality of the initial solution, and to maintain the feasibility of the solutions regarding time window constraints, sophisticated designed mutation and crossover operators are needed, as once the feasibility of solutions is lost, it is difficult to regain feasible solutions.

Ant colony optimisation (ACO) is another population search metaheuristic inspired by the behaviour of ants seeking paths between their colony and food locations. It is based on probabilistic techniques. In the real world, an ant may walk randomly from its territory to locations of food and lay down pheromones to communicate with other ants and mark the path. The other ants are not likely to keep travelling at random, but follow the marked path and lay more pheromones to reinforce this path. However, pheromones will evaporate at a particular rate. The longer the path takes to travel, the fewer pheromones there will be left for that path. So the shortest path is the most attractive path to the other ants to follow. Based on this mechanism, Dorigo (1992) first applied this method to solve TSP problems, and researchers then extended its use to solve VRP problems. In VRP, artificial ants are simulated to construct feasible routes. At each stage, an ant may travel from node i to j with a probability function that concerns both the amount of pheromone laid on arc ij and the distance between ij. This route construction can be done by a single ant one after another (sequential) or by multiple ants together (parallel) (Mazzeo and Loiseau, 2004). Then, the pheromone trials are updated to improve those feasible routes. There are two types of updating, local and global (Bell and

McMullen, 2004). Local updating represents the evaporation of pheromones to ensure that no one path becomes too dominant and avoid trapped in a local optimal solution. Global updating represents the accumulation of pheromones to reinforce favourable paths, either to all solutions or to solutions made by elite ants or to the best solution found. After trial updating, route improvement strategies are made such as 2-opt heuristic or greedy ranked candidate list. Stopping criteria is similar to GA. ACO can deal with large size problems and according to Bullnheimer et al. (1999), the performance of ACO can compete with other metaheuristics such as TS and SA, and outperform neural network method. The limitation is the need to choose appropriate parameters which impact performance a lot. Hybridisation of metaheuristics is also well applied, which are a combination of more than one metaheuristics and acclaims to perform better than any of the metaheuristics alone (Talbi, 2002).

2.3 Models for determining vehicle emissions

As emissions are affected by travel speeds, vehicle types and many other factors, the shortest or quickest route may not be optimal with respect to emissions or fuel consumptions. Also, the relationship between travel speeds and CO_2 emissions is nonlinear with a single minimum. During a congestion period, a vehicle experiences slow speeds, and it frequently needs to accelerate to catch the speed of traffic flow or decelerate when traffic slows down. This driving pattern generates more emissions than steady-state free-flow speeds. When a vehicle drives at a steady-state free-flow speed, its engine operates smoothly, and there is no acceleration behaviour where the vehicle generates the most emissions of all. Vehicle engine will experience high load when the vehicle travels at high speed, which is another reason for high emissions levels.

In this case, green vehicle routing depends on the correct computation of the carbon emissions to generate a route plan that is genuinely greener than the distance or travel time minimisation route plan (Turkensteen, 2017). Various emissions models differ in nature of estimating emissions and fuel consumptions. The nature of models describes whether the model is a macroscale model that aggregates total emissions and works as a rough estimate or it is a microscale model that is capable of predicting a relatively more accurate second-by-second emissions. Some models are developed based on physical studies of engines and fuel technologies, and some models are constructed using on-road measurements and real-life experiments data. Factors that influence emissions or fuel consumptions in general fall into four categories: vehicles (e.g. weight, engine type, engine size, and frontal area shape), drivers (e.g. driving behaviour), environmental conditions (e.g. road slope and road surface resistance) and traffic conditions (congestions). Among all those factors, driver related factors are hard to measure and often excluded from all emissions models. The choice of emissions models which is convenient to use or fits the description of the problem faced is essential before operational planning. The decision also depends on the data availability, as accuracy usually relies on the amount of information provided. In literature, an agreement has been made that emissions are proportional to fuel consumption, so the models measuring fuel consumptions are equivalent to models measuring emissions directly. A detailed review of existing emissions models can be found in Palmer (2007) and in Demir et al. (2011). Some models consider a large number of factors, which make them hard to implement, and can only be applied for short trips. Models frequently in use in the literature of green vehicle routing problem are the MEET project, the NAEI project and Comprehensive Emissions Model (CEM). Regardless of the emissions models applied, CO_2 emissions could be 15-20% higher in real-life traffic conditions Palmer (2007). One possible reason is that most emissions models predict emissions based on hot stabilised engine conditions and do not consider cold engine starts. If cold engine starts are considered, the computed emissions will be 10% higher.

Emissions come from the burning of fossil fuels. Unlike other pollutants emitted by vehicles, CO_2 emissions are proportional to the amount of fuel consumed (Ubeda et al., 2011). To some content, minimising CO_2 emissions is equivalent to minimising fuel consumption. According to U.S. Energy Information Administration (2017), standard diesel fuel emits 2.68 kg CO_2/L and petrol fuel without ethanol emits 2.35 kg CO_2/L. This fuel conversion factor can also be estimated based on analysing the chemical reaction function of the fuel burning process. Lichty (1967) stated that the fuel combustion chemical reaction is $C_{13}H_{28} + 20 \cdot O_2 \rightarrow 13 \cdot CO_2 + 14 \cdot H_2O$. Standard diesel ($C_{13}H_{28}$)'s molecular mass is 184 and the molecular mass of CO_2 is 44. Based on the reaction function, one unit of $C_{13}H_{28}$ can generate 3.11 unit of CO_2. Diesel density is 0.84 kg/L, so fully burning of one litre diesel fuel will generate $3.11 \times 0.84 = 2.61$ kg CO_2. These conversion factors may be slightly different across nations. Department for Environment, Food and Rural Affairs of the UK government (2010) guided companies to report emissions. In their report, the fuel-emissions conversion factors are provided for different types of fuel and pollutants. For diesel, it is 2.641 CO_2/L and 2.301 CO_2/L for petrol.

2.3.1 Distance or fuel based emissions model

Distance or fuel based emissions models are macroscale level emissions models, which approximate the overall emissions for the whole trip. Fuel consumptions or emissions are calculated based on total distance travelled, fuel types and fuel conversion rate, assuming the vehicle travels at a free flow speed. Those models can be adjusted to capture the difference in vehicle payload, but they ignore factors like vehicle speeds, vehicle acceleration, and road characteristics.

2.3.2 The MEET and the NAEI Project

The MEET project aimed to provide a basic, European-wide procedure for evaluating the impacts of transportion on the air pollution. The duration of this project was from 1996 to 1998, and the final results were shown in Hickman et al. (1999). It was funded by the European Union. They developed several models for different types of vehicles with various weight ranges. Those models are microscale level models. Vehicle emissions levels are calculated based on travel speeds, vehicle mass and vehicle types, and can be adjusted to road gradient as well as vehicle load. MEET suggests the following emissions models:

$$E = \left(K + av + bv^2 + cv^3 + \frac{d}{v} + \frac{e}{v^2} + \frac{f}{v^3}\right) GC \cdot LC \cdot distance \tag{2.2}$$

where K, a, b, c, d, e, f are vehicle specified parameters, v is travel speed in km/h, E is emissions in g, and $distance$ is in km, GC is the road gradient correction factor and LC is the load correction factor. These parameters were estimated based on real-life experimental data. The MEET model is an average of various speed-emissions curves of a range of driving cycles. A driving cycle means a particular pattern of driving behaviour containing a different combination of stops, starts, acceleration and deceleration. So the effect of acceleration is implicitly included. The main drawback of this model is its vehicle-specified parameters settings only depend on vehicle weight range and usage, not considering vehicle make or engine types. Figliozzi (2010), Jabali et al. (2012) and Saberi and Verbas (2012) applied this model for carbon dioxide emissions reduction vehicle routing problems.

Similar models can be found in Road Vehicle Emissions Factors, which is developed in the UK by TRL (transportation research lab) and named NAEI (Boulter et al., 2009). The NAEI database is based on a large number of measurements from various programmes conducted over years. The database compiled as part of the MEET project also included data from TRL. The most recent update of it is on 2002. The NAEI project suggests the following hot engine emissions models:

$$E = (a/v + b + cv + dv^2 + ev^3 + fv^4 + gv^5) \cdot distance \tag{2.3}$$

where $a - g$ are vehicle categories and pollutants types specified coefficients. The formulation could also be adjusted for road gradient and vehicle load. Detailed formulas for carbon dioxide emissions can be seen in Appendix E of Boulter et al. (2009).

2.3.3 Comprehensive Emissions Model (CEM)

Comprehensive Emissions Model starts to develop by Barth et al. (2000) in the year 1996 in the University of California, sponsored by the National Cooperative Highway Research Program and the US Environmental Protection Agency. The majority of work was finished in 2000 and maintained till 2008. The aim was to develop a microscale level model to predict second-by-second vehicle emissions. Vehicle emissions are calculated considering factors such as vehicle mass, engine specified parameters, road specified parameters, vehicle speeds, acceleration and so on. The model is shown in equation (2.4).

$$FR = \phi\left(kNV + P/\eta\right)/44 \tag{2.4}$$

where ϕ is the fuel-to-air mass ratio, k is the engine friction factor (0.2), N is the engine speed, V is the engine displacement in litres (in range 2 to 8), P is the second-by-second engine power output in kilowatt, and η is the efficiency parameter for engines (0.4). Engine speed can be approximated by vehicle speeds:

$$N = S\left(\frac{R(L)}{R(L_g)}\right)v \tag{2.5}$$

where S is a vehicle specified parameter which is the ratio of engine-speed divided by the vehicle speed when in top gear L_g, $R(L)$ is the gear ratio in gear $L = 1, 2 \ldots L_g$, and v is vehicle speed in m/s.

The power demand function is in terms of vehicle mass, vehicle speeds and other road specified parameters, as shown in equation (2.6).

$$P = \Big(Ma + Mg\sin\theta + 0.5C_d\rho Av^2 + MgC_r\cos\theta\Big)\,v\Big/\,1000\eta_{tf} + P_{acc} \tag{2.6}$$

where M is vehicle mass in kg, a is vehicle acceleration rate, g is the gravitational constant ($9.81\,\mathrm{m/s^2}$), θ is road slope in radians, C_d is the coefficient of aerodynamic drag (0.7), ρ is the air density ($1.2041\,\mathrm{kg/m^3}$), A is the area of vehicle frontal surface area (between $2.1\,\mathrm{m^2}$ and $5.6\,\mathrm{m^2}$), C_r is the coefficient of rolling resistance (0.01), η_{tf} is the vehicle drive train efficiency (0.4) and P_{acc} is the power used for air conditioning and other vehicle accessories (assume 0). Observing the above equations, most parameters are predetermined and vehicles related. The variables in the equation (2.6) that may change in time are vehicle mass, vehicle speeds and acceleration rates. In practical use of CEM, researchers often make some realistic assumptions and use a simplified version of this model. For instance, Ehmke et al. (2016) used the model shown in equation (2.7).

$$E = \frac{1}{32428}\left(33\frac{d}{v} + 0.00462dv^2 + 0.000275(u+l)d\right), \tag{2.7}$$

where E is the total amount of emissions in kg, d is travel distance, v is travel speed, u is the mass of the vehicle (for a medium vehicle, $u = 6350\,\text{kg}$; for a heavy vehicle, $u = 12\,700\,\text{kg}$) and l is vehicle load in kg. Bektaş and Laporte (2011), Demir et al. (2012) and Franceschetti et al. (2013) applied this model for carbon dioxide emissions reduction vehicle routing problems.

2.3.4 COPERT software

Ntziachristos et al. (2000) developed COPERT. It can measure not only CO_2 emissions but also other major GHG emissions. This model is also based on real-life experiments and takes into account road gradient and speed changing as well. It generates an emissions rate that depends on vehicle types, engine emissions standard and the average travel speed. Scott et al. (2010) used this computer program along with the payload correction and heavy-duty vehicle correction factors. There are other emissions models, which may be more complex and accurate. As they are used very rarely and do not appear in most of literature on green vehicles, we do not discuss them in detail. Those models are four-mode elemental fuel consumption model, the running speed fuel consumption model, the air quality model, and the traffic situation models. One can refer to Boulter et al. (2007) for a detailed summary of other emissions models. A detailed comparison based on discrepancies between results yield and on-road consumptions results of vehicles for several vehicle emissions models can be found in Demir et al. (2011). Readers can also refer to Frey et al. (2010) for an overview of emissions measurement methods.

2.3.5 Comparing emissions models

In this section, the MEET project and the CEM models are compared in terms of their nature, factors considered, and their performance comparing to actual road data. As the NAEI model is an extended version of the MEET with UK data, MEET project will be used as a representative. The MEET project models were constructed based on on-road measurements and real-life experiments data, while the CEM were developed using physical studies. The CEM have been tested in real road experiments under 23 vehicle categories by Barth et al. (2005). The comparison of factors considered in these models is shown in Table 2.1.

Factors	CEM	MEET
Vehicle related		
Total vehicle mass	X	X
Engine size	X	
Engine temperature		X
Oil viscosity	X	
Gasoline type	X	X
Vehicle shape	X	
Environment related		
Road gradient	X	X
Wind conditions	X	
Surface conditions	X	
Traffic related		
Speed	X	X
Acceleration	X	

Table 2.1: Comparison of models regarding factors included (Source: Adapted from Demir et al. (2011)).

As shown in the above table, the CEM is a more complicated model which considers more factors that have an impact on fuel consumptions or emissions. CEM model is based on second-by-second tailpipe emissions data collected from a variety of laboratory driving tests. It is an instantaneous emission model different from regression-based models such as the MEET. Instantaneous emission models could be more accurate in predicting traffic emissions because they consider acceleration and vehicle load for example. In practice, however, to implement the CEM requires detailed vehicles or road specified parameters, which are hard to obtain in real-life situations. The problem of interests considers a maintenance service company where each engineer carries a toolkit with a constant weight, so weight (load) may not be an important factor as for groceries delivery companies. In this case, the MEET and the NAEI models are more suitable.

Erlandsson et al. (2008) conducted on-road measurements of vehicles with different weights on a highway segment of 100 km. These kinds of on-road fuel measurements are typically implemented using engine and chassis dynamometer tests, tunnel studies, remote censoring and onboard instrumentation readings (Demir et al., 2011). The results from the CEM and the MEET models were compared to the on-road measurements. The results show that on average the CEM tends to overestimate fuel consumptions and emissions, while the MEET model tends to underestimate the amount. The average absolute difference percentage of the CEM model is 30.33% and for the MEET model 30%. In summary, both models are capable of giving a relatively accurate prediction of emissions levels on a second-by-second basis, but the MEET model requires less vehicle or road specified parameters estimation, which makes it easier to implement. Furthermore, in on-road measurement tests, the MEET project yield slightly better results than the CEM models. In this report, the MEET project models together with its UK version the NAEI models will be used to calculate emissions levels.

2.4 Value based demand fulfilment/ Demand management in VRP

Revenue management has an important impact on profitability and requires sufficient knowledge of customer preferences and flexibility. Tailored services, such as Amazon recommendation advertisement which are based on customers' searching information, could win increase of customers' loyalty and satisfaction. Revenue management arises in the airline industry for setting flight fares. The main differences between the airline industry and online grocery or service delivery are that the cost in the airline business is sunk cost at order in-take while the cost of delivery is a variable that interdependent among orders, as orders that located close to each other will have less transportation cost. Another difference is that for airline business customer heterogeneity lies in willingness-to-pay and demand flexibility only. On the other hand, delivery business has more dimensions like customer locations and time window requirements. Airline fares are determined based on flights' popularity. Agatz et al. (2013) gave an example of Albert.nl's dynamic pricing policy of delivery slots based on their popularity and showed it could smooth the demand by reducing the ratio of demand gap from 3 : 1 to 1.5 : 1. Demand gap was defined as the number of customers in the busiest time window divided by the one in the least busy time window (Agatz et al., 2013).

Revenue management in service delivery business models considers time slots management problems or pricing problems (Agatz et al., 2013). Time slot management is classified into two categories in Agatz et al. (2013)'s review paper, i.e., differentiated slotting and dynamic slotting. Differentiated slotting studies the problem of deciding the number of slots offered in each area and the length of the delivery time windows before order intake. On the other hand, dynamic slotting makes decisions during the actual sales process, for example, closing a slot when it reaches capacity. A more advanced case is to reserve scarce time windows for the most beneficial customers by considering opportunity costs, instead of first-in-first serve orders in-take. As for the home attended service providers, the costs are two fold, the expenses for technicians' wages or service fees and delivery fees, e.g. fuel cost. Delivery cost is directly related to the location of each potential customer considering existing customers, so this generates the problem of assigning the customer to a cost-effective time window.

With slotting decisions, customers are pushed away from certain options by restricted time slots length or closing a slot early as mentioned before, while pricing decisions provide a finer way to pull customers into cost-effective options by providing differentiated prices. Uniform pricing can make demand imbalanced, while differentiated pricing can help to smooth demand based on demand forecasting. Furthermore, dynamic pricing is even more powerful to adjust the demand in real-time. As technology developed, differentiated or dynamic pricing of delivery options is applicable for two reasons. One reason is the flexibility of both retailers and customers. Retailers can easily confidentially communicate with customers by phone or email to provide discounts and negotiate with customers in real-time. Customers have the flexibilities to change delivery options to some extent as well. Additionally, data availability enriched by the storage of all historical sales information enables more accurate demand forecasting, and also increase the possibility of customer relationship management. However, if the pricing policy is too complicated, customers may get confused and feel distrustful of the company (Garbarino and Lee, 2003).

For pricing decisions, the idea of incentive pricing, more commonly known as dynamic pricing, has been executed in a variety of industries to encourage customers to purchase products or service at the 'right time at the right price' (Lin, 2006). Introduced in the airline industry (Rothstein, 1971; Littlewood, 2005), it has now widely spread to areas such as the hotel industry, car rental, fashion, cruises and for stadium tickets. The use of dynamic pricing has been shown to be successful, and this has encouraged innovative ways of applying the approach.

For a detailed review of dynamic pricing, please refer to Elmaghraby and Keskinocak (2003). Cleophas and Ehmke (2014) classified the application of demand management in VRP field as value-based demand. In traditional dynamic pricing, the cost of each product is the same, and price is differentiated according to time-value and demand (e.g. airline ticket). In their paper, a combination problem of cost minimal routing and value-based order acceptance techniques were investigated. Yang et al. (2014) investigated dynamic delivery pricing problems in e-fulfilment of an online grocery store. They studied an advanced and the state of the art demand model, which is the multinomial logit choice model. They formulated the pricing problem as a stochastic dynamic programming problem and solved it using approximation method. Two different ways were proposed to estimate opportunity costs. However, the cost of delivery was estimated using a simplified model, and vehicle routing part of the problem was not investigated.

Among the existing literature, the problem closest to the one in this research is addressed by Campbell and Savelsbergh (2006). These authors considered the problem of online groceries delivery, where customers came with their preferred time windows, and used a two-stage process to reduce travel costs; the first stage solving a vehicle routing and scheduling problem and the second solving a dynamic pricing problem. In their dynamic VRPTW problem, requests for a service for a week could arrive one week beforehand, and these were scheduled at the customers' preferred time windows. Then the new arriving customers were inserted into one of the existing routes. They created a set of feasible routes for existing customers, and the new customer was inserted into these feasible routes to find the lowest insertion cost time windows. Construction and insertion heuristics were used to build initial schedules and check insertion feasibility and costs following Bent and Van Hentenryck (2004)'s approach. The minimum insertion costs from all schedules for each time window t were calculated and represented as C_t. The insertion cost C_t integrated routing and pricing models; it was an output of the routing model and an input of the dynamic pricing model. At the second stage, a dynamic pricing model was constructed based on C_t. The decision variables in their model were incentives given to selected time windows. It was assumed that giving an incentive for a time window would increase the probability that a customer chose it and equally decreased the probabilities of other time windows without incentives. They aimed to shift customers' demands by giving pricing incentives to delivery time windows that result in minimum operating costs. Their paper offered insights into the use of incentive schemes to substantially reduce delivery costs, considering a distance problem. However, the limitation of their pricing model is that the amount of incentives offered is highly restricted to customer's initial probability of choosing a time window, and this means they need to pre-select the time windows that will receive incentives instead of incorporating this into the optimisation procedure.

There is very little research on combining the areas of scheduling and pricing, and as far as we are aware there is none on using pricing incentive for improving the schedule of vehicles when the objective is to reduce carbon dioxide emissions costs.

The problem that is considered in this thesis is that of online installations and maintenance service delivered to customers' sites. Customers have a time preference as to when they would like to receive a service. We aim to use dynamic pricing to provide customers with incentives to purchase in time windows that have low CO_2 emissions levels and emissions cost. It is a significant problem because it can help reducing operations costs, improving corporate reputations while also increasing profitability.

Demand forecasting is an essential element in the analysis of customer choice behaviour according to changing prices. In the book by Ben-Akiva and Lerman (1985), the application of discrete choice demand model in transportation systems has been studied, including the methods of estimating unknown parameters in the demand model, and its goodness of fit has been justified. Choice models predict the likelihood of customers purchasing a specific product from a set of related products based on their relative attractiveness. Similar model fitting and measurement methods also exist for linear demand models. In this research, we consider that only price will affect each delivery options' attractiveness.

2.5 Summary

In this chapter, literature related to this research is reviewed. The problem of interests is pricing and scheduling of engineers of a service provider, so studies on the vehicle routing problems with time window constraints and its variants are reviewed. The primary objective of our problem is to minimise CO_2 emissions which makes the problem a green vehicle routing problem. Furthermore, the travel speed is one of the most important factors that affects emissions from vehicles and it is changing during the day as there are regular congested periods when people go to work and return from work. Many studies of green VRP apply time-dependent VRP settings and those papers are reviewed. There is an interesting problem of optimal departure times. As travel speeds are time-dependent, different departure times will make a difference in the final solutions. Several papers considering the departure times optimisation problems are reviewed. The dynamic pricing problem is often made online and requires a relatively short solution time. Exact methods normally consume more computational time hence metaheuristics are also applied. The state of the art metaheuristics methods are of two types: local search or population based search. Tabu search, simulated annealing and variable neighbourhood search are local searches and Genetic algorithm and ant colony algorithm belong to the second type. Those metaheuristics are explained and reviewed in this chapter. Accurate estimation of CO_2 emissions is important for this research so all existing emissions models are reviewed in details and compared. Pricing part is reviewed in the context of vehicle routing and scheduling, and to our best knowledge, the pricing of green services is new in literature.

In the next chapter, the CO_2 emissions calculator uses in this research and its real-life application in a field service provider will be described. The potential benefits and impact of introducing green logistics in field scheduling and routing will be demonstrated as well.

3 CO_2 emissions calculator

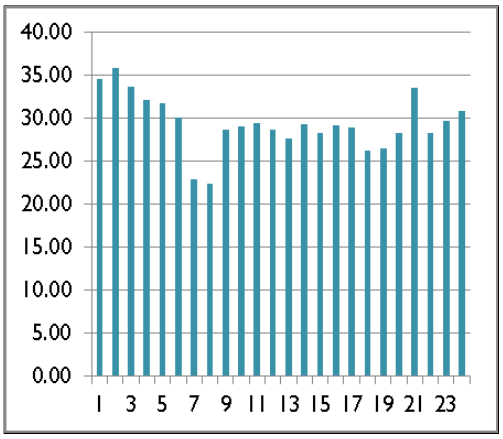
In this chapter, the CO_2 emissions calculator used in this research is described. Different models that could be used to calculate emissions and their comparisons are reviewed in Chapter 2. Furthermore, the importance of accurate estimations of emissions for green VRP/VSP is addressed in Chapter 2 as well. The CO_2 emissions calculator will be applied throughout this research and it has been implemented in a real-life field service provider as part of their introducing green logistics into the business logic plan. The real-life benefits and impacts are observed.

The CO_2 emissions calculator is programmed as a Java application with several features. The application includes built-in speed profiles for each geometric area (domain) based on historical data, and an embedded road distance estimation algorithm based on a linear regression model. Furthermore, it has two well-applied emissions models that end users can choose from depending on the information that they have available. Both emissions models require the knowledge of a vehicle details and the application can map this to the corresponding emissions formula. A Java desktop graphical user interface (GUI) application is developed using Java Swing framework to help end users to map a vehicle to a specified emissions formula. This chapter demonstrates the data analysing and cleaning process to obtain the speed profiles, the linear regression model on road distance estimation, and the business logic of the GUI application. To test the accuracy of the emissions calculators, the MEET emissions calculator's computational results will be compared with vehicle manufacturing information on a real road journey. The discrepancy between CO_2 emissions minimisation and traditional travel time minimisation will be demonstrated using numerical experiments with the NAEI emissions calculator and discussed in depth. Furthermore, in the last section, business applications and real-life impacts of the emissions calculator are demonstrated.

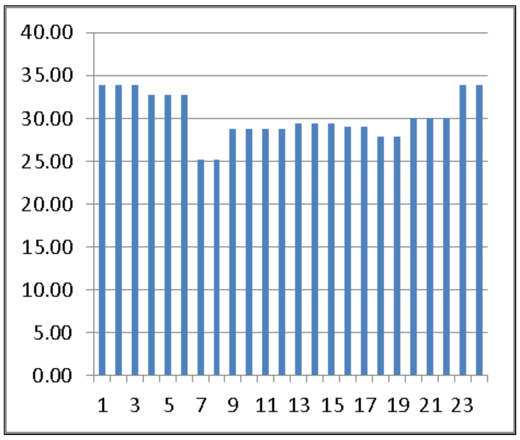
The rest of chapter is organised as follows. The three important components of the CO_2 emissions calculator namely the time-dependent speed profile and the emissions rate profile, the algorithm to estimate road distance based on coordinates and vehicle details mapping to emissions formulas are described in Section 3.1, Section 3.2 and Section 3.3 respectively. The emissions calculator is verified on a real-life test case in Section 3.4. Section 3.5 discusses the differences between CO_2 emissions minimisation objective with traditional objectives such as travel time. The real-life business applications of the proposed emissions calculator are demonstrated in Section 3.6 together with its impact and benefits. Conclusions are given in Section 3.7.

3.1 The speed profile & the emissions rate profile

Based on the real-life travelling journal of a field service provider's engineers, a real-life travel speed profile was generated for London, UK. This profile captured the average traffic condition throughout the day. It was expected that there would be two peak-hour periods, where the travel speed is slow. The two peak-hour periods are when people go to work and from work. From the real-life field service provider's live tasks database, we have task locations so that we can calculate road distance between tasks. We also have the start time for travelling to each task and the arrival time, so we were able to calculate the total travel time between tasks. In this way, we calculated the travel speed in the hour that engineer started to travel to a task. A day was divided into 24 one-hour time slots, 0:00-24:00. If a travel journey happened in one slot, the travel speed in that slot was recorded. If a travel journey happened across several slots, the same travel speed was recorded for every slot. Then for each slot we took the average speed for that slot of 121 days to construct a speed profile for that domain, as shown in Figure 3.1(a). We have cleaned the data set by eliminating speed data over 200 km/h, as that number was far above the national speed limit and was treated as a fraud record. After data cleaning, we have had 7795 speed data entries for London.



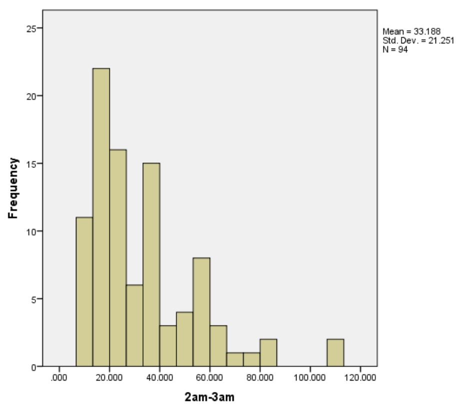
(a) Hourly speed profile



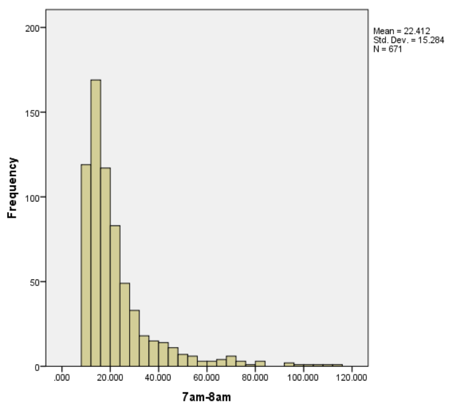
(b) Smoothed hourly travel speed profile

Figure 3.1: Speed profile of London, UK (km/h).

In Figure 3.1, we can see that there are two peak-hour periods, one early morning and the other in the late afternoon. That matches our expectation. The problem with this profile is for some hours we have many data entries, but for some hours we have only a few. That is because most of the tasks happen in engineer working hours, which are 07:00 -19:00. Detailed statistics of selected time slots are shown in Figure 3.2.



(a) Speed data for time slot 2am-3am



(b) Speed data for time slot 7am-8am

Figure 3.2: Statistics and frequency plots of selected time slots.

Comparing Figure 3.2(a) and 3.2(b), we can see that in time slot 2:00am-3:00am, there are not many data entries (94) in these 121 days. The lack of data is because not many tasks are appointed at this time of the day. The standard deviation of this data set is 21.25. For time slot 7:00am-8:00am, we have in total 671 data entries in 121 days, and the standard deviation of this data set is much smaller, which is 15.284. To reduce the total variance of all the data, several time slots are grouped to smooth the data pattern. The grouping criteria are to cluster time slots with similar means and try to allocate only one group for midnight period. The reason is that there is less traffic at midnight, and travel speeds should be similar.

We have grouped slots with similar speed profile for example 06:00-08:00 or 17:00-19:00. This is carried out first by eyeballing slots with similar speed and then those slots were tested using independent t-tests to compare if their means are significantly different. The speed profile after smoothing is shown in Figure 3.1(b). Before grouping, the mean variance is 19.359, and after it is 18.983. Through clustering, we reduce the variance by 1.92%. The field service provider updates this speed profile every three months automatically. Data storage requires only the mean value for each hour and the number of entries. Ehmke et al. (2016) used a similar method for doing speed analysis for 24 hours in Stuttgart, Germany, with 230 million speed data (3 years), and they clustered each domain into sub-districts.

3.2 Estimating road distance from latitude and longitude

Given the locations (longitude and latitude) of two sites, line distance between these two places can be calculated using the Haversine formula (Source: `http://www.movable-type.co.uk/scripts/latlong.html`) as shown below:

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2) \tag{3.1}$$

$$c = 2 \cdot \arctan 2(\sqrt{a}, \sqrt{1-a}) \tag{3.2}$$

$$d = R \cdot c \tag{3.3}$$

where φ is latitude, λ is longitude, R is earth's radius (mean radius = 6371 km), note that angles need to be in radians. This formula calculates the shortest distance over the earth's surface between two sites ignoring terrains. However, in real life vehicles do need to follow the road network to travel between two locations. Road distance tends to be longer than line distance calculated from the formula.

Google Maps could be used to get a better estimation of road distance, but more time needs to be spent to get responses from the server especially in large-scale computations. To establish a quick and reasonably accurate method for travel distance estimation, we tried to work out a relationship between the Haversine distance and the real road distance using a large set of sample data. In each domain, 2614 task pairs were randomly selected, and the line distance between each pair of tasks was calculated. Then, Google Maps Distance API was called using Java to get road distance estimates between the same pairs of tasks. Based on Yang et al. (2014), there is a linear relationship between Haversine distance and road distance, a linear regression analysis can be carried out. We applied a simple linear regression technique to train a linear regression model on a training data set and tested the model on a test data set. The 2614 data pairs were randomly divided into a training dataset and a test dataset with the proportion of $2:1$. So the training data set contains 1742 data points, and the test dataset includes 872 data points. It is reasonable to assume that for each domain the parameters in equation (3.4) are slightly different, as the geographic areas of domains are different. For example, London is a small domain comparing to Scotland. R software was used to carry out the linear regression method. Let us take London as an example. The regression results are shown in Appendix A together with the R script. The linear model is a suitable model with a 93.71% R^2 value, indicating high goodness of fit value. P-value of the F statistic is $0.000 < 0.05$, which shows the model is significantly better than the intercept-only model at the 95% confidence level. From the coefficients table, both the constant term and the independent variable are statically significant at the 99% confidence level, which means both the constant and the line distance are necessary. The formula for London is summarised as:

$$RoadDistance = 0.668 + 1.305 \times LineDistance + error. \tag{3.4}$$

The visualisation of this linear model with the data points in the training set and the test set are shown in Figure 3.3 and Figure 3.4 respectively.

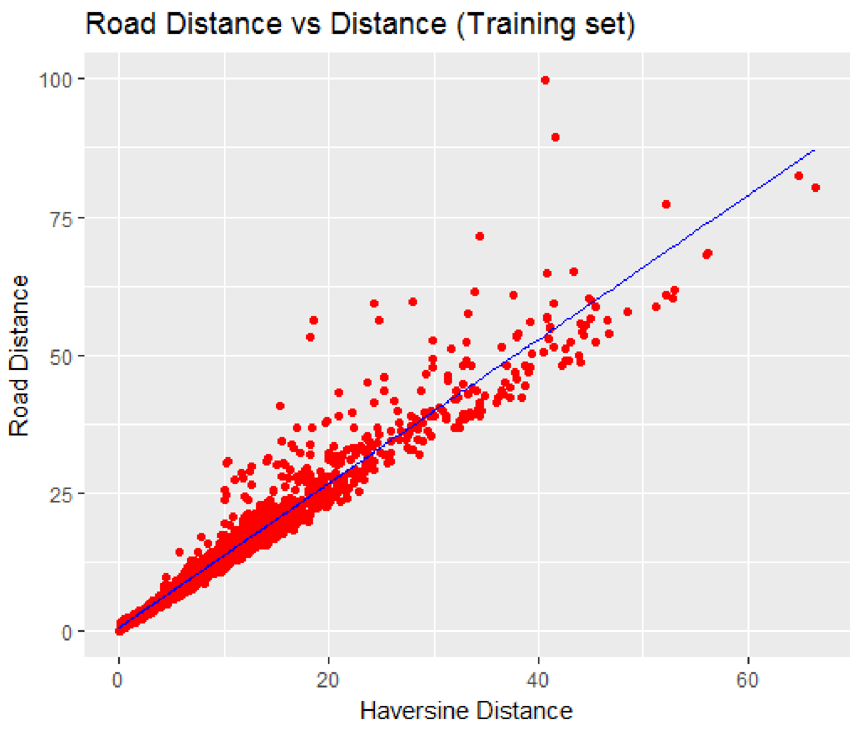


Figure 3.3: Road Distance vs. Haversine Distance (Training set) (Red: real data, Blue: Learning model).

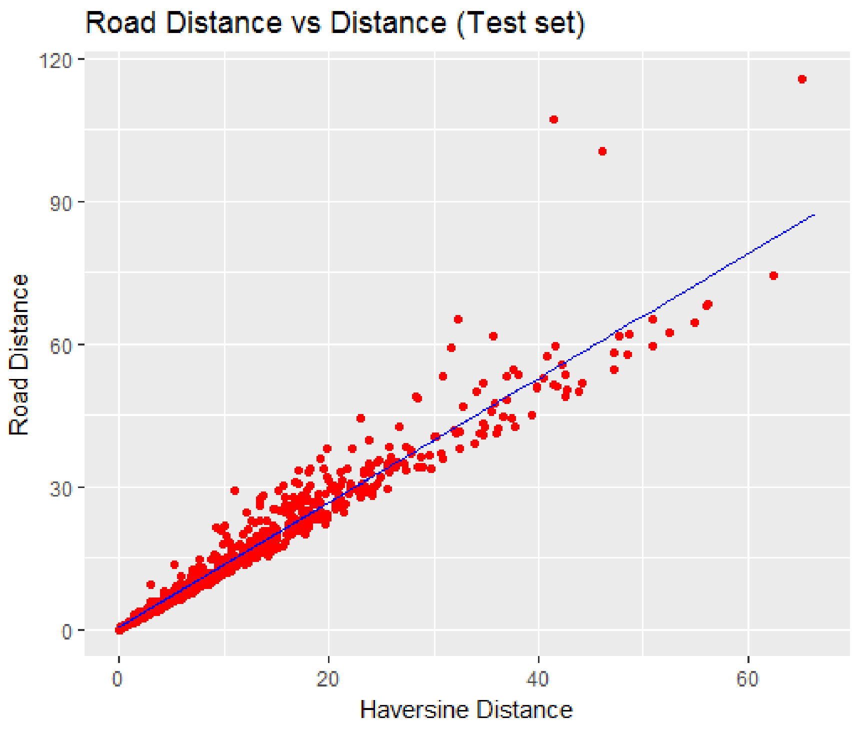


Figure 3.4: Road Distance vs. Haversine Distance (Test set) (Red: real data, Blue: Learning model).

3.3 Vehicle specification mapping to emissions models

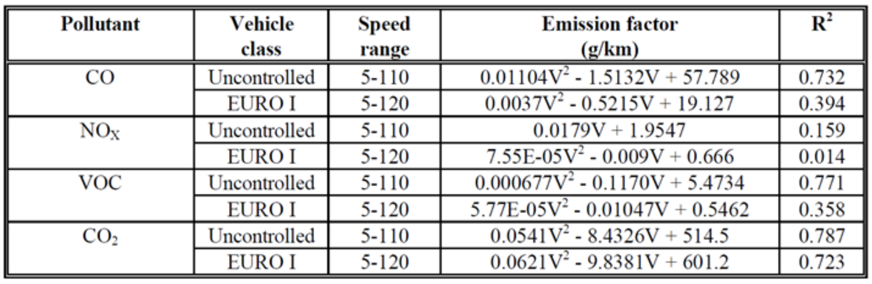
There are 265 CO_2 emissions formulas for different vehicles specifications in the NAEI project and 33 different emissions formulas in the MEET project. Both projects have formulas for various types of pollutants such as CO_2, CO, HC and NO_x. Given the information of an engineer's vehicle details as listed in Table 3.1, each vehicle was mapped to a particular formula within the NAEI/MEET formulas set. A Java desktop application GUI was developed which automatically helps users to match vehicles details with its corresponding numerical emissions formulas. Appendix B shows a snapshot of the Java desktop GUI application.

Vehicle details	NAEI Model	MEET Model
Basic Type (e.g. LDV, HDV)	✓	✓
Fuel Type	✓	✓
Gross Weight	✓	✓
Emissions standard	✓	
Engine capacity	✓	

Table 3.1: Information of engineer's vehicle details.

3.4 Test the accuracy of the emissions calculator

In this research, in order to calculate the amount of carbon dioxide emissions generated by vehicles, we use the instantaneous CO_2 emissions model proposed by the MEET project and its UK version the NAEI project. The MEET project aimed to provide a basic, European-wide procedure for evaluating the impacts of transportation on air pollution. This would include comprehensive and up-to-date information on emissions rates and traffic characteristics, as well as methods of calculation. For different types of vehicles, it provided different emissions calculating formulas. For example, the emissions model for light-duty gasoline vehicles with weight less than 3.5 t is shown in Figure 3.5. We focus on the last row, which is the model for carbon dioxide emissions.

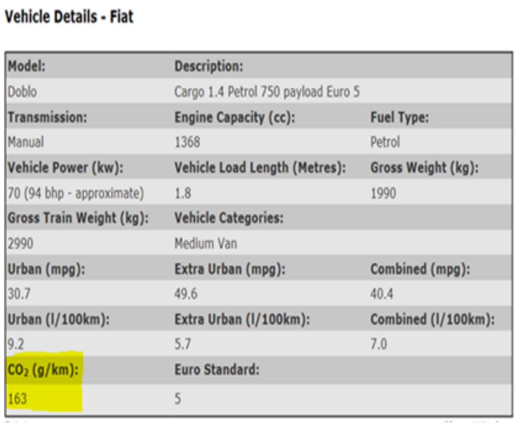


Pollutant	Vehicle class	Speed range	Emission factor (g/km)	R^2
CO	Uncontrolled	5-110	$0.01104V^2 - 1.5132V + 57.789$	0.732
	EURO I	5-120	$0.0037V^2 - 0.5215V + 19.127$	0.394
NO_X	Uncontrolled	5-110	$0.0179V + 1.9547$	0.159
	EURO I	5-120	$7.55E\text{-}05V^2 - 0.009V + 0.666$	0.014
VOC	Uncontrolled	5-110	$0.000677V^2 - 0.1170V + 5.4734$	0.771
	EURO I	5-120	$5.77E\text{-}05V^2 - 0.01047V + 0.5462$	0.358
CO_2	Uncontrolled	5-110	$0.0541V^2 - 8.4326V + 514.5$	0.787
	EURO I	5-120	$0.0621V^2 - 9.8381V + 601.2$	0.723

Figure 3.5: An example of the MEET project emissions models: speed dependency of emissions factors for gasoline light duty vehicles<3.5 t (Table A22 in the MEET project report).

To validate the accuracy of the MEET project models, we compare the results given by the MEET models with vehicle manufacturing information and fuel efficiency studies. For the real-life field service provider, one of their fleets includes medium vans similar to Fiat Doblo. The manufacturing information of Fiat Doblo is shown in Figure 3.6. It is a petrol car, and its gross weight is 2990 kg. This vehicle belongs to the category of light-duty petrol vehicles and is mapped to the CO_2 formula in Figure 3.5.

(a) Picture of Doblo



Vehicle Details - Fiat

Model:	**Description:**	
Doblo	Cargo 1.4 Petrol 750 payload Euro 5	
Transmission:	**Engine Capacity (cc):**	**Fuel Type:**
Manual	1368	Petrol
Vehicle Power (kw):	**Vehicle Load Length (Metres):**	**Gross Weight (kg):**
70 (94 bhp - approximate)	1.8	1990
Gross Train Weight (kg):	**Vehicle Categories:**	
2990	Medium Van	
Urban (mpg):	**Extra Urban (mpg):**	**Combined (mpg):**
30.7	49.6	40.4
Urban (l/100km):	**Extra Urban (l/100km):**	**Combined (l/100km):**
9.2	5.7	7.0
CO_2 (g/km):	**Euro Standard:**	
163	5	

(b) Details of Doblo

Figure 3.6: Fiat Doblo manufacturing information.

In this study, we take a daily schedule of an engineer who drives this car to customer sites for providing the services, as a test case. The schedule of Engineer Bob (alias) on 17th July 2016 is selected. The journey of this engineer on this day is shown in Figure 3.7.

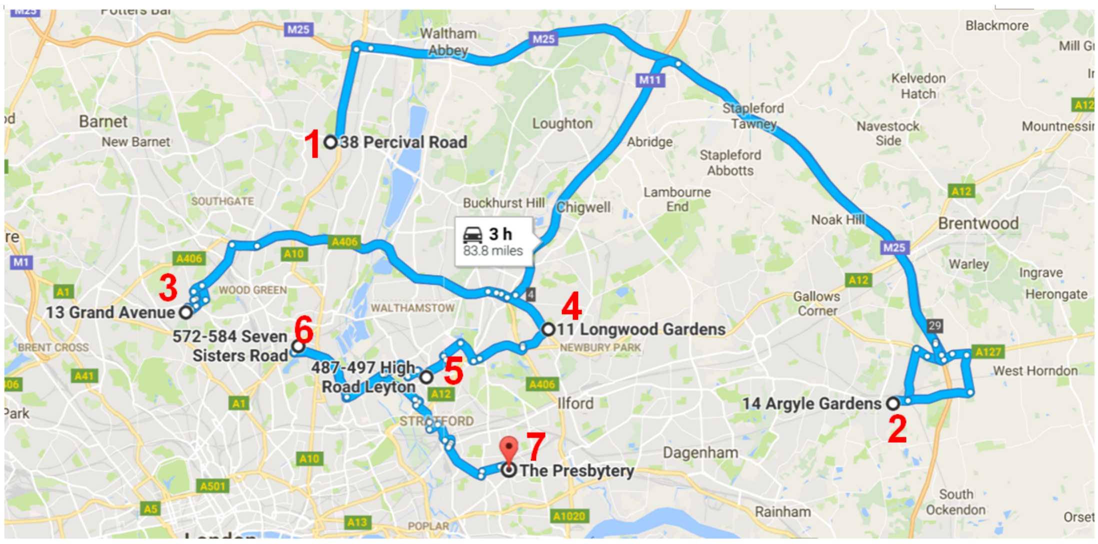


Figure 3.7: One-day live schedule of an Engineer.

Based on the mapped emissions formula of the MEET project in Figure 3.5, we calculate the amount of carbon dioxide emissions for each journey of this engineer on this day. The results are shown in Table 3.2. The vehicle is estimated to have generated total CO_2 emissions of 37.557 kg for the daily schedule. As can be seen from the table, the speeds of the journeys are very slow (around 30 km/h). Manufacturers do not provide emissions or fuel consumptions information for such slow speed. So we compare the emissions results given by the MEET formula and the manufacturer information for the conditions for which manufacturer information is available. According to Fiat manufacturing information, its CO_2 emissions rate is 163 g/km. The total distance travelled in this particular schedule is 119.988 km. Based on manufacturing information this route plan will generate 19.397 kg CO_2. This figure is based on the most efficient free-flow travel speed. If we substitute the free-flow travel speed (113 km/h) into the MEET model, we get 22.122 kg CO_2. This figure is similar to the previous calculation.

However, travelling at the free-flow speed is impossible especially in an urban area where there is a speed limit of 48 km/h in the UK. Let us look at the fuel efficiency information of Fiat Doblo. The urban miles per gallon (mpg) is 30.7, which means one litre of fuel can power travel of 10.805 km in a metropolitan area.

One litre of petrol burnt thoroughly will generate 2.31 kg CO_2, based on material carbon composition. So based on its fuel efficiency in an urban area, it will emit 25.44 kg CO_2. On the other hand, using the MEET formula with 48km/h urban travel speed limit will produce 27.891 kg CO_2. Again this result is similar to the calculation based on manufacturing information.

TaskID	StartTravelTime	Distance(km)	Speed(km/h)	Emission(kg)
1	09:30	30.896	28.735	9.790
2	11:13	27.193	28.735	8.616
3	16:55	10.241	27.856	3.293
4	18:17	18.056	27.856	5.806
5	19:19	6.462	30	2.005
6	20:15	8.414	30	2.610
7	22:20	6.542	34	1.899
8	23:17	12.184	34	3.537
Total				37.557

Table 3.2: Carbon dioxide emissions amount calculated by MEET model.

Under each of the above two conditions, there is only a small difference between the results of the two calculation methods. Considering that the manufacturer test may not have included the waiting and acceleration and deceleration at traffic lights, we can conclude that the MEET project model generates a comparable amount of CO_2 as provided by manufacturing information and fuel efficiency studies, under different scenarios. Therefore, the MEET model will be in use in this research together with its UK version the NAEI model.

3.5 The objective of minimising CO_2 emissions

A primary question that arises about green vehicle routing problem is whether it is different from the traditional travel time minimisation problem. In this section, we are going to demonstrate the possible conflicts between travel time objectives and emissions objectives under different scenarios and test the scenarios with numerical experiments.

The total amount of CO_2 emissions is:

$$E = \int_{t=0}^{t=T} f(v_t)\, v_t dt, \tag{3.5}$$

where E is the emissions in g and v_t is the time-dependent travel speed in km/h. For a specific type of vehicles (Light goods diesel vehicle, EURO 5, 1.76-3.5 t), the NAEI formula R110 is:

$$f(v_t) = \frac{3903.4}{v_t} + 65.04 + 0.44246 v_t + 0.01324 v_t^2 - 6.7705 \cdot 10^{-5} v_t^3 + 7.07 \cdot 10^{-7} v_t^4, \tag{3.6}$$

where $f(v_t)$ is the emissions rate in g/km. Solve $f^{'}(v_t) = 0$ gives us the optimal speed that minimise emissions in the range of valid speeds. $v_t^* = 49.440$ km/h. The relationship between emissions and travel speeds or travel time are plotted in Figure 3.8, and we can observe that the relations are not linear nor monotonic.

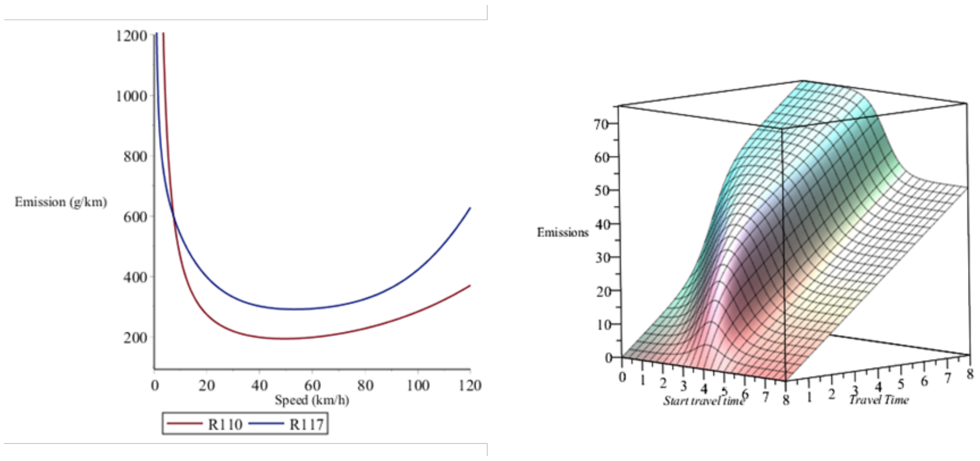
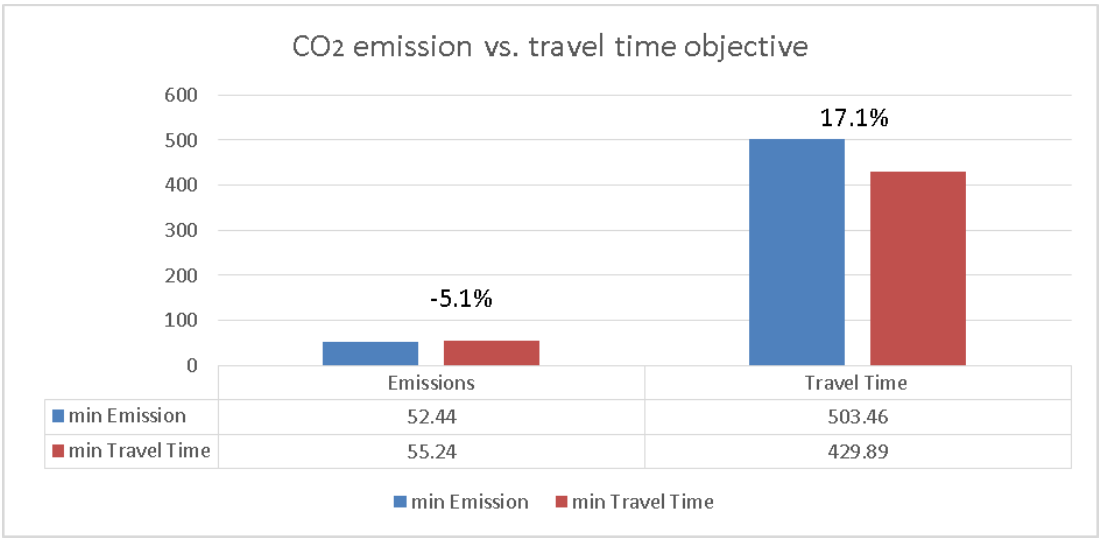


Figure 3.8: Left: relationships between CO_2 emissions and travel speed for two different vehicles (R117 on p.55); Right: relationships between CO_2 emissions, start travel time and total travel time derived from equation (3.5).

3.5.1 Scenario 1: constant speed versus time-dependent travel speed

If the travel speed is a constant, which is normally assumed to be a constant in the traditional VRP problem, we will have $v(t) = v$. Then equation E can be rewritten as: $E = f(v) \cdot v \cdot T$, where $f(v)$ is constant as well. In this case, minimising T is the same as minimising E. In the time-dependent travel speeds settings, minimising T and E can be different. To minimise T, the scheduler may choose to schedule tasks to travel at 70 km/h time period and during 50 km/h time period to carry out the work and avoid travelling. In this way, the scheduler could save the total travel time. However, for traversing the same pair of nodes, we know that travel at 50 km/h is the optimal speed which causes an increase in travel time compared to travel at 70 km/h but reduces the amount of CO_2 emissions.

Assuming a homogeneous fleet as we want to test the effect of time-varying travel speed only, we carried out a numerical experiment on a small test case with five customers and two vehicles. The test case was formulated as a VRP model and solved to optimality using Xpress. The details of the mathematical model will be shown in Chapter 6. Xpress applies a powerful branch and bound method to tackle this form of optimisation model to global optimality. We first solved the time-dependent VRP model with the objective to minimise total CO_2 emissions, and then we solved the same model with the objective to minimise total travel time. The results of the global optimal solutions are shown in Figure 3.9. The results demonstrate that for this specific type of commercial van under the normal traffic condition with fluctuating speeds, there is a discrepancy between CO_2 emissions minimisation and travel time minimisation. The greenest route plan can reduce the total emissions by 5.1% but at a sacrifice of 17.1% total travel time.



	Emissions	Travel Time
🟦 min Emission	52.44	503.46
🟥 min Travel Time	55.24	429.89

Figure 3.9: Comparisons between CO$_2$ emissions and travel time objective.

3.5.2 Scenario 2: different vehicles

Next, we consider the difference of emissions produced by different vehicle types. For a specific type of vehicle, the NAEI formula R110 is in equation (3.6). For another type of vehicle (Rigid heavy goods diesel vehicle, EURO V, 3.5-7.5 t), the NAEI formula R117 is:

$$f(v_t) = \frac{502.59}{v_t} + 690.15 - 26.109 v_t + 0.65957 v_t^2 - 8.3582 \cdot 10^{-3} v_t^3 + 5.2817 \cdot 10^{-5} v_t^4 - 1.1815 \cdot 10^{-7} v_t^5. \tag{3.7}$$

Let us address the constant speed scenario. If the travel speed is 70 km/h, to minimise travel time, there is no difference between allocating tasks to which type of vehicles. However, when considering minimising emissions, the allocation is vital. In this case, the scheduler will try to allocate more travel to vehicle R110 as it generates less emissions on speed $[10, 120]$ as shown in the graph on the left in Figure 3.8.

Assuming constant travel speed as we want to test the effect of different vehicles on emissions only, we carried out a numerical experiment on the same test case as in Scenario 1. The travel speed is set to be 70 km/h for all periods, and the two vehicles, one commercial van and the other is a heavy goods van (HGV), have different emissions rates at this speed: 210 g/km and 559 g/km respectively. We first solve the VRP model with the objective to minimise CO_2 emissions, and then we solve the same model with the objective to minimise total travel time. The constraints are two vehicles available (one commercial and one HGV), working duration of engineers and time window constraints of tasks. The GeoMap demonstration of the global optimal solutions is shown in Figure 3.10, where dots mean locations on the map and lines indicates routes between places.

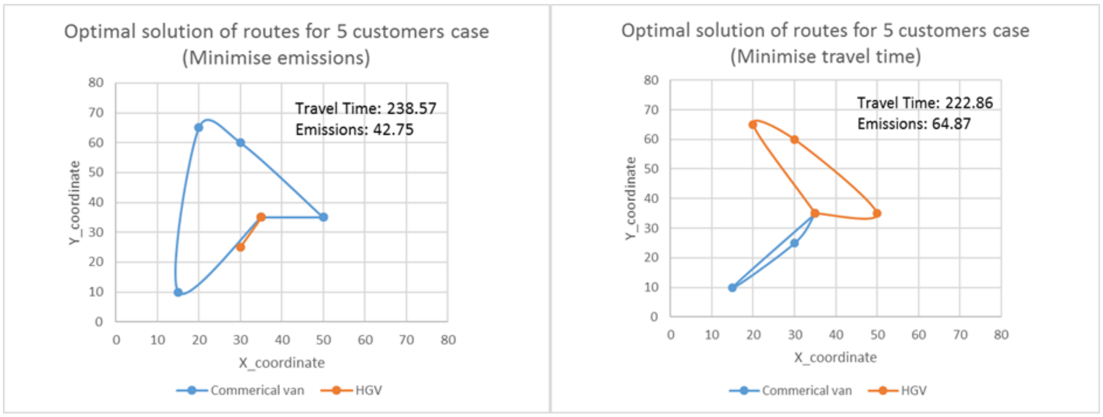


Figure 3.10: Comparisons between CO_2 emissions and travel time objective on Cartesian coordinate map system (tested on randomly generated locations).

The results demonstrate that even with a constant travel speed, there is still a conflict discrepancy between CO_2 emissions minimisation and travel time minimisation if the fleet is heterogeneous. The greenest route plan can reduce the total emissions by 34.1% but at a sacrifice of 6.7% entire travel time. Based on the GeoMap, we could observe that with emissions minimisation routes (Figure 3.10, left), more tasks are assigned to the commercial van, and with travel time minimisation routes (Figure 3.10, right), tasks are evenly assigned to both vehicles.

3.6 Benefits and business application of CO_2 emissions calculator

The calculator of CO_2 emissions has been introduced into a field service provider to increase the business awareness of green logistics. For the industry collaboration, the emissions calculator has been implemented in different ways for different users and various purposes. Benefits are observed or expected to be observed since the introduction of green logistics into their real-life field scheduling and routing decision system. A summary of all the benefits and business applications is demonstrated in Figure 3.11. The details of business applications and the impacts are explained in Section 3.6.1 and 3.6.2.

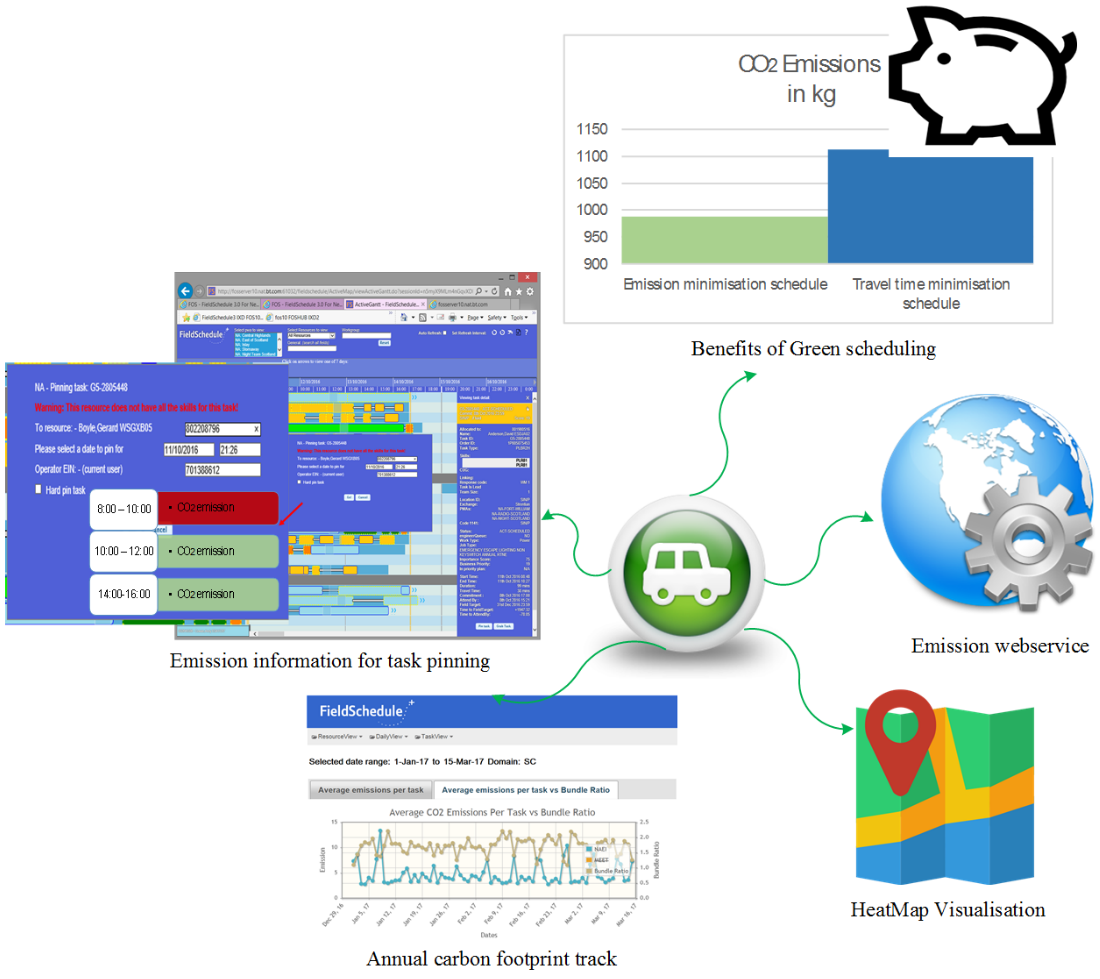
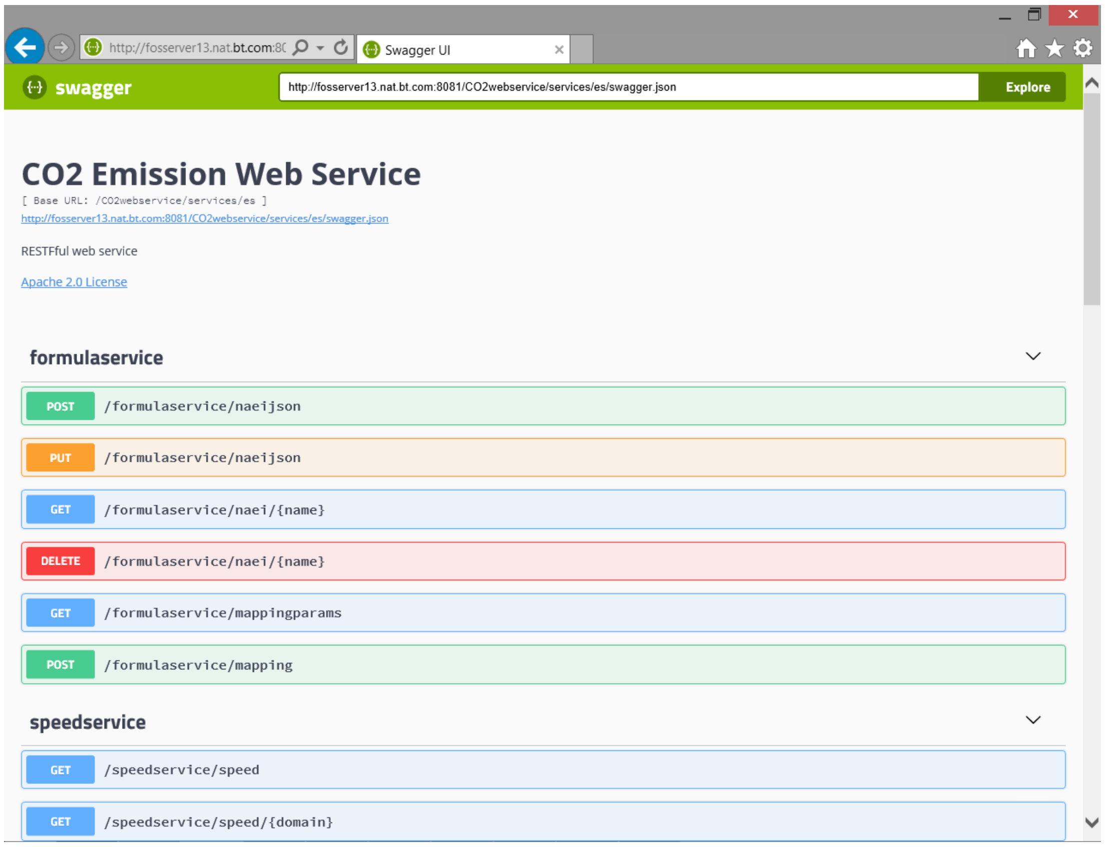

CO2 Emissions in kg
1150
1100
1050
1000
950
900
Emission minimisation schedule
Travel time minimisation schedule
Benefits of Green scheduling
NA - Pinning task: GS-2805448
Warning: This resource does not have all the skills for this task!
To resource: - Boyle,Gerard WSGXB05
Please select a date to pin for
Operator EIN: - (current user)
Hard pin task
8:00 – 10:00
CO2 emission
10:00 – 12:00
CO2 emission
14:00-16:00
CO2 emission
Emission information for task pinning
Emission webservice
FieldSchedule
Selected date range: 1-Jan-17 to 15-Mar-17 Domain: SC
Average emissions per task
Average emissions per task vs Bundle Ratio
Average CO2 Emissions Per Task vs Bundle Ratio
Annual carbon footprint track
HeatMap Visualisation

Figure 3.11: Business applications.

3.6.1 Web portal and web service for on-demand CO_2 emissions data consumption

A web portal was designed to track the field service provider's carbon footprint at different granularities for a selection of time periods. The historical data of company's schedules come from backend data schema. This web portal was developed under Java JSF framework, and the connection to the company's database is built using Hibernate.

Users such as a tasks manager may want a one-time check on one schedule or want to populate the emissions to a specific data table in the data schema for storage, and the application used by this manager is running on a different platform with a different framework. In this situation, web service is required. Web service is defined as the communication between client and server application through the World Wide Web's (WWW) HyerText Transfer Protocol (HTTP). RESTful Web Services are based on Representational State Transfer (REST) architecture, where data and functionalities are treated as resources and each can be assessed using a unique Uniform Resource Identifier (URI), e.g. web link. It regulates both the client and server side architecture and designs a stateless communication protocol, e.g., HTTP. The advantages of RESTful Web Services are loose coupling, which means the client side need not be aware of the back end code or implantation details. So this provides a platform and programming language independence. The implementation uses Java Apache CXF (JAX-RS). The web service is deployed on the company server using Tomcat 8.5. Swagger.io is implemented to visualise the resources and functionalities of the CO_2 emissions services to generate beautiful and interactive documentation automatically. Front-end users could easily test and try the service out with the swagger user interface of the web service. The snapshot of the web service is shown in Figure 3.12. The try-out of the web services is shown in Appendix C.


http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/swagger.json

CO2 Emission Web Service

[Base URL: /CO2webservice/services/es]

http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/swagger.json

RESTFful web service

Apache 2.0 License

formulaservice

POST /formulaservice/naeijson

PUT /formulaservice/naeijson

GET /formulaservice/naei/{name}

DELETE /formulaservice/naei/{name}

GET /formulaservice/mappingparams

POST /formulaservice/mapping

speedservice

GET /speedservice/speed

GET /speedservice/speed/{domain}

Figure 3.12: CO_2 emissions web service.

The available services are:

- The NAEI emissions formulas lookup
- Vehicles specifications map to the NAEI formulas
- Calculate emissions of travelling to a task based on a speed profile or another web service
- Calculate emissions of travelling to a task based on telematics data
- Populate emissions table of a period based on speed profile
- Populate emissions table of a period based on telematics data
- Populate average emissions per day table of a period
- Populate average emissions per day per engineer table of a period

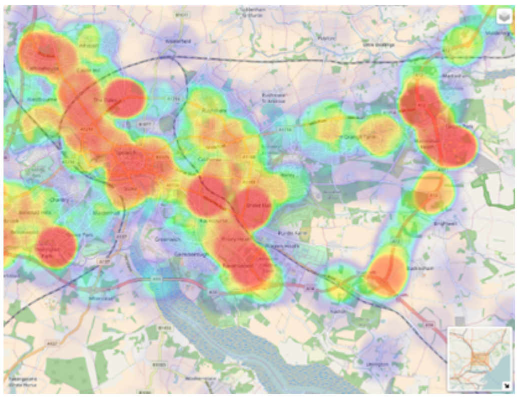
The emissions of travelling to a task can be used for tasks pinning which is allocating or recommending tasks to an engineer. With the information of CO_2 emissions listed alongside the tasks recommended to engineers, engineers may select the task with the smallest amount of CO_2 emissions. This will help the company to reduce emissions in the long run.

3.6.2 Heat Map visualisation

The business impact of the application of introducing green logistics into field service scheduling and routing can be illustrated graphically, by analyzing a sample corresponding to a scheduling profile of 179 engineers, with activities between 1st May and 15th May 2017. This sample resolves 4173 unique tasks in Colchester and Ipswich domain, where 94 active service sites reported activities. It can be noticed from Figure 3.13(a), higher emissions from 7 to 9 hrs. compared to the picture reported in Figure 3.13(b), which, in turn, reports the recorded emissions from 11 to 13 hrs. In the figure, roads in the purple represent the transited route with certain CO_2 emissions. Yellow colour indicates higher CO_2 emissions and red colour spots a concentrated CO_2 emissions point. The heat maps shown in the figure reflect the situation in a working day. The amount of emissions of a particular point on the map is an accumulated amount for the time period measured, so the higher level of emissions concentration is caused by two possible reasons, 1) high frequency of travel, 2) congestion. As the task levels are similar in the two periods, the worse congestion in the 7 to 9hrs period makes the heat map more amber in colour.

The threshold values are adjusted for comparison purposes according to the desired snapshot. Threshold values define the range of values corresponding to threshold colours. For example, if the emissions value is in the range of medium to high, the colours shown on the map will be red and amber colours. Therefore, it is categorised in according to certain ranges, such as

$$t_{low} = x[min, \mu - \sigma^2] \tag{3.8}$$


(a) from 7 hrs. to 9 hrs. (Inclusive).

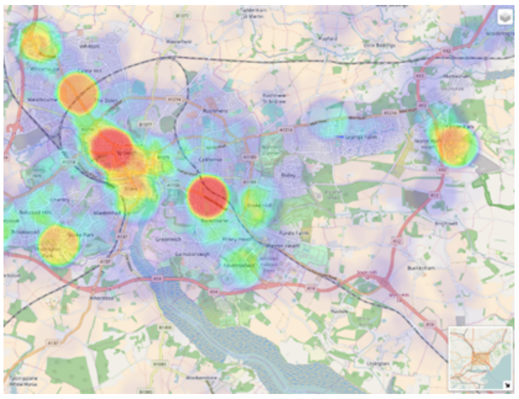

(b) from 11 hrs. to 13 hrs. (Inclusive).

Figure 3.13: CO_2 Heat map visualisation.

$$t_{medium} = x[\mu - \sigma^2, \mu + \sigma^2] \tag{3.9}$$

$$t_{high} = x[\mu + \sigma^2, max], \tag{3.10}$$

where x corresponds to the CO_2 value to be represented, min is the minimum CO_2 value recorded, μ is the mean of the CO_2 emissions after computation, σ^2 is the standard deviation of this CO_2 computation in the specific time window being evaluated and max is the highest recorded value for the CO_2 in the sample in that slot. Therefore, the resulting thresholds are not fixed and correspond to the particular desired granularity.

3.7 Summary

In this chapter, the components of a Java program used to calculate CO_2 emissions are introduced. The first component calculates the speed profiles for each domain, and specifies how they are constructed. Then a simple linear regression model is built to estimate the road distance from latitude and longitude. Furthermore, it has two well-applied emissions models that end users can choose from depending on the information obtained. Both emissions models require the knowledge of vehicle details and these are mapped to corresponding emissions formulas. A desktop Java GUI application is developed using Java Swing to help end users to do the mapping. To test the accuracy of the emissions calculators, the MEET emissions calculator's computational results are compared with vehicles manufacturing information on real road journeys of a one-day schedule and the MEET one is believed to perform equally well. Results show that the emissions calculators give a comparable amount of CO_2 emissions as provided by manufacturing information and fuel efficiency studies, under different scenarios. Furthermore, in the last section of this chapter, the discrepancy between CO_2 emissions minimisation and traditional travel time minimisation are demonstrated using numerical experiments with the NAEI emissions calculator. Numerical experiments illustrate the scenarios where the discrepancies between objectives exist, and hence it is reasonable to distinguish the problem of minimising emissions from the traditional travel time minimisation problem. The CO_2 calculator has been introduced to a field service provider on its real-life field scheduling system. The business applications including a web portal, web services, and a heat map visualisation are explained, which demonstrates the potential real-life impact of green logistics.

In the next chapter, the CO_2 emissions calculator proposed in this chapter will be applied to an incentive pricing problem to minimise CO_2 emissions of delivery vehicles.

4 Preliminary incentive pricing model to minimise CO_2 emissions of delivery vehicles

In this chapter, we consider a problem of scheduling services at customer sites by allocating them to engineers each with a vehicle. Customer orders arrive dynamically online or by a call, and each customer order comes with a preferred time window. A time window is a period of time within which the service must start, e.g., a two-hour time window such as 8:00am-10:00am. We propose a new approach to this problem which applies low-emissions vehicle scheduling techniques with dynamic pricing to reduce CO_2 emissions and maximise profit. When a customer requests for service with a preferred time window, the company will provide the customer with additional service time window options and their corresponding prices. Incentives are included in the prices to influence the customer's choice in order to reduce CO_2 emissions. To help the company in determining the incentives, our approach solves the problem in two phases. The first phase tries to schedule the task in each of the possible time windows with the objective of minimising CO_2 emissions and the second phase solves a dynamic pricing model to decide the actual time window for the service so that the expected profit can be maximised. The customer sites considered are all in a confined geographical area hence the travelling distances within the area are not significantly different but the time of travel makes a more significant difference in vehicle CO_2 emissions. Therefore, the first phase problem can be considered as a vehicle scheduling problem (VSP) with different times of travel between customers at different times of a day due to traffic condition. The VSP is to allocate the service tasks to the schedules of a fleet of vehicles initially located at a depot. Since traffic conditions have a significant influence on CO_2 emissions, the amount of emissions generated and travel time per trip is modelled as being time-dependent to capture congestion patterns. The second phase determines the prices for the time window options provided to the customer. The prices will include incentives to encourage the customer to choose options with low CO_2 emissions cost. As the cost of engineer wages is fixed, the only cost varying with delivery option is emissions cost or fuel cost. A cost value is assigned to each kg of CO_2 emissions. Environmental sustainability is a concern for all, hence the discounted delivery windows will be tagged as green/environmentally friendly windows showing the amount of emissions that can be saved by choosing such options, which may further influence customers to choose these windows. The problem considered here is common for services in which there are many competitors and customer retention is important. For example, this may apply to home delivery or pick-up services, as well as installation or maintenance services.

We develop a new time-dependent CO_2 emissions minimisation scheduling model and an improved incentive dynamic pricing model for the two phases, respectively. The proposed approach is tested through computational studies that simulate the situation of a service delivery company. Assumptions are made to simplify the model such as homogeneous distance and linear demand model, and those assumptions will be removed in the following chapters to provide a more realistic model.

The rest of the chapter is organised as follows. Section 4.1 describes the problem in more detail and presents a two-phase solution framework. Mathematical models for the two phases are formulated in Sections 4.2 and 4.3 respectively. The simulation results and sensitivity tests are shown in Section 4.4. Finally, concluding remarks are given in Section 4.5.

4.1 Problem description and solution approach

We study a problem where a company sends engineers to customer sites to provide maintenance and installation services in a confined area. The company divides each day into several time windows. Typically each customer makes a request for the service through a website or telephone. The customer's order normally comes with a preferred time window for the service to start. Instead of simply accepting the customer requested time window, the company can provide the customer with a number of service time window options and their corresponding prices. Incentives are included in the prices to influence the customer's choice to help the company ensure a schedule which reduces CO_2 emissions. To deliver the services, each engineer is driving a vehicle, typically a van with the necessary tools. Therefore, the assignment of tasks to the engineers is equivalent to assignment of tasks to the vehicles. The distances among customer sites and the depot are all not far and maybe considered the same, e.g. 5 km. The average travel speeds are fluctuating during the day because of the traffic conditions, so the problem can be considered as a vehicle scheduling problem (VSP) with time windows extended to capture different times of travel between customers at different times of a day, rather than traditional vehicle routing problem.

Traffic conditions and travel speeds have a significant influence on CO_2 emissions, and so the level and cost of emissions are time-dependent. Though the level of emissions is also related to other factors such as vehicle load and road characteristics, this research assumes homogenous vehicles and focuses on timing of travel to take into account the different congestion conditions. Based on the traffic pattern of the area, a day is divided into several time slots, e.g. one hour, such that travel speed as well as the emissions level within a time slot can be considered the same.

Most previous research assumes that all information is known at the time of planning and hence solves a static vehicle scheduling problem. In this study we deal with a more dynamic nature of demand, where at the start of the schedule planning horizon, the planner has an initial set of previously accepted jobs with their agreed time windows. Meanwhile, new customers will arrive in the system dynamically each with an original preferred time window for service start. At the time of each new customer arrival, the problem is to determine the incentives and hence the prices for the service for different time windows and, once the customer selects a service time window based on this price information, to update the schedule by including both the new task and the existing tasks not performed yet. The overall objective is set to maximise the expected profit which is defined as the standard service price minus the expected incentives and the emissions cost, aiming for reducing the amount of

CO_2 emissions as well as increasing profit.

We propose a new approach to this problem which applies both low-emissions vehicle scheduling techniques and dynamic pricing. An initial schedule of the vehicles is constructed based on information of the initial set of tasks. Whenever a new customer order arrives with an original time window preference, prices for the task to be performed in different time windows are determined and presented to the customer; the new task will be scheduled to start in the time window that the customer selects based on the price information. The actual start times and the vehicles for the unserved existing tasks may be changed in the schedule to minimise the emissions cost. However, their agreed time windows will remain satisfied in the updated schedule. Emissions cost is defined as the monetary value of the amount of emissions, which is the cost of its fuel, set to £1.371/kg in this study. The monetary value of emissions is not accurate and it could be more to include the value of damage to environment or damage to the company's goodwill. The number here is for demonstration purpose, and the methodology and results will not be affected if a different value is chosen.

Therefore, we need to make both pricing and scheduling decisions each time a new customer order arrives. These two decisions are interrelated and so an integrated model would be difficult to formulate and solve. We propose to make these two decisions in two phases as outlined below.

The first phase deals with the scheduling decisions. Although the scheduling decision needs the information about the customer selection of time window for the new task which in turn needs the pricing decisions, we know for sure that the new task must be in one of the time windows. Thus, we solve several possible scheduling problems each assuming the new task being allocated to a different time window. Those scheduling problems are formulated as mixed integer linear programming models. The solution of each problem provides a schedule minimising emissions cost with the new task scheduled to start in the corresponding time window.

The second phase makes pricing decisions using the emissions costs obtained in the first phase. Based on the new customer's original preference, the probabilities for the customer to choose each time window are estimated. The way of these probabilities changing with pricing changes is assumed to be known. The pricing problem in the second phase can then determine the incentives/prices for each time window so that the expected profit is maximised. Profit is calculated as the fixed charges of service minus incentives offered minus the emissions cost. Once the customer selects a time window based on the pricing results, the corresponding schedule obtained in the first phase can be used. The second phase problems are modelled as non-linear programming models with linear constraints and solved.

We will model the scheduling problem of the first phase and the pricing problem of the second phase in Sections 4.2 and 4.3, respectively. Note that the scheduling model can also be used for scheduling the initial set of customer jobs. Hence all the decisions can be made with these two models. The framework of the approach is illustrated in Figure 4.1. Within this framework the models are used to help the planner to respond to customer requests and schedule the tasks.

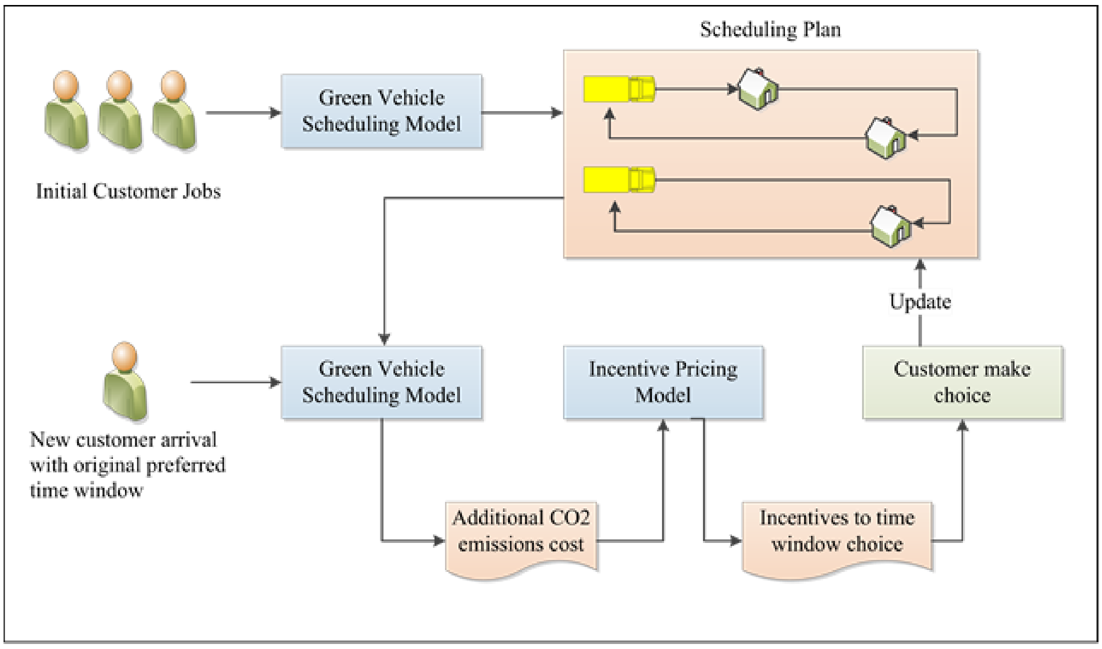

Scheduling Plan
Green Vehicle Scheduling Model
Initial Customer Jobs
Update
Green Vehicle Scheduling Model
Incentive Pricing Model
Customer make choice
New customer arrival with original preferred time window
Additional CO2 emissions cost
Incentives to time window choice

Figure 4.1: The framework of the solution approach.

4.2 First phase: The green vehicle scheduling model

The first phase model schedules or reschedules all the service tasks on hand to the vehicles with the objective to minimise the emissions cost. There are a fixed number of vehicles available in the depot at the start of the planning horizon; they will all be scheduled in the day. At the start of the planning horizon, the tasks considered in the model are the initial set of customer orders, while the tasks in each of the subsequent models include the new customer order and the unserved existing tasks. Each task has a required time window for service to start. The time window for the new task is the assumed possible time window. The service times of different tasks can be different to represent different duration of tasks.

The whole planning horizon is ranged from 0 to T, where T is the last time a new customer can be served, and where 0 indicates the start of a working end and T indicates the end of a working day correspondingly. Following Hickman et al. (1999), we divide the whole planning horizon into several equal time slots. Each time slot has an average travel speed. Based on the NAEI emissions model previously mentioned, CO_2 emissions could be computed for each time slot accordingly. Figure 4.2 presents a speed profile over the planning horizon for London, UK, generated based on travel speed and common congestion periods summarised from Road Congestion and Travel Times reports (Department for Transport, 2016). The amounts of emissions are computed based on this profile.

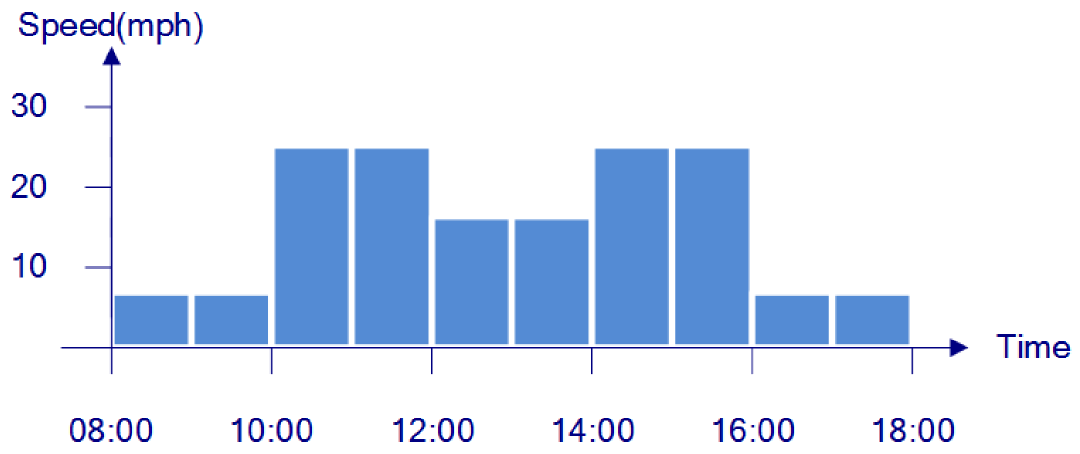


Figure 4.2: Speed profile over the planning horizon for London, UK.

Note that as time goes on, when a new customer order arrives, the planning horizon for the model will be shorter than the initial planning horizon. To keep the model general, we redefine the time so that the planning horizon for the model always starts at time 0 and ends at T. T will be smaller in later run of the model. Other time related parameters such as time windows and time slots will also be adjusted accordingly. At the time a new customer arrives, some existing tasks may have started but not completed. Such tasks have to be continued without change. Until the completion time of such a task, the vehicle assigned to this task will not be available.

Notations

Parameters:

i, k: index of customers;

j: index of vehicles;

N: total number of customers;

V: total number of vehicles available;

$M1$: a big positive number;

T: end time of the planning horizon;

n: total number of time slots in the planning horizon, the length of each slot is T/n;

b_i, e_i: beginning and ending times of the required time window in which a vehicle must arrive to customer i;

s_i: service time needed for customer i;

τ_t: travel time between any two sites;

a_j: the time when vehicle j becomes available;

c_t: emissions cost for a trip in time slot t.

Variables:

C_i: emissions cost of serving customer i;

u_i: time point a vehicle starts travelling to customer i;

w_i: time point the service at customer i starts;

$$x_{ij} = \begin{cases} 1, \text{ if customer } i \text{ is allocated to vehicle } j \\ 0, \text{ otherwise} \end{cases};$$

$$y_{ik} = \begin{cases} 1, \text{ if customer } i \text{ is served before customer } k \text{ by the same vehicle} \\ 0, \text{ otherwise} \end{cases};$$

$$z_{it} = \begin{cases} 1, \text{ if travelling to customer } i \text{ starts in time slot } t \\ 0, \text{ otherwise} \end{cases}.$$

When a new customer requests service, the scheduling problem needs to be re-solved.

This is formulated as follows.

Minimise $\sum_{i=1}^{N} C_i$

Subject to

$$C_i \geq c_t + (z_{it} - 1)T, i = 1, \ldots N; t = 1, \ldots, n \tag{4.1}$$

$$\sum_{j=1}^{V} x_{ij} = 1, i = 1, \ldots, N \tag{4.2}$$

$$\sum_{t=1}^{n} z_{it} = 1, i = 1, \ldots, N \tag{4.3}$$

$$y_{ik} + y_{ki} = 1, i, k = 1, \ldots N, i \neq k \tag{4.4}$$

$$b_i \leq w_i \leq e_i, i = 1, \ldots, N \tag{4.5}$$

$$w_i \geq u_i + \tau_t + (z_{it} - 1)T, i = 1, \ldots, N; t = 1, \ldots, n \tag{4.6}$$

$$w_i + s_i \leq u_k + M1(3 - x_{ij} - x_{kj} - y_{ik}), i, k = 1, \ldots, N; j = 1, \ldots, V \tag{4.7}$$

$$u_i \geq (t-1)(T/n)z_{it} + T(z_{it} - 1), i = 1, \ldots, N; t = 1, \ldots, n \tag{4.8}$$

$$u_i + \tau_t \leq t(T/n) + T(1 - z_{it}), i = 1, \ldots, N; t = 1, \ldots, n \tag{4.9}$$

$$x_{ij}, y_{ik}, z_{it} \in \{0, 1\}, i, k = 1, \ldots, N; t = 1, \ldots, n \tag{4.10}$$

$$w_i, u_i \geq 0, i = 1, \ldots, N \tag{4.11}$$

The objective is to minimise emissions cost. Constraints (4.1) define the emissions cost of customer i. Constraints (4.2) specify that each customer is served once and by only one vehicle. Constraints (4.3) ensure that travelling to each customer i starts in only one time slot. Constraints (4.4) indicate the sequence of serving customers, a sub-tour elimination constraint. Constraints (4.5) ensure that the time window requirements of each customer must be satisfied. Constraints (4.6) require that service start time must allow for travel time between customers. Constraints (4.7) will take effect when $x_{ij} = 1$, $x_{kj} = 1$, $y_{ik} = 1$, which means customers i and k are both served by vehicle j, and customer i is served before customer k. Under this circumstance, the constraints ensure that the vehicle can start travelling to customer k only after it completes serving customer i. Constraints (4.8) to (4.9) identify the start and end time slot indexes. These help us to link continuous variable u and binary variable z. Constraints (4.10) and (4.11) are binary and non-negativity constraints.

4.3 Second phase: The pricing model

When a new customer arrives with an initial preferred time window, the planner will compute a price menu for each available time window. Incentives are given to time windows which produce lower emissions; with the aim of shifting the customer's demand. The incentive pricing model is adopted from Campbell and Savelsbergh (2006), whose model uses a distance cost function compared to the emissions cost function used in the current study. The emissions cost of each time window is calculated from the output of the first phase green vehicle scheduling model. These are used in the pricing model to decide which windows should receive incentives and the amounts of incentives. We also aim to improve on their model in two ways. The amount of incentives they offer is highly restricted to the lowest initial probability of choosing one time window and the time windows which will be given incentive are pre-determined, refer to Campbell and Savelsbergh (2006) for more details. Our model amends their model by imbedding the selection of time windows into the optimisation, and in our model incentives are not restricted by the lowest initial probability. In line with Campbell and Savelsbergh (2006), we assume that the company has knowledge of the likelihood of a customer selecting a specified time window and the effect of price change on the customer's buying behaviour. With the advent of the internet, it is reasonable to assume that businesses have access to vast historical data about customers. For example, online grocery shopping and delivery companies have developed interfaces for customers to book a service and monitor their purchasing behaviour as well as their reactions to incentives or discounts.

Notations

Parameters:

h: index of time windows;

O: sets of all time windows;

p^h: probability of the current customer choosing time window h if no incentives are offered;

C^h: additional emissions cost of including the current customer into an optimised schedule (emissions cost is proportional to emissions);

r: revenue that the current customer will bring;

β: price sensitivity parameter;

B: maximum incentive that may be applied to time window h;

$M2$: a large positive number.

Variables:

I^h: incentive for time window h;

z: amount of probability reduced;

qu^h: the probability that time window h which gets an incentive will be chosen by customer;

qv^h: the probability that time window h which does not get an incentive will be chosen by customer;

$$\alpha^h = \begin{cases} 1, \text{ if } I^h \geq 0 \\ 0, \text{ if } I^h = 0 \end{cases};$$

$$X^h = \begin{cases} 1, \text{ if } p^h - z > 0 \\ 0, \text{ if } p^h - z \leq 0 \end{cases}.$$

The pricing model is formulated as follows:

Maximise $\sum_{h \in O}(r - C^k - I^h) \cdot qu^h + \sum_{h \in O}(r - C^h) \cdot qv^h$

Subject to

$$0 \leq I^h \leq B \cdot \alpha^h, \quad \forall h \in O \tag{4.12}$$

$$qu^h \leq p^h + \beta \cdot I^h, \quad \forall h \in O \tag{4.13}$$

$$qu^h \geq p^h + \beta \cdot I^h + (\alpha^h - 1), \quad \forall h \in O \tag{4.14}$$

$$qu^h \leq \alpha^h, \quad \forall h \in O \tag{4.15}$$

$$\alpha^h + qv^h \geq p^h - z, \quad \forall h \in O \tag{4.16}$$

$$qv^h \leq p^h - z + 1 - X^h + \alpha^h, \quad \forall h \in O \tag{4.17}$$

$$p^h - z \leq X^h + \alpha^h, \quad \forall h \in O \tag{4.18}$$

$$qv^h \leq X^h + \alpha^h, \quad \forall h \in O \tag{4.19}$$

$$qv^h \leq 1 - \alpha^h, \quad \forall h \in O \tag{4.20}$$

$$\sum_{h \in O} (qu^h + qv^h) = 1 \tag{4.21}$$

$$I^h \geq \frac{1}{M2} + M2 \cdot \alpha^h - M2, \quad \forall h \in O \tag{4.22}$$

$$\alpha^h \leq 2 \cdot M2 \cdot I^h, \quad \forall h \in O \tag{4.23}$$

$$z, qu^h, qv^h \geq 0, \quad \forall h \in O \tag{4.24}$$

$$X^h, \alpha^h \in \{0, 1\}, \quad \forall h \in O \tag{4.25}$$

The objective is to maximise expected profit. The first part of the objective function represents the expected profits from the time windows which are given incentives and the second part represents the expected profits from the windows which do not receive any incentive.

Constraints (4.12) restrict the incentive to be in range 0 to B. Constraints (4.13) - (4.15) calculate the adjusted probability $qu^h = p^h + \beta \cdot I^h$ for all h receiving incentives. If time window h does not receive an incentive, $qu^h = 0$. If a time window does not receive an incentive, the probability of the customer choosing it will reduce, this is represented by constraints (4.16) - (4.20). If the adjusted probability of a time window which does not receive an incentive is greater than zero, i.e. $p^h - z > 0$ then the adjusted probability, $qv^h = p^h - z$, otherwise, $qv^h = 0$. Constraints (4.21) ensure that the probabilities for all time windows sum to one. Constraints (4.22) and (4.23) indicate that if a time window receives an incentive, the amount of incentive is greater than zero. Constraints (4.24) and (4.25) are nonnegativity and binary constraints.

Remark 1 Offering no incentives is a feasible solution.

When $I^h = 0$ for all h, from constraints (4.22) we have $\alpha^h = 0$. Then from the first four constraints (4.12) to (4.15) we have $qu^h = 0$. Constraints (4.16) to (4.20) can be satisfied by $X^h = 1$ and $qv^h = p^h - z$. Then since based on constraints (4.21) $\sum_{h\in O}\left(qu^h + qv^h\right) = 1$, $\sum_{h\in O}\left(0 + p^h - z\right) = 1$. Since the initial probabilities satisfy $\sum_{h\in O} p^h = 1$, we know that $z = 0$. So the solution with no incentives satisfy all the constraints, i.e., feasible, and the corresponding objective value is $\sum_{h\in O}\left(r - C^h\right)\cdot p^h = r - \sum_{h\in O} C^h \cdot p^h$.

Remark 2 If an incentive is offered in the optimal solution, then the solution will reduce the expected carbon dioxide emissions compared to a model with no incentive.

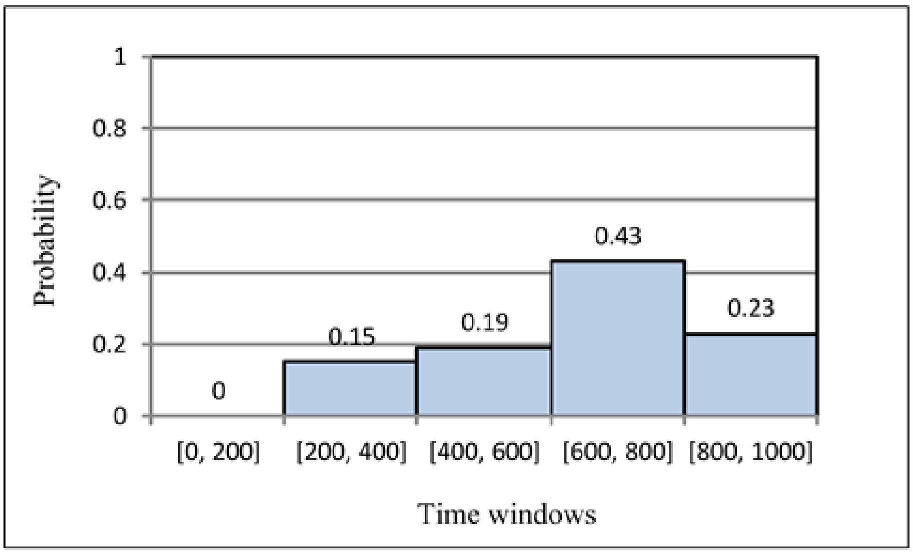
Because no incentive is a feasible solution, if $I^h > 0$ then by definition the expected revenue $\sum_{h\in O}\left(r - C^h - I^h\right)\cdot qu^h + \sum_{h\in O}\left(r - C^h\right)\cdot qv^h > r - \sum_{h\in O} C^h \cdot p^h$. The expected revenue can be re-written as $r - \sum_{h\in O} C^h\left(qu^h + qv^h\right) - \sum_{t\in O} I^h qu^h$. $\sum_{h\in O} I^h qu^h$ is an additional cost to the objective function then the expected carbon dioxide emissions cost $\sum_{h\in O} C^h\left(qu^h + qv^h\right)$ must be lower for an overall revenue increase.

4.4 Numerical experiments

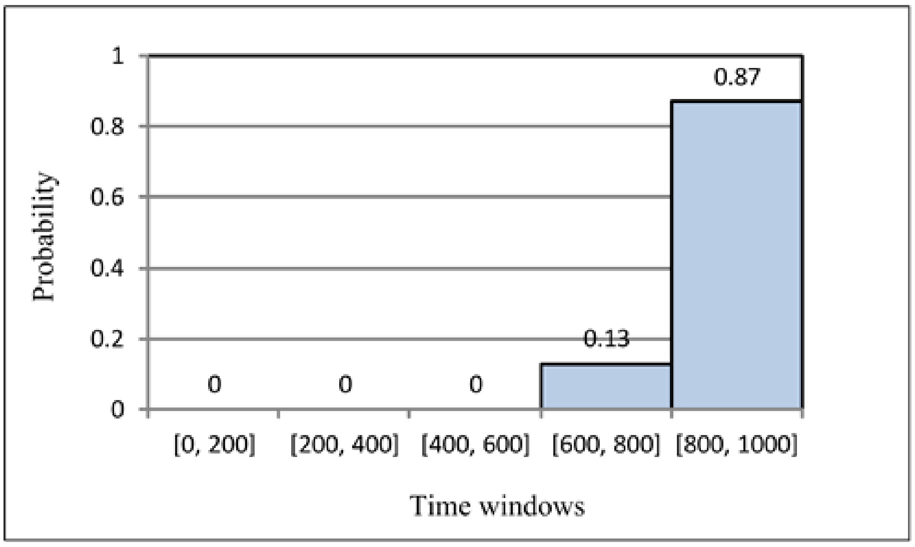
The green scheduling and pricing approach is tested using numerical experiments. The first phase model is solved using Xpress optimiser, which applies a powerful branch-and-bound method to solve mixed integer linear programing models. The second phase model is solved by Lingo optimiser which deploys both successive linear programming (SLP) and generalized reduced gradient (GRG) algorithms to solve non-linear programming models. In the experiments, the whole planning horizon is set to be one day from 8:00am to 6:00pm. For convenience we choose a time unit such that the length of this planning horizon is 1000. The whole period is divided into five equal time windows that customers may choose for their requested service to start. Reflecting the congestion conditions, the speed profile is expressed using ten time slots as illustrated in Figure 4.2. A time slot refers to a time interval associated with time-dependent travel speeds, while a time window is used to describe an option of time periods for customer booking. The solution approach described in Section 4.2 is implemented simulating customer arrivals and the service operations for the whole planning horizon. The models in the first and second phases are solved using Xpress Optimiser and Lingo, respectively. We generate an initial set of customers at the beginning of the day and simulate subsequent customer arrivals using Poisson process. For each customer arrival, a preferred time window is randomly generated. The probability for the customer accepting each available time window is calculated using a triangular distribution with the preferred time window as the mode. Details of the triangular distribution are provided in Appendix D. The green scheduling model is run to obtain the emissions cost for the task being performed in each available time window. The pricing model is then run to decide the incentives given for each of these time windows. Table 4.1 shows the inputs and outputs of the pricing model for an example customer. The customer arrives at time 284. His initial preference is time window [600,800]. The initial probabilities of choosing the available time windows and the emissions costs for the time windows are given in Table 4.1. The pricing model offers incentives for time window [800,1000] which has the lowest emissions cost. The probability profiles before and after incentives can be seen in Figure 4.3. As can be seen from the figures in Table 4.1 the expected emissions cost with the incentive is lower than the emissions cost for the customer initial preferred time window. The customer's final choice of the service time window is randomly generated using the adjusted probability profile. The task is then scheduled in the selected time window, and this customer becomes an existing customer.

	[0,200]	[200,400]	[400,600]	[600,800]	[800,1000]
Customer Arrival Time		284			
Available Time Windows	–	✓	✓	✓	✓
Initial Probabilities	–	0.133	0.267	0.4	0.2
Emissions Costs	–	50	35	35	20
Incentives	–	0	0	0	6.5
Adjusted Probabilities	–	0	0	0.15	0.85

Table 4.1: An example customer for illustration.



(a) Before incentive



(b) After incentive

Figure 4.3: Customer Choice Probability.

4.4.1 Performance of dynamic pricing

We first compare our approach with the green scheduling method without incentives. The green scheduling method is implemented using the same simulation framework. In this method the scheduling model in Section 4.3 is applied to schedule each customer's task always to their preferred time window. We assume there are five vehicles available at the depot and an initial set of 10 customers at the start of the planning horizon. We simulate the customer arrivals using a Poisson process with mean inter-arrival time of 50, and therefore on average there are approximately 20 new requests per day. The service time for each customer is different, which is randomly generated in the range of [55, 150] to mimic the uncertainties in the service times of this type of problems. We generate 10 set of customer arrival times, their initial preferred time windows and service times, each for one day. Each set of data is called a scenario. Because each customer's final choice is uncertain, the result may be different in different runs even for the same scenario. Therefore, we run each of these 10 scenarios 10 times and calculate the average emissions and profit. We assume each customer will bring a revenue of 30, which is about 10 times the amount of emissions cost, as we also need to cover other cost, e.g. engineer wages, to make profits. The profit is calculated by the revenue minus incentives and the emissions cost. We compare the two methods in terms of the CO_2 emissions cost and the profit. The average computational time is on average 5 minutes per customer even for this simplified model, so using software solver may not be applicable in real life. The results are shown in Table 4.2 and Table 4.3 respectively.

Scenarios	No. of new customers	Ratio of existing customers	Average emissions of 10 replications with incentives	Emissions with no incentive	CO_2 Savings (%)
1	20	0.33	75.81	78.88	3.8974
2	15	0.40	66.99	70.41	4.8570
3	20	0.33	94.77	117.1	19.0690
4	20	0.33	104.49	146.71	28.7749
5	15	0.40	60.39	62.58	3.4943
6	21	0.32	79.13	81.07	2.3890
7	20	0.33	83.92	112.14	25.1684
8	18	0.36	74.25	79.07	6.0915
9	17	0.37	96.62	104.79	7.7931
10	18	0.36	214.56	229.46	6.4951
Average		**0.35**			**10.8030**

Table 4.2: Comparing CO_2 emissions.

Scenarios	No. of new customers	Ratio of existing customers	Average profit of 10 replications with incentives	Profits with no incentive	Improvement in profit (%)
1	20	0.33	824.19	821.12	0.3744
2	15	0.40	683.01	679.59	0.5032
3	20	0.33	835.23	812.9	2.7469
4	20	0.33	855.51	813.29	5.1907
5	15	0.40	689.61	687.42	0.3181
6	21	0.32	850.87	848.93	0.2281
7	20	0.33	846.08	817.86	3.4509
8	18	0.36	765.75	760.93	0.6330
9	17	0.37	743.38	735.21	1.1108
10	18	0.36	775.44	760.54	1.9596
Average		**0.35**			**1.6516**

Table 4.3: Comparing profits.

The results presented in Table 4.2 and 4.3 clearly demonstrate the effects of incentives on both the emissions cost and the profit generated. Our incentive model reduces the CO_2 emissions cost by 10.80% on average, compared to a green vehicle scheduling method without incentive. Meanwhile the average profit increases by 1.65%. The average results show that the incentive pricing model can both increase profits and reduce carbon dioxide emissions significantly.

4.4.2 Sensitivity of the two phase approach

4.4.2.1 The ratio of existing customers

The average percentage of existing customers was about 33% in the initial setting (10 from a total of 30 on average). We change the ratio of existing customers in the system but use the same setup for other parameters as the first experiment. We increase the average percentage of existing customers to about 50% (15 out of 30) and then about 67% (20 out of 30). The number of new customers arriving will now be 15 and 10 respectively. We examine how the number of existing customers in the system affects the emissions savings and profit. The emissions cost and profit results are shown in Tables 4.4 and 4.5, respectively. For the ratio of 53% and 67% existing customers, the average savings for emissions cost become 5.77% and 3.37% respectively and the average profits become 0.82% and 0.36% respectively. These results demonstrate that as the percentage of existing customers increase, there are fewer new customers arriving during the day and so the system becomes less dynamic. The use of the pricing techniques is restricted. Therefore there is less room for the savings in CO_2 emissions cost and for profit improvement.

Scenarios	No. of New Customers	Ratio of existing customers	Emissions cost savings (%)	Improvement in profit (%)
1	15	0.50	7.8613	0.7895
2	14	0.52	0.0000	0.0000
3	18	0.45	1.7345	0.1535
4	9	0.63	8.7048	2.5403
5	13	0.54	0.4348	0.0367
6	14	0.52	22.6454	2.9654
7	10	0.60	0.1767	0.0155
8	15	0.50	12.2448	1.2925
9	17	0.47	3.8942	0.3635
10	9	0.63	0.0000	0.0000
Average		**0.53**	**5.7696**	**0.8157**

Table 4.4: Comparing emissions and profit with around 50% existing customers.

Scenarios	No. of New Customers	Ratio of existing customers	Emissions cost savings (%)	Improvement in profits (%)
1	9	0.69	2.3998	0.2126
2	12	0.63	2.7225	0.2742
3	8	0.71	2.9319	0.2733
4	6	0.77	4.7424	0.4323
5	8	0.71	3.1670	0.2876
6	16	0.56	3.9259	0.6653
7	11	0.65	3.4454	0.4299
8	11	0.65	7.0376	0.7564
9	11	0.65	2.8539	0.2645
10	8	0.71	0.4831	0.0405
Average		**0.67**	**3.3709**	**0.3637**

Table 4.5: Comparing emissions and profit with around 67% existing customers.

4.4.2.2 Variation of emissions level over congested/non-congested period

We test the performance of our pricing model in a situation where there is less variation in emissions cost for different time slots. For example, there is less congestions and the travel speed varies less during the day like in Yorkshire and the Humber, UK (Department of Transport, 2016). The less variation speed profile of a day of Yorkshire and the Humber, UK is obtained as shown in Figure 4.4.

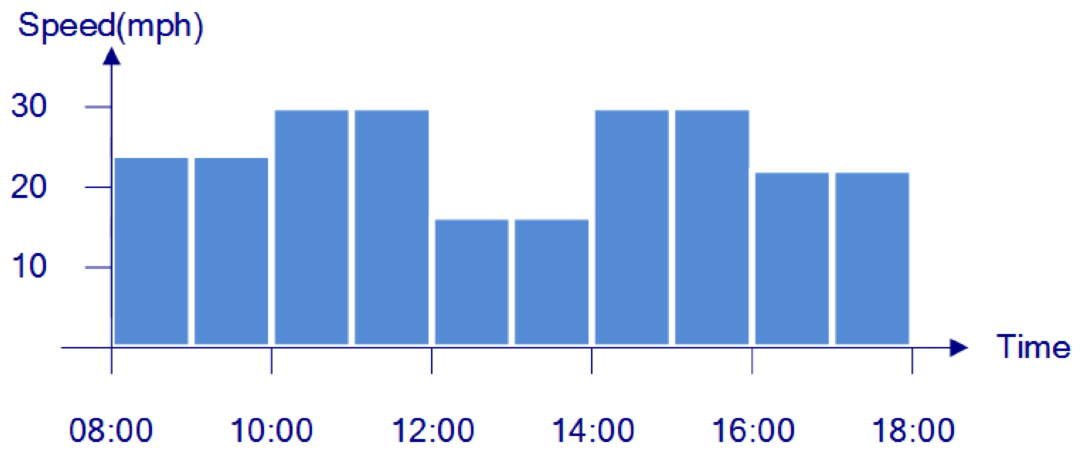


Figure 4.4: Speed profile with lower variation for Yorkshire and the Humber, UK.

Assuming this new speed profile, we test our model using the same 10 scenarios as in 4.2 and 4.3, and each scenario is run for 10 times. The results of emissions cost and profit are shown in Table 4.6. With the new speed profile our model only saves 0.23% on CO_2 costs. The profit improvement also becomes 0.02% only. As the emissions levels vary less across the day, the differences between the costs of different time slots are much smaller. Then the attempt to use incentives to shift customers' demands makes less difference. Hence the speed profile plays a central role to the effectiveness of the pricing model.

Scenarios	Ratio of existing customers	Average emissions of 10 iterations with incentives	Emissions with no incentive	Emissions cost savings (%)	Improvement in profit (%)
1	0.33	65.10	65.23	0.2054	0.0161
2	0.40	54.17	54.22	0.0861	0.0067
3	0.33	67.44	67.87	0.6385	0.0503
4	0.33	69.42	69.61	0.2750	0.0215
5	0.40	53.73	53.74	0.0140	0.0011
6	0.32	66.93	67.18	0.3788	0.0295
7	0.33	67.18	67.26	0.1140	0.0089
8	0.36	60.65	60.86	0.3385	0.0264
9	0.37	60.76	60.81	0.0822	0.0064
10	0.36	71.92	72.04	0.1596	0.0125
Average	**0.35**			**0.2292**	**0.0179**

Table 4.6: Results for the speed profile with lower variation.

4.4.2.3 Patterns of customer arrivals

We further test the effect of customer arrival patterns on the performance of our model. We run 10 scenarios that have the same existing customers as the first 10 scenarios in the first experiment, but subsequent customers arrive in three different patterns. In each pattern, the total number of customers is the same, which is 30. It means the number of new arriving customers is 20 for each pattern. For Pattern 1, the average numbers of customers arrived in the first and second halves of the planning horizon are 10 and 10 respectively. For Pattern 2, the average numbers of customers arrived in the first and second halves of the planning horizon are 5 and 15 respectively, and for Pattern 3, they are 15 and 5 respectively. Compared to Pattern 1 where customers arrive more evenly across the planning horizon, Pattern 3 provides the planner more opportunity to price dynamically and influence the customer's choice while Pattern 2 provides less opportunity. The savings on the emissions cost and the profit improvement for the three arrival patterns are shown in Table 4.7.

Patterns	Arriving customers [0,500]	Arriving customers [500,1000]	Emissions cost savings (%)	Improvement in profits (%)
1	10	10	13.6612	2.5405
2	5	15	5.2594	1.7866
3	15	5	17.2565	3.0734

Table 4.7: Results for different patterns of customer arrival.

The experimental results show that when more customers arrive in the system early, more savings in CO_2 emissions cost and more improvement in profit are expected. Over time we would expect strategic customers to learn that earlier arrival into the system results in them receiving the service for a lower price. This may shift the customer arrival patterns over time to the more profitable one.

4.5 Summary

In this chapter, we have studied a problem where a company sends engineers with vehicles to customer sites to provide services. Throughout the day customers call or visit a website requesting for services with preferred time windows and for each customer the company needs to schedule the service task to the vehicles. We proposed a new two-phase approach to this problem. The first phase solves vehicle scheduling model with the objective of minimising CO_2 emissions and the second phase solves a dynamic pricing model to maximise profit. The approach was tested through computational experiments. The results showed a significant reduction in the amount of CO_2 emissions as well as significant improvement in the total profits. We also carried out sensitivity tests, of which the results showed that with less dynamics in the system, i.e., less new coming customers, or lower emissions variation, the reduction in CO_2 emissions and profit improvement would be lower. The patterns of customer arrival also affect the results. Although our models were developed under certain assumptions, they are applicable for some real-life situations. The new approach is an attempt in the new direction of research to exploit the combination of green VRP/VSP and dynamic pricing techniques. In this chapter, we assumed that the level of CO_2 emissions depends only on different time slots of a given day, and used exact optimisation methods to solve our mathematical models. The emphasis is on verifying the overall approach especially the benefits of incentive pricing.

Under this framework, we will extend the study to consider the actual travel distance in the schedule. The first phase problem then becomes a time-dependent green vehicle routing problem which is a relatively new research topic. Like for most VRP problems, it is difficult to solve the green VRP optimally, especially for the problem we are facing which requires a quick response to the customer. In the next chapter, the development of heuristic solution algorithms will be discussed to reduce computation time.

5 Solution algorithms

Vehicle routing problems were proved to be NP-hard (Lenstra and Rinnooy-Kan, 1981). Therefore, it is unlikely to solve them optimally in polynomial time. Exact solution methods consume a large amount of computational efforts and so the size of the problem they can solve is restricted. For a real-life online service booking problem such as that we study, where the problem size is large and a response needs to be given to a customer in a short time, it is more appropriate to use a heuristic or metaheuristic method, which generates an acceptable solution in a short time. A survey paper on vehicle routing problems has found that metaheuristics (71.25% of papers reviewed) were used more often than exact methods and classical heuristics, furthermore simulation and real-time solution methods were rarely applied (Braekers et al., 2016). Many metaheuristics have been applied in literature to solve vehicle routing problems and green vehicle routing problems. The state of the art metaheuristics used in green vehicle routing problems were reviewed by Lin et al. (2014), including tabu search, simulated annealing, variable neighbourhood search, genetic algorithm and ant colony optimisation.

In this chapter, some different metaheuristic methods are implemented and tested for the green time-dependent vehicle routing problems with time windows, which is the first phase problem in the thesis. They are neighbourhood based metaheuristics including tabu search, two variants of variable neighbourhood search (VNS), simulated annealing and a new self-adaptive simulated annealing algorithm. We exclude the use of the genetic algorithm and other evolutionary algorithms because the time window constraints make it hard to maintain solution feasibility when performing the mutation or crossover operation in these algorithms. It would be time-consuming to revise infeasible solutions. This will not be suitable for the situation where quick response is needed. On the other hand, time window constraints can help to reduce the neighbourhood size for the neighbourhood based algorithms.

The rest of chapter is organised as follows. In Section 5.1 the problem is described. Section 5.2 presents the method for constructing initial solutions for all the implemented solution algorithms. Section 5.3 shows different ways of generating neighbourhood solutions. VNS and its variants and tabu search are introduced in Section 5.4 and Section 5.5 respectively. Section 5.6 shows the real-life test cases applied for comparing those algorithms, and the comparison results are illustrated in Section 5.7. Section 5.8 introduces the classic SA and proposed a new self-adaptive SA, together with the comparisons between them. Summary of this chapter is in Section 5.9.

5.1 Problem description

The green time-dependent vehicle routing problem with time windows studied here for service delivery is new and has some different features compared to the traditional time-dependent vehicle routing problem. As with traditional models, the travel speed is time-dependent, time window constraints are considered. The new features are 1) the objective is CO_2 emissions minimisation; 2) heterogeneous fleet is used; 3) skill matching constraints. The vehicle scheduling and routing problem of this study involves real-life tasks locations in the UK. $V = v_0, v_1, \ldots, v_n$ is a set of nodes representing task locations and v_0 represents the depot, which is the location where engineers start and end their daily work journeys. $A = \{(v_i, v_j)|v_i, v_j \in V, i \neq j\}$ is the arc set in the road network. We assume engineers will always take the shortest path between two nodes, so there is only one path linking the same pair of nodes. Each arch of A is associated with a non-negative cost C_{ij}, which represents the cost of travelling from v_i to v_j. In this study, this cost is the amount of CO_2 emissions computed using emissions models mentioned previously. The travel speed v_t is a time-dependent speed, derived from our speed profile. Engineers may drive different vehicles which have different emissions formulas. Each task has a specified skill requirement and each engineer has a set of skills that determine the types of tasks he/she can perform. A task can only be scheduled to an engineer with the specified skill. Each task has a time window requirement $[b_i, e_i]$, where b_i is the earliest possible time to start this task and e_i is the latest time. The service start time must be in this range. Each vehicle/engineer has a working period, all travels and works can only be scheduled within this working period. In summary, the problem is a time-dependent green vehicle routing problem with a heterogeneous fleet, and with the time window and skill matching constraints considered. The ways of estimating CO_2 emissions were described in Chapter 3 including road distance estimations, tailored speed profiles and vehicle specification mapping to emissions formulas.

5.2 Initial solution

Initial feasible solutions are constructed using an insertion heuristic as in Solomon (1987). This algorithm generates a good quality initial feasible solution, which is important to the success of tabu search algorithm. Reduced VNS (RVNS) algorithm is more robust to the quality of the initial solution but also requires it to be feasible. Given a list of engineers who are working for the period and a list of tasks to be scheduled, a traditional insertion heuristic will randomly choose a task from the task list and insert into the best and feasible partial route. In vehicle routing problems with time windows and skills matching constraints, this way will lead to problems as it may first insert some tasks that are less difficult to accommodate and leave the route unable to accommodate those difficult tasks to be added later. To overcome this, an extended insertion heuristic is introduced here. First, all tasks are ranked in ascending order of their easiness to insert. The easiness is measured by task time window width and task requested skill's rareness. We want to schedule the most difficult task first (or the easiest task last). The insertion algorithm is shown in Algorithm 5.1. In step 2,

$$Easiness_i = \alpha \cdot Time\,Window\,Width(i) + \beta \cdot Task\,Rareness(i),$$

and

$$Task\,Rareness(i) = Number\,of\,Eng\,has\,the\,skill/Times\,the\,skill\,are\,requested.$$

The smaller the $Easiness_i$, the harder to accommodate the task. In this study,

α and β are tuned to be 0.01 and 300 for good performance. The cost in step 6 is the additional amount of CO_2 emissions when inserting this task to the current partial route.

Algorithm 5.1 Insertion algorithm

```
1: **Function** InsertionAlgorithm;
2: **Rank** tasks in descending order of their difficulty (i.e., ascending order of the $Easiness_i$)
3: **Get** ordered taskList
4: $i \leftarrow 1$
5: **Repeat**
6:     insert($task_i$) to the lowest cost and feasible position in current partial route
7:     $i \leftarrow i + 1$
8: **Until** $i$ = taskList length
```

5.3 Neighbourhood operators

In this research, six different operators for generating neighbouring solutions are used to enable a broader exploration of the solution space as shown in Figure 5.1. Those operators are a mix of nodes, links and chains changes. For neighbour generating operators in general, one could refer to Bräysy and Gendreau (2005). The details of the six operators are shown below:

- 2-Opt: this is applied to a single route by breaking two links and reconnect. For the example in Figure 5.1, before changes the route is A-B-C-D-E-F-G-H and after changes the route is A-B-F-E-D-C-G-H.

- Swap: this is applied to a single route by swapping the position of two nodes. For the example in Figure 5.1, before changes the route is A-B-C-D-E and after changes the route is A-D-C-B-E.

- Exchange: this is applied to a pair of routes by swapping the positions of two nodes each from one route. For the example in Figure 5.1, before changes the route is A-B-C and D-E-F and after changes the route is A-B-E and D-C-F.

- Relocation: this is applied to a pair of routes by taking out one node from one route and insert it into another. For the example in Figure 5.1, before changes the route is A-B-C-D and E-F-G-H and after changes the route is A-B-D and E-F-C-G-H.

- 2-Opt*: this is applied to a pair of routes by breaking one link from each route and reconnect two routes. For the example in Figure 5.1, before changes the route is A-B-C-D and E-F-G-H and after changes the route is A-B-D and E-F-C-G-H.
- Or-Opt: this is applied to a single route by moving a chain of k nodes to a new position, e.g. in the example in Figure 5.1, chain D-E is relocated. Before changes the route is A-B-C-D-E-F and after changes the route is A-E-D-B-C-F.

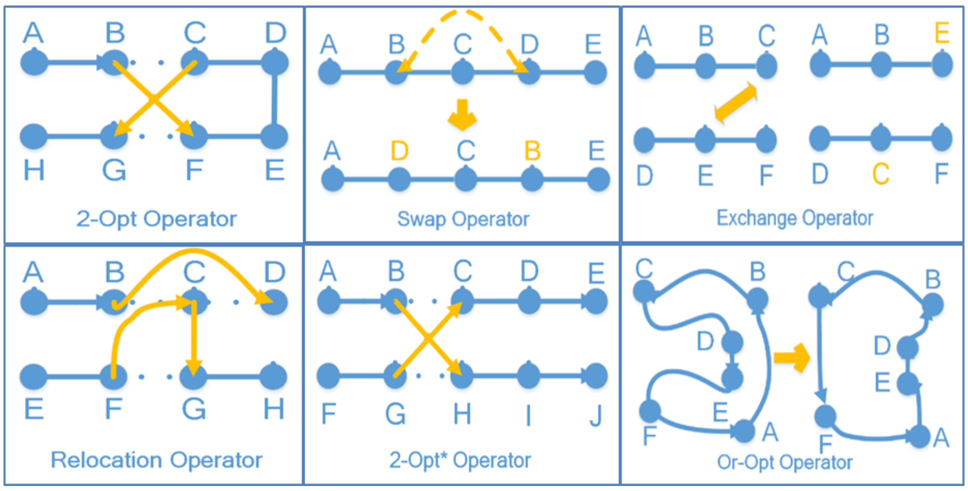


Figure 5.1: Neighbourhood operators.

5.4 Variable neighbourhood search

Variable neighbourhood search is a metaheuristic algorithm based on systematic changes of neighbourhoods both in the descent phase to find a local minimum, and in the perturbation phase to emerge from the corresponding valley. Since it was introduced, this method has been developed rapidly and found success in combinatorial optimisation problems. VNS's idea is based on the fact that a local minimum found in one neighbourhood structure is not necessarily the local minimum for other neighbourhood structures; A global minimum is a local minimum in the combination of all possible neighbourhood structures; Local minima found by different neighbourhoods are usually close to each other.

5.4.1 Variable neighbourhood descent (VND)

There are several variants of VNS, and in this research, variable neighbourhood descent and reduced variable neighbourhood search will be studied. The procedure of variable neighbourhood descent is outlined in Algorithm 5.2. The algorithm terminates when no improvement can be achieved. During the local search, a solution is generated using the current neighbourhood operator k, and once the best improvement is found, it will be accepted (best improvement strategy) and the current neighbourhood operator will keep generating new solutions until no improvement can be found with the current neighbourhood operator. In steps 5 to 8 is local search using one neighbourhood. In step 7, when the best neighbour x' is better than the current solution x, meaning that x' has a lower CO_2 emissions level than x, then the current solution will be updated. When no improvement could be achieved with this current neighbourhood operator, we move to the next neighbourhood operator $k + 1$ in step 9. This searching process will terminate until no improvement is found using all six neighbourhood operators.

Algorithm 5.2 Variable neighbourhood descent

```
 1: **Function** VND(x);
 2: **Repeat**
 3:     $k \leftarrow 1$
 4:     **Repeat**
 5:         **Repeat**
 6:             $x^{'} \leftarrow BestImprovement(x, k)$
 7:             **If** $x^{'}$ is better than $x$, then $x \leftarrow x^{'}$
 8:         **Until** $x = x^{'}$
 9:         $k \leftarrow k + 1$
10:     **Until** $k = k_{max}$
11: **Until** $no\ improvement\ could\ be\ found$
```

5.4.2 Reduced variable neighbourhood search (RVNS)

The problems with descent algorithms are that it can get trapped to the local optimal and cannot escape from it even though we use variable neighbourhood operators. RVNS tries to avoid this by introducing a shake step. Algorithm 5.3 outlines the procedure of RVNS. The stopping criterion in this research is chosen to be a maximum runtime. In step 5, $Shake(x, k)$ means we randomly generate a solution x' from the kth neighbourhood of x. Step 6 is the neighbourhood search using the current neighbour generating operator, representing the same process as steps 5 to 8 in VND.

Algorithm 5.3 Reduced variable neighbourhood search

```
1: **Function** RVNS(x);
2: **Repeat**
3:     $k \leftarrow 1$, $CPU\ time \leftarrow 0$
4:     **Repeat**
5:         $x^{'} \leftarrow Shake(x, k)$
6:         $x \leftarrow NeighbourhoodSearch(x, x^{'}, k)$
7:         $k \leftarrow k + 1$
8:     **Until** $k = k_{max}$
9: **Until** $CPU\ time > time\ max$
```

5.5 Tabu search

Tabu search is a metaheuristic search method employing local search methods and has been widely used in vehicle routing problems with time window constraints. Different from descent methods, the objective is allowed to deteriorate in order to avoid local minima. To prevent cycling, solutions that are recently visited are prohibited and stored in a tabu list. Algorithm 5.4 illustrates the procedure of TS. The stopping criterion is usually either when it reaches the maximum total number of iterations $iter\_max$ or the best solution has not been updated for a certain number ($iter\_cons\_max$) of iterations. Steps 6 to 8 in Algorithm 5.4 perform local search methods within a specified neighbourhood. This neighbourhood is generated at each iteration by a neighbourhood operator which is randomly selected from the previously mentioned six operators. Then the solution that has been recently visited (memorised in tabu list) or violates constraints will be eliminated. The reason for only keeping feasible solutions is that once an infeasible solution is accepted, it is hard to regain feasibility using tabu search algorithm. Then in step 7, the local search takes the best improvement strategy, which could be switched to the first improvement strategy. The structure of memory is a short-term memory with a fixed size of six, which means the six most recent solutions will be in the tabu list. $f(x)$ is the fitness score of the solution x, and in this research, it is the total CO_2 emissions of the solution routes.

Algorithm 5.4 Tabu Search

```
1: Function TabuSearch;
2:     $iter \leftarrow 1$, $iter\_cons \leftarrow 1$; $x_{best} \leftarrow x_{initial}$, $x_{current} \leftarrow x_{initial}$
3: Repeat
4:     $k = random(k = 1 \dots 6)$
5:     Generate neighbourhood by $k$th operator: $neighbour_k(x_{current})$
6:     Remove tabu or infeasible solution in $neighbour_k(x_{current})$
7:     $x_{current} \leftarrow$ the best solution of $neighbour_k(x_{current})$
8:     Update tabu list by adding $x_{current}$, removing the least recent one
9:     If  $f(x_{current}) < f(x_{best})$, $iter\_cons \leftarrow 1$
10:    Else $iter\_cons \leftarrow iter\_cons + 1$, $iter \leftarrow iter + 1$
11: Until $iter = iter\_max$  or  $iter\_cons = iter\_cons\_max$
```

5.6 Real-life test cases

Five real-life data samples are taken from a field service provider with similar problem size (around 100 tasks per sample). Table 5.1 shows the exact number of tasks in each case. The initial feasible solution for each case is generated using the extended insertion algorithm proposed in Section 5.2. The objective values of the initial solutions in terms of the amount of CO_2 emissions in kg is also shown in Table 5.1 together with two other performance measures of the solutions: the number of vehicles or engineers scheduled to fulfil all the tasks and the number of trips that are bundled together which means tasks at the same location. Our different test cases have different features. For example, in Figure 5.2, comparing test case 1 and test case 3, test case 3 has a higher number of widely spread tasks on the map, that is why its initial solution has higher CO_2 emissions.

Test case	No. of Task	No. Engineer	CO_2 kg	Bundle Trip
Case 1	103	19	187.613	46
Case 2	110	18	150.044	51
Case 3	108	20	236.522	46
Case 4	100	17	201.032	42
Case 5	106	24	233.129	41

Table 5.1: Initial solution.

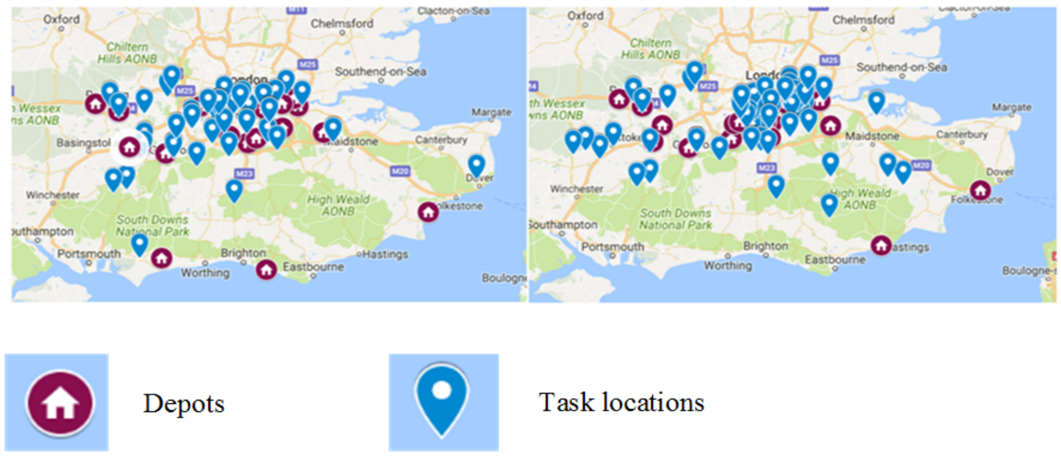


Figure 5.2: Test case 1 (left) vs. test case 3 (right).

5.7 Comparison of metaheuristics on real-life test cases

5.7.1 Variable neighbourhood descent vs. reduced variable neighbourhood search

For each test case, we first use VND to solve the problem and get a solution together with its objective value as well the CPU time used, then we let RVNS run for the same CPU time and compare the results. As RVNS has a stochastic nature, this algorithm is run 30 times to conduct statistical analysis. Based on the procedure of Chen et al. (2016), a one-sample t-test is carried out to see if the mean solution of RVNS is significantly different from the VND, and the significance level is set to be 0.1. The results are shown in Table 5.2 with the best solution highlighted in bold. Other than test case 2, RVNS yields significantly better solutions than VND. Comparing the emissions results, on average RVNS gives 3.7% less emissions with the same computational time. Results suggest that RVNS outperforms VND.

Algorithm	Emissions kg	Case 1	Case 2	Case 3	Case 4	Case 5
RVNS	Best	**148.95**	**143.32**	**158.90**	**130.61**	**161.95**
	Average	160.56	146.74	181.00	165.97	183.73
	S.D	6.28	2.11	14.17	15.14	10.15
VND		171.87	144.34	187.34	170.81	198.52
CPU time (seconds)		100	100	150	300	600
p-value		0.000	0.000	0.020	0.091	0.000
Different?		Y	Y	Y	Y	Y

Table 5.2: Comparison between VND and RVNS.

5.7.2 Reduced variable neighbourhood search versus tabu search

The tabu search algorithm randomly generates neighbourhood solutions, so this algorithm is also run 30 times for each test case. The stopping criterion is set to be 15000 maximum evaluations or 300 non-improving evaluations, and then the computational time is fluctuating, unlike RVNS which is set to be terminated at a specific maximum run time. These ways of setting termination criteria are based on the standard practice applied in literature. We also let both algorithms run for the same time for a fairer comparison. Another five test cases are generated and TS and RVNS are run for the same time to solve those test cases (Appendix E). Conclusions drawn from this new test are similar to those in Table 5.3. Table 5.3 shows the comparisons of these two solution algorithms by means of average CO_2 emissions and average computational time. The best solution value of CO_2 emissions is highlighted in bold. Among the five test cases, these two methods generate similar solutions for case 3 and case 5 by means of CO_2 emissions, but the computational time of tabu search algorithm is on average 5.37 times longer than RVNS. For the other test cases, tabu search algorithm gives significantly better results than RVNS, the CO_2 emissions are on average 5.35% less, but at the expense of 11.72 times of computational time. Although the tabu search algorithm finds the best solution in most cases, results suggest that RVNS has a good trade-off between solution quality and computational time for industrial application.

	Emissions	Case 1	Case 2	Case 3	Case 4	Case 5
RVNS	Best	148.95	143.32	158.90	**130.61**	**161.95**
	Average	160.56	146.74	181.00	165.97	183.73
	S.D	6.28	2.11	14.17	15.14	10.15
	CPU Time	100	100	150	300	600
Tabu	Best	**142.52**	**131.57**	**156.31**	140.79	163.24
	Average	156.44	137.83	178.82	153.66	188.73
	S.D	9.00	2.74	7.01	6.15	10.19
	Average Time(s)	1642.14	1616.32	1492.69	1668.29	1665.90
	S.D.	521.45	503.35	288.39	740.51	457.93
p-value		0.044	0.000	0.454	0.000	0.105
Different?		Y	Y	N	Y	N

Table 5.3: Comparison between TS and RVNS.

The vehicle routing problem is solved using three metaheuristics: RVNS, VND and TS. Experiments are carried out to compare three different metaheuristics on five typical real-life cases. On average RVNS gives 3.7% less emissions with the same computational time compared to VND, and therefore VND will not be included in the final solution engines. For most test cases, the tabu search algorithm gives CO_2 related results significantly better than RVNS: the CO_2 emissions are on average 5.35% less, but at an expense of 11.72 times of computational time. Nevertheless, the significantly lower running time cost of RVNS places this algorithm at the top position of industrial operational acceptable algorithm underlying a real-time decision-making support solution. The industrial solution relies on fast computations returning a set of time-contextual recommendations about the CO_2 impact and feasible time windows regarding a task-engineer assignment decision. Therefore, reduced variable neighbourhood algorithm is selected as the solution engine for the next chapter, and the performance of VNS is improved by parallelism.

5.8 Simulated annealing (SA) and self-adaptive simulated annealing (ASA)

Simulated annealing is another well-implemented algorithm for VRPTW. It is a stochastic algorithm consisting of two random processes: one for generating new solutions and the other for accepting or rejecting the new solutions. The probability of accepting a new solution or in the thermodynamics of metal annealing, the probability of state changes is determined by the Boltzmann distribution of the energy difference between two states:

$$P = e^{-\frac{\triangle E}{T}}, \quad \forall \triangle E > 0 \tag{5.1}$$

In SA, the simulation process mimics the temperature cooling process. When the system temperature is high, an increase in state energy will be accepted with a higher chance and the system performs a coarse search of the searching space to find a good minimum. As the system temperature goes down, it performs a fine search, worse solution is less likely to be accepted and it tries to find a better minimum. The probability distribution of accepting worse solution when temperature decreases is shown in Figure 5.3.

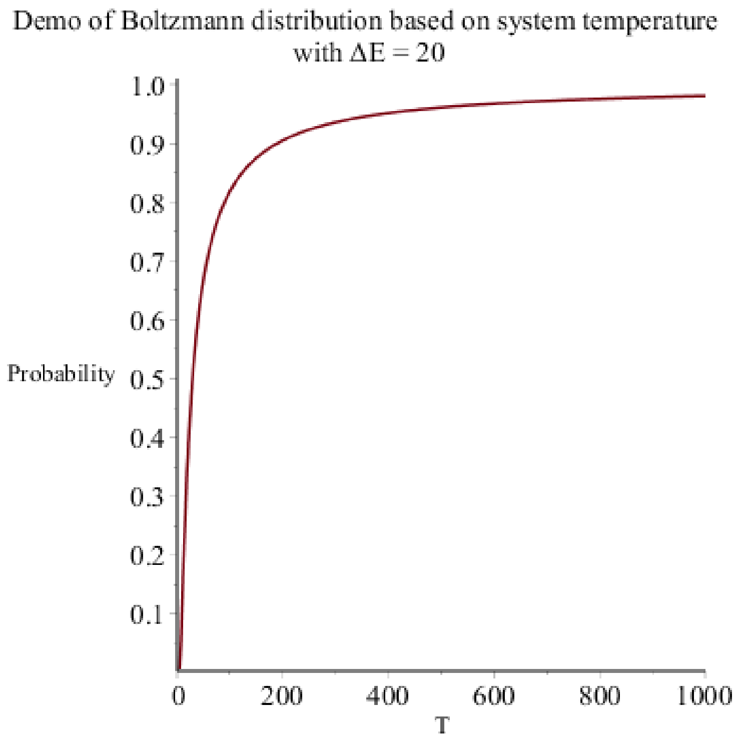


Figure 5.3: Boltzmann distribution based on system temperature when $\triangle E > 0$.

The pseudo codes of SA with large neighbourhoods are shown in Algorithm 5.5 and Algorithm 5.6. The neighbourhoods used are the same as in Figure 5.1, and the function *Shake* in Algorithm 5.6 is the same as in Algorithm 5.3.

Algorithm 5.5 Simulated Annealing with large neighbourhoods Algorithm

```
1:  **Function** SA($x$);
2:  Setting the system configurations
3:  $T \leftarrow T_{init}$
4:  **Repeat**
5:      $n \leftarrow 0$
6:      **Repeat**
7:          $x^{'} \leftarrow BestShake(x)$
8:          $\triangle E \leftarrow f(x^{'}) - f(x)$
9:          **IF** $\triangle E < 0, x \leftarrow x^{'}$
10:         **Else**, $x \leftarrow x^{'}$ with probability $P = e^{-\frac{\triangle E}{T}}$
11:         $n \leftarrow n + 1$
12:     **Until** $n = N(t)$
13:     Update($T$)
14: **Until** $T < T\_min$
```

Algorithm 5.6 Best Shake Algorithm in SA

```
1: **Function** BestShake($x$);
2: $k \leftarrow 0, best \leftarrow big\ number$
3: **Repeat**
4:     $x^{'} \leftarrow Shake(x)$
5:     **IF** $f(x^{'}) < best, best = f(x^{'})\ and\ x \leftarrow x^{'}$
6:     $k \leftarrow k + 1$
7: **Until** $k = k\_max$
```

The configurations of the system include system initial temperature, which is a big number, the termination temperature, which is small enough, the cooling strategy which is in step 13, a cooling rate for the corresponding cooling strategy and $N(t)$ the number of iterations to carry out at each temperature T. Romeo and Sangiovanni-Vincentelli (1991) show that an effective cooling strategy is essential to reducing the amount of time required to get the optimal solution. There are two commonly used cooling strategies: one is $T_{i+1} = \alpha T_i$ (Kirkpatrick et al.,

1983) or $T_{i+1} = T_i/(1 + \beta T_i)$ (Lundy and Mees, 1986). The latter one works when $N(t) = 1, \forall t$ means temperature updates every iteration. The graphic demonstrations of the above two cooling strategies are shown in Figures 5.4 and 5.5. For the Kirkpatrick et al. (1983) cooling strategy, when α is small, the system is cooling too fast and terminate for a short time, a good α is reported to be in the range of [0.85, 0.96]. As to the Lundy and Mees (1986) cooling strategy, when β is a big value, the cooling process is too fast and not enough steps of searching are operated.

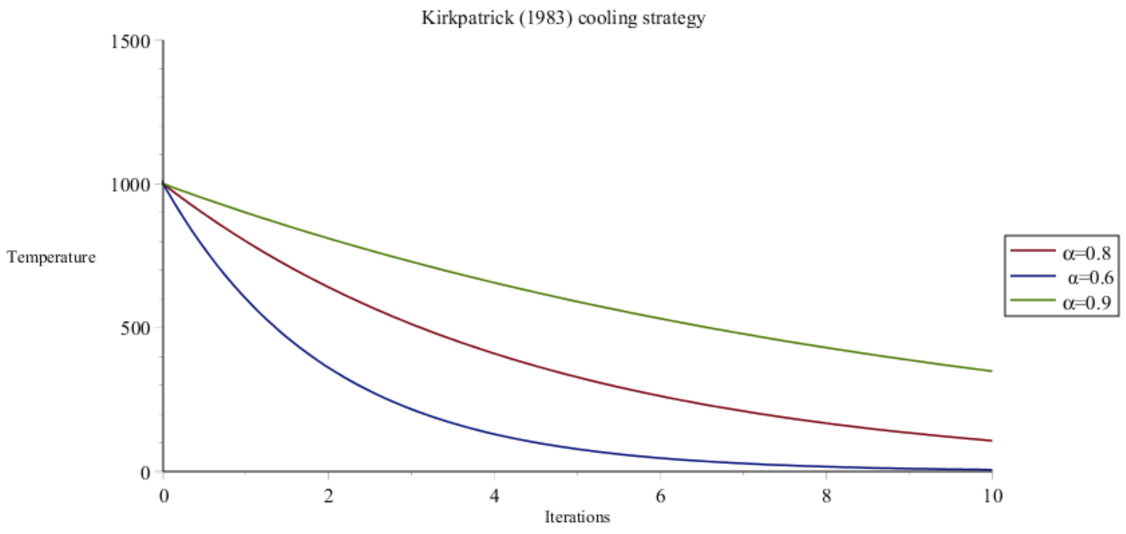


Figure 5.4: Kirkpatrick (1983) cooling strategy with $T_{init} = 1000$.

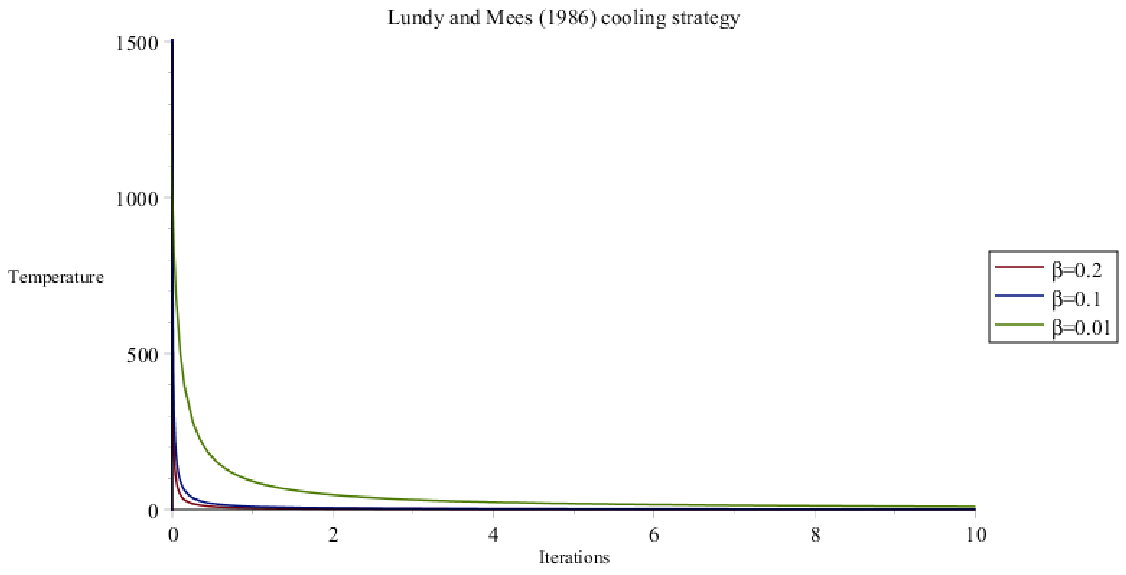


Figure 5.5: Lundy and Mees (1986) cooling strategy with $T_{init} = 1000$.

The problem of SA is that it requires a predefined set of parameters and system configurations, and for different problems these configurations are different. To demonstrate this issue, SA is applied on a sample problem with different configurations. The sample problem is randomly generated with 100 nodes on a Cartesian coordinate with random time windows and four vehicles available. Same speed profile as in Chapter 3 is applied. The results of various configurations are illustrated in Figures 5.6 - 5.9.

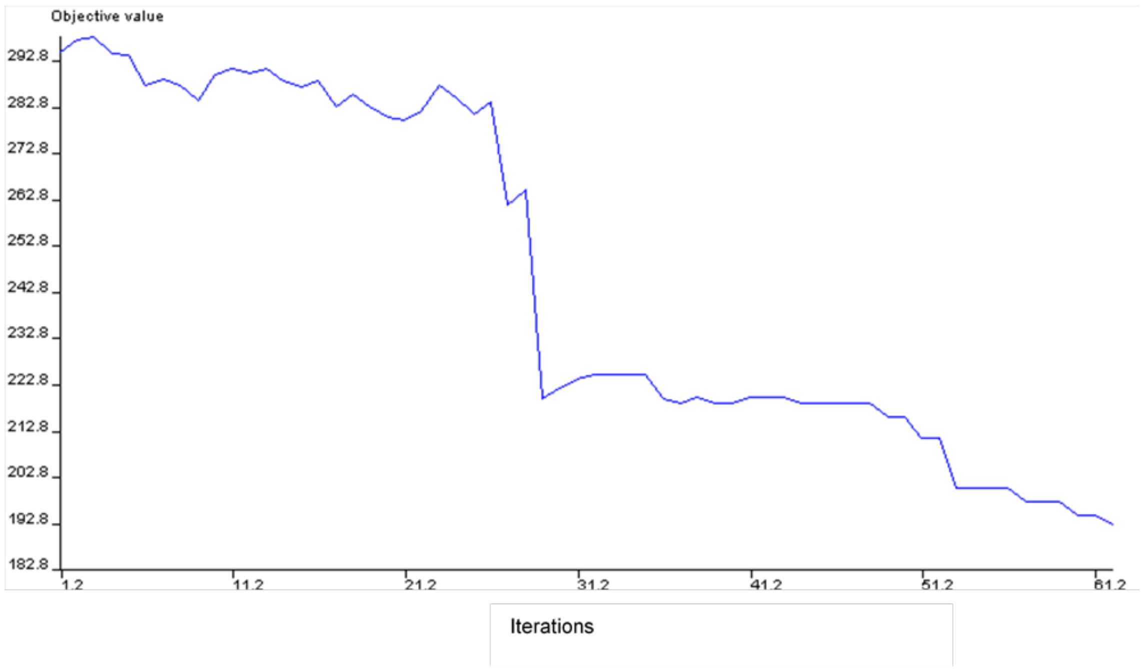


Figure 5.6: Configurations 1: $T_{init} = 10000$, Kirkpatrick cooling, $\alpha = 0.8$, $T_{min} = 0.01$.

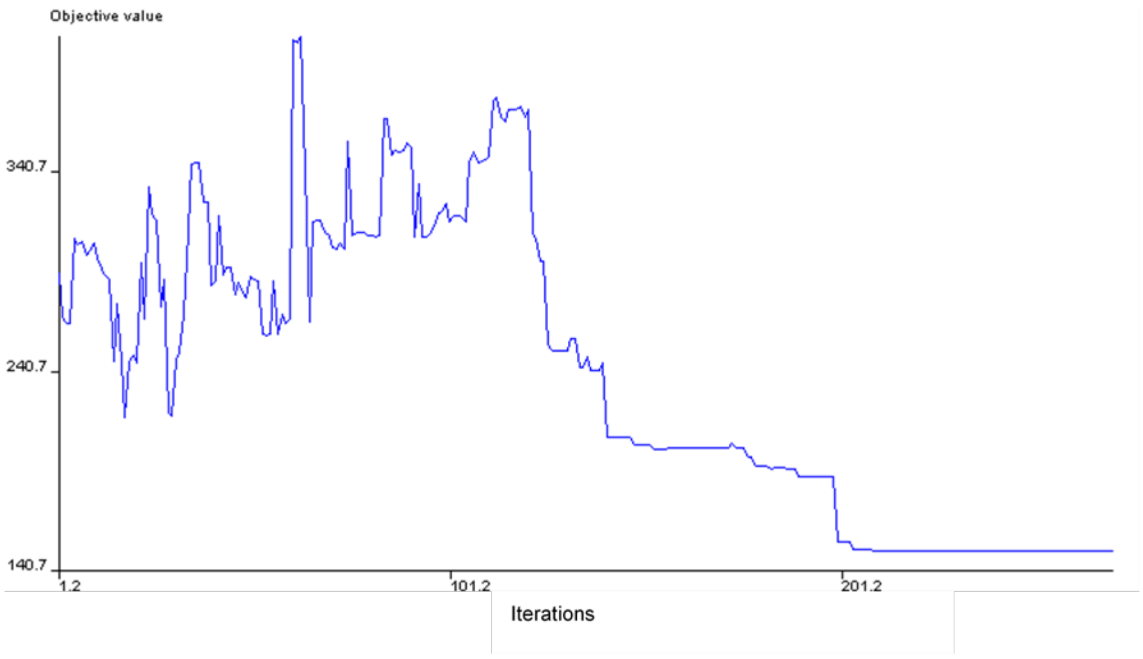


Figure 5.7: Configurations 2: $T_{init} = 10000$, Kirkpatrick cooling, $\alpha = 0.95$, $T_{min} = 0.01$.

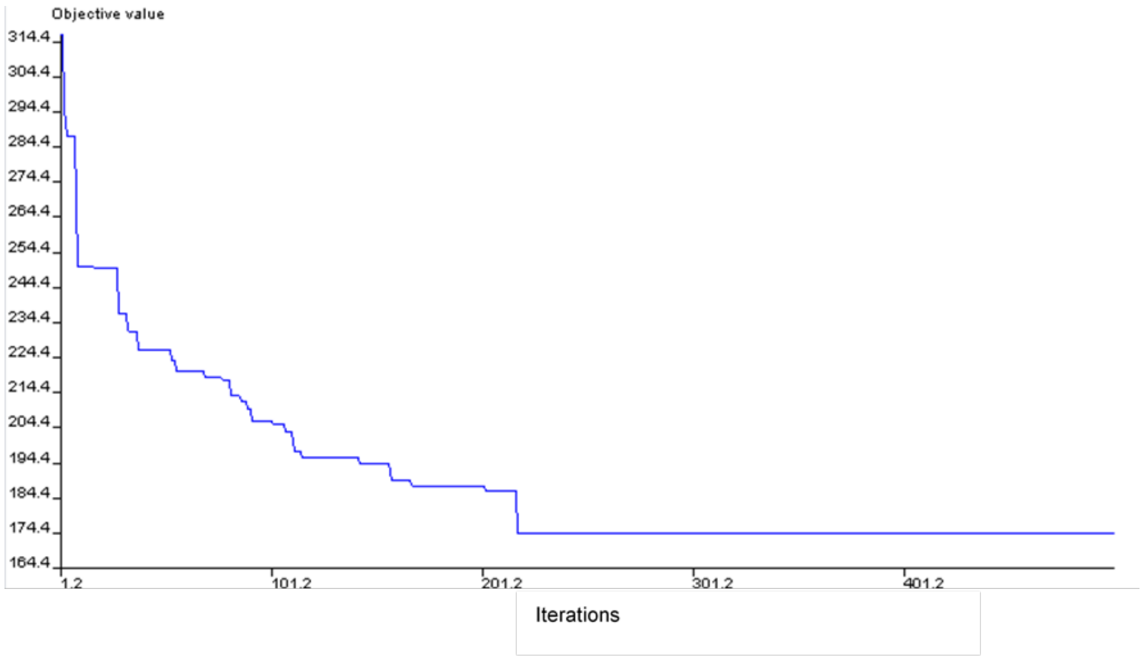


Figure 5.8: Configurations 3: $T_{init} = 10000$, Lundy and Mees cooling, $\beta = 0.2$, $T_{min} = 0.01$.

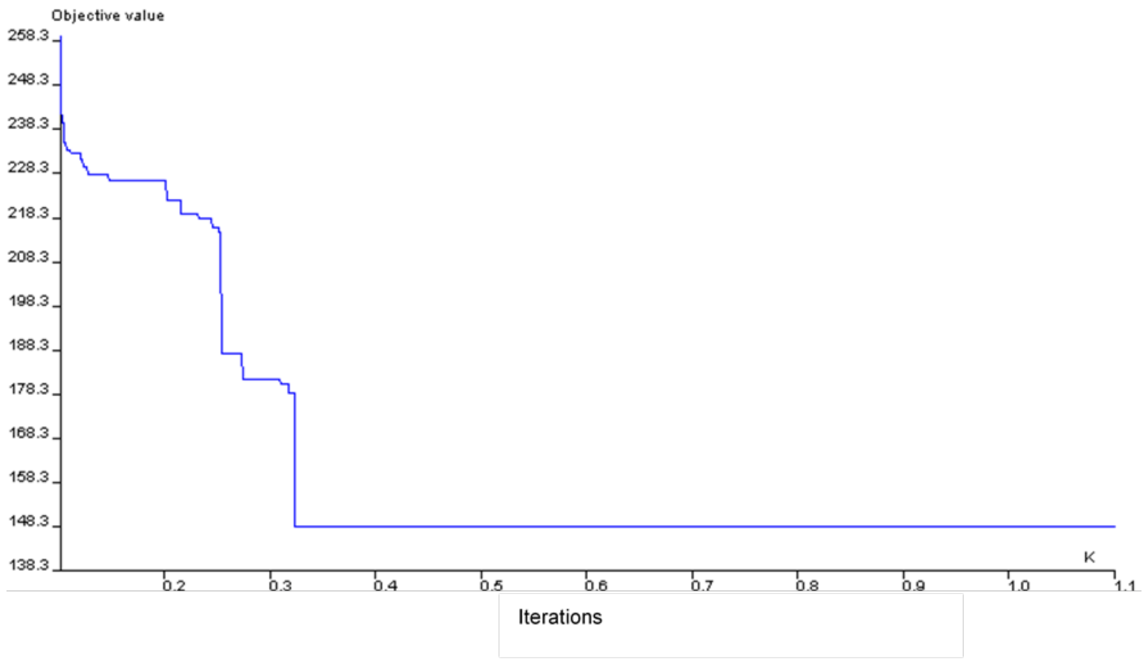


Figure 5.9: Configurations 4: $T_{init} = 10000$, Lundy and Mees cooling, $\beta = 0.1$, $T_{min} = 0.01$.

Among all those four configurations, Configuration 4 as in Figure 5.9 yields the best result but with the longest computational time 0.234 s while Configuration 1 (Figure 5.6) only consumes 0.031 s. Observations on the four searching paths show that the last three configurations run for more iterations, but the last part of the iterations show no improvement in the objective value. This is due to the fact that as temperature drops, the probability of accepting worse solution decreases and the search process are easily trapped to plateau solutions. To avoid this situation, we propose a new self-adaptive version of SA based on the moving variance of objective values (Algorithm 5.7). During the search process of this new version of SA, the moving variance of solutions of iterations are calculated, and if the moving variance is small which indicates a straight line in the search path, then for a worse solution we increase the probability of accepting it by temporary reheating the system temperature by a predefined adjust factor. With the self-adaptive SA, we introduced a few more control parameters. The reheating process only takes effect at the lower system temperature state, as when the system temperature is high, SA is doing a coarse search and the probability of accepting worse solution is already high and we don't want to increase it at that time. In the numerical experiments, the parameters are set to

$$small\ number = 10^{-6},\ \theta = 50000,\ AdjustFactor = 30.$$

Algorithm 5.7 Self-adaptive Simulated Annealing Algorithm

```
1:  Function Self-adaptive SA($x$);
2:  Setting the system configurations
3:  $T \leftarrow T\_init$
4:  Repeat
5:  $n \leftarrow 0$
6:     Repeat
7:        $\gamma \leftarrow 1$
8:        $x^{'} \leftarrow BestShake(x)$
9:        $\triangle E^{'} \leftarrow f(x^{'}) - f(x)$
10:       $\sigma^2 \leftarrow MovingVariance$
11:       IF $\triangle E < 0, x \leftarrow x^{'}$
12:       Else if $\sigma^2 < small\ number\ and\ T < \frac{T_0}{\theta}$, $\gamma \leftarrow AdjustFactor$, let $x \leftarrow x^{'}$
    with probability $P = e^{-\frac{\triangle E}{\gamma T}}$
13:       $n \leftarrow n + 1$
14:    Until n=N(t)
15:    Update ($T$)
16: Until  $T < T\_min$
```

5.8.1 Compare traditional simulated annealing and the new self-adaptive simulated annealing

Figure 5.10 demonstrates the searching progresses of a test case which is solved both by SA and self-adaptive SA. The SA reaches a local optimal solution in the first few iterations and gets trapped there, so a straight line is shown on the search path (blue line). However, with the self-adaptive SA, a long segment of the straight line will not be seen as the self-adaptive mechanism will increase the probability of accepting a worse solution if the search path is a straight line for a short while. We can see that the self-adaptive SA finds a better solution at a late stage of the search path and once it is experiencing a straight line, it will accept worse solution and then be able to escape from the local minimum. On this test case, self-adaptive SA can improve the optimal solution by 8.84% and the computational time is the same for 2.252 s. There are random natures of the SA algorithms, so we need to test them on the same test case many times and take the average improvement.

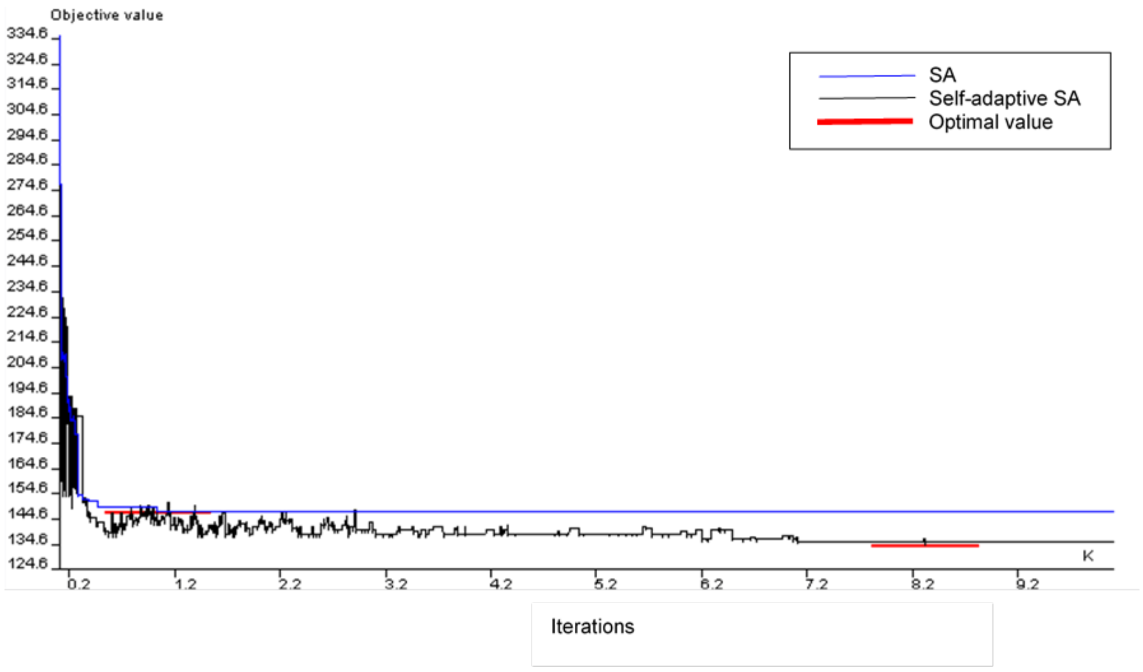


Figure 5.10: Comparisons of SA and adaptive SA with configurations: $T_{init} = 10000$, Lundy and Mees cooling, $\beta = 0.01, T_{min} = 0.01$.

The self-adaptive SA is tested on ten different test cases with 30 customers each, and both SA and self-adaptive SA are applied to each test case 200 times, the average solutions and the best solutions of each algorithm are demonstrated in Table 5.4. Self-adaptive SA on average achieves a better final solution on every test case, and on average improves the mean final solutions of SA by 12.76%. Furthermore, self-adaptive SA generates the best solution for all ten test cases, and the average lowest minimum is also 4.39% lower than that of SA. This shows the excellent performance of our new self-adaptive simulated annealing algorithm. This new self-adaptive SA will be applied in Chapter 7.

Problem	SA		Self-adaptive SA		Improvement	
	Average	Best	Average	Best	Average	Best
1	156.17	135.61	137.37	131.83	12.03816	2.787405
2	135.83	103.47	110.87	100.66	18.37591	2.715763
3	112.49	94.16	100.1	93.3	11.01431	0.913339
4	95.98	84.57	86.97	79.91	9.387372	5.510228
5	159.35	116	129.42	107.81	18.78255	7.060345
6	112.37	101.87	102.17	92.05	9.077156	9.639737
7	110.81	90.44	93.96	86.94	15.20621	3.869969
8	107.3	84.55	89.56	79.59	16.53308	5.866351
9	119.08	102.06	106.82	99.02	10.2956	2.97864
10	107.81	97.23	100.38	94.77	6.891754	2.530083
Average					**12.76021**	**4.387186**

Table 5.4: Comparison between SA and the new self-adaptive SA.

5.9 Summary

In this chapter, we first implemented the basic tabu search algorithm, variable neighbourhood descent algorithm, reduced variable neighbourhood algorithm and compared them on our green time-dependent VRP with time windows. RVNS outperformed VND for every test case. The computational time of tabu search algorithm was on average 5.37 times longer than RVNS for test cases of the same results. For the other test cases, tabu search algorithm gave better results than RVNS, the CO_2 emissions were on average 5.35% less, but at an expense of 11.72 times of computational time. Although for most of the test cases, the best solution was found by tabu search algorithm, results suggest that RVNS is a good trade-off for industry application. Then simulated annealing algorithm was tested, and the problems with this algorithm were revealed. For improvement, a new self-adaptive simulated annealing algorithm was proposed and compared with the previous normal SA. Experiment results showed a robust better performance of the self-adaptive version of

SA. Self-adaptive SA on average got a better solution on every test case, and on average improved the solutions of SA by 12.76%. Furthermore, self-adaptive SA generated the best solution for all ten test cases, and the average lowest minimum objective value was 4.39% lower than that of SA. Based on the comparison results, both VNS and ASA perform well for green vehicle routing problem with time windows and each has its own advantages. To explore the benefits of metaheuristics and make more contributions to metaheuristics in green logistics, both algorithms will be implemented. VNS is selected as the solution engine for Chapter 6, and the performance of VNS is improved by parallelism. The new ASA will be applied to the problem in Chapter 7.

In the next chapter, we will extend the problem in Chapter 4 to a real-life sized problem with a multinomial logit demand model. Furthermore, the VNS algorithm proposed in this chapter will also be used as the solution engine.

6 Online booking systems for green service delivery

In this chapter, we extend the problem studied in Chapter 4 to include real distances among the depot and customers sites, which makes the problem more general and applicable to any geographical settings instead of a confined area. Also, while Chapter 4 used software solver to solve the problem, in this chapter some of the algorithms investigated in Chapter 5 are applied. Furthermore, a more sophisticated discrete choice pricing model is applied.

Here we consider the same problem of a company who sends engineers with vehicles to customer sites to provide services. A customer's request may arrive anytime during the day. The company then provides the customer with a list of time window options and their corresponding prices. The prices reflect the cost of the delivery option, which is calculated using the cost of CO_2 emissions. We followed the two-phase solution framework as in Chapter 4. Once a new customer arrives, the system first solves a time-dependent vehicle routing problem to minimise emissions for each time window available. Then for each available option, the minimum additional cost for including this customer is calculated. In the second phase, a price will be determined for each option by solving a discrete choice pricing model based on the additional costs previously calculated. The whole booking system follows a time rolling mechanism. Metaheuristic methods are applied to real-life business applications which enable the solution framework to be applied online since shorter computational time is required. To be more specific, parallel variable neighbourhood search is applied to the first phase problem with a dynamic programming method to optimise departure times. The differential evolution algorithm solves the second phase problem. The solution framework is tested through simulation experiments. Dynamic pricing techniques are compared to fixed pricing strategies. Results show that dynamic pricing leads to a reduction in CO_2 emissions and an improvement in profits.

The rest of chapter is organised as follows. The problem description is in Section 6.1. The mathematical model for the first phase problem and algorithms to solve it are shown in Sections 6.2 and 6.3 respectively. Sections 6.4 and 6.5 formulated the mathematical model for the second phase problem and compare different algorithms to solve the model and choose to use the differential evolution algorithm. The simulation results and sensitivity tests are shown in Section 6.6. Finally, concluding remarks are given in Section 6.7.

6.1 Problem description

In our research, we study a problem where a company sends engineers to customer sites to provide maintenance and installation services in London, UK. For example, a pool of 100 customers is shown in Figure 6.1, where the home circle shows the location of the company site (depot). The real road distance between two coordinates (latitude and longitude) on the map is obtained by calling web service Google Maps Distance Matrix API. Traffic speed is not a constant over a day, and the hourly average travel speeds of London are shown in Figure 6.2. Customer requests will be handled whenever they arrive. The service time can be estimated according to the type of a task. Based on historical data, in our experiment, we assume that the maximum, minimum and mean service times for all customers are 480 minutes, 37 minutes and 92 minutes, respectively, and that the customer arrival process follows a Poisson distribution with mean inter-arrival time of 7.2 minutes.

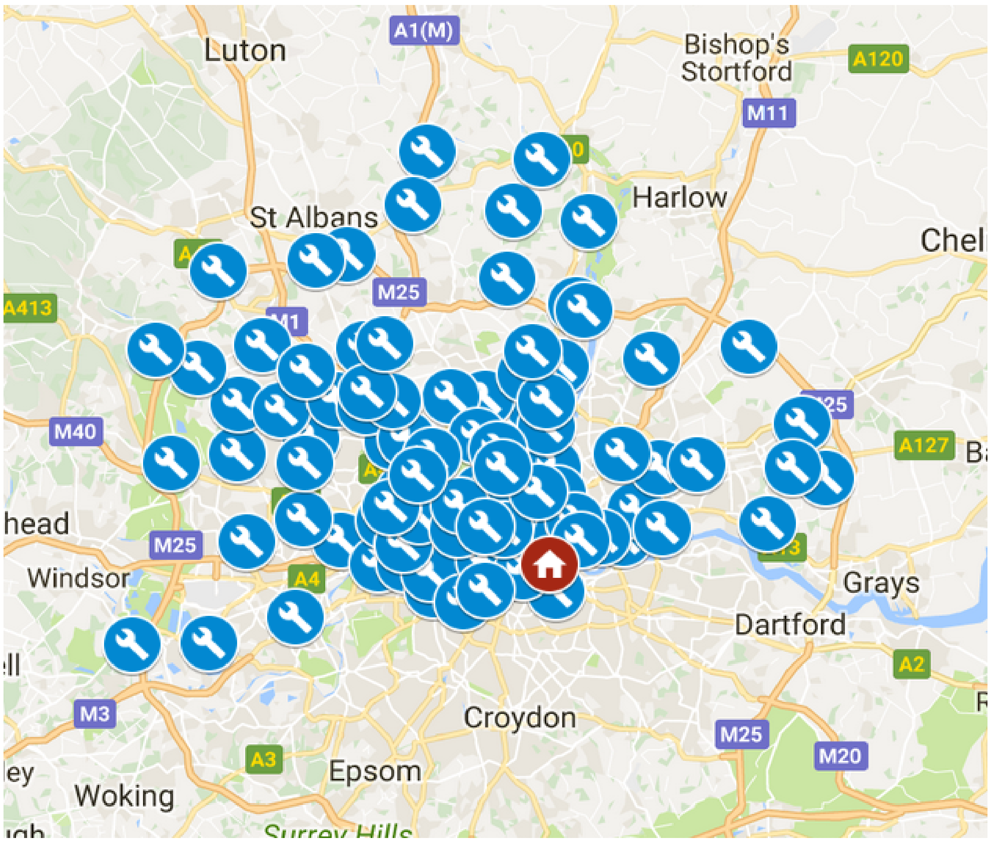


Figure 6.1: 100 Customers (Sample data) locations and depot location.

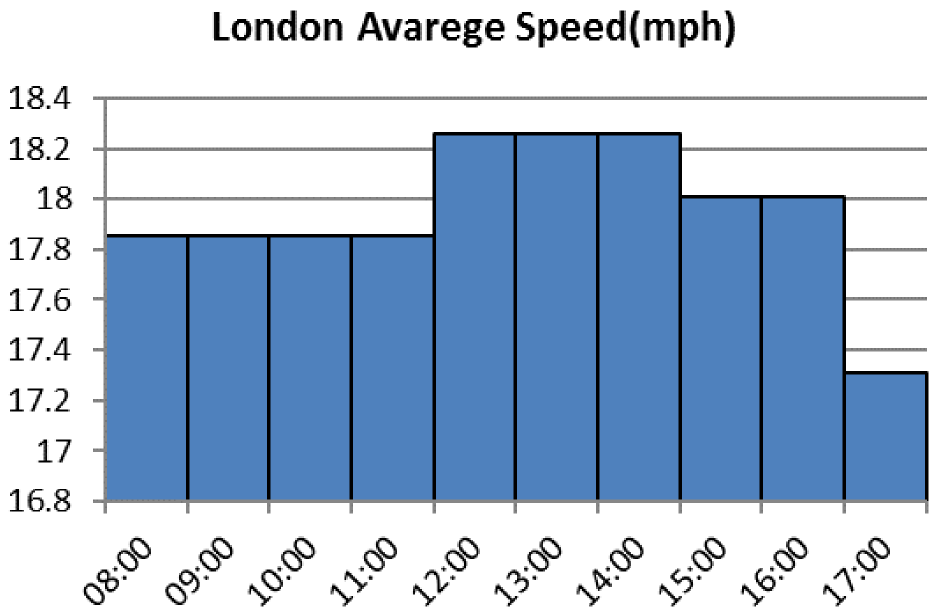


Figure 6.2: Average hourly travel speed profile of London, UK.

Customers book for the service through a website or by telephone. The company takes a rolling booking approach. The company's booking system starts at 6:00am every day to accept bookings, and ends at 18:00pm; and no booking is available after that time. Meanwhile, the engineers start to work at 8:00am and finish work at 18:00pm. The company provides five delivery options, and each delivery option has a two-hour time window. There are five cut-off times daily: 8:00am, 10:00am, 12:00pm, 14:00pm, 16:00pm. A customer missing the cut-off time for a delivery time window can book one of the next available time windows. An example of the customers' arrival versus five available time windows is demonstrated in Figure 6.3.

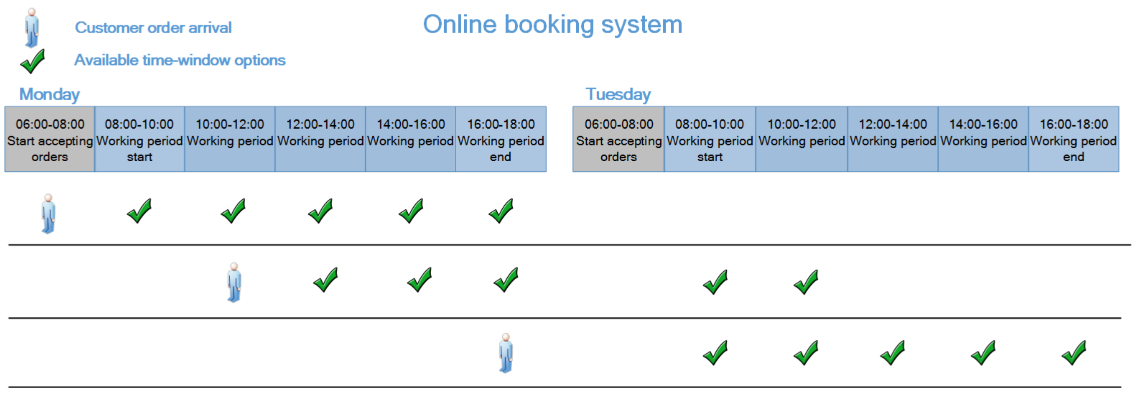

Customer order arrival
Available time-window options
Online booking system
Monday
06:00-08:00 Start accepting orders
08:00-10:00 Working period start
10:00-12:00 Working period
12:00-14:00 Working period
14:00-16:00 Working period
16:00-18:00 Working period end
Tuesday
06:00-08:00 Start accepting orders
08:00-10:00 Working period start
10:00-12:00 Working period
12:00-14:00 Working period
14:00-16:00 Working period
16:00-18:00 Working period end

Figure 6.3: Available time windows in the rolling mechanism.

With this rolling mechanism, each customer is guaranteed five delivery options. Prices for each time window option are dynamically determined by the company to influence the customer's choice.

The dynamic scheduling and pricing system works as follows: Once a new customer arrives to make a booking, the solution approach, as shown in the flowchart (see Figure 6.4), is taken. Upon the customer arrival, the system generates five delivery time window options based on the current arrival time. Then, for each delivery option, it tries to fit this customer into existing schedules by inserting it to make a new lowest-cost schedule. At the same time, the opportunity cost of this time window is computed. Opportunity cost is defined as the value of the time window option, with the ones closest to expire being least valuable because the company wants to minimise engineers' idle time and any unused time of engineer is wasted.

Repeating this for all five options provides a list of costs for all available delivery time window options. Then it inputs these into the dynamic pricing problem which provides a price menu for all the delivery options, and those are relayed to the customer. The customer chooses one of them, and the system updates the schedule accordingly. This process is repeated for all incoming customers. The solution algorithms are introduced in the following sections.

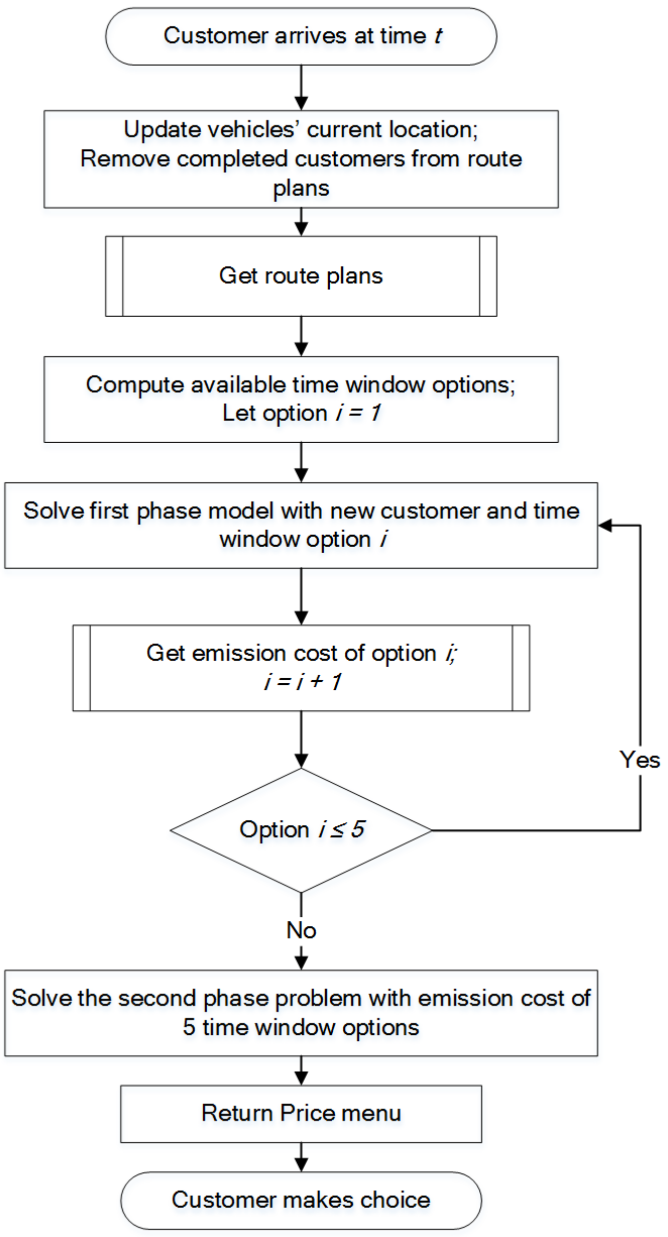

Customer arrives at time *t*
Update vehicles' current location; Remove completed customers from route plans
Get route plans
Compute available time window options; Let option *i = 1*
Solve first phase model with new customer and time window option *i*
Get emission cost of option *i*; *i = i + 1*
Option *i ≤ 5*
Yes
No
Solve the second phase problem with emission cost of 5 time window options
Return Price menu
Customer makes choice

Figure 6.4: Online pricing system flowchart.

6.2 First phase model: Green time-dependent VRPTW with heterogenous fleet

In this section, we present the mixed integer linear programming formulation for the green vehicle routing problem of the first stage. Different from the traditional vehicle routing problem, this model minimises CO_2 emissions and involves time-dependent travel speed, time window constraints and a heterogeneous fleet.

Index

i, j : index of customers; $i, j = 0$ for the depot;

k: index of vehicles;

t: index of times slots.

Parameters:

n: total number of customers;

K: total number of vehicles available;

K_j: set of vehicles/engineers that has the skill to do job at customer j;

M: a big positive number;

T: total time of the planning horizon;

S: total number of time slots;

d_{ij}: distance of travelling from i to j;

s_i: service time at customer i; $s_0 = 0$;

b_i, e_i: begin time and end time of time window of customer i;

b_0, e_0: start and end times of the working period for engineers;

L_t: start time of time slot t;

v_t: travel speed in time slot t in km/h;

c_{kt}: emissions rate of vehicle k in time slot t in kg/km.

Variables

C_{ijk}: the amount of emissions (in kg) generated by travelling from i to j by vehicle k;

$$x_{ijk} = \begin{cases} 1, \text{ if vehicle k travel from customer i to j} \\ 0, \text{ otherwise} \end{cases};$$

u_i: time point when a vehicle starts travelling to customer i, for $i = 1, \ldots, n$;

U_k: time point when vehicle k travelling back to depot;

w_i: time point when the service at customer i starts, for $i = 1, \ldots, n$;

W_k: time point when vehicle k gets back to depot;

τ_{ij}: travel time from customer i to customer j;

$$z1_{it} = \begin{cases} 1, \text{ if travelling to customer i starts in time slot t} \\ 0, \text{ otherwise} \end{cases}, \text{for } i = 1, \ldots, n;$$

$$z2_{it} = \begin{cases} 1, \text{ if travelling to customer i ends in time slot t} \\ 0, \text{ otherwise} \end{cases}, \text{for } i = 1, \ldots, n;$$

$$Z1_{kt} = \begin{cases} 1, \text{ if vehicle k travelling back to depot starts in time slot t} \\ 0, \text{ otherwise} \end{cases};$$

$$Z2_{kt} = \begin{cases} 1, \text{ if vehicle k travelling back to depot ends in time slot t} \\ 0, \text{ otherwise} \end{cases}.$$

Minimize $\sum_{i=0}^{N} \sum_{j=0, i \neq j}^{N} \sum_{k=1}^{K} C_{ijk}$

Subject to

$$\sum_{i=0, i \neq j}^{n} \sum_{k \in K_j} x_{ijk} = 1, \quad \forall j = 1, \ldots, n \tag{6.1}$$

$$\sum_{i=0, i \neq j}^{n} x_{ijk} = \sum_{i=0, i \neq j}^{n} x_{ijk}, \quad \forall j = 1, \ldots, n; k = 1, \ldots, K \tag{6.2}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{K} x_{0jk} \leq K \tag{6.3}$$

$$\sum_{i=1}^{n}\sum_{k=1}^{K} x_{i0k} = \sum_{j=1}^{n}\sum_{k=1}^{K} x_{0jk} \tag{6.4}$$

$$b_i \leq w_i \leq e_i, \quad \forall i = 1, \ldots, n \tag{6.5}$$

$$b_0 \leq W_k \leq e_0, \quad \forall k = 1, \ldots, K \tag{6.6}$$

$$w_i - u_j + M \sum_{k=1}^{K} x_{ijk} \leq M - s_i, \quad \forall i = 1, \ldots, n; j = 1, \ldots, n; i \neq j \tag{6.7}$$

$$w_i - U_k + M x_{i0k} \leq M - s_i, \quad \forall i = 1, \ldots, n; k = 1, \ldots, K \tag{6.8}$$

$$w_j \geq u_j + \tau_{ij} + M\left(\sum_{k=1}^{K} x_{ijk} - 1\right), \quad \forall i = 0, \ldots, n; j = 1, \ldots, n; i \neq j \tag{6.9}$$

$$W_k \geq U_k + \tau_{i0} + M(x_{i0k} - 1), \quad \forall i = 1, \ldots, n;\ k = 1, \ldots, K \tag{6.10}$$

$$u_j \geq L_t + T\left(z1_{jt} - 1\right), \quad \forall j = 1, \ldots, n;\ t = 1, \ldots, S \tag{6.11}$$

$$U_k \geq L_t + T\left(Z1_{kt} - 1\right), \quad \forall k = 1, \ldots, K;\ t = 1, \ldots, S \tag{6.12}$$

$$u_j \leq L_{t+1} + T\left(1 - z1_{jt}\right), \quad \forall j = 1, \ldots, n;\ t = 1, \ldots, S \tag{6.13}$$

$$U_k \leq L_{t+1} + T\left(1 - Z1_{kt}\right), \quad \forall k = 1, \ldots, K;\ t = 1, \ldots, S \tag{6.14}$$

$$u_j + \tau_{ij} \geq L_t + T\left(z2_{jt} + \sum_{k=1}^{K} x_{ijk} - 2\right), \quad \forall i = 0, \ldots n;\ j = 1, \ldots, n; i \neq j;$$
$$t = 1, \ldots, S \tag{6.15}$$

$$U_k + \tau_{i0} \geq L_t + T(Z2_{kt} + x_{i0k} - 2), \quad \forall i = 1, \ldots, n;\ k = 1, \ldots, K; t = 1, \ldots, S \tag{6.16}$$

$$u_j + \tau_{ij} \leq L_{t+1} + T\left(2 - z2_{jt} - \sum_{k=1}^{K} x_{ijk}\right), \forall i = 0, \ldots, n; j = 1, \ldots, n;\ i \neq j;$$
$$t = 1, \ldots, S \tag{6.17}$$

$$U_k + \tau_{i0} \leq L_{t+1} + T\left(2 - Z2_{kt} - x_{i0k}\right), \quad \forall i = 1, \ldots, n; k = 1, \ldots, K;$$
$$t = 1, \ldots, S \tag{6.18}$$

$$\sum_{t=1}^{S} z1_{jt} = 1, \quad \forall j = 1, \ldots, n \tag{6.19}$$

$$\sum_{t=1}^{S} Z1_{kt} = 1, \quad \forall k = 1, \ldots, K \tag{6.20}$$

$$\sum_{t=1}^{S} z2_{jt} = 1, \quad \forall j = 1, \ldots, n \tag{6.21}$$

$$\sum_{t=1}^{S} Z2_{kt} = 1, \quad \forall k = 1, \ldots, K \tag{6.22}$$

$$\tau_{ij} \geq \frac{d_{ij}}{v_t} + M\left(\sum_{k=1}^{K} x_{ijk} + z1_{jt} + z2_{jt} - 3\right), \quad \forall i = 0, \ldots, n;\ j = 1, \ldots, n; i \neq j;$$
$$t = 1, \ldots, S \tag{6.23}$$

$$\tau_{i0} \geq \frac{d_{i0}}{v_t} + M\left(x_{i0k} + Z1_{kt} + Z2_{kt} - 3\right), \quad \forall i = 1, \ldots, n;\ t = 1, \ldots, S; k = 1, \ldots, K \tag{6.24}$$

$$\tau_{ij} \geq L_{t+1} - u_j + \frac{[d_{ij} - v_t(L_{t+1} - u_j)]}{v_{t+1}} + M\left(\sum_{k=1}^{K} x_{ijk} + z1_{jt} + z2_{jt+1} - 3\right),$$
$$\forall i = 0, \ldots, n;\ j = 1, \ldots, n;\ i \neq j;\ t = 1, \ldots, S \tag{6.25}$$

$$\tau_{i0} \geq L_{t+1} - U_k + \frac{[d_{i0} - v_t(L_{t+1} - U_k)]}{v_{t+1}} + M\left(x_{i0k} + Z1_{kt} + Z2_{kt+1} - 3\right),$$
$$\forall i = 1, \ldots, n;\ t = 1, \ldots, S; k = 1, \ldots, K \tag{6.26}$$

$$C_{ijk} \geq c_{kt} \cdot d_{ij} + M\left(x_{ijk} + z1_{jt} + z2_{jt} - 3\right), \quad \forall i = 1, \ldots, n;\ j = 1, \ldots, n; i \neq j;$$
$$k = 1, \ldots, K;\ t = 1, \ldots, S \tag{6.27}$$

$$C_{i0k} \geq c_{kt} \cdot d_{i0} + M\left(x_{i0k} + Z1_{kt} + Z2_{kt} - 3\right), \quad \forall i = 0, \ldots, n; k = 1, \ldots, K;$$
$$t = 1, \ldots, S \tag{6.28}$$

$$C_{ijk} \geq c_{kt} \cdot v_t\left(L_{t+1} - u_j\right) + c_{kt+1} \cdot \left[d_{ij} - v_t\left(L_{t+1} - u_j\right)\right] + M(x_{ijk} + z1_{jt} + z2_{jt+1} -$$
$$3), \quad \forall i = 0, \ldots, n;\ j = 1, \ldots, n;\ i \neq j; k = 1, \ldots, K;\ t = 1, \ldots, S \tag{6.29}$$

$$C_{i0k} \geq c_{kt} \cdot v_t\left(L_{t+1} - U_k\right) + c_{kt+1} \cdot [d_{i0} - v_t(L_{t+1} - U_k)] + M(x_{i0k} + Z1_{kt} + Z2_{kt+1}$$
$$- 3), \quad \forall i = 1, \ldots, n;\ k = 1, \ldots, K;\ t = 1, \ldots, S \tag{6.30}$$

$$\tau_{ij} \leq M \sum_{k=1}^{K} x_{ijk}, \quad \forall i, j = 0, \ldots, n;\ i \neq j \tag{6.31}$$

$$0 \leq \sum_{t=1}^{S} t \cdot z2_{jt} - \sum_{t=1}^{S} t \cdot z1_{jt} \leq 1, \quad \forall j = 1, \dots, n \tag{6.32}$$

$$0 \leq \sum_{t=1}^{S} t \cdot Z2_{kt} - \sum_{t=1}^{S} t \cdot Z1_{kt} \leq 1, \quad \forall k = 1, \dots, K \tag{6.33}$$

and the nonnegativity and binary constraints for the variables.

The objective of the model is to minimise the total amount of CO_2 emissions generated by travelling along the links. Constraints (6.1) and (6.2) indicate that there is exactly one vehicle going into each customer node and the same vehicle leaves the customer node (each customer node is served only once). Constraints (6.3) and (6.4) mean that at most K vehicles can leave the depot and the same number will travel back to the depot at the end. Constraints (6.5) and (6.6) specify the time windows for starting customer services, and working periods for the engineers. Constraints (6.7) and (6.8) ensure that a vehicle can start travelling to the next customer only after finishing the task at the current customer. These also work as sub-tour elimination constraints. There may be a gap (idle time) between task finish time and the time of travelling to the next customer. Constraints (6.9) and (6.10) require that tasks at customers' cites can start only after the vehicle arrives there. Constraints (6.11) through (6.12) determine travel start and end times. Constraints (6.13) through (6.22) mean that the start or end of each travel must fall into at most two time slots. Constraints (6.23) through (6.26) calculate the travel time between two nodes with time-dependent travel speed. If the start and end of travel are in the same time slot, travel time equals travel distance divided by the travel speed of the time slot: $\tau_{ij} = \frac{d_{ij}}{v_t}$. If the start and end of travel are in two time slots, the travel time consists two parts, the first part is the end of the first time slot minus start travel time $(L_{t+1} - u_j)$ and the second part is the travel time in the second time slot $(\frac{[d_{ij} - v_t(L_{t+1} - u_j)]}{v_{t+1}})$. Constraints (6.27) through (6.30) determine the amount of CO_2 emissions associated with each link. Constraints (6.31) bound the travel time between two nodes to zero if they are not linked. Constraints (6.32) and (6.33) assume that the maximum gap between the start and end of travelling on any link is one time slot and the minimum is zero time slot, i.e., travelling between any two nodes will not span into three different time slots. In this model CO_2 emissions rate is different for different vehicle types and for different speeds. The emissions rate is calculated as demonstrated in Chapter 3.

6.3 Solution algorithms for the first phase problem

The problem of interest is an online problem which requires a quick system response. Optimisation software such as FICO Xpress solver will take too long to solve the first stage model. For a 10 customer case as shown in Section 6.3.3, the solver runs for 20 hours and then crushes as the system memory runs out. So in this section, a metaheuristic will be applied. Modern PCs have a parallel processor structure with several computing cores that support multi-threading and task parallelism. Considering the computational complexity of real-life vehicle routing problems (VRPs) and the parallel nature of the metaheuristics applied, it is appropriate to take advantage of parallel computing to boost system performance.

6.3.1 Parallel variable neighbourhood search

Variable neighbourhood search (VNS) was first developed in 1997 (Mladenović and Hansen, 1997). It finds its success in combinatorial optimisation applications. The basic idea is to find a local minimum in a descent method in one neighbourhood, and then try to escape this local minimum by exploring a distant neighbourhood of this solution in a sequential way among the candidate neighbourhoods or in a random fashion. The search result in a new neighbourhood is accepted only if the local minimum found in this neighbourhood is better. Unlike tabu search or simulated annealing (SA), VNS is not a trajectory-following method which accepts a less-favourable solution. The advantages of the VNS method are that it is parameter-free and easy to implement. Thus, it could be applied to different problems without extra effort on parameters tuning.

The pseudo code of the general VNS is shown in Algorithm 6.1. We use six

($k = 6$) common neighbourhood constructors for VRP type of problems: Cross, OrOpt, Relocate, Swap, 2 Opt and 2 Opt* (as shown in Chapter 5). During the local search, a solution is randomly generated using the current neighbourhood operator k (step 5) to avoid cycling, and variable neighbourhood descent method is used to get the best improvement solution based on the randomly-generated solution (step 6). Once the random solution represented by task assignment and travel sequence is generated, optimal departure times at each customer are computed in step 7. In step 8 and step 9, if the solution obtained is better than the current solution, it is accepted and we restart from neighbourhood operator 1. Otherwise, we move to the next neighbourhood operator. This searching process terminates until it meets certain criterion. Here we use a maximum CPU time as termination criterion. Both Algorithms 6.1 and 6.2 adopt a deterministic sequence of applying neighbourhood operators.

Algorithm 6.1 Variable neighbourhood search

```
1: Function $VNS(x)$;
2: Repeat
3: $k \leftarrow 1$;
4:    Repeat
5:       $x^{'} \leftarrow Shake(x, k)$
6:       $x^{''} \leftarrow VND(x^{'})$
7:       OptimalDepartureTime ($x^{''}$)
8:       If $x^{''}$ is better than $x, x \leftarrow x^{''}, k \leftarrow 1$
9:       Else $k \leftarrow k + 1$
10:   Until $k = k\_max$
11: Until $CPUtime > time_{max}$
```

Algorithm 6.2 Variable neighbourhood descent

```
1: Function $VND(x)$;
2: Repeat
3: $k \leftarrow 1$;
4:     Repeat
5:         $x' \leftarrow BestNeighbour(x, k)$
6:         If $x'$ is better than $x, x \leftarrow x', k \leftarrow 1$
7:         Else $k \leftarrow k + 1$
8:     Until $k = k\_max$
9: Until no improvement is obtained
```

We extended the solution algorithm by applying parallelism in a simple way called replicated parallel VNS (PVNS). The parallelism approach in this research aims to enlarge the search space by using multiple independent search threads without increasing the computational time. It follows the multi-start strategy concept by increasing the number of neighbour solutions to start a local search. Several starting solutions are generated in the same neighbourhood where each local search is run in parallel using Java Multithreading. Then the best among all solutions is taken as the final solution.

6.3.2 Optimise departure times by dynamic programming

The previous solution algorithms optimise the allocation of customers to vehicles and the sequence of travel, but this assumes that the vehicle travelled as soon as it finished a task at a customer location and that no waiting is allowed at the customer location. However, allowing waiting at the customer location upon job completion and travelling at a non-congested period may further reduce the amount of CO_2 emissions. For the problem of optimising departure times (which affects the time slot that actual travel happens, along with emissions), we formulate the problem as a finite state deterministic dynamic programming problem.

The dynamic programming recursion equation is:

$$V_i(x) = min\{c(x,y) + V_{i+1}(y) : y \in Stage(i+1)\}, \tag{6.34}$$

where $V_i(x)$ is the optimal amount of emissions for customer i; x is the start travel time to customer i; i is the customer visiting sequence number $0 \ldots n+1$, and $n+1$ denotes the depot; $c(x,y)$ is the amount of emissions if the travel to customer i starts at x and to customer $i+1$ at y. The boundary conditions are:

$$x + s + \tau \leq y, \quad \forall\, pair(x,y), \tag{6.35}$$

where s is the service time at customer i and τ is the travel time from customer $i-1$ to i. Given $V_{n+1}(x) = constant$, we could recursively solve all $V_i(x)$ by backward deduction. The pseudo codes of dynamic programming procedures to optimise the start travel time to each customer i are shown in Algorithms 6.3 and 6.4.

Algorithm 6.3 Dynamic programming algorithm

```
1: Function DynamicProgramming;
2: $i \leftarrow N$
3: Repeat
4:    current $\leftarrow task_i$
5:    bestStt $task_{i+1}$
6:    FindSubTimePoint (current, bestStt ($task_{i+1}$));
7:    For each point in findSubTimeSlot(previous)
8:       findMinEmissionPath(previous)
9:    Update bestStt($task_i$)
10:   $i \leftarrow i - 1$
11: Until $i = 0$;
```

Algorithm 6.4 Function FindSubTimePoint in Dynamic Programming

```
1: Function FindSubTimePoint (current, bestStt($task_{i+1}$));
2: Set $s \leftarrow$ time interval length/step size
3:    Index $\leftarrow$ 0
4:    While (EarliestTime(current)+$s * Index \leq$ LatestTime(current))
5:    Add EarliestTime(current)+$s * Index$ to SubTimePoint list
6:    Index $\leftarrow$ Index+1;
7: Return SubTimePoint list
```

The key part of applying dynamic programming to this problem is to find the earliest and latest times that will not influence the feasibility of the route plans. The earliest time is the start travel time of the initial schedule since when we construct initial schedules, the start travel time is set to be as earliest as possible. The latest time of travelling to customer i is the earliest of either:

$$time\ window\ end\ of\ customer_i - travel\ time\ to\ customer_i$$

or

$$\begin{aligned} &optimal\ start\ travel\ time\ of\ customer_{i+1} - service\ time\ of\ customer_i \\ &- travel\ time\ to\ customer_i. \end{aligned}$$

For $n+1$, which is the stage the vehicle travels back to the depot, the latest time is set to be the end of the engineer's working period.

To demonstrate the effects of dynamic programming on emissions, we select eight engineers' (ID is aliased) optimal route plans and applied dynamic programming to the optimal route plans to get the optimal departure times for each customer. The hourly emissions are computed and the heatmap R lattice plot is shown in Figure 6.5 where darker areas mean more emissions. We can tell that the colouring before dynamic programming is darker than the colouring after dynamic programming. The total amount of emissions before dynamic programming is 148.31 kg and after dynamic programming is 132.37 kg, which is a 10.75% reduction in the emissions amount. There are several journeys scheduled in the peak morning hours of 7:00am - 9:00am before we apply dynamic programming. However, when we optimise start travel time with dynamic programming, those journeys are shifted towards a later non-peak time slot. This dynamic programming method will be applied with PVNS in every local search stage.

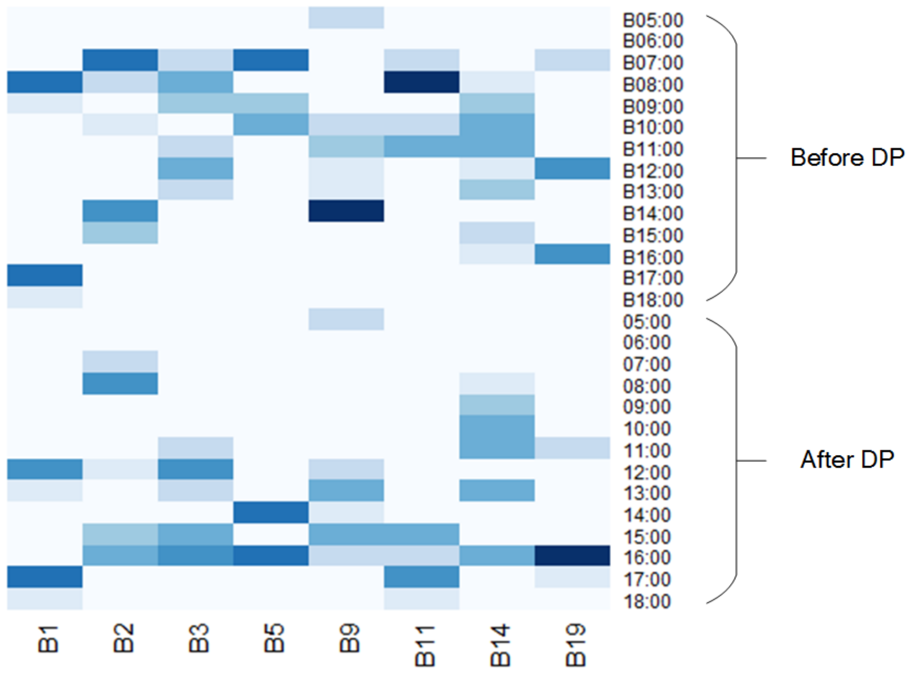


Figure 6.5: CO_2 emissions heatmap for 8 engineers before DP and after DP.

6.3.3 Comparison with software solver

Based on the MIP formulation in Section 6.3, the problem can be solved to optimal using Xpress optimiser for small problems. We generated ten small samples with customer sizes of 10 and compared the PVNS performance with the Xpress, which applied an exact method (branch-and-bound). This problem size is chosen because the largest problem size that could be solved within reasonable CPU times and system memory capacity was 10 customers. The numerical experiments are carried out on a PC with 64-bit Windows operating system, eight GigaBytes of RAM and an Intel i5 processor with quad-cores. PVNS is run on eight threads in parallel. The computational results are shown in Table 6.1.

On average, the gap between solutions found by PVNS and global optimal solutions given by Xpress is one percent, and seven out of 10 cases have a gap within 0.5 percent. Furthermore, the computational time of PVNS is on average 332.69 times faster than optimiser software. This indicates the PVNS metaheuristic method performs really well.

Problem	Objective value		Diff. in %	CPU times	
	Xpress	PVNS		Xpress	PVNS
1	51.26	51.38	0.2	32	10
2	42.88	43.12	0.5	2634	10
3	55.51	55.65	0.3	2928	10
4	32.94	32.94	0.0	9	10
5	49.11	49.59	0.98	5116	10
6	48.95	51.81	5.8	661	10
7	31.09	31.09	0.0	3679	10
8	47.03	47.40	0.79	10388	10
9	39.57	39.62	0.1	1094	10
10	39.51	39.59	0.2	6728	10
Average			**1.0**	**3326.9**	**10**

Table 6.1: Comparison results between Xpress solver and PVNS.

6.4 Second phase problem: delivery time windows pricing

Following the solution of vehicle scheduling and routing problems in the first stage, we consider how to dynamically price different service time window options for each arriving customer. A non-linear optimisation problem with the objective to maximise expected profit is derived.

$$\max : f = \sum_{i=1}^{5} (p_i - c_i - oppcost_i) P_i, \tag{6.36}$$

where f denotes expected profit, i denotes the time window option, parameters c_i is the emissions cost of option i, and $oppcost_i$ is the opportunity cost of option i; variables p_i is the price of option i, and P_i is the probability of choosing option i under the current price menu.

Profit is calculated as service price minus cost of CO_2 emissions, minus opportunity cost, where cost of CO_2 emissions is set to be £1 per kg. Opportunity cost is calculated as: $\alpha \cdot TimeDue$. Here $TimeDue$ is the period from current time until this time window option is no longer available. So, the time window is more valuable when it is further away from expiring. α is a decision-maker-specified weight parameter. Probability P_i is formulated as a multinomial logit (MLN) demand model.

6.4.1 Multinomial logit demand model

We model customer choice using random utility theory as in Ben-Akiva and Lerman (1985). The utility function of a choice i is: $U_{i,n} = V_{i,n} + \epsilon_{i,n}$, where V is the deterministic part: $V_n = \beta^T x_n$ and ϵ is the stochastic part following a Gumbel distribution. x_n is the vector of attributes of choice that influence utility. The choice with higher utility was chosen with greater probability. A customer n within the same category will be more likely to choose delivery option i if it has the highest utility among all options: $P_n(i) = P(U_{in} \geq U_{jn}, \forall\ j \in C_n, j \neq i)$ and C_n is the set of delivery options. The multinomial logit (MNL) model is expressed as

$$P_n(i) = e^{V_{in}} / \sum_{j \in C_n} e^{V_{jn}} \tag{6.37}$$

with $0 \leq P_n(i) \leq 1, \quad \forall i \in C_n$ and $\sum_{i \in C_n} P_n(i) = 1$.

In our problem, the vector of attributes x_n including alternative specific constants and the price are shown in Table 6.2. Walk away is also modelled as an option if the service is overpriced. We define customer reservation price as $p_{r,n}$.

Slots	Attributes					
	Alternative specific constant					Price
8:00-10:00	1	0	0	0	0	$p_{1,n}$
10:00-12:00	0	1	0	0	0	$p_{2,n}$
12:00-14:00	0	0	1	0	0	$p_{3,n}$
14:00-16:00	0	0	0	1	0	$p_{4,n}$
16:00-18:00	0	0	0	0	1	$p_{5,n}$
Walk away	0	0	0	0	0	$p_{r,n}$

Table 6.2: MNL Model choices and attributes.

In the following part, we will look at the properties of the demand model and optimal price menu.

Proposition 1. Price elasticity

Proposition 1.1. *If the price of one delivery option increases, its demand decreases.*

Proof $\frac{\partial P(i)}{\partial p_i} = -\beta P(i)(1-P(i)),\ as\ 1 > P(i) > 0$. $\beta > 0$ so $\frac{\partial P(i)}{\partial p_i} < 0$.

Proposition 1.2. *If the price of one delivery option increases, the demand for other delivery options increases.*

Proof $\frac{\partial P(i)}{\partial p_j} = -\beta P(i)P(j),\ as\ P(i) > 0,\ P(j) > 0,\ \beta > 0$ so $\frac{\partial P(i)}{\partial p_j} > 0$.

Proposition 2. *The price range for each delivery option is* $\left[min\{c_i\}, max\{\frac{\alpha_i}{\beta} + p_r\}\right]$.

Proof If the price is lower than the cost of the delivery option, and if $P(walk\ away) = \frac{e^{-\beta \cdot p_r}}{\sum_{i=1}^{n} e^{\alpha_i - \beta \cdot p_i + e^{-\beta \cdot p_r}}} \approxeq 1$, the customer will walk away without buying anything. When $e^{-\beta \cdot p_r} \gg \sum_{i=1}^{n} e^{\alpha_i - \beta \cdot p_i}$, which gives us $p_i > \frac{\alpha_i}{\beta} + p_r$. When the price for option i is greater than $\frac{\alpha_i}{\beta} + p_r$, the customer will walk away and this gives us the upper bound of the price for delivery option.

Proposition 3. *Greener delivery slots (if advertised) have smaller price elasticity than normal slots, which means smaller price discounts are enough to shift demand.*

Proof $\frac{\partial P(i)}{\partial p_i} = \frac{-\beta e^{\alpha_i - \beta p_i}}{\sum_{m=1}^{n} e^{\alpha_m - \beta \cdot p_m} + e^{-\beta \cdot p_r}}$, $\frac{\partial P(j)}{\partial p_j} = \frac{-\beta e^{\alpha_j - \beta p_j}}{\sum_{m=1}^{n} e^{\alpha_m - \beta \cdot p_m} + e^{-\beta \cdot p_r}}$, where i is the greener delivery option tagged and j is the normal delivery option. Then we have $\alpha_i > \alpha_j$, if they are fixed price initially $p_i = p_j$, we have $\frac{\partial P(i)}{\partial p_i} - \frac{\partial P(j)}{\partial p_j} = \frac{-\beta\left(e^{\alpha_i - \beta \cdot p_i} - e^{\alpha_j - \beta \cdot p_j}\right)}{\sum_{m=1}^{n} e^{\alpha_m - \beta \cdot p_m} + e^{-\beta \cdot p_r}} < 0$.

Proposition 4. *The optimal price for delivery option i has the following form: $p_i^* - c_i - \mu \tau_i = \frac{1}{\beta P(walk\ away)}$, where τ_i denotes the expiration time of delivery option i.*

Lemma 4.1. *Profit function f has a strong relative maximum at stationary point p^*, where $f' = 0$.*

Proof See Appendix F.

Proposition 5. *The optimal price for time window i decreases as CO_2 costs decrease.*

Proof We have $p_i^* - c_i - \mu \tau_i = \frac{\sum_{m=1}^{n} e^{\alpha_m - \beta \cdot p_m} + e^{-\beta \cdot p_r}}{\beta e^{-\beta \cdot p_r}}$ and move all p_i^* terms to the left and the rest to the right we have: $p_i^* - \frac{e^{\alpha_i - \beta \cdot p_i^*}}{\beta e^{-\beta \cdot p_r}} = c_i + \mu \tau_i + \frac{1}{\beta} + \frac{\sum_{m=1, m \neq i}^{n} e^{\alpha_m - \beta \cdot p_m}}{\beta e^{-\beta \cdot p_r}}$. Then take the partial derivative of c_i of both sides, $\frac{\partial}{\partial c_i}\left[p_i^* - \frac{e^{a_i - \beta \cdot p_i^*}}{\beta e^{-\beta \cdot p_r}}\right] = 1$. Based on the chain rule, we get: $\frac{\partial p_i^*}{\partial c_i} = \frac{1}{1 + p_i^* \cdot e^{\alpha_i - \beta \cdot p_i^* + \beta \cdot p_r}} > 0$.

Proposition 6. *The optimal price for time window i decreases as opportunity cost decrease if $\mu > 0$.*

Proof Like the previous proof, we can derive: $\frac{\partial p_i^*}{\partial \tau_i} = \frac{\mu}{1 + p_i^* \cdot e^{\alpha_i - \beta \cdot p_i^* + \beta \cdot p_r}}$ and $\frac{\partial p_i^*}{\partial \tau_i} > 0$ when $\mu > 0$.

6.5 Solution algorithms for the second phase problem

Let us consider a general formulation of the MNL demand model:

$$P(n=i)=\begin{cases} e^{U^i-\beta\cdot p_i}\Big/\left(e^{U^0}+\sum\limits_{i\in A}e^{U^i-\beta\cdot p_i}\right), & for\ n\in A\\ e^{U^0}\Big/\left(e^{U^0}+\sum\limits_{i\in A}e^{U^i-\beta\cdot p_i}\right), & for\ n=0\\ 0, & for\ n\notin A\end{cases} \tag{6.38}$$

and

$$\sum_{i=0}^{N}P(n=i)=1, \tag{6.39}$$

where A is the set of all available time window options, β is the sensitivity parameter and U^i is the initial utility of the customer, p_i is the price for time window option i. The second phase problem can be formulated as a non-linear multivariable programming problem without constraints:

$$\text{Max: } f:=\sum_{i\in A}P(n=i)\cdot(p_i-C_i), \tag{6.40}$$

where C_i is the cost for time window option i, and this already takes into consideration opportunity cost. In order to find p_i to maximise f, we calculate gradient matrix and in order to prove its maximum, we also calculate its Hessian matrix in equation (6.41).

$$g(p_i)=\begin{pmatrix}\frac{\partial f}{\partial p_1}\\ \vdots\\ \frac{\partial f}{\partial p_N}\end{pmatrix};\quad H(\mathbf{p})=\begin{pmatrix}\frac{\partial^2 f}{\partial p_1^2} & \cdots & \frac{\partial^2 f}{\partial p_1\partial p_N}\\ \vdots & \ddots & \vdots\\ \frac{\partial^2 f}{\partial p_N\partial p_1} & \cdots & \frac{\partial^2 f}{\partial p_N^2}\end{pmatrix}, \tag{6.41}$$

where f has a strong relative maximum at stationary point $\mathbf{p}^{(0)}$ if $H(\mathbf{p}^{(0)})$ is negative definite. Stationary point is where $g(p_i)=0$;

Theorem 1. *Let D_i be the minor of H former from the rows $1,2,\ldots,i$ and columns*

$1, 2, \ldots, i$, *then H is negative definite iff*

$$\begin{cases} D_i > 0, & \forall i \in \mathit{even} \\ D_i < 0, & \forall i \in \mathit{odd} \end{cases}. \tag{6.42}$$

Lemma 1.1. *f has a strong relative maximum at stationary point $\mathbf{p}^{(0)}$ if $H(\mathbf{p}^{(0)})$ is negative definite. Stationary point is where $g(p_i) = 0$.*

Proof of Lemma Similar to the proof of Proposition 4 see Appendix F.

6.5.1 Software solver (Exact method)

To compare different solution methods, we first use a mathematical software Maple to solve the maximisation problem. The software solves the problem by taking the first order derivative and let it equal zero. Let us consider a simple numerical example with two options.

Numerical Example 1:

We have two options for a customer, costs are: $C_1 = £5; C_2 = £10$, with sensitivity parameter $\beta = 0.1$. The initial utilities for two options and walk away are:

$$\begin{cases} U^0 = 0.5 \\ U^1 = 1 \\ U^2 = 0.5 \end{cases}$$

The customer prefers option 1 initially. The objective function can be rewritten as:

$$f := \frac{e^{1-0.1x}(x-5)}{1.648721271 + e^{1-0.1x} + e^{0.5-0.1y}} + \frac{e^{0.5-0.1y}(y-10)}{1.648721271 + e^{1-0.1x} + e^{0.5-0.1y}} \tag{6.43}$$

The gradient matrix is:

$$\begin{aligned}\frac{\partial f}{\partial p_1} =& -\frac{0.1e^{1-0.1x}(x-5)}{1.648721271+e^{1-0.1x}+e^{0.5-0.1y}} + \frac{0.1\left(e^{1-0.1x}\right)^2(x-5)}{\left(1.648721271+e^{1-0.1x}+e^{0.5-0.1y}\right)^2} \\ &+ \frac{e^{1-0.1x}}{1.648721271+e^{1-0.1x}+e^{0.5-0.1y}} + \frac{0.1e^{0.5-0.1y}(y-10)e^{1-0.1x}}{\left(1.648721271+e^{1-0.1x}+e^{0.5-0.1y}\right)^2},\end{aligned} \tag{6.44}$$

and

$$\begin{aligned}\frac{\partial f}{\partial p_2} =& \frac{0.1e^{1-0.1x}(x-5)e^{0.5-0.1y}}{\left(1.648721271+e^{1-0.1x}+e^{0.5-0.1y}\right)^2} - \frac{0.1e^{0.5-0.1y}(y-10)}{1.648721271+e^{1-0.1x}+e^{0.5-0.1y}} \\ &+ \frac{0.1\left(e^{0.5-0.1y}\right)^2(y-10)}{\left(1.648721271+e^{1-0.1x}+e^{0.5-0.1y}\right)^2} + \frac{e^{0.5-0.1y}}{1.648721271+e^{1-0.1x}+e^{0.5-0.1y}}.\end{aligned} \tag{6.45}$$

Set equations (6.44) and (6.45) equal to 0, we get:

$$\begin{aligned} p_1 &= 18.53404284, \\ p_2 &= 23.53404284. \end{aligned}$$

We get the optimal prices for the two options, and then we need to prove this point that we get the maximum value.

$$H(p_1, p_2) = \begin{pmatrix} -0.0191 \quad , & -1.1\times10^{-12} \\ -1.1\times10^{-12}, & -0.007 \end{pmatrix},$$

where

$$\begin{aligned} D_1 &= -0.0191 < 0, \\ D_2 &= 0.0191\times0.007 - (1.1\times10^{-12})^2 \approx 0.00013 > 0. \end{aligned}$$

According to Theorem 1, at this optimal price menu we obtain a strong local maximum value.

6.5.2 Modified Newton's method

Newton-based methods are widely used to solve non-linear optimisation problems. An initial starting point is needed:

$$X^k = (x_1^k, x_2^k, \dots x_n^k).$$

Then at each iteration, the method searches for and move to a neighbouring point:

$$X^{k+1} = X^k + \xi. \tag{6.46}$$

Instead of a linear approximation of the function value at this new point, Newton's method uses a quadratic approximation of the function value:

$$f(X^{k+1}) = f(X^k) + g(X^k)^T \xi + \frac{1}{2}\xi^T H(X^k)\xi. \tag{6.47}$$

We want to maximise f with respect to ξ, to decide step size.

$$\frac{\partial f(X^{k+1})}{\partial \xi} = g(X^k) + \frac{1}{2} \times 2H(X^k)\xi = 0. \tag{6.48}$$

We get step size as by solving equation (6.48):

$$\xi = -H(X^k)g(X^k). \tag{6.49}$$

Substitute into equation (6.46), we get:

$$X^{k+1} = X^k - H(X^k)g(X^k) \tag{6.50}$$

Often, in order to satisfy the Wolfe conditions, we use modified Newton method by adding modified step length, equation (6.50) becomes:

$$X^{k+1} = X^k - \lambda H(X^k)g(X^k), \quad \lambda \in (0, 1).$$

The determination of λ, in order to maximise $f(X^{k+1})$ needs future work.

However, Newton's method only has local convergence, during searches, H is sometimes singular or near singular, we modified our search by adjusting H. If H is singular or near singular, update H as:

$$H = H + I.$$

Then new H will be non-singular.

Numerical Example 1:

Previously we solved example 1 by solving $g(\mathbf{x}) = 0$, when x dimension increases, it is hard to solve it directly, we use modified Newton's method:

Starting point: (15,15)
The stopping criteria: $g(\mathbf{x}) < 10^{-6}$
Step length: $\lambda = 0.3$
The solution is: (18.534044; 23.534039)
Running time: 0.3s
The exact solution given by previous solution is (18.53404284, 23.53404284).

6.5.3 Levenberg–Marquardt method

Newton's method requires a good starting point, but LM method is more robust. It's easier to solve a minimisation problem, so our model can be changed to: min $-\sum_{n=1}^{N} P(n = i)(p_i - C_i)$.

We have an initial guess $\mathbf{p}^0$, set $\epsilon_0 = 1$. This initial guess can be of any value. Then set $k = 0$, we first calculate $g_k(\mathbf{p}^k)$, if $g(\mathbf{p}^k) < 10^{-6}$ terminate, else we calculate $H_k(\mathbf{p}^k)$, set $B_k := H_k + \epsilon_k \cdot I$. Then we try to do Cholesky decompose of B, If fail, set $\epsilon_k = \epsilon_k \cdot 4; B_k = H_k + \epsilon_k \cdot I$ until success. Then we denote $B_k = LL^T$, also based on Newton's method, we have $B_k(\mathbf{p}^{k+1} - \mathbf{p}^k) = -g(\mathbf{p}^k)$. We have:

$$Ly = -g(\mathbf{p}^k)$$

Solve y using forward substitution. So we have:

$$L^T \cdot \triangle\mathbf{p} = y$$

Solve $\triangle\mathbf{p}$ backward substitution. Then we update next iteration of solution by:

$$\mathbf{p}^{(k+1)} = \mathbf{p}^k + \triangle\mathbf{p}$$

Then we decide step size for next iteration by calculating $R = \frac{f(\mathbf{p}^k)-f(\mathbf{p}^{k+1})}{Q(\mathbf{p}^k)-Q(\mathbf{p}^{k+1})}$, where Q is the expected value and f is the real value. If $R < 0$, set $\epsilon_k = \epsilon_k \cdot 10$, and start over with $p^{k+1} = p^k$; Else if $R < 0.25$, set $\epsilon_k = \epsilon_k \cdot 10$ and update a^{k+1}; If $R > 0.75$, set $\epsilon_k = \epsilon_k/10$ and update a^{k+1}. Let $k = k + 1$, we repeat the above process until termination criteria satisfied.

Numerical Example 1:

We solve the same problem as before with starting point: (10,10)
The solution is: (18.534044; 23.534039)
Running time: 0.3 s
This solution is the same as previous solution given by commercial software Maple and by modified Newton method.

Numerical Example 2:

Example 1 has only two options, it is easier to solve. With five options or more, commercial software fails to give a solution or takes time to solve the problem.

i	0	1	2	3	4	5
C_i		5	6	5	8	7
U^i	0.2	0.3	0.5	0.3	1	0.5

Table 6.3: Numerical example 2.

Sensitivity parameter for this example is β=0.1.
We start with an initial guess: $\mathbf{p}_0 = (10, 10, 10, 10, 10)$.
The solution is : $\mathbf{p} = (21.91345391554212, 22.91345515023535,$
$21.913455150235343, 24.91345516933855, 23.91345515023535)$.
Solution time: 3 s.

6.5.4 Differential evolution method

Both Newton based method and LM method take time to solve non-linear optimisation problems, as they require either matrix inverse or matrix decomposition calculation. Furthermore, those methods apply a local search. There is possibly more than one local minimum solution, and Newton-based methods may only give a local optimal solution. Differential evolution algorithm is a global search heuristic. This method was first introduced in Storn and Price (1997). It is similar to other evolutionary algorithms with three stages: mutation, crossover and selection. In the mutation stage, each mutated solution is obtained by adding a current solution and the difference of two other solutions in the current generation. The mutation operator is:

$$x_{i,G+1} = x_{l,G} + F \cdot (x_{m,G} - x_{n,G}), \tag{6.51}$$

where l, m, n are indexes of the current generation and are randomly chosen for every iteration. F works as a control parameter to control the amplification of the difference component; it is in the range of 0 to 2. In the experiment, we used 0.5. In the crossover stage, the current generation solution vector changes several components with the mutated vector generated in the mutation stage. In the selection stage, the best mutated vector and crossover vector, in terms of objective values, are chosen to be in the next generation.

Numerical Example 2:

We solve the same problem as in LM method with 5 options. Control parameter is set to be 0.5. The maximum generation number is set to be 1000.

The solution is : $\mathbf{p} = (21.91345391554212, 22.91345515023535, 21.913455150235343, 24.91345516933855, 23.91345515023535)$.
Solution time: 0.1 s

This solution is the same as given by LM method, but with much shorter computational time. Also, this method is a global search method. In this research, differential evolutionary method will be used to solve the pricing problem.

To summarise, optimisation software can only solve problems to optimal of small size such as the one with two options. Differential evolution method generates the same solution as Levenberg–Marquardt method but in a shorter time. Differential evolution method will be applied to solve the second phase problem in this chapter.

6.6 Numerical experiments and results

In the numerical experiment, we test the performance of the online booking system with a sample customer pool of size 100 (mentioned in Section 6.2). Travel distance between locations is real-road distance. Customer requests are drawn from this pool repeatedly. We consider a homogeneous arrival pattern of customers. The average inter-arrival time is set to be 7.2 minutes, and the system is run for 12 hours each day, starting at 6:00 and closing at 18:00. This means that, on average, 100 customers arrive every day. The weight parameters $\boldsymbol{\beta}$ in the MNL demand model in section 6.1 are assumed to be [0.01, -0.50, -0.10, -0.20, 0.01, -0.50], with 0.01 corresponds to walk way, -0.50 is for 8:00-10:00, -0.10 is for 10:00-12:00, . . . , -0.50

corresponds to 16:00-18:00 respectively. These parameters are initially set based on experiences with customer preference, where customers prefer 8:00-10:00 and 16:00-18:00 the most as those time windows are off-work time. Furthermore, 10:00-12:00 is the least preferred time window for example. A numerical example demonstrating the meaning of the specified parameters is shown in Appendix G. Once the system starts running online, real-life data can be collected and stored in a database. Then those parameters can be learned and updated regularly.

The weight of the opportunity cost is tuned to 0.3 for the best performance in reducing emissions. If the weight of opportunity costs was set too high, the effect of improving profits would not be obvious, and the system would always try to sell slots that are closer to expiration even if they have higher emissions levels. On the other hand, a small weight of opportunity costs would lead to overfilling of future beneficial slots and waste those slots close to expiration; it would also reduce the system capacity as well as profit.

There are 20 engineers working each day. Each engineer has one vehicle. All vehicles start and end their daily work at the depot. In order to give a threshold value for comparison purposes, the total amount of emissions and profits are computed for a fixed price policy where customers are served based on their primary demand. We simulate the system for one month (30 days) to compute the emissions and profits for both the case of applying dynamic pricing and the case of applying fixed price policy. We simulate the choice of customer decision based on the MNL model to mimic real life. Hence, there are uncertainties. By taking the total and average of 30 days simulations, the results are more representative for long term. Different scenarios are considered in the experimental study, including the effects of the customers' perception, different customer classes (different price sensitivity). The weight of opportunity cost are also considered when making dynamic pricing decisions.

The computational results are shown in Table 6.4. The average computational time per customer is 10.3 second. This is acceptable for an online real-life booking system. Compared to fixed pricing (FP) policy, dynamic pricing policy consistently reduces the average emissions cost (even under the worst scenario) by 6.7 percent. The higher the customer values this service, the more DP could save on CO_2 emissions. DP starts to increase profitability when customers value the service more than its fixed price; and profit increases as reservation price increases because there is a negative relationship between reservation price and the probability of walkaway (proof in Appendix H). However, DP is observed to lead to a higher percentage of customer walkaway. To measure the effects of different customer classes, we fixed the reservation price to 50 to rule out the effects of customer walk away and only consider the effect of price sensitivity. The results show that DP achieves better results with less price sensitive (0.2) customers. To test the performance of DP with different weights of opportunity cost, the price sensitivity and reservation price are set to be 0.5 and 50 respectively. When the weight of opportunity cost increases, the pricing decision is made more on filling up slots rather than on shifting demand towards the lower emissions slot. The results demonstrate that the more weight on opportunity cost, the less reduction in emissions and the less improvement in profit. Overall, the performance of DP, compared to FP, is robust and a reduction in emissions and improvement in profits are observed.

Scenarios		Cost/Customer			Profit/Customer			Walkaway %	
		DP	FP	Diff.%	DP	FP	Diff.%	DP	FP
Reservation price	10	2.8	3.0	6.7	7.0	7.0	0	36.7	18.3
	15	3.5	4.1	14.6	7.8	5.9	32.2	8.7	1.7
	20	3.4	4.5	24.4	8.6	5.5	56.4	1.5	0.1
	50	3.3	4.6	28.3	8.7	5.4	61.1	0	0
Price sensitivity	0.2	3.3	5.1	35.3	12.9	4.9	163.3	0	0
	0.5	3.3	4.6	28.3	8.7	5.4	61.1	0	0
	0.8	3.7	4.4	15.9	7.4	5.6	32.1	0	0
Weight of opportunity cost	0.3	3.3	4.6	28.3	8.7	5.4	61.1	0	0
	0.6	3.9	4.1	4.9	7.5	5.9	27.1	0	0

Table 6.4: One-month simulation results of online pricing system.

6.7 Summary

This chapter developed a new mixed-integer linear programming model to solve the green vehicle routing problem with time window constraints and time-dependent travel time, and solved it using parallel variable neighbourhood search metaheuristics with a dynamic programming algorithm to optimise start travel time. Compared with exact method branch-and-bound which gave the global optimal solutions, the proposed solution method performed well with an average one percent gap to global optimal solutions and 332.69 times faster in CPU times. The pricing model was solved by differential evolution method and this method outperformed mathematical software and Newton types of search methods.

An innovative online pricing system with our proposed solution method for delivery services booking with rolling mechanisms was built and tested through a one-month simulation study to show the influence of dynamic pricing strategy versus the fixed price strategy. Each customer request was processed by the system in 10.3 second on average. Several simulation studies were carried out to test the robustness of the DP policy under various scenarios. Overall, the performance of DP, compared to FP, was more robust and the reduction in emissions and improvement in profits were observed.

This chapter extended the study in Chapter 4 and modelled customers demand using a discrete choice model. Future work will look at different customer demand model or different pricing policies to shift customers demand towards greener delivery options, this will be the focus of Chapter 7.

7 Incentives sharing pricing policy for green delivery time window booking

In this chapter, we consider a different setting to that considered in the previous chapters, in that previously when customers entered the website to purchase a service they chose one delivery time window. In this chapter customers have the option to choose either the standard two-hour delivery time window option or to be more flexible and participate in the green delivery program, where the customer chooses a two-hour time window together with several alternatives and the least-emissions-cost time window option will be assigned to this customer. To reward customers being flexible and caring about the environment, the cost savings, comparing to all customers who choose standard delivery, will be distributed to those customers who do not get their preferred time window option. This pricing policy needs no extra knowledge of customer demand modelling; it is a simple but highly applicable pricing policy.

We are still interested in scheduling a home attended service provider's operations. Comparing to a non-attended delivery problem, customers of attended services need to agree on a delivery time window. This is because service providers need to enter customer's house to do the job, e.g. installation or maintenance. For safety reasons, the customer is required to be physically present at the time of service. For commodities delivery business, home attended delivery is also common as the commodities may be food that perishes quickly or may be electronic equipment that is valuable. The delivery time window is normally one to four hours, within which the delivery person will arrive at the customer's doorstep (Agatz et al., 2013). For example, Sainsbury.com provides a one-hour delivery slot with differentiated prices. The narrower the time window, the more satisfied the customers will feel as it provides more certainty but the higher the cost will be for the retailer as it limits flexibility to shuffle customers around (Punakivi et al., 2001).

The models related to transportation decisions in home attended delivery are vehicle routing problem with time windows (VRPTW), dynamic vehicle routing problem (DVRP) and periodic vehicle routing problem (PVRP) (Agatz et al., 2008). VRPTW studies the problem to decide the routings to visit a set of customers, which is the same as in traditional vehicle routing problem, but the arrival time at each customer site must fall into a time window agreed with customers beforehand. In DVRP, customer requests arrive during operations and real-time routing adjustment is required. PVRP finds many applications in real life that deals with customers that require service or goods delivery periodically and decisions should be made on delivery routes for multiple periods considering the delivery frequency. For installation and maintenance service providers, the requests are not periodically arriving, and normally customers will send their requests a few days in advance so there is no need for real-time routing solutions. VRPTW will be studied as customers' orders are collected and scheduled all together as a deterministic problem.

Papers in the literature focused on either the time windows management problems or pricing problems separately. Though both are powerful tools in revenue management to reduce costs and improve profitability, research gaps are seen in the combination of the two powerful tools. In this research, the integrated time windows management and pricing problem will be studied in the context of home-attended service delivery.

7.1 Problem description

We study a problem where a company sends engineers to customer sites to provide maintenance and installation services. The company divides each day into several time windows. Typically, each customer makes a request for the service through a website or telephone at least one day before the service day. The customer service centre will ask the customer's preferred time window and offer the customer to participate in the green delivery program (see Table 7.1). The green delivery program is that instead of booking a two-hour time window as usual, the customer also provides a number of available time windows that are also suitable and the company will inform the customer the exact time window on the day of service delivery like DPD's business model (`www.dpd.co.uk`). So even if the customer chooses the green delivery program, a two-hour time window is guaranteed and the customer will not feel unsatisfied by uncertainty caused by wider time window. However, the customer may not get the initially preferred time window, while incentives will be given for customers who are flexible and participate in the green delivery option on the day of service delivery. The incentives work as a compensation for those customers who do not get the preferred delivery options. The incentives pricing problem is solved using a simple pricing policy which does not need prior knowledge of sophisticated customer demand formulation. The total cost savings as compared to following customers' initial preferences are fairly awarded to those customers who choose the green delivery option and do not get their initial preferred time windows.

	Standard Delivery Program	**Green Delivery Program**
Request	Specific one-hour time window (e.g. Monday 8am-10am)	Specific one-hour time window and several alternatives
Actual delivery	Two-hour time window	Two-hour time window
Preference guaranteed?	Yes	No
Confirmation of service time and price	Order in-take	On the day of actual delivery
Price	Fixed price	Fixed price minus dynamic incentives

Table 7.1: Comparing the standard delivery program and the green delivery program.

To deliver the services, each engineer is driving a vehicle, typically a van with the necessary tools. The goal is to allocate customers to engineers in a cost-saving manner. The average travel speeds are fluctuating during the day because of the traffic conditions, so the problem can be considered as a vehicle routing problem with time windows extended to capture different travel times between customers at different times of a day, rather than the traditional vehicle routing problem. Meanwhile, for those customers who provide multiple time windows, the problem becomes more complicated and is a multiple time windows VRPTW. To further demonstrate the benefits of the green delivery program, a simple routing problem with time windows is solved and the solutions are demonstrated in Figure 7.1. The standard delivery program generates a route where the vehicle must visit each customer site by the specified time window. However, customers who participate in the green delivery program are more flexible, and the resulting route has a shorter distance and delivery time. Furthermore, the computational efforts needed for the green delivery program are less, as the time windows are wider or even there are no time window constraints for customers who specify the whole day as the available period. After the optimal route solutions are computed, the delivery windows are determined and then communicated to those customers in the green delivery program.

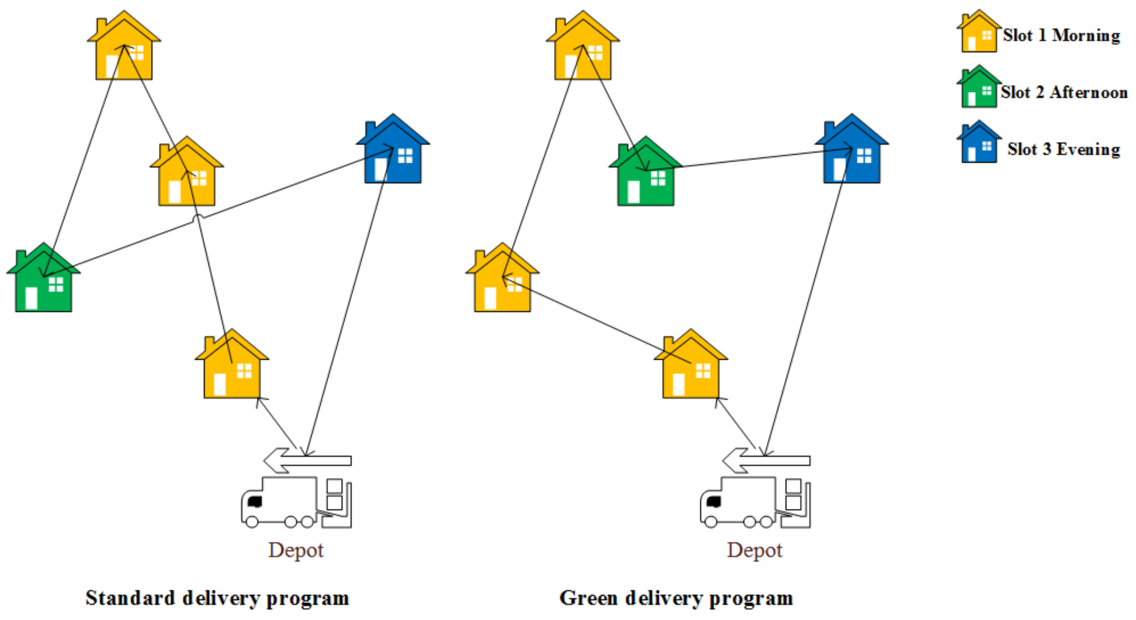


Figure 7.1: Routing solutions of the standard delivery program and the green delivery program.

Traffic conditions have a significant influence on CO_2 emissions, and so the level and cost of emissions are time-dependent. Though the level of emissions is also related to other factors such as vehicle load and road characteristics, this chapter assumes homogenous vehicles and focuses on the timing of travel to take into account the different congestion conditions. Based on the traffic pattern of the area, a day is divided into several time slots such that travel speed so as the emissions level within a time slot can be considered the same.

This study assumes that all information is known at the time of planning which is in the morning of every day and hence solves a deterministic vehicle routing problem. This is normal as the latest service booking time is at least one day before actual service delivery happens. The decisions on vehicle routes are made to minimise the total amount of CO_2 emissions, or fuel consumptions as the two are linearly related (Lichty, 1967). The solution framework is shown in Figure 7.2.

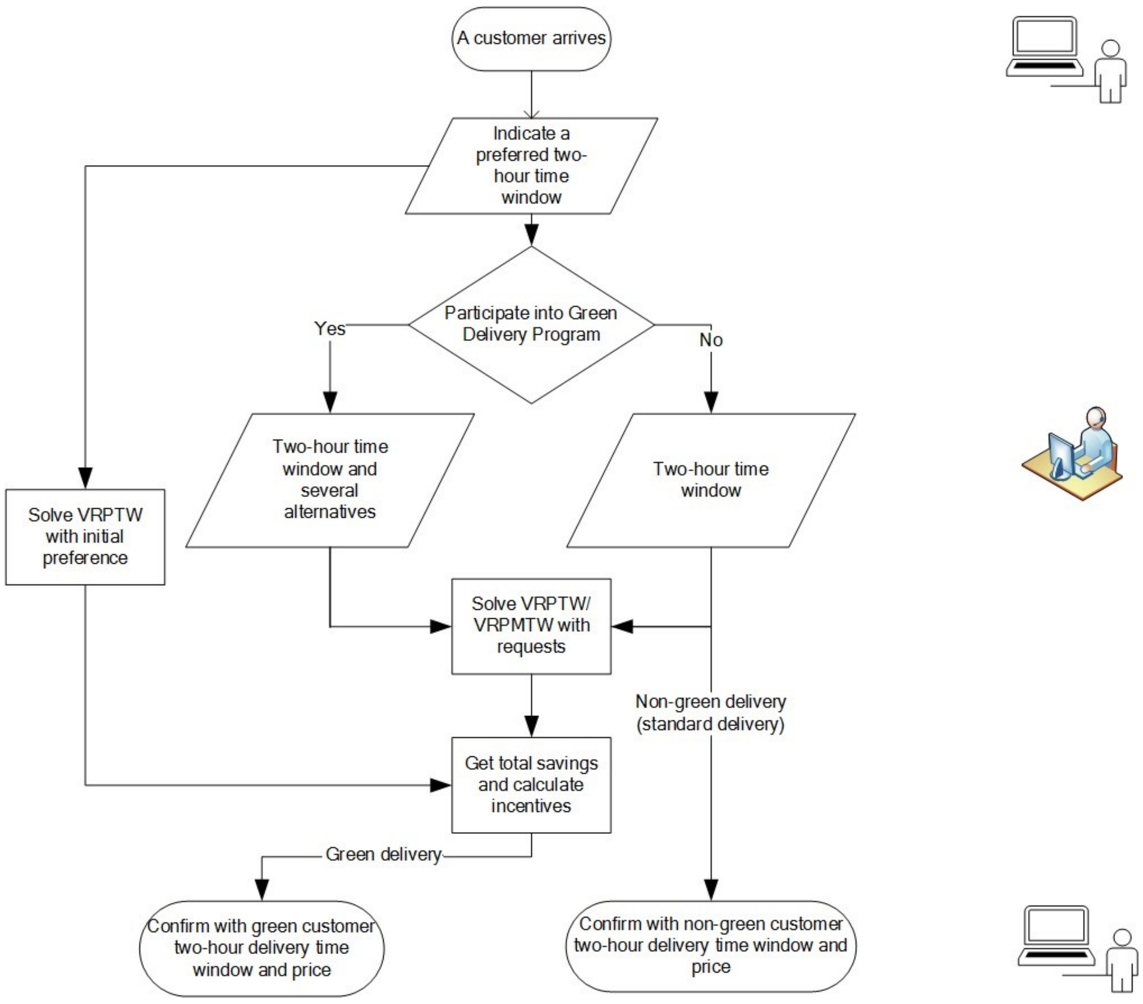

A customer arrives
Indicate a preferred two-hour time window
Participate into Green Delivery Program
Yes
No
Two-hour time window and several alternatives
Two-hour time window
Solve VRPTW with initial preference
Solve VRPTW/ VRPMTW with requests
Non-green delivery (standard delivery)
Get total savings and calculate incentives
Green delivery
Confirm with green customer two-hour delivery time window and price
Confirm with non-green customer two-hour delivery time window and price

Figure 7.2: Solution Framework.

7.2 Scenario 1: available time windows in the same day

We first consider a situation where if a customer participates in the green delivery program, then all the available time windows are on the same day. This scenario exists and is applicable for some urgent services where flexibility exists but normally in one day. Those time windows could be adjacent or not. For example, a customer originally picks 8:00am-10:00am and provides another two available time windows: 2:00pm-4:00pm, 4:00pm-6:00pm. Let us denote the number of time windows for customer i as nT_i, and in the previous example, $nT_i = 2$ and the first window has a width of two hours and the second one has four-hour width (2:00pm-4:00pm and 4:00pm-6:00pm are connected). The problem of interests can be formulated as a multiple time windows VRP problem.

Index:

i, j: index of customers; $i, j = 0$ for the depot;

k : index of vehicles;

t : index of time slots;

l : index of time windows options;

nT_i: index of the total number of time windows options.

Parameters:

n : total number of customers;

K : total number of vehicles available;

K_j : set of vehicles/engineers that has the skill to do job at customer j;

M : a big positive number;

T : total time of the planning horizon;

S : total number of time slots;

d_{ij} : distance of travelling from i to j;

s_i : service time at customer i; and $s_0 = 0$;

b_{il}, e_{il} : begin time and end time of the lth time window of customer i; and b_0, e_0 are start and end times of the working period for engineers;

L_t : start time of time slot t;

v_t : travel speed in time slot t in km/h;

c_{kt}: emissions rate of vehicle k in time slot t in kg/km. [2]

Variables:

[2]We consider homogenous vehicle in this chapter, $c_{kt} = c_t$ for all k, for generic purpose we model it as c_{kt}

C_{ijk} : the amount of emissions (in kg) generated by travelling from i to j by vehicle k;

$$x_{ijk} = \begin{cases} 1, \text{ if vehicle } k \text{ travel from customer } i \text{ to } j \\ 0, \text{ otherwise} \end{cases};$$

$$\varphi_{il} = \begin{cases} 1, \text{ if customer } i \text{ is assigned to his/her } l\text{th time window} \\ 0, \text{ otherwise} \end{cases};$$

u_i : time point when a vehicle starts travelling to customer i, for $i = 1, \dots, n$;

U_k : time point when vehicle k travelling back to depot;

w_i : time point when the service at customer i starts, for $i = 1, \dots, n$;

W_k : time point when vehicle k gets back to depot;

τ_{ij} : travel time from customer i to customer j;

$$z1_{it} = \begin{cases} 1, \text{ if travelling to customer } i \text{ starts in time slot } t \\ 0, \text{ otherwise} \end{cases}, \text{ for } i = 1, \dots, n;$$

$$z2_{it} = \begin{cases} 1, \text{ if travelling to customer } i \text{ ends in time slot } t \\ 0, \text{ otherwise} \end{cases}, \text{ for } i = 1, \dots, n;$$

$$Z1_{kt} = \begin{cases} 1, \text{ if vehicle } k \text{ travelling back to depot starts in time slot } t \\ 0, \text{ otherwise} \end{cases};$$

$$Z2_{kt} = \begin{cases} 1, \text{ if vehicle } k \text{ travelling back to depot ends in time slot } t \\ 0, \text{ otherwise} \end{cases}.$$

Minimize:

$$\sum_{i=0}^{N} \sum_{j=0, i \neq j}^{N} \sum_{k=1}^{K} C_{ikj} \tag{7.1}$$

Subject to:

$$\sum_{i=0, i \neq j}^{n} \sum_{k \in K_j} x_{ijk} = 1, \forall j = 1, \dots, n \tag{7.2}$$

$$\sum_{i=0,i\neq j}^{n} x_{ijk} = \sum_{i=0,i\neq j}^{n} x_{jik}, \forall j = 1,\ldots,n; k = 1,\ldots,K \tag{7.3}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{K} x_{0jk} \leq K \tag{7.4}$$

$$\sum_{i=1}^{n}\sum_{k=1}^{K} x_{i0k} = \sum_{j=1}^{n}\sum_{k=1}^{K} x_{0jk} \tag{7.5}$$

$$\sum_{i=1}^{n} x_{i0k} \leq 1,\ \forall k = 1,\ldots,K \tag{7.6}$$

$$\sum_{i=1}^{n} x_{0ik} \leq 1,\ \forall k = 1,\ldots,K \tag{7.7}$$

$$b_{il} + M\left(\varphi_{il} - 1\right) \leq w_i \leq e_{il} + M\left(1 - \varphi_{il}\right), \forall i = 1,\ldots,n; l = 1,\ldots,nT_i \tag{7.8}$$

$$b_0 \leq W_k \leq e_0, \forall k = 1,\ldots,K \tag{7.9}$$

$$w_i - u_j + M\sum_{k=1}^{K} x_{ijk} \leq M - s_i, \forall i = 1,\ldots,n; j = 1,\ldots,n; i \neq j \tag{7.10}$$

$$w_i - U_k + Mx_{i0k} \leq M - s_i, \forall i = 1,\ldots,n; k = 1,\ldots,K \tag{7.11}$$

$$w_j \geq u_j + \tau_{ij} + M\left(\sum_{k=1}^{K} x_{ijk} - 1\right), \forall i = 0,\ldots,n; j = 1,\ldots,n; i \neq j \tag{7.12}$$

$$W_k \geq U_k + \tau_{i0} + M\left(x_{i0k} - 1\right), \forall i = 1,\ldots,n; k = 1,\ldots,K \tag{7.13}$$

$$u_j \geq L_t + T\left(z1_{jt} - 1\right), \forall j = 1,\ldots,n; t = 1,\ldots,S \tag{7.14}$$

$$U_k \geq L_t + T\left(Z1_{kt} - 1\right), \forall k = 1,\ldots,K; t = 1,\ldots,S \tag{7.15}$$

$$u_j \leq L_{t+1} + T\left(1 - z1_{jt}\right), \forall j = 1,\ldots,n; t = 1,\ldots,S \tag{7.16}$$

$$U_k \leq L_{t+1} + T\left(1 - Z1_{kt}\right), \forall k = 1,\ldots,K; t = 1,\ldots,S \tag{7.17}$$

$$u_j + \tau_{ij} \geq L_t + T\left(z2_{jt} + \sum_{k=1}^{K} x_{ijk} - 2\right), \forall i = 0,\ldots,n; j = 1,\ldots,n; i \neq j; \\ t = 1,\ldots,S \tag{7.18}$$

$$U_k + \tau_{i0} \geq L_t + T\left(Z2_{kt} + x_{i0k} - 2\right), \forall i = 1,\ldots,n; k = 1,\ldots,K; t = 1,\ldots,S \tag{7.19}$$

$$u_j + \tau_{ij} \leq L_{t+1} + T\left(2 - z2_{jt} - \sum_{k=1}^{K} x_{ijk}\right), \forall i = 0, \ldots, n; j = 1, \ldots, n; i \neq j;$$

$$t = 1, \ldots, S \tag{7.20}$$

$$U_k + \tau_{i0} \leq L_{t+1} + T\left(2 - Z2_{kt} - x_{i0k}\right), \forall i = 1, \ldots, n; k = 1, \ldots, K; t = 1, \ldots, S \tag{7.21}$$

$$\sum_{t=1}^{S} z1_{jt} = 1, \forall j = 1, \ldots, n \tag{7.22}$$

$$\sum_{t=1}^{S} Z1_{kt} = 1, \forall k = 1, \ldots, K \tag{7.23}$$

$$\sum_{t=1}^{S} z2_{jt} = 1, \forall j = 1, \ldots, n \tag{7.24}$$

$$\sum_{t=1}^{S} Z2_{kt} = 1, \forall k = 1, \ldots, K \tag{7.25}$$

$$\tau_{ij} \geq \frac{d_{ij}}{v_t} + M\left(\sum_{k=1}^{K} x_{ijk} + z1_{jt} + z2_{jt} - 3\right), \forall i = 0, \ldots, n; j = 1, \ldots, n; i \neq j;$$

$$t = 1, \ldots, S \tag{7.26}$$

$$\tau_{i0} \geq \frac{d_{i0}}{v_t} + M\left(x_{i0k} + Z1_{kt} + Z2_{kt} - 3\right), \forall i = 1, \ldots, n; t = 1, \ldots, S; k = 1, \ldots, K \tag{7.27}$$

$$\tau_{ij} \geq L_{t+1} - u_j + \frac{\left[d_{ij} - v_t\left(L_{t+1} - u_j\right)\right]}{v_{t+1}} + M\left(\sum_{k=1}^{K} x_{ijk} + z1_{jt} + z2_{jt+1} - 3\right),$$

$$\forall i = 0, \ldots, n; j = 1, \ldots, n; i \neq j; t = 1, \ldots, S \tag{7.28}$$

$$\tau_{i0} \geq L_{t+1} - U_k + \frac{\left[d_{i0} - v_t\left(L_{t+1} - U_k\right)\right]}{v_{t+1}} + M\left(x_{i0k} + Z1_{kt} + Z2_{kt+1} - 3\right),$$

$$\forall i = 1, \ldots, n; t = 1, \ldots, S; k = 1, \ldots, K \tag{7.29}$$

$$C_{ijk} \geq c_{kt} \cdot d_{ij} + M\left(x_{ijk} + z1_{jt} + z2_{jt} - 3\right), \forall i = 0, \ldots, n; j = 1, \ldots, n; i \neq j;$$

$$k = 1, \ldots, K; t = 1, \ldots, S \tag{7.30}$$

$$C_{i0k} \geq c_{kt} \cdot d_{i0} + M\left(x_{i0k} + Z1_{kt} + Z2_{kt} - 3\right), \forall i = 0, \ldots, n; k = 1, \ldots, K;$$

$$t = 1, \ldots, S \tag{7.31}$$

$$C_{ijk} \geq c_{kt} \cdot v_t \left(L_{t+1} - u_j\right) + c_{kt+1} \cdot \left[d_{ij} - v_t \left(L_{t+1} - u_j\right)\right] + M(x_{ijk} + z1_{jt} + z2_{jt+1} - 3), \forall i = 0, \ldots, n; j = 1, \ldots, n; i \neq j; k = 1, \ldots, K; t = 1, \ldots, S \tag{7.32}$$

$$C_{i0k} \geq c_{kt} \cdot v_t \left(L_{t+1} - U_k\right) + c_{kt+1} \cdot \left[d_{i0} - v_t \left(L_{t+1} - U_k\right)\right] + M(x_{i0k} + Z1_{kt} + Z2_{kt+1} - 3), \forall i = 1, \ldots, n; k = 1, \ldots, K; t = 1, \ldots, S \tag{7.33}$$

$$\tau_{ij} \leq M \sum_{k=1}^{K} x_{ijk}, \forall i, j = 0, \ldots, n; i \neq j \tag{7.34}$$

$$0 \leq \sum_{t=1}^{S} t \cdot z2_{jt} - \sum_{t=1}^{S} t \cdot z1_{jt} \leq 1, \forall j = 1, \ldots, n \tag{7.35}$$

$$0 \leq \sum_{t=1}^{S} t \cdot Z2_{kt} - \sum_{t=1}^{S} t \cdot Z1_{kt} \leq 1, \forall k = 1, \ldots, K \tag{7.36}$$

and the nonnegativity and binary constraints.

The objective equation (7.1) of the model is to minimise the total amount of CO_2 emissions generated by travelling along the links. In this model, CO_2 emissions rate is the same as in Chapter 6. Constraints (7.2) and (7.3) indicate that there is exactly one vehicle going into each customer node and the same vehicle leaves the customer node (each customer node is served only once). Constraints (7.4) and (7.5) mean that at most K vehicles can leave the depot and the same number will travel back to the depot at the end. Constraints (7.6) and (7.7) require that each vehicle can at most leave and return to the depot once, and reuse of vehicle is prohibited. Constraints (7.8) and (7.9) specify that the services for each customer i must be in one of the nT_i available time windows, and the vehicles/engineers must return to depot in the working period for the engineers. Constraints (7.10) and (7.11) ensure that a vehicle can start travelling to the next customer only after finishing the task at the current customer. These also work as sub-tour elimination constraints. There may be a gap (idle time) between task finish time and the start time of travelling to the next customer. Constraints (7.12) and (7.13) require that tasks at customer sites can start only after the vehicle arrives there. Constraints (7.14) through (7.21) determine travel start and end times. Constraints (7.22) through (7.25) mean that the start or end of each travel must fall into only one time slot. Constraints (7.26) through (7.29) calculate the travel time between two nodes with time-dependent travel speed. Constraints (7.30) through (7.33) determine the amount of CO_2 emissions associated with each link. Constraint (7.34) bounds the travel time between two nodes to zero if they are not linked. Constraints (7.35) and (7.36) assume that the maximum gap between the start and end of travelling on any link is one time slot and the minimum is zero time slot, i.e., travelling between any two nodes will not span into three different time slots.

The model is tested and solved on a few small examples with 10 customers each, and every customer has an initial preferred time window with two-hour width. To validate the effect of the proposed green delivery program policy, all ten customers will participate in this program and will provide another alternative time window with a four-hour width ($nT_i = 2$). The alternative time window is non-overlapping with the initial preferred time window. All test cases are solved to optimality by Xpress optimiser on the Loughborough high-performance computer (HPC) cluster, HYDRA with 20 cores 64-bit Intel Xeon CPU and 64 GigaBytes of memory. The optimal solutions metrics are demonstrated in Table 7.2. The problems are solved using 20 threads in parallel. Initially, we tried to solve the problems on a personal computer (PC) with 64-bit Windows operating system, eight GigaBytes of RAM and an Intel i5 processor with quad-cores. However, those problems cannot be solved to optimality within 20 hours, after which the program will terminate due to insufficient system memory left. In the solution for each problem, the service starting times clearly indicate which time window is selected for each customer.

Instance	Emissions (kg)		Improvement (%)	Travel Time (mins)		Improvement (%)	Distance (km)		Improvement (%)	Idle Time (mins)		Improvement (%)	Vehicles		Improvement (%)
	nT=1	nT=2		nT=1	nT=2		nT=1	nT=2		nT=1	nT=2		nT=1	nT=2	
1	47.31	35.16	25.68	570.16	327.27	42.6	239.16	173.04	27.65	2129.84	372.73	82.5	3	1	66.67
2	42.51	28.33	33.36	483.3	287.46	40.52	215.77	141.28	34.52	1216.7	412.54	66.09	2	1	50
3	55.05	49.48	10.12	601.69	471.3	21.67	281.82	245.61	12.85	2098.31	228.7	89.1	3	1	66.67
4	36.14	27.26	24.57	576.2	246.92	57.15	183.65	134.55	26.74	1123.8	453.08	59.68	2	1	50
5	45.9	35.24	23.23	946.05	333.63	64.73	229.61	174.06	24.19	753.95	366.37	51.41	2	1	50
6	49.47	45.11	8.81	667.7	426.39	36.14	247.69	223.94	9.59	1032.3	273.61	73.49	2	1	50
7	31.51	27	14.3	549.55	250.79	54.36	158.05	134.92	14.64	1150.45	449.21	60.95	2	1	50
8	49.3	37.19	24.56	588.34	359.6	38.88	251.47	185.48	26.24	2111.66	340.4	83.88	3	1	66.67
9	44.32	34.43	22.32	645.79	349.42	45.89	222.23	172.6	22.33	1054.21	350.58	66.74	2	1	50
10	43.47	36.93	15.05	405.55	354.09	12.69	215.45	183.18	14.98	1294.45	345.91	73.28	2	1	50
Average			**20.2**			**41.46**			**21.37**			**70.71**			**55**

Table 7.2: Xpress results on test instances with customer size 10.

Procedures taken to improve the performance of the solver are shown in Appendix I. Customers participated in the green delivery program and provide flexibility will help reduce the total emissions by 20.20%. Furthermore, engineers' total travel time, travel distance and idle time are also reduced by 41.46%, 21.37% and 70.71% respectively. Idle time reduction is mainly achieved by reducing the number of engineers scheduled and tighten the schedule of each engineer, so a 55% reduction in the number of vehicles used/engineers scheduled is also observed. This helps the company to save staff cost as well.

For the pricing part, the total emissions cost savings of these ten customers and the number of customers being shifted are summarised in Table 7.3. Not all customers provided flexibility will be shifted, and some of them can still get their initial preference. The savings will be evenly distributed to customers who have been moved to another time windows. The average savings for ten customer cases are 1.40.

Instance	Savings (money unit)	Shifted Customer	Incentives (money unit)
1	12.148917	9	1.349879667
2	14.181729	6	2.3636215
3	5.570394	7	0.795770571
4	8.8803	7	1.268614286
5	10.66131	5	2.132262
6	4.359695	6	0.726615833
7	4.505483	5	0.9010966
8	12.105757	8	1.513219625
9	9.890963	6	1.648493833
10	6.542719	5	1.3085438
Average			**1.400811772**

Table 7.3: Incentives pricing of the ten instances.

7.3 Metaheuristic method for scenario one

Because the model cannot be solved on normal PCs in reasonable times, the new self-adaptive simulated annealing algorithm introduced in Chapter 5 is applied in this chapter to solve scenario one model. To deal with multiple time windows constraints, we amended the algorithm slightly. The first amendment is that when building initial solutions, the tasks are ranked in ascending order of time window width. In multiple time windows instances, the tasks are ranked by the biggest time window width of all available time windows. Furthermore, tasks are scheduled according to the biggest time window in the initial solution. Another change to the algorithm is that when checking the feasibility of a neighbouring solution, if start service of each customer is within any of the multiple time windows, then the solution is feasible.

To test the performance of the algorithm, we solved the same ten instances that the software solver solved before on the same PC mentioned before. The results are summarised in Table 7.4. The SA solution is an average solution of 10 repetitions as there is randomness in the search algorithm. The algorithm is able to produce solutions with an average of 2.68% gap, which is close to optimal. Furthermore, while the optimisation software cannot solve those problems within hours on local PC, the average solution time of the SA algorithm is 44.93 seconds. Based on industrial needs, the proposed SA algorithm is fast and the solution quality is very high.

Instance	Xpress solution (optimal solution)	ASA solution	Time (s)	Gap%
1	35.16	37.135	48.5	5.61
2	28.33	29.321	42.1	3.50
3	49.48	49.988	50.6	1.03
4	27.26	27.695	46.4	1.60
5	35.24	35.24	35.7	0.00
6	45.11	45.216	48.6	0.23
7	27.00	27.311	40.9	1.13
8	37.19	38.194	47	2.69
9	34.43	36.17	34.6	5.06
10	36.93	39.131	54.9	5.97
Average			**44.93**	**2.68**

Table 7.4: Comparisons between Self-adaptive SA with Xpress solver on the ten Senario 1 instances.

7.4 Scenario 2: available time windows in different days

For normal or standard service booking, a customer probably has another available time window on a different day of his/her initial preferred time window. Those time windows could be adjacent or not as in Scenario 1. Another example is a customer originally picks 8:00am-10:00am on Monday and provides another available time window 4:00pm-6:00pm on Tuesday. Different from the example in Scenario 1, these time windows are on different days. The customer may provide time windows on two different days, but the schedules and routing need to be solved every day for the current day. It is not feasible to solve the problems for two days as there will still be customers arriving for the next day, and optimal solutions cannot be obtained until the cut-off time of the next day, which is the end of the current day. With available time windows on two days, it is more beneficial to schedule the customer on the current day to reduce engineer idle time unless the current day yields much more emissions. To tackle this point, each shifted flexible customer, which means shifted to the next available time window on another day, is associated with an opportunity cost measured by the value of wasted utility time, where wasted utility time of customer i is defined as:

$$\varsigma_i = a\left(s_i + \overline{\tau_i}\right)$$

where a is a predetermined parameter indicating the weight of the wasted time, s_i is the service time and $\overline{\tau_i}$ is the average travel time to customer i, and

$$\overline{\tau_i} = \frac{\sum_{j \in N} d_{ij}}{n\overline{v}}$$

where set N is the set of n nearest neighbour of customer i and $\overline{v}$ is the mean velocity of the day. Wasted utility time is calculated using the total time dealing with a customer, which means if shifting this customer away, that particular amount of time which would be used to fill up engineers' idle time is wasted.

For customers who participate into the green delivery program (flexible customers), they will be guaranteed to be scheduled in one of their available windows, as they already provide flexibilities and it is not wise to refuse those customers especially at the last time window they choose. If flexible customers cannot be served or not beneficial to be served in the current day, they will definitely get their service on one of the alternative available dates. On the other hand, non-flexible customers only provide one-time window and they may or may not get their service depending on capacity. However, they will get the response sooner than flexible customers which are before or at the beginning of their preferred day. To decide which non-flexible customers to reject, the estimated potential profits loss of all the non-flexible customers are calculated by:

$$\pi_i = R_i - \frac{\sum_{j \in N} d_{ij}}{n} \cdot \overline{c}$$

where R_i is the revenue that a customer may bring, which is linearly related to the service time and $\bar{c}$ is the average emissions rate. This potential profit loss is treated as another type of opportunity cost.

The problem of interests can be formulated as an extended version of the Scenario 1 model with overbookings and the objective function is changed to maximise profit minus opportunity cost and additional constraints are added to deal with overbookings.

Index:

i, j : index of customers; and $i, j = 0$ for the depot;

k : index of vehicles;

t : index of time slots;

NF: set of non-flexible customers;

FF: set of flexible customers that still have available time windows in the future time;

FM: set of flexible customers that have been shifted before and today is the only available time window left.

Parameters:

n : total number of customers;

K : total number of vehicles available;

b_i, e_i : begin time and end time of the time window of customer i; and b_0, e_0 start and end times of the working period for engineers;

π_i : estimated average profit of customer i;

ς_i : opportunity cost of customer i;

R_i : revenue customer i brings which is linearly related the service time.

Variables:

C_{ijk} : the amount of emissions (in kg) generated by travelling from i to j by vehicle k;

$$x_{ijk} = \begin{cases} 1, \text{ if vehicle k travel from customer } i \text{ to } j \\ 0, \text{ otherwise} \end{cases};$$

w_i: time point when the service at customer i starts, for $i = 1, \ldots, n$.

Maximum:

$$\sum_{i=0}^{N} \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} R_i x_{ijk} - \sum_{i=0}^{N} \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} C_{ijk} - \sum_{i\in NF} \pi_i \left(1 - \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} x_{ijk}\right) - \sum_{i\in FF} \varsigma_i \left(1 - \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} x_{ijk}\right) \tag{7.37}$$

Subject:

$$\sum_{i=0,i\neq j}^{n} \sum_{k=1}^{K} x_{ijk} = 1, \quad \forall j \in \mathbf{FM} \tag{7.38}$$

$$\sum_{i=0,i\neq j}^{n} \sum_{k=1}^{K} x_{ijk} \leq 1, \quad \forall j \in \mathbf{FF,\ NF} \tag{7.39}$$

Constraints (7.3) - (7.7)

$$b_i \leq w_i \leq e_i, \quad \forall i = 1\ldots, n; \quad l = 1, \ldots nT \tag{7.40}$$

Constraints (7.9) – (7.36), and the nonnegativity and binary constraints.

The objective (7.37) is to maximise the total profit ($\sum_{i=0}^{N} \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} R_i x_{ijk} - \sum_{i=0}^{N} \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} C_{ijk}$) minus the opportunity cost of rejecting a non-flexible customer $\left(\sum_{i\in NF} \pi_i \left(1 - \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} x_{ijk}\right)\right)$ and minus the opportunity cost of shifting a flexible customer to a future time window rather than taking the near-to-expire time window $\sum_{i\in FF} \varsigma_i \left(1 - \sum_{j=0,i\neq j}^{N} \sum_{k=1}^{K} x_{ijk}\right)$). Constraints (7.38) are for flexible customers who have been shifted and have only one available time window left. It indicates that each customer in this group is served once and only once by one vehicle on the current day. Meanwhile, constraints (7.39) are for non-flexible customers and flexible customers who still have available time window options in the future, so the customer can be served once by one vehicle or not be served. Constraints (7.40) are the standard time window constraints. Different from Scenario 1 model, in one day, each customer only has one time window option, but flexible customers have multiple time windows across different days. The rest constraints are the same as in Scenario 1 model.

Scenario 2 model is first tested on a small example as in Scenario 1 for one day and the above formulation was solved by Xpress optimiser. In this instance, there are five new customers who are flexible and have one time window for this day and alternatives for other days, five new customers who are non-flexible and five existing customers who used to be flexible but have been shifted to this day with one time window too. In total, there are 15 customers booked for this day. The estimated capacity of one engineer is 10. So, the number of vehicles is one in this instance. The service times are all set to be 30, and the revenue from customers is all 30 unit. We first set the time value in the function of ς_i to be 0.1. The optimal solutions represented on X-Y coordinates are shown in Figure 7.3. As there is overbooking, not all customers are accepted in the optimal solution. When $\varsigma_i = 0.1$, less weights are put onto the opportunity cost of serving flexible customers today rather than another day, so all rejected customers are within this group. 12 customers are served. If increase ς_i to 0.5, opportunity cost takes too much weight this time, resulting all non-flexible customers being rejected. Same as before, 12 customers are served. Taking some value in between such as 0.3 will be an appropriate weight for opportunity cost. One customer from the flexible group is shifted to another day, and two non-flexible customers are rejected. Furthermore, the detailed performance metrics of the optimal routes are shown in Table 7.5. $\varsigma_i = 0.3$ gives the best routes among the three in the case of emissions, travel distance and travel time.

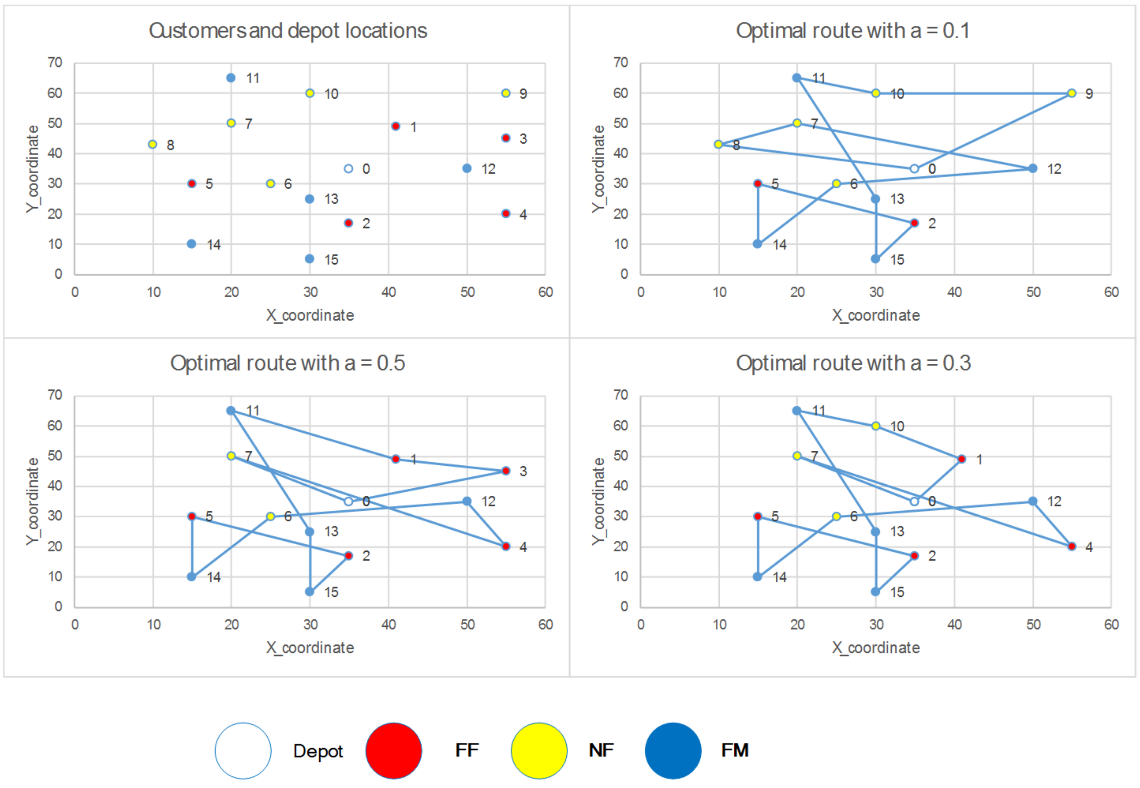


Figure 7.3: Xpress results regarding different weights of opportunity cost demonstration on X-Y coordinates.

Instance	Customer served	Customer rejected	Emissions	Travel Time	Distance
a = 0.1	12	3	61.9919	557.3326	306.1329
a = 0.3	12	3	**59.0449**	**535.4196**	**291.031**
a = 0.5	12	3	63.18748	578.1264	312.3845

Table 7.5: Xpress results with respect to different weights of opportunity cost (best result highlighted).

7.5 Solution algorithm for Scenario 2 problem

A modified self-adaptive simulated annealing algorithm is applied to Scenario 2 problem. To decide which customers to be included in the initial solution, all the tasks that belong to **FF** and **NF** are ranked according to their potential cost and time window width, the one with a smaller cost and a narrower time window will be inserted into initial solution first. Ten test instances are randomly generated and with five customers for each category. Time value a is set to be 0.3. All test cases are solved by Xpress optimiser on HPC first to get the optimal solution, and then solved by the modified SA. The comparison results are demonstrated in Table 7.6. SA performs well in terms of solution quality with an average optimality gap of 1.61%, furthermore the average solution time is 16 seconds.

Instance	Emissions (Xpress optimal)	No. customers (Xpress)	Emissions (ASA)	No. customers (ASA)	CPU Time	Gap
1	56.83	12	56.83	12	18.339	0.00
2	66.08	13	68.672	13	21.984	3.93
3	70.08	13	71.726	13	22.501	2.35
4	64.80	13	67.8	13	12.048	4.63
5	52.60	12	55.06	12	11.136	4.69
6	68.71	13	68.85	13	14.847	0.21
7	51.47	12	51.56	12	10.585	0.17
8	50.54	13	50.54	13	16.251	0.00
9	48.48	13	48.54	13	18.821	0.12
10	40.49	12	40.49	12	12.882	0.00
Average					**15.939**	**1.61**

Table 7.6: Comparisons between Self-adaptive SA with Xpress solver on the ten Senario 2 instances.

7.6 Simulation experiments on large size problem

A customer pool of 600 customers are generated for the simulation experiments of seven days. Artifical test cases are used instead of real life instances because the real life cases have rather long time windows e.g. a four hours time window which is not suitable to be grouped and does not make sense. To better study the effects of time windows flexibility, artificial data sets are created. Assuming there are three engineers on duty each day, 45 customers can be served approximately. We consider a scenario with overbooking where the company will accept 60 customers each day, among them there is a 50-50 percent chance that the customer will participate in the green delivery program. The geographical data are taken from the combination of all Solomon benchmark case locations of 600 customers (some customer may be at the same locations) and shown in Figure 7.4. The test case is a mix of Scenario 1 and Scenario 2, where customers who are flexible have time windows on the same day or on different days. The customers not being served are either non-flexible customers being rejected or flexible customers being shifted to alternate days. Initially, we consider for flexible customers $nT = 3$ [3] and the first time window has a width of two hours and the second two have four-hour width. Simulations of seven days are run on ten instances, and the results are summarised in Table 7.7.

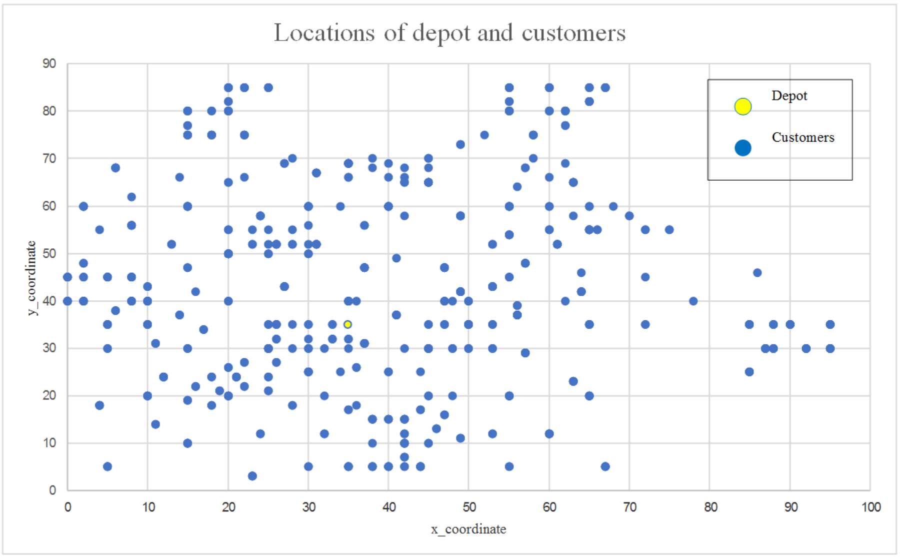


Figure 7.4: Locations of 600 customers and a depot for simulation experiments.

[3]The number of time windows customers provided are the same in the experiments, so we denote $nT = nT_i, \forall i$

Instances	With green delivery program						Traditional delivery			
	Total emissions (kg)	Average per customer (kg)	Served (no.)	Shifted (no.)	Rejected (no.)	Average incentives (money)	Total emissions (kg)	Average per customer (kg)	Served (no.)	Rejected (no.)
1	1102.79	3.16	349	65	71	1.19	1126.19	3.38	333	87
2	1087.01	3.11	349	80	71	0.84	1097.65	3.31	332	88
3	1094.92	3.16	347	71	73	1.46	1098.14	3.45	318	102
4	1060.44	3.02	351	86	69	2	1155.11	3.51	329	91
5	1082.05	3.12	347	80	73	1.3	1110.67	3.42	325	95
6	1136.71	3.28	347	81	73	0.91	1102.6	3.49	316	104
7	1057.38	3.05	347	76	73	1.46	1094.19	3.37	325	95
8	1129.01	3.22	351	66	69	1.33	1119.99	3.47	323	97
9	1091.91	3.07	356	77	64	1.69	1102.01	3.43	321	99
10	1062.21	3.01	353	85	67	1.6	1120.2	3.39	330	90
Average		**3.12**	**349.7**	**76.7**	**70.3**	**1.38**		**3.42**	**325.2**	**94.8**

Table 7.7: Seven days simulations results on 10 instances where **NF**:**FF**=1:1 and $nT = 3$.

The average emissions cost per customer is 3.12 when introducing the green delivery program and 50% of customers participating in it. However, the average emissions of traditional delivery and time window booking is 3.42 kg. This is an 8.77% reduction in average emissions per customer. Furthermore, with overbookings, the green delivery program is able to serve more customers and so reject fewer customers with an average of 70 rejections comparing to traditional delivery with an average of 95. This delivery system results in a 26.32% reduction in the number of customers refused. On average, the customers who participate in the green delivery program will get an incentive reward of 1.38.

7.6.1 Changes in the environment

The initial results show good potential and benefits of the proposed pricing and green delivery program. This section tests the robustness of the program when there are changes in the environment. Customers may not prefer the green delivery program or may prefer it more than expected, so the ratio of non-flexible customers and flexible ones may be changed. We consider a ratio of 2:1 and 1:2 separately. The results are listed in Table 7.8 and Table 7.9. The percentage reductions in emissions compared to traditional delivery strategy are 4.11%, 8.77% and 12.65% for the ratios of non-flexible customers versus flexible customers 2:1, 1:1 (Table 7.7) and 1:2 correspondingly. As the proportion of flexible customers increases which gives more room for optimisation, the benefits of the green delivery program get enlarged. There is also a positive trend in the percentage reduction in customers rejected, where the reductions are 18.59%, 26.32% and 35.89%. Furthermore, when more customers participate in the green delivery program, an increase in average incentives awarded are also increased, which are 0.99, 1.38 to 1.57.

Instances	With green delivery program						Traditional delivery			
	Total emissions (kg)	Average per customer (kg)	Served (no.)	Shifted (no.)	Rejected (no.)	Average incentives (money)	Total emissions (kg)	Average per customer (kg)	Served (no.)	Rejected (no.)
1	1064.73	3.16	337	42	83	0.9	1076.24	3.27	329	91
2	1142.59	3.4	336	55	84	0.03	1120.62	3.41	329	91
3	1114.02	3.16	352	55	68	1.34	1103.44	3.37	327	93
4	1146.44	3.34	343	38	77	1.41	1130.19	3.5	323	97
5	1119.62	3.29	340	50	80	0.87	1091.33	3.42	319	101
6	1127.04	3.26	346	53	74	1.08	1115.97	3.42	326	94
7	1108.41	3.24	342	52	78	1.46	1118.29	3.46	323	97
8	1135.94	3.34	340	50	80	1.17	1138.23	3.51	324	96
9	1135.6	3.35	339	50	81	0.72	1102.39	3.46	319	101
10	1127.75	3.19	354	51	66	0.94	1109.18	3.32	334	86
Average		**3.27**	**342.9**	**49.6**	**77.1**	**0.99**		**3.41**	**325.3**	**94.7**

Table 7.8: Seven days simulations results on 10 instances where **NF**:**FF**=2:1 and $nT = 3$.

Instances	With green delivery program						Traditional delivery			
	Total emissions (kg)	Average per customer (kg)	Served (no.)	Shifted (no.)	Rejected (no.)	Average incentives (money)	Total emissions (kg)	Average per customer (kg)	Served (no.)	Rejected (no.)
1	1077.57	3.09	349	98	71	0.92	1100.94	3.35	329	91
2	1040.58	2.92	356	106	64	1.57	1074.73	3.39	317	103
3	1047.6	2.9	361	102	59	2.26	1153.9	3.54	326	94
4	1034.9	2.91	356	94	64	1.66	1077.6	3.35	322	98
5	1130.26	3.17	356	103	64	1.33	1156.79	3.56	325	95
6	1029.71	2.81	366	103	54	1.8	1078.86	3.32	325	95
7	1058.28	2.96	358	77	62	2.23	1113.35	3.44	324	96
8	1084.92	2.98	364	96	56	1.64	1109.52	3.41	325	95
9	1059.52	2.89	366	103	54	1.05	1056.14	3.19	331	89
10	1110.54	3.09	359	98	61	1.27	1121.9	3.44	326	94
Average		**2.97**	**359.1**	**98**	**60.9**	**1.57**		**3.4**	**325**	**95**

Table 7.9: Seven days simulations results on 10 instances where **NF**:**FF**=1:2 and $nT = 3$.

The number of available time windows may also affect the performance of the new pricing strategy. We consider for flexible customers $nT = 2$ where the first time window has a width of two hours and the second one has a four-hour width, $nT = 4$ where the first time window has a width of two hours and the others have four-hour width. The results are demonstrated in Table 7.10 ($nT = 2$), Table 7.7 ($nT = 3$) and Table 7.11 ($nT = 4$). It shows that the more time windows the customer provided, the more emissions that can be saved, the percentage savings of emissions are 4.09%, 8.77% and 11.73% for $nT = 2 \ldots 4$ respectively. Similar results are observed for the rejection ratio, the reductions in customers rejected are 14.45%, 26.32% and 36.82%. Meanwhile, the amounts of incentives also improve as the number of flexible customers increases (0.94, 1.38 and 1.55).

Instances	With green delivery program						Traditional delivery			
	Total emissions (kg)	Average per customer (kg)	Served (no.)	Shifted (no.)	Rejected (no.)	Average incentives (money)	Total emissions (kg)	Average per customer (kg)	Served (no.)	Rejected (no.)
1	1158.57	3.44	337	48	83	0.91	1123.59	3.57	315	105
2	1135.14	3.34	340	46	80	0.98	1124.8	3.47	324	96
3	1065.59	3.27	326	69	94	0.96	1131.74	3.47	326	94
4	1112.48	3.25	342	54	78	1.65	1124.39	3.51	320	100
5	1113.59	3.3	337	54	83	0.81	1105.79	3.43	322	98
6	1125.7	3.25	346	54	74	1.15	1105.33	3.43	322	98
7	1095.86	3.19	343	56	77	0.4	1062.84	3.26	326	94
8	1100.31	3.26	338	51	82	0.85	1086.24	3.38	321	99
9	1130.96	3.38	335	52	85	1.5	1183.91	3.61	328	92
10	1006.43	3.08	327	60	93	0.15	1015.31	3.1	327	93
Average		**3.28**	**337.1**	**54.4**	**82.9**	**0.94**		**3.42**	**323.1**	**96.9**

Table 7.10: Seven days simulations results on 10 instances where **NF**:**FF**=1:1 and $nT = 2$.

Instances	With green delivery program						Traditional delivery			
	Total emissions (kg)	Average per customer (kg)	Served (no.)	Shifted (no.)	Rejected (no.)	Average incentives (money)	Total emissions (kg)	Average per customer (kg)	Served (no.)	Rejected (no.)
1	1111.19	3.13	355	83	65	1.54	1134.52	3.49	325	95
2	1085.68	3.04	357	95	63	1.69	1120.58	3.49	321	99
3	1075.63	3.01	357	87	63	1.65	1106.49	3.42	324	96
4	1103.49	3.04	363	98	57	1.21	1124.6	3.37	334	86
5	1059.02	2.91	364	90	56	1.98	1094.85	3.4	322	98
6	1069.04	3	356	99	64	1.6	1085.94	3.45	315	105
7	1070.82	3.04	352	90	68	1.67	1151.73	3.47	332	88
8	1093.21	3.04	360	104	60	1.27	1096.12	3.4	322	98
9	1106.65	3.02	367	87	53	1.69	1103.08	3.42	323	97
10	1051.48	2.88	365	99	55	1.16	1041.59	3.2	326	94
Average		**3.01**	**359.6**	**93.2**	**60.4**	**1.55**		**3.41**	**324.4**	**95.6**

Table 7.11: Seven days simulations results on 10 instances where **NF**:**FF**=1:1 and $nT = 4$.

7.7 Summary

In this chapter, a new pricing strategy was studied where customers who participate in the green delivery program share the incentives. The green delivery program allows the customers to provide alternative time windows, different from traditional delivery where only one time window is selected. The incentives are determined by the amount of emissions saved compared to traditional delivery. When customers provide more than one time window, the decision problem for customer assignment and engineers routing becomes a multiple time windows VRP problem which attracts little attention in the literature. A new self-adaptive simulated annealing algorithm was developed in Chapter 5 and applied in this chapter. Two different scenarios of problems were considered when the alternative time windows are either on the same day or on different days. Furthermore, in Scenario 2, overbooking is considered as well. To test the performance of self-adaptive SA, ten instances each are constructed for both scenarios, and solved by Xpress optimiser as well. The results showed that for Scenario 1 the gap is 2.68% and for Scenario 2 is 1.61%. The solution quality is acceptable, meanwhile the computational time is much faster with an average of 44.93 s and 15.94 s respectively. However, Xpress optimiser was unable to solve those instances on local machine within hours and 8Gb of memory, so the instances were solved by Xpress optimiser on Loughborough University HPC. Real-life sized instances were randomly generated and simulated for seven working days with both Scenario 1 and Scenario 2 properties. Results showed that the proposed incentive sharing policy could help save emissions and improve the number of customers served when there is overbooking. Changes in the environment were simulated to test the robustness of the new policy. Test results showed that more flexible customers or more time windows provided by each customer could further enlarge the benefits of reducing emissions and reducing the number of customers refused.

In the next chapter, we will conclude the findings of this research and summarise the contributions. Furthermore, future research topics will also be proposed.

8 Conclusions

The objective of the research presented in this thesis has been to apply dynamic pricing policies to shift customer demand and lead them to choose more environmental friendly delivery options in the context of the online service delivery industry, such as a telecommunication company which provides home attended installation and maintenance service. A significant portion of CO_2 emissions comes from road transportation. Emissions from vehicles are affected by travel speed which varies during the day, so emissions levels for different delivery time slots of a day are different. The aim is to find the potential benefits of implementing dynamic pricing in the sense of both emissions reduction and profits improvement.

8.1 Contributions and findings

To understand the state-of-the-art of the field and ensure the originality of the thesis work, a literature review was conducted first focusing on four different research areas related to the thesis. First, since the research interests are in online service delivery with the goal to reduce emissions, the literature on green vehicle routing problems with time windows was reviewed. Most relevant works in the literature acknowledge that emissions are affected by travel speed and that travel speed is a time-dependent variable so as emissions. There are congested periods in a day, during which emissions are much higher. Research gap was seen in multiple time window VRP literature. Little attention has been paid to this topic. By providing multiple time window options, there would be a potential to reduce CO_2 emissions further in this case. The second related research area is how to estimate emissions from vehicles. Several different emissions models were reviewed and compared. The NAEI and the MEET project formulas are the most suitable emissions models for this research based on data availability. The accuracy of the emissions models was testified in Chapter 3. Thirdly, state of the art solution algorithms for VRPTW were reviewed and compared to select the algorithms for use. Tabu search, variable neighbourhood search and variants, simulated annealing and variants have been implemented and compared in this thesis. At last, the literature of revenue management in time window booking was reviewed. There are differentiated time window pricing and dynamic pricing for time windows. There have been some previous works applying revenue management technique to reduce traditional delivery costs such as distance and travel time and smooth demand. To the best of our knowledge, there has been no research applying dynamic pricing technique to reduce emissions of delivery.

CO_2 emissions are affected by travel speed, travel distance, and vehicle details, etc. For calculating emissions from different route plans and for real-life applications to make an impact, a CO_2 emissions calculator was built in Java. The unique features of the system are: 1) automated vehicle details mapping to emissions formulas, which is not provided in any commercial emissions software; 2) tailored speed profiles for different regions in the UK, based on real-world traffic data and the data of driving behaviours of engineers from a field service provider; 3) computing emissions data for every granularity like road segment level, task level, engineer level, daily level and so on. Although fuel consumption data gives companies an idea of the overall emissions associated with each vehicle and engineer, the emissions data gives more detailed information on fuel consumption e.g. per task; 4) Map visualizing (Heat map): the emissions on travels are coloured and benchmarked against different levels of CO_2 emissions. The emissions calculator has the NAEI and the MEET emissions formulas built in, and the results from it were compared with manufacturing data and proved to be relatively accurate. It is also the first emissions calculator that includes vehicle details in emissions formulas to differentiate the emissions calculation. An interesting finding from experiment results is that minimising emissions is not the same as minimising travel time. Conflicts exist between traditional objectives and green objectives. Real life business applications in a telecommunication company were taken as an example to demonstrate the impact of the research.

Three different application situations of service delivery booking were then considered, and the potential benefits of applying dynamic pricing to each of them were investigated, as compared to applying traditional fixed price booking. Customers may arrive anytime during the day and book for services.

In the first situation, customers who book before the day of service are allocated to the time windows they requested. A two-phase solution framework was proposed to deal with customer requests for service on the current day. Customer locations are concentrated in a relatively small geographical area and so travelling times between locations and their differences could be ignored. Each customer request comes with a preferred service time window, but may accept other time windows with certain probabilities which change with prices change. The first phase model for this situation solves the problem for each time window option as a vehicle scheduling problem with the objective to minimise emissions and get the cost of accommodating the customer in the time window. The costs of all time windows are the inputs for the second phase model. The second phase model solves the pricing problem to determine the incentives for each time window option with the objective of maximising the expected profit. A linear probability demand model was considered in this situation. The models in both phases were solved using software optimiser which yielded optimal solutions but was limited to problem size due to long computational time. Simulation experiments were carried out on small size instances to compare the performance of dynamic pricing with fixed price policy. Significant reduction in the amount of CO_2 emissions, as well as significant improvement in the total profits, were observed. Sensitivity tests were carried out, in which the results showed that with less dynamics in the system, i.e., less new coming customers, or lower emissions variation, the reduction in CO_2 emissions and profit improvement would be lower. The patterns of customer arrival were also found to affect the results.

To extend the problem to a general setting with no limits on location distances, routing decisions need to be included in the first phase problem. Vehicle routing problems are NP-hard. Real life problems are not solvable within a reasonable amount of time and within limited memory usage. Metaheuristic methods are often applied to large size problems. The state of art algorithms, including tabu search, variable neighbourhood search and variants, simulated annealing and a new self-adaptive simulated annealing algorithm, were implemented and compared on real-life instances. The results showed that RVNS is more suitable than VND and TS for practical application where quick solutions with reasonable good quality are required. Comparison between traditional SA and the new proposed self-adaptive SA demonstrates that on average improves the average optimal solutions by 12.76%. We have not seen any previous work comparing several algorithms on green VRPTW problem.

With the knowledge of how to calculate emissions and the appropriate solution algorithms, the problem was extended to real life problems in the second situation and the customer demand was considered to follow a more complicated discrete choice model comparing to the linear one. Different from previous booking system in the first situation, it takes a rolling mechanism so all customers face the same number of available time windows. Opportunity costs were also introduced to try to arrange customers to those near-to-expire time windows. The problem was also solved with the two-phase framework, while new mathematical models were proposed for each phase. The first phase model was extended to consider time-dependent travel speed, real-life road distance matrix, and a heterogeneous fleet. A parallel variable neighbourhood search algorithm was used to solve this problem and compared with Xpress solver on small size instances. Meanwhile, a dynamic programming algorithm was proposed to determine the optimal departure times to customer sites. The results showed that the PVNS could solve the problems with an average optimality gap of one percent and get the solution 332.69 times faster than the software solver. The second phase solves a nonlinear programming model. Differential evolution search method was selected to solve this problem as it outperforms mathematical software and other Newton type search methods in the sense of the quality of results and computational time. The whole solution process for dealing with one customer was within seconds, so it is applicable in real life business. Dynamic pricing policy reduced emissions by 6.7% compared to fixed pricing policy. More experiments were carried out to test the impact of changes in the environment, and the results showed that higher valuation of the service would lead to more emissions reductions, but dynamic pricing might lead to an increase in the number of walk away customers. The results also demonstrated the effects of the weight put on opportunity cost when making the pricing decisions. The relationship is negative with more weight of opportunity cost cause less reduction in emissions. Overall, the performance of dynamic pricing, compared to fixed pricing policy, is robust and a reduction in emissions and improvement in profits are observed.

Both linear demand model or discrete choice demand model requires historical data on customer booking behaviours with respect to varying prices to estimate parameters in the models. However, those data are hard to obtain. With limited access to real-life data on this, a new demand-model-free dynamic pricing policy was proposed in the third situation. In this situation, customers book one day in advance and provide an initial preferred time windows together with other available time windows. The problem needs to be solved at the beginning of each day considering all the requests for services on that day, which may be viewed as the first phase vehicle routing and scheduling problem in the first situations extended to a multiple time windows VRP. We built a new mixed integer linear programming model that determines the task schedule with vehicle routes and allocates service time window for each customer simultaneously, rather than in two phases. The total profit is compared with that of the schedule where all customers are served in their preferred time windows. The savings due to flexible time window options are then shared among the customers receiving service in alternative time windows. This integrated model is even more difficult to solve. This problem is novel. There is very little research on multiple time windows VRP in literature, but applications can be found in practice. Procedures to improve Xpress optimiser performance were tested on small test cases. For real-life size test cases, metaheuristics were applied. This pricing strategy was applied on two different scenarios, one is that all the initial preference and available time windows are on the same day, which is the case of urgent service where flexibility may exist but service needs to be completed on the same day. The second scenario is that the initial preference and available time windows can be on different days, so opportunity costs can be included in the model. Opportunity cost is calculated based on the time value of the time window, and a time window which is near to expire has less opportunity cost than a later one. Furthermore, in the second scenario, overbookings was considered, so the decisions of which customer to refuse were included. To measure the performance of this pricing policy, the problems were first solved by Xpress optimiser to get an idea of the percentage savings of CO_2 emissions, comparing to the case where each customer only has one time-window. Considering the complexity of this problem, for large size problem instances, a self-adaptive simulated annealing algorithm was applied.

This research has made a number of significant contributions. One main contribution is a new approach to reducing emissions by applying dynamic/incentive pricing techniques and combining it with time-dependent vehicle routing/scheduling models in a two-phase framework. Using this framework, each customer order is handled as it arrives. Decisions are made, with customer interaction, on the time window to perform the task and on the price. Another main contribution is that we proposed a new incentive sharing pricing policy to reduce emissions in which the task scheduling, vehicle routing and time window choice problems are modelled together as an integrated problem. Such a situation has received little attention in the literature.

8.2 Future works

In this research, we found that there may be conflicts between traditional objectives and CO_2 emissions minimisation objective. Research could be carried out in this direction to investigate the relationship between objectives, as companies want to introduce green logistics into their operations but they are afraid that doing so may cause profit loss and affect other performances. To convince industry to take this approach, a multi-objective VRP model with dynamic pricing may be necessary. The multi-objective function could be treated as a linear combination of all the objectives and solved using standard methods as mentioned in Chapter 5, or each objective is treated as a player and the problem is solved by a cooperative game theoretic approach.

With historical data collection, parameter estimation with machine learning could be applied to the demand model. To measure the accuracy of a demand model, goodness of fit tests could be carried out with the MNL model and real-life data sample as shown in Ben-Akiva and Lerman (1985). Furthermore, dynamic pricing may cause a regret effect, which will change customer choice, and further impact the performance of dynamic pricing and CO_2 emissions. These could be included in future research.

Electric vehicles have recently been introduced into the fleet of companies such as Sainsbury and Royal Mail. Electric vehicles are restricted by the range and only suitable for short trips. The scheduling of electric vehicles during peak hours and traditional vehicles during off-peak hours can help reduce emissions largely. However, introducing electric vehicles will bring additional constraints, such as the total trip length must be below a certain value. Another example is the scheduling of electric cars must also take the charging port location and recharging time constraints into consideration. Moreover, the estimated electricity consumption of electric vehicles also depends on travel speed and other factors, an efficient and accurate way of measuring the electricity consumption can be an interesting topic.

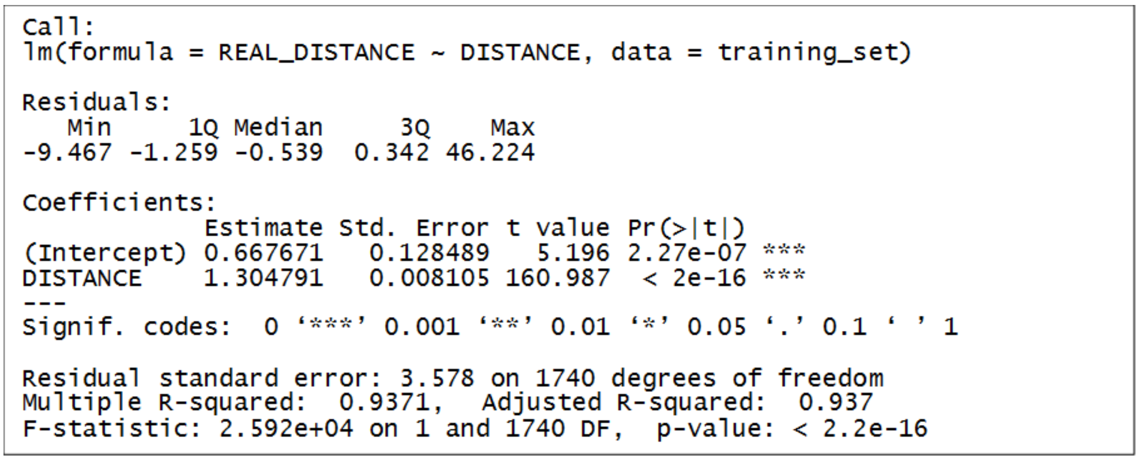
The road network details were not considered in this thesis for the reason that the problem of interests is the delivery of service in a local area, and the travel time and travel distance are not long. For longer distance travelled, there are many different routes and the differences among them are big. It is important to plan beforehand what route to take, and the drivers also tend to follow the plan for long distance trips. However, for short distance journey, when the company accepts a customer's order, they will plan for the future for which they do not have information on the road traffic condition. A reasonable estimate of the travel cost will be acceptable and sufficient. Moreover, the difference between route options is relatively small compared to that in long distance drive. The drivers are also familiar with the road options in the local area, so they may use experiences to choose the road rather than following predetermined road plans which is time-consuming for the company to make. The study may be extended to cases with medium and long travel instances by considering road network.

In this thesis, the revenue management technique applied to manage time window booking is dynamic pricing or incentive pricing, this could be extended to differentiated pricing for each delivery time windows based on the popularity and corresponding emissions levels.

Appendices

A Summary of regressor and R script of linear regression model

A.1 R results



```
Call:
lm(formula = REAL_DISTANCE ~ DISTANCE, data = training_set)

Residuals:
   Min      1Q Median      3Q     Max 
-9.467 -1.259 -0.539  0.342 46.224 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 0.667671   0.128489   5.196 2.27e-07 ***
DISTANCE    1.304791   0.008105 160.987  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.578 on 1740 degrees of freedom
Multiple R-squared:  0.9371,	Adjusted R-squared:  0.937 
F-statistic: 2.592e+04 on 1 and 1740 DF,  p-value: < 2.2e-16
```

A.2 R script

```
#Importing the dataset
dataset = read_xlsx('DISTANCE_MATRIX_DATA_TABLE1.xlsx')
dataset = dataset[,3:4]

library(caTools)
set.seed(123)
split = sample.split(dataset$REAL_DISTANCE, SplitRatio = 2/3)
training_set = subset(dataset,split == TRUE)
test_set = subset(dataset,split == FALSE)

#Train regression model with the Training set
regressor = lm(formula = REAL_DISTANCE ~ DISTANCE,
data = training_set)

#Predicting the Test set results
```

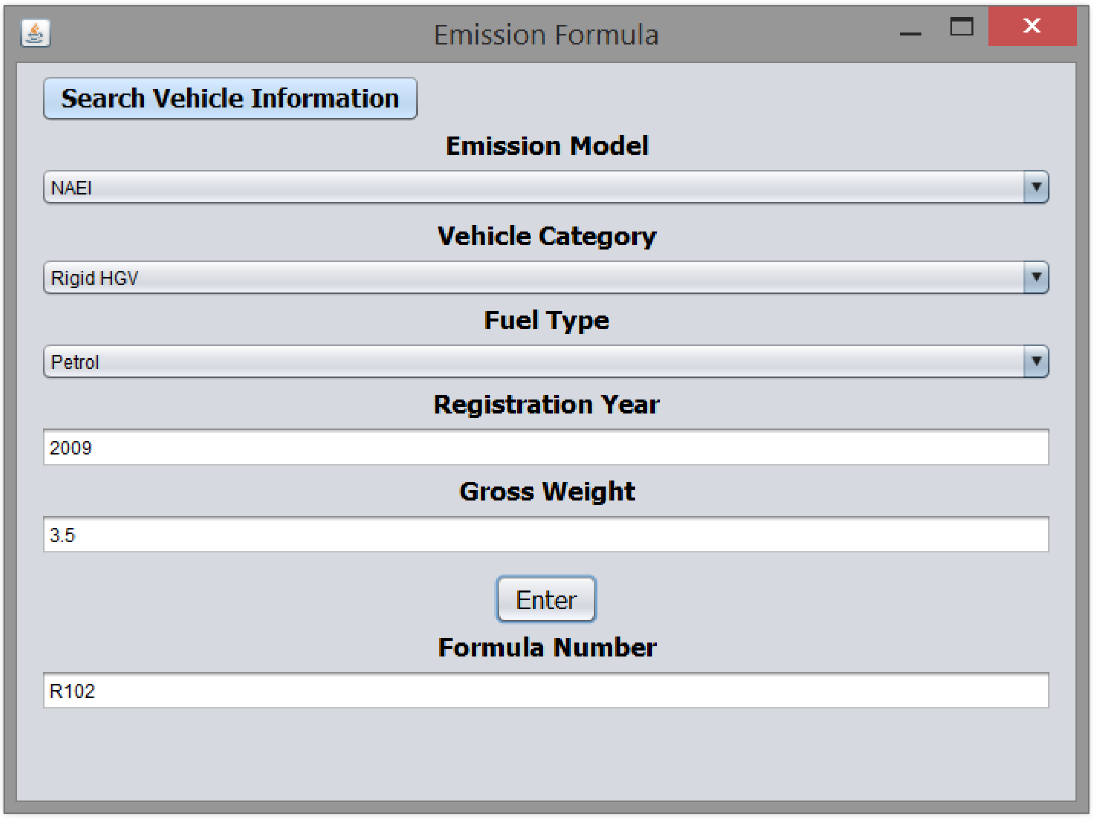
```
REAL_DISTANCE_pre = predict(regressor,newdata = test_set)

#Visualising the Training set results
library(ggplot2)
ggplot()+
geom_point(aes(x = training_set$DISTANCE, y = training_set$REAL_
   DISTANCE),
colour = 'red')+
geom_line(aes(x = training_set$DISTANCE, y = predict(regressor,
   newdata = training_set)),
colour = 'blue')+
ggtitle('Road Distance vs Distance (Training set)')+
xlab('Haversine Distance')+
ylab('Road Distance')

ggplot()+
geom_point(aes(x = test_set$DISTANCE, y = test_set$REAL_DISTANCE),
colour = 'red')+
geom_line(aes(x = training_set$DISTANCE, y = predict(regressor,
   newdata = training_set)),
colour = 'blue')+
ggtitle('Road Distance vs Distance (Test set)')+
xlab('Haversine Distance')+
ylab('Road Distance')
```

B Vehicles mapping to emissions formulas GUI

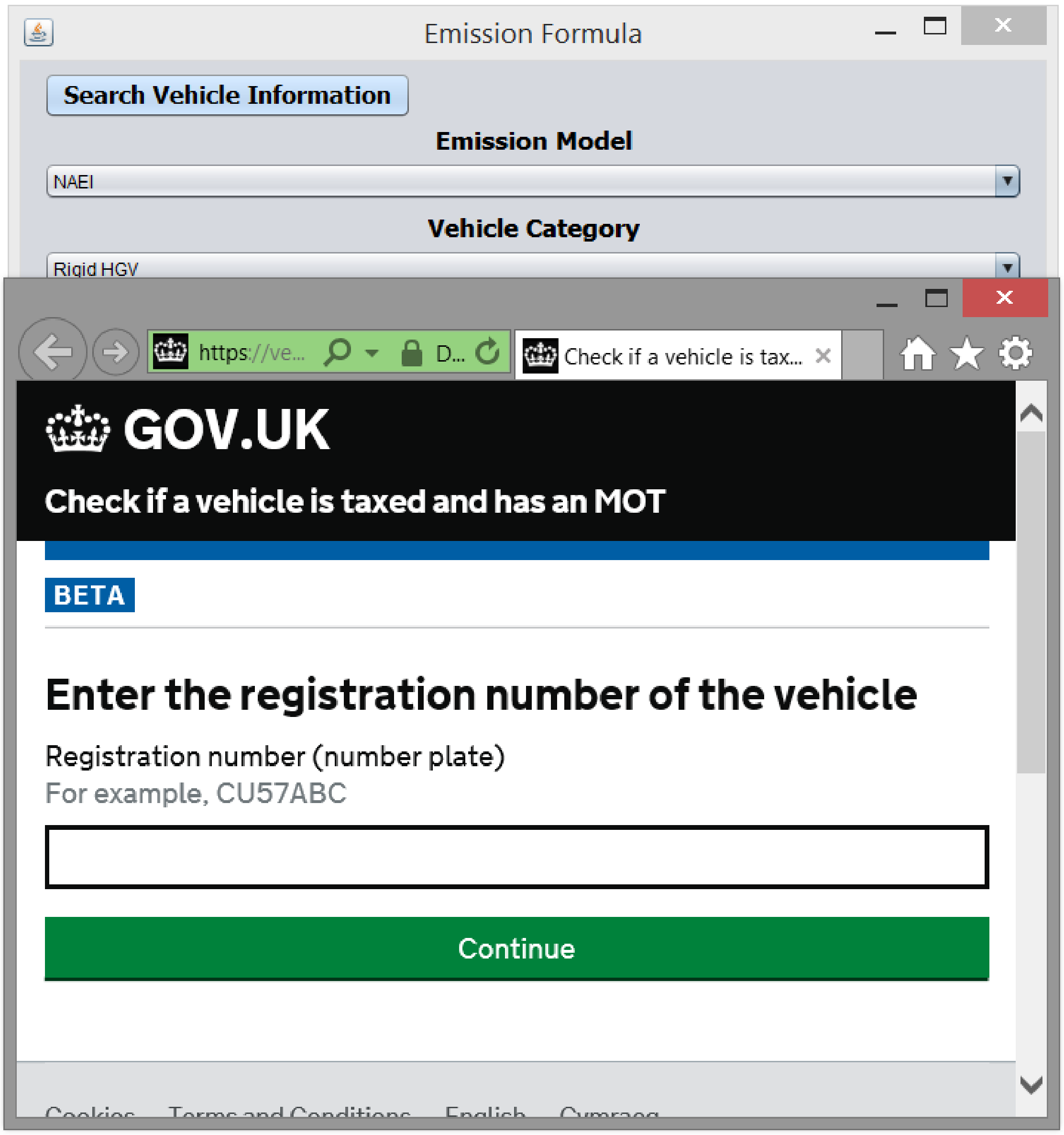
The user interface of the emissions formula mapper is shown in Figure B.1 with the following information entered and press Enter button, the user will get the corresponding emissions formula number.



Emission Model: NAEI
Vehicle Category: Rigid HGV
Fuel Type: Petrol
Registration Year: 2009
Gross Weight: 3.5
Formula Number: R102

Figure B.1: Vehicles mapping to emissions formulas GUI.

If the user only knows the vehicle registration number, he/she can simply press the ‘Search Vehicle Information’ button, and the government website (as demonstrated in Figure B.2) will pop out to help to get the required information.


Emission Formula

Search Vehicle Information

Emission Model

NAEI

Vehicle Category

Rigid HGV

https://ve... D... Check if a vehicle is tax...

GOV.UK

Check if a vehicle is taxed and has an MOT

BETA

Enter the registration number of the vehicle

Registration number (number plate)

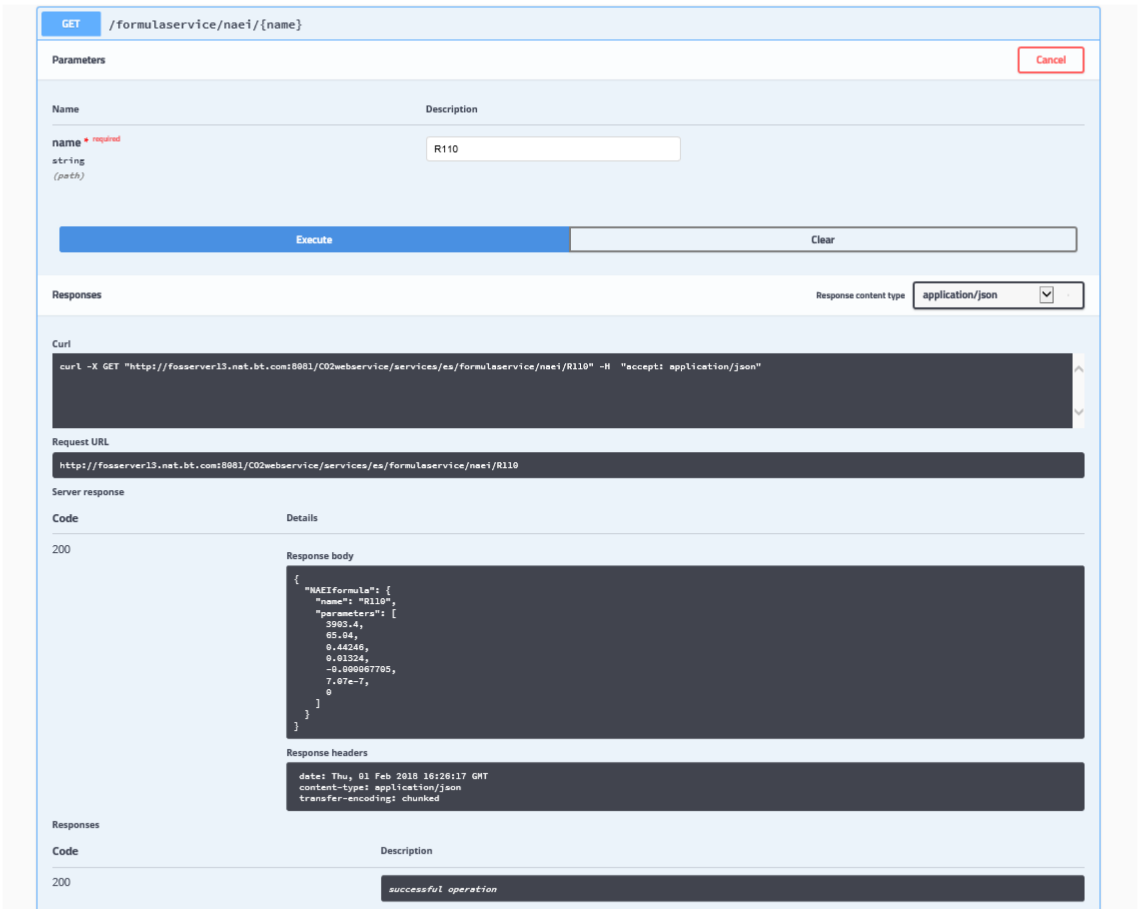
For example, CU57ABC

Continue

Figure B.2: Vehicles mapping to emissions formulas GUI.

C Examples of CO_2 emissions web service

C.1 The NAEI Formulas lookup: Get the NAEI emissions formula R110



R110

Curl

```
curl -X GET "http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/formulaservice/naei/R110" -H  "accept: application/json"
```

Request URL

```
http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/formulaservice/naei/R110
```

Server response

Code	Details
200	Response body

```
{
  "NAEIformula": {
    "name": "R110",
    "parameters": [
      3903.4,
      65.04,
      0.44246,
      0.01324,
      -0.000067705,
      7.07e-7,
      0
    ]
  }
}
```

Response headers

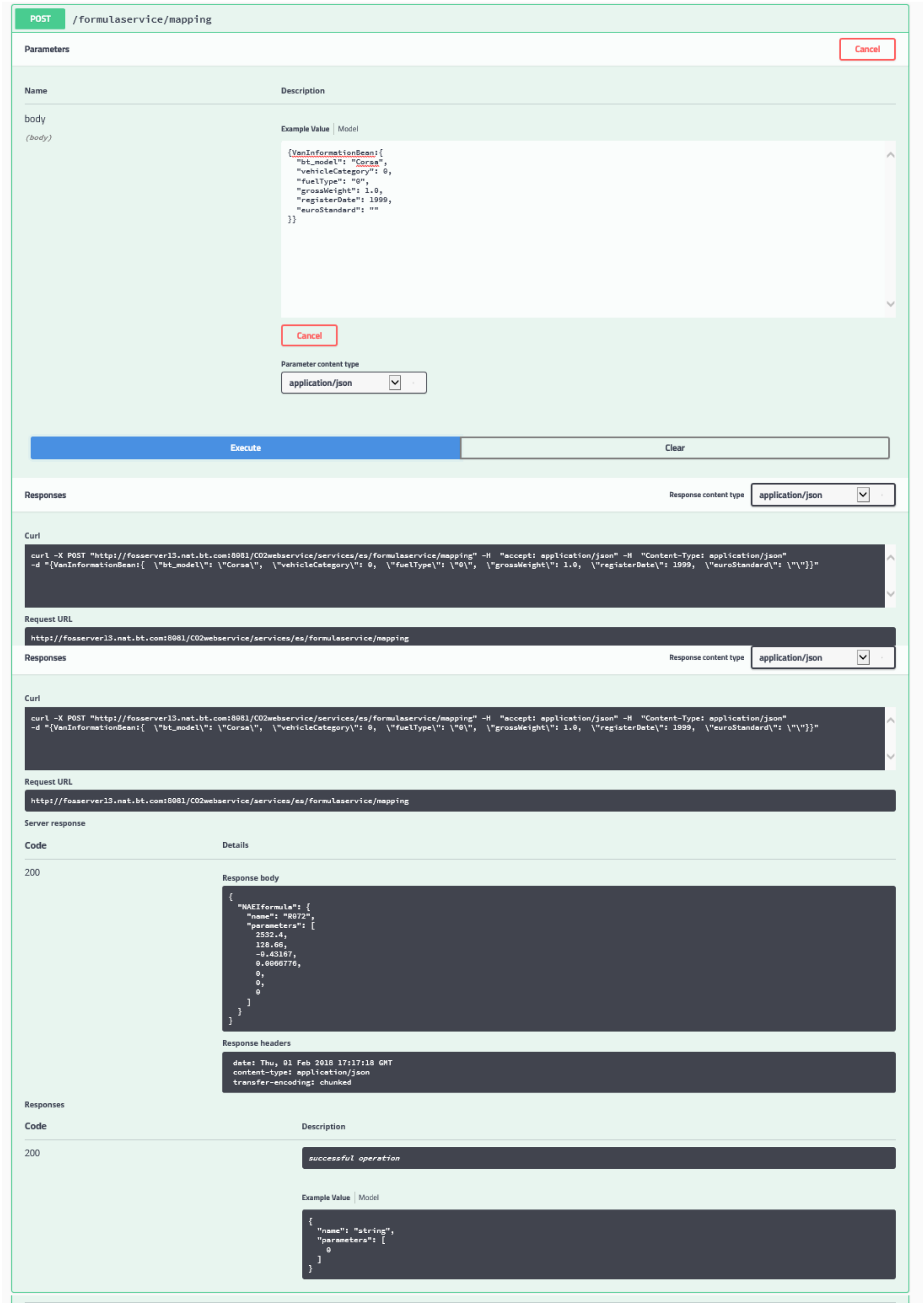
```
date: Thu, 01 Feb 2018 16:26:17 GMT
content-type: application/json
transfer-encoding: chunked
```

Responses

Code	Description
200	*successful operation*

Figure C.1: Get the NAEI emissions formula R110.

C.2 Vehicle specification mapping service: Post vehicle speciation and map it to a corresponding formula



/formulaservice/mapping

Parameters

Name | Description

body *(body)*

Example Value | Model

```
{VanInformationBean:{
  "bt_model": "Corsa",
  "vehicleCategory": 0,
  "fuelType": "0",
  "grossWeight": 1.0,
  "registerDate": 1999,
  "euroStandard": ""
}}
```

Parameter content type: application/json

Responses — Response content type: application/json

Curl

```
curl -X POST "http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/formulaservice/mapping" -H  "accept: application/json" -H  "Content-Type: application/json"
-d "{VanInformationBean:{  \"bt_model\": \"Corsa\",  \"vehicleCategory\": 0,  \"fuelType\": \"0\",  \"grossWeight\": 1.0,  \"registerDate\": 1999,  \"euroStandard\": \"\"}}"
```

Request URL

```
http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/formulaservice/mapping
```

Responses — Response content type: application/json

Curl

```
curl -X POST "http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/formulaservice/mapping" -H  "accept: application/json" -H  "Content-Type: application/json"
-d "{VanInformationBean:{  \"bt_model\": \"Corsa\",  \"vehicleCategory\": 0,  \"fuelType\": \"0\",  \"grossWeight\": 1.0,  \"registerDate\": 1999,  \"euroStandard\": \"\"}}"
```

Request URL

```
http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/formulaservice/mapping
```

Server response

Code | Details

200

Response body

```
{
  "NAEIformula": {
    "name": "R072",
    "parameters": [
      2532.4,
      128.66,
      -0.43167,
      0.0066776,
      0,
      0,
      0
    ]
  }
}
```

Response headers

```
date: Thu, 01 Feb 2018 17:17:18 GMT
content-type: application/json
transfer-encoding: chunked
```

Responses

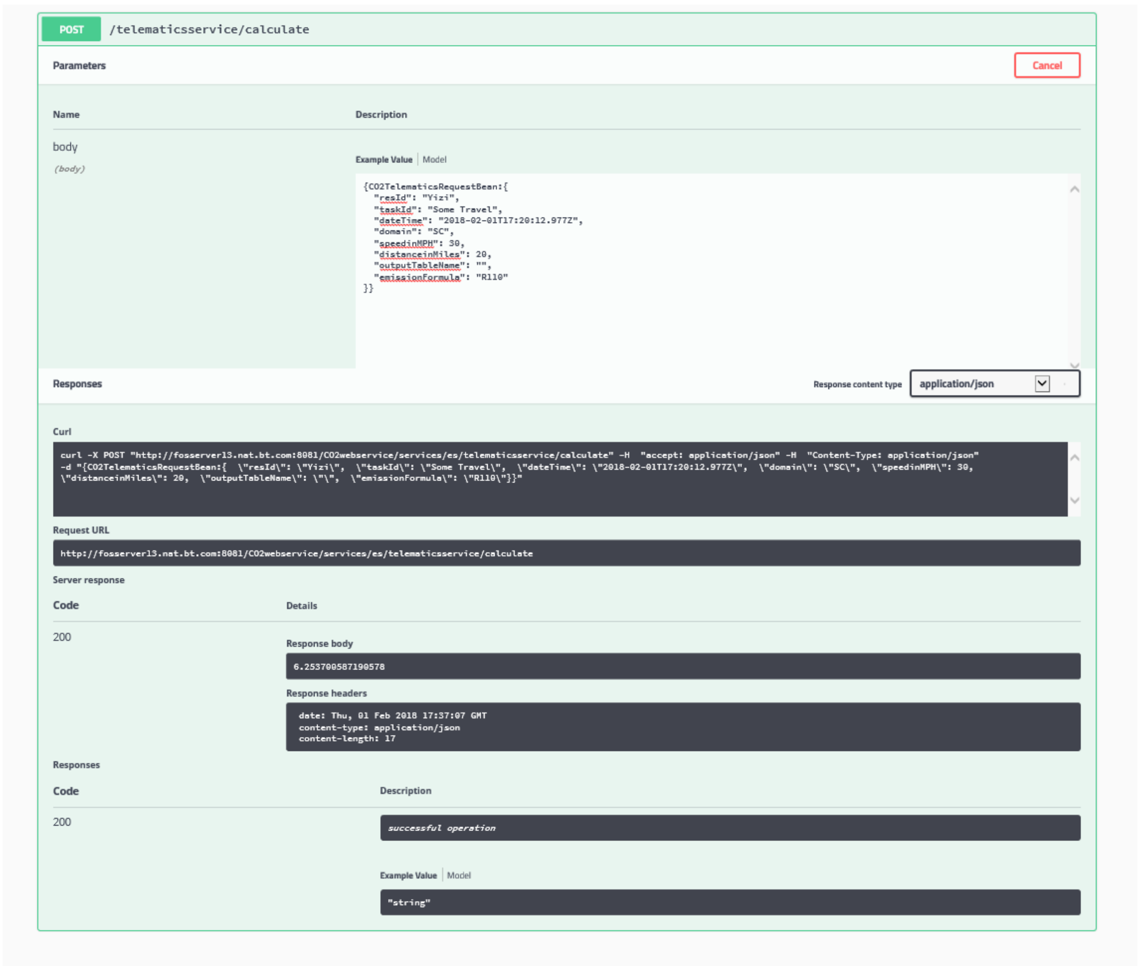
Code	Description
200	*successful operation*

Example Value | Model

```
{
  "name": "string",
  "parameters": [
    0
  ]
}
```

Figure C.2: Post vehicle speciation and map it to a corresponding formula.

C.3 Emissions calculation service: Calculate CO_2 emissions with telematics data



/telematicsservice/calculate

Parameters

Name: body (body)

Description

Example Value | Model

```
{CO2TelematicsRequestBean:{
  "resId": "Yizi",
  "taskId": "Some Travel",
  "dateTime": "2018-02-01T17:20:12.977Z",
  "domain": "SC",
  "speedinMPH": 30,
  "distanceinMiles": 20,
  "outputTableName": "",
  "emissionFormula": "R110"
}}
```

Responses

Response content type: application/json

Curl

```
curl -X POST "http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/telematicsservice/calculate" -H  "accept: application/json" -H  "Content-Type: application/json"
-d "{CO2TelematicsRequestBean:{  \"resId\": \"Yizi\",  \"taskId\": \"Some Travel\",  \"dateTime\": \"2018-02-01T17:20:12.977Z\",  \"domain\": \"SC\",  \"speedinMPH\": 30,
\"distanceinMiles\": 20,  \"outputTableName\": \"\",  \"emissionFormula\": \"R110\"}}"
```

Request URL

```
http://fosserver13.nat.bt.com:8081/CO2webservice/services/es/telematicsservice/calculate
```

Server response

Code: 200

Details

Response body

```
6.253700587190578
```

Response headers

```
date: Thu, 01 Feb 2018 17:37:07 GMT
content-type: application/json
content-length: 17
```

Responses

Code: 200

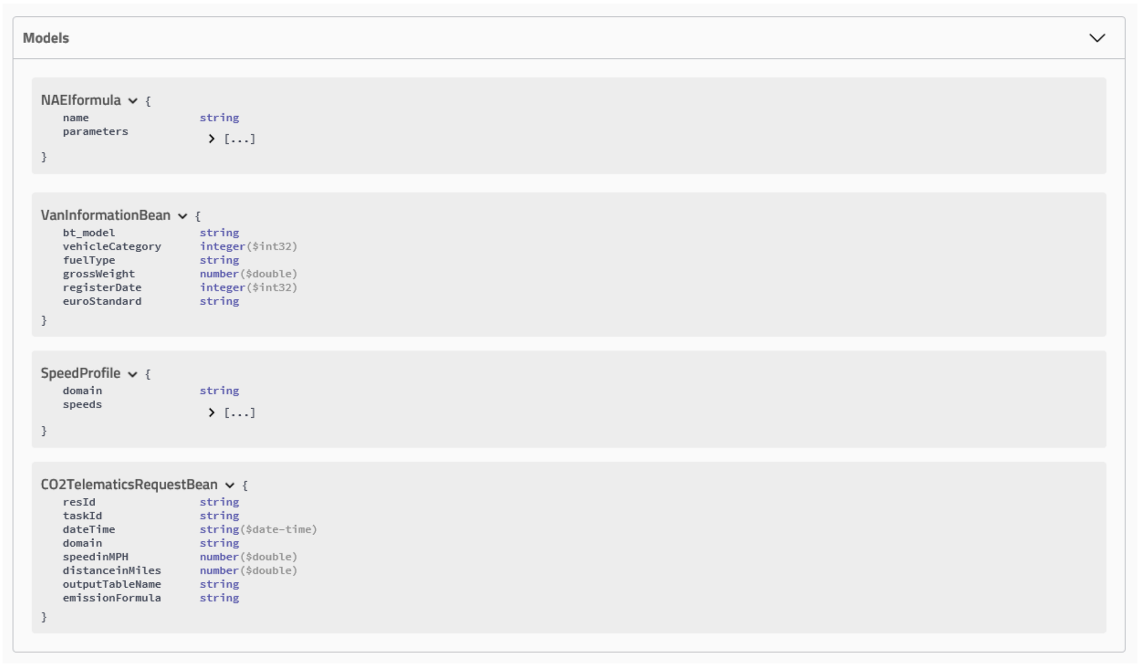
Description: *successful operation*

Example Value | Model

```
"string"
```

Figure C.3: Calculate CO_2 emissions with telematics data.

C.4 Resources documentation



```
Models

NAEIformula ⌄ {
    name                 string
    parameters
                          > [...]
}

VanInformationBean ⌄ {
    bt_model             string
    vehicleCategory      integer($int32)
    fuelType             string
    grossWeight          number($double)
    registerDate         integer($int32)
    euroStandard         string
}

SpeedProfile ⌄ {
    domain               string
    speeds
                          > [...]
}

CO2TelematicsRequestBean ⌄ {
    resId                string
    taskId               string
    dateTime             string($date-time)
    domain               string
    speedinMPH           number($double)
    distanceinMiles      number($double)
    outputTableName      string
    emissionFormula      string
}
```

Figure C.4: Java beans class documentation.

D Triangular method for calculating initial probabilities

For each customer, the probabilities that this customer will choose each available time windows are calculated using triangular distribution. The company is a monopoly and therefore the probability that a customer walks away without purchasing the service is zero. The probabilities of choosing the preferred time window and the other time windows are calculated based on triangular distribution:

$tmin$: customer arriving time window or the earliest time window available, whichever is earlier.
x: any of the available time windows; $x = tmin, .., 5$.
d: customer initial preferred time window.

The probabilities of each available time windows are:

$$p(x) = \begin{cases} \dfrac{2*(x - tmin + 1)}{(6 - tmin + 1)*(d - tmin + 1)}, & \text{if } tmin \le x \le d. \\ \dfrac{2*(6 - x)}{(6 - tmin + 1)*(6 - d)}, & \text{if } d \le x \le 5. \end{cases} \tag{D.1}$$

E Compare TS and VNS when the computational time is the same

	Emissions	Case 6	Case 7	Case 8	Case 9	Case 10
RVNS	Best	**327.33**	**239.47**	**299.35**	**293.51**	**265.84**
	Average	339.99	260.75	325.31	339.84	282.65
	S.D	8.00	15.53	12.20	15.49	11.80
	CPU Time	60	130	34	80	60
Tabu	Best	362.10	253.31	330.42	338.44	319.49
	Average	364.30	262.90	354.29	341.30	319.50
	S.D	3.88	2.42	2.22	1.76	0.02
	CPU Time	60	130	34	80	100
	p-value	0.000	0.456	0.000	0.608	0.000
	Different?	Y	N	Y	N	Y

Table E.1: Comparison between TS and RVNS.

F Proof of Proposition 4 and Lemma 4.1

First, calculate the stationary point satisfying the first order condition:

$$\begin{cases} \frac{\partial f}{\partial p_1} = P_1 + \sum_{n=1}^{N} \frac{\partial P_n}{\partial p_1} \left(p_n - c_n - \mu\tau_n\right) = 0, \\ \quad . \\ \quad . \\ \quad . \\ \frac{\partial f}{\partial p_N} = P_N + \sum_{n=1}^{N} \frac{\partial P_n}{\partial p_N} \left(p_n - c_n - \mu\tau_n\right) = 0, \end{cases} \tag{F.1}$$

For simplicity, denote $q_i^n := \frac{\partial P_n}{\partial p_i}$ and $\Lambda_n := p_n - c_n - \mu\tau_n$. Then equation (F.1) can be rewritten as:

$$\begin{pmatrix} q_1^1 & \dots & q_1^N \\ \vdots & \ddots & \vdots \\ q_N^1 & \dots & q_N^N) \end{pmatrix} \begin{pmatrix} \Lambda_1 \\ \vdots \\ \Lambda_N) \end{pmatrix} = \begin{pmatrix} -P_1 \\ \vdots \\ -P_N \end{pmatrix} \tag{F.2}$$

By Cramer's rule, the solution of F.1:

$$\Lambda_n = \det \begin{pmatrix} q_1^1 & \dots & q_1^{n-1} & -P_1 & q_1^{n+1} & \dots & q_1^N \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ q_N^1 & \dots & q_N^{n-1} & -P_N & q_N^{n+1} & \dots & q_N^N \end{pmatrix} \Bigg/ \det \begin{pmatrix} q_1^1 & \dots & q_1^N \\ \vdots & \ddots & \vdots \\ q_N^1 & \dots & q_N^N \end{pmatrix} \tag{F.3}$$

Due to the properties of P_i, we have:

$$q_i^n \begin{cases} \beta P_i \left(P_i - 1\right), & i = n \\ \beta P_n P_i, & i \neq n \end{cases} \tag{F.4}$$

In F.3, the denominator is:

$$\det\begin{pmatrix} q_1^1 & \dots & q_1^N \\ \vdots & \ddots & \vdots \\ q_N^1 & \dots & q_N^N \end{pmatrix} = \det\begin{pmatrix} \beta P_1(P_1-1) & \beta P_2 P_1 & \dots & \beta P_N P_1 \\ \beta P_1 P_2 & \beta P2(P_2-1) & & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \beta P_1 P_N & \beta P_2 P_N & \dots & \beta P_N(P_N-1) \end{pmatrix}$$

$$= \beta^N P_1 P_2 \dots P_N \det\begin{pmatrix} P_1-1 & P_2 & \dots & P_N \\ P_1 & P_2-1 & \dots & P_N \\ \vdots & \vdots & \ddots & \vdots \\ P_1 & P_2 & \dots & P_N-1 \end{pmatrix}$$

$$= \beta^N P_1 P_2 \dots P_N \det\begin{pmatrix} P_1-1 & P_2 & \dots & P_N \\ 1 & -1 & \dots & 0 \\ . & 0 & . & . \\ . & . & \ddots & . \\ 1 & 0 & \dots & -1 \end{pmatrix}$$

$$= \beta^N P_1 P_2 \dots P_N \det\begin{pmatrix} P_1+P_2+\dots+P_N-1 & 0 & \dots & 0 \\ 1 & -1 & \dots & 0 \\ . & 0 & . & . \\ . & . & \ddots & . \\ 1 & 0 & \dots & -1 \end{pmatrix}$$

$$= \beta^N P_1 P_2 \dots P_N \det \begin{pmatrix} -P_0 & 0 & \dots & \dots & 0 \\ 1 & -1 & \dots & \dots & \cdot \\ 1 & 0 & \dots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ddots & 0 \\ 1 & 0 & \dots & 0 & -1 \end{pmatrix}$$

$$= (-\beta)^N P_1 P_2 \dots P_N P_0$$

In F.3, the nominator is:

$$\det \begin{pmatrix} q_1^1 & \dots & q_1^{n-1} & -P_1 & q_1^{n+1} & \dots & q_1^N \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ q_N^1 & \dots & q_N^{n-1} & -P_N & q_N^{n+1} & \dots & q_N^N \end{pmatrix}$$

$$= \det \begin{pmatrix} \beta P_1(P_1-1) & \beta P_2 P_1 & \dots & \beta P_{n-1} P_1 & -P_1 & \beta P_{n+1} P_1 & \dots & \beta P_N P_1 \\ \beta P_1 P_2 & \beta P_2(P_2-1) & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \beta P_1 P_N & \cdot & \cdot & \beta P_{n-1} P_N & -P_N & \cdot & \cdot & \beta P_N(P_N-1) \end{pmatrix}$$

$$= \beta^N P_1 P_2 \dots P_N \det \begin{pmatrix} P_1 - 1 & P_2 & \cdots & P_{n-1} & -1 & P_{n+1} & \cdots & P_N \\ P_1 & P_2 - 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ P_1 & P_2 & \cdot & \cdot P_{n-1} & -1 & P_{n+1} & \cdot & P_{N-1} \end{pmatrix}$$

$$= \beta^{N-1} P_1 P_2 \dots P_N \det \begin{pmatrix} -1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdot & \cdot & -1 & \cdot & \cdot & 0 \\ 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & 0 & -1 & 0 & \cdot & -1 \end{pmatrix}$$

$$= (-1)^N \beta^{N-1} P_1 P_2 \dots P_N$$

Then (F.3) becomes:

$$\Lambda_n = \frac{1}{\beta P_0} \tag{F.5}$$

Let us calculate Hessian matrix, there are two elements in the Hessian matrix: $\frac{\partial^2 f}{\partial p_i^2}$ and $\frac{\partial^2 f}{\partial p_i \partial p_j}$.

$$\frac{\partial^2 f}{\partial p_i^2} = 2\frac{\partial P_i}{\partial p_i} + \sum_{n=1}^{N} \frac{\partial^2 P_n}{\partial p_i^2} \left(p_n - c_n - \mu \tau_n\right). \tag{F.6}$$

It is easy to show that:

$$\frac{\partial^2 P_n}{\partial p_i^2} = \begin{cases} \beta^2 P_i (P_i - 1)(2P_i - 1), & i = n \\ -\beta^2 P_i P_n (1 - 2P_i), & i \neq n \end{cases} \tag{F.7}$$

Then the equation (F.6) when it satisfies the first order condition becomes:

$$\begin{aligned} \frac{\partial^2 f}{\partial p_i^2} &= 2\beta P_i (P_i - 1) + \frac{1}{\beta P_0} \left(\beta^2 P_i (P_i - 1)(2P_i - 1) + \sum_{n=1, n\neq i}^{N} \left(-\beta^2 P_i P_n (1 - 2P_i) \right) \right) \\ &= 2\beta P_i (P_i - 1) + \frac{1}{\beta P_0} \left(-\beta^2 P_i (2P_i - 1) P_0 \right) \\ &= -\beta P_i \\ &\leq 0 \end{aligned}$$

$$\frac{\partial^2 f}{\partial p_i \partial p_j} = \frac{\partial^2 f}{\partial p_j \partial p_i} = \frac{\partial P_i}{\partial p_j} + \frac{\partial P_j}{\partial p_i} + \sum_{n=1}^{N} \frac{\partial^2 P_n}{\partial p_i \partial p_j} \left(p_n - c_n - \mu \tau_n\right) \tag{F.8}$$

$$\frac{\partial^2 P_n}{\partial p_i \partial p_j} = \begin{cases} \beta^2 P_i P_j (2P_i - 1), & n = i \\ \beta^2 P_i P_j (2P_j - 1), & n = j \\ 2\beta^2 P_i P_j P_n, & n \neq i, j \end{cases}$$

Then the equation (F.8) when it satisfies the first order condition becomes:

$$\begin{aligned} \frac{\partial^2 f}{\partial p_i \partial p_j} &= 2\beta P_i P_j + \frac{1}{\beta P_0}\left(\beta^2 P_i P_j (2P\_1) + \beta^2 P_i P_j (2P_j - 1) + \sum_{n=1, n\neq i,j}^{N} 2\beta^2 P_i P_j P_n\right) \\ &= 2\beta P_i P_j + \frac{1}{\beta P_0}(-2\beta^2 P_i P_j P_0) \\ &= 0 \end{aligned}$$

So the Hessian matrix has the following form:

$$H(x) = \begin{pmatrix} -\beta P_1 & \cdot & \cdot & 0 \\ 0 & -\beta P_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdot & \cdot & -\beta P_N \end{pmatrix} \tag{F.9}$$

The Hessian matrix is a diagnostic matrix; it is trivial that Hessian matrix is negative definite.

G A numerical example demonstrating the meaning of the specified parameters in MNL demand model

The parameters in the MNL demand model is assumed to be [0.01, -0.50, -0.10, -0.20, 0.01, -0.50]. This means for instance, the fixed price policy is 50 for each time window, then the probabilities estimated by MNL model for time slots [8:00, 10:00]. . . [16:00, 18:00] and the probabilities of walk away are shown in Figure G.1

with different customer's perception of the service, which is measured by reservation price (the value customer think the service worth). When the reservation price (e.g., £5) is below the price of the service (e.g., £10), then the customer will most likely to walk away without purchasing. On the other hand, if the customer valued the service £50 much more than the tagged price, the probability of walk away will be small (0.01%).

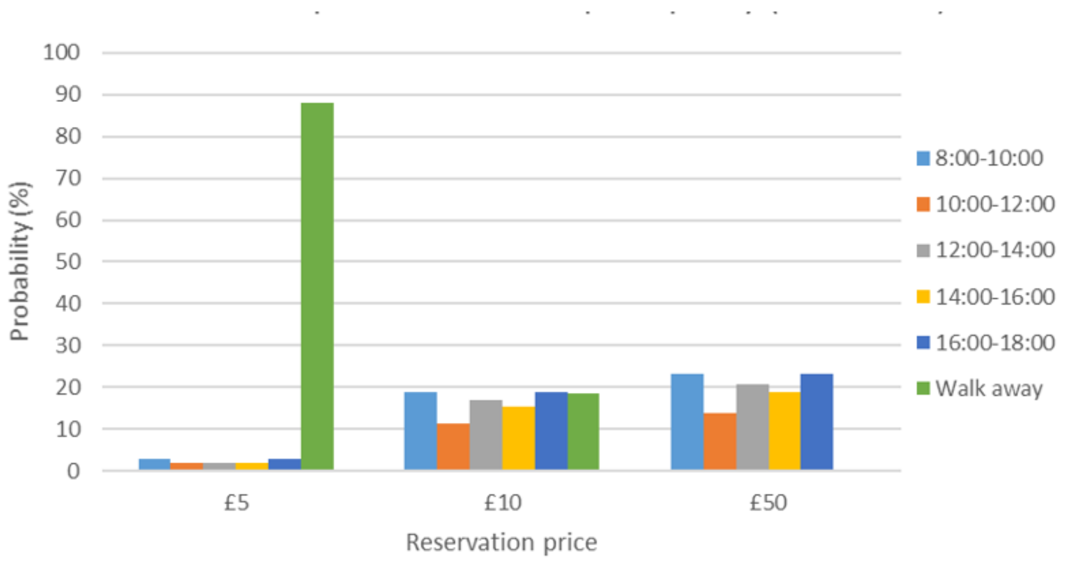


Figure G.1: Probability distributions of choice with different reservation price under fixed price policy (£10 for all).

H The relationship between reservation price and the probability of walkaway

Based on the MNL model, we have:

$$P\,(walk\ away) = \frac{e^{\beta(walk\ away)p_r}}{A + e^{\beta(walk\ away)p_r}},$$

where A is a constant, as the prices for all time windows are fixed.

$$\frac{\partial P(walk\ away)}{\partial p_r} = \frac{Ae^{\beta(walk\ away)p_r}}{(A + e^{\beta(walk\ away)p_r})^2} > 0,$$

indicates that when the reservation price increases, the probability of walk away decreases.

I Improving Xpress MIP solver performance

1. Switch from Xpress-mosel to Xpress-optimiser because of memory utilisation issue. Xpress-optimiser require .lp file, which is generated by mosel interface.

2. Add redundant constraints:

$$z2_{i,n} \leq M\left(1 - z1_{i,t}\right), n = t+2, \ldots, N \tag{I.1}$$

$$z1_{i,n} \leq M\left(1 - z2_{i,t}\right), n = 1, \ldots, t-2 \tag{I.2}$$

$$z2_{k,n} \leq M\left(1 - z1_{k,t}\right), n = t+2, \ldots, N \tag{I.3}$$

$$z1_{k,n} \leq M\left(1 - z2_{k,t}\right), n = 1, \ldots, t-2 \tag{I.4}$$

$$C_{ijk} \leq M\left(1 - x_{ijk}\right) \tag{I.5}$$

$$U_k \leq M \sum_{i=1}^{N} x_{i0k} \tag{I.6}$$

$$W_k \leq M \sum_{i=1}^{N} x_{i0k} \tag{I.7}$$

   Among them, constraints (I.1)-(I.4) are effective, the rest seem not.

3. Aggressive cut
   CUTSTRATEGY: Set it to 3 to allow for more cuts to be added to the problem
   CUTFREQ: How frequently to create cuts during the branch-and-bound search. You can try setting this to something more aggressive, like every 2 nodes.

4. Using loop

   - Set stop criteria Gap
   - Save mip solutions
   - Reduce gap
   - Load mip solutions
   - Restart

5. Adjust big M value from 20000 to 2000 (reduce roundoff error problems)
6. As time windows maybe adjacent, if they are adjacent, they are treated as one time windows and constraints are changed for those customers

References

Agatz, N., A. M. Campbell, M. Fleischmann, J. Van Nunen, and M. Savelsbergh (2013). Revenue management opportunities for internet retailers. *Journal of Revenue and Pricing Management 12*(2), 128–138.

Agatz, N. A. H., M. Fleischmann, and J. A. E. E. van Nunen (2008). E-fulfillment and multi-channel distribution–a review. *European Journal of Operational Research 187*(2), 339–356.

Akcelik, R. (1982). Derivation and calibration of fuel consumption models. internal report air 367-3. Technical report, Australian Road Research Board.

Apaydin, O. and M. T. Gonullu (2008). Emission control with route optimization in solid waste collection process: A case study. *Sadhana 33*(2), 71–82.

Baker, B. M. and M. A. Ayechew (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research 30*(5), 787–800.

Barth, M., F. An, T. Younglove, G. Scora, C. Levine, M. Ross, and T. Wenzel (2000). The development of a comprehensive modal emissions model. *NCHRP Web-only document 122*, 25–11.

Barth, M. and K. Boriboonsomsin (2008). Real-world carbon dioxide impacts of traffic congestion. *Transportation Research Record: Journal of the Transportation Research Board* (2058), 163–171.

Barth, M., T. Younglove, and G. Scora (2005). Development of a heavy-duty diesel modal emissions and fuel consumption model. California PATH Research Report. Technical report, UCB-ITS-PRR-2005-1, University of California, Riverside.

BBC News (2014, March 17). Ocean acid: The effect of CO_2 on life on the seabed. `http://www.bbc.co.uk/news/magazine-26755189/`. Accessed 2014-04-07.

Bektaş, T. and G. Laporte (2011). The pollution-routing problem. *Transportation Research Part B: Methodological 45*(8), 1232–1250.

Belhaiza, S., P. Hansen, and G. Laporte (2014). A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Computers & Operations Research 52*, 269–281.

Bell, J. E. and P. R. McMullen (2004). Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics 18*(1), 41–48.

Ben-Akiva, M. and S. R. Lerman (1985). *Discrete choice analysis: theory and application to travel demand*, Volume 9. MIT press.

Bent, R. W. and P. Van Hentenryck (2004). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research 52*(6), 977–987.

Blue & Green Tomorrow (2017, February 10). 4 promising business opportunities for green entrepreneurs. `https://blueandgreentomorrow.com/features/4-promising-business-opportunities-green-entrepreneurs/`. Accessed 2018-01-03.

Boulter, P. G., T. J. Barlow, and I. S. McCrae (2009). Emission factors 2009: Report 3-exhaust emission factors for road vehicles in the united kingdom. *TRL Published Project Report*.

Boulter, P. G., I. S. McCrae, and T. J. Barlow (2007). *A review of instantaneous emission models for road vehicles*. TRL Limited Wokingham, United Kingdom.

Braekers, K., K. Ramaekers, and I. Van Nieuwenhuyse (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering 99*, 300–313.

Bräysy, O. and M. Gendreau (2005). Vehicle routing problem with time windows, Part II: Metaheuristics. *Transportation Science 39*(1), 119–139.

Breidenich, C., D. Magraw, A. Rowley, and J. W. Rubin (1998). The kyoto protocol to the united nations framework convention on climate change. *American Journal of International Law 92*(2), 315–331.

Bullnheimer, B., R. F. Hartl, and C. Strauss (1999). An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research 89*, 319–328.

Campbell, A. M. and M. Savelsbergh (2006). Incentive schemes for attended home delivery services. *Transportation Science 40*(3), 327–341.

Chen, B., R. Qu, R. Bai, and H. Ishibuchi (2016). An investigation on compound neighborhoods for VRPTW. In *International Conference on Operations Research and Enterprise Systems*, pp. 3–19. Springer.

Cleophas, C. and J. F. Ehmke (2014). When are deliveries profitable? *Business & Information Systems Engineering 6*(3), 153–163.

Demir, E., T. Bektaş, and G. Laporte (2011). A comparative analysis of several vehicle emission models for road freight transportation. *Transportation Research Part D: Transport and Environment 16*(5), 347–357.

Demir, E., T. Bektaş, and G. Laporte (2012). An adaptive large neighborhood search heuristic for the pollution-routing problem. *European Journal of Operational Research 223*(2), 346–359.

Demir, E., T. Bektaş, and G. Laporte (2014). The bi-objective pollution-routing problem. *European Journal of Operational Research 232*(3), 464–478.

Department for Business, Energy & Industrial Strategy (2016). 2016 UK greenhouse gas emissions, provisional figures. `https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/604408/2016_Provisional_Emissions_statistics.pdf`. Accessed 2017-04-07.

Department for Environment, Food and Rural Affairs of the UK government (2010, October). Greenhouse gas (GHG) conversion factors. `http://archive.defra.gov.uk/environment/business/reporting/conversion-factors.htm`. Accessed: 2017-10-19.

Department for Transport (2016). Road congestion and travel times collection. `https://www.gov.uk/government/collections/road-congestion-and-reliability-statistics`. Accessed: 2017-11-07.

Dorigo, M. (1992). *Optimization, learning and natural algorithms.* Ph. D. thesis, Politecnico di Milano.

Eglese, R. and D. Black (2010). Optimizing the routing of vehicles. In A. McKinnon, S. Cullinane, A. Whiteing, and M. Browne (Eds.), *Green Logistics: Improving the Environmental Sustainability of Logistics*, Chapter 10, pp. 215–228. London: Kogan Page.

Eglese, R., W. Maden, and A. Slater (2006). A road timetable^{TM} to aid vehicle routing and scheduling. *Computers & Operations Research 33*(12), 3508–3519.

Ehmke, J. F., A. M. Campbell, and B. W. Thomas (2016). Vehicle routing to minimize time-dependent emissions in urban areas. *European Journal of Operational Research 251*(2), 478–494.

Elmaghraby, W. and P. Keskinocak (2003). Dynamic pricing in the presence of inventory considerations: Research overview, current practices, and future directions. *Management Science 49*(10), 1287–1309.

Erlandsson, L., J. Almen, and H. Johansson (2008). Measurement of emissions from heavy duty vehicles meeting Euro IV/V emission levels by using on-board measurement in real life operation. In *16th International Transport and Air Pollution CongressTechnical University Graz.*

Favaretto, D., E. Moretti, and P. Pellegrini (2007). Ant colony system for a VRP with multiple time windows and multiple visits. *Journal of Interdisciplinary Mathematics 10*(2), 263–284.

Figliozzi, M. (2010). Vehicle routing problem for emissions minimization. *Transportation Research Record: Journal of the Transportation Research Board* (2197), 1–7.

Fisher, M. L. (1994). A polynomial algorithm for the degree-constrained minimum k-tree problem. *Operations Research 42*(4), 775–779.

Franceschetti, A., D. Honhon, T. Van Woensel, T. Bektaş, and G. Laporte (2013). The time-dependent pollution-routing problem. *Transportation Research Part B: Methodological 56*, 265–293.

Frey, H. C., K. Zhang, and N. M. Rouphail (2010). Vehicle-specific emissions modeling based upon on-road measurements. *Environmental Science & Technology 44*(9), 3594–3600.

Garbarino, E. and O. F. Lee (2003). Dynamic pricing in internet retail: effects on consumer trust. *Psychology & Marketing 20*(6), 495–513.

Gendreau, M., A. Hertz, and G. Laporte (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research 40*(6), 1086–1094.

Gendreau, M., A. Hertz, and G. Laporte (1994). A tabu search heuristic for the vehicle routing problem. *Management Science 40*(10), 1276–1290.

Giddens, A. (2009). *The politics of climate change.* Cambridge, UK.

Glover, F. (1989). Tabu search—part I. *ORSA Journal on Computing 1*(3), 190–206.

Glover, F. (1990). Tabu search—part II. *ORSA Journal on Computing 2*(1), 4–32.

GOV.UK (2017, April). Vehicle tax rates. `https://www.gov.uk/vehicle-tax-rate-tables`. Accessed 2017-04-07.

Greater London Authority (2007). The mayor's climate change action plan. `http://www.energyforlondon.org/wp-content/uploads/2016/07/CCAP-2007.pdf`. Accessed 2017-01-07.

Greenshield, B. (1935). A study of capacity. In *Proceedings of the Highway Research Board*, Volume 14, pp. 967–976.

Hansen, P., N. Mladenović, and J. A. M. Pérez (2010). Variable neighbourhood search: methods and applications. *Annals of Operations Research 175*(1), 367–407.

Harrabin, R. (2013, October 17). Human role in warming 'more certain'- UN climate chief. `http://www.bbc.co.uk/news/science-environment-24204323`. Accessed: 2017-10-19.

Hashimoto, H., T. Ibaraki, S. Imahori, and M. Yagiura (2006). The vehicle routing problem with flexible time windows and traveling times. *Discrete Applied Mathematics 154*(16), 2271–2290.

Hemsley, R. (2014, May 1). Is green brand perception worth the effort? `https://realbusiness.co.uk/hr-and-management/2014/05/01/is-green-brand-perception-worth-the-effort/`. Accessed 2018-01-03.

Hickman, J., D. Hassel, R. Joumard, Z. Samaras, and S. Sorenson (1999). MEET-Methodology for calculating transport emissions and energy consumption. European Commission, DG VII. Technical report, ISBN 92-828-6785-4, Luxembourg.

Ichoua, S., M. Gendreau, and J. Y. Potvin (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research 144*(2), 379–396.

Jabali, O., T. Van Woensel, and A. G. de Kok (2012). Analysis of travel times and CO_2 emissions in time-dependent vehicle routing. *Production and Operations Management 21*(6), 1060–1074.

Jemai, J., M. Zekri, and K. Mellouli (2012). An NSGA-II algorithm for the green vehicle routing problem. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 37–48. Springer.

Kim, S., M. E. Lewis, and C. C. White (2005). Optimal vehicle routing with real-time traffic information. *IEEE Transactions on Intelligent Transportation Systems 6*(2), 178–188.

Kirkpatrick, S. (1984). Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics 34*(5-6), 975–986.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi (1983). Optimization by simulated annealing. *Science 220*(4598), 671–680.

Kok, A. L., E. W. Hans, and J. M. J. Schutten (2011). Optimizing departure times in vehicle routes. *European Journal of Operational Research 210*(3), 579–587.

Kordic, M. (2016, April 8). Electric vehicles may replace gas cars by 2025, MIT Professor says. `https://www.greenoptimistic.com/electric-vehicles-replace-gas-cars-2025/#.Wnsi1zflvF6`. Accessed 2018-02-07.

Laroche, M., J. Bergeron, and G. Barbaro-Forleo (2001). Targeting consumers who are willing to pay more for environmentally friendly products. *Journal of Consumer Marketing 18*(6), 503–520.

Lenstra, J. K. and A. H. G. Rinnooy-Kan (1981). Complexity of vehicle routing and scheduling problems. *Networks 11*(2), 221–227.

Li, B. (2011). The multinomial logit model revisited: A semi-parametric approach in discrete choice analysis. *Transportation Research Part B: Methodological 45*(3), 461–473.

Li, H., T. Lv, and Y. Li (2015). The tractor and semitrailer routing problem with many-to-many demand considering carbon dioxide emissions. *Transportation Research Part D: Transport and Environment 34*, 68–82.

Lichty, L. C. (1967). *Combustion engine processes.* McGraw-Hill.

Lin, C., K. L. Choy, G. T. S. Ho, S. H. Chung, and H. Y. Lam (2014). Survey of green vehicle routing problem: past and future trends. *Expert Systems with Applications 41*(4), 1118–1138.

Lin, K. Y. (2006). Dynamic pricing with real-time demand learning. *European Journal of Operational Research 174*(1), 522–538.

Littlewood, K. (2005). Special issue papers: Forecasting and control of passenger bookings. *Journal of Revenue and Pricing Management 4*(2), 111–123.

Lorkowski, S., P. Mieth, K. U. Thiessenhusen, D. Chauhan, B. Passfeld, and R.-P. Schäfer (2004). Towards area-wide traffic monitoring-applications derived from probe vehicle data probe vehicle data. In *Applications of Advanced Technologies in Transportation Engineering (2004)*, pp. 389–394.

Lundy, M. and A. Mees (1986). Convergence of an annealing algorithm. *Mathematical Programming 34*(1), 111–124.

Maden, W., R. Eglese, and D. Black (2010). Vehicle routing and scheduling with time-varying data: A case study. *Journal of the Operational Research Society 61*(3), 515–522.

Mazzeo, S. and I. Loiseau (2004). An ant colony algorithm for the capacitated vehicle routing. *Electronic Notes in Discrete Mathematics 18*, 181–186.

Mladenović, N. and P. Hansen (1997). Variable neighborhood search. *Computers & Operations Research 24*(11), 1097–1100.

National Geographic (n.d.). Effects of global warming. `https://www.nationalgeographic.com/environment/global-warming/global-warming-effects/`. Accessed: 2017-10-19.

Ntziachristos, L., Z. Samaras, S. Eggleston, N. Gorissen, D. Hassel, A. J. Hickman, et al. (2000). Copert III: Computer programme to calculate emissions from road transport, methodology and emission factors (version 2.1). Technical report, European Energy Agency (EEA), Copenhagen.

Ohnishi, H. (2008). Greenhouse gas reduction strategies in the transport sector: Preliminary report. *Tech. rep., OECD/ITF Joint Transport Research Centre Working Group on GHG Reduction Strategies in the Transport Sector, OECD/ITF, Paris.*

Palmer, A. (2007). *The development of an integrated routing and carbon dioxide emissions model for goods vehicles.* Ph. D. thesis, Cranfield University, UK.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research 31*(12), 1985–2002.

Punakivi, M., H. Yrjölä, and J. Holmström (2001). Solving the last mile issue: reception box or delivery box? *International Journal of Physical Distribution & Logistics Management 31*(6), 427–439.

Qian, J. and R. Eglese (2014). Finding least fuel emission paths in a network with time-varying speeds. *Networks 63*(1), 96–106.

Qian, J. and R. Eglese (2016). Fuel emissions optimization in vehicle routing problems with time-varying speeds. *European Journal of Operational Research 248*(3), 840–848.

Romeo, F. and A. Sangiovanni-Vincentelli (1991). A theoretical framework for simulated annealing. *Algorithmica 6*(1-6), 302.

Rothstein, M. (1971). An airline overbooking model. *Transportation Science 5*(2), 180–192.

Saberi, M. and İ. Ö. Verbas (2012). Continuous approximation model for the vehicle routing problem for emissions minimization at the strategic level. *Journal of Transportation Engineering 138*(11), 1368–1376.

Sbihi, A. and R. W. Eglese (2007). The relationship between vehicle routing & scheduling and green logistics-A literature survey. Working Paper. The Department of Management Science, Lancaster University. < http://eprints.lancs.ac.uk/48900/ > [Access date 10-April- 2017].

Scott, C., N. Urquhart, and E. Hart (2010). Influence of topology and payload on CO_2 optimised vehicle routing. In *European conference on the applications of evolutionary computation*, pp. 141–150. Springer.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research 35*(2), 254–265.

Storn, R. and K. Price (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization 11*(4), 341–359.

Taillard, É., P. Badeau, M. Gendreau, F. Guertin, and J. Y. Potvin (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science 31*(2), 170–186.

Talbi, E. G. (2002). A taxonomy of hybrid metaheuristics. *Journal of Heuristics 8*(5), 541–564.

Toth, P. and D. Vigo (2003). The granular tabu search and its application to the vehicle-routing problem. *Informs Journal on Computing 15*(4), 333–346.

Toth, P. and D. Vigo (Eds.) (2014). *The vehicle routing problem.* SIAM.

Transport for London (2008). London freight plan - sustainable freight distribution: a plan for London. `http://www.bestufs.net/download/NewsEvents/articles/London-Freight-Plan_07.pdf`. Accessed 2017-01-07.

Turkensteen, M. (2017). The accuracy of carbon emission and fuel consumption computations in green vehicle routing. *European Journal of Operational Research 262*(2), 647–659.

Ubeda, S., F. J. Arcelus, and J. Faulin (2011). Green logistics at Eroski: A case study. *International Journal of Production Economics 131*(1), 44–51.

Ubeda, S., J. Faulin, A. Serrano, and F. J. Arcelus (2014). Solving the green capacitated vehicle routing problem using a tabu search algorithm. *Lecture Notes in Management Science 6*, 141–149.

U.S. Energy Information Administration (2017, April). How much carbon dioxide is produced from burning gasoline and diesel fuel? URL:`https://www.eia.gov/tools/faqs/faq.php?id=307&t=9`. Accessed: 2017-10-19.

Van Woensel, T., R. Creten, and N. Vandaele (2001). Managing the environmental externalities of traffic logistics: The issue of emissions. *Production and Operations Management 10*(2), 207–223.

Van Woensel, T., L. Kerbache, H. Peremans, and N. Vandaele (2008). Vehicle routing with dynamic travel times: A queueing approach. *European Journal of Operational Research 186*(3), 990–1007.

Willard, J. A. G. (1989). Vehicle routing using r-optimal tabu search. Master's thesis, The ManagementSchool, Imperial College, London.

WMO and UNEP (2007). Intergovernmental panel on climate change. In *World Meteorological Organization.*

Yang, X., A. K. Strauss, C. S. M. Currie, and R. Eglese (2014). Choice-based demand management and vehicle routing in e-fulfillment. *Transportation Science 50*(2), 473–488.

Zanni, A. M. and A. L. Bristow (2010). Emissions of CO_2 from road freight transport in london: Trends and policies for long run reductions. *Energy Policy 38*(4), 1774–1786.

Zhou, Y., A. Liret, J. Liu, E. Ferreyra, R. Rana, and M. Kern (2017). Decision support system for green real-life field scheduling problems. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pp. 355–369. Springer.