Tiered-Facility Vehicle Routing Problem with Global Cross-Docking

Anthony Smith

UNIVERSITEIT
iYUNIVESITHI
STELLENBOSCH
UNIVERSITY

100
1918 · 2018

Dissertation presented for the degree of
Doctor of Philosophy
in the Faculty of Engineering at Stellenbosch University

Promoter: Prof JH van Vuuren
Co-promoters: Prof P Toth, Ms L Bam

March 2018

Declaration

By submitting this dissertation electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: March , 2018

Abstract

The service delivery of public healthcare is severely threatened due to insufficient resources. A current South African public healthcare organisation responsible for processing pathological specimens makes use of a public transportation network containing laboratories of multiple tiers corresponding to different processing capabilities in its business model. The effective transportation of specimens between facilities in this network may potentially lead to significant financial cost savings.

The quest to establish a mathematical model for the transportation of specimens in this transportation network has led to the formulation of a novel variant of the celebrated *vehicle routing problem* (VRP) in the operations research literature, called the *tiered-facility vehicle routing problem with global cross-docking* (TVRPGC). This tri-objective combinatorial optimisation problem calls for the efficient route scheduling of a vehicle fleet tasked with the transportation of pathological specimens. The objectives of the model are to minimise the total distance covered by the fleet (so as to save on transportation variable costs), to minimise the number of vehicles required for specimen collection (so as to save on transportation fixed costs), and to minimise the travel time of the vehicle which spends the longest time on the road (so as to ensure specimen integrity and balance driver workload). The model constraints take into account maximum driver autonomy (a constraint on the time a vehicle may spend on the road), specimen collection demand, permissible workloads at the various facilities in the network of processing laboratories, and requirements in terms of which laboratories are capable of processing the various specimens. Crucially, the model also allows for the novel feature of local hand-over of specimens at facilities between vehicles.

The aforementioned model is validated, and exact and approximate solution techniques are developed for the model and implemented on a computer. These techniques draw inspiration from a thorough study of the prototype VRP in the literature — the capacitated VRP.

Investigations are launched into (i) the computational complexity of the exact solution procedure, (ii) the quality of solutions returned by the approximate solution technique with respect to a real-life instance of the TVRPGC within a South African pathology healthcare service provider context, and (iii) the desirability of a facility clustering-based approach toward decomposing instances of the TVRPGC into smaller problem subinstances.

Uittreksel

Dienslewering in publieke gesondheidsorg word noemenswaardig deur onvoldoende hulpbronne gekniehalter. ’n Huidige Suid-Afrikaanse gesondheidsorg-organisasie wat verantwoordelik is vir die analasie en verwerking van patologiese monsters maak van ’n publieke vervoernetwerk met laboratoria van verskeie verwerkingskapasiteite in sy besigheidsmodel gebruik. Die doeltreffende vervoer van monsters tussen fasiliteite in hierdie netwerk mag potensieel tot beduidende finansiële kostebesparings lei.

Pogings tot die daarstelling van ’n wiskundige model vir die verskeping van monsters in hierdie vervoernetwerk het gelei na die formulering van ’n nuwe variasie op die gevierde *voertuigroeteringsprobleem* (VRP) in die operasionele navorsingsliteratuur wat as die *veelvlakkige-fasiliteit voertuigroeteringsprobleem met globale kruiskonsolidasie* (VVRPGK) bekend staan. Hierdie driedoelige kombinatoriese optimeringsprobleem vra na die doeltreffende roete-skedulering van ’n vloot voertuie geoormerk vir die verskeping van patologiese monsters. Die doele van die model is om die totale afstand wat deur die vloot voertuie afgelê word, te minimeer (besparing van verskepingsveranderlike-koste), om die getal voertuie in die vloot wat vir monsterverskeping benodig word, te minimeer (besparing van verskepingsvastekoste), en om die tydsduur van die voertuig wat die langste tyd op die pad deurbring, te minimeer (handhawing van monsterintegriteit en die balansering van voertuigbestuurderwerkslading). Die modelbeperkings neem in ag die maksimum bestuurderoutonomie (’n beperking op die tydsduur wat ’n bestuurder op die pad mag deurbring), aanvraag na monsterverskeping, toelaatbare werksladings by die onderskeie fasiliteite in die netwerk van verwerkingslaboratoria, en vereistes in terme van watter laboratoria daartoe in staat is om die onderskeie monsters te verwerk. Die kruks van die model is egter die nuwe kenmerk waarvolgens lokale kruiskonsolidasie van monsters tussen voertuie by enige fasiliteit toegelaat word.

Die bogenoemde model gevalideer daar word ook eksakte en benaderde oplossingstegnieke vir die model ontwikkel en rekenaarmatig geïmplementeer. Hierdie tegnieke vind inspirasie uit ’n deeglike studie van die prototipe VRP in die literatuur — die gekapasiteerde VRP.

Daar word ondersoek ingestel na (i) die berekeningskompleksiteit van die eksakte oplossingsmetodologie, (ii) die kwaliteit van oplossings gelewer deur die benaderde oplossingstegniek in die konteks van ’n realistiese VVRPGK geval in die Suid Afrikaanse patologiese gesondheidsdiens, en (iii) die wenslikheid van ’n fasiliteitsgroeperingsbenadering tot dekomposisie van VVRPGK gevalle na kleiner probleemdeelgevalle.

Acknowledgements

The author wishes to acknowledge the following people and institutions for their various contributions towards the completion of this work:

- My promoter, Professor JH van Vuuren, for the impact he has had on me both on a personal and academic level. The effort he expends over and above the requirements of his role as a promoter is truly inspiring, and has lifted my expectation of myself and the work that I have produced to a much higher level than when I first stepped through his door. The life lessons he has instilled in me and my colleagues of the *Stellenbosch Unit of Operations Research in Engineering* (SUnORE) will ripple through the remainder of our lives. I am eternally grateful for his friendship, guidance and support throughout the completion of this dissertation.

- My co-promoter, Professor P Toth, for showing an interest in my work during the early stages and providing mathematical assistance and modelling suggestions in times of dire need.

- My co-promoter, Ms L Bam, for initially providing the inspiration for my research and providing industry contacts and support throughout.

- My fellow SUnORE colleagues for their high-quality banter, usually at my expense, and their support and suggestions throughout the course of work towards this dissertation. The office memories, laughter, and friendship will remain long after the submission of this dissertation.

- Other friends and family for their encouragement and moral support during the completion of work towards this dissertation. Their understanding during times of unavailability and great pressure, especially toward the end of this work, is much appreciated.

- The Department of Industrial Engineering at Stellenbosch University, and SUnORE in particular, for the use of its great office space and computing facilities.

- Finally, SUnORE for the financial support that made this research possible.

Table of Contents

List of Acronyms

Acronym	Meaning
ACO	Ant Colony Optimisation
ACS	Ant Colony System
ACVRP	Asymmetric Capacitated Vehicle Routing Problem
AD	Average Distance
ADM	Average Distance-Between Means
AEX	Alternating Edge Crossover
ALNS	Adaptive Large Neighbourhood Search
AP	Assignment Problem
APN	Average Proportion of Non-Overlap
ARP	Arc Routing Problem
BaB	Branch-and-Bound
BaC	Branch-and-Cut
BCP	Branch-and-Cut-and-Price
BCTP	Bi-Objective Covering Tour Problem
CCCP	Continuous Capacitated Clustering Problem
CCP	Capacitated Clustering Problem
CCVRP	Cumulative Capacitated Vehicle Routing Problem
CLARA	Clustering Large Applications
CPMP	Continous P-Median Clustering Problem
CPTP	Capacitated Profitable Tour Problem
CTP	Covering Tour Problem
CVRP	Capacitated Vehicle Routing Problem
DARP	Dial-a-Ride-Problem
DIANA	Divisive Analysis Clustering Algorithm
DoH	Department of Health
FANNY	Fuzzy Analysis
FOM	Figure of Merit
GA	Genetic Algorithm
GAP	Generalised Assignment Problem
GRPs	General Routing Problems
HFVRP	Heterogeneous Vehicle Routing Problem
HGreX	Heuristic Greedy Crossover
IRP	Inventory Routing Problem
MACO	Multi-Objective Ant Colony Optimisation
MIP	Mixed Integer Programming
MOO	Mulit-Objective Optimisation
MVCTP	Multi-Vehicle Covering Tour Problem

Acronym	Meaning
NHLS	National Health Laboratory
PAM	Partitioning Around Medoids
PCVRP	Price Collecting Vehicle Routing Problem
PDP	Pickup and Delivery Problem
PDPTW	Pickup and Delivery Problem with Time Windows
POCSX	Partially Optimised Cyclic Shift Crossover
PSMDA	Penalised Static Move Descriptor Algorithm
PSO	Particle Swarm Optimisation
PVRP	Periodic Vehicle Routing Problem
SCVRP	Symmetric Capacitated Vehicle Routing Problem
SDVRP	Split Delivery Vehicle Routing Problem
SOM	Self-Organising Maps
SOTA	Self-Organising Tree Algorithm
SPPRC	Shortest Path Problem with Resource Constraints
SST	Shortest Spanning Tree
ST	Single Truck
SVM	Support Vector Machine
TOP	Team Orienteering Problem
TP	Transportation Problem
TSP	Travelling Salesman Problem
TTC	Truck-and-Trailer Combination
TTRP	Truck-and-Trailer Routing Problem
TVRPGC	Tiered-Facility Vehicle Routing Problem with Global Cross-Docking
UPGMA	Unweighted Pair Group Method with Arithmetic Mean
VEGA	Vector Evaluated Genetic Algorithm
VMIP	Vendor Managed Inventory Problem
VRP	Vehicle Routing Problem
VRPB	Vehicle Routing Problem with Backhauls
VRPMS	Vehicle Routing Problem with Multiple Synchronisation
VRPSD	Vehicle Routing Problem with Simultaneous Pickup and Delivery
VRPTW	Vehicle Routing Problem with Time Windows

List of Figures

List of Tables

List of Algorithms

CHAPTER 1

Introduction

Contents

1.1 Background

Healthcare is one of the quintessential pillars of any civilised society, with the first evidence of surgery dating back to the stone age [402] and the profession of a medical doctor being evident in the hieroglyphics of the ancient Egyptians, dating back to 3 000 BC [269]. After hygiene and nutrition, the advancement of healthcare has been the third-most prevalent factor in the increase of the average human life expectancy [179], with modern medicine now being able to combat fatal afflictions of yesteryears. These advancements have resulted in a doubling of the average life expectancy over the past 200 years, with a highest average life expectancy of 89.52 being achieved in the Principality of Monaco [236].

Unfortunately, the benefits of these advancements have not been distributed equally. The global discrepancies in these benefits are highlighted in Figure 1.1, which clearly illustrates that Africa has been the greatest laggard in respect of the benefits of healthcare. Philosophically speaking, the value of a human life is priceless, but in reality this number varies from R3.4 million to R24.2 million based on the value of a statistical human life[1] [68]. However controversial the notion of placing a value on a human life, it is widely accepted that every individual should have access to healthcare.

In a utopian society, free high-quality healthcare would be available to everyone. Unfortunately, however, only a select few have the resources available to guarantee them access to quality healthcare — a situation reminiscent of Orwell's notion that "all animals are equal but some animals are more equal than others" [331]. The vast majority of the world population relies on public health care provided by the state, with over 400 million people not having access to even

[1] The economic term *value of a statistical human life* is a measure of how much wealth an individual is willing to exchange for small changes in mortality risks. If, for example, a 100 000 people were each willing to spend a R100 to decrease their chances of mortality, the estimated statistical value of a human life would be R10 million.

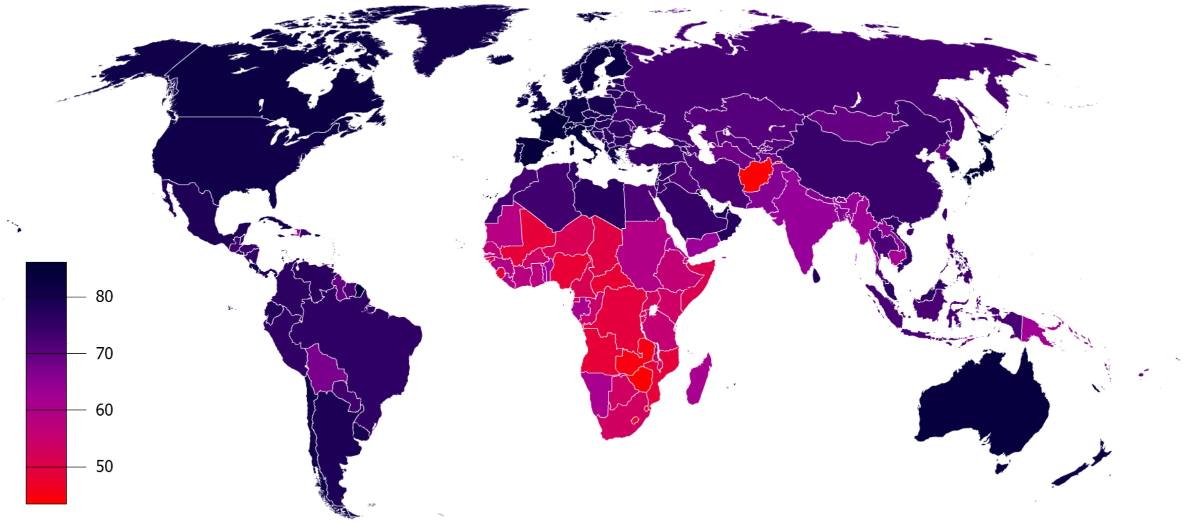


FIGURE 1.1: *National life expectancies (in years) [450].*

the most basic form of healthcare [449]. States have varying levels of resources and expertise at their disposal, with most developing states being severely under-resourced. Figure 1.2 illustrates the vast discrepancies in the ratios of trained physicians to patients in each country, with Tanzania and Malawi registering 50 000 patients per physician [241].

A blanket approach to public health care policies and infrastructure is not possible due to the typically large geographic fluctuation of resources, demographics and region-specific needs, as outlined above. A conference was therefore held in January 2008 in Maputo, Mozambique with a view to establish, in collaboration with the World Health Organisation, the Centre for Disease Control and Prevention, the United States Agency for International Development, the American Society for Clinical Pathology, the Clinton Foundation, the Bill and Melinda Gates Foundation and the Supply Chain Management System [396], recommendations for health care services of developing nations with limited resources. This conference gave rise to *inter alia* a framework for tiered, integrated national networks of pathology laboratories with the aim of strengthening these laboratory capacities in resource-limited countries. Amongst several other African states, South Africa took part in the development of this framework (and is also a signatory of the subsequent *Maputo Declaration*). A prominent South African pathological medical service provider, in fact, currently makes use of such a tiered laboratory network. Its laboratories are partitioned into four tiers: primary laboratories (tier 1), district laboratories (tier 2), regional laboratories (tier 3) and national laboratories (tier 4). Several other organisations in other countries also already employ tiered laboratory networks, such as the National Health Service of the United Kingdom, the Ethiopian Public Health Institute of Ethiopia, and the public healthcare services of eight different African countries [453].

The number of facility tiers in such pathological laboratory systems and the sophistication of specimen testing performed at each tier typically vary, depending on the population served, the physical infrastructure available (including road conditions), basic resources available (such as water and electricity) and the availability of trained technical personnel in-country [396]. The tier allocation is, however, typically nested in the sense that a facility of tier i can process any type of pathological specimen that can be processed at a facility of tier j if $j < i$, but certain specimen types exist which can be processed at a facility of tier i that cannot be processed at any facility of a lower tier. Facilities of the lowest tier represent customers at which the commodities

FIGURE 1.2: *Global physician-to-patient ratios [241].*

originate and have to be collected — these facilities have no specimen processing or storage capabilities — their only role is to introduce new specimens into the system. Facilities of higher tiers may or may not introduce new specimens into the system, but their distinguishing feature is that they all offer specimen processing capabilities.

The efficient management of the pathological specimen transportation logistics[2] of a tiered network is crucial for its effective implementation. An often overlooked component in the development and operation of such a tiered pathological system is logistics costs and supply chain management. The healthcare sector is associated with notoriously poor industry standards in supply chain management [177]. Innovative logistics management is expected to help provide better quality service delivery by pathological medical service providers whilst lowering the cost of healthcare provision.

Pathology service provision, and especially publicly provided pathology service provision, face severe financial pressure in a number of countries. In South Africa, for example, the National Health Laboratory Service is currently in a financial crisis with debts exceeding R5-billion [10], while in the United Kingdom the National Health Service pathology section has embarked on a long-term rationalisation process with a view to reduce costs significantly [92]. Although specimen collection and transportation are crucial activities to the operation of a pathology service, these activities do not represent specialised laboratory services or require large numbers of staff with specialised skills. Accordingly, these activities should be prioritised when investigating mechanisms by which to reduce pathology service delivery costs.

[2]Logistics has played an integral role in global development for over 5 000 years [127], but the term was only coined during the wars between the ancient Greek city states and the budding Roman empire when military officers, called *Logistikas*, were assigned duties related to the supply and distribution of resources [45]. The first evidence of logistics may, however, be found in the material handling technology used during the construction of the Egyptian pyramids, progressing through to the use of Greek rowing vessels which laid the foundation for intercontinental trade, eventually leading to the defeat of the Germans during World War II and finally resulting in the modern commercial notion of supply chain management [127].

The provision of decision support, based on a mathematical modelling approach, to logistics managers tasked with routing decisions for a fleet of vehicles dedicated to pathological specimen collection is therefore expected to be of considerable financial benefit to a pathology service.

1.2 Informal problem description

The problem considered in this dissertation is that of decision support aimed at the improvement of the transportation network system efficiency associated with the collection and delivery requirements of commodities in a tiered-facility processing system (such as the type of pathological specimen collection described in §1.1). A mathematical modelling approach towards achieving such a decision support capability gives rise to a new type of *vehicle routing problem* (VRP), henceforth called the *tiered-facility vehicle routing problem with global cross-docking* (TVRPGC).

Each commodity within a TVRPGC network is required to be transported from a collection facility and delivered to a facility capable of processing the commodity. The efficiency of the transportation process may be improved by the pursuit of three objectives, namely

- to minimise the total distance travelled by vehicles during the commodity collection and delivery process (a variable cost objective),
- to minimise the maximum length of time that any commodity collection and delivery vehicle spends on the road (a driver autonomy-related objective aimed at the avoidance of driver fatigue and the preservation of commodity integrity), and finally
- to minimise the maximum number of vehicles utilised during the specimen transportation process (a fixed cost objective).

The ability of a facility to process a commodity depends on the type of commodity as well as the personnel and equipment present at the facility. Within a TVRPGC network, certain commodities may typically only be processed at facilities of a certain tier or higher. The facility tiers are therefore assumed to exhibit nested processing capabilities in the sense that a facility of a specific tier can process all commodity types that are processable at facilities of lower tiers, but there are certain types of commodities that are processable at the facility tier in question which are not processable at any lower tiered facility.

If, upon arrival at a facility, it is found that a commodity has been delivered to a facility that does not have the required processing capabilities, the commodity will require additional transportation to a higher-tiered facility. A commodity may therefore possibly have to be transported to several intermediate facilities due to the processing limitations of the different tiers of facilities before being delivered to a laboratory of a suitable tier for its processing. A facility may consequently exhibit a positive or a negative demand for commodities of any particular type, depending on whether commodities of this type are required to be collected from or deposited at the facility.

The facilities of a TVRPGC organisation are only available for commodity collection and delivery during specific time windows (*i.e.* during certain hours of the day). The number of commodity transportation vehicles available is also limited, and the maximum time that a vehicle can travel before returning to its home depot should typically remain below a pre-specified limit.

The different types of commodities are all associated with varying expiration time windows, depending on the nature of the commodities and the storage techniques utilised to maintain commodity integrity. Commodities must therefore be delivered to facilities that are able to process them within a certain time frame.

The TVRPGC described above is rich in requirements and complexity, and exhibits novel features not accommodated by the existing VRP models in the academic literature. These novel aspects include:

- Facilitation of global cross-docking of commodities (handover of commodities from one vehicle to another at any facility within the network) so as to allow for the eventual delivery of commodities at facilities of appropriate tiers.
- Construction of a daily commodity transportation vehicle routing schedule, but allowing for the possibility that the daily routes of the vehicles are not exactly the same every day (due to stochastic variability of commodity samples entering the system).
- Roads of the same distance may not be traversable within the same time due to differing road conditions (some roads perhaps being in rural areas, for example, and hence inducing difficult travel conditions).

The problem is furthermore multi-objective in nature, requiring the pursuit of Pareto-optimal (trade-off) solutions instead of unique optimal solutions as is traditionally the case in VRP models. For these reasons, a novel VRP model is established in this dissertation in order to accommodate the unique requirements of the type of commodity transportation problem described above. An approximate, efficient solution methodology for the TVRPGC is also designed, implemented and tested.

Although specifically tailored to the pathological specimen transportation problem described in §1.1, it is anticipated that the TVRPGC may admit various other applications. One such alternative application may involve the collection and delivery of parcels for a national postal service. The tiers of sorting facilities in this application may be determined by the intended destinations of parcels, such as local, regional, provincial, national or international destinations. Once delivered at a sorting facility, a parcel may require additional transportation to a sorting facility of an appropriate tier.

1.3 Dissertation objectives

The following nine objectives are pursued in this dissertation:

I To *conduct* a thorough survey of the literature related to the multi-objective combinatorial optimisation problem considered in this dissertation with a view to develop an understanding of how similar mathematical models have been formulated and solved in the past for similar problems. In particular, literature related to the following areas is to be consulted:

   (a) vehicle routing problems in general, and the archetypal *capacitated vehicle routing problem* (CVRP) in particular,

   (b) methods of data clustering into groups according to a similarity measure, as well as measuring the quality of such clustering,

   (c) exact methods for solving combinatorial optimisation problems, and

   (d) (meta)heuristic methods for solving combinatorial optimisation problems.

II To *study* the CVRP in depth with a view to lay a sound foundation for a later mathematical model formulation applicable to the TVRPGC. This study is to take the form of:

(a) a personal computer implementation of a suitable mathematical model for the CVRP (which may be generalised at a later stage so as to incorporate the additional requirements of the TVRPGC) within a software environment that will allow exact solution of small instances of the CVRP,

(b) a verification and a validation of the model implementation in Objective II(a) in the context of benchmark test instances of various sizes from the literature (*i.e.* of different levels of complexity) for which optimal solutions are known,

(c) the design and implementation of suitable metaheuristics capable of approximately solving large instances of the model derived in pursuit of Objective II(a),

(d) a verification and a validation of the metaheuristic algorithmic implementation of Objective II(c) in the context of the same benchmark test instances considered in pursuit of Objective II(b), and

(e) an investigation into the improvement in solution time gained and the trade-off degradation in solution quality incurred by incorporating a heuristic clustering step aimed at vehicle service zone formation when solving the model of Objective II(a) before implementing the exact solution technique of Objectives II(a)–(b) or the metaheuristic solution technique of Objectives II(c)–(d).

III To *formulate* a mathematical model for the TVRPGC in the form of a tri-objective combinatorial optimisation problem, taking into account all the constraints dictated by typical practical situations. This formulation should build upon the basic underlying CVRP model formulation of Objective II(a).

IV To *establish* hypothetical TVRPGC test instances of various sizes (*i.e.* of different levels of complexity) for the model of Objective III, and to make these available online.

V To *ascertain* the complexity associated with solving the TVRPGC model of Objective III exactly by implementing the model within an appropriate exact solution software environment, and then attempting to solve the model exactly in the context of the small test instances of Objective IV.

VI To *design* and *implement* a metaheuristic solution approach capable of solving even the large TVRPGC test instances of Objective IV. This should include a parameter evaluation experiment aimed at identifying suitable values for all the parameters required in implementation of the metaheuristic within the context of the test instances.

VII To *validate* the approximate solution approaches of Objective VI with respect to a real-world TVRPGC case study in the context of pathological specimen collection by a healthcare service provider.

VIII To *investigate* the potential extent of improvements in solution time that may be gained and the corresponding trade-off degradations in solution quality incurred by incorporating a heuristic clustering step aimed at vehicle service zone formation when solving the TVRPGC model of Objective III, before implementing the metaheuristic solution techniques of Objective VI in the context of the model test instances of Objective IV.

IX To suggest sensible, related future work which may be pursued in follow-up studies to the research reported in this dissertation.

1.4 Dissertation scope

The range of facilities, vehicle fleet composition, demand requirements and modelling approaches that may be considered in the context of a TVRPGC transportation network is extensive. In order to narrow down the scope of the TVRPGC, the following scope delimitations are adopted in this dissertation:

- The attributes of individual commodities transported within the network and all the different types of processing that they require at facilities are not modelled explicitly. Since there may be numerous types of commodities and even more types of possible processing requirements to which they may be subjected at facilities, the modelling of all these individual requirements would be impractical. Modelling consideration is therefore limited to the overall flow of commodities within the TVRPGC transportation network as opposed to individual commodity processing requirements. More specifically, the set of commodities is partitioned into subsets corresponding to the lowest facility tiers at which they may be processed.

- Route scheduling only involves the main routing operations of a tiered facility network. Emergency routing requirements (as a result of unforseen, urgent demand or vehicle breakdown) and other *ad hoc* demands are not considered.

- The locations, demand, and operating hours of every facility are assumed to be known before the onset of the route scheduling planning window.

- Commodity deterioration is limited to the travel time required to deliver a commodity to any facility in the TVRPGC network as it is assumed that once a commodity reaches any facility it can be stored and preserved in such a manner that the deterioration of the commodity is slowed down to the point where it is henceforth negligible.

- The delivery of a commodity to an appropriate facility tier (capable of processing it) may occur over several TVRPGC planning periods, depending on the number of facility tiers present, as the routing schedule of vehicles is only designed to ensure the delivery of a commodity to a strictly higher tier at the end of a TVRPGC planning period than that of the tier at which it was initially collected during that period and, as such, it may take a commodity several planning periods to eventually arrive at a facility capable of its processing.

1.5 Dissertation organisation

This dissertation comprises fourteen further chapters, following this introductory chapter. These chapters are partitioned into four parts. The first such part is a literature review and consists of three chapters. The first of these chapters, Chapter 2, is devoted to a general introduction to and brief history of various incarnations of the VRP. The classical VRP is discussed in some detail and descriptions are provided of numerous variations on the VRP that are present in the literature. These variations are further elaborated upon in terms of their underlying network structure, the type of transportation requests accommodated, the intra-route constraints enforced, the vehicle fleet composition available, various inter-route constraints enforced and, finally, the optimisation objectives adopted.

Chapter 3 is devoted specifically to the CVRP — the archetypal VRP — because this problem forms the basis on which a model for the TVRPGC is later established. Various solution approaches that have been proposed in the literature for solving the CVRP are described. The chapter initially contains a presentation of different model formulations of the CVRP available in the literature, and this is followed by a review of various classical exact solution approaches that have been proposed for solving these mathematical models. More recent exact solution approaches are then described. The discussion next progresses towards approximate solution approaches put forward for solving instances of the CVRP, starting with a number of classical heuristics that have appeared in the literature. More powerful metaheuristics and hybrids of these methods are finally reviewed and appropriate measures of the quality of solutions returned by these methods are discussed in the closing sections of the chapter.

The literature on clustering methods is briefly reviewed in Chapter 4, because one of the objectives in this dissertation is to ascertain the desirability of decomposing large instances of the CVRP and the TVRPGC into smaller instances of these respective problems by following a clustering approach in respect of customers. The different clustering paradigms available in the literature, and the applications of clustering algorithms within each of these paradigms, are elaborated upon. Crucial features of any successful clustering algorithm are described, such as the determination of an appropriate number of clusters, validating the clustering returned by an algorithm and a methodology for the comparison of the quality of results returned by different clustering techniques. Finally, admissibility criteria are presented for clustering algorithms in the form of available methodologies for assessing the sensitivity and stability of clustering algorithm results.

Part II of the dissertation consists of four further chapters, which are focused on the formulation of models for the CVRP, their relevant solution methodologies and, finally, the use of clustering algorithms for partitioning the customer sets of large, real-life CVRP instances into smaller sub-problems that are easier to solve. The first chapter of Part II, Chapter 5, consists of a brief model assumption section, a description of the particular mathematical model for the CVRP adopted in this dissertation and a discussion on the properties of this mathematical model as well as the viability of solving the model exactly.

Two metaheuristics for solving instances of the CVRP model of Chapter 5 are described in Chapter 6. The first is a genetic algorithm and the second is an ant colony optimisation implementation. Details pertaining to the genetic algorithm, as well as its constituent components (such as the representation of chromosomes, the construction of an initial population, the selection of the parent chromosomes and, finally, the crossover and mutation operators employed) are described. A selection of these components are then elaborated upon, highlighting the particular algorithmic implementation process followed in this dissertation within the context of the CVRP. The ant colony optimisation algorithm is similarly presented, accompanied by descriptions of its key components (such as the pheromone trail updating method adopted, the initial heuristic matrix generation method implemented and the tour refinement methodology employed).

In Chapter 7, a metaheuristic parameter evaluation experiment is carried out for both the genetic algorithm and the ant colony optimisation algorithm of Chapter 6. This parameter evaluation is performed in respect of three well-known CVRP benchmark instances with varying degrees of complexity. The performances of the algorithms are then compared, after which a brief discussion on the results obtained concludes the chapter.

The final chapter of Part II, Chapter 8, contains a presentation of a clustering approach that may be followed to decompose large CVRP instances into smaller, more manageable sub-problems. Key features of the algorithm (such as the determination of the number of clusters and the steps undertaken to determine which clustering algorithm to implement) are presented. The inclusion of this clustering approach within an (approximate) CVRP solution framework is validated against several well-known benchmark instances and the results are finally discussed briefly.

The specific modelling and solution approaches pertaining to the TVRPGC is the topic of Part III of this dissertation, which contains three chapters. A novel TVRPGC model formulation is presented and validated in respect of a small, hypothetical test instance in Chapter 9. The discussion on the mathematical model formulation contains descriptions of the parameters, variables, objectives and constraints employed. The potential and versatility of the mathematical model is demonstrated and the results are discussed to consolidate the importance of establishing a mathematical programming model of the TVRPGC. The need for an approximate solution technique applicable to realistically sized instances of the TVRPGC model is also highlighted in this chapter.

A tri-objective ant colony optimisation algorithm, tailored to the specific requirements of the TVRPGC, is subsequently designed and implemented in Chapter 10. The chapter is focused around key aspects of the algorithm, including pheromone monitoring, initial heuristic determination, route construction paradigms, non-dominated front determination and constraint handling. The algorithm contains several parameters and a thorough parameter evaluation is therefore also performed in order to determine a set of suitable values for these parameters in the context of slightly modified, well-known CVRP instances in Chapter 11. A brief discussion of the results closes both the chapter and Part III.

Part IV of the dissertation consists of two chapters, which are focused on the application of the ant colony optimisation algorithm to a real-life instance of a TVRPGC. The first chapter of this part, Chapter 12, contains a detailed description the real-life instance with all the relevant information presented in such a manner so as to allow for future researchers to replicate the study while still maintaining client confidentiality. The second chapter of the part, Chapter 13, is dedicated to a real-world validation of the approximate solution approaches developed in this dissertation for the TVRPGC by means of a practical case study based on the data presented in Chapter 12. The results of the case study are discussed and the potential benefits of adopting the proposed solution methodology in future endeavours within the South African healthcare sector are highlighted.

Part V contains the summary and conclusion of the dissertation, and consists of two final chapters. The first chapter of this part, Chapter 14, contains a concise summary of the contributions of the dissertation as well as an appraisal of these contributions. Chapter 15 finally closes the dissertation and contains a number of recommendations in respect of future work following on the work reported in this dissertation.

Part I

Literature Review

CHAPTER 2

Vehicle Routing Problems

Contents

This chapter contains a brief introduction to the VRP in §2.1 according to a taxonomy proposed by Toth and Vigo [428]. Examples of the most prevalent VRP variants are included to illustrate the real-life applicability of the modifications proposed to the original, classical VRP in each variation. The first class of variants arises from modifications of the underlying network structure and is elaborated upon in §2.2. This is followed by a discussion on variations arising from different transportation requests in §2.3. Variations induced by various intra-route and inter-route constraints are discussed in §2.4 and §2.5, respectively. The discussion then turns to the different vehicle fleet characteristics in §2.6. In the VRP literature, three different classes of objectives are typically pursued. These three broad classes are described briefly in §2.7. The chapter closes in §2.8 with a brief summary of the work presented in the chapter.

2.1 Introduction

The VRP was first introduced in an article by Dantzig and Ramser [107] in 1959. They were interested in the real-world application of delivering gasoline to gas stations. The first mathematical model for the VRP was proposed in the paper, formulated simply as the celebrated *Travelling Salesman Problem*[1] (TSP) with the addition of a capacity constraint. An algorithmic solution approach was also suggested for the VRP. The algorithm proposed by Dantzig

[1]The problem of finding a minimum-weight closed route in a weighted, complete graph, visiting every vertex of the graph exactly once.

and Ramser was limited to small instances of the problem, but in 1964 Clarke and Wright [87] proposed an efficient greedy heuristic for obtaining good solutions to larger instances of the VRP. While the VRP is a generalisation of the TSP, it is much more difficult to solve than the TSP. Exact algorithms exist for the TSP which routinely solve instances with hundreds or even thousands of vertices [12] while the best exact algorithms for the VRP can currently only solve instances with roughly a hundred vertices [27, 174].

With such great forerunners, significant research interest has been generated by the problem, leading to extensive publications on a number of variations on the problem, with a Google scholar search of the words *vehicle routing problem* returning over 35 900 entries. In practice, vehicle routing may encapsulate one of the most significant success stories in operations research [198]. This success is demonstrated in the case of *United Parcel Service* (UPS), for example, where 103 500 drivers follow computer generated vehicle routes every day. These drivers, who implement a solution to a customised VRP variant, visit 7.9 million customers and handle an average of 15.6 million packages per annum [433].

Toth and Vigo [428] have suggested a classification system for variations on the VRP in terms of:

- The underlying network structure,
- the type of transportation requests,
- the constraints that affect each vehicle route individually (intra-route constraints),
- the vehicle fleet composition and their home locations,
- various inter-route constraints, and
- the optimisation objectives.

The remainder of this chapter is devoted to a description of a number of variations on the classical VRP of Dantzig and Ramser [107], following the above taxonomy.

2.2 Network Structure

In the VRP, tasks are associated with points in space which are usually modelled as the vertices of a geo-spatial graph. The VRP may therefore be considered as a *vertex routing problem* as opposed to an *Arc Routing Problem* (ARP) in graph theory in which tasks are associated with connections or links between the vertices. ARPs find real-life application in street sweeping and inspection, salt gritting and snow removal [97, 145, 454]. In urban areas, the vertices of the geo-spatial graph are densely populated and a large variety of tasks, such as finding good garbage collection routes, may be modelled as ARPs. *General Routing Problems* (GRPs) allow for a mixture of tasks associated with the vertices and edges of a graph [330]. Another key difference between a VRP and an ARP is the granularity of the underlying data and network resolution [428].

A further important problem characteristic of the VRP is the nature of the underlying data. If these data are symmetric and the movement of vehicles between vertices is unrestricted, the problem can be modelled on an undirected underlying graph. If, however, there is restriction of movement between vertices or the associated data related to vehicle flow are asymmetric, the problem has to be modelled on a directed, mixed or *windy* graph (see, for example, [210]).

2.3 Types of Transportation Requests

Following the taxonomy introduced by Toth and Vigo [428], the following popular VRP variations based on the nature of transportation requests occur in the literature:

Distribution of goods from a depot to customers. This is the most widely associated type of transportation request in the family of VRPs.

Delivery and Collection. In this case, items are collected from customers in addition to delivering items to customers, usually at the beginning or the end of the supply chain. This problem variation is also referred to as a *many-to-one VRP* when it is acceptable to assume that the collection and delivery begins and ends at the depot. The earliest and simplest problem variation in this class is the *VRP with backhauls* (VRPB) [426]. This variation deals with a bi-partitioned set of customers — the one subset requiring items to be delivered while the other subset requires items to be returned to the depot. This variation is limited to mutually exclusive delivery and collection subsets of customers, while the *VRP with Simultaneous Pickup and Delivery* (VRPSD) [313] deals with customers who have both types of transportation requests. A relaxation of the VRPSD is the *VRP with Divisible Deliveries and Pickups* [206], which allows for the collections and deliveries to be performed over separate trips by the same vehicle.

Simple Visits and Vehicle Scheduling. The first descriptor in this problem variation is applicable when the service required is simply to visit a location or customer as in the TSP; the location does not require any service. Another possibility (encapsulated by the second descriptor) is that certain vehicle route segments have to follow a sequence or schedule specified *a priori*.

Alternative and Indirect Services. Real-life situations exist where services can be performed in an alternative manner, such as delivering a parcel to either a customer's work address, home address or a central drop-off point within a vicinity of pre-specified radius from the customer. The *Multi-Vehicle Covering Tour Problem* (MVCTP) [212] is the instance where the vehicle visits a point that is close enough to the customer. This problem is, for example, applicable in the delivery of medicine to rural villages [212].

Point-to-Point Transportation. This problem variation deals with the transportation of goods or people between two specific points. The one point usually acts as a collection point and the other as a delivery point. In most instances, these locations are not the depot and applications arise in freight transportation and in passenger transport systems. The variation is referred to as a *many-to-many VRP*, and called the *Pickup-and-Delivery Problem* (PDP) [120] when applied to goods transportation or the *Dial-a-Ride-Problem* (DARP) [428] when applied to passenger transport.

Repeated Supply. This problem variation is associated with goods delivery where a customer requires repeated supply over an extended planning horizon. There are two major subcategories within this variation, namely the *Periodic VRP* (PVRP) [98] and the *Inventory Routing Problem* (IRP) [70]. The PVRP involves a two-stage planning process. Visiting patterns are decided upon during the first stage from a set of admissible patterns specified by the customers. The second stage involves solving the VRP for each day. The IRP exhibits uniqueness from the other variants as the model is not aimed at responding to customer demand; the delivery company instead determines when to visit a customer and how much stock to deliver in order to prevent stock-outs. A variant of the IRP is the

Vendor Managed Inventory Problem (VMIP) [130]. The IRP is often applied in *supply chain management* settings to reduce the well-known bullwhip effect [281]. The VMIP assumes the use of dynamic data to monitor the customer's stock levels and schedules deliveries accordingly.

Non-split and Split Services. In each of the above-mentioned variations it is assumed that all service tasks are completed by a single vehicle. These variations are collectively referred to as non-split VRPs. There are, however, cases where the demand exceeds the capacity of a vehicle or, as Dror and Trudeau [146] showed, savings can be incurred by splitting service requests into several smaller requests. The *Split Delivery VRP* (SDVRP) [146] allows for shipments to be split into arbitrarily smaller shipments.

Combined Shipment and Multi-modal Service. This problem variation is similar to the SDVRP, but differs from it in the sense that the individual shipments remain intact while several vehicles transport the shipment from supplier to customer through the use of intermediate transfer points. This is referred to as a *combined shipment VRP* and is commonly applied in multi-modal transport networks.

Routing with Profits and Service Selection. This problem variation deals with the case where limited resources, such as a limited fleet size, lead to only a subset of the customers being serviced. It was first introduced as a TSP variation and only later extended to a VRP variation. The classical approach towards solving this type of problem was to perform a two-stage decision process with request acceptance preceding a vehicle routing stage. There are, however, financial benefits to performing these stages simultaneously [428]. According to Feillet [156], there are three categories of the problem variation. The first category contains instances in which the routing costs and profits are combined into a single objective and is referred to as *Capacitated Profitable Tour Problem* (CPTP) [14]. The second category applies to instances where the objective is to maximise profit with an upper bound on route length, referred to as the *Team Orienteering Problem* (TOP) [15]. The final category is referred to as the *Prize Collecting VRP* (PCVRP) [416], where there is a lower bound on profit to be collected and the objective is to minimise the routing cost. More recently there has also been interest in this problem variation where customers are able to be serviced by a privately owned fleet or an outsourced fleet. This approach is resorted to when capacity of the fleet is exceeded or for economic reasons [350].

Dynamic and Stochastic Routing. The *Dynamic VRP* [424] is a variation in which relevant system conditions only become available during operations, whereas the term *stochastic* refers to the system conditions being uncertain, but the uncertainty is linked to a probability distribution. The dynamic component usually relates to customers' locations and demand profiles, while the stochastic element usually results from uncertain demand and travel time [428].

2.4 Intra-route Constraints

A key factor in defining a specific variation on the classical VRP is the constraints that determine the feasibility of the vehicle routes.

The first class of general constraints of this form is the class of *loading constraints*. These constraints constitute the simplest type of constraint as they can be written as an overall bound on a resource consumed at each vertex [428]. A VRP formulation may consist of several capacity constraints which limit the loading of vehicles, such as volume, weight and space constraints. In

VRP variants where multi-dimensional packing problems are also considered, such as in [186, 238], the capacity constraints become more complicated as the shipment and cargo containers used to load vehicles have to be described in either two or three dimensions. Further capacity complexity is also experienced when the delivery fleet consists of vehicles with multiple compartments as, for example, in [119] where the vehicle capacity is modelled as a one-dimensional vector consisting of partitioned sets.

Route length constraints also form a simple class of constraints that can be applied globally to limit various types of resource consumption along the edges of the VRP graph. These constraints can either be modelled as spatial distance constraints or as route duration constraints.

Intra-route constraints arise either from a limited number of available vehicles, small vehicle capacities or other constraints that limit the number of stations visited within a route so that feasibility can only be achieved through reuse of vehicles. The reuse of vehicles has become more popular with the advent of renewable energy vehicles which have a limited range and are required to refuel/recharge regularly [155, 391].

Time window constraints control the scheduling of resources in a VRP. These constraints usually limit the period during which a customer can receive items, but more complex applications have arisen recently as a result of legislation limiting the driving schedules of drivers and enforcing mandatory resting periods. These constraints may also be used to incorporate time-dependent travel time durations into the model so as to consider the period of the day, which affects traffic flow. This type of constraint can typically be partitioned into three subclasses, namely, *resource allocation constraints* (which repartition activities among resources), *sequencing constraints* (which ensure the execution order of resources) and *scheduling constraints* (which facilitate the selection of execution dates relating to the sequence of customer visitation [124]). Time window constraints may alternatively be classified according to their effect on the feasibility of routes. Time windows spanning relatively long periods are referred to as *soft time windows* as they have little effect on micro-scheduling, while time windows spanning relatively short periods are referred to as *hard time windows*.

2.5 Inter-route Constraints

Inter-route constraints are global constraints that affect the feasibility of routes, depending on how routes are constructed. The first such class of constraints involves workload distribution in an attempt to achieve fairness in route construction. This class of constraints is usually formulated in terms of the difference between the maximum and minimum route distances/times covered by vehicles. This value should remain within a certain threshold. Balancing the workload of course also relates to time windows [428].

The second class of inter route constraints occurs when different vehicles have to compete for globally limited resources, as in [227], for example. The simplest example of this kind of constraint relates to multi-depot problems where only a certain number of vehicles can be present at a depot or involves route characteristics such as how many vehicles may cross a certain edge of the VRP graph.

The third class of inter-route constraints is concerned with the synchronisation of routes as in the *VRP with Multiple Synchronisation* (VRPMS) constraints [142].

Drexl [142] offered a classification of synchronisation-related VRP modelling constructs in terms of the following five criteria:

Task synchronisation. This notion may be considered a clustering component of a VRP as one must decide which vehicle or vehicles are assigned to a specific task. The tasks can be separated according to volume as in the SDVRP or by periods as in the PVRP, or may be transshipped between vehicles as in the *Cross-docking VRP* [288].

Operation synchronisation. This type of synchronisation occurs when different vehicles are required to perform a service, either at the same location or at different locations, but in a certain order of precedence. This notion is exemplified in [192] where a pair of service technicians perform services for each customer, with the first technician having to set up a supply at some source before the second technician can perform the service at the customer site.

Movement synchronisation. This kind of synchronisation occurs when two or more vehicles must perform an itinerary along their routes at the same time. Such a case occurs in [382], for example, where the routes of snow ploughers have to be scheduled in such a manner that they are on the same arc of the VRP graph at the same time instant.

Load synchronisation. This type of synchronisation ensures that the correct load amounts are collected, delivered, and transshipped among all vehicles at all locations and their combinations when interacting.

Resource synchronisation. This kind of synchronisation is similar to the previously mentioned inter-route constraint where, at any time instant, the consumption of resources must remain within certain capacities in order to be considered feasible.

The above-mentioned notions of synchronisation usually occur in combinations, as illustrated in [143], for example, where a vehicle transfers a portion of its load to a trailer at which time this vehicle is exclusively available for this operation and the transfer time depends on the volume of load transferred. According to Drexl [142], the most studied variants of the VRPMS are *location routing problems* and the *N-echelon VRP* [103, 343, 429], the *PDPTW with transshipments* [447], and the *simultaneous vehicle and crew routing and scheduling problem* [144, 211].

2.6 Vehicle Fleet Characteristics

A popular variation on the classical VRP is called the *multi(ple) depot VRP* [367]. In this type of VRP, a homogeneous fleet of vehicles is considered, which start and end their routes at different depots. It is quite simple to extend the standard formulation of the VRP to incorporate this characteristic. In principle, each vehicle can have a unique depot, but a limited number of vehicles are usually assigned to a depot as a result of limited capacity at the depot. In some instances, the multiple depots can act as intermediate replenishment facilities [105]. Instances of this type of VRP frequently rely strongly on the reuse of vehicles.

Another VRP variation is applicable when the fleet of vehicles is heterogeneous and is called the *Heterogeneous/Mixed Fleet VRP* (HFVRP) [26]. The HFVRP allows for vehicles that differ either in their capacities, speeds, fixed costs, variable costs or the set of customers that they can service. In this VRP, the fleet is partitioned into a number of subsets of homogeneous vehicles. This affects the model formulation in the sense that general characteristics are replaced by vehicle-specific coefficients. Many variations of this type of VRP exist in the literature, exhibiting a wide variety of characteristics. According to Toth and Vigo [428], the following vehicle fleet-related characteristics vary across formulations of VRPs in the literature:

- The vehicle fleet may either be limited or unlimited. Using sufficiently large fleet sizes, however, results in the problem simplifying to the unlimited variant.

- The fixed costs associated with the vehicles are either ignored or considered. These fixed costs are only applicable when not all vehicles are utilised. A vehicle's fixed cost is associated with the maintenance of the vehicle and the effort required to assign a driver to the vehicle. Considering this type of overhead cost may be crucial in some applications. The magnitude of fixed costs generally depends on the ownership of the fleet (*i.e.* whether the fleet is owned by the decision maker or by a third party).

- The routing costs along the edges of the VRP graph are either vehicle-dependent or vehicle-independent.

A summary of the classification of the different VRP variants mentioned above, as presented by Toth and Vigo [428], may be found in Table 2.1.

TABLE 2.1: *Heterogeneous VRP variants classification. Adapted from [428].*

Acronym	**Problem Name**	**Fleet Size**	**Fixed Costs**	**Routing Costs**
HVRPFD	Heterogeneous VRP with Fixed Costs and Vehicle-dependent Routing Costs	Limited	Considered	Dependent
HVRPD	Heterogeneous VRP with Vehicle-dependent Routing Costs	Limited	Ignored	Dependent
FSMFD	Fleet Size and Mix VRP with Fixed Costs and Vehicle-dependent Routing Costs	Unlimited	Considered	Dependent
FSMD	Fleet Size and Mix VRP with Vehicle-dependent Routing Costs	Unlimited	Ignored	Dependent
FSMF	Fleet Size and Mix VRP with Fixed Costs	Unlimited	Considered	Independent

Another fleet characteristic VRP variation accommodates the use of trailers in the formulation. The *Truck-and-Trailer Routing Problem* (TTRP) [75] allows for the use of at least two groups of vehicles, namely normal vehicles without trailers, called *Single Trucks* (STs), and *Truck-and-Trailer Combinations* (TTCs). The use of trailers is an attractive option in most VRPs due to the added capacity, but it has costs associated with it. There may also be site-dependency conditions that define whether a customer is visitable by a TTC. The visitability condition usually depends on the manoeuvrability of the TTC and/or limited space at the customer. Customers that are accessible by TTCs are referred to as regular customers in such VRP formulations, while inaccessible customers are called *truck customers*. There are three types of routes in this type of VRP. The first is a *pure ST* (where an ST visits any type of customer), the second is a *pure TTC* (where a TTC only visits regular customers) and finally there is a *mixed TTC* (where a TTC visits all types of customers but is required to decouple the trailer at appropriate customers before visiting a *truck customer* and returning to the relevant customer at a later stage to recouple the trailer).

Drexl [141] generalises the TTRP according to three characteristics. The first characteristic is whether the fixed costs associated with vehicles and trailers are considered, the second characteristic is whether there are optional locations for the temporary storage of trailers and alternate loading areas, and the third characteristic is whether or not there are time window considerations.

2.7 Problem Objectives

The objective of a VRP normally involves pure routing cost minimisation. There are, however, instances in which other objectives are considered. In fact, there are three main categories of VRP objectives according to Toth and Vigo [428], namely single-objective VRPs, hierarchical objective VRPs, and multi-criteria VRPs.

2.7.1 Single-objective Optimisation

There are numerous variations on the classical single-objective VRP. The simplest modification is the specification of desirable arcs or edges and subsequently the avoidance of undesirable arcs or edges. This modification allows the VRPB and site-dependent VRP to be transformed into a standard CVRP [428]. In another variation, selection of tasks is allowable and a profit component is added [156]. Alternatively, variable costs are included in the formulation so as to obtain a heterogeneous fleet VRP.

In the service industry, *customer satisfaction* is crucial and is often considered the overriding objective. Alternatively, this objective can be incorporated into a VRP formulation in conjunction with a cost objective by adopting a weighted latency formulation [428].

An alternative formulation is considered in the *cumulative capacitated VRP* (CCVRP), where the objective is to minimise the sum of arrival times at customers. This type of VRP is usually applied in a humanitarian context for disaster relief [325] or in cases where waiting times are undesirable in the context of customer satisfaction. An objective related to customer dissatisfaction is applied to a VRP with a soft time window in [225], where the objective is to minimise the time that every delivery request is on board a vehicle.

A common alternative objective is the adoption of a *min-max objective* in an attempt to minimise the travel distance, travel duration or workload of the busiest route. This type of objective is often incorporated as an alternative to enforcing balancing constraints, but a perfectly balanced solution often results in highly inefficient routes [428]. Generally speaking, most attributes related to utilised or consumed resources can be represented in the form of cost functions [428]. The use of *penalties* in VRP formulations also helps to guide metaheuristics in solving problem instances approximately, because these techniques often consider both feasible and infeasible solutions (in which case infeasible solutions are penalised in the objective function).

In the recent *green vehicle routing problem* [41], energy consumption and pollution emissions are considered which lead to more complex objective functions. It was shown in [41], for example, that the inclusion of energy emission and fuel consumption in the formulation significantly increases the effort required to obtain good solutions.

2.7.2 Hierarchical Objectives

In formulations of VRPs there are often conflicting objectives, such as minimising route length, route duration, delivery completion time and customer dissatisfaction. The minimisation of the number of vehicles used conflicts with these objectives and so a common hierarchical approach to solving these problems involves first minimising the number of vehicles utilised and then, with this number fixed, minimising the subsidiary objectives. In the VRPTW, a common heuristic approach is to apply hierarchical optimisation, with the first iteration focusing on minimising the number of vehicles, while the second iteration is concerned with the minimisation of the route length [428]. This approach is followed in [58], where an instance of the VRPTW is solved approximately by means of a two-stage heuristic and the results are compared with the results obtained by different heuristics. A promising solution approach in this context, in terms of both the solution time required and the objective function value, is that of Bräysy [57], who employed a two-stage local search and variable neighbourhood search algorithm.

2.7.3 Multi-criteria Optimisation

In a multi-objective optimisation problem the objective is to

$$\text{minimise } \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_d(\boldsymbol{x})) \quad (2.1)$$

$$\text{subject to } \boldsymbol{x} \in \mathcal{M}, \quad (2.2)$$

where $d \geq 2$ is the number of objective functions, $\mathcal{M}$ is the feasible solution space and $\boldsymbol{x} = (x_1, x_2, \ldots, x_\gamma)$ is the decision variable.

Multi-objective VRPs are mainly employed in three ways according to Jozefowiez *et al.* [250]: to extend classic academic problems in an attempt to improve their practical application, to generalise classic problems, and to study real-life cases where the objectives have been defined clearly by the decision maker. When extending classic academic problems, the problem definition usually remains unaltered and new objectives are simply added. The objectives thus added normally involve driver workload [282], customer satisfaction [397], and commercial distribution [370]. Another occurence of multi-objective vehicle routing optimisation arises in the generalisation of problems by adopting additional objectives instead of enforcing certain constraints. In the literature, this approach has been applied mostly in the context of the VRPTW, where the time window constraints are replaced by one or several objectives [32, 182, 181, 328, 357, 414].

Boffey *et al.* [48] mention another example of generalising a standard routing problem, called the *bi-objective covering tour problem* (BCTP) [249] in which the standard *covering tour problem* (CTP) [187] is generalised. In the standard CTP, the aim is to find a tour on a network in which the nodes visited are at most a given distance d away from the customers. In the BCTP, the parameter d is removed and replaced by an objective that aims to maximise coverage. In [372], a bi-objective model formulation is also adopted to determine a route through a subset of markets in order to collect a set of products while simultaneously minimising the travel distance and purchasing cost. Such problems are usually solved as single-objective problems in which the two objective functions are combined into a single composite function [250].

Several examples of multi-objective routing problems applied to real-life instances are described in [250]. These include the following:

Transport delivery routing. El-Sherbeny [401] solved a Belgian transportation problem in which a given amount of goods had to be delivered to a set of customers in pursuit of eight objectives.

Urban school-bus route planning. Bowerman *et al.* [52] solved a problem involving school-bus route planning for urban areas. A set of students living in different locations required access to buses that would transport them from their residences to school and *vice versa*. The authors proposed a multi-objective model formulation with four objectives, namely the minimisation of total route length, the minimisation of walking distance for the students, fair workload distribution to the drivers, and fair distribution of distances travelled by each bus.

Urban garbage collection. Lacomme *et al.* [267] solved the garbage collection problem of Troyes, where the garbage had to be collected along the city streets and delivered to a garbage treatment facility. Two objectives were considered, namely minimisation of the total route distance and minimisation of the longest route.

Merchandise transport routing. Tan *et al.* [414] solved a bi-objective routing problem for a Singapore logistics company involving the routing of truck-and-trailer vehicles that could be separated in order to reach otherwise inaccessible locations. Different time windows were considered for the customers and the vehicle fleet size was not fixed.

Hazardous product distribution. Giannikos [189] solved the problem of transporting hazardous products while considering four main objectives. The four objectives were minimisation of the total operating cost, equitable distribution of risks among city centres, minimisation of the perceived risks and equitable distribution of the disutility caused by the operation of the treatment facilities.

Tour planning for mobile healthcare facilities. Doerner *et al.* [131] attempted to solve a tour planning problem for mobile healthcare facilities associated with an emerging country experiencing population growth. The limited resources in such a context typically leads to a restricted healthcare budget and the authors proposed a cost-effective route plan for mobile healthcare facilities. Selection of the stops of the mobile facilities occurred in pursuit of the following objectives: maximisation of the efficiency of the workforce deployment and maximisation of average accessibility to the public.

Over the last several years, numerous techniques have been proposed for modelling multi-objective optimisation problems. According to Jozefiewz *et al.* [250], these techniques can be grouped into three broad categories: scalar methods, Pareto methods and methods that do not relate to either category. Scalar methods involve the use of mathematical transformations such as weighted linearisation, whereas trade-offs between the objective function values are directly compared across solutions in Pareto methods [115].

Scalar techniques exhibit several disadvantages as they usually take the form of weighted linear objective function formulations which require the objectives to be weighted according to importance. In addition, this method only facilitates determination of solutions on the convex hull of the Pareto-optimal set [188]. The advantage of this method is its simple implementation and the fact that it can be solved using any single-objective heuristic. Weighted linear aggregation has been applied to multi-objective routing problems in conjunction with problem-specific heuristics [282, 472], local search algorithms [337, 370] and genetic algorithms [328].

In an alternative scalar approach, the method of goal programming may be utilised. According to this approach, a point in the objective function space (*goal*) is chosen, and a search is conducted to minimise the distance between the current solution and the goal. The main drawback or difficulty associated with this approach is selecting the goal.

Another scalar approach is referred to as the *ϵ-constrained method* (described in some detail in [90, 334]). According to this approach, the problem is formulated as a single-objective optimisation problem and the other objectives are considered as constraints, expressed in the form $g_i(x) \leq \epsilon_i$. The ϵ-constrained method is a generalisation of Pareto outcomes as it forms a very similar formulation to the single-objective problem and therefore it can be solved by the standard *branch-and-cut algorithm* developed for single-objective optimisation problems [181].

A final scalar approach is encapsulated in the class of *Lexicographic methods*, in which each objective is assigned a priority value and the problem is then solved as a series of intermediate single-objective problems in descending order of objective function importance. Once an objective has been optimised, its value is fixed and it becomes a constraint in a new formulation of the problem.

Pareto methods were mainly introduced by Golberg [194] to be used in conjunction with genetic algorithms which apply the notion of Pareto dominance directly [250]. Although the notion of

Pareto-optimality does not allow any specific compromise or trade-offs between the objective functions to be favoured, it is a useful, flexible aid to decision makers.

Non-scalar and non-Pareto algorithms may also be found in the literature. The first such solution approach was the *Vector Evaluated Genetic Algorithm* (VEGA). It was initially proposed by Schaffer [388] and was the first use of a genetic algorithm for solving a multi-objective optimisation problem. The VEGA partitions a population of candidate solutions into d sub-populations during each iteration, where d is the number of objectives. This creates a smaller population in which genetic operators are applied.

There have also been numerous variations on well-known multi-objective metaheuristics that have been tailored to solve different variations of the VRP (elaborated upon in the following chapter).

2.8 Chapter Summary

A brief history of the celebrated classical VRP was given in §2.1, describing its origin and first real-life application. The remainder of the chapter was devoted to a description of the large number of variations on the classical VRP that have occurred in the literature since the late 1950s. The narrative was organised according to the VRP taxonomy proposed by Toth and Vigo [428]. This taxonomy includes variations based on the underlying network structure (§2.2), the type of commodity transportation requests (§2.3), the types of constraints that affect each individual vehicle's route (called *intra-route constraints*, §2.4), various *inter-route constraints* (§2.5), the types of constraints that affect the vehicle fleet composition (§2.6), and the problem objectives specified (§2.7).

CHAPTER 3

The Capacitated Vehicle Routing Problem

Contents

This chapter contains a literature review on formulations of the CVRP and solution methodologies that have been adopted to solve instances of the CVRP. These solution approaches range from exact methodologies presented in §3.2 and §3.3, respectively. These exact methodologies adopt various techniques in solving smaller instances of the CVRP, such as applying valid cuts (§3.4), pricing (§3.5) and branching and route enumeration (§3.6). Exact methodologies are typically limited to solving smaller instances of the CVRP. Accordingly, heuristics and metaheuristics capable of solving large problem instances are described in §3.7 and §3.8, respectively. Recently developed hybrid metaheuristics capable of solving large problem instances efficiently to within a few percent of optimality are presented in §3.9. The aspects to consider when selecting an approximate solution approach are described briefly in §3.10 and the chapter closes in §3.11 with a brief summary of its content.

3.1 Model formulations

This opening section contains a review of three classical model formulations of the CVRP. These formulations are the *edge-set formulation* of Laporte and Norbet [275] (§3.1.1), the *set partition formulation* of Balsinki and Quandt [29] (§3.1.2) and the *capacity-indexed formulation* of Pessoa [346] (§3.1.3).

3.1.1 The edge-set formulation of Laporte and Norbet

Let $G = (\mathcal{V}, \mathcal{E})$ be a complete graph with vertex set $\mathcal{V} = \{0, 1, \ldots, n\}$ and edge set $\mathcal{E}$, where the vertices in $\mathcal{N} = \{1, \ldots, n\}$ represent the customers and the vertex 0 denotes the depot. Let c_e denote the cost of traversing the edge $e \in \mathcal{E}$ and let q_i denote the demand of customer $i \in \mathcal{N}$. Suppose each vehicle is capacitated in the sense that it can satisfy a total demand of at most Q, and that there are K such vehicles. Given a nonempty subset $\mathcal{S} \subsetneq \mathcal{V}$, define $q(\mathcal{S}) = \sum_{i \in \mathcal{S}} q_i$ and $r(\mathcal{S}) = \lceil q(\mathcal{S})/Q \rceil$, and let $\delta(\mathcal{S})$ denote the set of edges with exactly one endpoint in $\mathcal{S}$. Finally, let x_e be a decision variable denoting the number of times the edge $e \in \mathcal{E}$ is traversed. Then the objective in the CVRP formulation of Laporte and Norbet [275] is to

$$\text{minimise} \sum_{e \in E} c_e x_e \tag{3.1}$$

subject to

$$\sum_{e \in \delta(\{i\})} x_e = 2, \qquad i \in \mathcal{N}, \tag{3.2}$$

$$\sum_{e \in \delta(\{0\})} x_e = 2K, \tag{3.3}$$

$$\sum_{e \in \delta(\mathcal{S})} x_e \leq 2r(\mathcal{S}), \qquad \mathcal{S} \subseteq \mathcal{N},\ \ \mathcal{S} \neq \emptyset, \tag{3.4}$$

$$x_e \in (0, 1), \qquad e \in \mathcal{E},\ \ \mathcal{E} \notin \delta(0), \tag{3.5}$$

$$x_e \in \{0, 1, 2\}, \qquad e \in \delta(0). \tag{3.6}$$

The objective function (3.1) is aimed at minimising the total cost associated with edges selected for inclusion in vehicle routes. Constraint set (3.2) ensures that exactly two edges incident with customer i are traversed, one to reach the customer and one to leave the customer, while constraint (3.3) ensures that edges are selected to comprise exactly K routes, one for each vehicle, with each route starting and ending at the depot. Constraint set (3.4), also referred to as the *subtour elimination constraints* in the literature, contains exponentially many constraints[1] together requiring that the number of routes entering and leaving each subset $\mathcal{S} \subseteq \mathcal{N}$ is at most the number of vehicles required to satisfy the demand of all customers in $\mathcal{S}$. This constraint set also prohibits the formation of disjoint subtours within vehicle routes. Finally, constraint sets (3.5) and (3.6) specify the integral nature of the decision variables, which are binary for all edges not incident to the depot. The reason for additionally allowing decision variables corresponding to edges incident with the depot to take on the value 2 is to accommodate potential routes from the depot which include only one customer.

3.1.2 The set partitioning formulation

Balinski and Quandt [29] proposed an alternative formulation of the CVRP for K vehicles. Let Ω be the set of all possible closed routes including the depot which respect the vehicle capacity constraint. The formulation incorporates the cost c_r of route $r \in \Omega$ and a parameter a_{ir} which denotes the number of times customer $i \in \mathcal{N}$ is visited along route $r \in \Omega$. Finally, the binary variable λ_r takes the value 1 if route $r \in \Omega$ is utilized, or the value 0 otherwise. The objective is to

$$\text{minimise} \sum_{r \in \Omega} c_r \lambda_r \tag{3.7}$$

[1] According to the Binomial Theorem [22], there are $\sum_{i=1}^{|\mathcal{N}|} \binom{|\mathcal{N}|}{i} = 2^{|\mathcal{N}|} - 1$ constraints in constraint set (3.4).

subject to

$$\sum_{r\in\Omega} a_{ir}\lambda_r = 1, \qquad i \in \mathcal{N}, \tag{3.8}$$

$$\sum_{r\in\Omega} \lambda_r = K, \tag{3.9}$$

$$\lambda_r \in \{0,1\}, \qquad r \in \Omega. \tag{3.10}$$

The objective function (3.7) again aims to minimise the total cost of all the routes selected. Constraint set (3.8) ensures that each customer is visited exactly once, while constraint set (3.9) enforces the selection of exactly K routes, one for each of the K vehicles. Finally, constraint set (3.10) ensures that the decision variables are binary.

Toth and Vigo [428] noted that one of the advantages of adopting the formulation (3.7)–(3.10) is that its linear programming relaxation often exhibits a small optimality gap. A significant disadvantage of the formulation, however, is the exponential size of the set Ω in relation to $|\mathcal{N}|$.

It was found in the 1980s, however, that the above formulation may be rendered more practical by changing the definition of Ω so as to obtain a more tractable pricing problem. The new definition of Ω is generalised to denote the set of all *walks* leaving the depot and returning to the depot in such a manner that the capacity constraint is not violated. The same customer can be visited multiple times in such a walk, but the demand satisfaction is summed depending on the number of times a customer is visited.

The model (3.7)–(3.10) is also sometimes called the *q-routes*[2] formulation of the CVRP and was used in a Lagrangian relaxation[3] solution approach developed by Christofides *et al.* [84]. The selection of the definition of Ω plays an integral part in the design of algorithms for solving the model, but even if Ω only contains elementary routes[4], the bounds provided by the above formulation are not feasible for use within an exact algorithm for large problem instances and may require reinforcement by the introduction of additional cuts [428].

3.1.3 Capacity-indexed formulation

The so-called *capacity-indexed formulation* of the CVRP was proposed by Pessoa [346], and is an extension of the formulation of the *Asymmetric CVRP* (ACVRP). Let $G_d = (\mathcal{V}, \mathcal{A})$ be a directed graph with vertex set $\mathcal{V} = \{0, 1, \ldots, n\}$ and arc set $\mathcal{A}$, where $\mathcal{N} = \{1, \ldots, n\}$ again denotes the set of customers and the vertex 0 denotes the depot. In this formulation it is assumed that the demand q_i exhibited by customer $i \in N$ is an integer in the set $\mathcal{Q} = \{1, \ldots, |\mathcal{Q}|\}$. Let c_a denote the cost of traversing the arc $a \in \mathcal{A}$, and define for any nonempty subset $\mathcal{S} \subsetneq \mathcal{V}$ the notation $\delta^-(\mathcal{S}) = \{(i,j) \in \mathcal{A} : i \in \mathcal{V}\backslash\mathcal{S}, j \in \mathcal{S}\}$ and $\delta^+(\mathcal{S}) = \{(i,j) \in \mathcal{A} : i \in \mathcal{S}, j \in \mathcal{V}\backslash\mathcal{S}\}$. That is, $\delta^-(\mathcal{S})$ denotes the set of all arcs entering $\mathcal{S}$ and $\delta^+(\mathcal{S})$ denotes the set of all arcs leaving $\mathcal{S}$. Finally, let x_a^q be a binary variable taking the value 1 if the arc $a \in \mathcal{A}$ is selected and the demand of the vertex at the end of a is $q \in \mathcal{Q}$, or the value 0 otherwise. Then the objective is to

$$\text{minimise} \sum_{a\in\mathcal{A}} c_a \sum_{q\in\mathcal{Q}} x_a^q \tag{3.11}$$

[2] A *q-route* is a route along which the total demand is exactly q, starting from the depot, passing through a subset of customers and then returning to the depot.

[3] *Lagrangian relaxation* is the approach where constraints that are hard to satisfy are allocated a weight, called a *Lagrangian multiplier*, and assigned to the objective function — a process in which the constraints in question are said to have been *dualised* into the objective function. The problem of minimising/maximising the Langrangian function of the dual variables (the vector of *Lagrangian multipliers*) is known as the *Lagrangian dual problem* whose solution provides a bound on the optimal objective function value of the original optimisation problem.

[4] An elementary route is a route in which no vertex occurs more than once.

subject to

$$\sum_{a\in\delta^-(\{i\})}\sum_{q\in Q} x_a^q = 1, \qquad i \in \mathcal{N}, \tag{3.12}$$

$$\sum\sum_{a\in\delta^+(\{0\})} x_a^q = K, \tag{3.13}$$

$$\sum_{a\in\delta^-(\{i\})} x_a^{q_i} - \sum_{a\in\delta^+(\{i\})} x_a^{q_\ell - q_i} = 0, \qquad i \in \mathcal{N},\ \ q = \{q_i, \ldots, |\mathcal{Q}|\}, \tag{3.14}$$

$$x_a^q \in \{0,1\}, \qquad a \in \mathcal{A},\ \ q \in \mathcal{Q}, \tag{3.15}$$

$$x_{\delta^+(\{i\})}^q = 0, \qquad i \in \mathcal{N},\ \ q \in \mathcal{Q}. \tag{3.16}$$

The objective function (3.11) yet again aims to minimise the total cost of all arcs included in vehicle routes. Constraint set (3.12) ensures that exactly one arc approaching customer $i \in \mathcal{N}$ is selected and that the demand exhibited by that customer is exactly one of the values in the set $\mathcal{Q}$. Constraint (3.13) ensures that exactly K arcs leaving the depot are selected, while constraint set (3.14) is a conservation of demand satisfaction flow-type constraint, ensuring that if an arc entering vertex i is selected, then an arc departing from vertex i and entering some other vertex ℓ must be selected. The arc selection must satisfy the demand of both vertices i and ℓ, and must also respect the vehicle capacities. Constraint set (3.15) ensures the binary nature of the decision variables, while constraint set (3.16) finally ensures that all routes end at the depot. The reason for interest in the capacity-indexed formulation (3.11)–(3.16), is the fact that cuts can be expressed over the variables in a robust[5] manner [428].

3.2 Classical exact CVRP solution approaches

The foundations of exact solution methodologies for instances of the CVRP were derived from the extensive work in the field of solving the TSP exactly [428], but considerable progress is still required for real-life instances of the CVRP to be solved satisfactorily (*i.e.* within reasonable time frames). Three classes of exact algorithms for the CVRP are reviewed in this section, namely the classes of *branch-and-bound algorithms* (§3.2.1), *set-partitioning algorithms* (§3.2.2) and *branch-and-cut algorithms* (§3.2.3).

3.2.1 Branch-and-bound algorithms

From the time of the early work of Christofides and Eilon [82] until the late 1980s, the most successful algorithms for the CVRP were mainly tree-search algorithms based on the *Branch-and-Bound* (BaB) method [428]. These algorithms employed techniques similar to those applied to the TSP, involving basic combinatorial relaxations[6] based either on the *Shortest Spanning Tree*[7] (SST) problem or the *Assignment Problem*[8] (AP) [428]. These approaches were limited

[5]According to the classification system proposed by de Aragao and Uchoa [112], a cut is *robust* when the value of the dual variable corresponding with it can be associated with the costs in the pricing sub-problem.

[6]These relaxations involve the exclusion of the subtour elimination constraint set (3.4) and the inclusion of an extended graph G' obtained by adding $K-1$ copies of the depot.

[7]A *shortest spanning tree* of an edge-weighted graph is a spanning tree (*i.e.* a connected, acyclic subgraph on the same vertex set as that of the larger weighted graph) whose weight (the sum of the weights of its edges) is no larger than the weight of any other spanning tree of the graph.

[8]The classical *assignment problem* is the problem of finding a minimum cost assignment of n agents to n tasks (one agent per task).

to instances with tens of customers, and through the introduction of Lagrangian relaxations these algorithms were strengthened to their best possible performances. The next step was the introduction of the notion of cutting planes[9].

There are three basic ways in which to obtain bounds for BaB algorithms in the context of the CVRP. The first such method is to generate bounds by adopting an assignment and matching modelling approach. The first occurrence of such a relaxation was a slight variation on the relaxation proposed by Little *et al.* [293] for the TSP. The corresponding relaxation for the CVRP is applicable to the standard formulation for a directed CVRP, but ignores the constraint that prohibits sub-tour formation. The resulting problem is referred to as the *Transportation Problem*[10] (TP). Infeasibility of any CVRP solution to this relaxed problem is likely to be as a result of either the vehicle capacity constraint being exceeded or there being subtours that do not visit the depot. Some of the early relaxation algorithms altered the graph of the TP by adding $K-1$ copies of the depot, where K denotes the vehicle fleet size [428]. This transformed the problem into an equivalent AP with each customer connected to one of the depot copies. This relaxation approach was first adopted by Christofides and Eilon [82] in 1969 who solved two instances of the *symmetric CVRP* (SCVRP) with six and thirteen customers, respectively, and by Laporte *et al.* [274] in 1986 who applied it to an instance of the ACVRP in conjunction with a BaB algorithm. This relaxation approach, when applied to the SCVRP, results in the so called *b-matching problem*[11]. The b-matching solution approach has been shown empirically to outperform the AP-based solution approach for instances ranging from 44 to 199 customers [425].

The second manner in which to obtain bounds is based on the use of spanning trees and shortest paths. The relaxations associated with instances of the SST problem are obtained by weakening the constraints related to the flow of vehicles through the set of vertices. The relaxed constraint only acts to impose restrictions on the connectivity of the solutions and ignores the degree requirements of the vertices. Christofides [82] was the first to attempt this relaxation approach by applying the *1-tree*[12] relaxation methods introduced by Held [226] in the context of the TSP. Fischer [164] applied this approach directly to an instance of the SCVRP, but adopted a relaxed k-tree[13] approach instead and attempted to determine a k-tree with a depot degree of $2K$ while still imposing a capacity constraint. The quality of the k-tree bound is, however, experimentally rather poor [425]. It has nevertheless successfully been incorporated into an efficient Lagrangian bound [428]. The relaxation of the SCVRP in [84], applicable to the formulation (3.7)–(3.10), employs the routes generated to determine overall bounds for the algorithm so as to improve efficiency. Adopting this approach, Christofides *et al.* [84] solved CVRP instances with 25 customers to optimality. The approach was later improved upon by Hadjiconstantinou *et al.* [213] and currently the notion of a q-route plays an integral role in the widely used class of branch-and-cut-and-price algorithms for the CVRP [428].

The final manner in which to generate bounds for BaB algorithms is by adopting Lagrangian and

[9]In mathematical optimisation, the *cutting-plane* method iteratively refines the feasible domain through the introduction of additional linear inequalities, called *cuts*, without eliminating optimal solutions.

[10]In the *transportation problem*, an optimal plan is sought for the distribution of units of products from several points of origin to several destinations.

[11]The *b-matching problem* requires the determination of the minimum cost associated with tours covering all the vertices in a manner such that the tour degree of vertex v_i is b_i, where $b_i = 2$ for each customer vertex $i \in \mathcal{N}$ and $b_i = 2|K|$ for the depot vertex $i = 0$.

[12]A *1-tree* of a graph $G = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V} = \{1, \dots, n\}$ is a shortest spanning tree on the vertices in $\{2, \dots, n\}$ together with two edges incident to vertex 1. This results in a *1-tree* having exactly one cycle, which contains vertex 1, and vertex 1 always having a degree of two.

[13]A *k-tree* is a chordal graph all of whose maximal cliques have the same size, namely $k + 1$, and all of whose minimal clique separators also have the same size, namely k [339].

additive approaches. The use of combinatorial relaxations typically result in poor results and, when used in conjunction with a BaB approach, it is limited to small instances [428]. Fischer [164] and Miller [311] adopted a Lagrangian relaxation approach toward solving the CVRP, dualising some of the relaxed constraints. Good Lagrangian multiplier values may be determined by means of standard gradient optimisation methods[14] [428]. The major difficulty associated with using Lagrangian relaxation, however, is the exponential cardinality of the set of relaxed constraints which restricts the explicit dualisation of all of the constraints into the objective function [428]. This problem may be alleviated by iteratively adding to the Lagrangian relaxation only those constraints that are violated by successive relaxation solutions [164, 311]. The approach of Fischer [164] required between 2 000 and 3 000 iterations to perform the subgradient optimisation, but resulted in the overall Lagrangian bound being significantly better than that achieved by the previously adopted K-tree relaxation approach. It allowed for instances of the data sets in [82] and [83] containing a hundred vertices to be solved to optimality within 60 000 seconds. The approach of Miller [311] produced a very tight Lagrangian bound to within 98% of the optimal objective function value for the eight test instances of the data set in [82] containing at most fifty vertices, which were each solved within 15 000 seconds.

Slight improvements on these Lagrangian bounds were reported by Martinhon [302] who combined the use of so called *comb inequalities*[15] and *multistar inequalities*[16]. Fischetti and Toth [162] achieved improvements with respect to the AP bound by combining several relaxations into an overall *additive* bounding procedure[17]. Hadjiconstantinou *et al.* [213] adopted an interesting additive approach in the context of the set partitioning formulation (3.7)–(3.10) of the CVRP together with the dual of its linear programming relaxation. They combined different relaxation approaches based upon the notions of q-routes and shortest paths to obtain feasible solutions to the dual problem which could then be used as valid lower bounds.

Almost all the branching techniques that have been applied in the literature to VRPs have their origins in studies related to the TSP. The first such technique is referred to as *branching on arcs*, and was introduced by Christofides [82] in 1969. According to this scheme, partial routes are extended, starting from the depot, until a certain vertex is reached. The problem is modelled as a TSP by eliminating the real depot and replacing it by K artificial depots, all located in the same position, where K denotes the number of vehicles deployed. The algorithm tests, before branching to a new node in the search tree, whether any constraints are violated and, if violations occur, then the branch under consideration is eliminated from the search tree. In [302], the arc selection procedure is based on the effect it has on the Lagrangian solution and when no such partial solution exists, the arc joining the customer exhibiting the largest unserved demand with the depot is selected. In most BaB algorithms for the CVRP, a *best-bound-first* search protocol is adopted as branching scheme, where branching is performed on the pending node of the branching tree that is deemed the most likely (according to an estimate) to produce the optimal solution [428].

[14] *Gradient optimisation methods* are algorithms designed in such a manner to solve problems of the form $\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$, with the search direction defined by the gradient of the function f at the current point $\boldsymbol{x} \in \mathbb{R}$.

[15] *Comb inequalities* are a highly useful cutting plane class of valid inequalities for the TSP, which involve connectivity violations on arc flows over vertex subsets (referred to as *handle* and *teeth* vertices) of customers in the graph representation of the distribution network.

[16] A *multistar* consists of the complete subgraph on a subset of *nucleus* vertices, together with a mutually exclusive subset of *satellite* vertices. Edges join every satellite vertex to every nucleus vertex. A *multistar inequality* restricts the manner in which the two subset graphs are connected by constraining the total weight of all selected edges between the respective subsets.

[17] An *additive* bounding procedure allows for the combination of different lower bounding procedures, each exploiting a different substructure of the problem under consideration. Additive bounds are considerably more effective than AP bounds when implemented in a BaB algorithm [428].

A variety of reduction rules have also been incorporated in CVRP model formulations in order to reduce the solution space and hence increase the computational speed of CVRP algorithms. These rules again have their origins in work related to the TSP. The solution space may, for example, be reduced by removing selected arcs that cannot belong to optimal solutions. Such reduction rules may either be applied directly to the original problem formulation or to solutions of subproblems in the BaB search tree. The resulting arcs define complete routes and paths, with some of them entering or leaving the depot node. Reduction of the search space occurs when complete routes of the imposed set are removed from the search space to create a reduced graph. There are generally two types of reduction rules [428]. The first type aims to remove all arcs from the graph which, if used, would produce infeasible solutions in terms of the capacity constraint associated with the vehicles. The second type of reduction rule aims to remove all arcs form the graph which, if used, would not improve the incumbent solution.

Fischetti [161] proposed a dominance test which improves the performance of the branching scheme. The dominance rule analyses a node of the BaB search tree where a partial visitation sequence of customers $i, \ldots, j$ is fixed and can be considered fathomed[18] if there exists a lower cost ordering of the customers in a visitation sequence starting with customer i and ending with customer j [428], with the improved ordering usually being determined through the application of heuristics.

3.2.2 Early set partitioning algorithms

As mentioned, the set partitioning formulation of §3.1.2 utilises an exponential number of binary variables. The definition of the set Ω is an integral part of this formulation as it implicitly considers the feasibility of routes. In attempts to reduce the cardinality of Ω, alternative formulations involve only considering maximal-feasible circuits among those with the same cost and restricting the dual solution space by only considering non-negative values.

The major drawback of the set partitioning formulation is the large number of variables required to solve the problem which, in non-tightly constrained instances with dozens of customers, can easily run into the billions [428]. A counteractive approach to this problem is to utilise a *column generation*[19] approach to solve the linear programming relaxations of these models, as described in [56]. The column generation method initially starts with a small subset of routes $\Omega' \subseteq \Omega$ and then proceeds to solve the linear relaxation of the corresponding reduced model, obtaining the optimal dual variables associated with the constraints. Utilising this dual information, the column generation problem, also called the *pricing problem*, searches for routes not in Ω' with the most negative reduced costs or proves that no such routes exist [428]. If no such route exists, the current solution to the problem is optimal and the procedure terminates. Otherwise, if a route is found, a new iteration is performed. The bounds produced by this approach are typically extremely tight [428], which has led to extensive research in this field resulting in superior *Branch-and-Cut-and-Price* (BCP) algorithms being developed.

[18] A node in a BaB search tree is *fathomed* if it represents a feasible solution to the original problem or can be eliminated because of knowledge that further branching from the node cannot lead to an optimal solution.

[19] *Column generation* exploits the notion that most variables will be non-basic (and hence assume a value of zero) in an optimal solution. Therefore only a subset of variables need to be considered when solving the optimisation problem. Column generation leverages this notion, generating only the variables which have the potential to improve the objective function.

3.2.3 Branch-and-cut algorithms

Branch-and-Cut (BaC) algorithms are based on the seminal work by Laporte [276] in which he considered a relaxation of the standard undirected CVRP with the capacity constraints and the integrality of variables removed. A solution to such a relaxation of the CVRP is either feasible for the standard CVRP, in which case the algorithm terminates, or otherwise the solution is infeasible, in which case the algorithm continues to consider alternative solutions.

Within a BaC algorithm a large variety of cuts can be applied. These cuts include:

- *TSP-related valid inequalities.* A first attempt to generate valid inequalities involved generalising constraints that were initially developed for the TSP.
- *Capacity constraints.* The computation of the value of $r(\mathcal{S})$ in (3.4) determines the set to which the inequality belongs. If the smallest value for the quantity $r(\mathcal{S}) = \sum_{i\in\mathcal{S}} q_i/Q$, is considered, the inequalities are referred to as *fractional capacity inequalities.* When $r(\mathcal{S}) = \lceil\sum_{i\in\mathcal{S}} q_i/Q\rceil$ is considered, the inequalities are referred to as *rounded capacity inequalities.* The constraints are referred to as *weak capacity inequalities* when $\mathcal{S}$ is given and a legitimate value for $r(\mathcal{S})$ is the optimal value for the corresponding bin packing problem. Finally, when $r(\mathcal{S})$ is equal to the minimum number of vehicles required to service all vertices of $\mathcal{S}$, the inequalities are referred to as *global capacity constraints.*
- *Framed capacity inequalities.* First introduced by Augerat [18], these constraints are an extension of the weak generalised capacity constraints described above and produce a lower bound on the number of vehicles required to service the set of customer vertices.
- *Comb inequalities.* Introduced by Chvátal [86] and Padberg [209] for the symmetric TSP, these inequalities may be transformed into tight triangular form and may then be used as valid inequalities for the CVRP [428]. Comb inequalities are defined in terms of certain vertex sets called the *handle* and the *teeth*, as described in Footnote 15.
- *Hypotour inequalities.* Sets of constraints aimed at determining subgraphs of G which cannot include feasible solutions to the CVRP [428].
- *Multistar inequalities.* Introduced by Hall [13] for the CVRP with unit demands, see Footnote 16.

Ralphs *et al.* [358] proposed a BaC algorithm that separates the capacity constraints of the CVRP by means of three heuristics. If the heuristics fail to identify a violated inequality, they proposed the use of a decomposition algorithm for determining additional constraints. The first stage involves expanding the original problem instance graph through the addition of $|K| - 1$ copies of the depot with corresponding edges. According to the fractional solution obtained from the extended graph, a decomposition algorithm determines whether the solution can be written as a convex combination of Hamiltonian cycles[20] [428]. If so, these Hamiltonian cycles are analysed to find violated capacity constraints. If not, the branching stage is initiated. Finally, when the fractional solution can no longer be decomposed, a *Farkas inequality*[21] is created. The

[20] A Hamiltonian path is a path in a (directed or undirected) graph that visits each vertex exactly once. A Hamiltonian cycle is a closed Hamiltonian path.

[21] A *Farkas inequality* states that when considering two subsystems, one or the other contains the solution, but not both or none. In mathematical terms, let $\boldsymbol{A}$ be a real $m \times n$ matrix. Let $\boldsymbol{b}$ be an m-dimensional real vector. Then, exactly one of the following statements is true:

1. There exists an $\boldsymbol{x} \in \mathbb{R}^n$ such that $\boldsymbol{Ax} = \boldsymbol{b}$ and $\boldsymbol{x} \geq \boldsymbol{0}$.
2. There exists a $\boldsymbol{y} \in \mathbb{R}^m$ such that $\boldsymbol{y}^T\boldsymbol{A} \geq \boldsymbol{0}$ and $\boldsymbol{y}^T\boldsymbol{b} < 0$.

first stage of the decomposition algorithm hinges on *a priori* knowledge of Hamiltonian cycles. Then an enumerative search is performed to determine a preset number of cycles. Following the enumerative search, a column generation algorithm is initiated to create additional cycles dynamically.

Lysgaard *et al.* [297] proposed a new BaC algorithm. The algorithm involves four novel separation techniques for the generation of valid inequalities: rounded capacity constraints[22], strengthened comb inequalities, framed capacity inequalities[23] and hypotour inequalities[24]. The algorithm uses techniques introduced by Letchford *et al.* [283] to separate homogeneous multistar and partial multistar inequalities. In addition, it also employs mixed integer Gomory cuts[25] to perturb the current fractional solution at the root node [428].

The BaC algorithm of Achuthan *et al.* [1] employs separation of rounded capacity constraints through the use of heuristics and was later improved upon by Achuthan *et al.* [2] who separated additional inequalities related to multistar constraints. Baldacci *et al.* [27] presented a new integer programming formulation derived from a two-commodity network modelling approach. Since the flow variables in such an approach can be expressed in terms of arc variables, all the constraints of the standard CVRP are still applicable to the new model. Baldacci *et al.* [27] separated four families of inequalities: rounded capacity inequalities, hypotour inequalities, comb inequalities and generalised capacity constraints.

3.3 More recent exact approaches

Since the pioneering work of Desrosiers *et al.* [125], column generation has been the favoured approach for designing exact algorithms for the VRPTW. This methodology has performed exceptionally well in dealing with instances involving hard time windows. In the case of soft time windows, the problem can be considered as a CVRP in which the column generation technique under-performs. The favoured technique for exactly solving instances of the CVRP during the early 2000s was BaC algorithms that separate complex families of cuts by polyhedral investigation [319]. Such algorithms worked well for small problem instances, but instances of larger than 50 customers were not thus solvable.

Then Fukasawa *et al.* [174] developed an algorithm utilising a combination of column generation and cut generation called a BCP algorithm. The algorithm employs robust cuts. This allows the structure and size of the sub-problem to remain unchanged [428]. *Non-robust cuts* alter

[22] *Rounded capacity constraints* were introduced by Naddef and Rinaldi [320]. Determining $r(\mathcal{S})$ in (3.4) is NP-Hard, since it is equivalent to finding an optimal solution to the two-dimensional *bin packing problem.* By replacing $r(\mathcal{S})$ with $k(\mathcal{S}) = \max\{\lceil d(\mathcal{S})/D\rceil, \lceil a(\mathcal{S})/A\rceil\}$, where D and A represent the respective capacities and $a(S)$ denotes a subset such that $a(S_i) \leq A$ for each cluster S_i, a valid lower bound is obtained. These constraints are known as rounded capacity cuts.

[23] *Framed capacity inequalities* are a highly successful cutting plane class of valid inequalities for the CVRP, and may be defined for some $\mathcal{S} \subseteq \mathcal{N}$ as

$$x(\delta(\mathcal{S})) + \sum_{i=1}^{p} x(\delta(\mathcal{S}_i)) \geq 2r(\mathcal{S}, \Omega) + 2\sum_{i=1}^{p} r(\mathcal{S}_i),$$

where $\Omega = (\mathcal{S}_1, \ldots, \mathcal{S}_p)$ is a partition of $\mathcal{S}$ and $r(\mathcal{S}, \Omega)$ is the minimum number of vehicles required to service $\mathcal{S}$, given that the capacity inequality for each subset $\mathcal{S}_i$ holds with equality [297].

[24] The hypotour inequality for some set $\mathcal{S} \subset \mathcal{E}$ requires that at least one edge in $\mathcal{S}$ appears in the feasible CVRP solution [297].

[25] A *Gomory cut* [200] is a linear constraint with the property that it is strictly stronger than its parent. This is achieved by enforcing appropriate integer-linear forms so as to generate valid linear inequalities that cut off an undesirable fractional solution from the relaxed solution space.

the size or structure of the sub-problem, and each additional cut therefore renders the problem harder, which makes robustness a desirable attribute. Baldacci *et al.* [27] and Jepsen *et al.* [245], however, showed that non-robust cuts can be used effectively if they are separated in a controlled way, avoiding an excessive impact on pricing.

Fukasawa *et al.* [174] presented a BCP algorithm that is able to solve various benchmark instances containing up to 135 customers to optimality. The columns included by the algorithm are linked by q-routes which avoid k-cycles. The algorithm is therefore a relaxation of elementary routes that allow multiple visits to a customer on condition that at least k distinct customers are visited between successive visits of the same customer. The separated cuts imposed are robust with respect to q-route pricing and the algorithm is able to identify when column generation at the root node is too slow, in which case it automatically switches to a BaC paradigm.

Baldacci *et al.* [27] also developed a CVRP solution approach based on column and cut generation. The algorithm's columns are linked with elementary routes. The cuts are effective, but non-robust. Strengthened capacity cuts and clique cuts[26] are, however, separated, which makes the pricing harder. A sequence of cheaper lower bounding procedures produces good estimates of the optimal values of the dual variables. The expensive pricing procedure is only dealt with during the last stage, with the dual variable achieving a good upper bound convergence. The algorithm does not branch as it finishes at the root node by enumerating all elementary routes with reduced costs smaller than the duality gap [428]. The set-partitioning problem containing all these routes is then passed on to a *Mixed Integer Programming* (MIP) solver. The algorithm is able to solve almost all the CVRP instances covered in [174], usually in much less time, but due to the exponential aspect of some of the constraints it has failed in some instances, assigning large numbers of customers to single vehicles.

Pessoa *et al.* [346] developed an improved version of the algorithm proposed by Fukasawa *et al.* [174]. The cuts form an extended formulation which incorporate capacity indices, as in the formulation (3.11)–(3.16), and are also separated. Through the use of dynamic programming, the complexity of forming *q-routes* is not altered. Pessoa *et al.* [346] borrowed the idea of utilising route enumeration in combination with an MIP solver from Baldacci *et al.* [27], with the added feature of using a hybridisation in conjunction with traditional branching so as to avoid premature failure when the root gap is too large.

Baldacci *et al.* [28] also introduced an improved version of their original algorithm [27]. A new relaxation (called *ng-routes*[27]) was introduced which is superior to the notion of *q-routes* without the use of *k-cycles*. This relaxation was employed in the earlier bounding procedure and was used to accelerate the pricing of elementary routes. The latter part of the algorithm is also further enhanced through the consideration of multiple dual solutions. Subset row cuts[28] and weak subset row cuts[29] are separated, which has a reduced effect on pricing compared to when clique cuts are generated. The end result is a faster and more robust algorithm than that proposed in [27], which has the ability to solve instances with a large number of customers per vehicle.

[26] A *clique cut* is a complete subgraph of some given graph. A clique cut is a relationship among a set of binary variables with the property that at most one variable in the group can be positive in any integer feasible solution. Clique cuts construct a graph respecting this property and through optimisation methods finds maximal cliques in the graph.

[27] *ng-Routes* are used to solve the pricing subproblem. They are elementary forward routes which are used to determine paths for vehicles to follow.

[28] Introduced by Jepsen *et al.* [245], *subset row cuts* are a family of inequalities that are specifically linked to the rows of the set packing problem. A set packing problem seeks to ascertain whether there is some collection of k subsets within a list of subsets, all from a given set $\mathcal{S}$, that are pairwise disjoint (no two of them share an element).

[29] *Weak subset row cuts* are weak dominance criteria employed when using label-setting algorithms to solve the pricing problem.

Contardo [96] utilised non-robust cuts and route enumeration in a novel manner. The columns in his approach are associated with *q-routes* without 2-cycles, which yields a relatively poor relaxation, while the partial elementarity of the routes is enhanced by non-robust strong-degree cuts[30]. Subset row cuts, robust cuts from the set partitioning formulation (3.7)–(3.10) and strengthened capacity cuts are also separated. The enumeration of all the elementary routes is aimed at the generation of a pool of columns until the duality gap is sufficiently small to produce a pool of reasonable size (the pricing starts through inspection). At this stage, an aggressive separation of non-robust cuts takes place, resulting in very small optimality gaps. This yields computational results that are reportedly very consistent and the approach has been used to solve instances with 151 customers and 12 vehicles [96].

Røpke [378] improved on the algorithm of Fukasawa *et al.* [174] by incorporating two key differences. The first difference is that more effective *ng-routes* are utilised as opposed to *q-routes* avoiding *k-cycles*. The second key difference is that a highly sophisticated and aggressive branching procedure is performed, resulting in drastically reduced enumeration trees. The algorithm yields similar results to those reported in [28] and [96]. Running the algorithm for a long time, it was shown to be able to solve CVRP instances with 151 customers and 12 vehicles to optimality.

Contardo and Martinelli [96] improved upon the work of Contardo [96] through the use of *ng-routes* instead of *q-routes* avoiding 2-cycles and through the application of the *decremental state space relaxation*[31] technique introduced by Righini and Salani [374]. The performance of the dynamic pricing scheme was improved and the authors incorporated the method proposed in [239], which results in the edge variables being fixed through reduced costs.

Pecin *et al.* [340] developed a BCP algorithm that incorporates features from all the above-mentioned approaches. The algorithm is able to solve CVRP instances of up to 200 customers and 16 vehicles to optimality within a reasonable time. The introduction of a limited memory subset row cut is the most important novel feature of this algorithm as it weakens the traditional subset row cuts and can be altered dynamically. This results in the cuts being considerably less costly during pricing without reducing their effectiveness. The method utilises capacity indices from [346] which allows for fixing of variables by reduced costs, thus improving on the results reported in [239]. The columns in the BCP algorithm are associated with *ng-routes* and column generation stabilisation can also be implemented through dual smoothing [344]. The dynamic programming pricing utilises bi-directional searches, which is a slight variation on the method proposed by Righini and Salani [373], as the concatenation phase is not necessarily performed at half capacity. The algorithm performs aggressive hierarchical, strong branching which may be seen as a hybridisation of route enumeration and branching. The algorithm is able to perform a rollback and remove the offending cuts (cuts that are responsible for making the algorithm slower) when a round of non-robust cuts results in the pricing taking too long.

3.4 The generation of valid cuts

Various families of inequalities that have been used to reinforce the *edge-set formulation* of §3.1.1 and the *set partitioning formulation* of §3.1.2 are described in more detail in this section.

[30]Introduced by Contardo *et al.* [95], *strong degree cuts* are a family of valid inequalities that have been proven to impose partial elementarity.

[31]Introduced by Righini and Salani [374], *decremental state space relaxation* iteratively reduces the relaxation of the solution space according to the structure of the optimal solution of the relaxed problem.

3.4.1 Cuts over the edge variables

Besides the *rounded capacity cuts* in (3.4), numerous families of valid CVRP cuts have been introduced over the edge variables. The paper [296] contains a reference to the package CVRPSEP of heuristic separation techniques for the families of framed capacities, rounded capacity, multistars, strengthened combs and extended hypertours. These families play a significant role in BaC algorithms for the *edge-set formulation* of §3.1.1 but only rounded capacity cuts and strengthened comb inequalities can improve the *set partition formulation* of §3.1.2 [428]. The reason for the lack of improvement by the other families in the latter CVRP formulation is that the other families are already included implicitly in (3.8) and (3.9). Letchford and Salazar-González [284], for example, proved that all generalised large mutlistar cuts are directly implied by (3.8) and (3.9), even if the definition of Ω includes all *q-routes*.

3.4.2 Strengthened capacity cuts

Baldacci *et al.* [27] introduced a family of cuts which is defined over the variables in the *set partitioning formulation* of §3.1.2. For every nonempty set $\mathcal{S} \subseteq \mathcal{N}$ and $r \in \Omega$, a binary variable $\zeta_{\mathcal{S}r}$ takes the value 1 when at least one vertex is visited along route r within the set $\mathcal{S}$. The *strengthened capacity cuts* are

$$\sum_{r\in\Omega} \zeta_{\mathcal{S}r}\lambda_r \geq r(\mathcal{S}), \quad \mathcal{S} \subseteq \mathcal{N}. \tag{3.17}$$

Constraint set (3.17) monitors the flow of routes and ensures that all customers are visited at least once. It is a strong inequality as it is not deceived by routes that enter and leave $\mathcal{S}$ more than once [428]. Strengthened capacity cuts are non-robust in nature as they alter the pricing sub-problem and it is therefore necessary to continue solving it through dynamic programming in order to add an additional binary dimension that monitors whether a partial route has already visited $\mathcal{S}$ or not.

3.4.3 Subset row cuts

Jepsen *et al.* [245] introduced a family of cuts defined over the variables of the *set partitioning formulation* of §3.1.2. Given a set $\mathcal{C} \subseteq \mathcal{N}$ and a multiplier p satsfying $0 < p < 1$, a $(\mathcal{C}, p)$-*subset row cut* has the form

$$\sum_{r\in\Omega} \left\lfloor p \sum_{i\in\mathcal{C}} a_{ir} \right\rfloor \lambda_r \leq \lfloor p|\mathcal{C}| \rfloor , \tag{3.18}$$

where a_{ir} is the number of times customer i is visited by route r, and is valid as it can be achieved by a Chvátal-Gomory rounding[32] that corresponds to (3.8) [428]. There is a special class of subset row cuts called 3-*subset row cuts*, where $|\mathcal{C}| = 3$ and $p = 1/2$. In the more recent literature this class is favoured above the more general clique cuts employed by Baldacci *et al.* [27] as they are more suitable to column generation, which has a less pronounced impact on the pricing sub-problem [428]. With that said, 3-subset row cuts still require an additional binary dimension that indicates the parity of the number of visits made to a node in $\mathcal{C}$.

[32] *Chvátal-Gomory rounding* is the procedure of applying Chvátal-Gomory cuts to valid inequalities. A Chvátal-Gomory cutting plane for a problem $\mathcal{P}$ is an inequality of the form $\boldsymbol{c}^T\boldsymbol{x} \leq \lfloor\delta\rfloor$, where $\boldsymbol{c}$ is an integral vector and $\boldsymbol{c}^T\boldsymbol{x} \leq \delta$ is valid for $\mathcal{P}$ [154].

3.4.4 Strong degree cuts

Contardo *et al.* [94] introduced a family of cuts related to strengthened capacity cuts over a set $\mathcal{S}$ of cardinality 1 referred to as *strong degree cuts*. Given a vertex $i \in \mathcal{N}$ and a route $r \in \Omega$, a binary variable coefficient ζ_{ir} is defined to take the value 1 if r visits i. The corresponding degree cut takes the form

$$\sum_{r \in \Omega} \zeta_{ir} \lambda_r \geq 1. \tag{3.19}$$

For these cuts, the definition of Ω has to allow for the inclusion of non-elementary routes. This results in the strengthened degree cuts forbidding routes with cycles containing certain vertices.

3.4.5 Limited memory subset row cuts

The use of limited memory subset row cuts was first introduced in [340] and is a generalisation of the notion of $(\mathcal{C}, p)$-subset row cuts. It requires an additional set $\mathcal{M}$ satisfying $\mathcal{C} \subseteq \mathcal{M} \subseteq \mathcal{N}$, and can be written as

$$\sum_{r \in \omega} \alpha(\mathcal{C}, \mathcal{M}, p, r) \lambda_r \leq \lfloor p|\mathcal{C}| \rfloor, \tag{3.20}$$

where the coefficient α is a function of $\mathcal{C}, \mathcal{M}, p$ and r, and is computed using another algorithm. The advantage of employing limited memory-subset row cuts over classical subset row cuts is their reduced impact on the labelling algorithms used in the pricing scheme [428]. In practical instances, even with hundreds of customers, the minimal sets of routes seldom have a cardinality exceeding 15. Pecin *et al.* [340] used a two-phase strategy to obtain small memory sets. The first stage involves identifying violated $(\mathcal{C}, p)$-subset row cuts, with the second stage identifying a minimal set $\mathcal{M}$ such that limited memory subset row cuts achieve the same violation.

3.5 Pricing approaches

Different approaches that have been implemented to handle the pricing subproblem occurring when attempting to solve an instance exactly, are described in more detail in this section.

3.5.1 q-routes and elementary routes

The pricing sub-problem for solving the linear relaxation of the set partitioning formulation of §3.1.2 by column generation has been modelled numerous times as a *Shortest Path Problem with Resource Constraints* (SPPRC). The problem is defined over a directed graph, with an unrestricted cost function. The objective is to find a shortest path whose resource consumptions do not exceed the capacity. The pricing of q-routes corresponds to a SPPRC with a single resource, but the pricing of elementary routes requires the definition of n additional resources. When pricing an SPPRC, the approach is to create a label, which denotes the vertices visited along a route, an approach first introduced by Desrochers *et al.* [121]. These labels are created iteratively and appended to existing labels only if all the resource constraints are still satisfied.

The main concern when designing a forward dynamic label setting algorithm for the SPPRC is how to mitigate the efficiency deterioration due to the growth of the list family. Executive decisions therefore have to be made at each iteration, such as keeping the path with the smaller cost of two paths ending in the same vertex. When pricing q-routes, the basic dominance rule is enough to keep the total complexity of the problem pseudo-polynomial. For elementary routes, however, the complexity is exponential [428]. In attempts at reducing the complexity of pricing elementary routes, the following ideas have been put forward:

- *Incorporating stronger dominance rules.* The basic dominance rule used in earlier labelling algorithms can be improved by considering the monotonic effect of resource consumption in possible extensions of the label [428]. This approach was shown in [73] to achieve an improvement over the basic dominance rule by considering the expected improvement of reduced costs that could be experienced by traversing certain nodes.

- Implementing a *bidirectional search.* The ratio of customers per vehicle is a strong indicator of algorithmic performance. In an attempt at better exploiting this fact, Righini and Salani [373] reduced the capacity constraint by half, which results in a smaller family of lists. The final routes can be created during a concatenation phase through the exploitation of the symmetry of the CVRP. An adverse effect of this approach is that the concatenation phase is quite costly in itself and has not been proven to be better than a conventional forward search.

- Introducing *completion bounds.* If a lower bound, $B(L)$, can be imposed on the costs of all the routes that can be obtained through extending a list L and $B(L) > 0$, then the list L can be removed from the family.

- Employing *decremental state space relaxation.* First proposed by Boland *et al.* [49], this technique starts by only considering those *q-routes* which are likely to contain cycles. The vertices that are visited numerous times are included in the next iteration of the label setting algorithm and as the algorithm progresses, the set of customers that are not visitable increases until elementary routes are constructed.

3.5.2 q-Routes with k-cycle elimination

The basic concept behind this approach is that a vertex i cannot be revisited unless k other customers have been visited between successive visitations of vertex i. It was realised early on by Christofides *et al.* [85] that through the use of 2-cycle elimination, the definition of Ω significantly improves the bounds generated by set partitioning relaxation with a negligible effect on the complexity of the sub-problems that result. When pricing ordinary q-routes, the pricing algorithm utilises so-called *buckets* in which, through basic dominance rules, instances are limited to at most one label. With 2-cycle elimination, however, the buckets can hold two labels. When performing k-cycle elimination for $k \geq 3$, the label storage decision requirements become more intricate. Algorithms have been proposed for this purpose by Hoshino and de Souza [233], and by Irnich and Villeneuve [240]. In [240], the number of labels kept within the buckets was limited to a maximum of $k!$, but the number of labels kept during a run exhibited high variation. In [233], a more efficient label selection procedure was proposed, involving the use of deterministic finite automata. For certain instances, 4-cycle elimination is not sufficient to obtain bounds close to the optimal solution and it was suggested in [174] that k-cycle elimination above the value $k = 4$ is not feasible.

3.5.3 ng-Routes

A simple alternate approach to cycle-elimination for obtaining partial elementarity is to select a subset $\mathcal{S}$ of customers that cannot be revisited. This label-setting algorithm is limited to instances where $\mathcal{S}$ has a size similar to the size of $\mathcal{N}$. Otherwise it is not likely to produce near-elementary routes. Baldacci *et al.* [28] proposed a more efficient manner to obtain partial elementarity in routes by exploiting the fact that cycles are likely to appear when pricing non-elementary routes [428]. They used a more limited memory mechanism that only considers edges associated with a small cost and is restricted to relatively small neighbourhoods. This allows for cycles to be constructed over a vertex only if certain constraints are maintained. The application of ng-routes is pseudo-polynomial. In order to deal with large instances, Pecin *et al.* [341] proposed an adaptation of decremental solution space relaxation with the initial stage utilising ng-routes. This adaptation is able to deal with instances of up to 200 customers.

3.5.4 Pricing with non-robust cuts

The variations of label-setting algorithms for dealing with non-robust cuts require an additional dimension in the labels, typically one dimension per cut [428]. For the SPPRC, for instance, the additional definition is monotonically increasing along a path. In the case of subset row cuts, however, the dimensions related to cuts never inhibit path extensions and are used more commonly to reward or penalise certain path extensions. The dominance rules associated with non-robust cuts also need to be modified. These modifications include adding two terms to a simple cost inequality. The terms added are lower bounds that monitor the potential gains resulting from extending a path in a certain manner. Contardo *et al.* [94] showed that the imposition of non-robust cuts may be a more efficient manner by which to impose route elementarity than the conventional definition of resources avoiding vertex revisitations.

3.5.5 Column generation concerns

Column generation is normally used to solve linear programs with large numbers of variables. The efficiency of the method is related to how many variables it is expected to deal with. For the CVRP *set partitioning formulation* of §3.1.2, for instance, the number of variables is approximately equal to $\binom{n}{n/K}$ [428]. The expected number of iterations before convergence on pricing is therefore highly dependent on the number of customers per route. In instances with a large value of n/K, additional treatments are typically required, such as those mentioned in [147, 344], where a form of dual stabilisation is used to assist convergence of such problems. In any case, regardless of the number of customers per route, it is advisable to implement faster pricing heuristics when utilising column generation. Such heuristics mainly involve altering the label-setting algorithm in three main ways. The so-called *scaling* technique involves running the label-setting algorithm with a factor g. The demand and capacity are divided separately by g. The *sparsification* technique consists of only selecting possible extensions from a subset of edges that are likely to result in routes with negative reduced cost. The technique of *bucket pruning* involves only storing a small set of labels per bucket which means that many non-dominated solutions that are unlikely to result in optimal solutions, are not considered.

3.6 Branching vs. route enumeration

Two main approaches employed in the literature towards reducing the duality gap during solution of the CVRP are compared in this section. These approaches are typically applied either because it is no longer possible to find additional violated cuts or because very incremental improvements in the lower bound are observed. When imposing non-robust cuts, additional separation may result in excessively expensive pricing subproblems [428]. Various branching and enumeration methodologies put forward in recent articles are discussed in this section.

3.6.1 Branching

Branching can be applied to both the *edge-set formulation* of §3.1.1 and the *set partitioning formulation* of §3.1.2, but directly applying it to the *set partitioning formulation* may lead to the formulation of non-robust cuts. This can be avoided by applying the branching technique on the edge variables of the *edge-set formulation* and employing branching constraints to create robust cuts for the *set partitioning formulation*.

Fukasawa *et al.* [174] implemented branches over sets in a BCP context, with the sets being selected through strong branching. Every set in a collection of candidate sets is evaluated heuristically by applying a small number of column generation iterations to its children nodes. In [374], a much better BCP is utilised in which simpler branching occurs over individual edges. The strong branching phase initially starts by assessing thirty candidate edges and then ranking them. The best candidate edges are then fully evaluated and, if successful, a candidate induces the incumbent solution. The remaining edges achieving a high probability of improving the incumbent solution are then better evaluated. This strong process of branching is the key to the success of the approach and is aided by the collection of statistics along the enumeration tree, allowing for the solution of an instance with 151 customers and 12 vehicles.

Despite the success of strong branching BCP, it still remains very pricing-intensive, even with the use of smart accelerating ideas [428].

3.6.2 Route enumeration

Baldacci *et al.* [27] utilised route enumeration in an attempt to reduce the duality gap after the root node. The key facet of their approach is that a route can only be part of a solution that improves the upper bound if:

- its reduced cost is smaller than the gap, and
- there are no other routes visiting the same vertex which achieve a smaller cost.

The enumeration can then be performed by a label-setting algorithm producing a set $\mathcal{R} \subset \Omega$. Through the use of an MIP solver limited to $\mathcal{R}$, the enumeration then proceeds subject to a known upper bound implemented as a cut-off. The next stage determines whether the restricted path is infeasible or not. If infeasible, it proves that the solution attaining the upper bound is optimal.

The efficiency of the label-setting algorithm depends on the cardinality of the set $\mathcal{R}$, and this cardinality depends, in turn, on the duality gap and the number of customers per route. For small ratios of n/K, the set $\mathcal{R}$ is likely to be small, even with a large associated gap, and in general these cases allow for sets of no larger than a few tens of thousands of routes to be created. This allows for the resulting restricting model to be solved very timeously.

The above-mentioned enumeration process was improved upon by Baldacci *et al.* [28], who considered two dual solutions with associated lower bounds. The first dual solution is obtained earlier in the algorithm, while the second dual solution relates to the final solution at the root node.

A different improvement approach was followed by Contardo [96], who tested at intervals whether the current solution is likely to lead to a set of fewer than five million routes. If the test is positive, the set is stored in a pool, and column and cut generation proceeds. From this point onwards, the pricing procedure proceeds by applying a straightforward inspection as opposed to employing a conventional label-setting algorithm. This allows for aggressive separation of non-robust cuts and after each improvement of the lower bound, the pool is reduced by performing reduced cost fixing. In most instances, the column and cut generation algorithm terminates with a zero gap, but if this is not the case, the corresponding pool is sent to an MIP solver.

3.6.3 Hybridising branching and route enumeration

Route enumeration plays an integral role in some of the best-performing CVRP algorithms, but it is still an inherently exponential space procedure that is susceptible to failure when applied to larger/harder instances. The hybrid strategy applied in [346] and [347] performs route enumeration until a limit of 80 000 iterations is reached, after which a BCP algorithm continues *via* branching.

In a more sophisticated approach by Pecin *et al.* [340], route enumeration is attempted on every node v, allowing for large sets to be analysed. The procedure is structured in the following manner:

- If the enumeration is successful, the node v continues to be considered by pricing through inspection. An unlimited number of subset row cuts and clique cuts are separated and routes are eliminated through fixing by reduced costs. Once the final set has fewer than 20 000 routes, the node is considered further by an MIP solver, while if the set is too large, the BCP algorithm performs branching.
- The enumeration in most instances is successful, but if the enumeration fails, aggressive hierarchical strong branching is performed [428].

3.7 Classical heuristics

Recent development of decomposition algorithms has allowed for CVRP instances of roughly a hundred customers to be solved to optimality according to varying time requirements. Real-life CVRP instances, however, often require much larger numbers of customers to be accommodated subject to predictable time requirements, which calls for the efficiency of heuristics. Furthermore, exact algorithms are typically very problem-specific and the development of flexible heuristics that are able to handle a variety of objectives and side constraints is therefore a desirable prospect.

VRP heuristics are almost as old as the problem itself. In the seminal paper by Dantzig and Ramser [107], the authors introduced a simple heuristic based on successive matchings of vertices through the solution of linear programs and the elimination of fractional solutions by trial and error [428]. Since then a wide variety of *constructive* and *improvement heuristics* have been proposed, culminating in the recent development of powerful metaheuristics that are able to compute solutions within seconds that lie within less than one percent of the best known objective function values for relatively large problem instances [428]. Metaheuristics have dominated the research evolution of approximate VRP solution techniques over the past ten years, with the notion of hybridisation at the forefront of this evolution.

3.7.1 Broad overview

Classical heuristics may be classified into three broad categories, namely [425]:

Constructive heuristics, which do not include an improvement phase, but rather iteratively build a feasible solution while continually considering solution cost.

Two-phase heuristics, which involve decomposing the problem into its two natural components, route construction and feasible clustering of vertices into routes. The *cluster-first, route-second* paradigm involves first clustering the vertices into feasible clusters and then constructing vehicle routes from the clusters. Conversely, a route is constructed on all vertices and then segmented into feasible routes respecting the vehicle capacities according to the *route-first, cluster-second* paradigm.

Improvement methods, which aim to enhance any feasible solution by performing ordering exchanges on a sequence of edges and vertices.

Examples of members of the three categories of heuristics described above are reviewed in the remaining subsections of this section. The distinction between improvement and constructive methods is, however, often blurred as most constructive approaches contain an improvement stage at some point [425]. Most of the heuristics developed for VRPs also apply directly to the CVRP in particular, and they are usually able to accommodate an unspecified number of vehicles.

3.7.2 Constructive heuristics

The two main techniques applied in the class of constructive heuristics involve either merging together existing routes iteratively by utilising a *savings* criterion or gradually assigning vertices at an insertion cost to smaller, existing vehicle routes.

3.7.2.1 The Clarke and Wright savings algorithm

The seminal paper by Clarke and Wright [87] contains the proposal of perhaps the best-known heuristic for the VRP. The algorithm is based on the notion of savings incurred by including an additional vertex in an existing route. The algorithm is usually applied to instances where the number of vehicles is a decision variable and works equally well for instances on directed or undirected graphs. The algorithm works as follows:

Step 1 Calculate the savings $s_{ij} = c_{i0} + c_{0j} - c_{ij}$ for $i, j = 1, \ldots, n$ and $i \neq j$, where 0 denotes the depot. Construct n initial vehicle routes (each route containing only one vertex and the depot) and then order the savings in a nonincreasing manner.

Step 2 There are two main paradigms that can be applied in Step 2, namely:

- A parallel version: Starting from the beginning of the savings list, do the following: Given a savings value s_{ij}, determine whether there exist two routes, one containing the arc $(0, j)$ and the other containing the arc $(i, 0)$, that can feasibly be merged. If so, merge these two routes by removing $(0, j)$ and $(i, 0)$, and introducing (i, j), as illustrated in Figure 3.1.
- A sequential version: Consider sequentially every route $(0, i, \ldots, j, 0)$, determining a savings value s_{ki} or $s_{j\ell}$ that can feasibly be utilised to combine the current route with a separate route containing either the arc $(k, 0)$ or the arc $(0, \ell)$. If possible, implement the merge and repeat this operation with respect to the newly formed route. If no feasible merges remain, consider the next route and repeat the same procedure. Stop when no feasible route merges remain.

Toth and Vigo [425] showed that the parallel version dominates the sequential version, producing better results in less time.

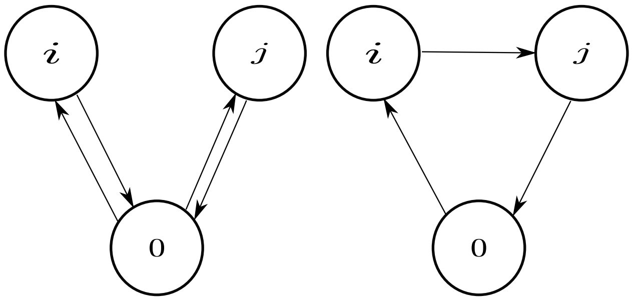


FIGURE 3.1: *An illustration of the Clarke and Wright savings method.*

3.7.2.2 Enhancements of the Clarke and Wright algorithm

The Clarke and Wright algorithm tends to create good results initially, but less-improving routes towards the end, often also including some circumferential routes [425]. This disadvantage was remedied by Gaskell [178] who proposed general savings of the form $s_{ij} = c_{i0} + c_{0j} - \lambda c_{ij}$, where λ represents a shape parameter. The parameter λ controls the emphasis placed on the distance between the vertices to be joined, with a larger value resulting in a higher emphasis placed. It was shown in [197] that values of λ between 0.4 and 1 produce good results.

The Clarke and Wright algorithm is rather time-consuming due to the fact that all savings have to be computed, stored and sorted. The savings heuristic requires two main issues to be addressed, namely the method of determining the maximum saving values and parameter storage requirements. Determining the maximum savings is the most time-demanding component. Three approaches may be considered in this respect [425]. The first approach involves a full sort being implemented in a straightforward manner, while the second approach involves an iterative limited sort that can be performed by means of a heap data structure [197]. The third approach involves an iterative computation of the maximum saving value [335]. Paessens [335] showed that $s_{ij} > \bar{s}$ in instances whenever $c_{0i} > \bar{s}/2$ and $c_{0j} > \bar{s}/2$, where $\bar{s}$ denotes the current maximum savings value. Paessens also reported numerical results illustrating that the iterative determination of maximum savings values tends, on average, to be the best approach.

In attempts at increasing the savings method's performance, some authors have only considered a small subset of all possible savings. Paessens [335], for example, disregarded all edges (i, j) satisfying $c_{ij} > \alpha \max_{k\in\{1,\dots,n\}} c_{0k}$ for some constant α. Nelson *et al.* [324] investigated the use of more complex data structures based on heaps in a bid to limit storage requirements which result in more efficient updating operations. They illustrated four different methods utilising adjacency information aimed at disregarding all edges associated with an interior vertex.

3.7.2.3 The matching-based savings algorithm

In both the papers [8] and [123], the authors independently proposed very similar algorithms that are interesting modifications of the savings algorithm. The algorithms involve calculation of the values $s_{pq} = t(\mathcal{S}_p) + t(\mathcal{S}_q) - t(\mathcal{S}_p \cup \mathcal{S}_q)$ at each iteration, where $\mathcal{S}_k$ is the set of vertices included in route k and $t(\mathcal{S}_k)$ is the length of an optimal TSP solution on the vertices in $\mathcal{S}_k$. Then a maximum-weight matching problem is solved over the vertices within the set $\mathcal{S}_k$ using the values s_{pq} as matching weights. Routes that correspond to an optimal matching are also merged if feasible.

Wark and Holt [446] proposed an alternative matching heuristic involving the merger of clusters of vertices at their endpoints. The matching weights are defined as ordinary savings, or alternatively in a manner favouring mergers of clusters whose route lengths and capacity requirements are far below the allowable values. Initialising with n back-and-forth routes[33], the algorithm successfully merges clusters resulting in only a few lines of columns of the savings matrix to be updated. This approach generates a tree of sets of clusters from which a best solution can be determined.

3.7.2.4 Sequential insertion heuristics

Mole and Jameson [317] proposed an algorithm involving the expansion of routes, one at a time. They employed two parameters λ and μ to expand the route under construction, by calculating

$$\beta(i,k,j) = \mu c_{0k} - c_{ik} + c_{kj} - \lambda c_{ij}. \tag{3.21}$$

The algorithm works as follows:

Step 1 Determine whether all vertices have been assigned to routes. If so, then stop. Otherwise, construct a back-and-forth route $(0, k, 0)$ with k representing any unassigned vertex.

Step 2 For every unassigned vertex k, determine the feasible insertion cost (3.21). If there are no feasible insertions, return to Step 1. Otherwise, insert the best vertex k^* into the emerging route.

Step 3 Optimise the current route using the 3-opt procedure[34], as described in [290].

An alternative sequential insertion heuristic was proposed by Christofides *et al.* [83]. The heuristic also employs user-controlled parameters λ and μ, and functions in two phases, as follows:

Phase 1 Sequential route construction

- Step 1: Initialise a route index with the value 1.
- Step 2: Calculate the insertion cost of all unassigned vertices which are able to initialise route k.
- Step 3: Insert the vertex with the lowest feasible insertion cost into the route under construction and perform a 3-opt optimisation.
- Step 4: If all vertices have been added, initialise a new path for construction.

Phase 2 Parallel route construction

- Step 1: Initialise k routes, where k represents the number of routes obtained at the end of Phase 1.
- Step 2: Associate all unassigned routes (*i.e.* determine which routes should be selected to have the respective construction techniques applied) while still considering feasibility and determining the minimum association cost matrix[35].

[33] A back-and-forth route exits from the depot, visiting a single customer, and then returning directly to the depot.

[34] A 3-opt analysis involves deleting three edges from a path, reconnecting the edges in all other possible ways, evaluating each reconnection, and then selecting the best one. This process is repeated for all different combinations of three edges. The most improving reconfiguration is implemented. This mechanism can be carried out in $O(n^3)$ time [425].

[35] The minimum association cost matrix contains the cost of assigning a vertex to a route.

- Step 3: Insert the unassigned vertices which minimise the overall route costs. Perform this operation until it is no longer possible to insert vertices into the route under construction. Perform 3-opt optimisation on the newly constructed route.
- Step 4: If it is still possible to construct new routes, go to Step 2. Otherwise, stop.

A comparison of the two methods described above was performed by Christofides [83]. The finding was that the sequential insertion heuristic of Christofides [83] was superior as it is able, on average, to generate better results in less computing time.

3.7.2.5 Two-phase methods

The family of two-phase heuristics occur in two paradigms, namely, a *cluster-first, route-second* paradigm and a *route-first, cluster-second* paradigm. Within the cluster-first, route-second paradigm there are several different methods. The simplest of these is referred to as the *elementary clustering method*. This method involves the formation of a single clustering of the vertex set and then proceeds to perform route construction on each cluster. Another method within this paradigm is the *truncated branch-and-bound* method. A third method in this paradigm is the *petal algorithm*, which generates a large family of overlapping clusters and forms the clusters from which a feasible route is constructed.

The elementary clustering method

The elementary clustering method was initially referred to as the *sweep algorithm* and was first introduced by Wren [457], but popularised by Gillet and Miller [190]. The sweep algorithm is limited to planar[36] instances of the VRP, where feasible clusters are initially formed by rotating a ray centered at the depot. Route construction is then performed by solving an instance of the TSP for each cluster.

There are a number of variations on this algorithm that apply different post-optimisation phases in which pairwise exchanges of vertices are performed over different clusters [425]. A simple implementation of this algorithm involves representing each vertex i by its polar coordinates (θ_i, ρ_i), where ρ_i represents the ray length and θ_i represents the angle. A vertex i is selected arbitrarily and assigned a value $\theta_i^* = 0$. The remaining angles are measured relative to $(0, i^*)$. The vertices are ranked in increasing order of their θ_i-values. The algorithm then works as follows:

Step 1 Select an unused vehicle k.

Step 2 Selecting the vertex with the smallest θ_i-value, assign the vertex to vehicle k as long as the maximum route length and the capacity constraint are not violated.

Step 3 Optimise each vehicle route by solving the corresponding TSP instance.

A variation on the elementary clustering method was proposed by Fisher and Jaikumar [165]. Their variation clusters by solving a *Generalised Assignment Problem* (GAP) as opposed to following the standard geometric approach [425]. The algorithm works as follows:

Step 1 Select a seed vertex $j_k \in \mathcal{V}$ to initialise each cluster k.

Step 2 Calculate the cost C_{ik} of assigning a customer i to cluster k as $C_{ik} = \min\{c_{0i} + c_{ij_k} + c_{j_k0}, c_{0j_k} + c_{j_ki} + C_{i0}\} - (c_{0j_k} + c_{j_k0})$.

[36] A graph is planar if it can be drawn in the plane without any pair of its edges crossing [455].

Step 3 Solve a GAP based on the associated C_{ij}-values, customer weights q_i and vehicle capacity Q.

Step 4 Solve a TSP for each of the clusters formed through the GAP solution.

The number of vehicles is a fixed *a priori* and the authors proposed a geometric partitioning approach according to customer weights in which the plane is partitioned into K cones. The seed vertices are dummy customers placed along the rays that bisect the cones. After cluster determination, the TSP instances are solved to optimality by adopting a constraint relaxation-based approach [425], as illustrated in [309].

Another variation on the elementary clustering method was introduced by Bramel and Simchi-Levi [55]. Their algorithm is a two-phase heuristic in which the seeds are determined during the first stage by solving a capacitated location problem[37], with the remaining vertices gradually included into the relevant routes during the second stage. The first stage involves locating K seeds among n customer locations in a manner that minimises the total distance from customers to their closest seed while still respecting the vehicle capacity constraint. The second stage involves constructing vehicle routes by inserting at each step the customer with the smallest insertion cost to the assigned route seed.

The truncated branch-and-bound method

The truncated BaB method was proposed by Christofides *et al.* [83] for problems with a fixed number of vehicles. The algorithm may be seen as a simplification of an earlier algorithm proposed by Christofides [81] and utilises a search tree that has as many levels as there are vehicle routes, with each level containing a set of feasible and non-dominated routes. The algorithm employs a variable $\mathcal{F}_h$ which denotes the set of unrouted vertices at level h and functions as follows:

Step 1 Let $h = 1$ and $\mathcal{F}_h = V \setminus \{0\}$.

Step 2 If $\mathcal{F}_h = \emptyset$, then stop. Otherwise, select an unrouted vertex $i \in \mathcal{F}_h$ and generate a set $\mathcal{R}_i$ of routes containing i and vertices in $\mathcal{F}_h$. The routes are gradually generated according to a linear combination of two criteria, namely savings and insertion cost.

Step 3 Evaluate the quality of each route in terms of both the standard TSP objective function and the length of the shortest spanning tree over the unrouted customers.

Step 4 Determine the route with the minimum resulting function value. The function in this case refers to a combination of the solution of the associated TSP and the length of the shortest spanning tree over the unrouted customers. Set $h \leftarrow h+1$ and let $\mathcal{F}_h = \mathcal{F}_{h-1} \setminus \mathcal{S}_{r^*}$, where $\mathcal{S}_{r^*}$ represents the vertex set of route r. Return to step 2.

In terms of solution quality, elementary clustering algorithms and the truncated BaB algorithm have been reported to outperform other constructive approaches [425].

The petal algorithm

The petal algorithm is an extension of the sweep algorithm [425]. It generates several routes, referred to as *petals*, and makes a final route selection by solving a set partitioning problem[38] in

[37] A capacitated location problem is the problem of assigning customers to clusters subject to a capacity constraint imposed on the number of customers per cluster.

[38] A set partitioning problem requires the partitioning of items in a given set into smaller subsets.

which the objective is to

$$\text{minimise} \sum_{k \in S} d_k x_k \tag{3.22}$$

subject to

$$\sum_{k \in S} a_{ik} x_k = 1, \qquad i = 1, \ldots, n, \tag{3.23}$$

$$x_k \in \{0, 1\}, \qquad k \in \mathcal{S}, \tag{3.24}$$

where $\mathcal{S}$ represents the sets of routes, $x_k = 1$ if route k belongs to the selected solution (or else $x_k = 0$), a_{ik} is a binary parameter that is equal to 1 if vertex i belongs to route k, and d_k is the cost associated with petal k. This algorithm was proposed by Balinski and Quandt [29], but is limited to small instances of $\mathcal{S}$. In both the papers [169] and [380], the authors introduced rules for generating a promising subset $\mathcal{S}' \subsetneq \mathcal{S}$ to be used in the heuristic approach. These heuristic rules were further improved upon in [366] where the subset $\mathcal{S}'$ was modified not only to include single vehicle routes, but also configurations consisting of two embedded or intersecting routes. This extension is referred to as the *2-petals algorithm* in the literature.

The route-first, cluster second paradigm

The first phase of two-phase methods in the route-first, cluster second paradigm of Beasley [39] involves creating a large TSP tour, and neglecting the VRP side constraints. The second phase then involves a decomposition of this tour into feasible vehicle routes. Beasley [39] describes how the second phase involves simply solving a standard shortest-path problem on an acyclic graph in $O(n^2)$ time. In the shortest-path algorithm, the cost C_{ij} of travelling between the vertices i and j is taken as $c_{0i} + c_{0j} + \ell_{ij}$, where ℓ_{ij} represents the cost of travelling from i to j along the TSP tour of the first phase.

3.7.3 Improvement heuristics

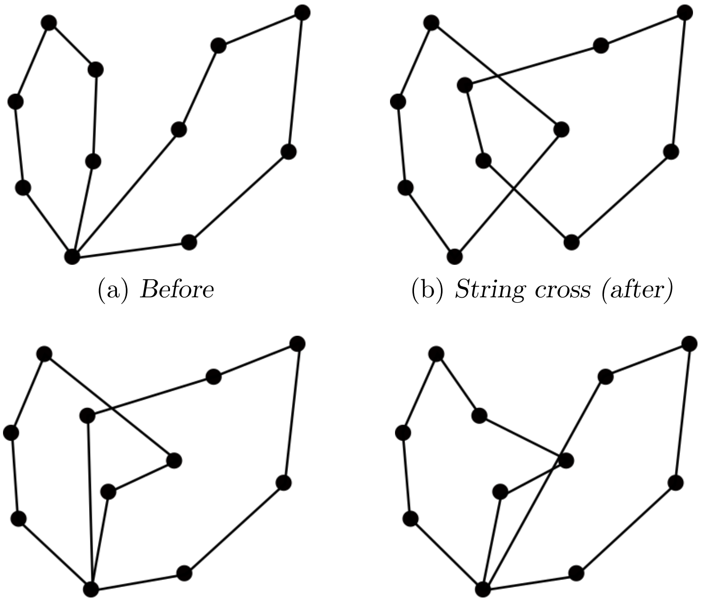
Improvement heuristics for the VRP may be applied to each vehicle route separately which allows any heuristic for the TSP to be applied. Alternatively, improvement heuristics may also be applied to several vehicle routes at once. The latter type of application allows for exploitation of multi-route structures.

3.7.3.1 Single-route improvements

Most of the single-route improvement procedures in the literature may be described in terms of Lin's [290] λ-opt mechanism [425]. Several improvements have been proposed on the standard λ-opt mechanism. Lin and Kernighan [291] suggested modifying λ dynamically throughout the search, whereas Renaud *et al.* [365] developed a restricted version of the 4-opt mechanism which creates a subset of promising reconnections between two so-called *chains*. An empirical study performed by Johnson and McGeoch [247], led to the conclusion that the dynamic λ-approach in [291] produced the best results, on average.

3.7.3.2 Multiroute improvements

Thompson and Psaraftis [422] introduced a general b-cyclic, k-transfer solution scheme. The scheme considers a circular permutation of b routes and k customers from every route, and



(a) *Before* (b) *String cross (after)*

(c) *String exchange (after)* (d) *String relocation (after)*

FIGURE 3.2: *Multiroute improvement operations.*

exchanges them to the next route of the cyclic permutation. Van Breedam [434] subsequently proposed an improved scheme involving string cross, string exchange, string mix, and string relocation methods (which may simply be viewed as special cases of 2-cyclic exchanges [425]):

- The string cross operation involves exchanging two strings (or chains) of vertices through crossing two edges of two different routes, as illustrated in Figures 3.2(a) and 3.2(b).
- The string exchange operation involves exchanging two strings of at most k vertices between two routes, as illustrated in Figures 3.2(a) and 3.2(c) for $k = 1$.
- The string relocation operation involves exchanging a single string of at most k vertices from one route to another, as illustrated in Figures 3.2(a) and 3.2(d) for $k = 1$. Here k is normally limited to a value of 1 or 2.
- The string mix operation is a selection of the best solution emanating from the string relocation operation and string exchange operation.

The scheme of Van Breedam [434] requires specification of a set of input parameters that influence the behaviour of the local improvement procedure. These parameters include the initial solution, the string length k, the evaluation procedure for a string of length $k > 1$, and an operation selection strategy. Van Breedam [434] noted that better-quality results are obtained in less computing time when the algorithm is provided with a good initial solution. Furthermore, the best solutions are obtained, on average, when using string exchange with $k = 2$, but this operation requires twice as much computing time as when using a value of $k = 1$.

3.8 Metaheuristics

Metaheuristics applicable to the VRP may be classified broadly into population-based methods and trajectory-based methods [428]. Population-based methods evolve a population of solutions which are combined iteratively in an attempt to create better populations over time and are based on mechanisms inspired by nature. The general procedure of implementation of these algorithms is to generate an initial population randomly, then evaluate the fitness of each individual of this population and perform various operations to generate a new population. Population-based algorithms, when applied to multi-objective problems, are concerned with three main issues [91]:

- How to select individuals in order to give preference to non-dominated solutions over dominated solutions,
- how to retain identified non-dominated solutions throughout the search process in order to eventually report non-dominated solutions with respect to previous populations, and
- how to maintain diversity within the population so as to avoid convergence to a local optimum.

Trajectory-based methods, on the other hand, explore the solution space by moving at each iteration from one solution to another, neighbouring solution as a result of applying local changes within a specified neighbourhood of the current solution [428]. Trajectory-based methods therefore require a neighbourhood relation definition in terms of the solution space. Typically, every candidate solution has numerous neighbours and the choice to move to a particular neighbour of the current solution is based on information limited to the neighbourhood.

3.8.1 Population-based algorithms

The class of population-based metaheuristics is inspired by natural concepts such as evolution, the movement of animals, and natural communication methods. Members of this class utilise high-level guidance strategies based on various memory structures, such pools of solutions represented by chromosomes, neural networks, or pheromone matrices [428]. The most successful VRP metaheuristics all implement a local search component to guide the algorithm to a promising set of solutions and thus most population-based metaheuristics in the VRP literature are inherently hybrid in nature [428].

3.8.1.1 Genetic algorithms

Genetic algorithms (GAs) are the most popular population-based metaheuristics in the VRP literature. The notion of a GA was first introduced by Holland [232] in 1976 — it is a probabilistic search algorithm based on natural selection. A GA is initialised by selecting a set of candidate solutions called a *population*. A single solution in this population is often referred to as a *chromosome*. The population size typically remains constant throughout the iterative process and the fitness of each chromosome is evaluated according to the VRP objective function. Chromosomes are probabilistically selected to form the next generation based on their fitness values and have reproduction operators performed on them.

When generating a new generation, the operations of mutation and crossover are randomly applied, and the average fitness values of the populations are calculated and compared. This process is repeated until a certain termination criterion is met, which is typically based on a convergence property of the average fitness value. GAs are popular due to the crossover and mutation mechanisms that allow the algorithm to escape from local optima.

The first successful application of a GA to a VRP was demonstrated in [351]. The application combined genetic operators (selection and crossover) with an effective local search mechanism that replaced the conventional mutation operator. The GA operators were applied to giant-tour solutions which allowed for simple permutation-based crossovers. The local search component was applied to the complete solution representation. Diversity management is, however, crucial in GAs so as to avoid local optima.

A simple GA can be described as follows. Initialise the algorithm by randomly generating a population of chromosomes $\mathcal{X}^1 = \{\boldsymbol{x}_1^1, \ldots, \boldsymbol{x}_n^1\}$. Then at each iteration $t \in \{1, \ldots, T\}$, apply steps 1 through to 3 below k times with $k \leq N/2$. Finally, apply Step 4:

- Step 1: (Reproduction phase) Select two parent chromosomes from $\mathcal{X}^t$.
- Step 2: (Recombination phase) Produce two offspring chromosomes from the selected parent chromosomes by means of a crossover operator.
- Step 3: (Mutation phase) Apply a random mutation to the offspring solution.
- Step 4: (New generation creation phase) Form $\mathcal{X}^{t+1}$ from $\mathcal{X}^t$ by removing $2k$ chromosomes from $\mathcal{X}^t$ and replacing them with the $2k$ offspring chromosomes.

This algorithm utilises a parameter T, which represents the number of generations, and a parameter k which represents the number of chromosomes selected per generation. The algorithm aims to improve the set of chromosomes over the course of various iterations until a suitable set of high-quality solutions is obtained.

In Step 1, the parents are typically selected probabilistically with a bias in favour of the best chromosomes. In Step 2, the offspring solutions are typically generated by applying a crossover operator to chromosome vector representations of two parents. In Step 3, the offspring solutions are slightly modified by altering vector entries with a small probability, called the *mutation probability*. Finally, Step 4 represents the generation replacement mechanism.

GAs perform particularly well in VRP instances where complicated constraints, such as time windows, are present and are able to produce highly competitive results due to the robust nature of the algorithmic structure. According to Yang [463], the two most notable advantages of employing GAs to solve VRP instances are their ability to deal with complex optimisation problems and their suitability to parallelism. They are also robust in nature, being able to deal with a wide variety of objective functions whether stationary or non-stationary, linear or nonlinear, continuous or discontinuous, and with or without random noise.

An interesting GA was proposed by Schmitt and John [390] for the time-constrained CVRP. This variation adopts a route-first, cluster-second paradigm which allows for genetic operators to be applied to megaroutes over all vertices. A solution is obtained by applying a sweep algorithm, starting with the vertex in the first position of the generated string.

Efficient GAs have recently been proposed for the CVRP. The algorithm proposed by Nagata and Bräysy [321] utilises an adaptation of the *edge-assembly crossover* operator which had previously achieved success in the context of the TSP [428]. This approach employs a crossover operator that considers the graph associated with the merger of partial chromosomes of the two parents. It then selects several cycles within the graph, rotating between the edges of the parents. The offspring are created by removing and adding the selected edges from and to the respective parents. A repair procedure is necessary as the resulting graph does not always represent a feasible VRP solution (it may contain subtours and/or tours not connected to the depot). The vehicle capacity constraint may also be violated and thus a greedy heuristic is applied to merge disconnected routes and a local search is performed to reduce the capacity infeasibility.

Vidal *et al.* [435] presented a hybrid GA that builds upon the one proposed earlier by Prins [351] through employing an advanced diversity control mechanism. It incorporates a bicriteria evaluation of the individuals in each population. The evaluation is based on solution quality and the contribution of solutions to population diversity. Vidal *et al.* proposed an interesting fitness function $\phi_{\mathcal{P}}$ for evaluating the desirability of a solution S in population $\mathcal{P}$. The fitness function is a weighted sum of the rank of S (in terms of solution quality) with respect to the solution cost ϕ^{cost} and in terms of maintaining its diversity ϕ^{div} within the population. The diversity term is modelled as the Hamming distance[39] between solutions. The fitness function is given by

$$\phi_{\mathcal{P}}(S) = \phi^{cost}(S) + \left(1 - \frac{\mu^{elite}}{|\mathcal{P}|}\right) \phi^{div}(S), \tag{3.25}$$

where μ^{elite} is a parameter governing the relative weight of the respective terms. This improved fitness function allows the algorithm to depend on an efficient granular local search without risking premature population convergence [428].

3.8.1.2 Ant colony optimisation

Ant colony optimisation (ACO) was first introduced in the early 1990s by Dorigo [137] and its working is based on the foraging behaviour of ants. At the core of the method of ACO lies the notion of indirect communication between ants through a pheromone trail, which allows them to find attractive paths between their current locations and a source of interest.

In general, the ant colony optimisation algorithm generates an approximate solution to an optimisation problem by iterating through two steps [47]. The first step involves constructing candidate solutions using a pheromone model (a parameterised probability distribution over the solution space), while the second step involves manipulating the pheromone values through the use of candidate solutions in an attempt to bias the search towards higher quality solutions.

The basic model of the algorithm proposed by Dorigo [137] in the context of a VRP associates two values with each edge (v_i, v_j) of a complete, weighted graph G on the set of all customers. The first value is the so-called visibility n_{ij} (the inverse of the length of the edge between vertex i and vertex j), and the second value is the pheromone trail Γ_{ij}. The visibility parameter is static, while the pheromone trail is updated dynamically. During each iteration, r artificial ants begin at each vertex of the graph and construct r new tours through the use of a probabilistic nearest neighbour heuristic equipped with a modified distance measure. The modified distance measure is derived from the parameters n_{ij} and Γ_{ij} in a manner that favours vertices close together and incident with edges associated with high pheromone levels. After each iteration, the pheromone levels associated with the edges are updated by allowing a small fraction $1 - \rho$, with $\rho \in [0, 1]$, of the pheromone values to be removed and placed on the edges of the new tours created during the iteration. The pheromone trail is increased by $\Delta_{ij}^k = 1/L_k$ for each ant k having traversed edge (v_i, v_j), where L_k is the length of the tour constructed by the ant. The trail value for the edge (v_i, v_j) is also updated by the substitution

$$\Gamma_{ij} \leftarrow \rho\Gamma_{ij} + \rho \sum_{k=1}^{r} \Delta_{ij}^k. \tag{3.26}$$

[39]In information theory, the Hamming distance between two strings of equal length is the number of positions in which the corresponding symbols are different. It measures the minimum number of substitutions required to change one string into the other, or the minimum number of errors that can transform one string into the other.

The value of the evaporation parameter ρ plays an important role during the initial stages of the search, preventing convergence to a poor solution, but must be almost insignificant during latter stages of the search so as to avoid conditioning the search too strongly towards the end.

According to Ding *et al.* [129], there are inherent weaknesses within the method of ant colony optimisation, namely:

- The search usually becomes trapped at a locally optimal solution,
- it often requires considerable computation time to obtain the final solution, and
- parameter adjustment is difficult in order to obtain good solutions.

The general conclusion drawn from the papers [138, 139] is that despite the excellent results produced on occasion by the method of ACO, it requires hybridisation with a local optimiser to remain competitive with other metaheuristics and local search algorithms. Kawamural *et al.* [255] proposed a complex hybrid of the ACO metaheuristic with a 2-opt improvement procedure and probabilistic acceptance rules that mirror those of simulated annealing. This hybridisation was applied to two CVRP instances containing 30 and 60 customers, respectively, and was able to find an optimal solution in both cases.

Bullnheimer *et al.* [66] proposed a hybrid ant system in which vehicle routes are first improved through 2-opt optimisation before any trail updates are performed. The algorithm also incorporates vehicle capacity and distance savings into the vertex selection process. The algorithm is further improved by the use of elitist ants, which are assumed always to return to the incumbent solution. In the paper, the authors performed computation experiments in respect of fourteen problem instances given in [83]. The results indicate that it is beneficial to apply the 2-opt optimisation procedure and to employ elitist ants.

Bullnheimer *et al.* [65] refined the algorithm proposed in [66], by:

- Adding a distance savings component directly to the visibility component and removing the computationally expensive capacity term from the algorithm,
- reducing the neighbourhood of candidate vertices to $\lfloor n/4 \rfloor$, and
- limiting the trail update to the best five solutions found during each iteration, and weighting the pheromone levels according to the solution's rank.

These refinements led to shorter computation time requirements and better quality solutions.

Reimann *et al.* [364] developed a more successful algorithm than the one proposed in [65]. The algorithm generates new solutions through a savings-based procedure and a local search method. The savings function differs from that in the standard Clarke and Wright algorithm and can be described as $\chi_{ij} = \tau_{ij}^{\alpha} - s_{ij}^{\beta}$, where τ_{ij}^{α} is the pheromone value that measures the effectiveness of combining vertices i and j in the previous iteration, with α and β being user-controlled parameters. The combination of the two vertices is based on the probability $p_{ij} = \chi_{ij}/(\sum_{(h,\ell)\in\Omega_k} \chi_{h\ell})$, where Ω_k represents the set of feasible (i,j)-combinations resulting in the k best savings.

More recently, Reed *et al.* [363] proposed first clustering the vertices and then applying the ACO algorithm to the clustered vertices in what they refer to as an ant colony system algorithm. They improved upon the basic method of ant colony optimisation in the following key areas:

- *Route construction*: The ants' movements to new vertices are determined by means of a pseudorandom proportional rule which incorporates a random, uniformly distributed variable. This adaptation allows, with a small probability, for an ant to make the best move based on the pheromone trails and heuristic knowledge; otherwise, it implements the standard selection rule.

- *Pheromone updating*: The algorithm utilises two types of pheromone updating, namely global and local updates. A local update is performed every time an ant traverses an edge while a global update is only performed when an ant produces the best tour uncovered so far. This allows for faster convergence as route construction is focused around the best tour.

The algorithm was able to solve VRP instances with over 500 customers to optimality, but in larger instances the clustering mechanism leads to poorer solutions.

3.8.1.3 Particle swarm optimisation

Particle swarm optimisation (PSO) was first introduced by Kenndy and Eberhart [256] and is an evolutionary algorithm which originated from modelling the unpredictable choreography of a flock of birds. In PSO, the population of candidate solutions is referred to as the *swarm* and each potential solution to the problem is referred to as a *particle*. The underlying algorithm of PSO is based on an update rule of the form

$$x_i(t) = x_i(t-1) + v_i(t), \tag{3.27}$$

where $x_i(t)$ is the position of a solution at time t and $v_i(t)$ is a velocity term that controls the movement of the particle within the search space through the solution space. PSO has become popular according to Reyes-Sierra and Coello [369] due to the main algorithm being relatively simple and its implementation being straightforward. Additionally, there is extensive source code available online for implementing the method. The second reason for its popularity is that PSO has been found to be very effective in a wide variety of applications and that it produces good results at a very reasonable computational effort. PSO is also popular due to its quick convergence on the current best solution, but the convergence is not necessarily around a globally best solution and the updating equations do not satisfy global convergence constraints [157].

PSO has been applied successfully by Ai and Kachitvichyanukul [4] to the VRP with simultaneous pickup and delivery, where it outperformed several competing algorithms in respect of well-known benchmark instances. The algorithm employs a velocity function based on three terms, namely inertia, cognitive learning and social learning terms. Typically, PSO is implemented in a hybridised manner [78, 193, 299] as a local-improvement method when applied as an approximate solution approach to VRPs.

3.8.2 Trajectory-based algorithms

Trajectory-based algorithms start from an initial solution x_t and move iteratively to the next solution x_{t+1} within a predefined neighbourhood $N(x_t)$ of x_t, for $t \in [0, 1, 2, \ldots, T]$, where T denotes the number of iterations allotted to the algorithms. Five popular trajectory-based algorithms are reviewed briefly in this section.

3.8.2.1 Simulated annealing

The main principle behind the method of simulated annealing is that a random solution is drawn at each iteration from within a neighbourhood of the current solution. The method can be described as follows for a minimisation problem with an objective function f: During iteration t of the search, a solution x is drawn from the neighbourhood $N(x_t)$ of the current solution x_t. If $f(x) \leq f(x_t)$, then $x_{t+1} = x$. Otherwise,

$$x_{t+1} = \begin{cases} x, & \text{with probability } p_t \\ x_t, & \text{with probability } 1 - p_t, \end{cases}$$

where p_t is normally a decreasing function of t and the difference $f(x) - f(x_t)$. Three common stopping conditions for the iterative search processes, such as the method of simulated annealing, include [425]:

- The objective function value of the incumbent solution has not improved by π_1% for at least k_1 consecutive cycles.
- The number of accepted moves has been less than π_2% for k_2 consecutive cycles.
- A total of k_3 iterations have been performed (in other words $T = k_3$).

An early implementation of simulated annealing within the context of the CVRP can be found in [375]. The authors defined the neighbourhood structure using several mechanisms. These mechanisms included reversing part of a route, exchanging vertices between separate routes, and moving a part of a route into a new section of the same route.

Osman [332] introduced a more successful implementation of simulated annealing to solve CVRP instances. It utilises better starting solutions, the parameters are adjusted during a trial phase, better neighbourhoods are explored, and the cooling schedule is adaptive. The overall algorithm adopts a two-phase solution approach. The first phase utilises the Clarke and Wright algorithm to generate an initial solution. The second phase then involves implementation of a simulated annealing search to refine the solution. The cooling procedure was also novel at that stage. The typical approach was to implement a decreasing cooling function, but Osman introduced a cooling function that decreases continually as long as the current solution is modified. The results produced by this approach were generally good, but occasionally it missed an optimal solution by quite a significant margin and it rarely uncovered an optimal solution.

A considerable amount of research has recently been conducted on variations of simulated annealing, either as straight heuristics or as components of metaheuristics. Vincent [437] introduced a multi-start simulated annealing algorithm. This algorithm differs from the conventional form as it acts within a multi-start hill climbing context, the neighbourhood construction is different, local search procedures are performed after each temperature reduction in the multi-start algorithm, and the initial solution is generated in a novel manner. The results obtained from this algorithm are promising, with the algorithm finding the most best new solutions for a number of benchmark instances when competing against four other algorithms.

3.8.2.2 Deterministic annealing

The method of deterministic annealing is very similar to simulated annealing, except that a deterministic rule is used for acceptance of a move during each iteration. Standard implementations of this technique involve threshold acceptance (introduced by Dueck and Scheuer [151]) and record-to-record travel (introduced by Dueck [150]). According to the threshold-accepting algorithm, a solution $x \in N(x_t)$ is accepted during iteration t if $f(x) < f(x_t) + \theta_1$, with θ_1 being a user-controlled variable. In record-to-record travel, on the other hand, the best solution encountered during the search is x^* (record). During iteration t, a solution x is accepted if $f(x) < \theta_2 x^*$, with θ_2 again being a user-controlled parameter (usually with a value slightly larger than 1).

Golden *et al.* [196] applied the record-to-record algorithm to twenty large instances of the CVRP and compared the results thus obtained to those obtained by the tabu search of Xu and Kelly [461]. They demonstrated that the record-to-record algorithm was able to produce results in much less computation time and in general produced a higher quality solution. More recent implementations of determinisitc annealing within the context of the CVRP may be found in [207, 287], which employ slight variations of the decision inequality to produce effective results, while still being easy to implement.

3.8.2.3 Tabu search

The tabu search algorithm moves from a solution x_t to the best non-tabu solution x_{t+1} within a neighbourhood $N(x_t)$ of x_t during iteration t. The algorithm avoids cycling by declaring revisitation of solutions that share attributes with x_t as forbidden or *tabu*. The forbidden visitation status of a solution may be revoked whenever the current solution corresponds to a new best known solution.

One of the earlier attempts at applying tabu search to the CVRP is due to Willard [452]. He proposed a method that transforms the solution into a giant tour through replication of the depot. Neighbourhoods are defined as all feasible solutions that are attainable from the current solution by means of 2-opt or 3-opt exchanges. The next solution is selected as the best non-tabu element of the neighbourhood of the current solution during each iteration.

A slightly improved version of the aforementioned algorithm was introduced by Pureza and Franca [354], where the neighbourhood of the solution is defined by moving a vertex to a different vehicle route or swapping vertices between routes. The results returned by this algorithm were slightly better than the results produced in [452], but still relatively poor. The improved tabu search managed to illustrate that more complex search mechanisms are required to obtain good results for VRP instances.

Another novel neighbourhood definition was introduced by Osman [332]. He defined the neighbourhood of the current solution by means of the so-called λ-interchange generation mechanism. This mechanism includes a combination of 2-opt moves, vertex reassignments and interchanges. There are two implementable search strategies in the tabu search of Osman [332]. The first strategy searches the entire neighbourhood and then selects the best non-tabu solution. The second search strategy searches the neighbourhood until it discovers a non-tabu solution that improves the incumbent solution. The latter approach produced excellent results [425] but still has room for improvement.

A rather involved variant of tabu search was introduced by Laporte et al [184]. The algorithm is referred to as a tabu-route algorithm and introduced numerous novel aspects to the tabu search algorithm. The neighbourhood of a solution is defined as all solutions reachable by removing a vertex from a vehicle route of the solution and adding it into another route containing its p nearest neighbours, using a generalised insertion procedure [185]. This either eliminates a route or creates a route. The algorithm also considered routes that are infeasible — facilitated by incorporating a penalty component into the objective function, thus reducing the chances of the solution becoming trapped at a local optimum. The algorithm produces high-quality results and often yields an optimal solution [425].

Xu and Kelly [461] introduced an interesting tabu search variation. The neighbourhood definition is more sophisticated, involving swapping vertices between two routes, global repositioning and local route optimisation. The algorithm is governed by several parameters which are dynamically adjusted throughout the search. A pool of solutions is stored and periodically used to reinitiate the search with new parameters. The algorithm produces high-quality results, obtaining the best known solution to a number of VRP benchmark instances, but it requires large computation times and parameter tuning tends to be cumbersome [425].

Rochat and Tailard [376] adopted an adaptive memory procedure. Implementation of an adaptive memory involves keeping track of a pool of good solutions that is dynamically updated. More specifically, some of the elements of the pool are periodically extracted and combined in order to produce new good solutions. The extraction procedure gives a larger weighting to routes belonging to the best solutions and results in the algorithm producing good results when applied to benchmark data [425]. Similar approaches were presented in [417, 418] for solving the CVRP. The key difference between these searches and the one presented by Rochat and Tailard [376] is that new partial solutions are constructed by combining promising vertex sequences present in the adaptive memory. The solutions are then refined by applying a tabu search.

Toth and Vigo [427] introduced a granular tabu search solution approach. The main concept behind this approach is that edges with large weights are less likely to form part of an optimal solution. Therefore, through eliminating edges which exceed a *granularity threshold*, several unpromising solutions are not considered during the search process. Neighbouring solutions are generated by means of a limited number of exchanges within the same route or between two routes. The algorithm is able to produce excellent results in limited computation time [425].

More recently, Zachariadis and Kiranoudis [466] introduced a tabu search algorithm that also produces good results. They implemented a *penalised static move descriptor algorithm* (PSMDA) which penalises the cost labels for static move descriptors used in the 2-opt approach. This diversifies the search process while still applying a neighbourhood reduction policy so as to reduce the computational time requirements. They employed the algorithm proposed by Paessens [335] to generate an initial solution and then refined this solution by applying the PSMDA. The algorithm was applied to the benchmark data proposed by Golden *et al.* [199], was able to solve large instances to within 0.19% of the best known solution, and exhibited good stability [466].

3.8.2.4 Iterated Local Search

The origins of the method of iterated local search can be traced back to the paper by Baxter [36]. It is a simple heuristic and is easily implemented in conjunction with other local search heuristics, such as steepest descent or tabu search. The underlying concept behind this solution approach is to apply an embedded local search algorithm until a stopping condition is met, and then perturbing or distorting the results in order to obtain a new starting point for the embedded algorithm. This procedure is repeated until a certain stopping criterion is met, such as that the maximum number of iterations have been performed or that an acceptable quality solution has been obtained.

The perturbation operation is the most crucial aspect of the algorithm. This operator is application-specific and has to be designed with care so as to allow for significant alterations to the solution which are not easily undone by the embedded algorithm, while also not completely destroying the structure of the original solution [428].

Chen *et al.* [79] applied an iterative variable neighbourhood search descent algorithm in the context of VRPs. The perturbation operator employed exchanges a number of consecutive customers within a vehicle route. The results thus obtained were compared with those obtained when implementing a random restarting point and illustrate conclusively that utilising a perturbation operator yields superior results. Uchoa and Ochi [409] proposed a hybrid metaheurisitc which combines an iterative local search heuristic with a set partitioning algorithm to create vehicle routes. The perturbation operator performs combined exchanges of customers.

3.8.2.5 Variable neighbourhood search

Variable neighbourhood search is a metaheuristic for building heuristics, aimed at solving combinatorial and global optimisation problems. The basic concept is to apply systematic change in a neighbourhood combined with a local search.

This method was proposed by Mladenović and Hansen [316] as a general search procedure. It works in conjunction with several neighbourhoods, which are often of increasing complexity [428] and is usually embedded with a 2-opt or 3-opt mechanism. The algorithm starts with an initial solution and iteratively applies these mechanisms on the neighbourhoods in a descending fashion until no further improvements are possible. After the last neighbourhood has been applied, a new cycle may be initialised.

Variable neighbourhood search was successfully applied to a large-scale VRP by Kytöjoki *et al.* [266]. The algorithm comprises seven improvement heuristics that are both inter-route and intra-route based. The initial solution is constructed sequentially until all customers have been routed. A route is first initialised with a seed customer and then customers are added one by one to this route. The initial route construction is terminated before the route is a 100% full so as to allow the improvement phase more room to work with. The second phase incorporates all seven improvement heuristics to refine the solution. The algorithm was able to solve the large CVRP instances proposed in [199] to within 1.71% of the best-known solution and when applied to the benchmark data in [287], it was able to solve these instances to within 0.41% of the best-known solution.

3.9 Hybridisations

Hybridisations are algorithms that rely on combined concepts borrowed from various algorithmic paradigms, such as large neighbourhoods, local searches, collective intelligence sharing, solution perturbation mechanisms, integer programming techniques, population-based searches, constraint programming, data mining, tree searches and parallel computing [428]. This field offers considerable promise and some of the more discernible families of algorithms within this broad class are discussed very briefly in this section.

3.9.1 Population-based search and local search

Population-based heuristics have been complemented by local search procedures for a long time. The converse approach, where local search procedures are complemented by population and recombination concepts, is more recent. This approach is referred to as *Adaptive Memory Programming* in the literature and a full description of this solution approach is given in [413]. Throughout the search process, recurrent fragments from elite solutions are stored in an *adaptive memory* component and subsequent routes utilise this information as a starting point in order to create new solutions. This approach has led to successful heuristics for the CVRP, as may be seen in [376, 417]. Another key concept is *guidance*. This concept extracts and utilises promising sequences of solutions, as demonstrated in [278]. An alternative manner in which promising sequences of solutions can be exploited is through data mining, as illustrated in [385].

3.9.2 Meta-meta hybridisations

A common hybrid approach is to combine several concepts borrowed from different metaheuristics. These can either be applied concurrently or sequentially. Kytöjoki [266] combined variable neighbourhood search with an approach known as *guided local search*. The approach temporarily penalises certain solutions so as to be able to escape from local optima. A common approach in this research area is to perform multiple restarts from different initial solutions, as demonstrated in [99, 352, 368].

In addition to restarts, Prins [352] performed several random local searches from the incumbent solution, resulting in a population of solutions at each iteration. Cordeau and Maischberger [99] combined several concepts, such as pertubation, tabu memory and a guided local search objective — all within in a parallel context to solve instances of the CVRP.

3.9.3 Hybridisations with large neighbourhoods

In a variable network search, the variety of neighbourhoods is crucial [428]. One possible method of improving the variety of neighbourhoods is to rely on structurally different neighbourhoods by employing SWAP or RELOCATE operators, neighbourhoods based on ruin-and-recreate moves or ejection chains [428]. Pisinger and Ropke [349] offered a good example through their use of the *Adaptive Large Neighbourhood Search* (ALNS) algorithm. The algorithm utilises a roulette wheel mechanism to select structurally different large neighbourhoods. Mester and Bräysy [306] introduced a complex hybridisation that uses guided local search, large neighbourhoods based on ejection chains and an evolutionary strategy based on a one-to-one exchange principle.

3.9.4 Hybridisations with mathematical programming solvers

A *matheuristic* is the combination of metaheuristics with mathematical programming solvers or other exact algorithms. A successful strategy is to store high-quality routes in a pool and then apply an integer programming solver to the set covering formulation of the CVRP. Such matheuristics have been applied successfully to CVRP instances with large numbers of short routes, as demonstrated in [208, 318, 409]. Non-matheuristics[40] have also been applied to instances of the VRP; such an approach is illustrated in [400]. The method utilises constraint programming algorithms for solution reconstruction.

3.9.5 Parallel algorithms

Parallel and cooperative search mechanisms have led to hybrid heuristics that are capable of producing good solutions to the CVRP. Cranic and Toulose [104] adopted a high-level parallelism which is able to consider different solutions and search trajectories in different threads. At any stage during the search procedure, the algorithm maintains a population of solutions which

[40]The term *non-matheuristics* refers to the incorporation of constraint programming in the solution approach, as opposed to matheuristics that typically incorporate integer programming techniques.

can be analysed, exchanged and stored. Parallel searches are therefore able to complement the exploration capacities of population-based searches and the fast improvement mechanisms of local searches [428]. The most successful parallel algorithms benefit from heterogeneous collaborating solvers, such as genetic algorithm and tabu search cooperation [278], shaking and set covering solvers[41] [208], or multiple tabu searches with different neighbourhood structures [246].

3.9.6 Decompositions or coarsening phases

Many of the more recent metaheuristics are complemented by a coarsening or decomposition phase, allowing them to handle large instances of the CVRP efficiently. A common approach is to rely on the customer-to-route assignments of an existing solution and then combine subsets of different routes and customer assignments to create subproblems. Several methods may be used to define the subsets [428]. These include generating customer subsets

- randomly,
- relative to a proximity criterion between routes,
- through increasing the polar angle associated with the depot, and
- partitioning the geometric space into sections.

An alternative instance size reduction technique consists of identifying recurring edges in good solutions, which may then be merged temporarily, allowing focus to be shifted to the remaining decision variables [442].

3.9.7 Diversification vs. intensification

The balance between diversification and intensification is crucial in the design of any metaheuristic. Most metaheuristics aim to strike a fine balance between the two search strategies when attempting to solve a CVRP instance. The different strategies are summarised in Table 3.1.

3.9.8 Unified algorithms

Recently, there has been an increasing number of variants of the CVRP with additional constraints, objectives and decision variables [428]. These additional features can be accommodated by the numerous hybrid heuristics that are available in the literature, but it is becoming more important for heuristics to be flexible enough to be applied to numerous variants instead of being problem-specific heuristics.

There are algorithms that are flexible enough to deal with several multi-attribute VRPs, such as the *unified tabu search*. This algorithm was proposed by Cordeau *et al.* [100] and is able to solve classical VRPs as well as PDPs with any combination of multiple depots, multiple planning periods, duration constraints, and time windows.

The algorithm was later improved by Cordeau and Maischberger [99], who incorporated it into a parallel iterated tabu search heuristic. Similarly, the ALNS algorithm of Pisinger and Ropke [349] is able to solve VRPs and PDPs with multiple depots, backhauls, vehicle-customer compatibility constraints, and time windows efficiently.

[41] The parallel algorithm developed by Groer *et al.* [208] incorporates a metaheuristic algorithm referred to as a *shaker* to provide initial solutions, which are then passed to an MIP solver, referred to as *set covering solver*, to solve a set-covering problem with columns (vehicle routes) of the routes provided by the metaheuristics.

TABLE 3.1: *Key metaheuristic strategies [428].*

Diversification		Intensification	
Technique	**Algorithm**	**Technique**	**Algorithm**
Noise in edge costs	Adaptive Large Neighbourhood Search		
Penalisation of known solution features	Guided Local Search	Incentives on elite solution features	Adaptive Memory Programming
Incentive on under-explored solution features		Pheromones	Ant Colony Optimisation
Penalised infeasible solutions	Tabu Search		
Crossover and recombinations	Genetic Algorithms, Evolutionary Strategy, Scattered Search	Enumerative Neighbourhoods: RELOCATE, SWAP, 2-OPT	Variable Neighbourhood Search
Large neighbourhoods: ruin-and-recreate, ejection chains, set covering	Large scale neighbourhood search	Large neighbourhoods: ruin-and-recreate, ejection chains, set covering	Large scale neighbourhood search
Randomised choice of neighbours: Random first improvement, top % savings	Variable Neighbourhood Search, Adaptive Large Neighbourhood Search	Greedy choice of neighbours: Best improvement, best savings	Variable Neighbourhood Search, Adaptive Large Neighbourhood Search
Deteriorating moves	Simulated Annealing	Deterioration thresholds	Record-to-Record travel
Tabu memories	Tabu Search	Elite guiding solutions	Path Relinking
Population of solutions	Genetic Algorithms, Evolutionary Strategy	Exchange of elite solutions	Parallel
Parents and survivors selection with respect to diversity	Hybrid Genetic Algorithm	Parents and survivors selection with respect to solution cost	Genetic Algorithms, Evolutionary Strategy
		Adoption of method components relatively to performance measures	Adaptive Large Neighbourhood Search
Fixed under-explored solution elements, uncoarsening	Parallel, Multi-level algorithms	Fixing variables from elite solutions, coarsening	Parallel, Multi-level algorithms
		Decomposition phases based on routes from an elite solution	Various Methods

The iterated local search algorithm of Ochi *et al.* [409] is able to solve numerous problems with multiple depots, heterogeneous fleets, and simultaneous or mixed pickups and deliveries efficiently. Vidal *et al.* [436] proposed a *unified hybrid genetic search* algorithm with the intention of being highly flexible. This was achieved by limiting problem-specific features to small modular components. This allows the algorithm to be competitive in the context of thirty different problems [428].

The design of more unified solvers poses a serious challenge, depending on the nature and number of attributes considered. As long as only a few characteristics are considered, a unified method for merging all attributes is feasible. There are, however, certain attributes, such as soft time windows or loading constraints, which lead to time consuming solution evaluation and local search procedures. The use of dummy variables or values when dealing with a unified algorithm can have adverse effects on the computation time. Also, special care has to be taken in the calibration of parameters which may be highly correlated to problem-specific components and thus when creating more flexible algorithms, it is necessary to study the number of parameters and their sensitivity with respect to several problem features.

3.10 Aspects to consider when selecting a metaheuristic

When selecting a metaheuristic for implementation, there are several factors to consider. The first such factor is the computational time requirements of the heuristic. Most real-life instances of the CVRP require solutions within a limited time period and so heuristics that are able to generate solutions in a timely manner are favoured. Another factor to consider is the solution quality achieved by the metaheuristic. Most real-life instances of the CVRP are not exactly solvable and so it is desirable that a metaheuristic should be able to obtain solutions to such instances that are close to optimal. The simplicity of a metaheuristic is also a key determining factor. The algorithm should be simple in the sense that it should be easy to understand, implement and fine-tune its parameter values. The metaheuristic should also not have too many parameters and the sensitivity of solution quality with respect to these parameter values should be considered. It is also desirable that the metaheuristic should be able to solve alternative variations of the VRP with minimal alterations (*i.e.* the metaheuristic should be flexible). Another determining factor is the robustness of the metaheuristic — it should be able to generate good solutions consistently.

3.11 Chapter summary

The focus in this chapter was on the different methodologies that can be employed to obtain solutions to VRP instances and, more specifically, to the archetypal CVRP. In §3.1, three different CVRP model formulations were reviewed and this was followed in §3.2 by descriptions of classical exact solution approaches for the CVRP. Three main solution approaches were covered in this section and the progression and improvement of the methods in this class were documented. This naturally led to §3.3, in which the working of newer and more powerful exact algorithms was described. The discussion focussed on the progression over time of the various modern solution approaches and the key novel features of each approach resulting in the continuous improvement in this research area.

The main families of valid inequalities that have been utilised in more recent work to reinforce the formulations of §3.1 were discussed in §3.4, focusing on the family of cutting planes and various specialised inequality constructs used to reinforce CVRP formulations.

In §3.5, the main ideas proposed in the literature for pricing routes were discussed. The two main techniques utilised to reduce the duality gap of column and cut generation over the *set partitioning formulation* were further discussed in §3.6. The latter section also included an example of a hybrid approach that is a combination of the methods of branching and route enumeration.

In §3.7, various classical heuristic VRP solution approaches were reviewed and grouped into two overarching classes, namely constructive heuristics and improvement heuristics. More powerful metaheuristic VRP solution algorithms were discussed in §3.8. These algorithms were again grouped into two broad categories, namely population-based algorithms and trajectory-based algorithms. Real-life applications and results obtained by the various approximate solution approaches for these applications were compared.

Recently, most of the VRP algorithms have tended to be combinations of concepts from multiple solution approaches. These hybridisations were discussed briefly in §3.9. Finally, the key aspects that should be considered when selecting a metaheuristic for implementation were summarised in §3.10.

CHAPTER 4

Clustering Algorithms

Contents

Clustering involves an unsupervised classification of patterns into groups [244], and various different clustering approaches exist in the literature. This chapter opens with a brief introduction in §4.1 to the notion of clustering and its applications, and this is followed by a brief description in §4.2 of the more commonly adopted approaches when attempting to solve clustering problems. Typically adopted approximate data clustering approaches are presented in §4.3, with the focus shifting to clustering approaches applied to VRPs in §4.4. Key aspects present within all clustering algorithms, such as the determination of a suitable number of clusters, is elaborated upon in §4.5, and this is followed by a description of procedures available in the literature to validate the clusters produced by a clustering algorithm in §4.6. The discussion then turns to the criteria typically employed when evaluating the sensitivity of a cluster in §4.7, referred to as an *admissibility analysis*, and the chapter finally closes in §4.8 with a brief summary of its content.

4.1 Introduction

Organisation of data into sensible groupings is one of the most fundamental modes of understanding and learning [244]. Classification has played an integral role in the history of human development, facilitating understanding of novel phenomena. Humans naturally aim to identify descriptive features and further compare these features with known objects and phenomena during the learning process [307]. The classification of similar objects into groups is therefore an important human activity [254].

Classification of objects into smaller groups has furthermore always played an integral role in science. During the eighteenth century, Linnaeus[1] [292] and Sauvages[2] [114] provided extensive classifications of plants, animals, diseases and minerals. Hertzsprung and Russel [229] were able to classify stars according to their light intensity and surface temperature. There are numerous other examples of scientific contributions hinging on the ability to classify objects into groups [16, 61, 421].

Cluster analysis is the formal study of algorithms and methods for grouping observations, while *clustering* is an unsupervised machine learning technique used to group unlabelled objects together which are similar to one another in a multidimensional feature space, typically with the purpose of discovering some inherent structure within the data [59]. Data clustering is truly ubiquitous and has been employed in numerous applications [3, 126, 468].

The development of the research field of cluster analysis has left its mark in numerous other fields including taxonomy, psychology, statistics, mathematics, engineering, and medical research, with its first official appearance in 1939 in a study of sociological data [430], although references to clustering date back to Aristotle and Theophrastos in the fourth century B.C. Data clustering has evolved dramatically over the more than six decades since its formal inception, with the subject now appearing at the forefront of several interesting research fields such as high-throughput genomic data [46, 118, 173], mining big data [44, 458] and machine learning applications [371, 460].

Real-life instances of the CVRP require the need for transporting commodities from providers to various geographically dispersed customers or delivery stations. The cost of delivery can sometimes be greatly affected by grouping these customers or delivery stations together into clusters based on their needs or demands while still considering problem-specific constraints.

Clustering activities usually strive for inner homogeneity of data residing in the same cluster [251]. In the context of the pathological specimen transportation application described in Chapter 1, the geographical locations of the customers in a cluster must be similar, while other characteristics such as processing capabilities of laboratories and potential legislation issues of multi-provincial service areas might also affect clustering. The magnitude of a problem instance, due to the number of facilities, may perhaps call for an effective clustering algorithm to segregate the problem instance into more manageable sub-problem instances.

Three criteria differentiate between the manners in which objects are clustered: the degree of granularity desired, the distance measure employed and the clustering objective [89].

4.2 Data clustering approaches

Clustering is, in general, a difficult combinatorial optimisation problem which has been shown to be NP-complete [180]. There are numerous different approaches towards solving clustering problems in the literature. A taxonomy of these approaches was suggested by Jain *et al.* [244], and an elaboration on this taxonomy is illustrated graphically in Figure 4.1.

The two broad categories of clustering methods mentioned in the previous section are shown in Figure 4.1. Algorithms in the two categories follow different paradigms when clustering. Hierar-

[1] Carl Linnaeus established three kingdoms, namely *Regnum Animale*, *Regnum Vegetabile* and *Regnum Lapideum* in his seminal work *Systema Naturae* in which classifications are based on five levels (kingdom, class, order, genus and species).

[2] Francois de Sauvages established a methodical nosology for diseases in which the classification system involves ten major classes of disease.

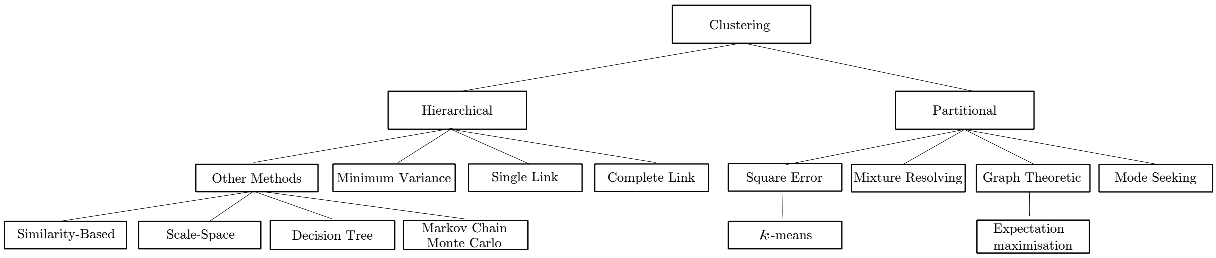


FIGURE 4.1: *A taxonomy of clustering approaches (adapted from [244]).*

chical clustering methods begin with singleton clusters and iteratively combine clusters until no more profitable combinations exist. In contrast, partitional clustering methods iteratively partition clusters until a certain termination criterion is met — typically, a pre-defined parameter stipulating the number of clusters sought.

4.2.1 Hierarchical clustering algorithms

Typically, hierarchical clustering algorithms are variations on the well-known single-link [406], complete-link [257], and minimum-variance (Ward's method) [445] algorithms — the most popular being the single-link and complete-link algorithms [244]. The main difference between these two algorithmic approaches toward data clustering is the manner in which they characterise the similarity between pairs of clusters. According to the single-link approach, the distance between two clusters is taken as the minimum of the distances between all pairs of data points drawn from the two clusters (one point in each cluster), while according to the complete-link approach, the distance between two clusters is taken as the maximum of all pairwise distances between two data points in different clusters. Both approaches adopt a bottom-up clustering paradigm, where clusters are iteratively merged to form larger and larger clusters based on the respective distance fitness functions.

It has been shown by Baeza-Yate [20] that the complete-link algorithm typically produces tightly bound or compact clusters, while Nagy [322] showed that the single-link algorithm suffers from the so-called *chaining effect* — the tendency to produce clusters that are elongated in feature space. It has also been observed, from a pragmatic point of view, that the complete-link algorithm tends to produce more useful hierarchies in numerous applications [243].

Hierarchical algorithms are typically considerably more versatile than partitional algorithms, as they are more adept at handling data sets that do not exhibit isotropic clusters[3] [322]. Day [110], however, noted that the time and space complexities of partitional algorithms are typically lower than those of hierarchical algorithms.

Hierarchical agglomerative clustering algorithms begin with singleton clusters and recursively merge clusters through the use of a proximity matrix to form larger and larger clusters until the desired number of clusters is reached. A tree representing such a structure is referred to as a *dendrogram* in the literature and allows for the exploration of the clustering hierarchy at different levels of granularity [43]. If one cluster dominates, highly skewed trees may form which, depending on the problem instance, could be beneficial (such as, for example, when balanced workload is not an objective within an assignment problem instance — in this case the data

[3]Isotropy is uniformity in all orientations. Accordingly, isotropic clusters are clusters that exhibit a similar nature for all features considered within the data provided.

points would represent tasks, while clusters would represent sets of tasks to be assigned to the various agents).

Leung *et al.* [285] proposed a hierarchical clustering algorithm inspired by human visual research based on scale-space theory. In this context, clustering is interpreted as a blurring process in which each data point is regarded as a light point and a cluster is represented as a blob. The algorithm merges blobs, initially consisting of a single light source, until the entire image becomes one light blob.

Li and Biswas [286] proposed the similarity-based agglomerative clustering algorithm which is able to handle both continuous and nominal data. The algorithm employs the Goodall similarity measure[4] [201] which focuses attention on less common matches of feature values in order to calculate the proximity of mixed data type observations.

Castro *et al.* [71] introduced a Markov chain Monte Carlo-based clustering method and an agglomerative likelihood tree algorithm, both of which are based on the maximum likelihood principle[5]. The algorithms are able to handle errors within the similarity matrix as they utilise a generative tree-structured model that represents relationships between the objects as opposed to directly modelling properties of the objects.

Basak and Krishnapuram [34] developed an unsupervised decision tree algorithm, with the most prominent property of the decision tree being its facilitation of the interpretation of the clustering results by virtue of a set of rules suggested by Quinlan [355]. The algorithm initialises at the root node of the decision tree and partitions the data based on a specified feature according to four different criteria emanating from information theory [205]. This process is repeated until the number of data points contained within a clustering node is less than some prespecified parameter. This algorithm was later incorporated into the commercialised software package of Basak and Krishnapuram [35] which produces a personalised list of items for a specified user.

4.2.2 Partitional clustering algorithms

Partitional algorithms aim to optimise a given clustering by iteratively relocating data points between clusters until a (locally) optimal partition is attained. These algorithms obtain a single partition of the data as opposed to an entire clustering structure. They are advantageous in applications that involve large data sets for which clustering according to a dendrogram is not feasible, although partitioning the data into a specified number of non-empty sets is in itself a computationally expensive task [462]. Even small-scale clustering problems, such as clustering 30 data points into three clusters, can lead to approximately 2×10^{14} possible partitions [294].

Partitional clustering algorithms often provide clearer insight into the main structure of the data since the larger clusters are generated during the early stages of implementation of the algorithm and are typically less likely to suffer from accumulated erroneous decisions, which cannot be corrected during the subsequent hierarchical process [254].

The major concern associated with the use of partitional algorithms is selecting a suitable number of output clusters. A recommendation for the number of clusters is given by Dubes

[4]The Goodall similarity measure incorporates typical measures of similarity of attributes between different data points. It also incorporates a probability component based on the likelihood of a random sample of two data points having similar values to the data features under consideration. The cumulative probability of the observed pair is then calculated and the different probabilities yielded by the different attributes are combined. A similarity matrix is finally calculated based on the complements of these combined probabilities.

[5]Given an experimental observation, one should utilise as point estimates of parameters of a distribution those values that yield the largest conditional probability for that observation, irrespective of the prior probability assigned to the parameters.

[148], but ultimately it should be determined empirically for better, context-specific results. Combinatorial searches for an optimal number of clusters is often infeasible and has led to the heuristic practice of executing the algorithm multiple times with different starting states, and then selecting the best configuration [244].

The most frequently used clustering criterion in partitional clustering algorithms is the squared error criterion, which is typically very effective in accommodating isolated and compact clusters. The squared error value associated with a clustering, containing n_c clusters, is

$$e^2 = \sum_{j=1}^{n_c} \sum_{i=1}^{n_j} ||\boldsymbol{x}_i^j - \boldsymbol{c}_j||^2, \tag{4.1}$$

where $\boldsymbol{x}_i^j$ denotes data point i belonging to the cluster j, $\boldsymbol{c}_j$ denotes the centroid of cluster j, and n_j denotes the number of data points within cluster j.

The most commonly used partitional clustering method is the *k-means method* [298]. This method assigns each data point to the cluster whose centre is nearest to it, with the centre being the centroid of all the data points already within the cluster. The algorithm keeps reassigning data points to clusters until a convergence criterion is met. Numerous improvements have been proposed for the standard k-means algorithm, such as the addition of a priority measure affecting cluster selection [180], the inclusion of a Minkowski distance metric in the clustering criterion [111], and the introduction of various improved initialisation procedures [7, 21, 72]. The k-means algorithm is popular due to its easy implementation and linear time complexity in terms of the number of data points that have to be clustered. The major drawback of the k-means algorithm is that the quality of its solutions depends on a reasonable initial partition. No efficient and universal method exists for determining initial partitions that leads to high-quality clustering results [462].

The best known graph theoretic partitional clustering algorithm is based on constructing a shortest spanning tree on the data, where the edge weights represent the inter-data distances. Edges with the largest weights are deleted from the tree so as to generate clusters of data points that are close to one another in feature space. The class of hierarchical algorithms is also related to graph theoretic clustering, as the graphs associated with single-link clusters are subgraphs of a minimum spanning tree of the data [203] and those associated with complete-link clusters form maximal complete subgraphs (which is related to vertex colouring of graphs) [25]. Hartuv and Shamir [224] treated clusters as highly-connected subgraphs, where "highly-connected" refers to graph connectivity (defined as the minimum number of edges whose deletion will disconnect a graph). Hartuv and Shamir recursively applied a minimum weight cut procedure aimed at disconnecting the graph by deleting the minimum number of edges so as to identify these highly-connected subgraphs.

There are several variations in the class of mixture-resolving algorithms, although the underlying assumption of these variations is the same. This assumption is that the data points to be clustered are drawn from one of several distributions, and the main goal is to identify the parameters of these distributions. Most of the clustering approaches in the class of mixture-resolving algorithms assume that the individual data points of the mixture under consideration follow a Guassian distribution [244]. More traditional mixture-resolving algorithms iteratively obtain a maximum likelihood estimate of the parameter vectors of the component densities [243]. More recent mixture resolving algorithms have tended to employ expectation maximisation procedures for estimating data distribution parameters [315]. Non-parametric techniques for density-based clustering also exist, such as the algorithm developed by Jain and Dubes [243].

This algorithm was inspired by the Parzen window approach[6] toward non-parametric density estimations.

The so-called *mean shift algorithm* is the most popular member of the class of mode-seeking algorithms [152]. The algorithm was introduced in 1975 by Fukunaga [175], but was made popular through the work of Cheng [80] and Comaniciu and Meer [93], who showed how finding the modes of a non-parametrically estimated probability density function could be implemented successfully to segment images.

4.2.3 Other clustering algorithms of note

Two additional clustering approaches which do not fall within the general taxonomy provided in Figure 4.1 are described in this section.

4.2.3.1 Rearrangement clustering

Once a cluster has been constructed, it is typically not revisited, although relocation schemes are sometimes utilised to redistribute data points between clusters in the spirit of seeking an improved clustering [43]. Rearrangement clustering requires the problem to be specified in matrix form. The rows in the matrix represent the data points that require clustering, while its columns represent their features [89]. The rearrangement scheme aims to maximise the overall similarity of data clustered together by rearranging the rows of the matrix. Since the columns of the matrix are independent of its rows, the columns can also be rearranged.

Transforming a rearrangement clustering problem into a TSP is relatively straightforward and facilitates leveraging of the capabilities of modern TSP solvers, such as Concorde [12], which are capable of solving large TSP instances quickly [88]. Transforming a clustering problem into a TSP instance has been suggested by several authors in the literature [6, 248]. The main notion behind this approach is that data points within clusters are visited consecutively and movements between clusters require sufficiently larger jumps than the distances between data points of the same cluster. Climer and Zhang [89] suggested adding n_c dummy cities which have large constant distances to all other cities and are infinitely far away from each other, if a clustering of the data containing n_c clusters is sought. An optimal TSP solution will then automatically find the boundaries of n_c clusters, because the dummy cities will separate the most distant cities.

4.2.3.2 Constrained clustering

Clustering is inherently an ill-posed problem — the goal is to partition data into some unknown number of clusters based on intrinsic information alone, while any external or side information may, in fact, be extremely useful in finding good data clusters [242].

In many clustering algorithms that have to deal with high-dimensional data, the focus is on determining the features according to which the data should be classified, whereas in the context of the VRP these features are easily determinable and it is the constraint aspect of the clustering that requires attention [356]. Constrained clustering or semi-supervised clustering has been addressed by relatively few authors in the literature, and those papers that have focused on constrained clustering have only dealt with a single type of constraint. The results of Bradley

[6] According to the Parzen window approach, bins are sought which achieve large counts in a multidimensional histogram of the input data.

et al. [53] illustrated that constrained clustering methods are less likely to become trapped at local minima. There are two aspects to consider when using side information in clustering. The first is determining how the side information should be specified and the second is to consider how side information is gathered in practice [76]. Several important questions have arisen since the use of background or contextual data in clustering algorithms, such as which constraints are most useful, and how they should be propagated to neighbouring points [439].

Tung *et al.* [432] claimed that the fundamental difference between unconstrained and constrained clustering problems is that feasible solutions to the constrained problem instances may not be able to satisfy the nearest representative property (*i.e.* that points should be assigned to the cluster with the representative closest to them). Tung *et al.* [432] imposed existential constraints[7] on individual clusters. Constrained clustering algorithms treat user-defined constraints as hard constraints and criteria such as the nearest representative property as soft constraints. The algorithms can typically accommodate only a single user-defined (hard) constraint, unless multiple conjunctive constraints have sets of pivots that may be manipulated independently [356]. Constraints that require averaging and summation also cannot be accommodated in the framework proposed by Tung *et al.* [432].

In traditional clustering approaches, including fuzzy neural networks and hybrid clustering, only geometric data attributes are typically considered, resulting in data points similar to each other in respect of non-geometric attributes remaining scattered [289]. The dual algorithm presented by Lin *et al.* [289] operates in the constraint (geometric shape) and optimisation (objective function) domain. The authors implemented a stable approach in which the complete-link algorithm is used to cluster and a *support vector machine*[8] (SVM) to classify. This approach was experimentally shown to provide more stable and effective results than those resulting from a modification of the similarity measure in the optimisation domain involving explicit specification of a penalty in the constraint domain and then applying a traditional clustering algorithm.

4.3 Metaheuristic clustering

The basic objective of optimisation search techniques is to find global or approximately global optima for NP-hard combinatorial optimisation problems with large solution spaces. As mentioned, clustering may be regarded as a combinatorial optimisation problem [462]. Simple local search algorithms, such as hill-climbing algorithms, may therefore be utilised to solve clustering problems, but these algorithms are susceptible to becoming trapped at local optima. More complex metaheuristic search algorithms, such as evolutionary algorithms [167], simulated annealing [258], tabu search [191], and deterministic annealing [231], have been applied successfully to clustering problems [462].

Hall *et al.* [217] designed a GA that is able to perform clustering. Several other studies have also successfully applied genetic operators to clustering problems with frameworks similar to that

[7]Let $W \subset D$ be any subset of pivot objects in a database D. Let $O_i \in D$ represent an object and let c be a positive integer. An existential constraint on a cluster $C\ell$ is a constraint of the form $count(O_i|O_i \in C\ell, O_i \in W) \geq c$.

[8]Introduced by Cortes and Vapnik [101], an SVM is a non-probabilistic linear binary classifier. The standard SVM accommodates a set of input data and predicts to which of two classes each data point belongs, after having built a model from a set of training examples, each marked as belonging to one of two categories. An SVM model is a representation of the examples as points in feature space, mapped in a manner such that the categories are divided by a gap that is as wide as possible. A good separation is obtained by the hyperplane that has the largest distance to the nearest training data point of any class, as the larger this margin, the lower the generalisation error of the classifier [101].

described by Hall *et al.* [217], but they differ in their definitions of an individual in the population, their solution encoding schemes, their fitness function definitions and their evolutionary operators [102, 303, 431]. The algorithm proposed by Tseng and Yang [431], for example, also incorporates a heuristic function to approximate the number of clusters. GAs are typically very effective in improving the performance of k-means clustering algorithms. Babu and Murty [19], for example, successfully implemented a GA for approximating an optimal initial seed selection for a k-means clustering algorithm. Krishna and Murty [263] later improved upon the algorithm of Babu and Murty [19] by proposing a fully hybridised genetic k-means algorithm which is capable of determining optimal clusterings for several benchmark data sets. Lozano and Larranaga [295] applied a GA to a hierarchical clustering problem in which they reformulated the clustering problem as an optimisation problem where the algorithm attempts to find the closest ultrametric distance [222] for a given dissimilarity (adopting the Euclidean norm). Tseng and Yang [431] used a single-linkage algorithm to partition the data into small subsets, before applying GA-based clustering in an attempt to reduce computational complexity. The fitness function was designed to adjust the different effects of the within-class and between-class distances as the search progresses.

Al-Sultan [410] designed a tabu search clustering algorithm in which a set of candidate solutions is generated from the current solution by employing a search strategy. The candidate clusterings that attain a pre-defined quality level in respect of the tabu search's objective function are appended to the tabu list (if not already in the tabu list). This is repeated until all candidates have been evaluated, after which the next iteration commences. Sung and Jin [411] introduced superior search processes with the incorporation of a packing and releasing procedure into their algorithm. Delgado *et al.* [117] presented a tabu search which may be applied to fuzzy clustering problems[9] and which produces good solutions in relatively short computation times. Scott *et al.* [394] proposed a hybridised clustering algorithm incorporating a tabu list within a GA to aid in the promotion of population diversity and computational efficiency.

Brown and Huntley [62] put forward a simulated annealing algorithm for evaluating different clustering criteria, while Selim and Alsultan [395] investigated the effect of input parameters on cluster formation when using simulated annealing as the clustering technique. Bandyopadhyay [31] proposed a simulated annealing reversible jump Markov chain Monte Carlo algorithm in which the clusters are independent of user choices and the algorithm is able to handle fuzzy clustering problems through the use of the Xie-Beni index criterion[10] [459]. The cluster centres are dynamically adjusted by applying five different operators, each with a different search paradigm associated with it.

The main drawback of employing algorithms such as those described in this section to solve clustering problems lies in their parameter value selection. Search methods such as these typically introduce more parameters than other methods and there are no theoretical guidelines to assist in this parameter selection. Hall *et al.* [217] provided some methods for determining parameter values, but most of the criteria are still determined empirically.

[9] Fuzzy clustering refers to objects being allowed to appear in multiple clusters with certain degrees of membership [467].

[10] The criterion is based on a validity function that is capable of identifying compact fuzzy partitions without assuming the number of substructures inherent in the data. The validity function depends on the data set, a geometric distance measure, the distance between cluster centroids, and the fuzzy partitions generated by any fuzzy algorithm implemented.

4.4 Clustering algorithms applied to the VRP

Chandran *et al.* [74] developed a clustering algorithm for solving the multiple-TSP. The objective in this problem is to balance the workload/distance between clusters of customers, with each cluster visited by a single salesman. The problem may also be interpreted in the context of a single salesman working over several days, visiting one cluster per day. The cluster length is determined by a nearest neighbour-type calculation [356]. A range of clusters is examined and the solution with the lowest coefficient of variation is chosen. The clusters are built iteratively by selecting those clusters that are furthest apart, making them the seeds. All unassigned points are then allocated to the nearest feasible seed.

Yücenur *et al.* [465] introduced a geometric shape-based genetic clustering algorithm for assigning clusters in the context of the multi-depot VRP. The number of depots is known in advance and each customer is assumed to receive a single visit by some vehicle. A random radius is determined for each depot and customers are assigned to the depot within whose radius they fall. Any point that lies outside these radii is assigned to the depot achieving the smallest Euclidean distance between the data point and the depot's cluster circumference. The algorithm achieves the same results as a simple nearest neighbour algorithm, but requires substantially less computation time.

Sahoo [412] introduced a balanced clustering algorithm which focuses on improving the shapes of vehicle routing clusters. The algorithm was inspired by an observation that most VRP objectives are concerned with minimising time and the number of vehicles required, but that visual attractiveness is also a key factor. A visually appealing cluster has the property that clusters are mutually exclusive (do not overlap). The working of the proposed algorithm consists of two phases, with the first phase focusing on route shapes. An initial solution is generated using a balanced clustering algorithm with insertion. Stops are based on time window and location constraints, and are balanced so that a single vehicle can serve each route. The algorithm also provides a solution to a waste collection problem with a homogeneous fleet of vehicles. The second phase employs a metaheuristic to improve the routes constructed during the first phase.

Beasley and Christofides [38] proposed an algorithm for providing a solution to VRPs with sparse feasibility graphs. The routes are constructed according to postal districts as opposed to the conventional home level[11].

Dondo and Cerdá [134] proposed a three-phase MILP model for the VRPTW that initially clusters customers in an attempt to create several smaller subproblems allowing for larger problem instances to be solved to optimality. During the first phase a feasible set of customer clusters is determined, while the vehicles are assigned to the clusters during the second phase. The customers are finally ordered according to vehicle arrival times during the third phase.

4.5 Determining the number of clusters

Automatically determining the number of clusters into which a data set should ideally be clustered has been one of the more difficult challenges associated with cluster analysis [242]. As

[11]The authors created a special algorithm as they were concerned that single customers would be left isolated. They therefore created a feasibility graph using Delaunay triangulation. The graph was adjusted to accommodate geographic features such as rivers. A Delaunay triangulation partitions the plane into triangles according to a given set of vertices. The triangulation is not unique and each vertex of the triangulation has to be on the circumference of a circle coinciding with the corners of the triangle.

mentioned, the typical approach is to execute a clustering algorithm multiple times with different numbers of clusters and then select the best clustering based on some pre-defined criterion. Figueiredo and Jain [160] designed a method for estimating the optimal number of clusters by combining the minimum message length criterion[12] [440, 441] with a Guassian mixture model [149]. Their approach begins by creating a large number of clusters and then successively merges these clusters together if the merger leads to a decrease in the minimum message length criterion.

Hansen and Yu [219] developed a related approach, but instead adopted the principle of minimum description length[13] for selecting clusters. The other criteria suggested in the literature for determining the number of clusters include the Bayes information criterion [392], the akaike information criterion [381], and gap statistics [423]. The key assumption, when partitioning data into optimal clusters, is that an optimal clustering is most resilient to random perturbations. The so-called Dirichlet process [158, 361] introduces a non-parametric prior for the number of clusters. It is more commonly applied to probabilistic models in order to determine a posterior distribution for the number of clusters, from which the most likely number of clusters may be determined. Despite all of the above criteria it remains difficult to determine an optimal number of clusters — this is often a subjective, empirically-based decision.

4.6 Cluster validation

The validity of data clusters has to be considered as clustering algorithms tend to generate clusters irrespective of whether there are natural clustering features present in the data. Cluster validity may be ascertained based on three different types of criteria: *internal criteria, relative criteria,* or *external criteria* [243]. Validity indices based on *internal* criteria assess the fit between the structure imposed through clustering and the features of the data, using only the features of the data. *Relative* indices involve a comparison of multiple structures (generated by different clustering algorithms) and a decision as to which algorithm produces superior clusters. Indices based on *external* criteria measure the quality of clusters by matching the clustering structure to *a priori* information.

Lange *et al.* [273] introduced the notion of cluster stability to validate data clusters. Cluster stability is a measure of variation in the clustering solution over sub-samples drawn from the input data. Different measures of variation can be used to determine different stability measures. David and Von Luxburg [438] suggested that the distance between model-based algorithms (k-means, Guassian, *etc.*) be used as a measure of the clustering stability. Shamir and Tisby [398] defined stability as the generalisation ability of the clustering algorithm. They argued that since many algorithms may be shown to be asymptotically stable, the rate at which the asymptotic stability is reached with respect to the number of samples drawn is a meaningful measure of stability.

4.7 Admissibility analysis of clustering algorithms

Fischer and Van Ness [163] performed a study in which they formally analysed clustering algorithms with the intention of comparing them and providing guidance in respect of clustering algorithm selection. They introduced a set of *admissibility criteria* for clustering algorithms

[12] In an optimal code, the binary message length of an event E with probability $P(E)$ is given by $\mathbf{length}(E) = \log_2(P(E))$ [399].

[13] The minimum description length principle is a principle in which the best hypothesis for a given set of data is the one that leads to the best compression of the data.

that test the sensitivity of clustering algorithms with respect to imposed changes that do not affect the core structure of the data. A cluster of points is called *A admissible* if it satisfies a criterion A. Four such criteria are:

1. *Convexity.* A clustering algorithm is *convex admissible* if the clusters generated by the algorithm exhibit convex hulls that do not intersect.

2. *Cluster proportion.* A clustering algorithm is *cluster-proportion admissible* if the cluster boundaries produced by the algorithm do not change despite duplicating some of the clusters an arbitrary number of times.

3. *Cluster omission.* A clustering algorithm is *cluster-omission admissible* if, when removing one of the clusters from the data set and executing the clustering algorithm again in respect of the remaining data, it produces identical clusters, excluding the removed cluster, as those produced before the cluster was removed.

4. *Monotonicity* A clustering algorithm is *monotone admissible* if the clusters generated by the clustering algorithm do not change even after performing a monotone alteration on the elements within the similarity matrix.

Fischer and Van Ness [163] proved that it is impossible to construct clustering algorithms that satisfy certain combinations of admissibility criteria. For example, if an algorithm is monotone admissible, it cannot be a hierarchical clustering algorithm.

A similar algorithmic comparative analysis was undertaken by Kleinberg [259], who considered the following three criteria:

1. *Scale invariance.* A clustering algorithm is *scale invariant* if an arbitrary scaling of the similarity matrix does not affect the clusters produced.

2. *Richness.* A clustering algorithm is *rich* if it is able to achieve all possible partitions on the data.

3. *Consistency.* A clustering algorithm is *consistent* if, when shrinking within-cluster distances and stretching between-cluster distances, the clusters generated remain the same.

4.8 Chapter summary

The basic theory behind clustering algorithms was reviewed very briefly in this chapter. A brief introduction to clustering was presented in §4.1 and a motivation for the need of clustering algorithms was given.

There are numerous suggestions in the literature as to clustering techniques for partitioning data into clusters, some of which were discussed briefly in §4.2. A selection of approximate solution approaches proposed for clustering problems was briefly reviewed in §4.3. Notable literature pertaining to clustering algorithms proposed for solving the VRP specifically was also discussed in §4.4.

The first stage in performing clustering is determining the number of clusters, which is a non-trivial procedure. Various approaches that may be employed to determine a suitable number of clusters for a given data set were elaborated upon in §4.5.

In §4.6, the importance of validating the clusters generated by a clustering algorithm was considered, as clustering algorithms tend to cluster data irrespective of whether or not there are natural partitioning features present in the data. The chapter closed in §4.7 with a discussion on measures that may be employed when comparing the performances of clustering algorithms.

Part II

Capacitated Vehicle Routing Problem

CHAPTER 5

CVRP Model Formulation

Contents

A description of the mathematical model adopted in this dissertation for the CVRP is presented in this chapter. A number of necessary assumptions required to formulate the model are presented along with the relevant model objective and constraints. The model is verified by implementing it in a commercially available MIP solver and solving the model in the context of a small problem instance from the literature for which an exact solution is known. The viability of an exact approach towards solving the model is also investigated by solving the model for larger problem instances and noting the solution times required. The chapter closes with a brief summary.

5.1 Introduction

The VRP has received considerable academic and industry attention since its inception, as outlined in Chapter 3. There have been numerous proposals in terms of mathematical model formulations for the CVRP, including the edge-set formulation of Norbet (described in §3.1.1), and the set partitioning formulation (described in §3.1.2). This chapter contains an alternative mathematical model for the CVRP. The aim of this model formulation is to serve as a foundation to be elaborated upon later in this dissertation when a mathematical model is formulated for the TVRPGC. This model borrows constructs from the various CVRP models reviewed in Chapter 3. The mathematical models presented in §3.1.1 and §3.1.2 are, however, inherently more powerful than the model presented in this chapter (as they are able to facilitate solution of much larger instances within acceptable time frames), but they lack the required flexibility to allow the alterations required to arrive at a model of the TVRPGC.

5.2 Model assumptions

In the mathematical model proposed in this chapter, certain assumptions are required in order to render possible a mathematical description of capacitated vehicle routing operations. These assumptions, although simplifying the model, still offer the same functional capabilities of other models proposed in the literature, and are described as follows:

1. *The nature of customers.* The transportation network consists of several customers, with the defining characteristic of a customer being that it exhibits a demand for goods collection by a vehicle. All demand for the goods collected at any customer in the transportation network has a certain volume associated with it, and these goods are required to be delivered to the depot. Each customer in the network may only have its demand satisfied fully by a single vehicle and only one vehicle may visit it over the planning horizon under consideration.

2. *The nature of vehicles.* It is assumed that all vehicles in the fleet are homogeneous and that the fleet size is known *a priori*. The vehicles are homogeneous in the sense that they traverse all arcs in the network at the same cost and they all have the same maximum vehicle capacity for goods collection associated with them.

3. *Home depot allocation.* The transportation network consists of a single home depot for all vehicles within the fleet at which all the vehicles must begin and end their goods collection routes. The location of the depot is fixed.

5.3 Mathematical model formulation

This section contains a detailed description of the sets of constraints and the objective function required to translate the CVRP into a formal mathematical model. After defining the model parameters and variables in §5.3.1 and §5.3.2, respectively, the model is formulated mathematically in §5.3.3.

5.3.1 Model parameters

Denote the set of vertices in the transportation network by $\mathcal{N}$ and let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be an undirected, weighted graph with vertex set $\mathcal{N}$ and edge set $\mathcal{E}$ representing all possible connections between vertices in $\mathcal{N}$, with the weight of an edge $(i, j) \in \mathcal{E}$ representing the expected cost c_{ij} of a vehicle traversing that edge. Let $\mathcal{V}$ represent the set of homogeneous vehicles that constitute the goods collection fleet. As mentioned in §5.2, the homogeneity of the vehicles implies that all vehicles have the same maximum capacity, $\mathcal{C}_{max}$, and that any two vehicles which traverse a given arc in $\mathcal{E}$ incur the same associated cost. Furthermore, let $h \in \mathcal{N}$ be the vertex that acts as the home depot for all vehicles. Finally, denote the demand volume for goods collection at customer $i \in \mathcal{N}$ by d_i and let $d_h = 0$.

5.3.2 Model variables

In the model formulation, decision variables are required to keep track of the movement of vehicles throughout the transportation network. In order to keep track of which vehicle is required to visit which customer, the decision variable

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \in \mathcal{V} \text{ travels directly from vertex } i \in \mathcal{N} \text{ to } j \in \mathcal{N}, \\ 0, & \text{otherwise} \end{cases}$$

is defined. An auxiliary variable u_{jk} is adopted in the spirit of Miller *et al.* [310] to monitor the permutation order in which vehicle $k \in \mathcal{V}$ visits customer $j \in \mathcal{N}$. This variable is employed to break subtours. More specifically, a customer that is visited later is assigned a larger value for this variable than a customer that is visited earlier in the route of any vehicle $k \in \mathcal{V}$.

5.3.3 Model objective and model constraints

The model proposed in this chapter follows the typical convention of most CVRPs in the literature in that the model aims to minimise the total cost associated with goods collection. This objective may be formulated mathematically as

$$\text{minimise} \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{V}} c_{ij}x_{ijk}. \tag{5.1}$$

The model contains numerous constraints reflecting the various requirements of a standard CVRP. The first such constraint is that every customer in the transportation network must be visited by a single vehicle during the planning horizon. The constraint set

$$\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{V}} x_{ijk} \geq 1, \qquad j \in \mathcal{N} \tag{5.2}$$

enforces this requirement. In addition, the flow conservation constraint set

$$\sum_{i\in\mathcal{N}} x_{ijk} - \sum_{\ell\in\mathcal{N}} x_{j\ell k} = 0, \qquad k \in \mathcal{V},\ j \in \mathcal{N} \setminus \{h\} \tag{5.3}$$

states that if any vehicle $k \in \mathcal{V}$ arrives at customer j, then that same vehicle must traverse an arc departing from customer j, for all $j \in \mathcal{N} \setminus \{h\}$. All vehicles within the transportation network must furthermore begin and end their routes at the depot. This requirement is enforced by the constraint set

$$\sum_{i\in\mathcal{N}\setminus\{h\}} x_{hik} - \sum_{j\in\mathcal{N}\setminus\{h\}} x_{jhk} = 0, \qquad k \in \mathcal{V}. \tag{5.4}$$

The key feature of the CVRP is the capacity of any vehicle in terms of the volume of goods that it is able to transport. The constraint set

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} d_j x_{ijk} \leq C_{\max}, \qquad k \in \mathcal{V} \tag{5.5}$$

ensures that the total volume of goods collected by vehicle $k \in \mathcal{V}$ does not exceed the maximum capacity of the vehicle. Finally, as mentioned in §5.3.2, subtour elimination is required. The set of constraints

$$u_{hk} = 1, \qquad k \in \mathcal{V} \tag{5.6}$$

$$2 \leq u_{ik} \leq |\mathcal{N}|, \qquad i \in \mathcal{N} \setminus \{h\},\ k \in \mathcal{V} \tag{5.7}$$

and

$$u_{ik} - u_{jk} + 1 \leq (|\mathcal{N}| - 1)(1 - x_{ijk}), \qquad i \in \mathcal{N} \setminus \{h\},\ j \in \mathcal{N} \setminus \{h\},\ k \in \mathcal{V} \tag{5.8}$$

is an adaptation of the well-known MTZ subtour elimination constraints [310]. The constraints together monitor the order in which a vehicle $k \in \mathcal{V}$ visits customers in $\mathcal{N}$, with a customer $i \in \mathcal{N}$ being visited earlier than a customer $j \in \mathcal{N}$ by some vehicle $k \in \mathcal{V}$ resulting in the variable u_{jk} having a larger value than u_{ik}.

5.4 A small worked example

The logic of the mathematical model of §5.3.3 is verified in this section, by implementing it in a commercially available mixed integer linear programming solver within the context of a well-known CVRP benchmark instance of Christofides and Eilon [82].

The benchmark instance size was chosen to be large enough to offer some computational complexity, but also small enough to be solved within a reasonable amount of time. The benchmark instance contains 22 customers who are serviced by four vehicles, with Customer 1 acting as the home depot for all vehicles. The location and demand of each customer is shown in Table 5.1. The capacity of each vehicle is 6 000 demand units and the cost coefficient c_{ij} is taken as the Euclidean distance between the locations of customers i and j in the plane.

TABLE 5.1: *Locations and demand for customers of the benchmark instance E-n22-k4 in [82].*

Customer	X-coordinate	Y-coordinate	Demand
1	145	215	0
2	151	264	1 100
3	159	261	700
4	130	254	800
5	128	252	1 400
6	163	247	2 100
7	146	246	400
8	161	242	800
9	142	239	100
10	163	236	500
11	148	232	600
12	128	231	1 200
13	156	217	1 300
14	129	214	1 300
15	146	208	300
16	164	208	900
17	141	206	2 100
18	147	193	1 000
19	164	193	900
20	129	189	2 500
21	155	185	1 800
22	139	182	700

The mathematical model of §5.3.3 was implemented in CPLEX 12.5 [235] in respect of the benchmark instance described above, and the instance was solved on an i7-4770 processor running at 3.40 GHz with a working memory limit of 6Gb within the Windows 7 operating system. An initial feasible solution was found within 0.64 seconds, while it required 112 320 seconds (31.2 hours) to reach an optimal solution. This optimal solution is illustrated in Figure 5.1, which incurs a total travel cost (as distance in this case) of 375. The solution in Figure 5.1 corresponds exactly to the optimal solution reported by Christofides and Eilon [82].

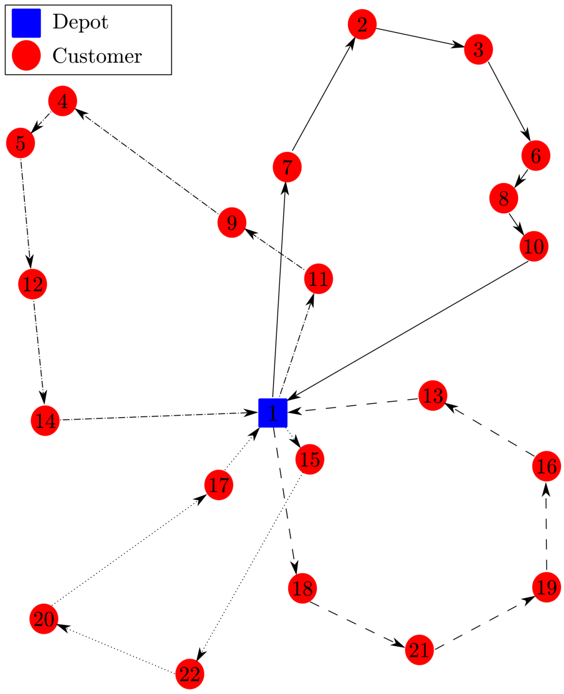


FIGURE 5.1: *An optimal solution to the benchmark instance E-n22-k4 in [82] involving twenty two customers and four vehicles.*

5.5 The complexity of solving two larger CVRP instances

In order to illustrate the computational complexity associated with computing an exact solution for the CVRP, two larger instances of the model of §5.3 were also implemented in CPLEX 12.5 on the same computer described earlier — one with 51 customers (and 5 vehicles) and one with 76 customers (and 8 vehicles). The branch-and-cut algorithm implemented by CPLEX was allocated 10 hours, 20 hours, 30 hours and 40 hours of run time, respectively, and the optimality gaps were recorded in each case.

TABLE 5.2: *Deviation from known optimal for the instances E-n51-k5 and E-n76-k8.*

Benchmark instance	Optimality gap 10 hours	Optimality gap 20 hours	Optimality gap 30 hours	Optimality gap 40 hours
E-n22-k4	4.81%	3.16%	0.05%	—
E-n51-k8	37.43%	33.52%	33.28%	31.70%
E-n76-k8	43.21%	39.24%	37.83%	34.86%

The results in Table 5.2 clearly indicate that the time complexity associated with computing an exact solution to the model of §5.3 for large problem instances is prohibitively large. For this reason it is desirable to investigate to what extent this time complexity may be mitigated at the expense of solution quality by a metaheuristic solution approach.

5.6 Conclusion

The mathematical model presented in this chapter may be solved to optimality within a reasonable time by means of an off-the-shelf commercially available MIP solver for small problem instances as was illustrated in §5.4. The exponential nature of the time complexity associated with solving the mathematical formulation is, however, such that exact solutions are not attainable for larger instances within a reasonable computational budget. Although there are more efficient model formulations in the literature [96, 174, 340] facilitating exact solutions of the CVRP instances involving 200 customers and 12 vehicles, the computation times required to reach these exact solutions are prohibitive. For this reason, metaheuristic solution procedures are developed for the CVRP in the following chapter. Although these procedures rarely yield exact solutions, they are nevertheless capable of producing high-quality solutions to even relatively large instances of the CVRP rather quickly.

CHAPTER 6

CVRP Solution Methodology

Contents

A number of different solution techniques that have been applied successfully to the CVRP were reviewed in §3.2–§3.8, varying from classical exact techniques to modern metaheuristics. In Chapter 5, the need was highlighted for an approximate solution approach in the case of large instances of this problem. An evolutionary (population-based) metaheuristic and a swarm intelligence metaheuristic for solving the CVRP are presented in this chapter. The evolutionary approach is based on the GA developed by Deb *et al.* [116] in conjunction with selected crossover and mutation operators adapted from the work of Puljić and Manger [353] among others. The swarm intelligence metaheuristic is based on the ant colony algorithm developed by Lee *et al.* [280] and Tan *et al.* [415] for generating approximate solutions to several CVRP benchmark instances. Each of the algorithms is elaborated upon, elucidating integral algorithmic components by means of pseudo-code descriptions. The chapter closes with a brief summary.

6.1 A genetic algorithm

GAs use random choices to guide a highly exploitative search through the solution space of an optimisation problem and are not limited by restrictive assumptions on the nature of the objective function of the problem (such as unimodality or the existence of derivatives). Goldberg [194] highlighted four reasons why GAs are successful in solving hard optimisation problems:

1. GAs work with a coding of the decision variable set, not the variables themselves.
2. GAs search from a population of points, not from a single point.
3. GAs use objective function information directly, not derivatives of or auxiliary knowledge about this function.
4. GAs use probabilistic transition rules (as opposed to deterministic rules).

These reasons, accompanied by several others, have led to GAs being adopted widely as a solution approach in the context of VRPs. There are several key components in the implementation of any GA, including the method of initial population generation, the selection method, a crossover operator and a mutation operator. Each of these components is described briefly in the remainder of this section, indicating exactly how they were implemented in the GA design of this dissertation for solving the CVRP.

6.1.1 Chromosome representation

Solution representation or chromosome representation is a key decision in the design of any optimisation algorithm. Chromosome representations can vary from continuous to binary, with the typical approach in VRP instances involving a vector of integers encapsulating the routes to which the various vehicles have been assigned. The vector sequence represents the order in which the vehicles visit customers. Most solution representations associate the value 0 with the depot. The depot, in the solution representation adopted in this dissertation, is not, however, assigned the value 0, due to limitations in the data structures of Rstudio (the software environment in which the GA was implemented). Rstudio does not use zeros in its data or loop structures, which made it easier to assign the depot a value of 1, with customer i being represented by the value $i+1$. A solution for an instance with two vehicles and eleven customers may, for example, be represented by

1	12	8	6	9	11	7	1	2	4	3	5	10	1

,

with 1 denoting the depot. In this solution, one vehicle departs from the depot and visits customers 11, 7, 5, 8, 10 and 6 (in this order) before returning to the depot. A second vehicle similarly departs from the depot and visits customers 1, 3, 2, 4 and 9 (in this order) before also returning to the depot.

6.1.2 Initial population

Population sizing has been one of the most important research topics in evolutionary computation [5]. The first step in the execution of a GA is to generate an initial population, with each member of the population representing a candidate solution to the optimisation problem at hand. The initial population is a key component in obtaining a good final solution. There are several factors to consider when selecting an initial population, including the number of individuals in the population, the diversity of the population, the fitness function according to which the quality of solutions are to be measured, the nature of the search space, the problem complexity and selection pressure [128]. Burke *et al.* [67] claim that a small population size may guide the algorithm towards poor solutions, while a large population size will expend more computational effort. Several authors [221, 342, 464] believe that the population size should be in direct relation to the difficulty of the problem (*i.e.* that larger problem instances require larger population sizes).

The general expectation is that an initial population of reasonably structured solutions will evolve to high-quality solutions within a relatively small number of iterations. This may, however, possibly come at the cost of a lack of diversity, which is required to obtain near-optimal solutions [23]. Two approaches toward initial population construction typically encountered in the literature are implemented in the GA design adopted in this dissertation and are compared in order to determine the desirability of structured initial populations *versus* random initial populations with respect to the eventual solution quality achieved by the GA.

The first approach is based on the algorithm proposed by Clarke and Wright [87], described in §3.7.2.1. The first step in this algorithm is to calculate *savings* values for each customer, which would be experienced by additionally including a customer in an existing route as opposed to having a vehicle depart from the depot, visit the relevant customer and then return directly to the depot. In an attempt at validating the implementation of the aforementioned method, the author compared the population thus generated with that reported in [359] for a small VRP instance. It was, however, found that the results published in the article were incorrect (the capacity reported in [359] is infeasible and there is a total capacity difference between the reported solution and problem declaration).

The second approach involves generating initial solutions randomly. A customer is selected at random to be visited first by the first vehicle. Another customer is randomly selected to be visited next by this vehicle from a list of remaining, feasible customers (respecting the vehicle capacity). This process is repeated until no feasible customers remain. If all customers have been assigned for vehicle visitation, the process terminates. Otherwise the process is repeated for a second vehicle in respect of the customers who have not yet been assigned for visitation. This overarching process is repeated, using as many vehicles as are required, until all customers have been assigned for visitation.

A combination of the aforementioned methods of population generation was implemented in the GA put forward in this dissertation for solving CVRP instances. More specifically, the algorithm is able to implement the respective methods in equal proportions when generating an initial population. The relative algorithmic performance resulting from the population size is determined by means of a parameter sensitivity analysis in the following chapter.

6.1.3 Selection method

The selection phase of each iteration of the GA consists of probabilistically selecting two parent chromosomes from the population for reproduction purposes. Two selection methods are implemented in the GA of this dissertation, and these methods are described in this section.

A popular method of selection is *roulette wheel selection* [195]. This approach is based on the notion of a hypothetical roulette wheel which has a sector associated with each chromosome within the population. The chromosomes that are more attractive (in terms of the fitness function associated with the optimisation problem at hand) are allocated a larger sector arc on the wheel, thus increasing their chance of being selected. This approach does not guarantee that fitter chromosomes are selected, only that, on average, a chromosome will be chosen with the probability proportional to its fitness. The first stage is to calculate the fitness

$$F = \sum_{i=1}^{P} X_i \tag{6.1}$$

of the population, where X_i denotes the fitness of chromosome i and P is the population size. The second stage is to calculate the selection probability p_i for each chromosome X_i as

$$p_i = \frac{F - X_i}{F(P-1)}, \tag{6.2}$$

which normalises the fitness of each chromosome relative to the population. The third stage is to determine the cumulative probability

$$q_i = \sum_{j=1}^{i} p_j, \qquad i = 1, \ldots, P, \tag{6.3}$$

and to generate a random number R in the range [0,1]. The final stage is to select chromosome i if $q_{i-1} < R \leq q_i$.

Another popular selection method is *tournament selection* [195]. The tournament selection strategy is a fitness-based selection scheme which begins by randomly selecting k chromosomes from the population. A random number r is generated and if this random number is below a certain predefined threshold, the fittest individual from the set of k candidates is chosen. Otherwise, an individual is selected randomly from the set of k candidates.

The GA implemented in this dissertation incorporates both the aforementioned selection methods by allocating each method equal opportunity when selecting chromosomes for the entire duration of the algorithmic execution.

6.1.4 Crossover Operators

Two popular VRP crossover operators are described and illustrated by means of examples in this section. The two operators are employed in equal proportions in the GA adopted in this dissertation. The proportion of the population to which the crossover operators are applied is determined by the so-called *crossover rate*, which is a real number within the range $[0, 1]$.

Each of the crossover operators implemented in the GA remove the depot (and decrement the index of each customer by one) from the chromosome under consideration before applying any of the mechanisms to create offspring. This methodology is employed to avoid generating infeasible routes, although implementations that allow for infeasible route considerations sometimes leave the depot vertex in the chromosome. The GA implemented in this dissertation performs all the necessary crossover operations on the altered chromosome and afterwards reintroduces the depot back into the chromosome so as to maintain feasibility with respect to the vehicle capacity constraint.

The first crossover operator is adapted from the work of Puljić and Manger [353], in which eight evolutionary crossover operators were compared in the context of the VRP. The crossover considered here is based on the results reported in [353], in which the *heuristic greedy crossover* (HGreX) operator [345] was found to outperform the other crossover operators.

The HGreX operator exhibits similar characteristics to the well-known AEX operator [277]. The child chromosome is formed by selecting from each vertex the cheaper of the two respective parent arcs. In case of infeasibility, a feasible customer is randomly selected. Consider, for instance, the parent chromosomes

$$\begin{aligned} p_1' &= (1\ 6\ 2\ 8\ 9\ 1\ 5\ 10\ 7\ 3\ 4\ 1), \\ p_2' &= (1\ 4\ 7\ 3\ 6\ 2\ 1\ 10\ 9\ 5\ 8\ 1), \end{aligned}$$

from which the depot may be removed, transforming the respective chromosomes to

$$\begin{aligned} p_1 &= (5\ 1\ 7\ 8\ 4\ 9\ 6\ 2\ 3), \\ p_2 &= (3\ 6\ 2\ 5\ 1\ 9\ 8\ 4\ 7), \end{aligned}$$

with respective traversal costs of

$$\begin{aligned} p_1 &: c_{51} = 2, c_{17} = 2, c_{78} = 6, c_{84} = 8, c_{49} = 3, c_{96} = 6, c_{62} = 4, c_{23} = 4, c_{35} = 3, \\ p_2 &: c_{36} = 6, c_{62} = 4, c_{25} = 2, c_{51} = 2, c_{19} = 6, c_{84} = 8, c_{47} = 3, c_{73} = 5, c_{98} = 8. \end{aligned}$$

The crossover scheme initialises the child chromosome by randomly selecting a vertex, for instance 5. Both arcs exiting vertex 5 in parents p_1 and p_2 are evaluated. In this particular case both these arcs enter vertex 1, resulting in the child chromosome

$$c = (5\,1 * * * * * * *).$$

During the next iteration, both arcs exiting vertex 1 are considered, and arc c_{17} achieves the lower cost. The resulting child chromosome at this point is therefore

$$c = (5\,1\,7 * * * * * *).$$

This process is repeated until the child chromosome appears as

$$c = (5\,1\,7\,3\,6\,2 * * *).$$

Now both arcs exiting from vertex 2 would result in subtour formation. Thus all the remaining unvisited vertices are compared, and the arc that joins vertex 2 and the unvisited vertex associated with the lowest cost is selected. Continuing in this fashion eventually leads to the child

$$c = (5\,1\,7\,3\,6\,2\,8\,4\,9).$$

The depot is inserted back to the chromosome after all the crossover operations are completed. The positions of the reinserted depots are chosen to ensure that all constraints are satisfied while aiming to minimise the total travel distance. Reinsertion of the depot into the chromosome c above yields the child

$$c' = (1\,6\,2\,8\,4\,7\,1\,3\,9\,5\,10).$$

The second crossover procedure was adapted from the work of Pierre and Zakaria [348], in which the *partially optimised cyclic shift crossover* (POCSX) operator was proposed. The operator is, in fact, an adaptation of the HGreX operator described above. The main difference between the POCSX and HGreX operators is that the POCSX operator does not limit its candidate vertex selection to two, but rather employs a pooling component to increase the set of potential candidates. The parent chromosomes are again slightly altered in that all depot delimiters are removed so that just the set of customer vertices remain (in the same manner as the previous example). The POCSX operator requires the values of two parameters to be specified before implementation, these being a *pool size parameter* and an *offset parameter*. Suppose both the pool size and the offset are 2, and consider the same edge costs as above. The process is illustrated iteratively in Table 6.1, with the shaded cells in the table representing the pool of vertices from which the algorithm may select. The child is initialised by selecting from the pool the vertex that is the shortest distance away from the depot. In this instance, the arc between the depot and vertex 2 is more attractive (assuming it has the shortest associated distance from the depot), resulting in it being placed in the child chromosome and being removed from the two parent chromosomes, as illustrated in Table 6.1(a).

During Iteration 1, the arc corresponding to the shortest distance exiting vertex 2 within the pool is the arc joining vertices 2 and 5. Thus vertex 5 is placed in the child chromosome and vertex 2 is removed from both parent chromosomes, as shown in Table 6.1(b).

This procedure is repeated four times, with the resulting chromosome illustrated in Table 6.1(c). The offset has reached the end of parent p_2 and thus has to be reset (starting from the beginning of the chromosome).

The crossover scheme continues in this fashion until the child has been assigned all unvisited vertices, resulting in the chromosome depicted in Table 6.1(d). The above-mentioned crossover mechanisms are implemented in the GA in equal proportions in this dissertation. The effectiveness of the implementation of the crossover strategy with respect to the crossover rate is determined through an extensive parameter sensitivity analysis in the following chapter.

TABLE 6.1: *Iterative overview of the POCSX operator applied to a hypothetical CVRP instance.*

p_1	5	1	7	8	4	9	6	2	3
p_2	3	6	2	5	1	9	8	4	7
child	*	*	*	*	*	*	*	*	*
pool	5	2							

(a) *Iteration 0*

p_1	5	1	7	8	4	9	6	3	*
p_2	3	6	5	1	9	8	4	7	*
child	2	5	*	*	*	*	*	*	*
pool	5	1							

(b) *Iteration 1*

p_1	4	6	3	*	*	*	*	*	*
p_2	3	6	4	*	*	*	*	*	*
child	2	5	1	7	9	8	*	*	*
pool	3	4							

(c) *Iteration 5*

p_1	*	*	*	*	*	*	*	*	*
p_2	*	*	*	*	*	*	*	*	*
child	2	5	1	7	9	8	4	3	6
pool	*	*							

(d) *Iteration 8*

6.1.5 Mutation operators

Mutation plays a key role in any GA, since it prevents the algorithm from becoming trapped at a local minimum or maximum. The proportion of the population that is subjected to mutation is determined by a pre-defined parameter referred to as the *mutation rate*, which is a real number within the range $[0, 1]$. Two mutation operators are implemented in this dissertation.

The first mutation operator is referred to as *swap mutation* in the literature. This operator employs a uniform mutation [308] principle, meaning that every element in the chromosome has an equal probability of undergoing mutation. The mutation scheme begins by randomly generating two random natural numbers r_1 and r_2 within the set $\{1, 2, \ldots, L_n\}$, where L_n denotes the length of the chromosome and the values of r_1 and r_2 denote the positions of the elements within the chromosome that will be altered. These two numbers are then used to perform a swap of the relevant elements within the chromosome. Suppose, for instance, that $r_1 = 2$ and $r_2 = 7$. Then the chromosome (after removal of depot entries)

$$p_1 = (4\,\mathbf{8}\,6\,7\,3\,\mathbf{5}\,1\,2)$$

is transformed into

$$p_1 = (4\,\mathbf{5}\,6\,7\,3\,\mathbf{8}\,1\,2)$$

according to the swap mutation operator. The resulting chromosome, although very similar to the original, may differ substantially in its fitness value due to the discrete nature of route distances.

The second mutation operator is referred to as *inversion mutation* [183], and involves selection of a substring within the parent chromosome and reversing the selected substring in order to create a mutated chromosome. The resulting chromosome lacks the specific vertex visitation characteristics of the original chromosome, and is therefore used to increase diversity rather than to enhance population solution quality during the optimisation search process [230]. Suppose, for instance, $r_1 = 4$ and $r_2 = 7$. Then the chromosome (after removal of depot entries)

$$p_1 = (4\,8\,6\,\mathbf{7}\,\mathbf{3}\,\mathbf{5}\,1\,2)$$

is transformed into

$$p_1 = (4\,8\,6\,\mathbf{5}\,\mathbf{3}\,\mathbf{7}\,1\,2).$$

The above-mentioned mutation operators are both employed in the GA implemented in this dissertation. The mutation operators are implemented in a manner as to allow the algorithm to utilise both the operators in equal proportions. The relative algorithmic performances as a result of the mutation rate adopted will be determined by means of a parameter sensitivity analysis in the following chapter.

6.1.6 Working of the algorithm

The working of the GA is described in pseudo-code form in Algorithm 6.1. The algorithm initialises with an initial population being generated by one of the techniques described in §6.1.2. The population is then ordered in terms of the fitness of each individual chromosome. In the case of the CVRP, the fitness is determined by the total travel cost of all routes undertaken by the fleet of vehicles in a solution. The while-loop spanning lines 1–34 is then performed until no improvement is experienced over Ω iterations, during which standard GA operations are performed. Here Ω is a user-specified parameter. The first such operation is repeatedly selecting pairs of parent chromosomes on which to perform one of the crossover techniques described in §6.1.4. The selection procedure follows one of the methods described in §6.1.3. Following the application of crossover operations on all pairs of chromosomes earmarked for crossover according to the crossover rate, a new population is formed of which a certain portion (as dictated by the mutation rate) is subjected to mutation according to one of the methods described in §6.1.5 in an attempt to improve solution diversity. Finally, a 2-*opt method* [106] mutation is applied over the population if the population solution quality does not improve over ω while-loop iterations. The value of ω is pre-specified so that $\omega \ll \Omega$.

The 2-*opt* mutation referred to above was derived from the seminal work of Croes [106], in which a greedy heuristic was developed for solving the TSP. The 2-*opt method* has a time complexity of $O(n^2)$, where n denotes the length of the subset of customers under consideration which, when performed several thousand times on large CVRP instances, becomes computationally rather expensive. Thus a decision was taken to limit the application of the 2-*opt method* to single vehicle routes as opposed to applying it to all of the routes in an attempt to reduce computational complexity. Thus all the routes are locally improved through the application of the 2-*opt method*.

6.2 An ant colony system algorithm

ACSs are one of the most successful strands of swarm intelligence algorithms [51, 50], achieving especially good results when applied to the TSP and variations on the VRP [11]. This is the reason why the ACS was selected for implementation in this dissertation as an alternative approximate approach toward solving the CVRP.

In any ACO algorithm, the two main phases are the ants' route construction and pheromone update procedures, as described briefly in Chapter 3. Lee *et al.* [280] implemented a *Simulated Annealing* (SA) algorithm to initialise the pheromone matrix. The value $\tau_0 = ML^*$ is used to generate the initial pheromone matrix, where M denotes the number of ants and L^* is the total travel distance of the incumbent solution yielded by the SA algorithm. The ACO algorithm implemented in this dissertation, however, implements the Clarke and Wright algorithm, described in §6.1.2, to generate the value L^* in an attempt at reducing the computational complexity.

Algorithm 6.1: Genetic algorithm

Input : Initial population, Ω, crossover rate, mutation rate, ω
Output: Final population

```
while tracker ≤ Ω do
    tracker = tracker + 1;
    traveldist = traveldistance(population, locations);
    cross = order(population,traveldist,hightolow);
    average = mean(traveldist);
    min = min(traveldist);
    if min ≤ incumbent then
        incumbent = min;
        tracker = 0;
    for j ← 1 to numbest do
        newpop[j] = population[cross[j]];
    c = 0;
    for k ← 1 to numcross do
        c = c + 1;
        fitness = fitness(traveldist);
        if k mod 2 = 0 then
            selection = roulette(fitness);
        else
            selection = tournament(fitness);
        offspring = crossover(population, selection, c);
    for ℓ ← (j + 1) to length(population) do
        newpop[ℓ] = offspring[ℓ − j];
    for m ← 1 to nummutation do
        gene = newpop[select];
        if m mod 2 = 0 then
            gene = inversemutation(gene);
        else
            gene = swapmutation(gene);
        newpop[select] = gene;
    if tracker mod ω = 0 then
        for imp ← 1 to 5 do
            gene =twoopt(gene);
            newpop[select] = gene;
    population = newpop;
Return population
```

6.2.1 Route construction

The route construction mechanism consists of M ants concurrently building routes from starting vertices randomly chosen in the network of N customer vertices. At each construction step, each ant applies a probabilistic proportional rule, as suggested by Wang *et al.* [444], to determine which vertex to visit next. The node selection mechanism involves the following three parameters:

1. A heuristic value η_{ij} denotes the attractiveness of a move along the arc joining vertex i to vertex j.
2. A parameter τ_{ij} denotes the pheromone level along the arc joining vertex i to vertex j, which is an indication of the past usefulness of the arc in previous route constructions.
3. A parameter s_{ij} denotes the savings value associated with including vertices i and j in a single vehicle route, as described in §6.1.2.

The route construction process also involves three parameters α, β, and γ during vertex selection. During route construction, ant k, located at vertex i, moves to an adjacent vertex according to the following pseudo-random proportional rule: A real number λ is randomly generated within the interval $[0, 1]$ according to a uniform distribution, and if this variable value is below a pre-determined threshold λ_o, the index of the vertex visited next is

$$n = \text{argmax}_{j\in\mathcal{N}_i^k}\{(\tau_{ij})^\alpha(\eta_{ij})^\beta(s_{ij})^\gamma\}, \tag{6.4}$$

where $\mathcal{N}_i^k$ denotes the set of feasible neighbours of vertex i for ant k (where feasibility refers to respecting the vehicle capacity). Otherwise, the probability of visiting vertex $j \in \mathcal{N}_j^k$ next is given by

$$\mathcal{P}_{ij} = \frac{(\tau_{ij})^\alpha(\eta_{ij})^\beta(s_{ij})^\gamma}{\sum_{\ell\in\mathcal{N}_i^k}(\tau_{i\ell})^\alpha(\eta_{i\ell})^\beta(s_{i\ell})^\gamma}. \tag{6.5}$$

This probability is used in conjunction with a roulette selection approach (see §6.1.3) to determine which vertex ant k should visit next, allowing for a biased exploration of the arcs.

6.2.2 Pheromone updating

The ACS allows for two phases of pheromone updates, a global updating method and a local updating method. The local pheromone update is adapted from Tan *et al.* [415], and is performed every time an arc is traversed. The local pheromone level along the arc (i, j) is updated as

$$\tau_{ij} \leftarrow \left(\rho + \frac{\delta}{L_k}\right)\tau_{ij}, \tag{6.6}$$

where ρ and δ are both user-defined parameters. The parameter ρ is referred as the *trail persistence* in the literature and typically is a real value within the range $[0, 1]$, while δ is an elitist-related parameter which typically takes an integer value within the range $[0, L^*]$, where L^* is the length of the incumbent solution. The variable L_k refers to the total travel distance of the route traversed by ant k.

The global update is, however, only applied to those arcs that appear in the best solutions uncovered by the entire colony of ants, by applying the substitution

$$\tau_{ij} \leftarrow \tau_{ij}\sum_{r=1}^{\sigma}\Delta\tau_{ij}^r + \Delta\tau_{ij}^* \tag{6.7}$$

to the relevant arcs, where

$$\Delta\tau_{ij}^{r} = \begin{cases} \frac{(\sigma-r)}{L_k}, & \text{if the } r^{th} \text{ best ant traverses arc } (i,j) \\ 0, & \text{otherwise} \end{cases}$$

and

$$\Delta\tau_{ij}^{*} = \begin{cases} \frac{\sigma}{L^*}, & \text{if arc } (i,j) \text{ is contained within the incumbent solution} \\ 0, & \text{otherwise.} \end{cases}$$

Accordingly, only the σ most elitist ants will deposit a pheromone trail in which the solution quality returned by the ant determines the quantity of pheromone deposited by the ant. This approach was suggested by Bullnheimer *et al.* [64] in an attempt to provide strong additional reinforcement of the edges belonging to the best solutions found so far. The incorporation of the ranking mechanism is aimed at avoiding the danger of over-emphasized pheromone trails caused by many ants following suboptimal routes.

6.2.3 Tour refinement

The algorithm incorporates the use of the 2-*opt method*, as described in §3.7.3.1. The same implementation approach, as described in §6.1.6, is adopted in an attempt to reduce the computational burden of the ACS algorithm.

6.2.4 Chromosome representation

The same chromosome representation described in §6.1.1 is also implemented in the ACS. During the pheromone-update stages of the algorithmic execution, the chromosome is again altered slightly so as not to include the depot in the tour representation. In such cases, the depot is added back into the representation at a later stage of algorithmic execution, as mentioned in §6.1.4.

6.2.5 Working of the algorithm

A high-level pseudocode description of the ACS employed in this dissertation is presented in Algorithm 6.2. The algorithm is initialised by calculating the heuristic and initial pheromone values, as discussed in §6.2.1. The ants then concurrently perform the same route construction approach during every iteration, thereby each iteratively building a full set of vehicle routes by selecting the next customer based on a probabilistic variable value, as described in §6.2.1. During each iteration, several local pheromone updates are performed, with an evaporation component incorporated to encourage exploration, and one global pheromone update is implemented in respect of the incumbent solution.

Finally, Algorithm 6.3 contains a pseudo-code description of the decision making involved in selecting the next vertex traversed by an ant, as discussed in §6.2.1. This selection process includes a probabilistic component allowing, in some cases, for a poorer candidate to be selected with a view to potentially improve solution diversity.

Algorithm 6.2: Ant colony system algorithm

Input : The number of customers (N), number of ants (M), number of iterations with no improvement, coordinates of the customers, demand of the customers, parameter values for $\alpha, \beta, \gamma, \lambda_o, \delta, \rho$

Output: Incumbent CVRP tour

$\eta_{ij} \leftarrow$ attractiveness($coordinates$);
$\tau_{ij} \leftarrow$ pheromoneinitial($coordinates$);
$s_{ij} \leftarrow$ clarkewright($distances$);
$choice_{ij} \leftarrow \tau_{ij} \times \eta_{ij}^{\beta} \times s_{ij}^{\gamma}$;
while $tracker \leq number\ of\ iterations\ of\ no\ improvement$ **do**
    $tracker = tracker + 1$;
    $starting \leftarrow$ startingnodes(M, N);
    for $k \leftarrow 1$ **to** M **do**
        $i \leftarrow starting[k]$;
        $tour[k, 1] \leftarrow starting[k]$;
        $load[k, 1] \leftarrow demand[starting[k]]$;
        $tabu[k, 1] \leftarrow starting[k]$;
    for $istep \leftarrow 1$ **to** $mstep$ **do**
        for $k \leftarrow 1$ **to** M **do**
            $i \leftarrow tour[k, istep]$;
            $check \leftarrow tabu[k,]$;
            $n \leftarrow$ selectnext($tabu, tour, \lambda_o, choice$);
            $tour[k, istep + 1] \leftarrow n$;
            if $n \leftarrow depot$ **then**
                $load[k, istep + 1] = 0$;
            else
                $load[k, istep + 1] \leftarrow load[k, istep] + demand[n]$;
                $tabu[k,$length($tabu$)$] \leftarrow n$;
        $\tau_{ij} \leftarrow$ localpheromone(τ_{ij},arc$_{ij}$,τ_o);
    $tour[\ ] \leftarrow$ twoopt($tour[\ ]$);
    $tourdists \leftarrow$ tourscalc($tour[\ ]$);
    $min \leftarrow tour$[which.min($tourdists$)];
    if $min \leq incumbent$ **then**
        $incumbent = min$;
        $tracker = 0$;
    $\tau_{ij}[\ ] \leftarrow$ globalpheromone($\tau_{ij}[\ ], incumbent, coordinates$);
Return $incumbent$

Algorithm 6.3: Vertex selection

Input : Customers already visited (tabu), the current capacity of goods collected (load), pheromone levels, attractiveness values of each arc, maximum capacity of vehicles

Output: Vertex to be added to partial route

```
allowed ← empty.vector;
if current.vertex = N then
    for i ← 1 to N do
        if !(i in tabu[k, ]) then
            allowed ← allowed + i;
    move ← sample(allowed);
else
    λ ← runif(1,0,1);
    feasible ← feasiblelist(load, tabu, demand, N, capacity);
    if length(feasible) == 0 then
        move ← depot;
    else if λ ≥ λ_o then
        move ← probability(feasible, pheromone, attractiveness);
    else
        move ← bestselect(feasible, choice, current.vertex);
Return move
```

6.3 Chapter summary

The aim in this chapter was to provide the reader with a clear explanation of the mechanisms employed in this dissertation to reach solutions to instances of the CVRP. The working of the GA adopted for solving the CVRP was briefly described in §6.1, considering its key aspects, namely the chromosome representation (§6.1.1), the method of initial population generation (§6.1.2), the method of chromosome selection (§6.1.3), the working of the crossover operators (§6.1.4), and finally the working of the mutation operators (§6.1.5). A pseudo-code description of the GA was finally presented in §6.1.6.

The second algorithm designed for solving instances of the CVRP in this dissertation is an ACS. The implementation of this algorithm was briefly described in §6.2, again considering its key aspects, namely the method of route construction (§6.2.1), the pheromone updating mechanism (§6.2.2), the method of tour refinement (§6.2.3), and finally chromosome representation (§6.2.4). A pseudo-code description of the working of the ACS followed in §6.2.5.

CHAPTER 7

CVRP Parameter Sensitivity Analysis and Results

Contents

As the application of the two CVRP approximate solution approaches of Chapter 6 require problem instance-dependent parameter settings, an extensive parameter sensitivity analysis is performed with respect to three well-known CVRP benchmark instances of Christofides and Eilon [82] in this chapter. The benchmark instances are described in detail in §7.1, after which the experimental design adopted is described in §7.2. The parameters employed in both the ACS and GA are elaborated upon in §7.3 and §7.4, respectively, elucidating the best combination of values assigned to these respective parameters. The most suitable parameter values uncovered during this parameter sensitivity analysis are then employed and the results returned by the algorithms are presented in §7.5 within the context of the aforementioned three benchmark instances of §7.1 in order to compare the relative performances of the algorithms. The relative performances of the algorithms are discussed in §7.6. The chapter closes in §7.7 with a brief summary of its contents.

7.1 Benchmark test instances

The solution techniques described in Chapter 6 are applied in this chapter to three well-known CVRP test instances from the Christofides and Eilon data set [82]. These test instances were chosen to represent respectively small, medium and large benchmark instances in order to determine the scaling effect of instance size on desirable values for the algorithmic parameters.

7.1.1 The E-n22-k4 instance

The E-n22-k4 test instance was considered in §5.4 in order to validate the mathematical model adopted for the CVRP in this dissertation. There are twenty two customers in this instance and the minimum number of vehicles is four, with the maximum capacity of each vehicle set to 6 000. The location and demand of each customer are shown in Table A.1, and the optimal objective function value for this test instance is 375. The depot is located at Customer 1 in this instance.

7.1.2 The E-n51-k5 instance

Th E-n51-k5 test instance has 51 customers and requires a minimum of five vehicles to satisfy the demand fully. The maximum capacity of each vehicle is 160, and the optimal objective function value is 521. The location and demand of each customer are shown in Table A.2, with the depot again located at Customer 1.

7.1.3 The E-n76-k8 instances

The E-n76-k8 test instance has 76 customers and requires a minimum of eight vehicles to fully satisfy the demand, with the maximum capacity of each vehicle set to 180. The location of the depot is at Customer 1, and the location and demand of each customer are shown in Table A.3. The optimal objective function value associated with this instance is 735. The locations of the customers in this instance have been utilised in several of the benchmark test instances published in [82], with different maximum capacities of the vehicles over the different benchmark test instances.

7.2 Experimental design

The CVRP model of Chapter 5 is solved thirty times in this chapter for each of the test instances in §7.1, for each of a number of parameter value configurations, and by both of the algorithms described in Chapter 6. Each algorithm is allowed to run until no improvement is experienced with respect to solution quality over 500 iterations. In addition, an exact solution approach is implemented in CPLEX 12.5 and is allotted a computation time budget of ten hours in each case, after which the best recorded solution is noted.

The algorithmic parameter sensitivity analysis is performed according to the experimental design described above in a similar manner for both algorithms of Chapter 6. The ACS employs several parameters that require a sensitivity analysis for successful implementation. These parameters are the parameters δ, λ, ρ, γ, α and β, as well as the number of ants implemented. The parameters are each set to base configuration values and are then altered individually according to a pre-defined range from these base values (keeping the remainder of the parameter values fixed at the base configuration values). The results are recorded in an attempt to determine the sensitivity of the parameters incorporated in the ACS of Chapter 6 for solving the CVRP. The base parameter values for the sensitivity analysis pertaining to the ACS are shown in Table 7.1, as recommended in [50] and [140].

TABLE 7.1: *Base configuration values for the parameter sensitivity analysis of the ACS.*

Parameters						
δ	λ_o	ρ	γ	α	β	no. of ants
80	0.9	0.8	9	2	5	21

The GA of Chapter 6 also employs several key parameters, the most notable parameters being the crossover rate and the mutation rate. These parameters, along with the population size parameter, are analysed in this chapter in an attempt to obtain the best GA parameter configuration. The base configuration value for the crossover rate is 0.7 and that of the mutation rate is 0.1, while the population size is fixed at thirty chromosomes, as suggested in [419].

All the numerical work reported in this chapter was performed on an i7-4770 processor running at 3.40 GHz with 8GB of memory within the Windows 7 operating system after having implemented the algorithms of Chapter 6 in R.

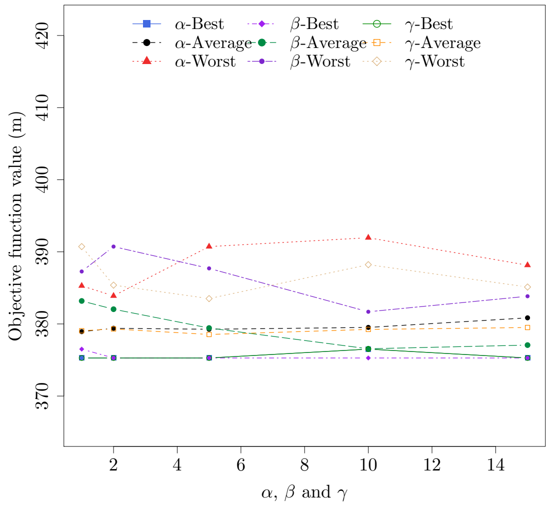
Due to considerable computation time requirements, the parameter sensitivity tests described above were, however, conducted on several computers with the processing capabilities mentioned above and using multiple R sessions running concurrently on each computer. This led to the decision not to record computation time during the parameter sensitivity analysis as there are several external factors that could have affected the computation time of an algorithm. General computation time tendencies were noticeable, but were not accurate enough to draw definitive conclusions.

7.3 Parameter sensitivity of the ant colony system

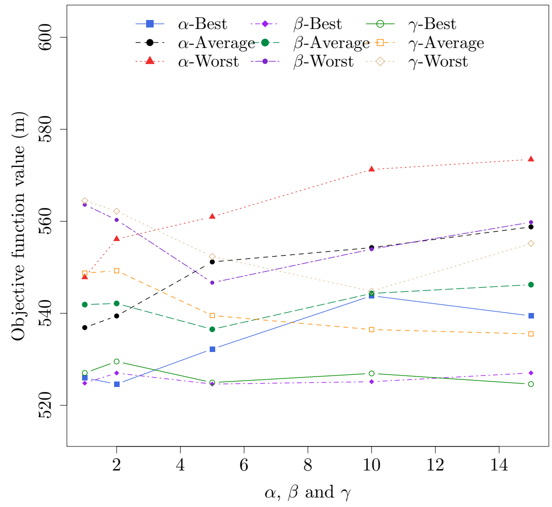
The parameters α, β and γ determine the respective weightings of the heuristic function and pheromone function in the selection of potential vertices for visitation by ants. This can be seen in the expressions in (6.4) and (6.5), which control the probabilistic selection fitness value of each vertex. The performance of the algorithm with respect to changes in α, β and γ is shown for the three CVRP instances of §7.1 in Figures 7.1(a)–7.1(c), together with the deviations in objective function values from that of the optimal solution shown in Table 7.2.

The ACS is typically very robust in respect of parameter values for the E-n22-k4 instance as the algorithm was able to find the optimal solution regardless of the values assigned to the parameters. The quality of solutions returned by the ACS with respect to α, however, typically improved for smaller values of the parameter in the larger instances. The general trend of decreasing solution quality with respect to the objective function value returned as a function of increasing values of α is clearly illustrated in Figures 7.1(a)–7.1(c). The aforementioned trend is further corroborated in Table 7.2 with the value of 2 for the parameter α returning the best solution in two of the three instances. The value of 2 was, accordingly, chosen as the value for the parameter α in the remainder of this chapter.

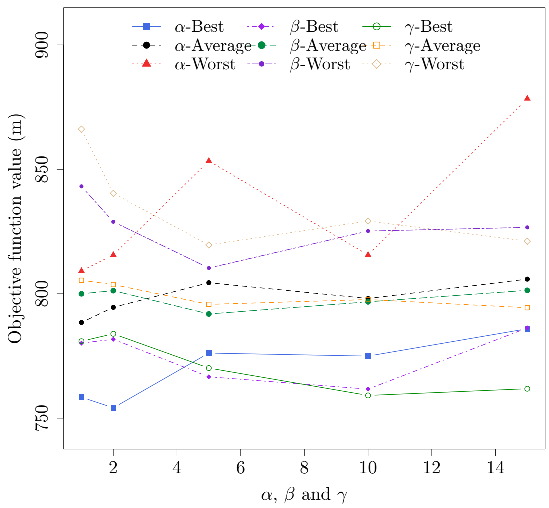
The effect of the parameter β proved to be slightly less consistent with respect to the solution quality returned by the ACS. The small instance again provided little insight in respect of the effect of this parameter, although the larger instances exhibited slightly conflicting trends. The solution quality for the instance E-n51-k5 typically performed better for smaller values of β, while larger values of this parameter returned solutions of a higher quality with respect to the E-n76-k8 instance, as highlighted in Figures 7.1(a)–7.1(c). A value of 5 for the parameter β was, however, decided upon as it returned the best result for the medium sized instance while still performing relatively well in respect of the larger instance, as highlighted in Table 7.2.



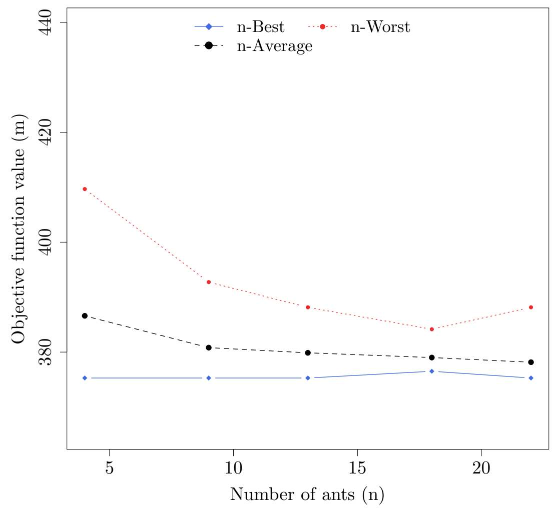
(a) *Sensitivity of α, β and γ in respect of E-n22-k4*



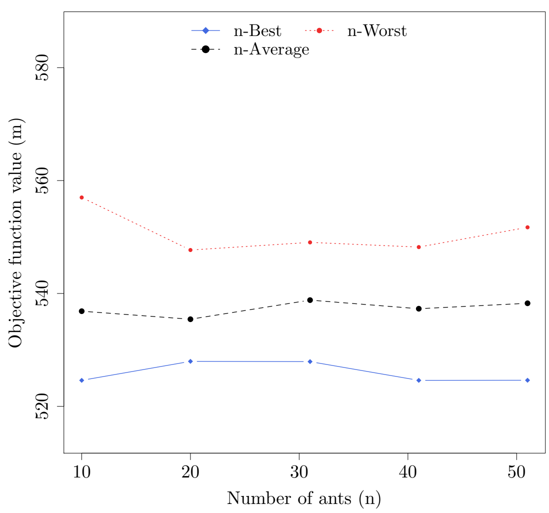
(b) *Sensitivity of α, β and γ in respect of E-n51-k5*



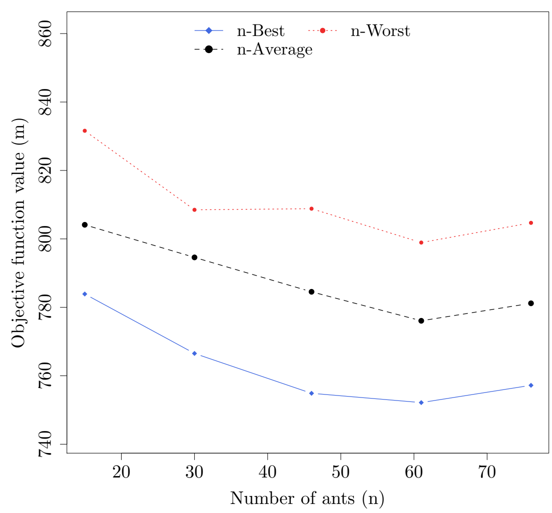
(c) *Sensitivity of α, β and γ in respect of E-n76-k8*



(d) *Sensitivity of the number of ants in respect of E-n22-k4*



(e) *Sensitivity of the number of ants in respect of E-n51-k5*



(f) *Sensitivity of the number of ants in respect of E-n76-k8*

FIGURE 7.1: *Sensitivity analysis of the parameters α, β, γ and the number of ants in the ACS.*

TABLE 7.2: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the parameters α, β and γ (measured as percentages).*

	n22			n51			n76		
	α	β	γ	α	β	γ	α	β	γ
1	0.001	0.003	0.001	0.372	0.155	0.578	3.186	6.144	6.248
2	0.001	0.001	0.001	0.117	0.574	1.053	2.606	6.356	6.654
5	0.001	0.001	0.001	1.570	0.117	0.180	5.599	4.301	4.780
10	0.003	0.001	0.003	3.780	0.215	0.558	5.434	3.630	3.283
15	0.001	0.001	0.001	2.946	0.579	0.117	6.929	6.974	3.650

Similarly, the parameter γ did not exhibit any affinity towards smaller or larger values. The general inconsistent effect of the parameter γ on solution quality is illustrated in Figures 7.1(a)–7.1(c), although Table 7.2 shows that a value of 15 for the parameter performs the best over the three test instances. Accordingly, it was decided to adopt a value of 15 for the parameter γ in the remainder of this chapter.

The number of ants employed determine how many agents traverse the graph during each iteration, building routes concurrently. In the ACS employed in this dissertation, the ants act independently during their route construction vertex selection, but they all share the same pheromone and heuristic function information. The general consensus in the literature is that more ants typically result in better solutions, although more ants dilute the effects of the global pheromone update component and also increase the computation effort [159]. The effects of the number of ants on the solution quality are illustrated in Figures 7.1(d)–7.1(f).

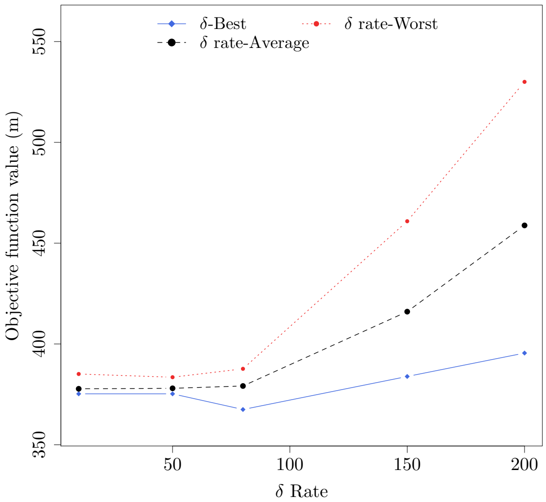
The deviation of the objective function values of solutions returned by the ant colony algorithm from those of the known optimal solutions are shown in Table 7.3 for different numbers of ants, where the rows correspond to the number of ants employed relative to the number of customers, N. The decision to select the factor 0.8 as the parameter setting was reached because the average solution quality improves as an increasing function of the number of ants employed and the ratio of 0.8 also yielded the best solution for all three of the test instances.

TABLE 7.3: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the parameter number of ants (measured as percentages).*

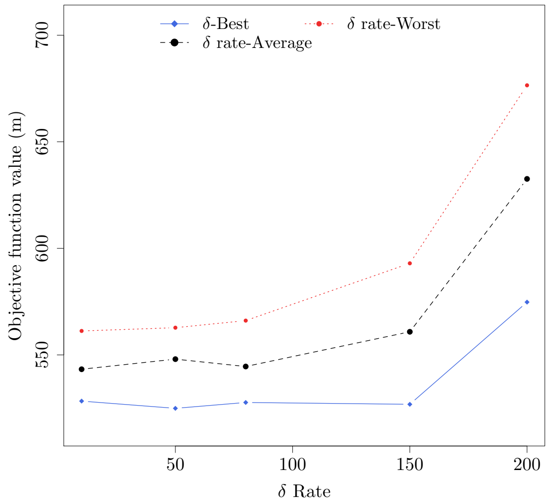
	n22	n51	n76
0.2N	0.001	0.693	6.652
0.4N	0.001	1.339	4.289
0.6N	0.001	1.330	2.704
0.8N	0.001	0.693	2.335
N	0.001	0.696	3.022

The pheromone update mechanism is arguably the most important aspect of any ACS, as it guides the algorithmic search through the solution space. The variable δ determines the weighting of the local pheromone update component, elucidated in the expression (6.6), with a larger value indicative of larger deposits of pheromone along the respective arcs. This affects the exploration aspect of the algorithm.

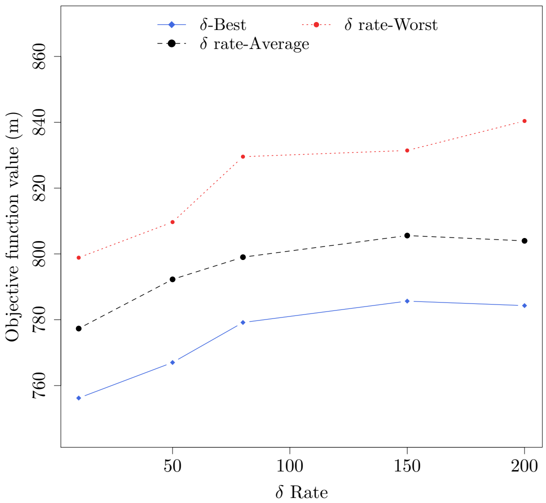
Figures 7.2(a)–7.2(c) contain a summary of the effects on the objective function value obtained when altering the values of parameter δ. A clear trend is visible, with larger values of the parameter δ returning solutions of a lower quality with respect to all three instances. This trend is further corroborated in Table 7.4 in which the deviation of the best reported solution for each



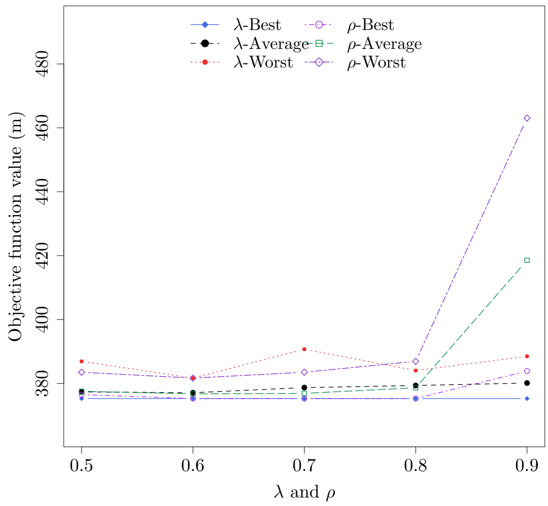
(a) *Sensitivity of δ in respect of E-n22-k4*



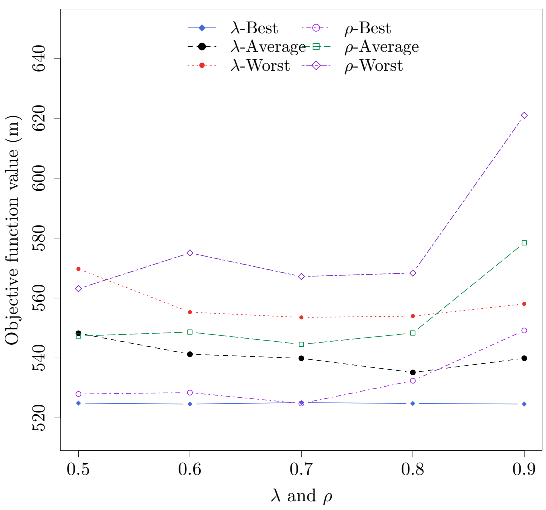
(b) *Sensitivity of δ in respect of E-n51-k5*



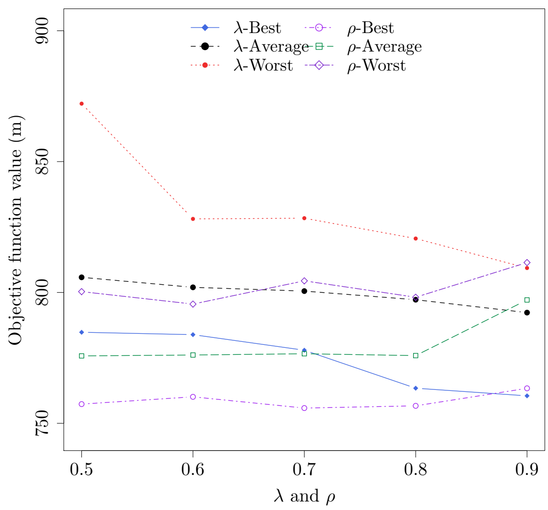
(c) *Sensitivity of δ in respect of E-n76-k8*



(d) *Sensitivity of λ and ρ in respect of E-n22-k4*



(e) *Sensitivity of λ and ρ in respect of E-n51-k5*



(f) *Sensitivity of λ and ρ in respect of E-n76-k8*

FIGURE 7.2: *Sensitivity analysis of the parameters λ, ρ and δ in the ACS.*

parameter configuration from the known optimal is reported. A value of 50 was selected for the parameter δ in the remainder of the chapter, as it returned the best solution for two out of the three test instances and the second best solution in the third test instance.

TABLE 7.4: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the parameter δ (measured as percentages).*

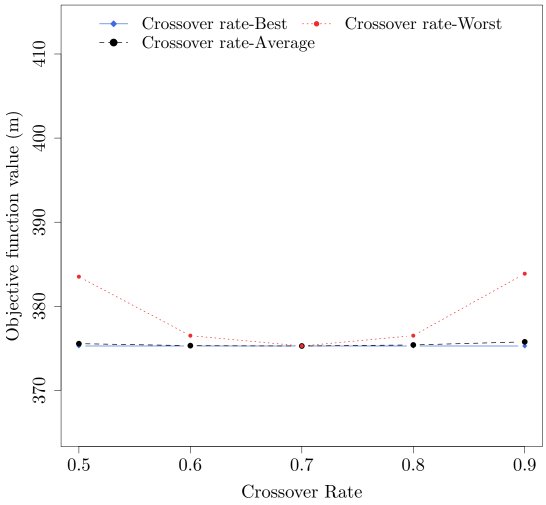
δ	n22	n51	n76
10	0.001	0.829	2.886
50	0.001	0.180	4.355
80	2.089	0.706	6.009
150	2.296	0.542	6.892
200	9.457	9.690	6.707

The functioning of the pheromone update mechanism is highly dependent on the values assigned to the parameters λ and ρ. The parameter λ determines the approach an ant will take in determining the next vertex to visit, according to the probabilistic selection rule described in §6.2.1 and the method of selecting the best feasible vertex (the two candidate strategies). Figures 7.2(d)–7.2(f) contain graphical presentations of the objective function values obtained when employing the different parameter configurations in respect of the three instances. The figures show a slight trend in that the larger values assigned to the parameter λ consistently return solutions of a higher quality with respect to the three test instances. The parameter ρ also affects the pheromone update mechanism, as shown in the expression (6.6), with the parameter determining the rate of evaporation of the pheromone trails deposited by the ants along the arcs. Figures 7.2(d)–7.2(f) exhibit a clear trend in the objective function value returned by the ACS when altering the values assigned to the parameter ρ with respect to the three instances, in that lower values yielded solutions of a higher quality.

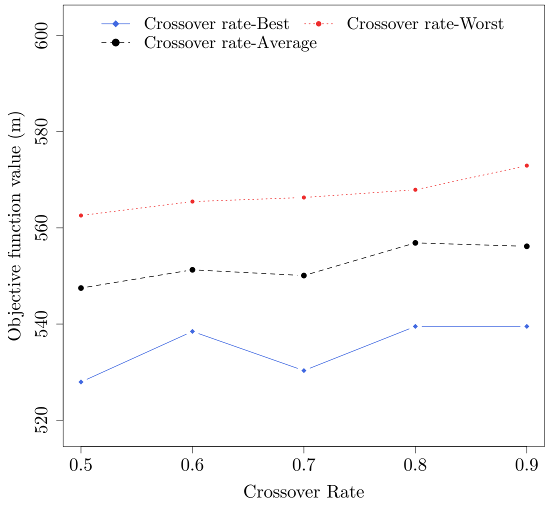
A summary of the deviations in objective function values relative to the objective function values of optimal solutions obtained when varying the parameters λ and ρ may be found in Table 7.5. The table corroborates the trends visible in Figures 7.2(d)–7.2(f) for the parameter λ in that the largest value, 0.9, assigned to the parameter λ returned the best solution in all three of the test instances. Thus a decision was taken to assign the parameter λ a value of 0.9 in the remainder of this chapter. The results presented in Table 7.5 for the parameter ρ also further corroborate the finding that smaller values of the parameter ρ yielded solutions of a higher quality. Accordingly, a value of 0.7 was decided upon for the parameter ρ as it performed consistently over the three test instances and returned the best solution for all three of the instances.

TABLE 7.5: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the parameters λ and ρ (measured as percentages).*

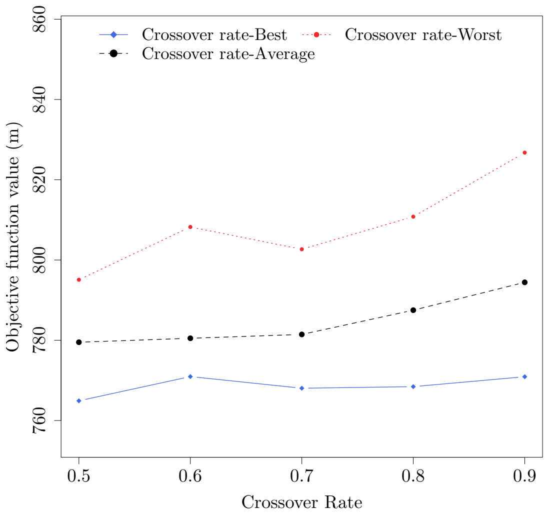
	n22		n51		n76	
	λ	ρ	λ	ρ	λ	ρ
0.5	0.001	0.002	0.180	0.759	6.774	3.039
0.6	0.001	0.001	0.117	0.847	6.650	3.417
0.7	0.001	0.001	0.215	0.155	5.841	2.833
0.8	0.001	0.001	0.155	1.605	3.866	2.949
0.9	0.001	0.031	0.120	4.804	3.468	3.860



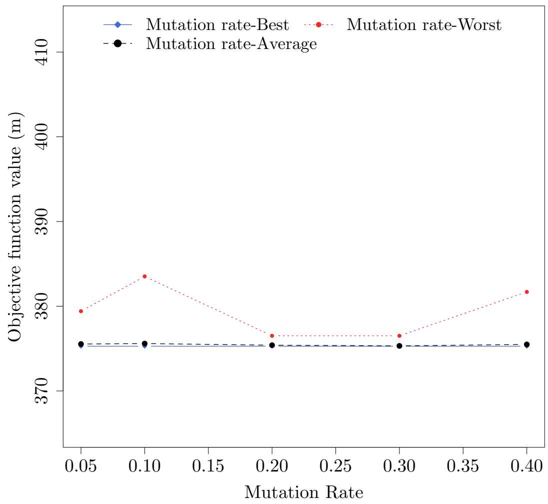
(a) *Sensitivity of the crossover rate in respect of E-n22-k4*



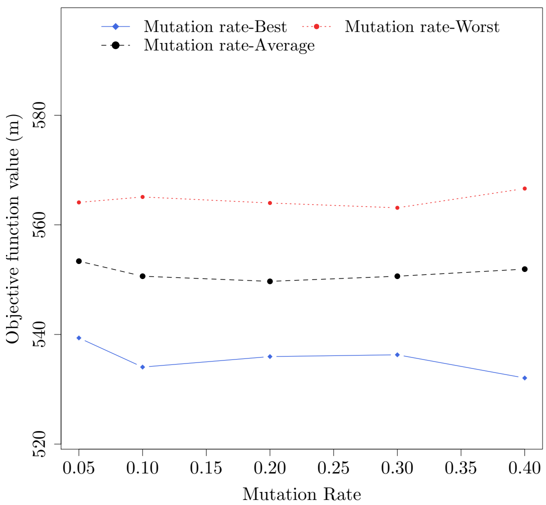
(b) *Sensitivity of the crossover rate in respect of E-n51-k5*



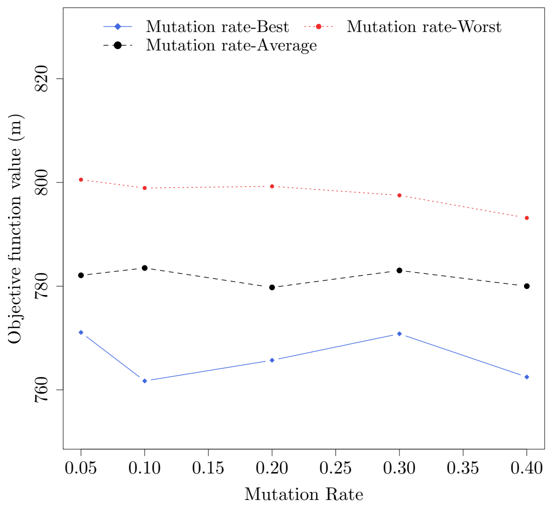
(c) *Sensitivity of the crossover rate in respect of E-n76-k8*



(d) *Sensitivity of the mutation rate in respect of E-n22-k4*



(e) *Sensitivity of the mutation rate in respect of E-n51-k5*



(f) *Sensitivity of the mutation rate in respect of E-n76-k8*

FIGURE 7.3: *Sensitivity analysis of the crossover and mutation rates in the GA.*

7.4 Parameter sensitivity of the genetic algorithm

The parameter sensitivity analysis performed on the test instance E-n22-k4 yet again offered little insight as the GA proved to be robust with respect to parameter values assigned to its operators. The algorithm was able to discover the optimal solution in over 90% of the tests performed and thus little deviation was experienced with respect to solution quality. Accordingly, the decision on the values assigned to parameters was heavily biased towards results returned for the larger test instances which offered more definitive conclusions.

The crossover rate determines the proportion of the population selected to undergo the reproduction phase. A high-quality parameter value for the crossover rate is essential in any effective implementation of a GA. This value should achieve a balance between the exploitation and exploration aspects of the metaheuristic. Modern GAs rely heavily on effective crossover operators, although interestingly the original GA proposed by Holland [195] did not incorporate a crossover operator at all, instead relying solely on a mutation operator and a probabilistic selection operator. The effect of the crossover rate on the quality of the solution returned by the GA of Chapter 6 is illustrated in Figures 7.3(a)–7.3(c) for the CVRP test instances of §7.1.

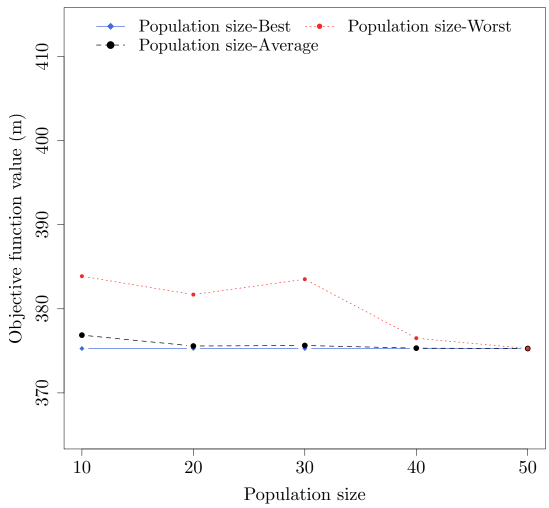
The effect of the crossover rate on solution quality was surprising, as the smallest value assigned to the crossover rate yielded solutions of the highest quality in respect of the three test instances. A sensitivity analysis with respect to altering the values of the crossover rate is shown in Table 7.6. A crossover rate value of 0.5 was decided upon for the remainder of this chapter as it returned the optimal solution for the small test instance and performed the best in the remaining two test instances. The general trend for the value assigned to the crossover rate is that a larger value typically resulted in results of a poorer quality returned by the algorithm, which is perhaps counter-intuitive as this limits the solution diversity and places a larger focus on exploitation within the algorithm.

TABLE 7.6: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the crossover rate (measured as percentages).*

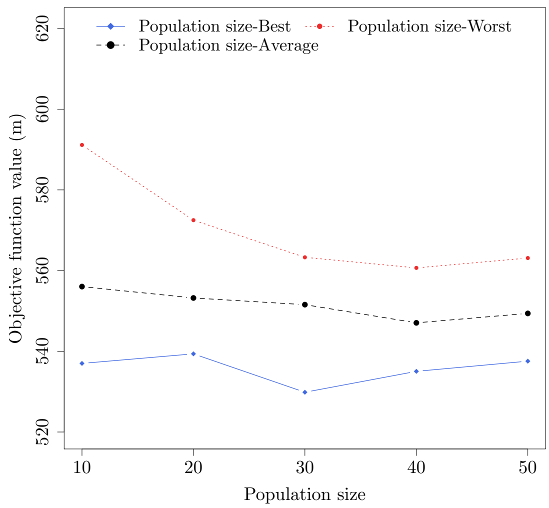
Crossover Rate	n21	n51	n76
0.5	0.001	1.333	4.070
0.6	0.001	3.352	4.892
0.7	0.001	1.788	4.496
0.8	0.001	3.553	4.550
0.9	0.001	3.553	4.887

The mutation rate is also a key parameter in most GA implementations. The mutation rate determines to a large extent the level of diversity within the population. A population exhibiting high diversity levels during early generations is more likely to obtain higher-quality solutions during later generations. The effects of varying the value of the mutation rate in respect of the objective function values obtained by the GA is depicted graphically in Figures 7.3(d)–7.3(f).

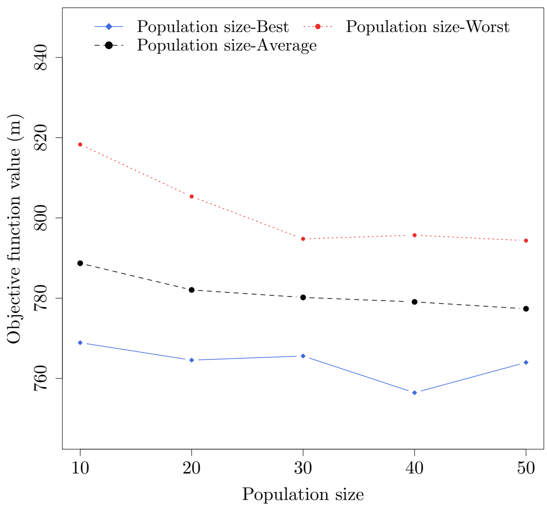
The mutation rate values also affected the solution quality in perhaps a surprising manner, with the largest parameter setting returning the best solution quality with respect to the three test instances. The results of varying the mutation rate value in terms of the subsequent optimality gap are shown in Table 7.7. A value of 0.4 was decided upon for the mutation rate in the remainder of this chapter. The superior performance of the GA with respect to high-valued mutation rates may be attributed to the algorithm relying heavily on diversity-management through the application of mutation operators as opposed to crossover operators. This bias may be attributed to the crossover operator's inherent weakness in retaining characteristics and information of the



(a) *Sensitivity of the population size in respect of E-n22-k4*



(b) *Sensitivity of the population size in respect of E-n51-k5*



(c) *Sensitivity of the population size in respect of E-n76-k8*

FIGURE 7.4: *Sensitivity analysis in respect of the GA population size.*

previous population in generating a new population, because crossover operators are typically rather disruptive as opposed to the more minor changes experienced in characteristics of the population when a mutation operator is applied. Additionally, incorporation of a swap mutation operator is highly beneficial. The application of the *2-opt* mechanism may have contributed to the bias towards applying large amounts of mutation within a population.

TABLE 7.7: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the mutation rate (measured as percentages).*

Mutation Rate	n22	n51	n76
0.05	0.001	3.526	4.909
0.1	0.001	2.505	3.635
0.2	0.001	2.872	4.178
0.3	0.001	2.935	4.873
0.4	0.001	2.122	3.740

The diversity within a population is related to the population size. A larger population tends to achieve greater diversity, although a population size that is too large can negatively affect the convergence of the algorithm towards good solutions. The effect of the population size on the solution quality is shown in Figure 7.4.

The objective function values obtained by the GA when varying the population size were expected — larger values of this parameter typically returned higher quality solutions in two out of the three test instances. The optimality gaps achieved with respect to the best-known objective function values are summarised in Table 7.8 as a function of population size. Thus a decision was taken to fix the population size at forty in the remainder of this chapter as this value resulted in consistently good solutions for the three test instances.

TABLE 7.8: *Deviation in the objective function value from that of the known optimal solution (i.e. the optimality gap) as a result of varying the population size (measured as percentages).*

Population size	n21	n51	n76
10	0.001	3.071	4.613
20	0.001	3.526	4.022
30	0.001	1.699	4.162
40	0.001	2.693	2.917
50	0.001	3.178	3.943

7.5 Incumbent numerical results

The algorithmic parameter values of the two CVRP solution approaches proposed in Chapter 6, as finalised in the previous two sections, were subsequently implemented. Each algorithm was allowed a budget of 1 000 iterations of no improvement as termination criterion. The optimal vehicle routes for the three CVRP test instances were obtained from the Capacitated Vehicle Routing Library [113]. In addition, each algorithm was applied thirty times in respect of each of the test instances of §7.1, adopting the parameter values described in §7.3 and §7.4. The results are summarised in Figure 7.5. The results of the aforementioned implementations were

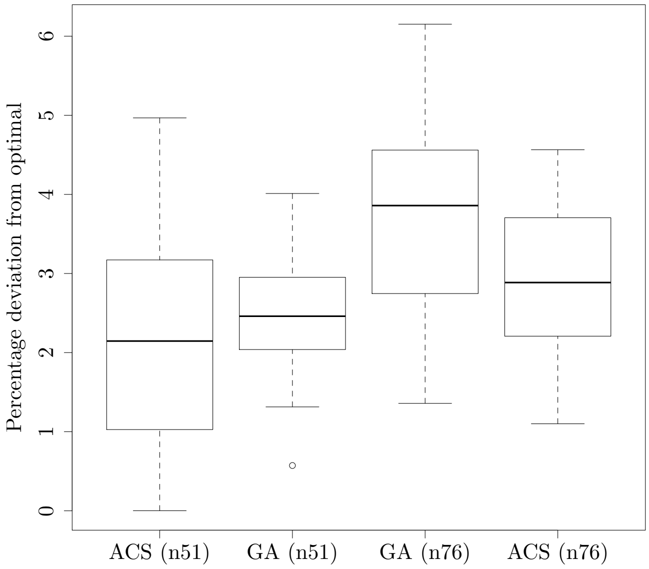


FIGURE 7.5: *Deviation in the objective function values returned by the algorithms from those of the known optimal solutions to the medium and large CVRP benchmark instances of §7.1 (measured as percentages).*

compared by means of an ANOVA test[1], which produces an F-statistic that may be employed to calculate the respective p-value[2], in order to determine statistically whether there are significant differences in the results yielded by the competing algorithms with respect to the E-n51-k5 and E-n76-k8 test instances at a 95% level of confidence. The E-n22-k4 instance was removed form consideration as both the GA and the ACS were able to reach optimality in all thirty test runs. An ANOVA test applied to the results returned by the ACS and GA with respect to the E-n51-k5 instance yielded a p-value of 0.29, statistically confirming that the respective means are indistinguishable, as elucidated in Figure 7.5. An ANOVA test similarly applied to the E-n76-k8 instance proved that there was not a statistically significant difference between the results produced by the two algorithms, returning a p-value of 0.454, as highlighted in Figure 7.5.

The routes were drawn out to illustrate the contrast in solution quality obtained by the competing algorithms. The CPLEX implementation (with a computational budget of 10 hours) and both approximate approaches were able to return the optimal solution to the E-n22-k4 instance, as shown in Figure 7.6(a). The CPLEX implementation was still executing the branch and bound algorithm when the time-out occurred. The returned solution was, in fact, optimal, but the branch and bound algorithm was still reducing the gap between the best lower bound and the upper bound represented by the solution obtained in order to prove its optimality when the computational budget expired.

[1]Fischer [166] developed the ANOVA test which incorporates both the sum of squares between sets of data and the sum of squares within sets of data to calculate whether there are significant differences in the data set means. If the group means are drawn from populations with the same mean values, the variance between the group means should be lower than the variance of the samples, following the central limit theorem. A higher ratio therefore implies that the samples were drawn from populations with different mean values.

[2]The p-value is the level of marginal significance within a statistical hypothesis test representing the probability wrongly rejecting the null hypothesis when, in fact, it is true. A smaller p-value means that there is stronger evidence in favour of the alternative hypothesis. If the p-value is smaller than v, there is a statistical difference between at least one of the means at a $(1 - v)$-level of confidence.

Table 7.9: *Deviation from the objective function value of the known optimal solution (i.e. the optimality gap) as a result of varying the solution approach (measured as percentages).*

	n21	n51	n76
ACS	0.001	0.001	1.101
GA	0.001	0.573	1.358
CPLEX	0.001	28.678	34.715

The CVRP routes for the E-n51-k5 test instance are shown in Figures 7.6(b)–7.6(c). The ACS was able to reach optimality, with the corresponding vehicle routes shown in Figure 7.6(b). The vehicle routes returned by the GA and those returned by CPLEX (with a computational budget of 10 hours) are similarly shown in Figures 7.6(c) and 7.6(d), respectively. The ACS performed the best as it was able to reach optimality. The GA, however, only managed to obtain a solution within 0.573% of the optimal solution, while the CPLEX implementation was only able to reach a solution within 28.6% of the optimal solution when the 10 hour cut-off time was reached.

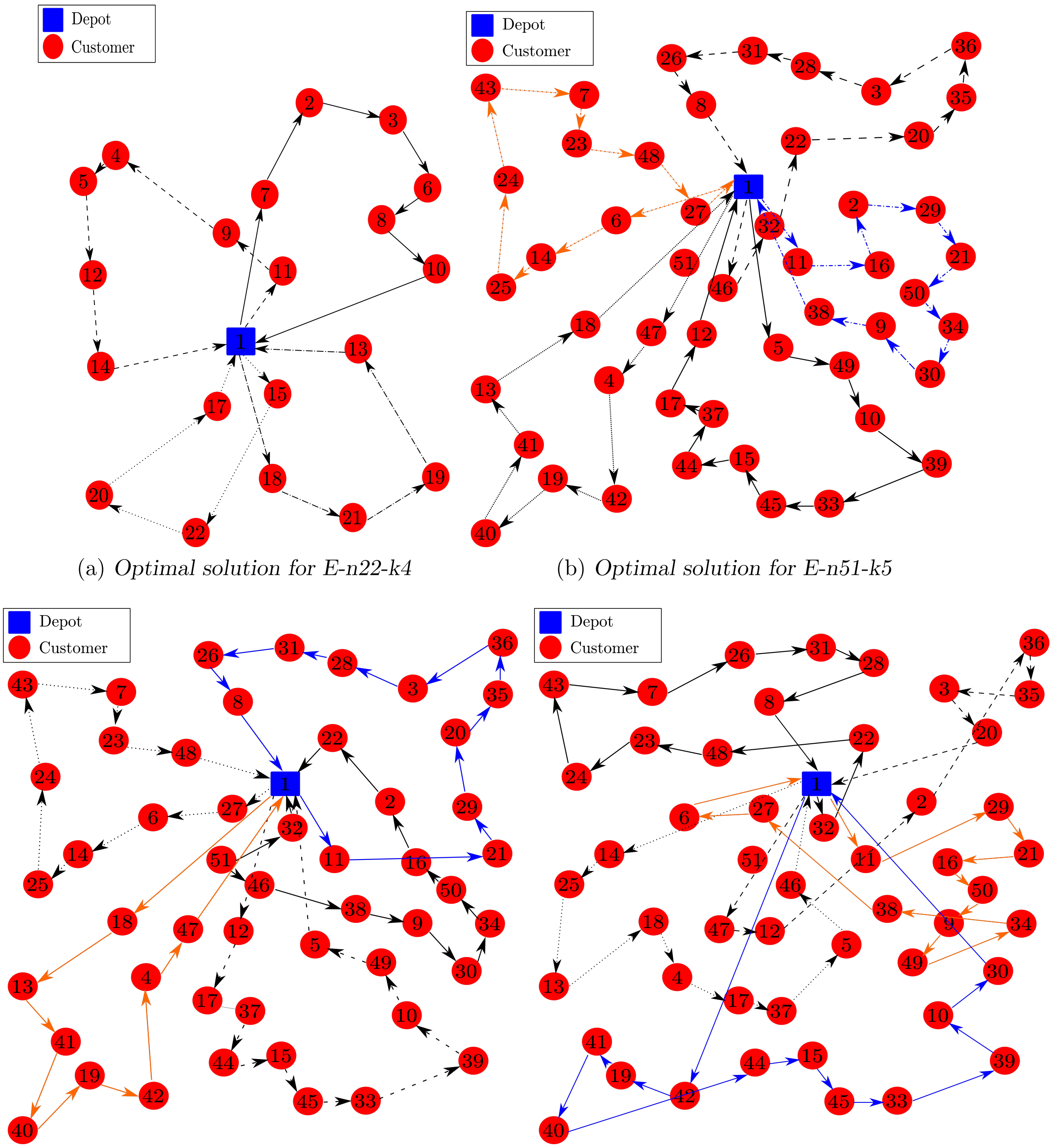
The vehicle routes for the E-n76-k8 instance are shown in Figure 7.7. The optimal solution is shown in Figure 7.7(a), while the incumbent solution obtained by the ACS is shown in Figure 7.7(b). The vehicle routes returned by the GA are shown in Figure 7.7(c), while those returned by CPLEX (with a computational budget of 10 hours) are shown in Figure 7.7(d). The GA outperformed the other solution approaches, yielding a solution within 1.101% of the optimal solution, while the ACS and CPLEX implementations were able to obtain solutions within 1.358% and 34.715% of the optimal solution, respectively. The results returned by the algorithms, with respect to the three CVRP instances of §7.1 are summarised in Table 7.9.

7.6 Discussion

The ACS and the GA performed similarly in the context of the CVRP test instances of §7.1 as illustrated in Figure 7.5, although the associated computation times were not recorded. The ACS nevertheless required notably less time to reach solutions of a high quality, but tended to become stuck at local optima for long periods of time. The GA typically required more computation time as the solution quality improved incrementally, but was less susceptible to becoming trapped at local optima. The results obtained by the GA are very comparable with the results in the published literature [42, 323, 443], although the associated computational time required is considerably longer than those reported in the literature. This discrepancy may be attributed to implementing the algorithm in the high-level RStudio environment — possible improvement may be sought by programming in a better suited, low-level environment. The inherent lack of information retention from previous iterations due to the disruptive nature of crossover operators may also be the reason for the slow convergence rate and large computation times of the GA.

The solutions quality yielded by ACS are, in fact, of a higher quality than those returned by the algorithm proposed by Reed *et al.* [363]. In their paper, Reed *et al.* tested their proposed ACO in respect of several test instances, with the E-n51-k5 included in their set of benchmark instances employed. They reported a deviation of 4.1% in respect of the objective function value from the best-known solution. Interestingly, they did not report computation times with respect to the E-n51-k5 instance either and considering that they did not impose a limit on the number of iterations of the *2-opt* algorithm, one may assume that the computation times were rather long, rendering the ACS employed in this dissertation even more competitive.

The performance of both algorithms of Chapter 6 may possibly be improved by incorporating



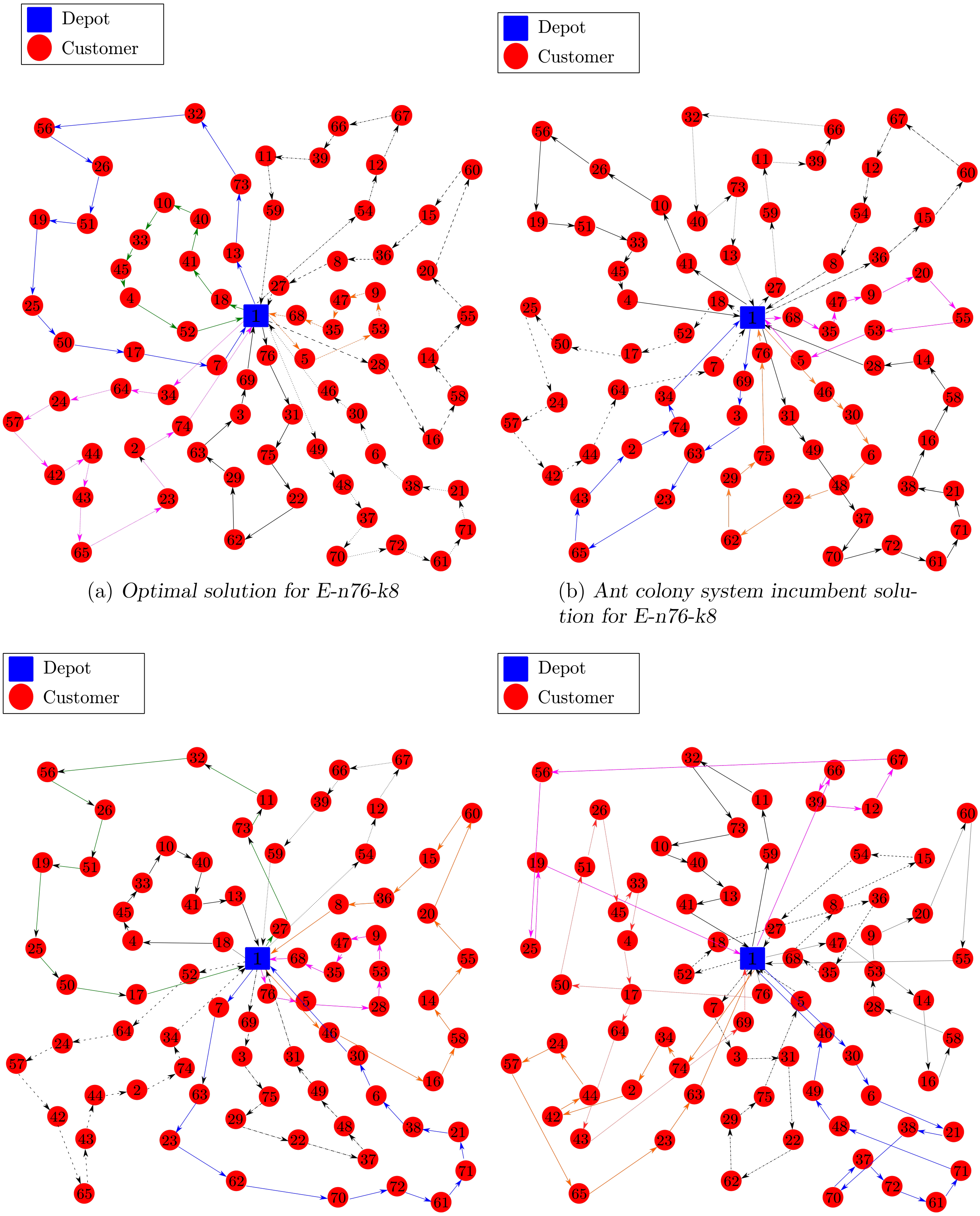
(a) *Optimal solution for E-n22-k4*

(b) *Optimal solution for E-n51-k5*

(c) *Genetic algorithm incumbent solution for E-n51-k5*

(d) *CPLEX incumbent solution for E-n51-k5*

FIGURE 7.6: *Optimal CVRP routes for the E-n22-k4 instance as well as an optimal solution to the E-n51-k5 test instance and incumbent CVRP routes for this test instance returned by the three solution methodologies employed in this dissertation.*



(a) *Optimal solution for E-n76-k8*

(b) *Ant colony system incumbent solution for E-n76-k8*

(c) *Genetic algorithm incumbent solution for E-n76-k8*

(d) *CPLEX incumbent solution for E-n76-k8*

FIGURE 7.7: *An optimal solution to the E-n76-k8 test instance from the literature as well as incumbent CVRP routes for this test instance returned by the three solution methodologies employed in this dissertation.*

constraint handling and penalty functions so as to allow the algorithms to accept infeasible solutions (as the bounds on the capacity constraint are extremely tight in all three CVRP instances considered in this chapter, with a minimum capacity to demand ratio of 0.95).

7.7 Chapter summary

In this chapter, the sensitivity of the parameters employed in the approximate solution approaches of Chapter 6 was analysed in the context of three CVRP benchmark test instances. These test instances were described in §7.1, all three being from the well-known Christofides and Eilon data set [82].

An experimental design undertaken in this chapter was elaborated upon in §7.2, highlighting the methodology adopted to determine a suitable configuration of algorithmic parameter values.

The ACS of §6.2 employs several parameters in its implementation. Accordingly, a parameter sensitivity analysis was performed in §7.3 and a similar approach was adopted in §7.4 to determine a good combination of parameter values for the GA of §6.1.

The results returned by these two algorithms, with the parameter values fixed according to the findings in §7.3 and §7.4, were presented in §7.5 in respect of the three aforementioned CVRP test instances. The results and relative performances of the respective algorithms were discussed in §7.6, and certain potential algorithmic implementation pitfalls, as well as possible improvements, were highlighted in the case of the GA.

CHAPTER 8

CVRP Solution by Clustering

Contents

Clustering has long been implemented as a phase in VRP solution approaches, from the humble beginnings of the route-first cluster-second classical heuristic approaches of the 1960s [39] to more recent metaheuristics utilising clustering components [134]. This chapter contains a brief review of the different distance measures used within clustering algorithms to determine the dissimilarity matrix in §8.1. The different clustering methods employed in the context of the CVRP are discussed in §8.2 and the different indices and criteria used to determine the number of clusters are presented in §8.3. The partitioning of the clusters is determined empirically by implementing several combinations of indices that return suggestions as to the number of clusters coupled with various distance measures that generate a large pool of suggestions in respect of the number of clusters. Several clustering algorithms are implemented and the best resulting cluster is selected according to the numerous internal and stability measures presented in §8.4. The clusters generated are then incorporated as the initial phase of a CVRP problem to partition the customers into subproblems, with the results presented in §8.5. The desirability of incorporating a clustering phase in the CVRP solution procedure is compared with respect to solution quality and computation time in §8.6 The chapter closes in §8.7 with a brief summary of the work contained within.

8.1 The distance measures employed in this dissertation

There are several distance measures in the literature for calculating the dissimilarity matrices of the clustering algorithms described in Chapter 4. The Euclidean distance, the maximum distance, the Manhattan distance, the Canberra distance and the Minkowski distance measures are incorporated in the clustering algorithms employed in this dissertation. These distance measures are reviewed briefly in this section in terms of two vectors $\boldsymbol{x} = [x_1, \ldots, x_d]$ and $\boldsymbol{y} = [y_1, \ldots, y_d]$.

The Euclidean distance is the straight-line distance between two points in Euclidean space and is given by

$$d_E(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{j=1}^{d} (x_j - y_j)^2 \right)^{\frac{1}{2}}. \tag{8.1}$$

This distance measure is the most commonly adopted in the literature. The next distance measure is referred to as the maximum distance measure and is the supremum norm between the components of $\boldsymbol{x}$ and $\boldsymbol{y}$. This distance measure is given by

$$d_m(\boldsymbol{x}, \boldsymbol{y}) = \sup_{1 \leq j \leq d} |x_j - y_j|, \tag{8.2}$$

which, in layman terms, is the maximum distance between two components of $\boldsymbol{x}$ and $\boldsymbol{y}$. The Manhattan distance is the distance between two points in a strictly grid-based plane, and is the sum of the horizontal and vertical distance components along a shortest path in the grid, as shown in Figure 8.1, with the red, blue and yellow lines representing different paths of the same Manhattan distance. The green line is representative of the Euclidean distance.

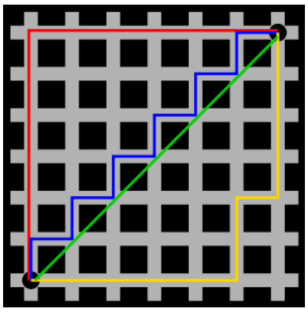


FIGURE 8.1: *Illustration of the Manhattan distance calculation.*

The Manhattan distance may be expressed as

$$d_M(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{d} |x_j - y_j|. \tag{8.3}$$

The Canberra distance measure was introduced by Lance and William [270], and was later improved upon by the same authors in [271]. The Canberra distance measure

$$d_C(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{d} \frac{|x_j - y_j|}{|x_j| + |y_j|} \tag{8.4}$$

is a weighted version of the Manhattan distance measure. The final distance measure considered in this dissertation is the Minkowski distance measure, introduced by Kruskal [264]. It is popular due to it generalising many other distance measures, such as the Euclidean and Manhattan distance measures. The Minkowski distance of order p between two points $\boldsymbol{x}$ and $\boldsymbol{y}$ is given by

$$d(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{j=1}^{d} |x_j - y_j|^p \right)^{\frac{1}{p}}, \tag{8.5}$$

for $p \geq 1$, since $p < 1$ violates the triangle inequality. If $p = 1$, the Manhattan distance measure is obtained as a special case of the Minkowski distance measure. Similarly, if $p = 2$, then the Euclidean distance measure is obtained as a special case of the Minkowski distance measure.

8.2 The clustering methods employed in this dissertation

The number of possible ways of assigning C distinct customers to n_c clusters in such a manner that each cluster has at least one customer, is given by

$$\frac{1}{n_c!}\sum_{i=1}^{n_c}(-1)\binom{n_c}{i}(n_c - i)^C, \tag{8.6}$$

which results in 18 490 198 597 75 082 317 124 304 (over 18 octillion) ways to partition a mere 40 customers into 6 clusters [63]. The super-exponential growth of the quantity in (8.6) as a function of increasing values of C severely limits the size of VRP problems that can be handled by exact clustering algorithms. Several authors have therefore proposed clustering methods aimed at clustering the customer sets of larger instances heuristically so as to obtain smaller, independent subproblems.

The first clustering method employed in this dissertation within the context of the CVRP is referred to as the *Unweighted Pair Group Method with Arithmetic Mean* (UPGMA) in the literature and is perhaps the most frequently used clustering algorithm [254]. The method was introduced by Sokal [407] and is an agglomerative, hierarchical clustering algorithm that produces a dendogram which can be cut at a desired height in order to produce the required number of clusters.

Divisive clustering methods are considered exotic as agglomerative approaches seem to be the most popular due to their favourable computational complexity. The *Divisive Analysis Clustering Algorithm* (DIANA) has, however, gained some popularity among researchers [254]. The algorithm is based on the work of Macnaughton-Smith *et al.* [305] and was developed by Kaufman and Rousseeuw [254]. All data points are placed in a single cluster and the algorithm iteratively partitions clusters based on inner dissimilarity. This procedure is repeated until all the data points are placed in singleton clusters.

Partitioning Around Medoids (PAM) is another greedy clustering algorithm [253] developed by Kaufman and Rousseeuw [254]. The algorithm initially selects n_c points to act as the cluster medoids, and then each remaining point is associated with its nearest medoid. While the cost of the configuration decreases, the points that are acting as medoids are swapped out with non-medoid points and the cost configuration is recalculated. This process is repeated until convergence occurs.

Clustering Large Applications (CLARA) is a sampling-based algorithm which implements PAM on a number of subsets [254]. This approach leads to improved performance on large datasets. Compared to PAM, CLARA can deal with much larger data sets. It also attempts to find n_c representative points that are centrally located in the various clusters. This is achieved by considering only a sample of data points and then performing the PAM algorithm to determine the medoids. Thereafter, each point not considered in the sample is assigned to the nearest medoid.

The algorithm *Fuzzy Analysis* (FANNY) was also developed by Kaufman and Rousseeuw [254], and implements fuzzy clustering. Fuzzy clustering is a generalisation of partitioning, which normally assigns a data point to one cluster only, while fuzzy clustering offers some ambiguity with respect to cluster membership. The algorithm only works for $n_c < \frac{o}{2}$ clusters, where o is the number of data points. The algorithm aims to minimise an objective function by determining the number of clusters. The objective function is a weighted function of the dissimilarity matrix and membership coefficients.

Self-Organising Maps (SOM) is an unsupervised machine learning technique that is popular among computational biologists and machine learning researchers. It was introduced by Kohonen *et al.* [268], is based on neural networks and is highly regarded for its ability to map and visualise high-dimensional data in two dimensions [59]. The goal of learning in the SOM algorithm is to cause the different parts of the network to respond similarly to certain input parameters.

Fraley and Raftery [170] developed a statistical model for performing cluster evaluation. The model consists of fitting a finite mixture of Guassian distributions to the data. The mixture components represent clusters, and cluster membership is estimated by means of a maximum likelihood approach.

The *Self Organising Tree Algorithm* (SOTA) was first proposed by Dopazo and Carazo [136] for phylogenic reconstruction, but was later applied to cluster microarray gene expression data by Herrero *et al.* [228]. The algorithm is an unsupervised neural network that grows, adopting the topology of a binary tree. The algorithm follows a divisive clustering approach that has a stopping criterion based on the approximate distribution of probability obtained by randomisation of the original data.

The final method employed in this dissertation in respect of CVRP customer clustering is perhaps the most popular, and is referred to as the k-means method in the literature. The term was coined by MacQueen [298] in 1967, but the original idea dates back to the work of Steinhaus [408] in 1956. The general notion of the k-means clustering algorithm is to select n_c points to act as initial cluster centres. Each data point is then associated with the nearest centre in order to create temporary clusters. A new gravity centre is calculated and each observation is reassigned to the nearest centre. This process is repeated until convergence is achieved. The difference between PAM and k-means clustering is that PAM calculates the medoids by minimising the absolute distance between the points and the selected centroid, as opposed to the k-means method which minimises the squared distance. As a result, the PAM algorithm is more robust to noise and outliers [254].

8.3 Indices for the number of clusters

Determining the number of clusters n_c for a given data set is a problem in its own right and is distinct from the process of actually solving the clustering problem. The determination of n_c is often ambiguous, with interpretations of a suitable number of clusters depending on the shape and scale of the distribution of points in the data set and the desired clustering resolution of the analyst. Increasing the value of n_c without an associated penalty will always reduce the error in the resulting clustering, to the extreme case where every data point is in its own cluster. If the number of clusters is not specified *a priori*, it must therefore be determined in some manner. This section contains brief descriptions of all the indices employed in the clustering algorithms of this dissertation to determine a desirable number of clusters.

The first such method was developed by Krzanowski and Lai [265] who proposed an argument for deriving a criterion for use in conjunction with the within-group sum-of-squares objective function to determine the optimal number of clusters. The method was validated using Monte Carlo simulations and provided promising results, as it was able to identify the correct number of clusters 85% of the time for grouped data.

Caliński and Harabasz [69] suggested a variance ratio criterion for determining the optimal number of clusters, which gives some insight into the structure of points. Hartigan [223] similarly proposed a statistic that incorporates a weighted ratio of within-cluster sum of squares and suggested that if the ratio remains above 10, another cluster should be added.

Sarle [386] proposed a cubic clustering criterion based on minimising the within-cluster sum of squares and evaluated its performance using Monte Carlo methods. The criterion performed poorly in respect of clusters that were highly elongated or irregularly shaped.

Scott and Symons [393] derived a maximum likelihood estimate for determining the number of clusters required, but the estimate requires the assumption that the underlying distributions are multivariate Gaussian. The estimate is then used to minimise the total within-group sum of squares and to maximise the between-group sum of squares.

Marriott [301] was able to resolve the interpretation and computation of minimising the determinant of the within-group dispersion matrix to determine the natural number of clusters for a data set.

Milligan and Cooper [312] analysed thirty stopping criteria by means of Monte Carlo simulation to determine external criteria that performed best. They used these criteria to develop an index that measures the trace of the within-clusters-pooled-covariance matrix.

Friedman and Rubin [172] introduced a function defined for all partitions of the data points into n_c clusters, and selected a partition for which the measure is maximal. This function is based on a pooled within-groups scatter matrix and a between-group scatter matrix.

Davies and Bouldin [109] formulated a general cluster separation measure which allows for the determination of the average similarity of each cluster with its most similar cluster. Incorporating this measure into a stopping criterion, the number of clusters may be determined by means of a criterion based on the sum ratio of within-cluster scatter to between-cluster separation.

Rousseeuw [379] developed the *silhouette index* which is based on a comparison of the tightness and separation of a clustering with the average silhouette width utilised in a stopping criterion to determine the optimal number of clusters.

Duda and Hart [149] proposed a ratio of the squared error for a two-cluster solution over that for a one-cluster solution. The process of cluster division stops if the ratio is too large. Otherwise the partition process continues. Two cluster numbers are returned by this method, the one being the largest value of n_c that remains below a critical value and the other being the smallest value of n_c that remains above a critical value.

Beale [37] proposed the use of an F-test ratio to test the hypothesis of the improved performance of C_2 *versus* C_1 clusters, where $C_2 > C_1$. In this case the F-test compares the increase in the mean square deviation from cluster centroids as one transitions from C_1 clusters to C_2 clusters. Clustering continues until the hypothesis that C_2 clusters is of a higher quality than C_1 clusters, is rejected.

Ratkowsky and Lance [362] proposed a criterion for determining the optimal number of clusters based on a ratio of the sum of squares distance between and within groups, divided by the number of clusters. The optimal number of clusters is the value that returns the maximum ratio.

Ball and Hall [30] argued that the average distance between the points and their respective cluster centres could serve as an informative measure for determining the number of clusters, with the largest difference between levels taken as the optimal solution.

Kraemer [262] developed the *Ptbiserial index*, which is simply a point-biserial correlation between the input dissimilarity matrix and a corresponding matrix consisting of 0 or 1 entries.

Tibshirani *et al.* [423] proposed using a gap statistic to determine the value of n_c based on the change in within-cluster dispersion with respect to the expected value under an appropriate reference null distribution.

Frey and Van Groenewald [171] introduced a ratio to be used as a general clustering stopping criterion. The ratio compares the difference between the average between-cluster distance from each of two hierarchy levels with the difference between the mean within-cluster distances from the two levels. The hierarchy level returning a value closest to 1 is selected.

Mclain and Rao [304] developed the program CLUSTISZ which employs a criterion consisting of a ratio of two terms. The first term measures the average within-cluster distance divided by the number of items within the cluster. The second term is the average between-cluster distance divided by the number of points within the cluster, with the minimum value returning the best value for n_c.

Baker and Hubert [24] introduced the so-called *gamma index* for determining a suitable number of clusters. This ratio is a normalised ratio of the number of consistent comparisons involving between-cluster and within-cluster distances and the number of inconsistent outcomes. The maximum value is taken to be indicative of the correct hierarchy level.

Rohlf [377] introduced the *G(+) index* which compares the number of inconsistent outcomes (the same number as in the *gamma* index), but it also considers the number of within-cluster distances. The minimum value of the index indicates the best number of clusters.

Dunn [153] defined a ratio between the minimal inter-cluster distance and maximal intra-cluster distance, with the largest value being indicative of a good number of clusters.

Halkidi *et al.* [216] introduced the *SD index* which is based on the concepts of average scattering for clusters and total separation between clusters. The average scattering is related to variance within each cluster and the total separation between clusters depends on the maximum Euclidean distance between the cluster centres, with the smallest value of the index being indicative of a good n_c-value.

Lebart *et al.* [279] introduced the *D-index* which is based on clustering gain in respect of intra-cluster inertia. The intra-cluster inertia measures the degree of homogeneity between the data associated with a cluster. It calculates their distances to a reference point typifying the profile of the cluster (generally the cluster centroid), with the aim of minimising the clustering gain.

Finally, Halkidi and Vazirgiannis [215] introduced the *SDbw index* which is based on the criteria of compactness and separation between clusters. The index consists of two terms, namely a scatter term and a density term. The scatter term aims to measure the variance prevalent in clusters while the density term aims to evaluate the inter-cluster density, by quantifying the average density in the region among clusters in relation to the density of the clusters themselves, with the aim of minimising the index.

8.4 Cluster validity measures employed in this dissertation

One of the most integral aspects in cluster analysis is the *post hoc* evaluation of clustering results so as to determine which partition best fits the underlying data [214]. In general terms, there are three approaches toward investigating cluster validity [420]. The first approach involves the use of external criteria, where the performance of the clustering is based on a pre-specified structure as a result of intuition of the data. The second involves the use of internal criteria, where the clustering performance is based on quantities that are functions of the vectors in the data set.

Finally, the third approach is based on relative criteria, where the clustering performance is compared with other clusterings formed by competing algorithms.

Three internal validation measures are employed in this chapter within the context of CVRP customer clustering. These measures are aimed at measuring the connectedness, compactness and separation of the cluster partitions. Connectedness relates to the extent to which data points are placed in the same cluster as their nearest neighbour, and is measured by means of a connectivity index of Handl *et al.* [218]. Let $\mathcal{O}$ represent the set of data points, and let nn_{ij} be the j-th nearest neighbour of data point $i \in \mathcal{O}$. Furthermore, define the parameter

$$m_{i,nn_{ij}} = \begin{cases} 0, & \text{if data points } i,j \in \mathcal{O} \text{ are grouped in the same cluster,} \\ \frac{1}{j}, & \text{otherwise.} \end{cases}$$

The connectivity for a cluster partitioning of $\mathcal{O}$ into n_c disjoint clusters may be formulated as

$$\text{Connectivity} = \sum_{i=1}^{|\mathcal{O}|} \sum_{j=1}^{L} m_{i,nn_{ij}}, \tag{8.7}$$

where L represents a parameter determining the number of nearest neighbours to use. The connectivity has a non-negative real value and should be minimised.

Another internal validation measure is referred to as the *silhouette width* in the literature. This width is defined as the average of each data point's *silhouette* value. This *silhouette* value measures the degree of confidence associated with assigning a data point to a cluster, with a value of 1 indicating a high level of confidence and a value of -1 representing a low level of confidence. The silhouette value for data point $i \in \mathcal{O}$ may be determined as

$$S(i) = \frac{b_i - a_i}{\max\{b_i, a_i\}}, \tag{8.8}$$

where a_i is the average distance between data point $i \in \mathcal{O}$ and all other data points in the same cluster, and b_i is the average distance between data point $i \in \mathcal{O}$ and all the data points in the nearest neighbour cluster. The *silhouette* value is in the range $[-1, 1]$ and should be maximised.

The final internal validity measure is the *Dunn index*, which is the ratio of the smallest distance between data points not in the same cluster to the largest intra-cluster distance. The ratio has a non-negative real value and should be maximised.

The next class of validity measures is referred to as *stability measures*, which compare clusterings in respect of the full data with clusterings based on removing each data feature column, one at a time. These measures work especially well if the data are highly correlated [59]. Four stability measures are employed in this chapter within the context of CVRP customer clustering. The first such measure is the *average proportion of non-overlap* (APN). The APN measures the average proportion of data points not placed in the same cluster by clustering based on the full data and clustering based on the data with a single column removed. Let C^i denote the cluster containing data point i in the original clustering, and let $C^{i,\ell}$ denote the cluster containing data point $i \in \mathcal{O}$ where the clustering is based on the data set with feature column ℓ removed. The APN may be calculated as

$$\text{APN}(n_c) = \frac{1}{M|\mathcal{O}|} \sum_{i=1}^{|\mathcal{O}|} \sum_{\ell=1}^{M} \left(1 - \frac{|C^{i,\ell} \cap C^i|}{|C^i|}\right), \tag{8.9}$$

where n_c denotes the total number of clusters and M is the number of features or columns present in the data. The APN is a real number in the interval $[0, 1]$, with values close to zero corresponding to highly consistent clustering results.

Another stability measure is referred to as the *average distance between means* (ADM), which computes the average distance between cluster centres for data points placed in the same cluster by clustering based on the full set of data *versus* clusters based on data with a single column removed. The ADM is given by

$$\text{ADM}(n_c) = \frac{1}{M|\mathcal{O}|}\sum_{i=1}^{|\mathcal{O}|}\sum_{\ell=1}^{M}\text{dist}(\bar{\boldsymbol{x}}_{C^{i,\ell}}, \bar{\boldsymbol{x}}_{C^i}), \tag{8.10}$$

where $\bar{\boldsymbol{x}}_{C^i}$ is the centroid of data points in the cluster which contains observation $i \in \mathcal{O}$ and $\bar{\boldsymbol{x}}_C^{i,\ell}$ denotes the centroid of the cluster that contains data point i with column ℓ removed. The ADM is a non-negative real value and smaller values are preferable.

The *average distance* (AD) is the third stability measure employed in this chapter, which is the average distance between data points placed in the same cluster resulting from clustering based on the full data *versus* clustering based on the data with a single column removed. The AD is given by

$$\text{AD}(n_c) = \frac{1}{M|\mathcal{O}|}\sum_{i=1}^{|\mathcal{O}|}\sum_{\ell=1}^{M}\frac{1}{|C^i|\,|C^{i,\ell}|}\left[\sum_{i\in C^i,\ j\in C^{i,\ell}}\text{dist}(i,j)\right], \tag{8.11}$$

where dist(i,j) denotes a distance measure between data points i and j. The ADM is a non-negative real value with a smaller value indicative of a more stable clustering.

The final stability validity measure employed in this chapter is the *figure of merit* (FOM), proposed by Datta and Datta [108], which measures the intra-cluster variance of the data points in the deleted column, where the clustering is based on the remaining samples (non-deleted columns). This measure estimates the mean error using predictions based on the cluster averages, and is determined by

$$\text{FOM}(\ell{,}n_c) = \left(\frac{1}{|\mathcal{O}|}\sum_{j=1}^{n_c}\sum_{i\in C_{j(\ell)}}\text{dist}(\boldsymbol{x}_{i,\ell}, \bar{\boldsymbol{x}}_{C_{j(\ell)}})\right)^{\frac{1}{2}}. \tag{8.12}$$

The FOM is multiplied by an adjustment factor $\sqrt{\frac{|\mathcal{O}|}{|\mathcal{O}|-n_c}}$ in order to reduce its tendency to decrease as the number of clusters increases, with the final value averaged over all the columns removed. The FOM is a non-negative real value, with smaller values indicative of better clustering performance.

8.5 Incorporating a clustering phase when solving the CVRP

The algorithms employed in this chapter to determine the number of clusters for CVRP customer sets were taken from the library developed by Charrad *et al.* [77] and the cluster validation process was implemented using the library developed by Brock *et al.* [59].

Twenty three indices for the number of clusters were described in §8.3. These indices depend on the distance measure adopted to populate the dissimilarity matrix. Five distance measures were described in §8.1. Combining these five distance measures with the twenty three cluster number indices yields a set of 115 suggestions as to the number of clusters to implement. The three cluster numbers returned with the highest frequency among these 115 suggestions were selected for analysis.

The results for the E-n22-k4 CVRP benchmark instance are shown in Table 8.1, with $n_c = 2$ being the most frequent number of clusters returned by the indices, followed by $n_c = 4$ and thirdly $n_c = 3$. The unmistakable best number of clusters to implement is $n_c = 2$, with nearly 50% of the indices returning this suggested number of clusters. The frequency results in terms

Table 8.1: *Frequency table of best number of clusters to implement for customer clustering of the E-n22-k4 CVRP benchmark instance.*

No. of clusters	1	2	3	4
Frequency	16	57	19	23

of the suggested number of clusters to implement for the E-n51-k5 benchmark are similarly shown in Table 8.2. The most frequently suggested number of clusters was $n_c = 5$, followed closely by $n_c = 2$ and $n_c = 3$, respectively. There is no obvious decision as to the number of clusters to implement as the results are rather evenly spread among the potential candidates, with a mere 3.5% percent separating the top three suggestions.

Table 8.2: *Frequency table of best number of clusters to implement for customer clustering of the E-n51-k5 CVRP benchmark instance.*

No. of clusters	1	2	3	4	5
Frequency	16	27	25	18	29

The frequency results in terms of the number of clusters suggested for the E-n76-k8 instance are finally shown in Table 8.3, with the most frequently suggested numbers of clusters being $n_c = 4$ and $n_c = 8$, and with $n_c = 2$ being the third most frequently suggested number of clusters. Once again there is no clear decision as to the best number of clusters to implement as the results are relatively evenly spread over the top four suggestions, including the value $n_c = 1$.

Table 8.3: *Frequency table of best number of clusters to implement for customer clustering of the E-n76-k8 CVRP benchmark instance.*

No. of clusters	1	2	3	4	5	6	7	8
Frequency	19	21	11	25	7	4	3	25

The next stage of the analysis involved determining the best clustering algorithm by which to compute clusters. The indicators described in §8.4 were used to determine the best clustering algorithm out of the nine described in §8.2. The results of the internal validity measures highlighted in §8.4 for the E-n22-k4 instance are shown in Table 8.4. The nine clustering algorithms considered in this section all returned the same clustering for the case of $n_c = 2$ clusters, which would logically lead to the conclusion that it is a good cluster configuration for $n_c = 2$.

The stability validity measures for the E-n22-k4 instance are shown in Table 8.5, which indicate that the algorithms' performances differ according to the stability measure employed. The decision to use the k-means algorithm was reached for $n_c = 2$ and $n_c = 4$, and use of the UPGMA algorithm was decided upon for $n_c = 3$. The decision was heavily biased towards the internal validity measures, as for CVRPs dense clusters should typically lead to better results, and because the coordinates do not offer enough information for stability testing as only two dimensions are considered. The best number of clusters according to these measures was $n_c = 2$, which reinforces the conclusions reached based on the number of cluster indices which predicted that $n_c = 2$ would produce the best clustering.

TABLE 8.4: *Internal cluster validity measures for the customer set of the E-n22-k4 CVRP benchmark instance.*

Clustering Method	Validation Measure	Number of clusters 2	4	3
UPGMA	Connectivity	4.939	20.752	12.224
	Dunn	0.394	0.403	0.395
	Silhouette	0.500	0.350	0.414
Kmeans	Connectivity	4.939	21.391	12.863
	Dunn	0.394	0.403	0.380
	Silhouette	0.500	0.352	0.406
DIANA	Connectivity	4.939	21.391	12.863
	Dunn	0.394	0.403	0.380
	Silhouette	0.500	0.352	0.406
FANNY	Connectivity	4.939	23.006	21.200
	Dunn	0.394	0.260	0.135
	Silhouette	0.500	0.330	0.285
SOM	Connectivity	4.939	25.439	13.467
	Dunn	0.394	0.189	0.332
	Silhouette	0.500	0.290	0.363
Model	Connectivity	4.939	22.677	14.000
	Dunn	0.394	0.372	0.173
	Silhouette	0.500	0.290	0.363
SOTA	Connectivity	4.939	28.905	16.563
	Dunn	0.394	0.140	0.198
	Silhouette	0.500	0.204	0.314
PAM	Connectivity	4.939	21.391	13.467
	Dunn	0.394	0.403	0.332
	Silhouette	0.500	0.352	0.363
CLARA	Connectivity	4.939	21.391	15.208
	Dunn	0.394	0.403	0.335
	Silhouette	0.500	0.352	0.359

The internal validity measures for the E-n51-k5 benchmark instance are shown in Table 8.6, with the k-means algorithm performing the most consistently over the different clusterings and performance measures. The SOM clustering algorithm also performed consistently well over the different clusterings, especially for the case of $n_c = 3$ clusters, where it produced the best result for two out of the three measures. These measures were the connectivity measure and the silhouette measure which would indicate a dense clustering of data points and a relatively high certainty associated with the clustering. The FANNY algorithm also performed relatively well for the case of $n_c = 3$ clusters, but for the other numbers of clusters, it performed poorly.

The stability cluster validity measures for the E-n51-k5 benchmark instance are shown in Table 8.7, with the k-means algorithm showing the most promising results once again. The PAM algorithm also consistently performed well over the clusterings. The DIANA algorithm performed the best for the smaller n_c-values, but achieved poor results for $n_c = 5$ clusters. The model-based clustering approach appeared to be the most stable for $n_c = 2$ clusters, as it produced the best result for three out of the nine stability measures, while the DIANA algorithm was the most stable for $n_c = 3$ clusters, producing the best results over all stability measures.

TABLE 8.5: *Stability cluster validity measures for the customer set of the E-n22-k4 CVRP benchmark instance.*

Clustering Method	Validation Measure	Number of clusters 2	4	3
UPGMA	APN	0.174	0.402	0.366
	AD	27.855	24.466	26.391
	ADM	12.116	14.692	16.736
	FOM	19.646	20.245	19.803
Kmeans	APN	0.145	0.362	0.512
	AD	11.901	12.632	18.335
	ADM	11.889	15.623	18.321
	FOM	19.699	20.487	20.137
DIANA	APN	0.231	0.415	0.386
	AD	27.936	24.963	26.923
	ADM	12.362	15.860	17.971
	FOM	19.675	20.210	20.174
FANNY	APN	0.240	0.427	0.438
	AD	27.675	24.920	27.544
	ADM	12.014	15.610	16.4213
	FOM	19.481	20.370	20.137
SOM	APN	0.231	0.404	0.430
	AD	27.936	24.918	27.432
	ADM	12.362	14.183	16.446
	FOM	19.675	20.245	20.137
Model	APN	0.149	0.340	0.449
	AD	27.936	24.539	28.443
	ADM	11.845	12.573	18.229
	FOM	19.699	20.487	20.137
SOTA	APN	0.174	0.402	0.366
	AD	28.023	24.985	26.976
	ADM	12.360	15.029	14.456
	FOM	19.675	20.210	20.174
PAM	APN	0.231	0.423	0.321
	AD	27.936	24.934	26.169
	ADM	12.362	15.697	12.621
	FOM	19.675	20.370	20.137
CLARA	APN	0.198	0.405	0.361
	AD	27.936	24.560	26.745
	ADM	12.276	15.033	14.349
	FOM	19.676	20.235	20.137

TABLE 8.6: *Internal cluster validity measures for the customer set of the E-n51-k5 CVRP benchmark instance.*

		Number of clusters		
Clustering Method	Validation Measure	5	2	3
UPGMA	Connectivity	38.944	16.368	25.826
	Dunn	0.155	0.118	0.123
	Silhouette	0.308	0.325	0.294
Kmeans	Connectivity	37.568	15.466	22.266
	Dunn	0.155	0.104	0.118
	Silhouette	0.318	0.326	0.362
DIANA	Connectivity	41.407	18.978	27.794
	Dunn	0.142	0.099	0.102
	Silhouette	0.282	0.324	0.306
FANNY	Connectivity	36.326	20.911	22.752
	Dunn	0.147	0.099	0.118
	Silhouette	0.247	0.321	0.356
SOM	Connectivity	41.911	17.056	21.394
	Dunn	0.188	0.099	0.118
	Silhouette	0.325	0.328	0.362
Model	Connectivity	39.012	17.043	23.690
	Dunn	0.163	0.093	0.116
	Silhouette	0.380	0.312	0.346
SOTA	Connectivity	47.185	20.911	28.789
	Dunn	0.053	0.099	0.102
	Silhouette	0.271	0.321	0.299
PAM	Connectivity	40.921	15.466	25.950
	Dunn	0.121	0.104	0.041
	Silhouette	0.313	0.326	0.350
CLARA	Connectivity	39.764	16.854	24.611
	Dunn	0.154	0.104	0.041
	Silhouette	0.331	0.320	0.353

In terms of connectivity, it seems that $n_c = 2$ clusters produce a significantly better clustering, but according to the other measures there is no discernible difference in performance of the clusterings, as corroborated by the number of cluster indices. It was decided to use the model-based clustering method for $n_c = 2$ clusters as it produced relatively good results over all three measures, and for $n_c = 5$ it was decided to use the k-means algorithm as it produced significantly better results in terms of connectivity. Finally, it was decided to use SOM for $n_c = 3$ clusters, as it performed relatively well.

The internal cluster validity measures for the E-n76-k8 benchmark instance are shown in Table 8.8. The UPGMA produced the best results in terms of connectivity for all three clusterings, achieving a close grouping of data points, which is advantageous in the context of the CVRP. The UPMGA also performed relatively well in respect of the Dunn index which indicates a good separation level between clusters. The k-means algorithm performed consistently well over all measures, indicating high-quality clusterings. The SOTA performed particularly well for $n_c = 4$ clusters, but produced poor-quality clusters for the other instances. The FANNY algorithm did not produce results for $n_c = 8$ clusters as it was not able to converge after a 1 000 iterations.

TABLE 8.7: *Stability cluster validity measures for the customer set of the E-n51-k5 CVRP benchmark instance.*

		Number of clusters		
Clustering Method	Validation Measure	5	2	3
UPGMA	APN	0.458	0.278	0.411
	AD	24.748	29.411	27.431
	ADM	16.500	12.427	18.782
	FOM	18.519	18.475	18.595
Kmeans	APN	0.526	0.250	0.480
	AD	24.024	29.093	27.355
	ADM	17.125	10.949	17.033
	FOM	17.982	18.398	18.557
DIANA	APN	0.517	0.260	0.339
	AD	25.190	29.093	27.355
	ADM	17.359	11.484	14.448
	FOM	18.772	18.479	18.545
FANNY	APN	0.553	0.334	0.461
	AD	24.847	29.779	27.346
	ADM	17.020	13.916	16.546
	FOM	18.479	18.481	18.562
SOM	APN	0.536	0.261	0.480
	AD	24.442	29.099	27.636
	ADM	16.527	11.375	17.397
	FOM	18.568	18.469	18.581
Model	APN	0.569	0.247	0.472
	AD	24.952	29.021	27.492
	ADM	17.118	10.952	17.033
	FOM	18.459	18.484	18.557
SOTA	APN	0.458	0.278	0.411
	AD	24.587	29.012	27.474
	ADM	17.864	11.037	17.033
	FOM	18.772	18.570	18.545
PAM	APN	0.542	0.259	0.461
	AD	24.525	29.080	27.474
	ADM	16.873	11.418	16.359
	FOM	18.595	18.468	18.568
CLARA	APN	0.551	0.350	0.458
	AD	24.682	29.955	27.465
	ADM	16.592	14.644	16.598
	FOM	18.619	18.468	18.574

TABLE 8.8: *Internal cluster validity measures for the customer set of the E-n76-k8 CVRP benchmark instance.*

		Cluster Size		
Clustering Method	Validation Measure	4	8	2
UPGMA	Connectivity	29.805	53.521	14.523
	Dunn	0.109	0.154	0.093
	Silhouette	0.343	0.313	0.330
Kmeans	Connectivity	30.184	58.121	22.901
	Dunn	0.110	0.163	0.062
	Silhouette	0.377	0.335	0.336
DIANA	Connectivity	40.362	71.451	22.901
	Dunn	0.097	0.126	0.062
	Silhouette	0.361	0.277	0.336
FANNY	Connectivity	34.008	N/A	25.801
	Dunn	0.111	N/A	0.030
	Silhouette	0.373	N/A	0.322
SOM	Connectivity	32.700	65.101	22.262
	Dunn	0.110	0.166	0.045
	Silhouette	0.377	0.314	0.335
Model	Connectivity	40.623	59.748	22.037
	Dunn	0.101	0.149	0.045
	Silhouette	0.367	0.235	0.335
SOTA	Connectivity	31.451	65.926	15.946
	Dunn	0.111	0.140	0.070
	Silhouette	0.396	0.293	0.321
PAM	Connectivity	35.906	60.511	22.525
	Dunn	0.088	0.140	0.067
	Silhouette	0.367	0.326	0.335
CLARA	Connectivity	37.499	60.023	19.503
	Dunn	0.059	0.140	0.075
	Silhouette	0.362	0.327	0.335

The stability cluster validity measures for the customer set of the E-n76-k8 CVRP benchmark instance are finally shown in Table 8.9. The k-means algorithm exhibited the most stability for three out of the four stability measures, but it achieved poor results in respect of the ADM measure, which shows that there is considerable cluster centre shifting between removals of column entries, but that the data points are still generally placed in the same clusters. The CLARA clustering program produced excellent results for $n_c = 8$ clusters, obtaining the best results for three out of the four measures. The UPGMA algorithm exhibited good stability for $n_c = 4$ clusters as it produced the best results for two out of the four measures.

The best cluster size in terms of connectivity is $n_c = 2$, but in terms of the other measures, there is no discernible difference between the different values for n_c, which is in accordance with the predictions by the number of clusters indices. The SOTA was chosen for $n_c = 4$ clusters, while the k-means algorithm was selected for $n_c = 8$ clusters. Finally, the UPGMA algorithm was selected for $n_c = 2$ clusters.

According to the internal measures, the clusters produced for the E-n22-k4 benchmark were of a higher-quality than the clusterings produced for the E-n51-k5 and E-n76-k8 instances. It is

TABLE 8.9: *Stability cluster validity measures for the customer set of the E-n76-k8 CVRP benchmark instance.*

		Cluster Size		
Clustering Method	Validation Measure	4	8	2
UPGMA	APN	0.460	0.591	0.278
	AD	25.620	23.435	29.411
	ADM	17.074	16.324	12.247
	FOM	18.391	18.842	18.475
Kmeans	APN	0.469	0.586	0.249
	AD	24.987	23.288	29.021
	ADM	18.411	18.698	18.484
	FOM	18.557	18.708	18.468
DIANA	APN	0.479	0.593	0.260
	AD	25.601	23.355	29.093
	ADM	17.498	16.671	11.484
	FOM	18.643	18.928	18.479
FANNY	APN	0.526	N/A	0.334
	AD	25.757	N/A	29.779
	ADM	17.523	N/A	13.916
	FOM	18.614	N/A	18.481
SOM	APN	0.514	0.641	0.274
	AD	25.613	23.119	29.205
	ADM	17.420	16.212	11.798
	FOM	18.617	18.884	18.481
Model	APN	0.540	0.647	0.274
	AD	25.914	23.349	29.021
	ADM	17.922	16.192	10.952
	FOM	18.628	18.782	18.484
SOTA	APN	0.479	0.593	0.260
	AD	25.601	23.355	29.093
	ADM	17.498	16.671	11.484
	FOM	18.643	18.928	18.479
PAM	APN	0.500	0.627	0.259
	AD	25.752	23.095	29.082
	ADM	16.966	16.092	11.418
	FOM	18.624	18.877	18.468
CLARA	APN	0.527	0.604	0.350
	AD	25.859	22.915	29.554
	ADM	16.870	15.712	14.644
	FOM	18.557	18.708	18.468

difficult to ascertain whether this is because it is easier for the algorithms to resolve clusters for smaller instances or whether the customers in E-n22-k4 are distributed in such a manner as to accommodate better clustering.

8.6 CVRP solutions upon clustering

The different clusterings that were produced as a result of the study performed in §8.5 are shown in Figures 8.2 and 8.3. These clusterings were treated as inducing individual CVRP instances and the ACS described in §6.2 was used to produce an approximate solution to each individual instance as it is computationally cheaper in respect to CVRP instances. The sub-problems created as a result of the clustering were able to be parallelised and accordingly the ACS could be executed in respect of all the subproblems concurrently. From a purely visual standpoint, the clusterings seem to achieve the aim of segregating the customer sets of the CVRP benchmark instances into more manageable subproblems.

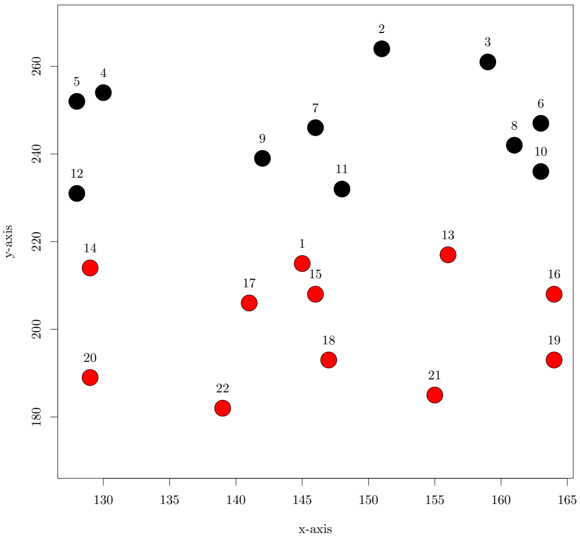
The subproblems for the E-n22-k4 CVRP benchmark instance with respect to the clustering of the customers are shown in Figures 8.2(a)–8.2(c). Similarly, the subproblems for the E-n51-k5 CVRP benchmark instance are shown in Figures 8.2(d)–8.2(f). Finally, the subproblems for the E-n76-k8 CVRP benchmark instance with respect to the clustering of the customers are shown in Figures 8.3(a)–8.3(c).

The different CVRP instances and their respective subproblems were solved using the same hardware and software platforms as mentioned before. An iteration limit of 1 000 was set and the processor time and the objective function deviation from that of the known optimal solution were recorded in each case. The results are shown in Table 8.10.

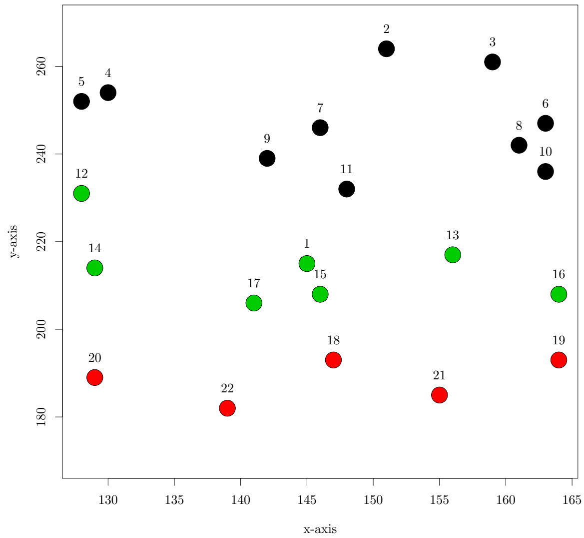
TABLE 8.10: *Comparison of the solution quality (measured as percentage) returned (in respect of the deviation of the objective function returned with and without incorporating a clustering phase) by the ACS and the required computation time (measured in seconds).*

Benchmark	Solution method	Optimality gap (%)	Processing time (s)
E-n22-k4	ACS	0	348
	ACS with $n_c = 2$	5.48	225.61
	ACS with $n_c = 3$	21.72	202.90
	ACS with $n_c = 4$	9.87	41.74
E-n51-k5	ACS	0.57	585.612
	ACS with $n_c = 2$	6.11	555.19
	ACS with $n_c = 3$	5.98	444.32
	ACS with $n_c = 5$	15.58	253.80
E-n76-k8	ACS	1.68	712.82
	ACS with $n_c = 2$	5.21	530.84
	ACS with $n_c = 4$	6.64	476.36
	ACS with $n_c = 8$	19.46	201.90

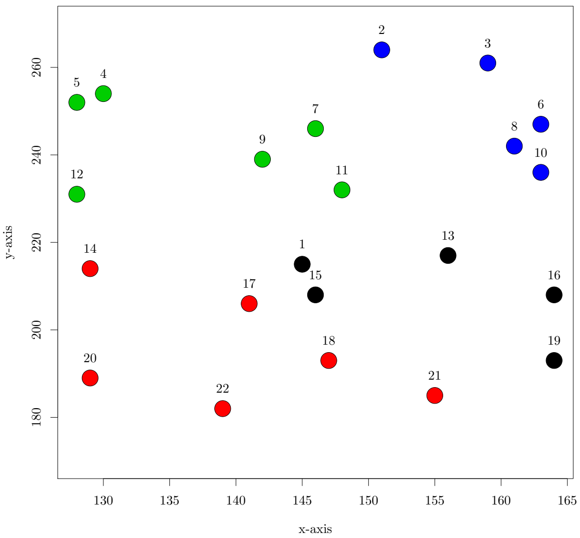
These results exhibit a noticeable decrease in computation time requirements when clustering is applied, but it also shows that the solution quality is negatively affected, as expected. This is due to several factors, including a lack of flexibility of the ACS, and the fact that a solution can be only as good as the clustering allows. Despite the ACS reaching an optimal solution for each subproblem, the solution quality is holistically still relatively poor when customer clustering is applied.



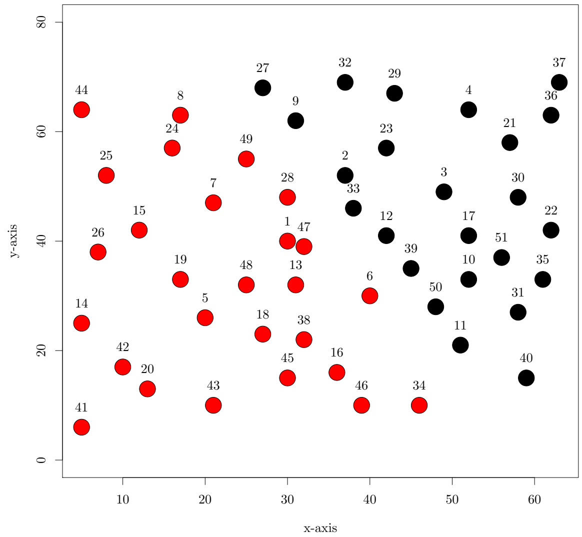
(a) *Clustering of 22 customers into 2 clusters*



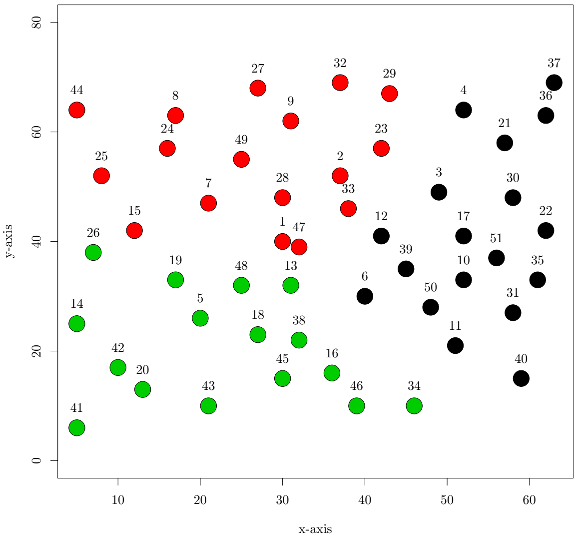
(b) *Clustering of 22 customers into 3 clusters*



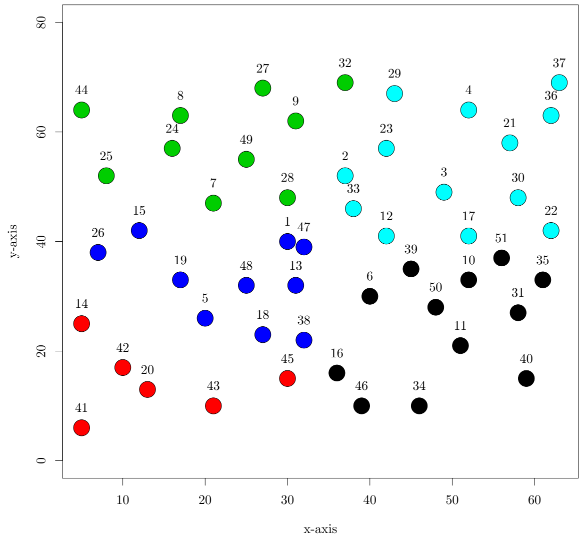
(c) *Clustering of 22 customers into 4 clusters*



(d) *Clustering of 51 customers into 2 clusters*

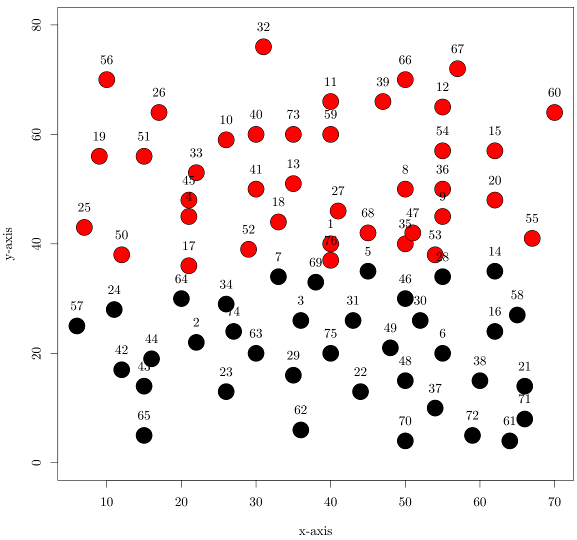


(e) *Clustering of 51 customers into 3 clusters*

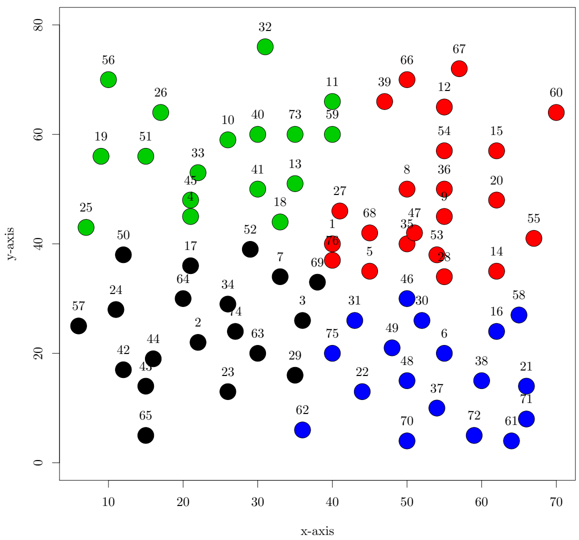


(f) *Clustering of 51 customers into 5 clusters*

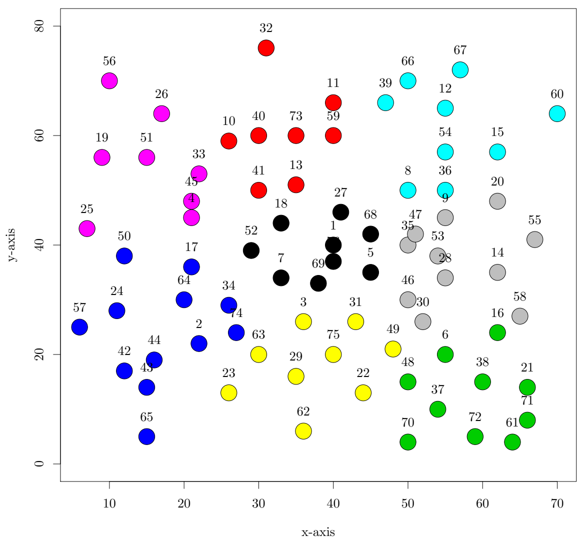
FIGURE 8.2: *Clustering for the customer sets of the E-n22-k4 and E-n51-k5 CVRP benchmark instances.*



(a) *Clustering of 76 customers into 2 clusters*

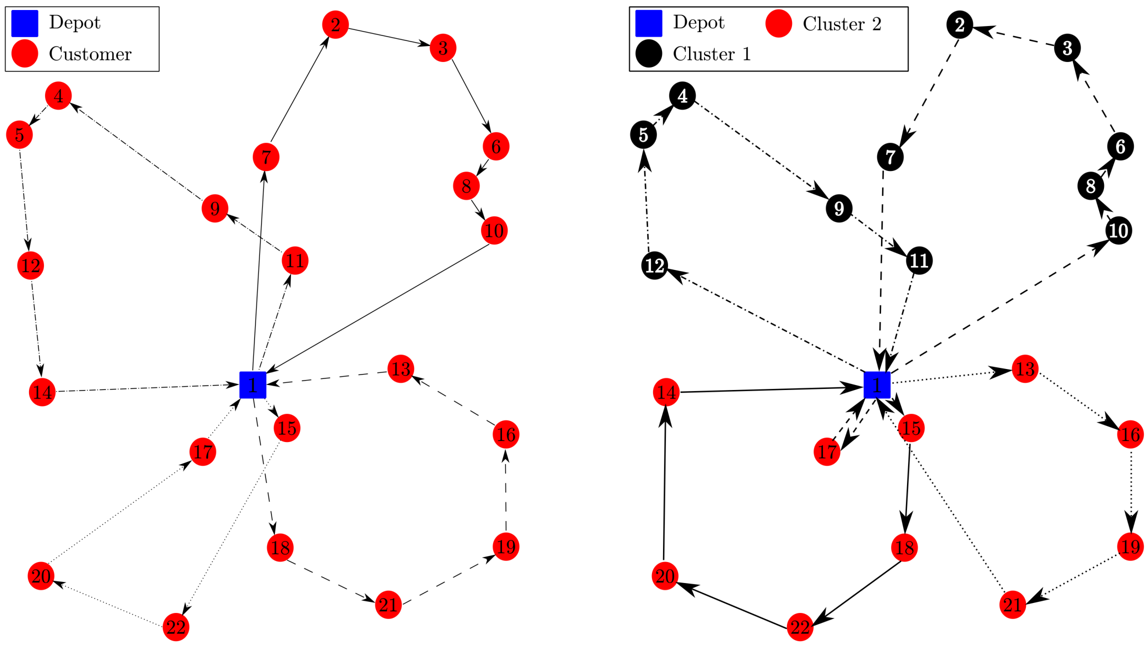


(b) *Clustering of 76 customers into 4 clusters*

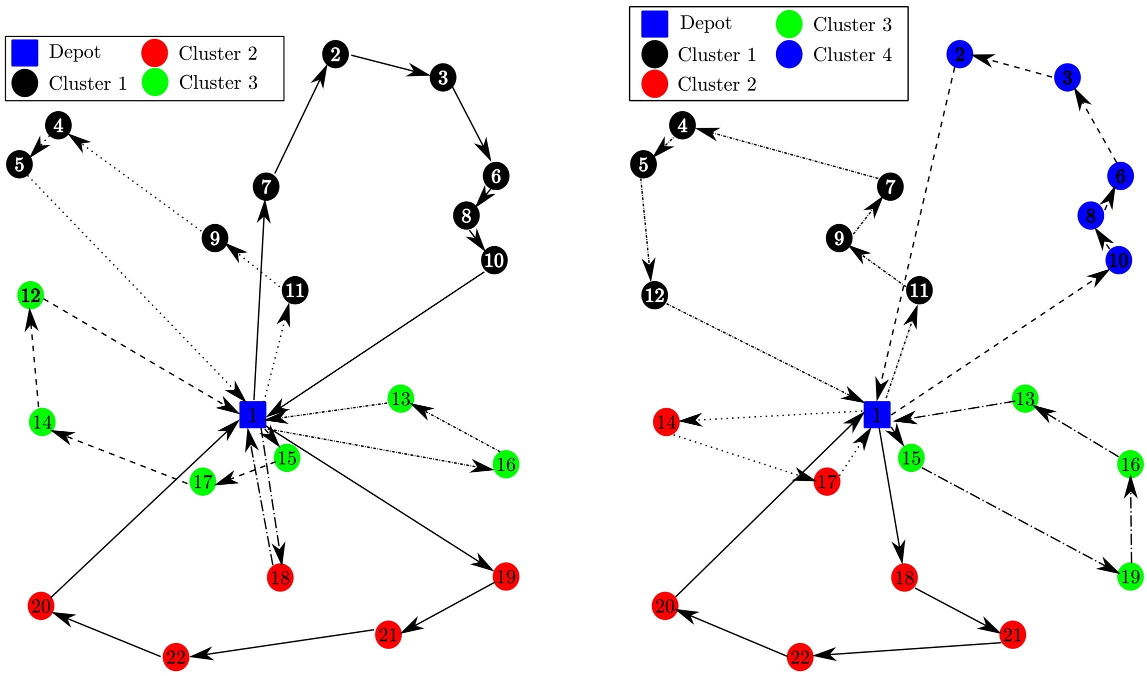


(c) *Clustering of 76 customers into 8 clusters*

FIGURE 8.3: *Clustering for the customer set of the E-n76-k8 CVRP benchmark instance.*

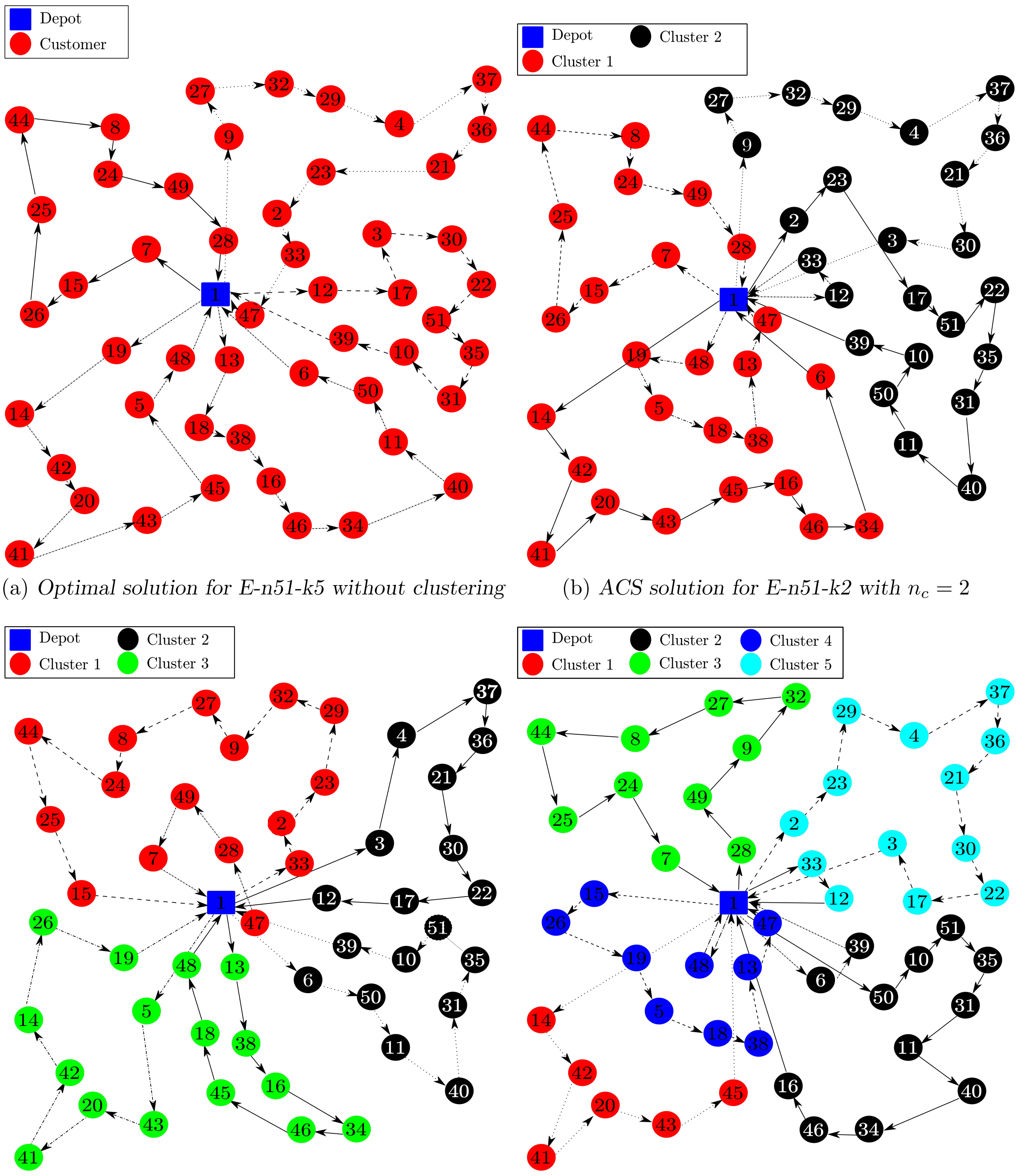


(a) *Optimal solution for E-n22-k4 without clustering* (b) *ACS solution for E-n22-k4 with* $n_c = 2$



(c) *ACS solution for E-n22-k4 with* $n_c = 3$ (d) *ACS solution for E-n22-k4 with* $n_c = 4$

FIGURE 8.4: *Proposed vehicle routes for the E-n22-k4 CVRP benchmark instance.*



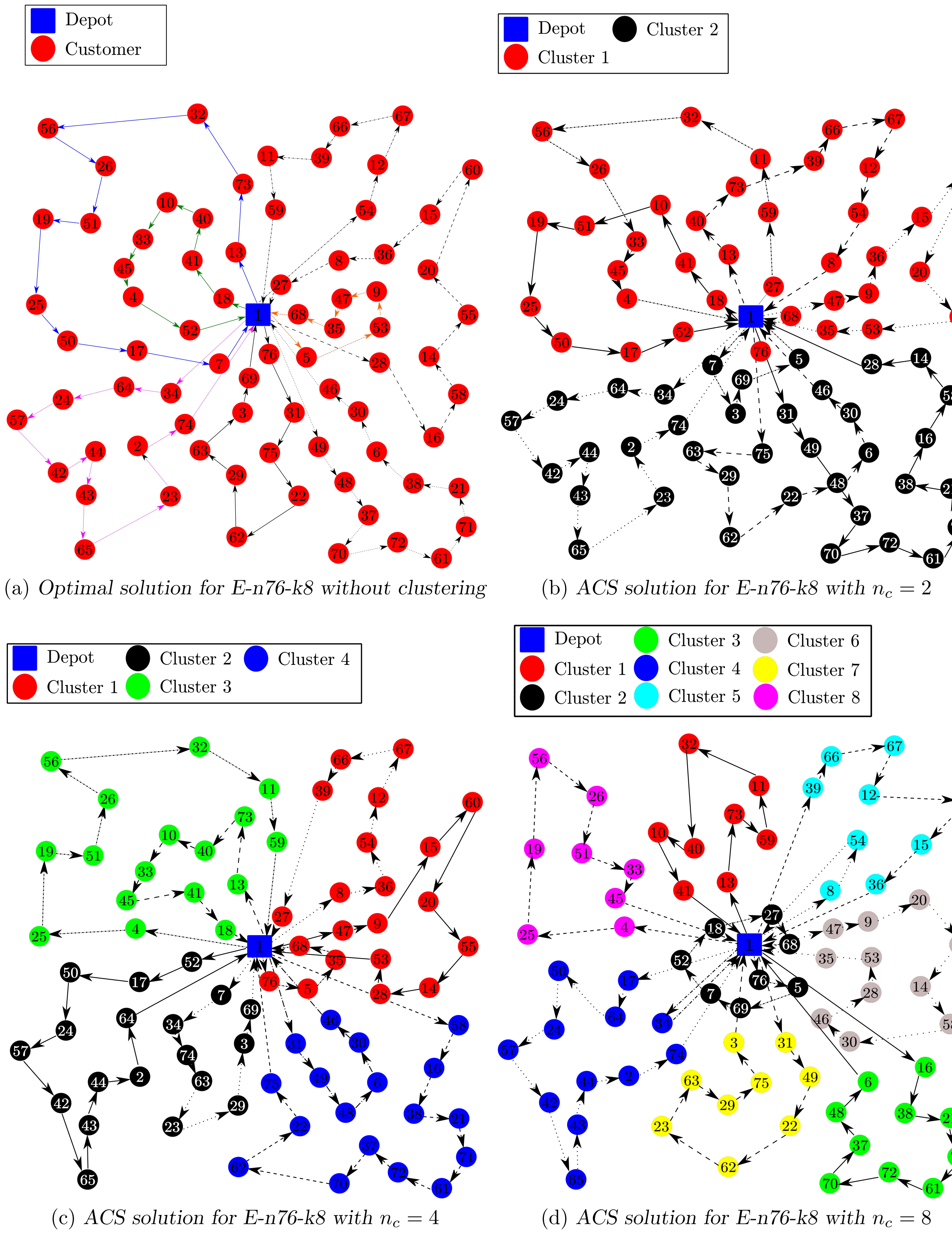
(a) *Optimal solution for E-n51-k5 without clustering*

(b) *ACS solution for E-n51-k2 with* $n_c = 2$

(c) *ACS solution for E-n51-k3 with* $n_c = 3$

(d) *ACS solution for E-n51-k5 with* $n_c = 5$

FIGURE 8.5: *Proposed vehicle routes for the E-n51-k5 CVRP benchmark instance.*



(a) *Optimal solution for E-n76-k8 without clustering*

(b) *ACS solution for E-n76-k8 with* $n_c = 2$

(c) *ACS solution for E-n76-k8 with* $n_c = 4$

(d) *ACS solution for E-n76-k8 with* $n_c = 8$

FIGURE 8.6: *Proposed vehicle routes for the E-n76-k8 CVRP benchmark instance.*

The cluster-first, route-second approach becomes more competitive as the instance size increases and as the cluster sizes themselves increase, thereby allowing for more flexibility within the ACS to reach better solutions. There are, however, methods in the literature, such as the 1-insert method proposed by Salhi and Nagy [383] and the method of micro-clustering proposed by Iochim *et al.* [237], for counteracting the lack of flexibility when applying clustering first.

The vehicle routings for the E-n22-k4 benchmark instance and its respective clustered subproblems, as generated by the ACS, are shown in Figure 8.4. Similarly, the vehicle routings for the E-n51-k5 benchmark instance and its resulting clustered subproblems are shown in Figure 8.5. Finally, the vehicle routings for the E-n76-k8 benchmark instance and its clustered subproblems are shown in Figure 8.6.

The clustering approach presented in this chapter, was designed to be incorporated as an initial phase within the approximate solution approach when considering a pathology service provider's transportation network. The decline in solution quality as a result of a lack of flexibility of the algorithm should not be of too much concern, as real-life instances are expected to contain thousands of customers. Thus, the expected loss in quality of a solution will be heavily outweighed by the potential reduction in the required computation time to reach such a solution.

8.7 Chapter summary

There are several factors to consider when employing a clustering algorithm to partition data sets. The first such consideration is the distance measure used to populate the dissimilarity matrix. Five commonly used distance measures were reviewed in §8.1. The next area of concern is deciding which clustering algorithm to implement; nine such algorithms were discussed in §8.2. The final prevalent concern before implementing a clustering algorithm, is determining the number of clusters to pursue, with twenty-three measures recommended in the literature for this purpose summarised in §8.3.

The aforementioned measures and algorithms were all implemented in respect of three well-known CVRP benchmark test instances. The internal validity and stability measures described in §8.4, were employed in §8.5 to ascertain the best combination of algorithms and number of clusters upon which to base the CVRP solution process, taking cognisance of both solution quality and computation time.

The results of the experiments were presented in §8.6, exhibiting a decline in solution quality when incorporating a clustering phase in the CVRP solution process. A rather drastic improvement in computation time required by the ACS algorithm to return approximate solutions to the three CVRP benchmark instances was, however, noted in each case.

Part III

Tiered Vehicle Routing Problem with Global Cross-Docking

CHAPTER 9

TVRPGC Model Formulation

Contents

A description of the mathematical model proposed in this dissertation for the TVRPGC, as described briefly in §1.2, is presented in this chapter. A number of necessary assumptions required to formulate the model are presented along with the relevant model objectives and constraints. The model is verified by implementing it in a commercially available MIP solver and solving the model in the context of a small, hypothetical problem instance. The results and complexity of the model are discussed and the chapter closes with a brief summary.

9.1 Introduction

The class of VRPs has enjoyed a long and colourful history since its inception in 1959 by Dantzig and Ramser [107], resulting in numerous variations on the celebrated CVRP prototype within this class. These variations have typically arisen due to the need to be able to accommodate a variety of practical considerations such as taking into account operating hours of facilities, adhering to limitations in infrastructure and incorporating diversity into the vehicle fleet. This has led to the introduction into the literature of widely accepted classes of model formulations accommodating these features, such as VRPs with time windows [122, 252, 260], VRPs accommodating local cross-docking [384, 447], VRPs with multi-echelon facilities [135, 343] and VRPs with trailer considerations [75, 414], to name but a few. Many of these problem variations were reviewed briefly in Chapter 2.

In most VRP applications, a characterisation of customers or facilities in terms of different commodity processing capabilities is not applicable. In this chapter, a variation on the VRP with time windows is, however, considered where commodities of different types have to be collected from a set of customers and processed in potentially different ways at a set of facilities within a transportation network. The variation in commodity type may be due to the nature of the commodities themselves, such as their purpose and processing requirements, as well as maintaining standards associated with a commodity, or may even be due to the intended destinations of the commodities, such as local, regional, provincial, national, or international destinations. The available commodity processing facilities are segregated according to their respective processing and storage capabilities into a set of tiers. This tier allocation is nested in the sense that a facility of tier i can process any type of commodity that can be processed at a facility of tier j if $j < i$, but there exist certain commodity types which can be processed at a facility of tier i that cannot be processed at any facility of a lower tier. Facilities of the lowest tier represent customers at which the commodities originate and have to be collected — these facilities have no commodity processing or storage capabilities — their only role is that they introduce new commodities into the system. Facilities of higher tiers may or may not introduce new commodities into the system, but their distinguishing feature is that they all offer commodity processing capabilities. Furthermore, all facilities, excluding facilities of the lowest tier, are assumed to offer the same storage capabilities.

Crucially, handover of commodities at facilities is allowed in the sense that a commodity requiring processing at a facility of a specific tier may be transported by one vehicle to a facility of a lower tier than the required one, and then be collected later by some other vehicle(s) which transports it to a facility of the required tier. This type of commodity handover, which may occur at a facility of any tier (save the lowest and the highest[1]), is referred to as *global cross-docking*[2]. Another novel feature of the VRP variation considered here is that demand for commodity collection is allowed to spill over into a subsequent planning period. Essentially, the assumption is made that the time continuum may be partitioned into planning periods of fixed length. One planning period is considered at a time, and if demand for commodity collection occurs at a facility after the last vehicle has departed from that facility, then this commodity is simply collected from the facility during the following planning period (all demand for commodity collection is assumed to be known at the beginning of the planning period). Individual commodities are not tracked as they travel through the system, but they nevertheless all require collection at their originating customers and transportation to facilities with adequate processing capabilities. This requirement is met by constructing a model which produces a flow route (perhaps consisting of several individual vehicle sub-routes) for commodities from any facility (except facilities of the highest tier) to a facility of a strictly higher tier, thereby facilitating delivery of the commodities to facilities of the tiers required, perhaps after repeated global cross-docking operations.

There may be many real-world applications of the type of VRP described above. Two such applications are mentioned in this introductory section — one in the healthcare sector and one in the postal services sector as described in §1.2. The facilities where commodities originate are referred to as facilities of tier zero as they do not offer any processing capabilities. In rural settings, the distribution of the commodity processing facilities is such that for commodities to reach a processing facility of the required tier, global cross-docking is a necessity, since it may be impossible for a single vehicle to deliver commodities originating in very rural settings over the long distances required to reach a suitable tier of processing facility in view of legal maximum driving times.

The commodity collection and processing system with global cross-docking and demand spill-over to subsequent planning periods described above is modelled in this chapter as a tri-objective

[1]Global cross-docking of commodities at facilities of the highest tier is not necessary as all commodities considered in the transportation network can be processed at facilities of the highest tier. Global cross-docking of commodities may also not occur at facilities of the lowest tier as they do not offer any processing or storage capabilities.

[2]As opposed to the traditional notion of *cross-docking* in the supply chain literature where goods are consolidated at a dedicated cross-docking facility [234, 288], referred to here as *local cross-docking*.

VRP which may form the basis of a decision support system capable of assisting tiered-facility services in respect of cost-effective planning, routing and scheduling of a fleet of homogeneous vehicles dedicated to commodity collection and delivery. The mathematical model for the TVRPGC derived in this chapter builds on a combination of various well-known variants of the celebrated CVRP in the literature, but exhibits various novel features, as outlined above. An acceptable trade-off between the three objectives is pursued in the model, namely minimisation of the cost associated with transporting commodities, minimisation of the difference between the longest and shortest travel times associated with vehicles (*i.e.* balancing of driver workload) and, finally, minimisation of the number of vehicles required to implement the commodity collection routing schedule.

9.2 Model assumptions

In the mathematical model for the TVRPGC proposed in this chapter, certain assumptions are required in order to render possible a mathematical description of tiered-facility routing operations, as described in §9.1. These assumptions, introduced in order to simplify the mathematical model, are, however, still able to offer a fair representation of real-life operations of tiered-facility networks in which global cross-docking occurs, as confirmed by an industry expert [33], and are as follows:

1. *The nature of the facilities.* The transportation network consists of customers, consolidation points, and facilities of varying commodity processing and storage capabilities, which are collectively referred to as *facilities*. Commodities introduced into the network of facilities exhibit varying processing requirements, which are in certain cases only satisfiable by some subset of facilities. Therefore, the facilities are segregated into a collection of tiers according to the commodity processing capabilities that they offer, with a higher tier suggestive of superior processing capabilities. The tiers are ordered in such a manner that the lowest-tier facilities only require commodity collection, the highest-tier facilities only offer processing capabilities, and all the other facilities both require commodity collection and offer processing capabilities as these facilities are all able to process certain commodities, but may require commodities to be transported to more capable facilities for processing. As mentioned in §9.1, the various tier levels of facilities are assumed to exhibit nested commodity processing capabilities. Facilities of the lowest and highest tiers furthermore do not offer any storage or consolidation capabilities. All other tiers of facilities, however, offer the same storage or consolidation capabilities.

2. *The nature of the vehicles.* It is assumed that a fleet of homogeneous vehicles is available for commodity collection. The capacities of the vehicles are assumed to be sufficiently large to handle any demand requirements. Unlike in most VRPs, the intended cargo is therefore assumed to be of negligible volume and weight. A capacity constraint may nevertheless easily be included in the model formulation, if required. This is, however, normally not necessary in both the healthcare and postal service applications mentioned in §9.1. Each vehicle may perform at most one route.

3. *Home depot allocation.* It is assumed that each vehicle has a fixed home depot which may be located at any of the facilities within the network. All vehicles must begin and end their routes at their respective home depots.

4. *Multiple visits and global cross-docking.* A facility may be visited by more than one vehicle during the planning period, although any specific vehicle may visit any facility at most

once during the planning period. In particular, a commodity may be delivered to a facility by a vehicle, and then later be collected by a different vehicle for further transportation in the network.

5. *Service times.* The service time of a facility by a vehicle is limited to the loading and/or unloading of commodities at the facility and does not include the processing times of the commodities. The facilities in the transportation network are not assumed to be operational for twenty four hours a day. Therefore, there is a need for collection and delivery of commodities by vehicles within certain time windows that reflect the operational hours of each facility.

6. *Rolling demand horizon.* It is assumed that demand for commodity collection occurs on a continual basis at all but the highest-tiered facilities, regardless of the time within the planning period. Unmet demand from the previous planning period may therefore be brought forward to the current planning period. This allows for a vehicle to deliver commodities to and collect commodities from the same facility without having to wait at the facility for all demand to have realised there. Demand for specimen collection that occurs at a facility after the last vehicle has departed from the facility may be satisfied during the following planning period.

7. *Facility visitation sequence.* For feasibility of a route, it is required that every facility (except the highest-tiered facilities) should be visited by at least one vehicle that also visits a higher-tier facility at a later stage within the planning period or should participate with another vehicle in cross-docking at a consolidation facility such that the specimens of the facility reach a strictly higher-tiered facility.

8. *Commodity destinations.* In a bid to reduce model complexity, individual commodity collection and transportation is not tracked explicitly in the model formulation as numerous types of commodities may be collected and an even larger number of possible types of commodity processing may be required by these commodities. The only constraint is that a commodity should eventually be delivered to a facility capable of processing it (perhaps over the course of several succesive planning periods).

9. *Commodity expiration.* The possible deterioration of the quality of a commodity over time is limited to the time it takes for the commodity to be collected from a facility of the lowest tier and transported to a facility that has the appropriate processing or consolidation capabilities (*i.e.* commodity deterioration occurs only as a result of being in transit). It is therefore assumed that once a commodity has been delivered to a facility (of tier greater than the lowest tier) for the first time, the commodity is either processed there or stored in such a manner that its expiration window remains unaffected during storage (*i.e.* in a vacuum or at a low temperature) or future transportation (*i.e.* repackaged in such a manner so as to retain the commodity's integrity).

9.3 Mathematical model formulation

This section contains a detailed description of the sets of constraints and planning objectives required to translate the TVRPGC, described briefly in §9.1 and elaborated upon in §9.2, into a formal MILP model. After defining the model parameters and variables in §9.3.1 and §9.3.2, respectively, the model objectives are formulated mathematically in §9.3.3. The focus then shifts in §9.3.4 to the formulation of the model constraints.

9.3.1 Model parameters

Suppose there are $f+1$ different tiers of facilities in the system, and that each facility tier (save the lowest) is associated with specific commodity processing capabilities. Suppose, furthermore, that indices are assigned to these facility tiers in such a manner that a facility of tier $d > 1$ possesses a superset of the processing capabilities of a facility of tier e for any $e \in \{1, \dots, d-1\}$, but that all laboratories of the same tier have identical processing capabilities. As mentioned in §9.2, the customers at which commodities originate for collection and the processing facilities, which may also exhibit demand for commodity collection, are together referred to as *facilities*. An indexing convention is, however, followed where all customers exhibiting no processing capabilities are referred to as *facilities of tier zero*, while all processing facilities of tier $d \in \{1, \dots, f\}$ are referred to as *facilities of tier d*. Let $\mathcal{F}^d$ denote the set of all facilities of tier $d \in \{0, 1, \dots, f\}$, and define $\mathcal{F} = \cup_{d=0}^{f} \mathcal{F}^d$ as the set of all the facilities. Any facility in $\mathcal{F}^0$ therefore has no commodity processing capability, but only exhibits demand for commodities to be collected there. Any facility in $\mathcal{F}^f$, on the other hand, only processes commodities, and exhibits no demand for the collection of such commodities. Finally, any facility in $\mathcal{F} \setminus (\mathcal{F}^0 \cup \mathcal{F}^f)$ may or may not exhibit demand for commodity collection as a result of cross-docking operations there and also offers certain processing capabilities. Facility $i \in \mathcal{F}$ furthermore has an associated vehicle arrival capacity γ_i (*i.e.* a limit on the number of vehicle arrivals the facility can accommodate during the planning period), a required service time of s_i time units and a service time window $[a_i, g_i]$ during which vehicles have access to the facility.

Let $\mathcal{V}$ represent the set of homogeneous vehicles that constitute the commodity collection fleet. As mentioned in §9.2, it is assumed that this set of vehicles is sufficiently large to facilitate feasible commodity collection routing and scheduling at a 100% service level. The homogeneity of the fleet implies that all vehicles have the same autonomy level μ (the maximum allowable route duration of a vehicle, measured in units of expected travel time) and that any two vehicles are expected to traverse a given road link within the transportation network in the same amount of time. Denote the subset of facilities acting as home depots for vehicles by $\mathcal{D}$ and denote the home depot of vehicle $k \in \mathcal{V}$ within this set by b_k. As is customary in the VRP literature, each home depot b_k is associated with a virtual, identical copy of the depot, denoted by b_k^+, in order to be able to distinguish between the departure time of a vehicle from its home depot and the later arrival time of the vehicle when returning to its home depot. In particular, b_k represents the home depot of vehicle $k \in \mathcal{V}$ when it departs from the depot, while b_k^+ represents the same home depot when the vehicle returns to the depot upon completion of its route. The departure time $T'_{b_k k}$ of vehicle $k \in \mathcal{V}$ from the depot b_k is known *a priori*.

The set of all commodities that have to be collected is partitioned into f distinct types, indexed by the set $\mathcal{S} = \{1, \dots, f\}$, according to the convention that a commodity of type $c \in \mathcal{S}$ can be processed at any facility in $\cup_{d=c}^{f} \mathcal{F}^d$. Each commodity of type $c \in \mathcal{S}$ is assumed to have an associated expiration time τ_c which is an upper bound on the time the commodity may be in transit before it is delivered to a facility in $\cup_{d=c}^{f} \mathcal{F}^d$.

Let $\mathcal{G} = (\mathcal{F}, \mathcal{E})$ be a complete, directed, weighted graph with vertex set $\mathcal{F}$ and arc set $\mathcal{E}$ representing all possible road network connections between facilities in $\mathcal{F}$, where the weight of an arc $(i, j) \in \mathcal{E}$ is the expected travel time t_{ij} of a vehicle traversing the arc from facility $i \in \mathcal{F}$ to facility $j \in \mathcal{F}$. It is assumed that the triangle inequality is upheld by these expected travel times.

The planning period is limited to a schedule of fixed length, implemented (possibly in slightly altered form) along a rolling horizon. Some subset of facilities in $\mathcal{F} \setminus \mathcal{F}^f$ may perhaps not exhibit demand for commodity collection within the planning period under consideration, due to demographic variability and fluctuating demand. Let the binary parameter α_{ic} therefore assume the value 1 if commodities of type $c \in \mathcal{S}$ have to be collected from facility $i \in \mathcal{F} \setminus \mathcal{F}^f$, or the value 0 otherwise.

Finally, let $\mathcal{N}$ denote a set of global event numbers associated with the vehicle routing schedule over the planning period. The elements of this set induce a global ordering of vehicle arrivals over time at the various facilities in the spirit of Dondo *et al.* [135] (who applied this model construct in the special case of local cross-docking in supply chain management). In their application, the arrival of each vehicle at a pre-specified local cross-docking facility was associated with a unique integer in such a manner that a later arrival of any vehicle at the facility was associated with a larger integer. In the application adopted in this dissertation, the practice of assigning the arrival of each vehicle a unique integer value is also implemented. The application of the aforementioned model construct adopted in this dissertation, however, differs from that of Dondo *et al.* [135] in the consideration of the arrival times of all vehicles at all of the facilities in the network, as opposed to at a specific cross-docking facility only. This model construct is applied to monitor the global cross-docking and tier-visitation of vehicles.

9.3.2 Model variables

In the model formulation, decision and auxiliary variables are required to keep track of the movement of vehicles and their service allocation to facilities. In order to facilitate the orchestration of global cross-docking operations, a global ordering is assigned to the arrivals of all vehicles in the routing schedule, as described above. The auxiliary variables

$$y_{nik} = \begin{cases} 1, & \text{if the arrival of vehicle } k \in \mathcal{V} \text{ at facility } i \in \mathcal{F} \text{ is global} \\ & \text{event } n \in \mathcal{N} \text{ during the current planning period,} \\ 0, & \text{otherwise} \end{cases}$$

achieve this purpose in conjunction with the auxiliary variables

$$z_{ijkn} = \begin{cases} 1, & \text{if the arrival of vehicle } k \in \mathcal{V} \text{ at facility } i \in \mathcal{F} \setminus (\mathcal{F}^0 \cup \mathcal{F}^f) \\ & \text{is global event } n \in \mathcal{N}\text{, following which vehicle } k \text{ also visits} \\ & \text{facility } j \in \mathcal{F}^\ell \text{ at some later stage, where facilities } i \text{ and } j \\ & \text{are of the same tier } \ell, \\ 0, & \text{otherwise,} \end{cases}$$

where $\mathcal{N}$ denotes a set of non-negative integers, with $|\mathcal{N}| = |\mathcal{F}|+(|\mathcal{V}|-1)+(|\mathcal{V}|-1)|\mathcal{F}\setminus(\mathcal{F}^0\cup\mathcal{F}^f)|$. The assignment decision variables

$$r_{ikn} = \begin{cases} 1, & \text{if global event } n \in \mathcal{N} \text{ involves the assignment of vehicle } k \in \mathcal{V} \text{ to} \\ & \text{visit facility } i \in \mathcal{F} \setminus (\mathcal{F}^0 \cup \mathcal{F}^f) \text{ and this vehicle later visits a} \\ & \text{facility of a higher tier than that of facility } i, \\ 0, & \text{otherwise} \end{cases}$$

are used in a disjunctive fashion to enforce appropriate facility visitation sequences. Finally, the flow decision variables

$$x_{ijk} = \begin{cases} 1, & 1 \text{ if vehicle } k \in \mathcal{V} \text{ travels directly from facility } i \in \mathcal{F} \text{ to } j \in \mathcal{F}, \\ 0, & \text{otherwise} \end{cases}$$

monitor the movement of vehicle $k \in \mathcal{V}$, while the non-negative, real auxiliary variables T_{ik} denote the time at which vehicle $k \in \mathcal{V}$ arrives at facility $i \in \mathcal{F}$, with T_{ik} assuming the value zero for all $i \in \mathcal{F}$ if vehicle k is not used.

9.3.3 Model objectives

Following the discussion in §9.1, the aim of the model proposed in this chapter is to pursue an acceptable trade-off between the realisation of three objectives. The first of these objectives is to minimise the expected global travel time[3] associated with the transportation of all commodities from the various original commodity collection facilities to appropriate facilities where they are to be processed or stored. This objective may be formulated mathematically as

$$\text{minimise} \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{F}}\sum_{k\in\mathcal{V}} t_{ij}x_{ijk}. \tag{9.1}$$

The second objective is to balance the workload of the delivery vehicles in terms of their total service travel times, that is to

$$\text{minimise} \max_{k\in\mathcal{V}} \left(T_{b_k^+k} - T'_{b_kk}\right). \tag{9.2}$$

The final objective is to

$$\text{minimise} \sum_{k\in\mathcal{V}}\sum_{j\in\mathcal{F}} x_{b_kjk}, \tag{9.3}$$

which is equivalent to minimising the number of vehicles required for commodity collection at a service level of 100% by reducing the number of vehicles departing from their home depots.

9.3.4 Model constraints

The model includes numerous constraints reflecting the various requirements of the TVRPGC in respect of the transportation of commodities. The first such constraint states that every vehicle must initially depart from and eventually return to its home depot at the end of its route, as required by Assumption 3 of §9.2. This constraint is enforced by requiring that

$$\sum_{j\in\mathcal{F}} x_{b_kjk} \leq 1, \qquad k\in\mathcal{V}$$

and that

$$\sum_{j\in\mathcal{F}} x_{jb_k^+k} = \sum_{j\in\mathcal{F}} x_{b_kjk}, \qquad k\in\mathcal{V}.$$

The constraint set

$$\sum_{i\in\mathcal{F}} x_{ijk} \leq \sum_{\ell\in\mathcal{F}} x_{b_k\ell k}, \qquad j\in\mathcal{F},\ \ k\in\mathcal{V}$$

ensures that any vehicle $k\in\mathcal{V}$ visits a facility $j\in\mathcal{F}$ at most once during the planning period according to Assumption 4. The flow conservation constraint set

$$\sum_{i\in\mathcal{F}} x_{ijk} - \sum_{\ell\in\mathcal{F}} x_{j\ell k} = 0, \qquad j\in\mathcal{F}\setminus\{b_k, b_k^+\},\ \ k\in\mathcal{V}$$

states that if any vehicle $k\in\mathcal{V}$ arrives at facility j, then the same vehicle must traverse an arc departing from facility j, for all $j\in\mathcal{F}\setminus\{b_k, b_k+\}$. Since not all facilities $i\in\mathcal{F}\setminus\mathcal{F}^f$ necessarily exhibit demand for commodity collection during the planning period, the constraint set

$$\sum_{j\in\mathcal{F}}\sum_{k\in\mathcal{V}} x_{ijk} \geq \overline{\alpha}_i, \qquad i\in\mathcal{F}\setminus\mathcal{F}^f$$

[3]The decision not to minimise the distance travelled by vehicles stems from possibly very rural locations of some of the facilities. The potentially poor quality of roads leading to these remote facilities in a developing context often brings about considerable deviations in the expected travel time per unit distance.

ensures that at least one vehicle $k \in \mathcal{V}$ should visit facility $i \in \mathcal{F} \setminus \mathcal{F}^f$ if there is actually demand for commodity collection at facility i, where

$$\overline{\alpha}_i = \begin{cases} 1, & \text{if } \sum_{c \in \mathcal{S}} \alpha_{ic} \geq 1 \\ 0, & \text{otherwise.} \end{cases}$$

The constraint set

$$T_{ik} + s_i + t_{ij} - T_{jk} \leq (1 - x_{ijk})M, \quad i \in \mathcal{F}, \; j \in \mathcal{F}, \; k \in \mathcal{V}$$

is included to monitor the arrival time of vehicle $k \in \mathcal{V}$ at each vertex along its route. This constraint set ensures, if vehicle $k \in \mathcal{V}$ travels from facility $i \in \mathcal{F}$ to facility $j \in \mathcal{F}$, that the time instant at which it starts to service facility j is bounded from below by the time instant at which it started servicing facility i together with the combined service time duration at facility i and the time required to travel from facility i to facility j. Here M is a large positive number. The services provided by tiered-facility organisations and the respective processing facilities are furthermore not typically twenty four hour operations, but should be rendered within acceptable time windows associated with each facility according to Assumption 5. Since there is a possibility that not all vehicles $k \in \mathcal{V}$ may be used, the constraint set

$$T'_{b_k k} + t_{b_k j} - M(1 - x_{b_k jk}) \leq T_{jk}, \quad j \in \mathcal{F}, \; k \in \mathcal{V}$$

defines the arrival time of vehicle $k \in \mathcal{V}$ at the first facility $j \in \mathcal{F}$ visited by vehicle k, where M is again a large positive number. If vehicle k is not used, the values of T_{ik} should be equal to zero for all $i \in \mathcal{F}$. The constraint set

$$a_i \sum_{j \in \mathcal{F}} x_{jik} \leq T_{ik} \leq g_i \sum_{j \in \mathcal{F}} x_{jik}, \quad i \in \mathcal{F}, \; k \in \mathcal{V}$$

states that vehicle k may not arrive at a facility $i \in \mathcal{F}$ outside of its associated time window and enforces the requirement mentioned above that if vehicle $k \in \mathcal{V}$ does not visit facility $i \in \mathcal{F}$, the value of T_{ik} is equal to zero. The constraint set

$$T_{b_k^+ k} - T'_{b_k k} \leq \mu, \quad k \in \mathcal{V}$$

ensures that vehicle $k \in \mathcal{V}$ does not undertake a route which is expected to take longer to complete than the allowable time autonomy level assigned to the vehicle. Apart from the multiple problem objectives, an aspect of the novelty of the VRP model formulated here is elucidated in the next constraint set. Each commodity of type $c \in \mathcal{S}$ has a certain time window associated with it during which the commodity remains viable. As discussed in Assumption 8, the specific requirements of each individual commodity and its intended purpose are not traced explicitly. Instead, a more abstract approach is taken by imposing the constraint set

$$T_{jk} - T_{ik} \leq \min_{c \in \mathcal{S}: \alpha_{ic} = 1} \{\tau_c\} + M \left(2 - \sum_{\ell \in \mathcal{F}} x_{\ell ik} - \sum_{\ell \in \mathcal{F}} x_{\ell jk} \right), \quad i \in \mathcal{F}^0, \; k \in \mathcal{V}, \; j \in \mathcal{F} \setminus \mathcal{F}^0,$$

which requires that a commodity is delivered to a facility able to process or store it in such a manner that its integrity is not affected (see Assumption 9). Here M is again a large positive number. The tiered nature of the facilities refers to the processing capabilities of the facilities: Every facility tier has an associated processing capability in respect of commodities, as described in Assumption 2. As the model does not, however, track individual commodity processing requirements, the more practical approach, described in Assumption 8, is adopted, whereby the number of vehicles arriving at a facility is limited in order to prevent processing bottlenecks. The constraint set

$$\sum_{k\in\mathcal{V}}\sum_{i\in\mathcal{F}} x_{ijk} \leq \gamma_j, \quad j \in \mathcal{F}\setminus\mathcal{F}^0$$

requires that the number of vehicles arriving at facility $j \in \mathcal{F}\setminus\mathcal{F}^0$ should not exceed the arrival capacity of the facility over the scheduling window. The novelty of the VRP model derived here is further showcased by the remaining constraint sets, which all contribute to controlling the sequencing of vehicle arrivals at facilities so as to facilitate global cross-docking. The constraint set

$$\sum_{i\in\mathcal{F}}\sum_{k\in\mathcal{V}} y_{nik} \leq 1, \quad n \in \mathcal{N}$$

ensures that the arrival of each vehicle at every facility $i \in \mathcal{F}$ is assigned at most one global event index $n \in \mathcal{N}$, with every facility actually exhibiting commodity collection demand being assigned a unique global event index by prescribing the constraint set

$$\sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{V}} y_{nik} \geq \overline{\alpha}_i, \quad i \in \mathcal{F}\setminus\mathcal{F}^f.$$

It is required that the global event indices assigned to vehicle arrivals should reflect the order of their arrival sequence in global time. The constraint set

$$T_{j\ell} - T_{ik} \geq M(y_{nik} + y_{mj\ell} - 2), \quad i,j \in \mathcal{F},\ k,\ell \in \mathcal{V},\ m,n \in \mathcal{N} : m > n$$

achieves this requirement by ensuring that $T_{j\ell} \geq T_{ik}$ if $y_{nik} = 1$ and $y_{mj\ell} = 1$. Here M is again a sufficiently large positive number. For every facility $i \in \mathcal{F}\setminus\mathcal{F}^f$ there must be some vehicle $k \in \mathcal{V}$ visiting a higher-tiered facility at some time after having visited facility i, as explained in Assumptions 7 and 8. The disjunctive constraint set

$$\sum_{k\in\mathcal{V}}\sum_{n\in\mathcal{N}}\left(r_{ikn} + \sum_{j\in\mathcal{F}^\ell} z_{ijkn}\right) \geq 1, \quad i \in \mathcal{F}^\ell,\ \ell \in \{0,\ldots,f-1\}$$

enforces this requirement. This constraint set ensures that for each facility i of tier $\ell < f$ there exists a vehicle $k \in \mathcal{V}$ visiting the facility with a corresponding event number $n \in \mathcal{N}$ such that $r_{ikn} = 1$ (indicating that vehicle k later visits some facility of a tier higher than ℓ) or $z_{ijkn} = 1$ for some facility j of tier ℓ, with $j \neq i$ (indicating that vehicle k later visits facility j), in accordance with Assumption 7. The linking constraint set

$$py_{nik} + \sum_{\substack{m\in\mathcal{N}\\ m>n}}\ \sum_{j\in\cup_{\ell=c+1}^{f}\mathcal{F}^\ell} y_{mjk} \geq (p+1)r_{ikn}, \qquad i \in \mathcal{F}^c,\ c \in \{0,\ldots,f-1\},\ k \in \mathcal{V},\ n \in \mathcal{N}$$

furthermore ensures that the variable r_{ikn} may only assume a value of 1 if vehicle $k \in \mathcal{V}$ actually visits facility $i \in \mathcal{F}^c$ and at some later stage also visits facility j of tier higher than c, where p denotes the number of vertices in the transportation graph $\mathcal{G}$. The constraint set

$$\sum_{j\in\mathcal{F}} x_{jik} = \sum_{n\in\mathcal{N}} y_{nik}, \quad i \in \mathcal{F},\ k \in \mathcal{V}$$

ensures that an event $n \in \mathcal{N}$ cannot be assigned to the arrival of a vehicle $k \in \mathcal{V}$ at a facility $i \in \mathcal{F}$, unless vehicle k actually visits facility i. The powerful disjunctive constraint sets above are highly dependent on the auxiliary variables r_{ikn}. The linking constraint set

$$\sum_{n\in\mathcal{N}} r_{ikn} \leq \sum_{n\in\mathcal{N}} y_{nik}, \quad i \in \mathcal{F},\ k \in \mathcal{V}$$

enforces the correct assignment of values to these binary variables. The global cross-docking component of the model allows for facilities of the same tier to have their commodities consolidated at any facility of that tier within the transportation network. The constraint set

$$y_{nik} + \sum_{\substack{m \in \mathcal{N} \\ m > n}} y_{mjk} \geq 2z_{ijkn}, \quad i, j \in \mathcal{F}^{\ell},\ \ell \in \{1, \ldots, f-1\},\ n \in \mathcal{N},\ k \in \mathcal{V}$$

finally ensures that the auxiliary variable z_{ijkn} only assumes the value 1 if a vehicle visits facility $i \in \mathcal{F}^{\ell}$ (with $\ell \neq 0, f$) and then at a later time also visits facility $j \in \mathcal{F}^{\ell}$, allowing for consolidation of commodities of both facilities at facility j, to be collected by a possibly different vehicle $k \in \mathcal{V}$ for transportation to a higher-tiered facility.

9.4 A worked example

The logic of the TVRPGC model of §9.3 is verified in this section by implementing it in a commercially available MILP solver within the context of a small, hypothetical problem instance. The aim of the worked example is not to evaluate experimentally the computational performance of the proposed MILP model (which may perhaps be improved substantially by reducing its "symmetry" characteristics, and by applying effective preprocessing procedures to decrease the number of variables and constraints), but rather to demonstrate its capability to deal with the global cross-docking properties and the peculiar constraints of the TVPRGC.

The hypothetical test instance considered in this section is presented in the context of the collection of pathological specimens by a healthcare service provider and their delivery to appropriate specimen testing laboratories, as described in §1.1. There are seven facilities of three different tiers in the test instance, and so $f = 2$ in this case. The first of these facilities, listed in Table 9.1, is the depot for all vehicles. Facilities 2, 5 and 6 are hospitals or clinics where pathological samples originate. These collection stations have no blood analysis capabilities, and so they are classified as facilities of tier zero. Facilities 3 and 4 are hospitals where blood sample analysis laboratories of tier one are located, while Facility 7 is a tier-two laboratory.

TABLE 9.1: *Seven facilities in a small, hypothetical test problem instance of a tiered-facility network.*

Facility Number	Facility Type	X-coordinate	Y-coordinate
1	Depot	190	190
2	0	230	210
3	1	220	260
4	1	110	230
5	0	150	270
6	0	50	180
7	2	10	0

The hypothetical test instance considered here was constructed in a manner to highlight the concept of global cross-docking. Hence some of the model parameters of §9.3.1 which do not affect cross-docking constraints, such as the imposition of time windows and the adherence to arrival capacities of facilities, were set to generally unconstraining values, so as to reduce the complexity of finding an initial feasible solution. Thus, the values $a_i = 0$, $g_i = 5\,000$ (expressed in minutes) were specified for every facility $i \in \mathcal{F}$. A maximum driver autonomy value of 740 minutes was also imposed in an attempt to prohibit a single vehicle from servicing all the customers. Finally, the arrival capacities of facilities were specified as $\gamma_i = 1$ for all facilities $i \in \mathcal{F}^0$, $\gamma_j = 2$ for both facilities $j \in \mathcal{F}^1$ and $\gamma_\ell = 3$ for the facility $\ell \in \mathcal{F}^2$.

The expected travel times between these facilities are shown in Table 9.2, and were calculated as the corresponding Euclidean distances between the facilities.

TABLE 9.2: *Travel times (in minutes) between the respective facilities.*

Facility	1	2	3	4	5	6	7
1	—	44.72	76.16	89.44	89.44	140.36	261.73
2	44.72	—	50.99	121.66	100.00	182.48	304.14
3	76.16	50.99	—	114.02	70.71	187.88	334.22
4	89.44	121.66	114.02	—	56.57	78.10	250.80
5	89.44	100.00	70.71	56.57	—	134.54	304.14
6	140.36	182.48	187.88	78.10	134.54	—	184.39
7	261.73	304.14	334.22	250.80	304.14	184.39	—

A complete enumeration of all feasible routes was performed, implemented in Wolfram's Mathematica [456], in order to generate the true Pareto front for the hypothetical problem instance in the cases where either two or three delivery vehicles are employed. This enumeration process consisted of seven phases:

Phase 1. A nonempty subset of the set of facilities was selected for visitation by a delivery vehicle. Since the depot (Facility 1) necessarily has to be included in the visitation set, this resulted in $\sum_{i=1}^{6}\binom{6}{i} = 2^6 - 1 = 63$ possible facility visitation subsets for any single vehicle.

Phase 2. The facility visitation subsets identified during Phase 1 were combined in order to form an assignment of customers to be visited by each vehicle in the fleet. This led to 3 969 (in the case of two vehicles) and 250 047 (in the case of three vehicles) facility-to-vehicle assignment alternatives, respectively.

Phase 3. From the set of facility-to-vehicle assignment alternatives constructed during Phase 2, all those alternatives in which not all facilities are visited, were removed. This reduced the set of facility-to-vehicle assignment alternatives to a total of 727 (in the case of two vehicles) and 115 464 (in the case of three vehicles) alternatives, respectively.

Phase 4. All alternatives in which the vehicle arrival capacities at facilities are exceeded, were removed next. Accordingly, all alternatives in which a facility of tier 0 appears more than once and all alternatives in which a facility of tier 1 appears more than twice were removed from consideration. This led to 214 (in the case of two vehicles) and 6 159 (in the case of three vehicles) remaining facility-to-vehicle assignment alternatives, respectively.

Phase 5. The orders in which facilities are visited by each vehicle were taken into account by permuting (in all possible ways) the non-depot facilities in each of the facility-to-vehicle visitation sets within the alternatives that remained after the filtering process of Phase 4, ensuring that the depot (Facility 1) remains in the first and last position of each permutation. This resulted in 54 288 (in the case of two vehicles) and 370 800 (in the case of three vehicles) potential vehicle routing combinations, respectively.

Phase 6. Infeasible vehicle routing combinations were next removed from those combinations identified during Phase 5. The infeasibilities considered occurred due to violations of the requirement that each facility of tiers 0 and 1 must be visited by a vehicle that visits a strictly higher-tiered facility or participates in cross-docking such that all pathological specimens are eventually able to reach a strictly higher-tiered facility. This resulted in

13 104 (in the case of two vehicles) and 72 662 (in the case of three vehicles) feasible vehicle routing combinations, respectively.

Phase 7. For each of the vehicle routing combinations that remained after the filtering process of Phase 6, (1) the total travel time and (2) the maximum driver autonomy were recorded. All vehicle routing combinations that were dominated in terms of both these objectives were then filtered out, and combinations that violated the individual vehicle autonomy specification (740 minutes per vehicle) were also removed, yielding only three (in the case of two vehicles) and two (in the case of three vehicles) Pareto-optimal vehicle routing combinations, as depicted in the objective function space in Figure 9.1.

Although it violates the driver autonomy bound of 740 mins, the objective function values of the optimal solution single-vehicle TSP are also included for reference purposes in Figure 1.

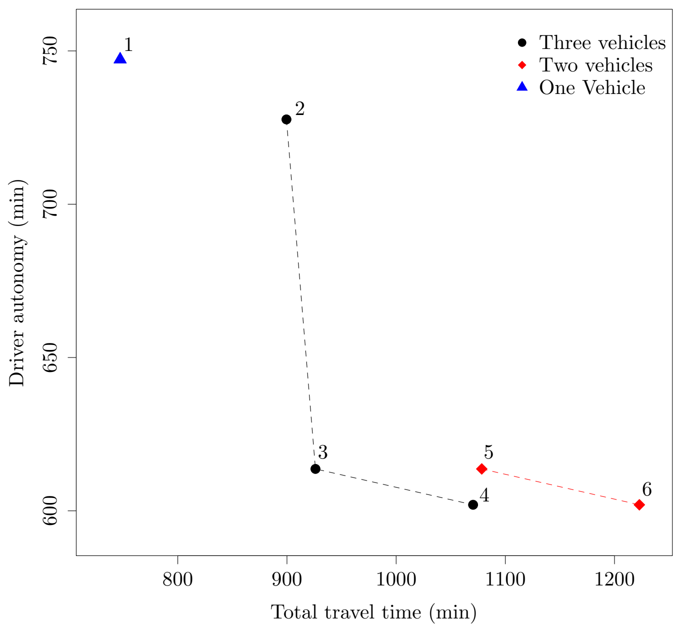


FIGURE 9.1: *True Pareto fronts for the hypothetical test problem instance in the cases of using one, two and three vehicles, respectively.*

The six numbered solutions of Figure 9.1 are depicted in the solution space in Figure 9.2. Among these solutions, the concept of global cross-docking is best illustrated in Solutions 2 and 5.

The mathematical model of §4 was also implemented in CPLEX 12.5 (on an i7-4 770 processor running at 3.40 GHz within a Windows 7 operating system) in respect of the problem instance described above in an attempt to validate the logic of the mathematical formulation. In order to accommodate the pursuit of trade-offs between minimising the total travel time and balancing the driver workload in a solution, the number of vehicles utilised was fixed first as two and then as three. Since CPLEX 12.5 can only handle single-objective MILPs, the decision was made to focus the CPLEX search on replicating Solutions 2 and 6. This allows for single-objective consideration, as the number of vehicles may be fixed, as described above, after which the non-relevant model objective may simply be disregarded.

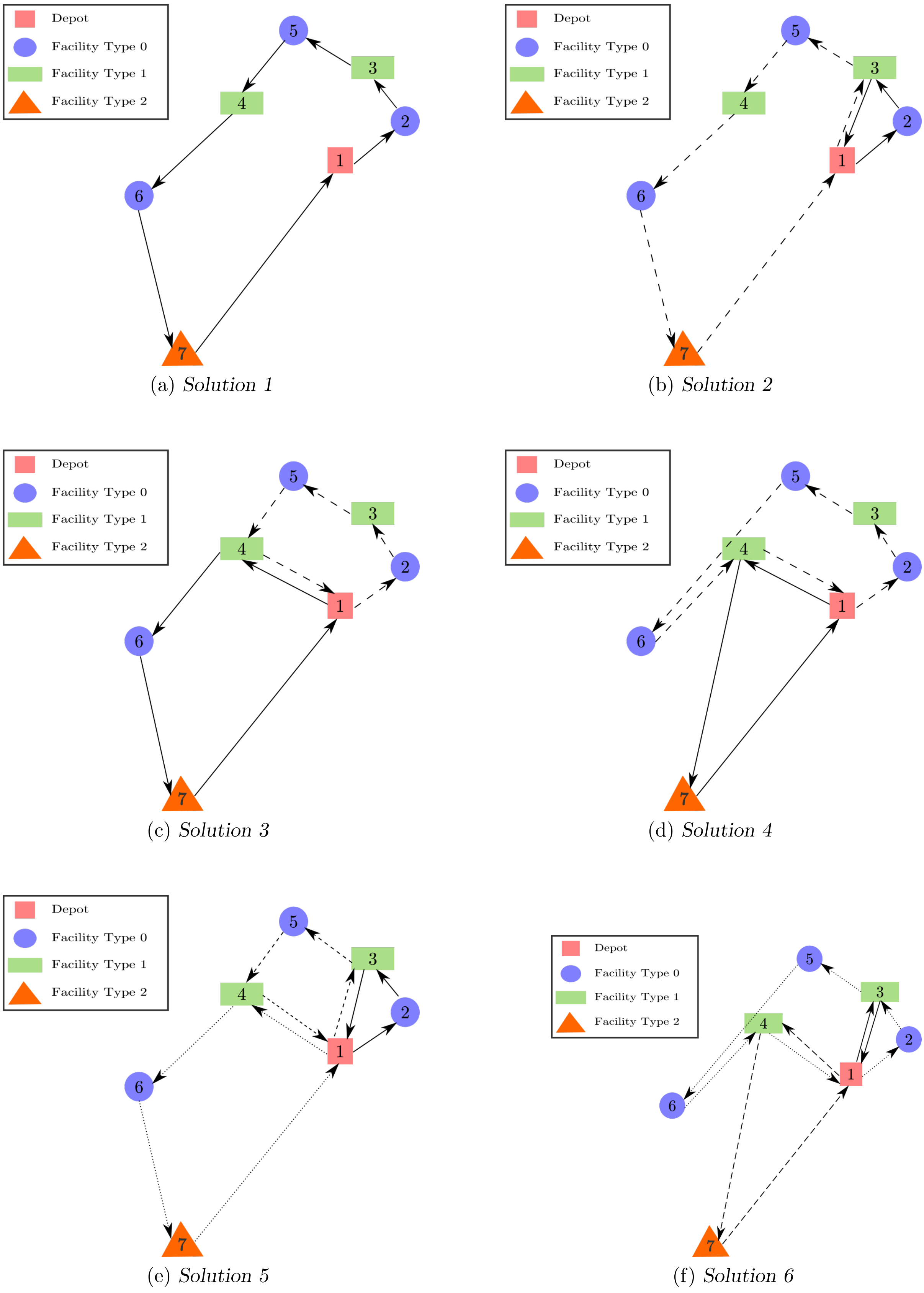


FIGURE 9.2: *The numbered solutions reported in objective function space in Figure 9.1 are depicted here in solution space.*

TABLE 9.3: *Non-zero decision variables returned by CPLEX 12.5 when the number of vehicles is set to two and model objective (2) is removed from consideration.*

Decision variable	Value				
x_{ijk}	$x_{121} = 1$	$x_{231} = 1$	$x_{381} = 1$	$x_{132} = 1$	$x_{352} = 1$
	$x_{542} = 1$	$x_{462} = 1$	$x_{672} = 1$	$x_{782} = 1$	
y_{nik}	$y_{132} = 1$	$y_{252} = 1$	$y_{342} = 1$	$y_{421} = 1$	$y_{531} = 1$
	$y_{562} = 1$	$y_{672} = 1$			
T_{ik}	$T_{21} = 228$	$T_{31} = 279$	$T_{32} = 77$	$T_{42} = 214$	$T_{52} = 157$
	$T_{62} = 293$	$T_{72} = 478$	$T_{81} = 740$	$T_{82} = 740$	
r_{ikn}	$r_{214} = 1$	$r_{321} = 1$	$r_{423} = 1$	$r_{522} = 1$	$r_{625} = 1$
z_{ijkn}					

Accordingly, the number of vehicles were fixed to two and objective (2) above was removed from consideration in order to replicate Solution 2. The values of the non-zero decision and auxiliary variables returned by CPLEX in this case are shown in Table 9.3. The facility index 8 in the tables refers to a copy of the depot (Facility 1).

The total travel time of the two vehicles in solution 2 is 899.53 minutes, while the time spent on the road by each of these vehicles was 171.87 and 727.66 minutes, giving a maximum driver autonomy values of 727.66 minutes.

Similarly, the number of vehicles was fixed to three and objective (1) above was removed from consideration in order to replicate Solution 6. The non-zero decision and auxiliary variables returned by CPLEX 12.5 in this case are shown in Table 9.4. The total travel time of the three vehicles in solution 6 is 1 222.79 minutes, while the time spent on the road by each of these vehicles was 152.32, 468.5 and 601.97 minutes, giving a maximum driver autonomy of 601.97 minutes.

TABLE 9.4: *Non-zero decision variables returned by CPLEX 12.5 when the number of vehicles is set to three and model objective (1) is removed from consideration.*

Decision variable	Value				
x_{ijk}	$x_{131} = 1$	$x_{381} = 1$	$x_{142} = 1$	$x_{472} = 1$	$x_{782} = 1$
	$x_{123} = 1$	$x_{233} = 1$	$x_{353} = 1$	$x_{563} = 1$	$x_{643} = 1$
	$x_{483} = 1$				
y_{nik}	$y_{142} = 1$	$y_{223} = 1$	$y_{333} = 1$	$y_{453} = 1$	$y_{572} = 1$
	$y_{663} = 1$	$y_{743} = 1$	$y_{831} = 1$		
T_{ik}	$T_{42} = 90$	$T_{23} = 90$	$T_{33} = 228$	$T_{53} = 299$	$T_{72} = 341$
	$T_{63} = 434$	$T_{43} = 513$	$T_{31} = 526$	$T_{81} = 603$	$T_{82} = 603$
	$T_{83} = 603$				
r_{ikn}	$r_{232} = 1$	$r_{636} = 1$	$r_{421} = 1$		
z_{ijkn}	$z_{5634} = 1$	$z_{3433} = 1$			

The solutions represented in Tables C.1 and 9.4 are exactly those depicted in Figures 9.2(b) and 9.2(f), respectively. The computation times required by CPLEX to reach these solutions are finally listed in Table 9.5.

Table 9.5: *Computation times (expressed in seconds) required by CPLEX 12.5 to generate the solutions in Figures 9.2(b) and 9.2(f) on an i7-4770 processor running at 3.40 GHz with a working memory limit of 6GB within the Windows 7 operating system.*

Solution	2	6
Time to find initial feasible solution	412 s	936 s
Time to find an optimal solution	429 s	4 973 s
Time to prove optimality	78 698 s	76 777 s

9.5 Chapter summary

A new type of VRP was introduced in §9.1, called the TVRPGC. It is an extension of the celebrated CVRP in which specimens have to be collected from a number of customers and which facilitates global cross-docking (*i.e.* cross-docking at virtually any vertex of the transportation network graph). The assumptions underlying the formalisation of the TVRPGC were discussed in §9.2. The model, which was presented in §9.3, also provides for the partitioning of intermediate facilities into a variety of tiers, arranged according to unique specimen processing capabilities and allows for the possibility of spill over of unmet demand for specimen collection into a next planning period. The model was verified by implementing it in a commercially available MILP solver and solving the model in the context of a small hypothetical problem instance in §9.4.

The worked example of §9.4 highlighted the combinatorial complexity associated with the TVRPGC. This type of complexity calls for the design of approximate solution methodologies in order to facilitate application of the model of §9.3 to real-world problem instances (which are expected to be considerably larger than the test instance considered in §9.4). One such approximate solution approach is presented in the following chapter.

CHAPTER 10

TVRPGC Solution Methodology

Contents

This chapter contains descriptions of the rationale and working of the newly designed MACO algorithm employed in this dissertation to solve instances of the TVRPGC approximately. A number of basic notions central to *multi-objective optimisation* (MOO) are discussed in §10.1. Typical performance measures of algorithms employed to solve MOO problems are next presented in §10.2. The MACO algorithm employed in this dissertation is finally presented in §10.4, and its key components are elaborated upon in pseudocode form, after having provided a motivation for selecting ACO as approximate solution approach in §10.3. The chapter finally closes in §10.5 with a brief summary of its content.

10.1 Basic notions in multi-objective optimisation

MOO is a subdiscipline of *multiple criteria decision making*, which is concerned with solving optimisation problems involving the pursuit of more than one objective concurrently. In MOO, the aim is to simultaneously maximise or minimise d objective functions, $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_d(\boldsymbol{x})$, which are functions of a vector of decision variables $\boldsymbol{x} = [x_1, x_2, \ldots, x_a]$. Suppose, without loss of generality, that all the objective functions are all to be minimised. Then an MOO problem may be formulated as

$$\text{minimise } \boldsymbol{f}(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_d(\boldsymbol{x})] \tag{10.1}$$

subject to the constraints

$$g_i(\boldsymbol{x}) \leq G_i, \quad i = 1, \ldots, u, \tag{10.2}$$

$$h_j(\boldsymbol{x}) = H_j, \quad j = 1, \ldots, v, \tag{10.3}$$

$$\boldsymbol{x} \in \mathbb{R}^a, \tag{10.4}$$

where $g_1(\boldsymbol{x}), \ldots, g_u(\boldsymbol{x})$ are the so-called inequality constraint functions and $h_1(\boldsymbol{x}), \ldots, h_v(\boldsymbol{v})$ are the equality constraint functions. Furthermore, $G_1, \ldots, G_u$ and $H_1, \ldots, H_v$ are assumed to be limiting values for the constraint functions. The set of all feasible decision vectors $\boldsymbol{x}$ form the so-called *decision space* of the problem, denoted by $\mathcal{X}$.

MOO techniques are employed in cases where the objective functions are conflicting, in which case a set of trade-off solutions is sought. This naturally leads to the notion of Pareto optimality as a result of the fact that there is typically no single solution $\boldsymbol{x}^*$ that minimises all the conflicting objective functions in (10.1) simultaneously. A feasible decision vector $\boldsymbol{x} \in \mathcal{X}$ *dominates* another decision vector $\boldsymbol{y} \in \mathcal{X}$, denoted by $\boldsymbol{x} \prec \boldsymbol{y}$, if $f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{y})$ for all $i \in \{1, \ldots, d\}$ and there exists at least one $i^* \in \{1, \ldots, d\}$ such that $f_{i^*}(\boldsymbol{x}) < f_i(\boldsymbol{y})$ [405].

A solution is said to be *globally non-dominated* or *Pareto optimal*, if no other feasible solution dominates it. The solutions in the Pareto optimal set $\mathcal{P}_s$ produce a set of objective function vectors, known as the *Pareto front* $\mathcal{P}_f$, that is $\mathcal{P}_f = \{\boldsymbol{f}(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{P}_s\}$.

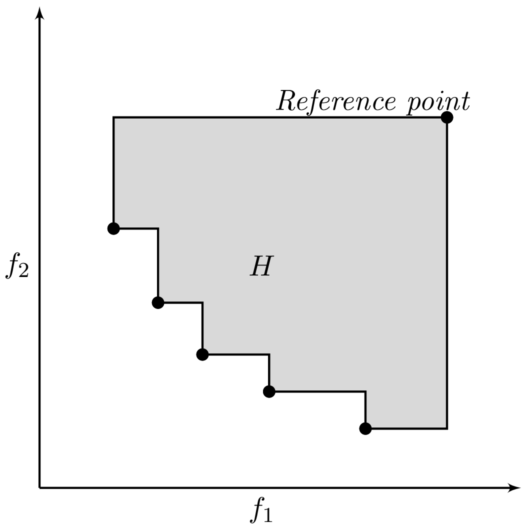
10.2 Performance measures for MOO algorithms

It is naturally difficult to measure and compare the quality of different Pareto front approximations as the Pareto optimal set is often not known. Measuring the performance of an MOO algorithm is difficult under these conditions [327]. Typical performance criteria for MOO algorithms are [300]:

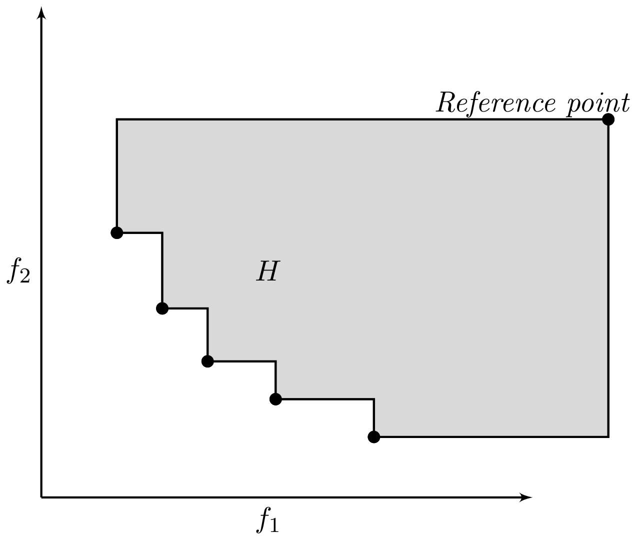
- *Accuracy*, which measures how close the non-dominated solutions generated are to the best-known prediction of the Pareto optimal set,
- *coverage*, which measures how many different non-dominated solutions are generated and how well they are distributed along the non-dominated front, and
- *the variance of each objective*, which measures the range of the non-dominated front along each axis in objective space.

Comparisons of the non-dominated solutions returned by two different MOO algorithms are not straightforward, as their representations in objective space may be incomparable [314]. Recent studies like those of Zitzler *et al.* [471], Paquete [336] and Fonseca *et al.* [168] are examples of the considerable effort expended to develop the necessary tools for better evaluation and comparison of MOO algorithms. Fonseca *et al.* [168] suggested three approaches toward evaluating Pareto front approximations. The first such method is to rank the Pareto front approximations according to the number of times the resulting Pareto front approximations dominate each other. The second approach is based on empirical attainment functions [314]. Attainment functions produce, with respect to the objective function space, the relative frequency by which each region is dominated by the approximation set produced by the algorithm. Finally, the third approach is to employ quality indicators. These quality indicators are typically either the hypervolume indicator [470], the R2 indicator [220] or the epsilon indicator [471]. The MACO algorithm designed in this chapter for solving the TVRPGC model of Chapter 9 will be evaluated using the hypervolume and unary-epsilon indicators because, according to Fonseca *et al.* [168], both these indicators are Pareto-compliant and represent the state of the art as far as indicators are concerned.

The hypervolume indicator [448, 469], also known in the literature as the *hyperarea metric*, the *S-metric*, or the *Lebesgue measure*, is denoted here by H. It measures the hypervolume in objective



(a) *Good reference point for calculating H*



(b) *Poor reference point for calculating H*

FIGURE 10.1: *The hypervolume H of a non-dominated front consisting of five points, indicated by the surface area of the shaded region. Both the objective functions f_1 and f_2 are to be minimised.*

space that is dominated by solutions in the non-dominated front with respect to a pre-selected, fixed vector in objective space, known as the *reference point*. This point has to be selected so that it is dominated by all solutions in the non-dominated front under consideration [469]. The hypervolume measure should be maximised by any MOO algorithm. Hypervolume is a popular measure of the quality of an approximately Pareto optimal set as well as, to a certain extent, of the spread of solutions within a non-dominated set across the objective space [448]. In bi-objective function space, the hypervolume is merely the area enclosed between the approximate Pareto front and the reference point, as illustrated in Figure 10.1. The hypervolume measure, however, has some disadvantages associated with its use, three of the main disadvantages being that it is sensitive to the relative scaling of the objective functions, that it is sensitive to the presence or absence of extremal points in the non-dominated front and that it is sensitive to the choice of the reference point [448]. Another disadvantage is that the results obtained by the hypervolume are biased towards the knee regions[1] of the Pareto front [17].

In order to highlight the challenge of selecting a good reference point so as not to bias the hypervolume indicator, consider the bi-objective optimisation problem example associated with Figure 10.1, and assume that the entries in objective space have been normalised. The reference point shown in Figure 10.1(b) is a poor choice, as it is visually apparent that the hypervolume is significantly more sensitive to changes in the values of the first objective function (f_1) than those of the second objective function (f_2). The reference point in Figure 10.1(a) would be a better, less biased choice.

The hypervolume is so far the only known indicator for MOO algorithm comparison which fulfils the property of strict monotonicity[2] [60]. The time required to compute the hypervolume indicator, however, grows exponentially with respect to an increase in the number of objectives d in (10.1).

The epsilon performance metric, proposed by Kollat and Reed [261], assigns a measure of per-

[1]Knee regions are potential parts of the Pareto front presenting maximal trade-offs between objectives [40].

[2]An indicator value $I(A)$ of a set A that dominates a set B, has to be larger than the indicator value $I(B)$ for the set B, assuming the indicator is to be maximised, in order for the indicator I to be classified as monotonic.

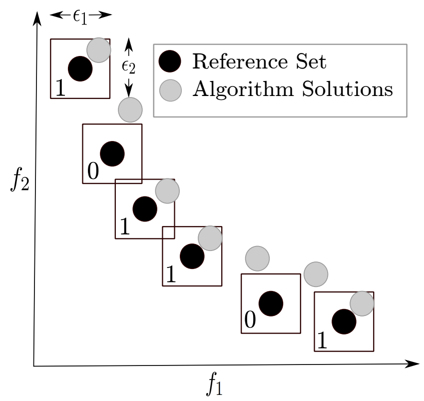


FIGURE 10.2: *Calculation of the epsilon indicator.*

formance by accounting for the proportion of solutions that fall within a user-specified distance ϵ from a reference set. Reference set solutions with matching approximation solutions[3] receive a score of 1, while those with no matching solutions receive a score of zero. This notion is illustrated in the context of a bi-objective MOO problem in Figure 10.2. The epsilon indicator value for the example described in Figure 10.2 is $\epsilon = \frac{4}{6} = 0.667$.

The epsilon indicator requires the definition of the parameters ϵ_1 and ϵ_2 by the analyst. The unary-ϵ indicator proposed by Zitzler [471], however, represents the factor by which an approximation set is worse than another with respect to all objectives (*i.e.* the smallest distance over which an approximation set must be translated in order to completely dominate the reference set), and does not require the selection of parameter values. For this reason, the unary-ϵ indicator was selected as the second performance measure employed in this dissertation. Smaller values of the unary-ϵ indicator correspond to better Pareto-approximation sets.

The unary-ϵ may be defined for a Pareto-approximation set $\mathcal{A}$ and a reference set $\mathcal{B}$ as

$$\text{unary-}\epsilon(\mathcal{A},\mathcal{B}) = \max_{\boldsymbol{z}^2\in\mathcal{B}} \min_{\boldsymbol{z}^1\in\mathcal{A}} \max_{1\leq i\leq d} \frac{z_i^1}{z_i^2}, \tag{10.5}$$

where $\boldsymbol{z}^1 = [z_1^1, \ldots, z_d^1]$ and $\boldsymbol{z}^2 = [z_2^1, \ldots, z_d^2]$ are objective function value vectors in $\mathcal{A}$ and $\mathcal{B}$, respectively.

10.3 Motivation for solution methodology chosen

ACO techniques in the literature have achieved good results, especially in the context of VRPs [140]. There have been numerous implementations of MACO algorithms for solving MOO problems in the literature [132, 133, 204], with a general taxonomy of these algorithms suggested by García-Martínez *et al.* [176]. The taxonomy suggested in [176] partitions these algorithms according to the number of pheromone trails and heuristic matrices employed, respectively. An empirical analysis was performed on all the different categories of MACO algorithms classified by García-Martínez *et al.* [176] in respect of several well-known TSP benchmark instances. This analysis showed that these MACO algorithms outperformed multi-objective genetic algorithms.

[3]Different approximations of the Pareto optimal set typically discover different points and different numbers of points. A matching approximation solution refers to the point within an approximation front that is the closest to the reference front point under consideration.

Thirteen algorithms were compared in the study, revealing that each algorithm performed well in certain aspects and poorer in others. The use of a single heuristic matrix and a single pheromone trail were reported to produce very good compromise solutions, but these algorithms mostly ignore the extremal points along the Pareto front, while the use of two heuristic matrices and two pheromone trails produce results that typically contain extremal points of higher quality, but achieve poor coverage along the rest of the non-dominated front.

Based on these findings, it was decided to implement a hybrid MACO algorithm in this dissertation for solving the TVRPGC model of Chapter 9. The algorithm employs three independent ant colonies, with two of the three ant colonies each focusing on a specific objective (as the minimisation of the number of vehicles is not modelled *explicitly* in the algorithm), while the third colony employs a weighted heuristic matrix and pheromone trail. The two independent ant colonies that focus solely on two of the three objectives of the TVRPGC, respectively, effectively function as a regular ACO, but they all communicate globally to form the non-dominated front.

The constructive nature of ACO algorithms was also a defining feature in the selection of this methodology as a solution approach in this study, as initially it was expected to assist in the construction of feasible solutions in terms of the sequence of facility tiers visited. During implementation of the algorithm it was, however, discovered that this method generally yields poor-quality solutions in its standard form since the solution space of the TVRPGC is tightly constrained, as was demonstrated in Chapter 9. Effective exploration of the solution space therefore requires inclusion of infeasible solutions by incorporation of a penalty function into the model.

It was subsequently decided to implement the multiplicative penalty function developed by Schlünz *et al.* [389] so as to improve the algorithm's exploration capabilities. Each constraint violation incurs a penalty value related to the magnitude of that violation. The total scaled constraint violation associated with the inequality constraint functions $g_1(\boldsymbol{x}), \ldots, g_u(\boldsymbol{x})$ in (10.2) is formed by aggregating these penalty values. The aggregation is given by

$$\mathcal{G}(\boldsymbol{x}) = \sum_{i=1}^{u} \max\left\{0, \frac{g_i(\boldsymbol{x}) - G_i}{G_i}\right\}. \tag{10.6}$$

Similarly,

$$\mathcal{H}(\boldsymbol{x}) = \sum_{j=1}^{v} \left|\frac{h_j(\boldsymbol{x}) - H_j}{H_j}\right| \tag{10.7}$$

represents the total scaled constraint violation associated with the equality constraint functions $h_1(\boldsymbol{x}), \ldots, h_v(\boldsymbol{x})$ in (10.3). The penalty function included in the model employs a severity factor γ as a free parameter whose value is typically determined empirically. The overall penalty function is defined as

$$\phi(\boldsymbol{x}) = \exp^{\gamma(\mathcal{G}(\boldsymbol{x})+\mathcal{H}(\boldsymbol{x}))} \tag{10.8}$$

and is incorporated into the objective functions by

$$\text{minimising } \boldsymbol{f}(\boldsymbol{x}) = \phi(\boldsymbol{x})[f_1(\boldsymbol{x}), \ldots, f_d(\boldsymbol{x})] \tag{10.9}$$

instead of (10.1). This multiplicative penalty function incorporates only one parameter whose value must be selected judiciously, namely the severity factor γ. In other constraint handling techniques documented in the literature, each class of constraints typically has its own severity factor — a situation which therefore requires considerably more parameter fine-tuning.

Raquel and Naval [360] suggested incorporation of diversity preservation within the population of solutions through the use of the *crowding-distance*, a notion borrowed from the well-known

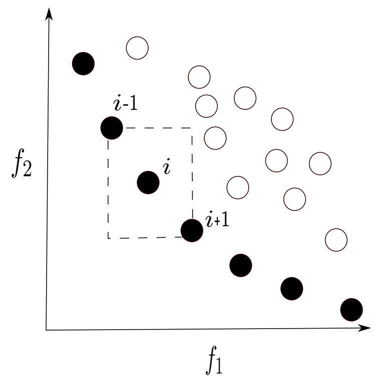


FIGURE 10.3: *Calculation of crowding-distance for Pareto front point i.*

NSGA-II algorithm [116]. This crowding-distance computation requires sorting of the population according to each objective function value in ascending order of magnitude. Thereafter, the extremal solutions are assigned an infinite distance value, while all other intermediate solutions along the non-dominated front are assigned a distance value equal to the absolute normalised difference in the objective function values of two adjacent solutions [116]. This process is illustrated in Figure 10.3 within the context of a bi-objective minimisation problem.

The notion of crowding-distance is incorporated into the MACO algorithm employed in this dissertation in the form of a weighted pheromone global update mechanism, with arcs that result in solutions with a larger crowding-distance receiving a stronger pheromone trail than arcs that produce solutions with a smaller crowding-distance. The extremal points of the archive are removed from consideration when implementing this mechanism.

10.4 The multi-objective ant colony optimisation algorithm

The underlying algorithmic approach was to initially incorporate a single-objective ACS, as described in §6.2, to construct initial routes based on the colony heuristic and pheromone matrices. The local pheromone update mechanisms are subsequently performed for each colony's pheromone. The sequence infeasibilities with respect to higher tier visitation of the constructed routes are then identified and corrected using several tailored approaches specific to the TVRPGC adopting a top-down paradigm. The routes are collectively examined and the respective penalty weighting applied with respect to the remaining constraints of §9.3.4, after which a global ranking is determined based on the NSGA-II ranking component complemented by a crowding distance function. Finally, a global pheromone update mechanism is applied to the respective pheromone matrices. The aforementioned mechanisms will be elaborated upon in the remainder of this section.

The initial construction phase of the MACO adopts the ACS described in §6.2 to generate initial routes. There are, however, more than one depot under consideration. The route construction begins through probabilistically selecting a depot based on the distance between the depot and the respective nearest customer. The probabilistic selection rule is derived from the roulette wheel mechanism (see §6.1.3).

The major component in any ACO algorithm is the pheromone update mechanism, elaborated upon in §6.2, and the selection of the next vertex to add during route construction. The method employed in the single-objective ACS of §6.2 produced solutions of a high quality and a similar approach was thus adopted for the route construction and local pheromone update mechanism phase in the context of the TVRPGC, as described in 6.2.1 and §6.2.2, respectively. The vertices visited are selected in a similar manner as described in 6.2.1, although, there are no capacity restrictions for the TVRPGC. The maximum driver autonomy μ is rather utilised as a stopping criterion, as opposed to the capacity constraint, during route construction. In an attempt to allow for more flexibility during the correction phase of the algorithm, a random number r_a is generated in the range $[\ell_b, 1]$. The new maximum driver autonomy μ' is determined as $\mu' = \mu r_a$, where $\ell_b \in (0, 1)$ is a predefined parameter. The adjusted driver autonomy value μ' is uniquely determined for each vehicle utilised and incorporated as the stopping criterion during the initial route construction phase of the MACO. The local pheromone update mechanism of §6.2.2 is adopted, although the update mechanism is only applied to the pheromone matrix under consideration.

A global pheromone update is applied to each of the colonies in a slightly different manner. The update for the colony focusing on minimising the total travel time is performed only in respect of the best solutions in terms of the shortest total travel time and employs the same global update mechanism of §6.2.2, adopting an elitist approach. The global pheromone update for the colony focusing on minimising maximum driver autonomy occurs in a similar manner, although only the best solutions with respect to minimisation of driver autonomy are considered.

The global pheromone update for the colony aiming to discover compromise solutions is designed to improve solution diversity with a view to encourage exploration and discovery of multiple compromise solutions. The first mechanism employed is a dynamic archive of best solutions found. The dynamism of the archive refers not only to the storing of the non-dominated solutions, but also to the storing of solutions that are dominated by e other solutions, with the value of e tending towards zero as the algorithm execution progresses. The calculation of the crowding distance c_i, as described in §10.3, is required for each solution i in the archive, excluding the extremal solutions. The crowding factor for solution i is determined by

$$F_i = \frac{c_i - \bar{c}}{\bar{c}_i}, \tag{10.10}$$

where $\bar{c}$ is the average crowding distance of all the solutions in the archive (excluding the extremal solutions). Each solution s in the archive is ranked twice according to the overall travel time and the maximum driver autonomy, with the crowding factor incorporated into the ranking. The two mixture pheromone matrices, the overall travel time pheromone matrix $\boldsymbol{\tau}_a$ and the driver autonomy pheromone matrix $\boldsymbol{\tau}_b$, are updated by the global pheromone update mechanism of §6.2.2 based on their respective weightings. The overall pheromone matrix for colonies aiming to discover compromise solutions is calculated by applying the substitution

$$\boldsymbol{\tau} = \alpha_r \boldsymbol{\tau}_a + (1 - \alpha_r)\boldsymbol{\tau}_b, \tag{10.11}$$

where α_r is a randomly generated number in the range $[0.25, 0.75]$ in an attempt to bias the compromise solution search into unexplored areas within the solution space.

The heuristic matrix, $\boldsymbol{\eta} = [\eta_{ij}]_{i,j \in \mathcal{V}}$ for the colony focusing on minimising the total travel time is simply calculated as the inverse of the travel time for the respective arc under consideration.

The determination of the heuristic matrix for the colony that aims to minimise the maximum travel time of all vehicles is slightly more complicated. The constructive nature of ACS algorithms does not allow for the fitness evaluation of the maximum driver autonomy as it is not able to predict the outcome of adding a vertex to a route in respect of the vehicles' route travel times. This problem is remedied by generating an initial population of routes that are feasible in terms of driver autonomy, but not in terms of the increasing tier visitation requirement of the TVRPGC. This initial population is then used to determine the heuristic value for the individual arcs, as described in Algorithm 10.1. Each of the routes generated in the initial population is ranked according to its travel time and the arcs constituting the routes are assigned weighted values according to the ranking of the route. The distances between the respective facilities are sorted and the corresponding arcs in the heuristic matrix are assigned distance values based on their weighted values previously assigned. The heuristic value is then taken as the inverse of the distance value assigned to it so that the heuristic matrices employed by the different colonies are of the same order of magnitude.

Algorithm 10.1: Heuristic determination for driver autonomy

```
Input  : Initial population generated randomly, locations of the customers, distance matrix
         of arcs between customers
Output: Heuristic matrix for route length
for i ← 1 to length(population) do
    for j ← 1 to num.routes(population[i]) do
        route.distance[length(route.distance) + 1] = distance(population[i, j]);
route.distances = sort(route.distances, ascending);
for i ← 1 to length(route.distances) do
    route = which(distance(population) == route.distance[i]);
    for j ← 1 to length(route) − 1 do
        heursitic.arc[route[j], route[j + 1]] = length(route.distance) − i;
unique.entries = unique(heuristic.arc);
perc = percentiles(unique(distance.matrix), unique.entries);
for i ← 1 to length(unique.entries) do
    heuristic.arc[heuristic.arc == unique.entries[i]] = 1/perc[i];
Return heuristic.arc
```

The heuristic matrix for the colony searching for compromise solutions is simply taken as a mixture of the two previously described heuristics, by weighting each matrix equally to create a single matrix. The route construction process employs the same probabilistic rule with respect to node selection as that described in §6.2.1.

The algorithmic implementation incorporates a function aimed at fixing the sequence in which vehicles visit the customers (*i.e.* ensuring that a customer is visited by a vehicle that later visits a facility of a strictly higher tier or another facility visited by a vehicle that later visits a facility of a strictly higher tier). As previously mentioned, the sequence fix function adopts a top-down approach to rectifying the visitation sequence of customers. The first stage is to determine that at least one route ends at a facility of the highest tier. If the routes generated do not contain such a sequence, a facility of the highest tier is inserted into one of the routes based on the roulette wheel mechanism, described in §6.1.3, with the probabilities calculated based on the insertion costs with respect to travel time.

The middle-tier sequence fix algorithm then considers all infeasible facilities $i \in \mathcal{F} \setminus (\mathcal{F}^0 \cup \mathcal{F}^f)$ with respect to customer sequence visitation. The middle-tier sequence infeasibilities are rectified according to four paradigms incorporated within a probabilistic sequence fix function. A pseudocode description of this function is given in Algorithm 10.2.

The first paradigm, *insert.before*, removes those customers that are not visited by a vehicle that later visits a facility of a higher tier and places them in a different route in which this requirement is indeed satisfied. The route and position is determined from a pool of candidates through a weighted probability function. The second paradigm, *add.higher.after*, adds a facility of a higher tier to the route at a later stage, with the position of this facility insertion determined probabilistically, based on insertion cost with respect to travel time. The third paradigm,

Algorithm 10.2: Middle-tier sequence fix

Input : Candidate routes, locations of the customers together with their respective tiers
Output: Candidate routes with the tier visitation sequence fixed

```
infeasibilities = determine.sequence.infeasibilities(routes, locations, tier);
for i ← 1 to length(infeasibilities) do
    new.route[1] = insert.before(routes, infeasibilities[i]);
    new.route[2] = add.higher.after(routes, infeasibilities[i]);
    new.route[3] = cross.docking(routes, infeasibilities[i]);
    new.route[4] = higher.tier.end(routes, infeasibilities[i]);
    total.travels = travels(new.route, locations);
    random = runif(1,0,1);
    if random ≤ best.select then
        routes = new.route[which.min(total.travels);
    else
        insert = roulette(total.travel);
        routes = new.route[insert];
Return routes
```

cross.docking, encourages the facilitation of global cross-docking. According to this paradigm, all customers are determined who are visited by a vehicle that later visits a facility of the appropriate tiers. Customers are selected from this set based on a weighted probability function biased towards lower insertion costs, and they are inserted at a later stage in the infeasible route. The final paradigm, *higher.tier.end*, simply assesses each infeasible route and adds an appropriately tiered facility at the end of the route. The algorithm functions based on a randomly generated number. If the random number is smaller than a pre-defined threshold, the repair paradigm associated with the lowest travel time is selected. Otherwise, a paradigm is selected by means of the roulette wheel mechanism.

The final phase of the sequence fix function is to rectify all sequence infeasibilities of facilities of the lowest tier. The lowest-tier fix function similarly incorporates two paradigms, namely to move an infeasible facility to a different route resulting in feasibility (based on insertion cost) or to add a higher tier at a later stage within the route. The algorithm is biased towards adding a higher tier at a later stage if there are numerous infeasible facilities within the route; otherwise, a paradigm is selected according to the roulette wheel mechanism.

The aforementioned sequence fix function is rather disruptive to the qualities of vehicle routes, resulting in sub-optimal facility sequence visitation within the routes. Accordingly, a probabilistic heuristic is applied to the routes once all the sequence infeasibilities have been rectified. There are three simple paradigms that are employed in an attempt to improve the solution quality. The first paradigm determines whether there is cross-docking present within the route. If cross-docking indeed occurs, then the facility at which consolidation occurs is swapped with a facility of the same tier within the route. The selected facility serves as the new consolidation point and a 2-opt mechanism is performed on the route, keeping the selected facility as the consolidation point. This also involves switching the original consolidation facility with the new facility in all routes that contain the original consolidation facility. A 2-opt mechanism that adheres to the sequence visitation feasibility is applied and if there is an improvement in solution quality, the proposed swap is adopted. It may, however, happen that only one objective function value is improved while the other objective function value decreases with respect to solution quality. In this instance, both solutions are stored in the archive. The second paradigm, involves simply applying a 2-opt mechanism to the route under consideration while still respecting the sequence visitation constraint. Finally, the third paradigm involves determining which depot is best to

Algorithm 10.3: MACO algorithm

Input : Locations of customers, number of ants employed, mean and standard deviation for normal distribution, maximum autonomy level for all routes

Output: Non-dominated front of solutions

```
population = initialpopulation(locations, autonomy);
heuristic.distance = initial.heuristic.dist(locations, population);
heuristic.arcs = initial.heuristic.arcs(locations, population);
pheromone.dist = pher.distance(locations);
pheromone.arcs = pher.arcs(locations, population);
pheromone.mixture = 0.5 × pheromon.arcs + 0.5 × pheromone.dist;
for i ← 1 to iterations do
    for cycle ← 1 to 3 do
        if cycle = 1 then
            pheromone = pheromone.dist;
            heuristic = heuristic.dist;
        else if cycle = 2 then
            pheromone = pheromone.arcs ;
            heuristic = heuristic.arc;
        else
            pheromone = pheromone.mixture ;
            heuristic = heuristic.mixture;
        for m ← 1 to no of ants do
            route = antcolony(locations, autonomy, timewindows);
            route = sequence.fix(route);
            route = post.optimisation(route) population = population + route;
        for d ← 1 to length(population) do
            a = autonomy.penalty(population[d]) × sequence.penalty(population[d]);
            x[d] = total.distance(population[d]) ×a;
            y[d] = arc.distance(population[d]) ×a;
        ranking = NSGA2.ranking((x, y));
        range = (max(x) − min(x), max(y) − min(y));
        front = population[ranking <= pop.keep];
        crowding = crowdingdistance(front, cycle);
        pheromone = globalpheromone(front, cycle, pheromone, crowding);
        population = front;
Return remove.infeasibilities(population)
```

act as the home depot for the vehicle under consideration. The three paradigms are selected probabilistically by generating a random number and employing the roulette wheel mechanism based on equal proportions of each paradigm being selected.

A pseudocode description of the entire MACO algorithm is provided in Algorithm 10.3, which illustrates all the relevant components of the algorithm. The time-window penalty factor for solution i in the archive is calculated, if a vehicle arrives at a time outside the relevant time window, as

$$s(i) = \max\left\{\left|\frac{a_i - T_{ik}}{a_i}\right|, \left|\frac{g_i - T_{ik}}{g_i}\right|\right\} + 1, \tag{10.12}$$

where T_{ik} is the arrival time of the vehicle, a_i the earliest possible arrival time of a vehicle at facility i and g_i the latest possible arrival time of a vehicle at that facility. The autonomy penalty factor for solution j within the archive is

$$a(j) = \max\left\{1, \frac{L_{mat} - \mu}{\mu}\right\}, \tag{10.13}$$

where L_{mat} is the maximum travel time of all the routes contained within the solution and μ is the maximum allowable driver autonomy level.

10.5 Chapter summary

Various basic notions related to MOO problems were described in §10.1, including a general formulation of an MOO problem. Performance measures that are typically employed in the literature to analyse Pareto front approximations were discussed briefly in §10.2. The discussion focused mainly on the two performance measures that will be used in the following chapter to compare different non-dominated fronts within the context of the TVRPGC, namely the hypervolume indicator and the R2 indicator. The MACO algorithm designed to solve the TVRPGC approximately is based on several influences from the literature. Section 10.3 contained a discussion on and motivations for the design of the MACO algorithm, together with an elaboration of its algorithmic components not previously discussed in this dissertation. The chapter closed with a more thorough discussion in §10.4 of the MACO algorithm employed later in this dissertation, with key aspects of the algorithm being highlighted by means of pseudocode descriptions.

CHAPTER 11

Parameter Evaluation for the MACO Algorithm

Contents

The MACO algorithm proposed in Chapter 10 incorporates several free parameters, most of which arise due to the novelty of the TVRPGC and limited guidelines offered with respect to effective algorithmic design in the literature. The parameters employed in the MACO algorithm require a sensitivity analysis in order to determine the best configuration of parameter values. The experimental design of this sensitivity analysis is described in §11.1. The MACO algorithm is applied to a real-life regional instance of the TVRPGC, and the parameter sensitivity analysis results for this instance are presented in §11.2. A clustering phase is also applied as part of the solution procedure in order to compare the performance of the MACO algorithm in respect of solution quality and computational expense when applied to unclustered data and to clustered data. The numerical results thus obtained are presented in §11.4. The numerical results are discussed briefly in §11.5 and the chapter closes in §11.6 with a brief summary of its content.

11.1 Experimental design

The design process of the MACO algorithm for the TVRPGC was an uncertain endeavour as there was very little guidance available in this respect within the literature. This uncertainty was mitigated by the inclusion of several algorithmic parameters. Various combinations of values for these parameters are evaluated and compared (in terms of solution quality returned) in this chapter according to the hypervolume indicator and the unary-ϵ indicator, as described in §10.2.

The parameters employed in the MACO algorithm of Chapter 10 are evaluated in this chapter in respect of the algorithm's resulting performance within the context of a realistic regional

TVRPGC instance. The instance is related to the operations of a national pathology healthcare service provider within the George region (located in the South African Southern Cape). There are sixty-seven facilities and one depot in this instance. The facilities are segregated according to their pathology processing capabilities in the following manner: one facility of tier 3, one facility of tier 2, four facilities of tier 1 and sixty one facilities of tier zero. The geographic distribution of the facilities is depicted in Figure 11.1 and the coordinates of the facilities are presented in Table 11.1. The expected travel times between the respective facilities was generated by means of OSRM [333] in order to reflect realistic travel times that consider the type of road and typical traffic congestion. The travel time matrix is available online [403].

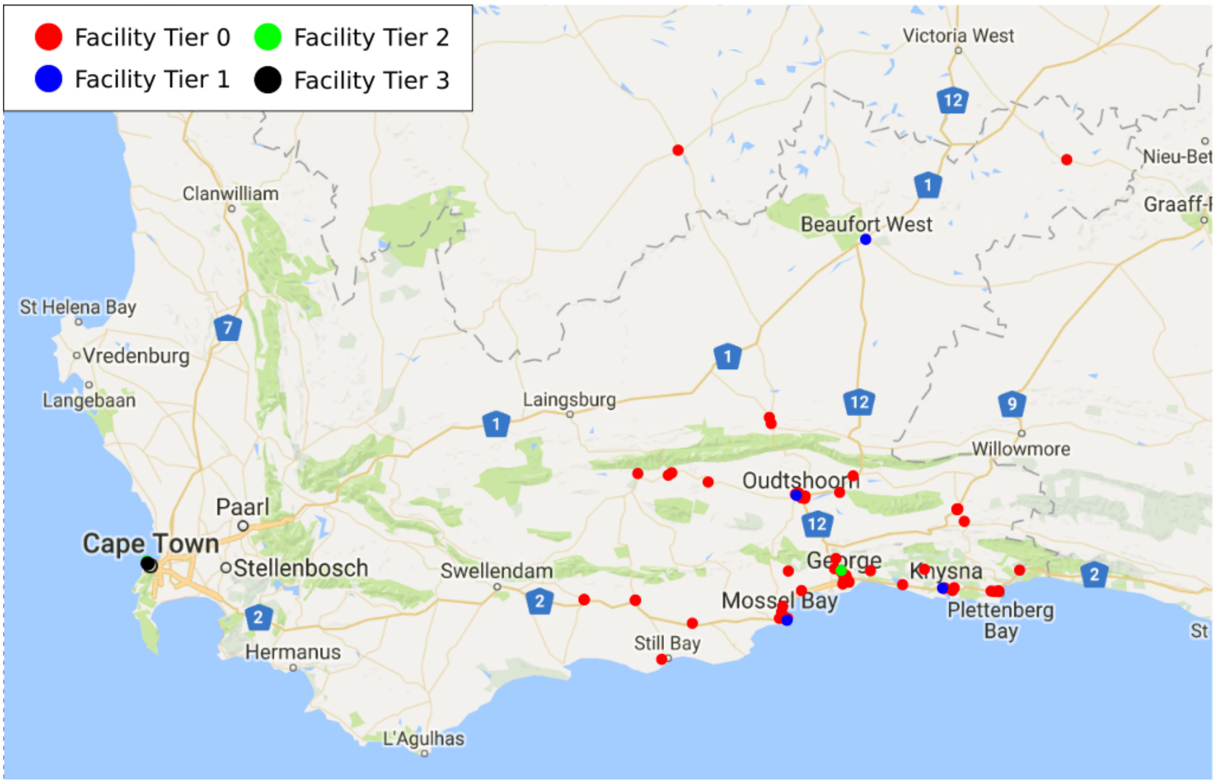


FIGURE 11.1: *The locations of facilities in the George TVRPGC instance.*

The TVRPG model of §9.3 was solved thirty times in respect of the George instance for a number of parameter value configurations of the MACO algorithm. The best, average and worst objective function values thus obtained were recorded for each parameter configuration. The MACO algorithm was allowed to run for a 1 000 iterations in each case.

All the numerical work reported in this chapter was carried out on an i7-4770 processor with a memory limit of 8GB running at 3.40 GHz within the Windows 7 operating system after having implemented the algorithm of Chapter 10 in R. Due to considerable computation time requirements of the experiments, the parameter sensitivity analysis described above was conducted in parallel on several computers with the processing capabilities mentioned above, with multiple R sessions running concurrently on each computer. This led to the decision not to record the computation time as several external factors could have affected the execution rate of the algorithm on the various machines. When incorporating a clustering component, however, the performance of the MACO algorithm was investigated on the same computer within a single session of R and the time was recorded in this case as most of the external factors that could have affected the variation in computation time were thus removed.

TABLE 11.1: *Coordinates and respective processing capabilities of facilities in the George TVRPGC instance.*

Name	Longitude	Latitude	Tier	Name	Longitude	Latitude	Tier
Facility 1	23.3222	−34.0505	0	Facility 2	23.0479	−34.0385	0
Facility 3	23.0536	−34.0351	0	Facility 4	23.1078	−34.0345	0
Facility 5	23.0981	−34.0477	0	Facility 6	23.3650	−34.0531	0
Facility 7	23.4885	−33.9503	0	Facility 8	23.3667	−34.0500	0
Facility 9	23.3426	−34.0524	0	Facility 10	23.3667	−34.0500	0
Facility 11	22.9346	−33.9460	0	Facility 12	22.8084	−34.0197	0
Facility 13	22.6213	−33.9533	0	Facility 14	22.4892	−34.0050	0
Facility 15	22.4941	−34.0068	0	Facility 16	22.4969	−34.0078	0
Facility 17	22.4790	−33.9750	0	Facility 18	22.4518	−33.9590	0
Facility 19	22.4125	−33.9432	0	Facility 20	22.4511	−33.9509	0
Facility 21	22.4603	−34.0167	0	Facility 22	22.4923	−34.0031	0
Facility 23	22.4842	−33.9931	0	Facility 24	22.4736	−33.9912	0
Facility 25	20.9565	−34.0907	0	Facility 26	20.9579	−34.0915	0
Facility 27	21.2550	−34.0937	0	Facility 28	21.2548	−34.0947	0
Facility 29	21.5859	−34.2041	0	Facility 30	21.4079	−34.3759	0
Facility 31	22.2205	−34.0483	0	Facility 32	22.1440	−33.9547	0
Facility 33	23.1251	−33.6574	0	Facility 34	23.1250	−33.6595	0
Facility 35	23.1295	−33.6565	0	Facility 36	23.1253	−33.6606	0
Facility 37	23.1667	−33.7167	0	Facility 38	22.4203	−33.8947	0
Facility 39	21.5030	−31.9178	0	Facility 40	23.7623	−31.9642	0
Facility 41	22.1291	−34.1865	0	Facility 42	22.1395	−34.1808	0
Facility 43	22.1106	−34.1234	0	Facility 44	22.0921	−34.1808	0
Facility 45	22.1161	−34.1766	0	Facility 46	22.1132	−34.1713	0
Facility 47	22.1051	−34.1496	0	Facility 48	22.0442	−33.2463	0
Facility 49	22.0333	−33.2167	0	Facility 50	22.5204	−33.4980	0
Facility 51	22.4421	−33.5773	0	Facility 52	22.2385	−33.6065	0
Facility 53	22.2416	−33.5953	0	Facility 54	22.2204	−33.6018	0
Facility 55	22.1849	−33.5868	0	Facility 56	22.1912	−33.5835	0
Facility 57	22.2038	−33.5798	0	Facility 58	21.2689	−33.4869	0
Facility 59	21.4461	−33.4948	0	Facility 60	21.4667	−33.4833	0
Facility 61	21.6780	−33.5276	0	Facility 62	23.0406	−34.0361	1
Facility 63	22.1363	−34.1896	1	Facility 64	22.5936	−32.3531	1
Facility 65	22.1898	−33.5914	1	Facility 66	22.4511	−33.9515	2
Facility 67	18.4158	−33.9067	3	Depot	22.4511	−33.9515	−1

Each of the algorithmic parameters were set to a base configuration value for all of the parameter sensitivity analysis runs, except for the value of the parameter under investigation which was altered within a pre-defined set. The base configuration values for all parameters are shown in Table 11.2.

TABLE 11.2: *Base configuration values for the parameters employed in the MACO algorithm for solving instances of the TVRPGC.*

Parameter	α	ants	best-select	β	δ	diversity	γ	ℓ_b	ρ
Value	1	20	0.8	1	100	0.8	1	0.8	0.8

The hypervolume indicator requires the determination of a reference point in order to evaluate the quality of the non-dominated fronts obtained. This point was selected by combining all the non-dominated fronts under consideration and sorting the collective attainment front so as to yield a single non-dominated front representative of the entire sensitivity analysis. The extremal points of this attainment front were then employed to normalise all the points within the various non-dominated fronts under consideration in a multiplicative fashion to coordinates within the unit square in a rescaled objective function space with one of its corner-points at the origin. The hypervolume reference point was then taken as $(1.05, 1.05)$ for all experiments performed in this chapter. The unary-ϵ indicator also requires a reference front. This reference front was selected in a similar fashion.

11.2 Parameter sensitivity analysis for the MACO

The parameter α of the MACO algorithm determines the respective weighting of the pheromone trail value during the probabilistic selection of vertices during route construction, as in most single- or multi-objective implementations of the ACO algorithm. The results of the hypervolume indicator sensitivity analysis for this parameter are shown in Table 11.3. The table shows that values larger than 2 for the parameter α tend to typically produce poor-quality results. The value of 2 for the parameter α produced the second most consistent results in terms of hypervolume for the George TVRPGC instance. The consistent behaviour of the parameter value $\alpha = 2$ was corroborated by the results of the unary-ϵ parameter sensitivity analysis, the results of which are shown in Table 11.4.

TABLE 11.3: *Hypervolume sensitivity analysis for the parameter α in respect of the George TVRPGC instance.*

	α=1	α=2	α=5	α=10
Best	0.966	0.961	0.951	0.983
Mean	0.912	0.905	0.896	0.890
Worst	0.870	0.860	0.842	0.825

TABLE 11.4: *Unary-ϵ sensitivity analysis for the parameter α in respect of the George TVRPGC instance.*

	α=1	α=2	α=5	α=10
Best	0.061	0.076	0.081	0.047
Mean	0.109	0.103	0.107	0.107
Worst	0.146	0.143	0.136	0.147

The number of ants employed in the MACO algorithm determines how many different potential routes are built during each iteration. As previously mentioned in §7.3, the number of ants, also affect the global pheromone influence of the algorithm and determines its rate of convergence. Typically, a larger number of ants employed in any ACO algorithm results in better quality solutions, due to an increase in solution diversity, but employing a large number of ants is computationally expensive. The results of the hypervolume parameter sensitivity analysis (shown in Table 11.5) and the unary-ϵ parameter sensitivity test (shown in Table 11.6) confirm that a larger number of ants is beneficial in respect of solution quality, although it requires considerably more computation time.

Table 11.5: *Hypervolume sensitivity analysis for the ants parameter in respect of the George TVRPGC instance.*

	10	20	50	60
Best	0.966	0.961	0.951	0.983
Mean	0.912	0.905	0.896	0.890
Worst	0.870	0.860	0.842	0.825

Table 11.6: *Unary-ϵ sensitivity analysis for the ants parameter in respect of the George TVRPGC instance.*

	10	20	50	60
Best	0.222	0.137	0.180	0.127
Mean	0.247	0.181	0.199	0.194
Worst	0.259	0.239	0.241	0.238

The MACO algorithm designed in Chapter 10 employs a number of probabilistic rules throughout its execution. The first instance of probabilistic determination occurs during the route construction by any of the ants, where a probabilistic rule is utilised to determine the vertices to append to the route. If the random number generated according to a uniform distribution is below the best-select parameter value, then the vertex with the largest probability is selected; otherwise, the roulette wheel mechanism is employed. The best-select parameter also affects the sequence-fix function, as it determines which of the four routes proposed by the function *sequencefix* is implemented. The practice of including a best-select parameter is popular in the literature. It is, in fact, a key component in maintaining solution diversity. The results of the hypervolume parameter sensitivity analysis for the best-select parameter may be seen in Table 11.7 and the corresponding results for the unary-ϵ parameter sensitivity analysis are shown in Table 11.8. These results indicate that the best proposed candidate should always be selected for implementation, which is testament to the computational complexity of the TVRPGC as the algorithm performs significantly better when convergence is heavily favoured over employing diverse solution attainment strategies.

Table 11.7: *Hypervolume sensitivity analysis for the best-select parameter in respect of the George TVRPGC instance.*

	0.5	0.7	0.9	1
Best	0.911	0.937	0.942	0.957
Mean	0.869	0.892	0.907	0.919
Worst	0.842	0.861	0.859	0.863

Table 11.8: *Unary-ϵ sensitivity analysis for the best-select parameter in respect of the George TVRPGC instance.*

	0.5	0.7	0.9	1
Best	0.084	0.079	0.069	0.058
Mean	0.113	0.097	0.106	0.098
Worst	0.154	0.118	0.165	0.145

The parameter β has a very similar influence as the parameter α during execution of the algorithm. This parameter determines the influence of the heuristic component in the establishment of the probabilistic value used during route construction when selecting vertices to append to a route. A large value of β indicates that the function of the pheromone update mechanism is poorly modelled as it would undermine the exploration history of all the ants during the algorithm's execution. The results of the unary-ϵ sensitivity analysis for the parameter β (shown in Table 11.10) were inconsistent across the different test instances, but the results of the corresponding hypervolume parameter sensitivity analysis (shown in Table 11.10) were unanimously in favour of the value $\beta = 5$.

The parameter δ has the function described in §6.2.2 — it serves as an elitist-related parameter such that solutions of a higher quality have a larger pheromone deposit among the relevant arcs during the local pheromone update mechanism. Higher values indicate a preference towards convergence as opposed to diversity management throughout the search. As previously mentioned, the MACO typically favours parameter values that contribute towards convergence, possibly

TABLE 11.9: *Hypervolume sensitivity analysis for the parameter β in respect of the George TVRPGC instance.*

	β=1	β=2	β=5	β=10
Best	0.948	0.948	0.972	0.950
Mean	0.893	0.898	0.910	0.884
Worst	0.853	0.851	0.857	0.831

TABLE 11.10: *Unary-ϵ sensitivity analysis for the parameter β in respect of the George TVRPGC instance.*

	β=1	β=2	β=5	β=10
Best	0.071	0.069	0.049	0.061
Mean	0.116	0.119	0.119	0.132
Worst	0.189	0.159	0.177	0.177

due to the combinatorial complexity of the TVRPGC. This trend is further corroborated in Tables 11.11 and 11.12 in which the best value for the parameter δ is 300 according to both the hypervolume and unary-ϵ indicators, respectively.

TABLE 11.11: *Hypervolume sensitivity analysis for the parameter δ in respect of the George TVRPGC instance.*

	δ=50	δ=100	δ=200	δ=300
Best	0.974	0.967	0.991	0.997
Mean	0.914	0.911	0.923	0.932
Worst	0.862	0.863	0.869	0.865

TABLE 11.12: *Unary-ϵ sensitivity analysis for the parameter δ in respect of the George TVRPGC instance.*

	δ=50	δ=100	δ=200	δ=300
Best	0.069	0.070	0.059	0.059
Mean	0.113	0.118	0.109	0.110
Worst	0.189	0.159	0.177	0.177

The diversity parameter is used solely in the execution of the ant colony searching for compromise solutions. This diversity parameter determines the percentage of the archive for that specific colony that consists of solutions with a large crowding distance (*i.e.* diverse solutions) in an attempt to increase overall solution quality. The results of the hypervolume parameter sensitivity analysis for the diversity parameter are shown in Table 11.13, and the corresponding unary-ϵ indicator parameter sensitivity analysis results are shown in Table 11.14. The results of these analyses are surprising. The two best parameter settings represent two conflicting paradigms. The first paradigm favours convergence of the algorithm to the best solutions found, while the second paradigm favours the incorporation of diversity of the population over the deemed fitness of the population. The decision to select a value of 0.7 for the diversity parameter was based on the relative consistency of the parameter value over the thirty test runs. The decision to employ a relatively small value for the diversity parameter does not contradict the decision to always select the best candidate (as was determined by the best-select parameter), as the algorithm still converges faster for the two colonies that do not employ the diversity parameter.

TABLE 11.13: *Hypervolume sensitivity analysis for the diversity parameter in respect of the George TVRPGC instance.*

	0.6	0.7	0.8	0.9
Best	0.942	0.959	0.941	0.956
Mean	0.907	0.906	0.902	0.905
Worst	0.876	0.853	0.848	0.853

TABLE 11.14: *Unary-ϵ sensitivity analysis for the diversity parameter in respect of the George TVRPGC instance.*

	0.6	0.7	0.8	0.9
Best	0.070	0.056	0.071	0.073
Mean	0.102	0.105	0.107	0.101
Worst	0.166	0.136	0.202	0.146

The severity-factor parameter determines the magnitude of the penalty function implemented in the MACO algorithm. The results of the hypervolume and unary-ϵ indicator sensitivity analyses for this parameter are shown in Tables 11.15 and 11.16, respectively. These results suggest that a larger penalty function value aids the algorithm during its search through the solution space. The value of 1.5 for the severity-factor parameter is implemented in the remainder of this dissertation due to the solution space being tightly constrained. This large value would still allow for reasonable exploration of the algorithm through infeasible regions of the solution space while simultaneously encouraging the discovery of feasible solutions.

Table 11.15: *Hypervolume sensitivity analysis for the severity factor parameter in respect of the George TVRPGC instance.*

	γ=0.75	γ=1	γ=1.25	γ=1.5
Best	0.974	0.958	0.939	0.994
Mean	0.910	0.909	0.906	0.951
Worst	0.868	0.889	0.904	0.909

Table 11.16: *Hypervolume sensitivity analysis for the severity factor parameter in respect of the George TVRPGC instance.*

	γ=0.75	γ=1	γ=1.25	γ=1.5
Best	0.151	0.176	0.143	0.084
Mean	0.165	0.169	0.180	0.146
Worst	0.175	0.201	0.206	0.209

The parameter ℓ_b is employed to allow the MACO algorithm the necessary flexibility after the initial routes have been constructed in order to adhere to the tier-visitation constraint with respect to the driver autonomy. The results of the hypervolume and unary-ϵ indicator sensitivity analyses for this parameter may be seen in Tables 11.17 and 11.18, respectively. In order to satisfy the tier-visitation constraint, the initial routes require considerable correction and thus the MACO algorithm favours values for the parameter ℓ_b that allow for a greater degree of flexibility with respect to driver autonomy. This trend is corroborated by the results of both the hypervolume and unary-ϵ indicator sensitivity analyses as a value of 0.6 returned solutions of the highest quality.

Table 11.17: *Hypervolume sensitivity analysis for the parameter ℓ_b in respect of the George TVRPGC instance.*

	ℓ_b=0.6	ℓ_b=0.7	ℓ_b=0.8	ℓ_b=0.9
Best	0.986	0.957	0.949	0.949
Mean	0.955	0.928	0.897	0.875
Worst	0.927	0.894	0.860	0.811

Table 11.18: *Unary-ϵ analysis for the parameter ℓ_b in respect of the George TVRPGC instance.*

	ℓ_b=0.6	ℓ_b=0.7	ℓ_b=0.8	ℓ_b=0.9
Best	0.054	0.075	0.107	0.099
Mean	0.087	0.101	0.137	0.171
Worst	0.114	0.139	0.179	0.218

Finally, the parameter ρ is used in some manner in most ACO algorithms — it directly affects the pheromone update mechanism of the MACO algorithm. The parameter ρ was implemented in both the local and global pheromone update mechanisms, in which it affects the rate of decay within the local pheromone update mechanism and the rate of convergence within the global pheromone update mechanism. The results of the hypervolume parameter sensitivity analysis (shown in Table 11.19) and the unary-ϵ parameter sensitivity analysis (shown in Table 11.20) for this parameter suggest that a value of $\rho = 0.8$ returns better quality results. This is a relatively large value, but considering that numerous ants are implemented in each colony, it is logical that the pheromone trails should be altered in small increments.

Table 11.19: *Hypervolume sensitivity analysis for the parameter ρ in respect of the George TVRPGC instance.*

	ρ=0.6	ρ=0.7	ρ=0.8	ρ=0.9
Best	0.915	0.941	0.950	0.930
Mean	0.869	0.879	0.890	0.893
Worst	0.840	0.842	0.787	0.858

Table 11.20: *Unary-ϵ sensitivity analysis for the parameter ρ in respect of George TVRPGC instance.*

	ρ=0.6	ρ=0.7	ρ=0.8	ρ=0.9
Best	0.070	0.090	0.056	0.067
Mean	0.119	0.123	0.113	0.115
Worst	0.148	0.152	0.179	0.165

The best values of the parameters employed in the MACO algorithm (according to the sensitivity analysis results discussed in this section) are finally summarised in Table 11.21.

TABLE 11.21: *The parameter values employed in the MACO algorithm for solving realistic instances of the TVRPGC, based on the sensitivity analysis performed in this section.*

Parameter	α	ants	β	best-select	δ	diversity	γ	ℓ_b	ρ
Value	2	20	5	1	300	0.7	1.5	0.6	0.8

11.3 Numerical results for the George TVRPGC instance

Adopting the parameter values described in Table 11.21, the MACO algorithm was implemented on the same computer mentioned in §11.1 and was allowed 1 000 iterations in order to produce a non-dominated front for the George TVRPGC instance. The maximum driver autonomy was set to 1 100 time units which matched the maximum driver autonomy within the George TVRPGC instance. The bounds on the time window constraint were fixed in a manner so as not to affect the solutions generated by the MACO algorithm for the TVRPGC, because the data received from the industry partner did not contain any information pertaining to time windows of the facilities. The non-dominated front yielded by the MACO may be seen in objective function space in Figure 11.2. This front only contains feasible solutions (all infeasible solutions returned by the MACO were removed from consideration) and provides a reasonable trade-off between the total travel time and the driver autonomy. The vehicle routes corresponding to the eleven non-dominated solutions of Figure 11.2 are provided in Tables B.1–B.11 in Appendix B.

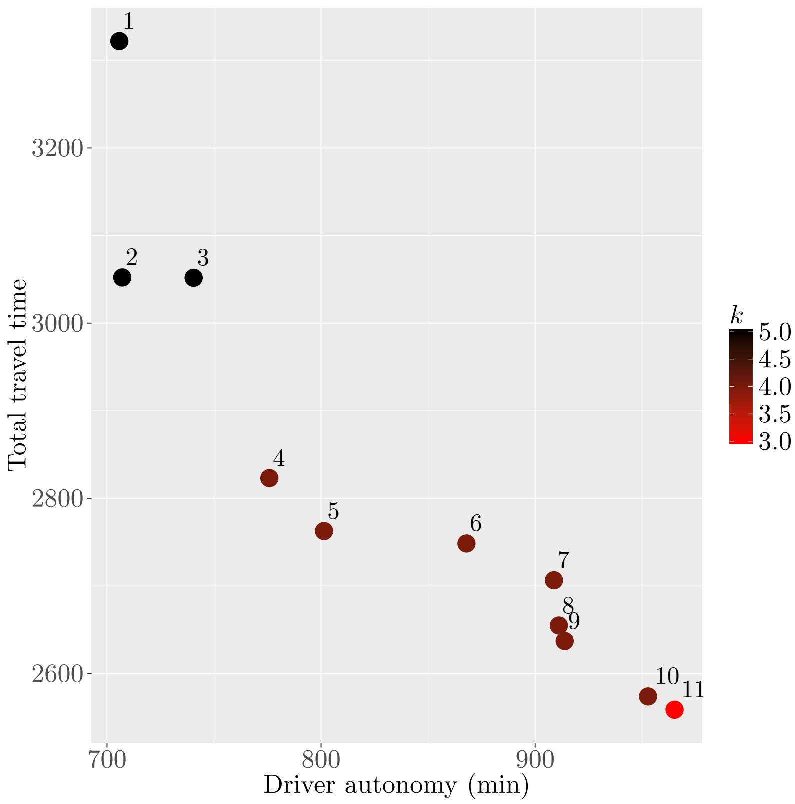


FIGURE 11.2: *Non-dominated front yielded by the MACO in respect of the George TVRPGC instance.*

The vehicle routes corresponding to Solution 11 in Figure 11.2 are shown in Figure 11.3 in order to elucidate the phenomenon of global cross-docking. The George TVRPGC instance requires that vehicles traverse large distances and contains numerous facilities within a rather confined area. Accordingly, the routes are not drawn out to scale in Figure 11.3 and several

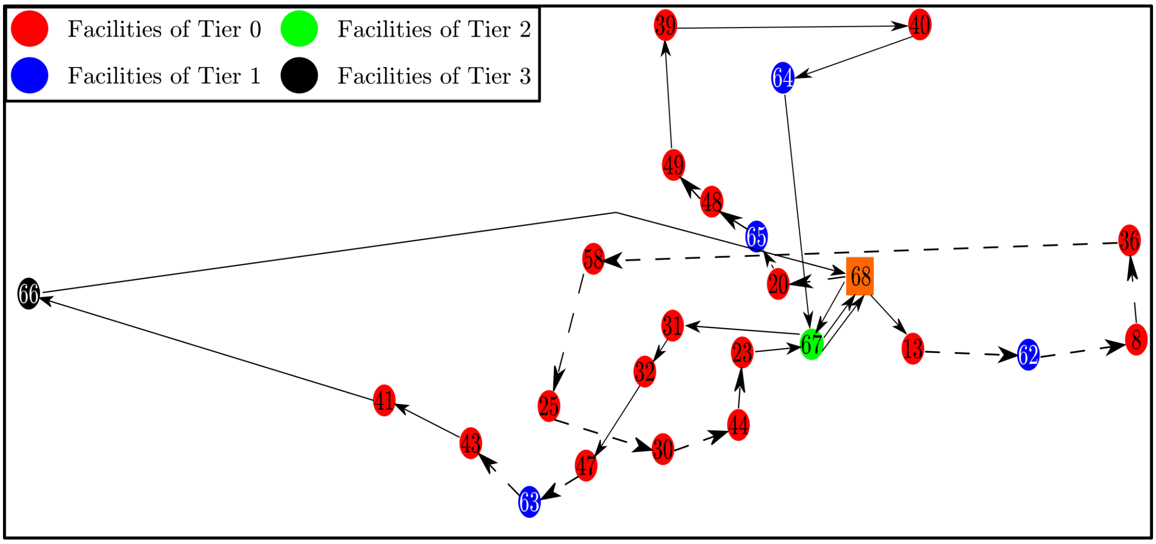


FIGURE 11.3: *The routes proposed by the MACO algorithm in Solution 11 in Figure 11.2 for the George TVRPGC instance (not drawn to scale).*

facilities that are not important in respect of the tier-visitation constraint do not appear on the figure so that the flow of vehicles through the transportation network may be seen more clearly. The first route collects pathological specimens from the central and northern regions, and then consolidates specimens at Facility 67 before returning to its home depot. Similarly, the second vehicle collects pathological specimens from the central and eastern regions, also consolidating specimens at Facility 67 before returning to the depot. Finally, the third vehicle leaves the depot and collects the consolidated pathological specimens from Facility 67, then proceeds to visit several facilities before delivering the pathological specimens to Facility 66 and returning to the depot.

The pathological healthcare service provider supplied the current routes actually employed within the George region, allowing for a direct comparison of the routes followed in practice and the routes suggested by the MACO algorithm. The three objective function values of §9.3.3 are compared in Table 11.22. The total travel time experienced the greatest improvement, which may be largely attributed to the incorporation of the global cross-docking mechanism (currently not practised efficiently by the pathological healthcare service provider), while also simultaneously reducing the number of vehicles and the driver autonomy.

TABLE 11.22: *Comparison between routes currently employed by the pathology healthcare service provider and a route yielded by the MACO algorithm (measured in percentages) for the George TVRPGC instance.*

Objective	Original	MACO	Percentage improvement
Total travel time (min)	4 153.7	2 558.6	38.4
Driver autonomy (min)	1 040	965.1	7.2
Number of vehicles	4	3	25

11.4 Including a clustering phase in the solution of the TVRPGC

As in the method adopted in §8.6, the trade-off of incorporating a clustering phase in the solution of the TVRPGC is investigated in this section within the context of the George TVRPGC instance. This trade-off occurs in respect of solution quality achieved and computation times required by the respective algorithmic implementations when generating non-dominated fronts.

The algorithmic parameter configuration suggested in Table 11.21 was again adopted and an iteration limit of 1 000 was imposed. The maximum driver autonomy was set to 1 100 and the quality of the non-dominated fronts returned by the two algorithmic implementations (with and without a clustering phase) are compared in Table 11.23 in terms of solution quality, according to both the hypervolume and unary-ϵ indicators.

TABLE 11.23: *Comparison of Pareto front approximation qualities as a function of the number of customer clusters for the George TVRPGC instance.*

Instance	No clusters	Two clusters	Three clusters	Four clusters
Hypervolume	0.981	0.453	0.682	0.629
Unary-ϵ	0.001	0.491	0.284	0.354

The results in the table clearly illustrate that performing clustering before applying the MACO algorithm to the TVRPGC is detrimental in terms of solution quality, as expected. The computation time required by the MACO algorithm for a clustered version of the TVRPGC is, however, a fraction of the time required by the MACO algorithm when applied to the full test instance in its entirety. The computation times may be seen in Table 11.24, where the clustered instances of the respective TVRPGC are parallelisable and so the computation time required by the algorithm to reach a final non-dominated front is simply the longest time for all the clusters to be resolved in terms of TVRPGC constraints. The MACO algorithm produced approximately twenty solutions for each of the clusters in each of their non-dominated fronts, which still had to be combined into one single attainment front. The initial procedure employed was simply to combine each of the solutions in all possible ways (approximately 160 000 solutions may be generated in this manner) and to apply the sorting algorithm of the NSGA-II [116]. This, however, took well over four hours to produce a single non-dominated front. A more efficient sorting algorithm was therefore employed whereby the candidate solutions were partitioned into pools of a 1 000 solutions each and these were then sorted individually. The solutions not in the separate non-dominated fronts were removed from further consideration. This process was repeated until only one non-dominated front remained. According to this divide-and-conquer process, the sorting time required was reduced to under 30 minutes, achieving the exact same non-dominated front as when the entire pool of candidate solutions was sorted.

TABLE 11.24: *Computation times (in seconds) required by the respective algorithms when solving the George TVRPGC instance.*

Computing Time	Cluster 1	Cluster 2	Cluster 3	Cluster 4
No clusters	4 490	—	—	—
Two clusters	1 182	583	—	—
Three clusters	370	536	796	—
Four clusters	238	467	581	280

The non-dominated front generated by the MACO algorithm for the instance partitioned into four clusters is shown in Figure 11.4. When compared to the non-dominated front of the same instance without clusters, shown in the same figure, it exhibits a significant decline of solution quality. The number of vehicles utilised in the clustered approach is, in most instances, more than double the number of vehicles used in the non-clustered implementation.

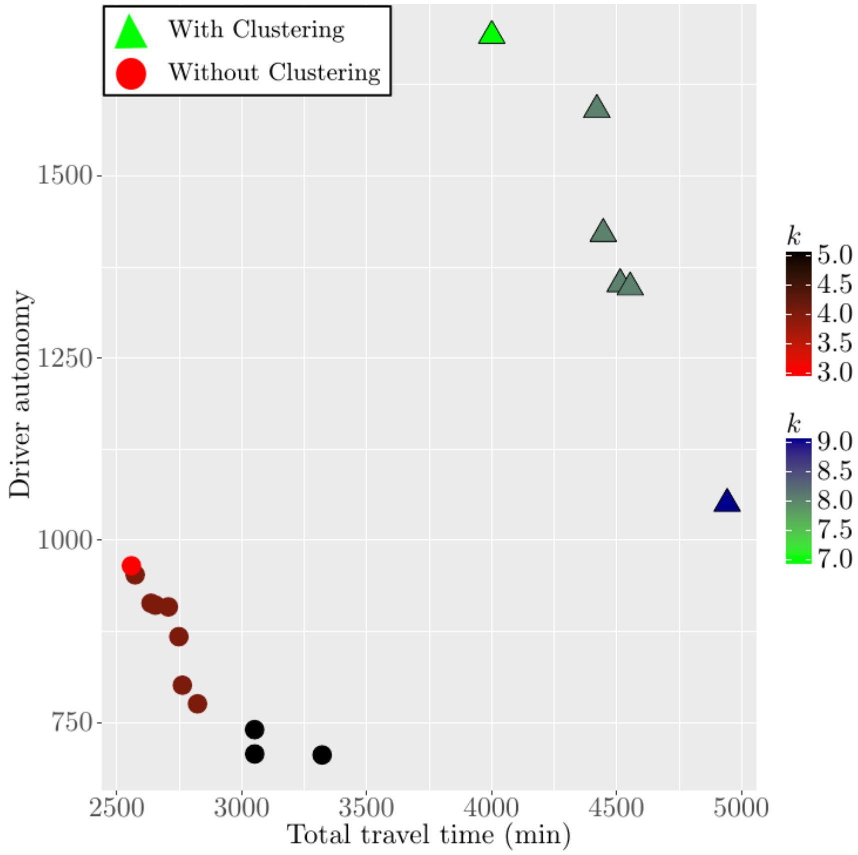


FIGURE 11.4: *The non-dominated fronts generated by the MACO algorithm with respect to the George TVRPGC instance and the same test instance clustered into four customer clusters.*

11.5 Discussion of results

The MACO algorithm was able to determine a relatively consistent range of compromise solutions with respect to total travel time and maximum driver autonomy for the George TVRPGC instance, but failed to utilise a wide range for the number of vehicles, with most of the non-dominated solutions being heavily biased towards utilising a limited combination of vehicles. This is a consequence of deciding not to model the number of vehicles employed *explicitly*. Future solution implementations may perhaps aim to incorporate this aspect explicitly. The combinatorial complexity of adding more objectives to an already difficult problem, however, causes an inherent decline in solution quality with respect to total travel time and driver autonomy compromise solutions.

The MACO algorithm designed in Chapter 10 for the TVRPGC is novel and thus further improvement is certainly achievable. The major areas to exploit are the sequence-fixing functions which would benefit from being able to incorporate a forward looking intuition during the alteration of routes, as the best sequence fixing solution during the current iteration may be a poor solution when resolving other sequence infeasibilities (the moves are not independent of one another).

Similar to the conclusion drawn in §8.6, the clustering component is anticipated to be an integral component when attempting to solve real-life TVRPGC instances (6 000 customers will not be unrealistic in the case of the pathology healthcare provider described in Chapter 1). The size of such a problem instance is well beyond the reach of a direct solution attempt by the MACO algorithm of Chapter 10 when clustering is not applied. The algorithm will also not suffer as severely from the lack of flexibility as experienced in the George TVRPGC instance, because in real-life instances, customers are expected not to be situated as densely around a single point as in the George instance. Furthermore, there are prevalent differences in customer locations due to the legislation adhered to by the pathology healthcare service provider and its provincial departments functioning relatively independently from the national organisation.

11.6 Chapter summary

The uncertainty involved in the design of an effective MACO algorithm for the TVRPGC led to the incorporation of nine algorithmic parameters. A sensitivity analysis was performed in this chapter with respect to these parameters in order to ascertain the best combination of parameter values for producing approximate Pareto fronts of high-quality for a realistic instance of the TVRPGC. The data pertaining to this test instance were presented in §11.1. This presentation was accompanied by a description of the experimental design adopted in the algorithmic parameter sensitivity analysis.

The values of two non-dominated front quality indicators, namely the hypervolume and unary-ϵ indicators, were presented in §11.2 for non-dominated fronts returned by various combinations of values of the algorithmic parameters. The Pareto front approximations returned by the MACO algorithm when adopting the best combination of parameters for the aforementioned TVRPGC instance were presented in §11.3. The corresponding vehicle routes of a solution on the non-dominated front was also illustrated in §11.3.

The TVRPGC instance was additionally partitioned into two to four customer clusters by means of the clustering methodology described in §8.2. The results obtained with and without incorporating a clustering phase in the TVRPGC solution procedure were compared in respect of solution quality and computation time in §11.4. Finally, the results obtained in §11.3 and §11.4 were discussed briefly in §11.5, with a focus on possible future algorithmic improvements of the MACO algorithm in respect of larger instances of the TVRPGC.

Part IV

Case Study

CHAPTER 12

Description of the Case Study

Contents

The real-life case study of the specimen transportation vehicle routing of a pathology healthcare service provider within the Western Cape region is presented in this chapter. A brief introduction of the current pathology healthcare service providers within South Africa and a motivation for the development of an approximate solution approach capable of handling large instances of the TVRPGC are presented in §12.1. The relevant data preparation and cleaning are described in §12.2, and this is followed by a presentation of the locations of the facilities of the pathology healthcare service provider in question in §12.3. The methodology adopted and all necessary assumptions required for the documentation of the current specimen collection vehicle routes implemented by the pathology healthcare service provider are presented in §12.4. Finally, the chapter closes with a brief summary of the work contained in the chapter in §12.5.

12.1 Background

There are numerous pathology healthcare service providers in South Africa, such as the National Health Laboratory Service [326], Ampath [9], Pathcare [338] and Lancet Laboratories [272]. These organisations, either public or private, are responsible for delivering affordable healthcare services to a public afflicted with the highest HIV/AIDS rate in the world and plagued by numerous other preventable diseases [451]. Resource management is critically important to the success of pathology healthcare service providers, and so these organisations seek methods by which to reduce operational costs so that they may continue to provide pathology healthcare services.

One such possible avenue of operational streamlining is the reduction of logistics costs associated with the collection and delivery of pathological specimens between the respective healthcare facilities within the jurisdiction of these organisations. With respect to logistics management, this may be achieved through the reduction of the number of vehicles required to service facilities, minimisation of the total distance travelled by the vehicles and offering fixed collection and delivery schedules for vehicles as to allow for strategy planning. These organisations typically service large networks of facilities dispersed over the entirety of the country, usually resulting in the logistics operations requiring several outsourced organisations to fulfil their collection and delivery demand. Collaboration between these outsourced organisations is generally very limited, resulting in large potential cost-saving benefits through the development of an ubiquitous system capable of handling all the vehicle routing needs of an organisation.

Specimen collection and destination facility data provided by a major South African pathology healthcare service provider are employed in the case study of this dissertation. Certain features of the data, are not, however, divulged for confidentiality reasons such as the identity of the organisation and actual names of the clinics. This anonymisation has nevertheless been performed in such a manner as to ensure that other researchers may apply their algorithms to the same case study data in the future.

12.2 Data preparation

Inevitably, some cleaning and preparation of the data obtained from the participating pathology healthcare service provider were necessary. The data received did not include the addresses of the facilities visited by vehicles, instead merely referring to them by informal names and the municipalities within which the facilities were located. The municipality names referred to in the data were cross-referenced with a second data set provided by the participating organisation in order to obtain the street addresses of these facilities. The street addresses of the facilities were then passed through a Google Maps API [202] so as to geocode the facilities' addresses. All duplicate entries were removed, resulting in 388 unique facility locations for the Western Cape province of South Africa.

The final stage of preparing the data involved generating the expected travel time matrix between the respective facilities. The initial approach was to utilise, once again, a Google Maps API [202], but Google limits non-commercial users to 6 000 requests a day. Thus, if attempting to modularise the requests it would have taken approximately 24 days to finalise. This long waiting time was, however, averted with the assistance of Mr Bennetto [329] and the OPSI servers, as a travel time matrix with numerous entries could thus be generated within a short time period using the OSRM development tool [333].

12.3 Facility locations

As mentioned, there are 388 healthcare facilities in this case study. Of these, there are three facilities of tier three, four facilities of tier two, eleven facilities of tier one and 359 facilities of tier zero, as well as eleven vehicle depots. The locations of these facilities are illustrated in Figure 12.1 and their (longitude, latitude) coordinates are provided in Table 12.1. The expected travel times matrix for the facilities shown in Figure 12.1 is furthermore available online [403].

TABLE 12.1: *Locations of healthcare facilities of the pathology healthcare service provider within the Western Cape province of South Africa.*

Name	Longitude	Latitude	Tier	Name	Longitude	Latitude	Tier
Facility 1	18.4588	−34.0123	0	Facility 2	18.4535	−34.0042	0
Facility 3	18.4746	−33.9493	0	Facility 4	18.4804	−34.0593	0
Facility 5	18.6212	−34.0466	0	Facility 6	18.6324	−34.0149	0
Facility 7	18.5732	−33.9892	0	Facility 8	18.5457	−33.9501	0
Facility 9	18.5268	−33.9945	0	Facility 10	18.3794	−34.1294	0

Name	Longitude	Latitude	Tier	Name	Longitude	Latitude	Tier
Facility 11	18.4161	−34.1318	0	Facility 12	20.1229	−33.7977	0
Facility 13	20.1290	−33.7826	0	Facility 14	20.0582	−33.8284	0
Facility 15	20.0854	−33.8375	0	Facility 16	19.8908	−33.8018	0
Facility 17	19.8933	−33.8177	0	Facility 18	19.8886	−33.7926	0
Facility 19	19.8287	−33.9473	0	Facility 20	19.8935	−33.7999	0
Facility 21	20.7266	−33.9049	0	Facility 22	20.6510	−34.0078	0
Facility 23	20.5384	−34.0353	0	Facility 24	20.4440	−34.0341	0
Facility 25	20.4381	−34.0257	0	Facility 26	20.4505	−34.0241	0
Facility 27	20.4499	−34.0243	0	Facility 28	20.0781	−33.9354	0
Facility 29	19.3013	−33.3634	0	Facility 30	19.4338	−34.2415	0
Facility 31	19.2822	−33.9899	0	Facility 32	19.2931	−33.9914	0
Facility 33	19.2862	−33.9935	0	Facility 34	19.2017	−33.4131	0
Facility 35	19.1879	−33.4167	0	Facility 36	19.1465	−33.2847	0
Facility 37	19.1385	−33.2862	0	Facility 38	19.3073	−33.3711	0
Facility 39	19.3113	−33.3692	0	Facility 40	19.3013	−33.3635	0
Facility 41	19.3458	−33.3501	0	Facility 42	19.3103	−33.3697	0
Facility 43	19.3200	−33.3329	0	Facility 44	19.3272	−33.2899	0
Facility 45	20.8513	−33.1943	0	Facility 46	20.0289	−33.3407	0
Facility 47	19.6674	−33.4832	0	Facility 48	19.5522	−33.5178	0
Facility 49	19.4434	−33.6443	0	Facility 50	19.4940	−33.6412	0
Facility 51	19.4563	−33.6220	0	Facility 52	19.3175	−33.6909	0
Facility 53	19.4161	−33.7249	0	Facility 54	19.4382	−33.6485	0
Facility 55	19.4087	−33.5903	0	Facility 56	18.7000	−33.9533	0
Facility 57	19.4593	−33.6507	0	Facility 58	19.4334	−33.6461	0
Facility 59	18.4953	−33.5647	0	Facility 60	18.4940	−33.5651	0
Facility 61	18.4736	−33.5124	0	Facility 62	18.3867	−33.3704	0
Facility 63	18.7032	−33.4657	0	Facility 64	18.7232	−33.4546	0
Facility 65	18.7237	−33.4547	0	Facility 66	18.7283	−33.4754	0
Facility 67	18.9956	−33.6646	0	Facility 68	18.9919	−33.6746	0
Facility 69	18.9908	−33.6836	0	Facility 70	18.9916	−33.7022	0
Facility 71	18.9703	−33.7278	0	Facility 72	19.0119	−33.8453	0
Facility 73	19.1026	−33.8960	0	Facility 74	18.5505	−33.9106	0
Facility 75	18.8607	−33.8157	0	Facility 76	18.9985	−33.7273	0
Facility 77	18.9877	−33.7224	0	Facility 78	18.9863	−33.7237	0
Facility 79	18.9841	−33.7132	0	Facility 80	18.9868	−33.7116	0
Facility 81	18.9662	−33.7315	0	Facility 82	18.4947	−33.5743	0
Facility 83	18.4885	−33.5503	0	Facility 84	18.9814	−33.7281	0
Facility 85	19.0450	−33.2929	0	Facility 86	19.0067	−33.1868	0
Facility 87	18.9980	−33.0783	0	Facility 88	18.9934	−33.0096	0
Facility 89	18.9942	−33.0182	0	Facility 90	18.7578	−32.9006	0
Facility 91	18.7627	−32.9069	0	Facility 92	18.6770	−33.0107	0
Facility 93	18.6644	−33.1503	0	Facility 94	18.8742	−33.3518	0
Facility 95	18.8855	−33.3709	0	Facility 96	18.9199	−33.3840	0
Facility 97	18.9853	−33.6040	0	Facility 98	18.9802	−33.6473	0
Facility 99	18.7935	−33.6468	0	Facility 100	18.9611	−33.6879	0
Facility 101	18.0015	−32.9121	0	Facility 102	17.8842	−32.8123	0
Facility 103	18.0596	−32.7888	0	Facility 104	18.1655	−32.7737	0
Facility 105	18.3411	−33.0610	0	Facility 106	18.0333	−33.0911	0
Facility 107	17.9449	−33.0089	0	Facility 108	17.9214	−33.0073	0
Facility 109	18.0093	−32.9149	0	Facility 110	18.0049	−32.7422	0
Facility 111	23.3222	−34.0505	0	Facility 112	23.0479	−34.0385	0
Facility 113	23.0536	−34.0351	0	Facility 114	23.1078	−34.0345	0
Facility 115	23.0981	−34.0477	0	Facility 116	23.3650	−34.0531	0
Facility 117	23.4885	−33.9503	0	Facility 118	23.3667	−34.0500	0
Facility 119	23.3426	−34.0524	0	Facility 120	23.3667	−34.0500	0
Facility 121	22.9346	−33.9460	0	Facility 122	22.8084	−34.0197	0
Facility 123	22.6213	−33.9533	0	Facility 124	22.4892	−34.0050	0
Facility 125	22.4941	−34.0068	0	Facility 126	22.4969	−34.0078	0
Facility 127	22.4790	−33.9750	0	Facility 128	22.4518	−33.9590	0
Facility 129	22.4125	−33.9432	0	Facility 130	22.4511	−33.9509	0
Facility 131	22.4603	−34.0167	0	Facility 132	22.4923	−34.0031	0

Name	Longitude	Latitude	Tier	Name	Longitude	Latitude	Tier
Facility 133	22.4842	−33.9931	0	Facility 134	22.4736	−33.9912	0
Facility 135	20.9565	−34.0907	0	Facility 136	20.9579	−34.0915	0
Facility 137	21.2550	−34.0937	0	Facility 138	21.2548	−34.0947	0
Facility 139	21.5859	−34.2041	0	Facility 140	21.4079	−34.3759	0
Facility 141	22.2205	−34.0483	0	Facility 142	22.1440	−33.9547	0
Facility 143	23.1251	−33.6574	0	Facility 144	23.1250	−33.6595	0
Facility 145	23.1295	−33.6565	0	Facility 146	23.1253	−33.6606	0
Facility 147	23.1667	−33.7167	0	Facility 148	22.4203	−33.8947	0
Facility 149	21.5030	−31.9178	0	Facility 150	23.7623	−31.9642	0
Facility 151	22.1291	−34.1865	0	Facility 152	22.1395	−34.1808	0
Facility 153	22.1106	−34.1234	0	Facility 154	22.0921	−34.1808	0
Facility 155	22.1161	−34.1766	0	Facility 156	22.1132	−34.1713	0
Facility 157	22.1051	−34.1496	0	Facility 158	22.0442	−33.2463	0
Facility 159	22.0333	−33.2167	0	Facility 160	22.5204	−33.4980	0
Facility 161	22.4421	−33.5773	0	Facility 162	22.2385	−33.6065	0
Facility 163	22.2416	−33.5953	0	Facility 164	22.2204	−33.6018	0
Facility 165	22.1849	−33.5868	0	Facility 166	22.1912	−33.5835	0
Facility 167	22.2038	−33.5798	0	Facility 168	21.2689	−33.4869	0
Facility 169	21.4461	−33.4948	0	Facility 170	21.4667	−33.4833	0
Facility 171	21.6780	−33.5276	0	Facility 172	18.8709	−33.9306	0
Facility 173	18.8560	−33.9232	0	Facility 174	18.8579	−34.0758	0
Facility 175	18.7746	−34.0622	0	Facility 176	18.8664	−34.1135	0
Facility 177	18.8526	−34.1337	0	Facility 178	18.8482	−34.0858	0
Facility 179	18.9131	−34.1223	0	Facility 180	18.8833	−34.1507	0
Facility 181	18.8611	−33.9346	0	Facility 182	18.8554	−33.9105	0
Facility 183	18.8801	−33.9255	0	Facility 184	18.8485	−33.9802	0
Facility 185	18.8496	−33.9176	0	Facility 186	18.9541	−33.9206	0
Facility 187	18.6650	−34.0274	0	Facility 188	18.6079	−33.8916	0
Facility 189	18.5582	−33.9847	0	Facility 190	18.5548	−34.0731	0
Facility 191	18.6373	−34.0620	0	Facility 192	18.6261	−34.0538	0
Facility 193	18.5998	−34.0505	0	Facility 194	18.6127	−34.0440	0
Facility 195	18.5836	−34.0171	0	Facility 196	18.6120	−34.0155	0
Facility 197	18.6102	−34.0110	0	Facility 198	18.6036	−33.9922	0
Facility 199	18.5851	−33.9918	0	Facility 200	18.5794	−33.9877	0
Facility 201	18.5661	−33.9873	0	Facility 202	18.5705	−33.9731	0
Facility 203	18.5548	−33.9886	0	Facility 204	18.5268	−33.9945	0
Facility 205	18.5054	−33.9940	0	Facility 206	18.5289	−33.9440	0
Facility 207	18.4485	−33.9320	0	Facility 208	18.4207	−33.9307	0
Facility 209	18.5269	−33.9944	0	Facility 210	18.6157	−34.0246	0
Facility 211	18.6246	−34.0238	0	Facility 212	18.4778	−33.9370	0
Facility 213	18.4169	−33.9047	0	Facility 214	18.6639	−34.0278	0
Facility 215	18.7190	−33.9974	0	Facility 216	18.5079	−34.0266	0
Facility 217	18.5033	−34.0356	0	Facility 218	18.4921	−34.0314	0
Facility 219	18.4879	−34.0671	0	Facility 220	18.4899	−34.0748	0
Facility 221	18.4839	−34.0863	0	Facility 222	18.4614	−34.0626	0
Facility 223	18.4399	−34.0786	0	Facility 224	18.4702	−34.1075	0
Facility 225	18.4274	−34.1374	0	Facility 226	18.3499	−34.1484	0
Facility 227	18.3515	−34.1487	0	Facility 228	18.4319	−34.0684	0
Facility 229	18.3585	−34.0389	0	Facility 230	18.3422	−34.0543	0
Facility 231	18.4462	−34.0194	0	Facility 232	18.4701	−34.0047	0
Facility 233	18.4653	−34.0216	0	Facility 234	18.4662	−34.0340	0
Facility 235	18.4804	−34.0593	0	Facility 236	18.4809	−34.0588	0
Facility 237	18.4920	−34.0443	0	Facility 238	18.5079	−34.0266	0
Facility 239	18.6678	−34.0519	0	Facility 240	18.7228	−34.0055	0
Facility 241	18.7176	−33.9872	0	Facility 242	18.6861	−33.9794	0
Facility 243	18.6806	−34.0063	0	Facility 244	18.6859	−34.0065	0
Facility 245	18.7038	−33.4652	0	Facility 246	18.6441	−33.9151	0
Facility 247	18.5702	−33.9532	0	Facility 248	18.5607	−33.9488	0
Facility 249	18.5436	−33.9478	0	Facility 250	18.5118	−33.8321	0
Facility 251	18.5298	−33.8268	0	Facility 252	18.4845	−33.9107	0
Facility 253	18.4817	−33.9075	0	Facility 254	18.4862	−33.9000	0

Name	Longitude	Latitude	Tier	Name	Longitude	Latitude	Tier
Facility 255	18.5098	−33.9957	0	Facility 256	18.5975	−33.9978	0
Facility 257	18.5046	−33.9616	0	Facility 258	18.6504	−34.0135	0
Facility 259	18.6468	−34.0136	0	Facility 260	18.6705	−34.0479	0
Facility 261	18.4241	−33.9249	0	Facility 262	18.6208	−34.0507	0
Facility 263	18.5925	−33.9999	0	Facility 264	18.5347	−33.9662	0
Facility 265	18.5175	−33.9588	0	Facility 266	18.5533	−33.9645	0
Facility 267	18.5483	−33.9677	0	Facility 268	18.6426	−33.9713	0
Facility 269	18.6579	−34.0423	0	Facility 270	18.6832	−34.0401	0
Facility 271	18.7057	−34.0440	0	Facility 272	18.7093	−34.0504	0
Facility 273	18.7022	−34.0522	0	Facility 274	18.6778	−34.0403	0
Facility 275	18.6708	−34.0513	0	Facility 276	18.6736	−34.0481	0
Facility 277	18.6074	−34.0499	0	Facility 278	18.5852	−33.9919	0
Facility 279	18.5871	−33.9968	0	Facility 280	18.5805	−34.0098	0
Facility 281	18.5653	−33.9964	0	Facility 282	18.5213	−33.9237	0
Facility 283	18.4895	−33.9220	0	Facility 284	18.4893	−33.9218	0
Facility 285	18.5037	−33.9115	0	Facility 286	18.5125	−33.9113	0
Facility 287	18.5580	−33.9186	0	Facility 288	18.5472	−33.8887	0
Facility 289	18.5407	−33.8586	0	Facility 290	18.4691	−33.9892	0
Facility 291	18.6633	−33.9806	0	Facility 292	18.4649	−33.9291	0
Facility 293	18.7182	−33.8640	0	Facility 294	18.7505	−33.8436	0
Facility 295	18.5565	−33.9809	0	Facility 296	18.4373	−33.9291	0
Facility 297	18.6776	−33.9270	0	Facility 298	18.6789	−33.9266	0
Facility 299	18.6859	−33.8724	0	Facility 300	18.7106	−33.8710	0
Facility 301	18.7233	−33.8674	0	Facility 302	18.7182	−33.8640	0
Facility 303	18.4655	−33.9417	0	Facility 304	18.3528	−34.0287	0
Facility 305	18.7052	−33.8416	0	Facility 306	18.7188	−33.7821	0
Facility 307	18.6544	−33.8302	0	Facility 308	18.6523	−33.8421	0
Facility 309	18.6578	−33.9024	0	Facility 310	18.6370	−33.9041	0
Facility 311	18.5970	−33.9234	0	Facility 312	18.5973	−33.9207	0
Facility 313	18.5994	−33.9361	0	Facility 314	18.6484	−33.9450	0
Facility 315	18.5843	−33.9374	0	Facility 316	18.5845	−33.9065	0
Facility 317	18.5935	−33.9047	0	Facility 318	18.7144	−33.8585	0
Facility 319	18.5816	−33.9489	0	Facility 320	18.5781	−33.9317	0
Facility 321	18.6079	−33.8916	0	Facility 322	18.7216	−33.8530	0
Facility 323	18.5640	−33.8901	0	Facility 324	18.5779	−33.9315	0
Facility 325	18.6027	−34.0404	0	Facility 326	18.5457	−33.9501	0
Facility 327	19.2194	−34.4258	0	Facility 328	19.2372	−34.4167	0
Facility 329	19.1332	−34.3859	0	Facility 330	19.0291	−34.3329	0
Facility 331	19.1983	−34.2296	0	Facility 332	19.4283	−34.2300	0
Facility 333	19.3503	−34.5907	0	Facility 334	19.4586	−34.4411	0
Facility 335	19.8932	−34.4704	0	Facility 336	20.0334	−34.5367	0
Facility 337	20.0543	−34.5417	0	Facility 338	19.0036	−34.1547	0
Facility 339	18.6258	−31.7685	0	Facility 340	18.5042	−31.6651	0
Facility 341	18.3447	−31.5572	0	Facility 342	18.2360	−31.8164	0
Facility 343	18.2615	−31.6292	0	Facility 344	18.5278	−31.6441	0
Facility 345	18.2572	−31.0512	0	Facility 346	19.4344	−30.9486	0
Facility 347	19.4418	−30.9620	0	Facility 348	19.1079	−31.3794	0
Facility 349	19.7675	−31.4626	0	Facility 350	19.7728	−31.4750	0
Facility 351	18.7378	−31.6070	0	Facility 352	18.8925	−32.1875	0
Facility 353	18.9014	−32.1976	0	Facility 354	18.6057	−32.1596	0
Facility 355	18.3052	−32.0944	0	Facility 356	18.3403	−32.3135	0
Facility 357	19.0113	−32.5861	0	Facility 358	19.0081	−32.5819	0
Facility 359	20.6646	−32.3989	0	Facility 360	18.6079	−33.8916	1
Facility 361	17.9914	−32.9132	1	Facility 362	23.0406	−34.0361	1
Facility 363	22.1363	−34.1896	1	Facility 364	22.5936	−32.3531	1
Facility 365	22.1898	−33.5914	1	Facility 366	18.6218	−34.0466	1
Facility 367	18.6673	−34.0357	1	Facility 368	19.2282	−34.4230	1
Facility 369	18.8570	−34.0765	1	Facility 370	18.5043	−31.6690	1
Facility 371	18.4158	−33.9067	2	Facility 372	19.4582	−33.6445	2
Facility 373	18.9716	−33.7265	2	Facility 374	22.4511	−33.9515	2
Facility 375	18.4617	−33.9408	3	Facility 376	18.4893	−33.9540	3

Name	Longitude	Latitude	Tier	Name	Longitude	Latitude	Tier
Facility 377	18.6129	−33.9109	3	Facility 378	18.6079	−33.8916	−1
Facility 379	17.9914	−32.9132	−1	Facility 380	23.0406	−34.0361	−1
Facility 381	22.1363	−34.1896	−1	Facility 382	22.5936	−32.3531	−1
Facility 383	22.1898	−33.5914	−1	Facility 384	18.6218	−34.0466	−1
Facility 385	18.6673	−34.0357	−1	Facility 386	19.2282	−34.4230	−1
Facility 387	18.8570	−34.0765	−1	Facility 388	18.5043	−31.6690	−1

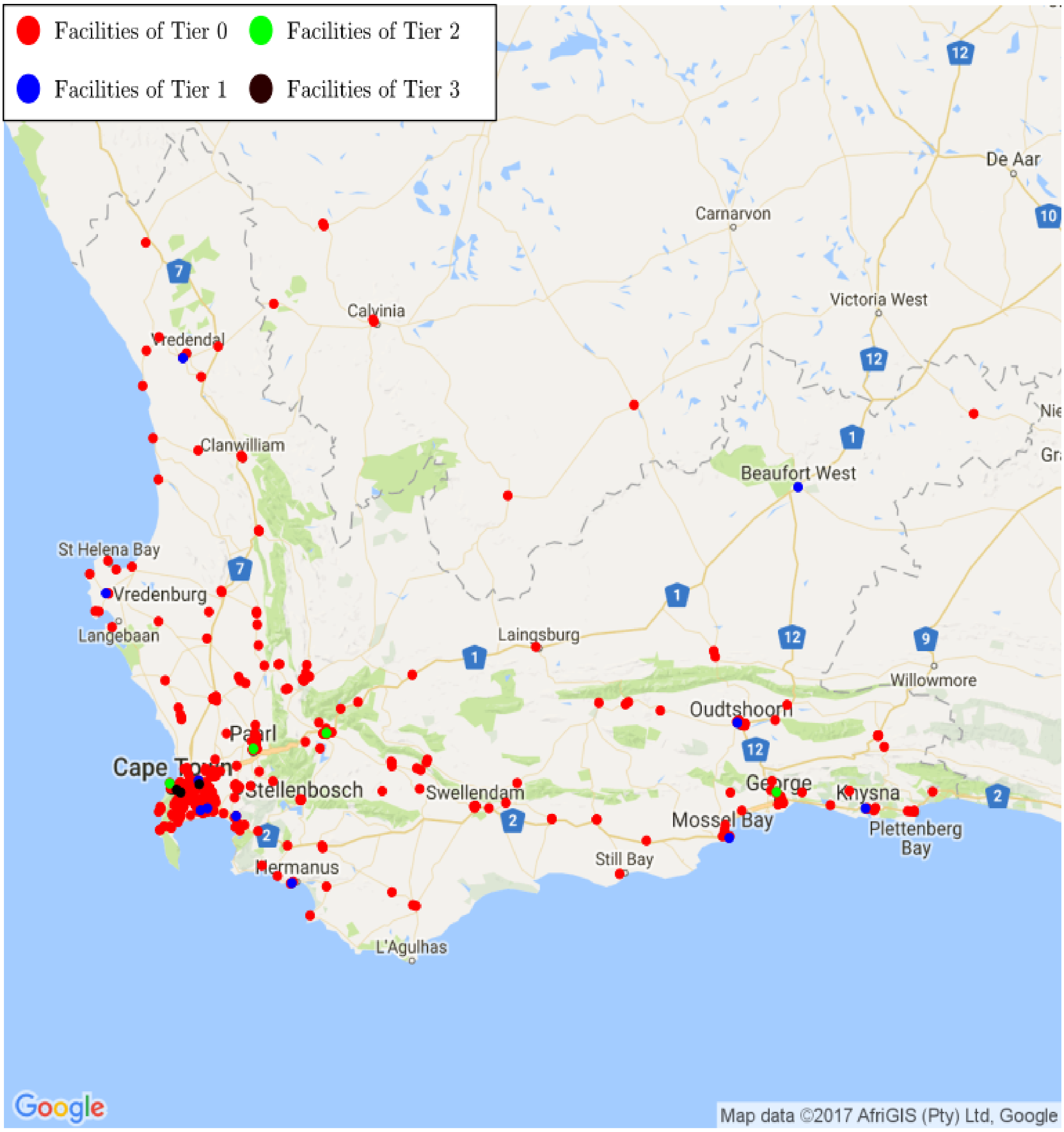


FIGURE 12.1: *Distribution of healthcare facilities of a pathological healthcare provider within the South African Western Cape.*

12.4 Currently implemented vehicle routes

The data received from the participating pathology healthcare service provider also contains the vehicle routes currently employed by the organisation. These routes span nine regions, namely Vredendal, Sutherland, Groote Schuur, Worcester, Paarl, Vredenburg, George, Green Point and Helderberg. The data received typically consisted of fixed vehicle routes that are repeated daily. Additional routes that are only travelled on specific days of the week if certain demand levels are reached (as well as emergency routes) were, however, also included. The data received furthermore contained routes within the city of Cape Town exclusively pertaining to vehicles travelling between the eight major hospitals in the municipality, resulting in several facilities being visited numerous times within the planning period. This was accommodated in the model of Chapter 9 by making virtual copies of the facilities and enforcing time window constraints on the various copies of the facilities so as to encourage a similar visitation schedule to the routes currently employed. The pathology healthcare service provider also currently employs a mobile clinic in the Caledon municipality — this particularity was accommodated in the model by setting the destinations of the mobile clinic as fixed destinations to be serviced by some vehicle, as heterogeneous vehicles are not considered in the formulation of the model of Chapter 9.

The number of routes employed in each region and the total travel time associated with each region are shown in Table 12.2. Additionally, a single vehicle is employed on a twenty-four hour basis that simply visits each of the eight major hospitals in the city of Cape Town five times during a planning period. The intra-hospital visitation schedule results in a total travel time of 686 minutes.

Table 12.2: *The number of routes and total expected travel time required to service each region.*

Region	Number of routes	Total travel time (min)
Vredendal	4	1 862.7
Sutherland	2	478.0
Groote Schuur	7	872.2
Worcester	8	1 866.7
Paarl	5	1 021.7
Vredenburg	2	538.1
George	12	4 153.7
Green Point	20	3 494.6
Helderberg	2	413.5
Caledon (Mobile)	2	473.9
Total	64	15 860.6

The above-mentioned routes employed by the pathology healthcare service provider to service the Western Cape pathological specimen collection and delivery demands result in a total travel time of 15 860.6 minutes and a maximum driver autonomy of 1 092 minutes. The aforementioned routes are followed by a total of 64 vehicles, although the motivation behind the choice of this number of vehicles remains unclear as it would seem from the data that two or more routes may often be serviced by a single vehicle. Perhaps, the vehicles were also used for other purposes. The ideal number of vehicles required was estimated according to the following logic: The maximum driver autonomy of a vehicle was set to 1 092 minutes and routes that end at a facility at which another route begins were allocated to a single vehicle if the driver autonomy constraint was not violated. This process resulted in an estimate of 36 vehicles required to service the routes suggested by the pathological healthcare organisation.

12.5 Chapter summary

In this chapter, a real-life instance of specimen collection and delivery of a pathology healthcare service provider within the Western Cape region was described in some detail. The current pathology healthcare service providers within South Africa were mentioned in §12.1 and a motivation was given for developing an approximate solution approach aimed at solving real-life TVRPGC instances.

The data received from an industry partner required a certain level of preparation in order to be able to apply the MACO of §10.4 to these data. The various steps of this preparation process were described in §12.2. Information pertaining to the location of the facilities within the real-life TVRPGC instance was presented in §12.3.

The vehicle routes currently implemented by the industry partner for specimen collection and delivery, which will be utilised in the following chapter to validate the results returned by the MACO of §10.4, were described in §12.4.

CHAPTER 13

Case Study TVRPGC Results

Contents

This chapter is devoted to a presentation and discussion of the results returned by the MACO algorithm with respect to the Western Cape TVRPGC instance described in Chapter 12. The experimental setup adopted in this chapter is described in §13.1. The results returned by the MACO algorithm and the potential financial benefits of adopting the vehicle routes thus proposed are elaborated upon in §13.2. A clustering phase is yet again incorporated into the solution approach and the subsequent trade-off between solution quality and computation time is elucidated in §13.3. The results obtained in this chapter are briefly discussed in §13.4, and this is followed by a brief summary in §13.5 of the work presented in this chapter.

13.1 Experimental setup

The MACO algorithm of §10.4 is applied in this chapter to the Western Cape TVRPGC instance described in Chapter 12. The MACO algorithm employs several parameters in its implementation and the values for these parameters uncovered in Chapter 11 are adopted. The MACO algorithm is allowed a limit of 1 000 search iterations after which the incumbent solutions are recorded.

Time window information was not available for the Western Cape TVRPGC instance described in Chapter 12. Accordingly, all the time window model information was specified in such a manner as to not be a limiting constraint or result in any penalisation of the objective function values during the implementation of the MACO algorithm within this case study. The maximum driver autonomy was taken as 1 100 minutes and the facility visitation capacity was set to the value $\gamma = 5$.

All the numerical work reported in this chapter was again performed on an i7-4770 processor running at 3.40 GHz with 8GB of memory within the Windows 7 operating system after having implemented the MACO algorithm of Chapter 10 in R.

13.2 Numerical results

The Pareto approximation yielded by the MACO algorithm for the Western Cape TVRPGC instance may be seen in Figure 13.1. The approximation contains nine feasible solutions. During execution of the MACO algorithm, approximately thirty solutions were, however, uncovered, with the majority of them being removed from final consideration due to certain constraint violations. This highlights the nature of the TVRPGC — its solution space is typically tightly constrained, allowing for even heavily penalised infeasible solutions to be competitive with feasible solutions.

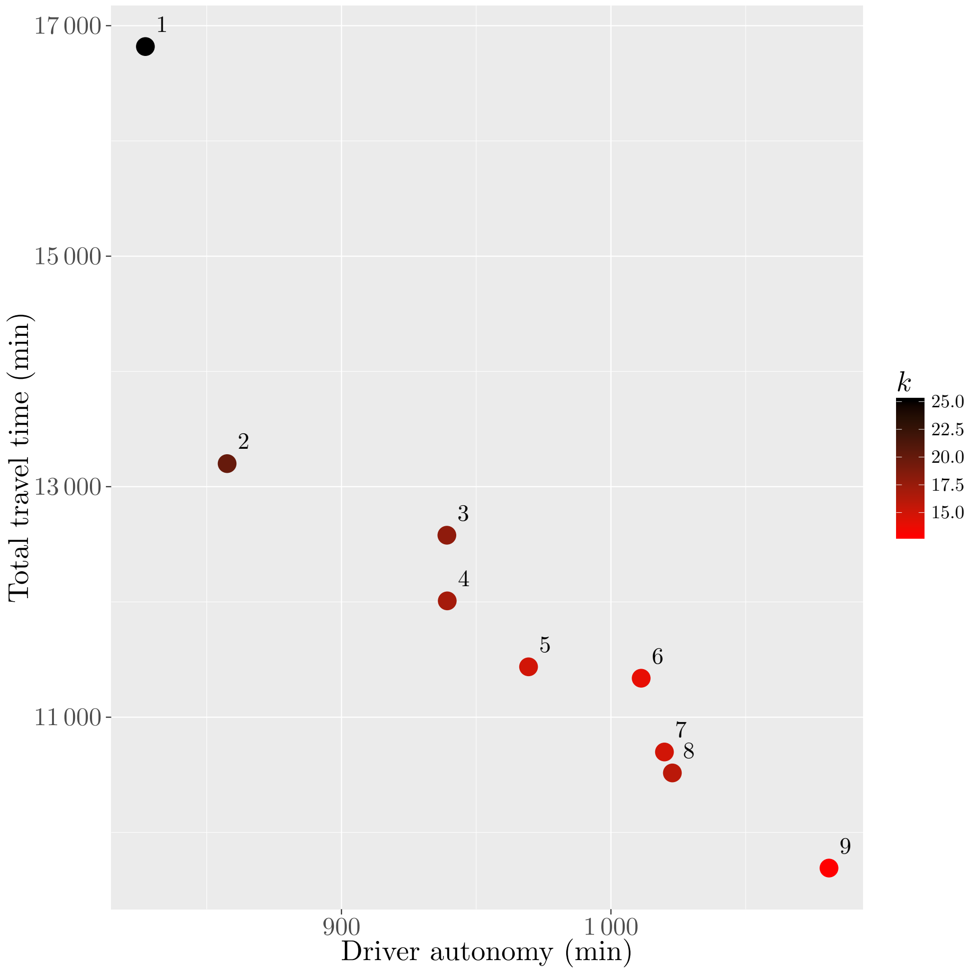


FIGURE 13.1: *Pareto approximation returned by the MACO algorithm for the Western Cape TVRPGC instance.*

The algorithmic implementation took eight hours and sixteen minutes to execute a thousand iterations. This computation burden is compatible with current industry practice as the routes employed by the industry partner are typically fixed and repeated on a daily schedule. Additionally, the planning period adopted by the pathology healthcare service provider is typically a monthly schedule, with minimal alterations to the routes being experienced throughout a planning period. The non-dominated front of Figure 13.1 exhibits a considerable trade-off in solution choices with respect to the total travel time and the number of vehicles utilised. The variation in driver autonomy is relatively small as the trade-off with respect to driver autonomy is limited to approximately 150 minutes. The routes proposed by the MACO algorithm associated with the points in Figure 13.1 are presented in Tables C.1–C.9 of Appendix C.

The improvement potentially to be experienced by the industry partner if Solution 9 of Figure 13.1 were to be adopted on a daily basis are summarised in Table 13.1 with respect to the three objectives pursued in this dissertation. These improvements are considerable — such improvements are typically not achievable when comparing optimisation results against industry standards. The industry partner considered in this dissertation does, however, have its operations independently coordinated based on municipality jurisdiction.

TABLE 13.1: *Comparison between routes currently employed by the pathology healthcare service provider and Solution 9 of Figure 13.1.*

Objective	Original	MACO	Percentage improvement
Total travel time (min)	15 175.7	9 690.8	36.1
Driver autonomy (min)	1 080.9	992.1	8.1
Number of vehicles	36	13	63.8

The largest potential improvement was experienced in the reduction of the number of vehicles required to perform the necessary collection and delivery of pathological specimens between the respective facilities. A reduction of over sixty-three percent was achievable by considering all facilities collectively, irrespective of municipality jurisdiction, and exploiting the cross-docking capability of the MACO algorithm, which is absent in the current implementation of the pathology healthcare service provider.

The improvements summarised in Table 13.1 may be translated into financial terms under the assumption of an average travel speed of 60km/h and adoption of the cost per kilometer vehicle maintenance rates suggested by the *South African Revenue Service* (SARS) [387]. The financial savings are elucidated in Table 13.2, with the first row referring to the fixed cost per year of owning a vehicle. The fixed cost refers to costs such as repayment costs, insurance, capital depreciation, and asset depreciation. The fixed cost of owning a single vehicle equates to a value of R28 492 annually. Therefore, the reduction in the number of vehicles employed within the fleet of the pathology healthcare service provider would induce an operating expense reduction of R626 824 per year. The second row of the table refers to the potential cost reduction with respect to fuel costs due to the proposed vehicle routes requiring less fuel to service the collection and delivery of pathological specimens. The potential fuel cost reduction amounts to R2 622 653 annually. The third row of the table refers to reduction in maintenance costs (a vehicle typically requires more maintenance the more kilometers it travels). The maintenance cost reduction results in a potential annual saving of R640 648 per year. The overall potential cost reduction of adopting Solution 9 of Figure 13.1 on a daily basis is therefore R3 890 125 per year. This value serves as a lower bound on the potential financial savings since certain costs are difficult to quantify and were therefore excluded from consideration. Such costs include the salaries of the drivers, the reduction in operating costs that may be experienced through the incorporation of a fixed routing schedule (*i.e.* fixed collection and delivery times of pathological specimens) and the cost reduction resulting from implementing routing solutions provided in-house as opposed to tenders placed to the public.

13.3 Including a clustering phase in the solution approach

As previously considered in §8.6, the trade-off of incorporating a clustering phase in the solution of the TVRPGC is investigated in this section within the context of the Western Cape TVRPGC instance. This trade-off occurs in respect of solution quality achieved and computation times required by the respective algorithmic implementations when generating non-dominated fronts.

TABLE 13.2: *Potential annual cost reduction in adopting a solution returned by the MACO algorithm for the Western Cape TVRPGC relative to the routing currently adopted by the pathology healthcare service provider (according to rates obtained from SARS [387]).*

Cost to run a vehicle annually	Rate	Reduction	Total Savings (R/p.a.)
Fixed cost (R/vehicle/p.a.)	28 492	23 vehicles	626 824
Fuel cost (R/km)	1.31	2 002 025 km/p.a.	2 622 653
Maintenance cost (R/km)	0.32	2 002 025 km/p.a.	640 648
Total			3 890 125

The algorithmic parameter configuration suggested in Table 11.21 was again adopted and an iteration limit of 1 000 was imposed. The maximum driver autonomy was set to 1 100 minutes and the quality of the non-dominated fronts returned by the two algorithmic implementations (with and without a clustering phase) are compared in Table 13.3 according to both the hypervolume and unary-ϵ indicators.

TABLE 13.3: *Comparison of Pareto front approximation qualities as a function of the number of customer clusters for the Western Cape TVRPGC instance.*

Instance	No clustering	Ten clusters
Hypervolume	0.936	0.819
Unary-ϵ	0.008	0.017

The above results show that there is a decline in solution quality when clustering is performed prior to applying the MACO algorithm. The decline in solution quality is, however, considerably less than that documented in §11.4 for the much smaller George TVRPGC instance, possibly due to the clusters each containing more facilities, which affords the MACO algorithm more flexibility. The computation time required by the MACO algorithm to approximate the non-dominated front for the clustered instance is, however, a fraction of the time required by the MACO algorithm when considering the Western Cape TVRPGC instance in its entirety. The respective computation times may be seen in Table 13.4. As described in §11.4, the solution procedure is parallelisable and thus the computation time required by the MACO algorithm to reach a full solution for the clustered problem instance is simply the longest time required to resolve a cluster (17.36 minutes). The MACO algorithm was able to discover numerous solutions for each of the clusters in each of their non-dominated fronts, which still had to be combined into a single attainment front. A divide-and-conquer approach was applied for this purpose (as previously described in §11.4), because 90 389 452 608 possible route combinations resulted from combining the clustered non-dominated fronts.

TABLE 13.4: *Computation times (in minutes) required by the respective MACO algorithms when solving the Western Cape TVRPGC instance.*

	Cluster									
Computing time	1	2	3	4	5	6	7	8	9	10
No clustering	498									
Ten clusters	17.36	14.55	4.13	4.72	4.94	5.73	5.08	3.29	3.24	7.03

The non-dominated front generated by the MACO algorithm for the Western Cape TVRPGC instance partitioned into ten clusters is illustrated in Figure 13.2. When compared to the non-dominated front of the same instance without clustering, illustrated in the same figure, it exhibits a decline in solution quality with respect to both total travel time and driver autonomy. The number of vehicles k, however, typically improved with the incorporation of a clustering phase in the solution approach of the MACO algorithm.

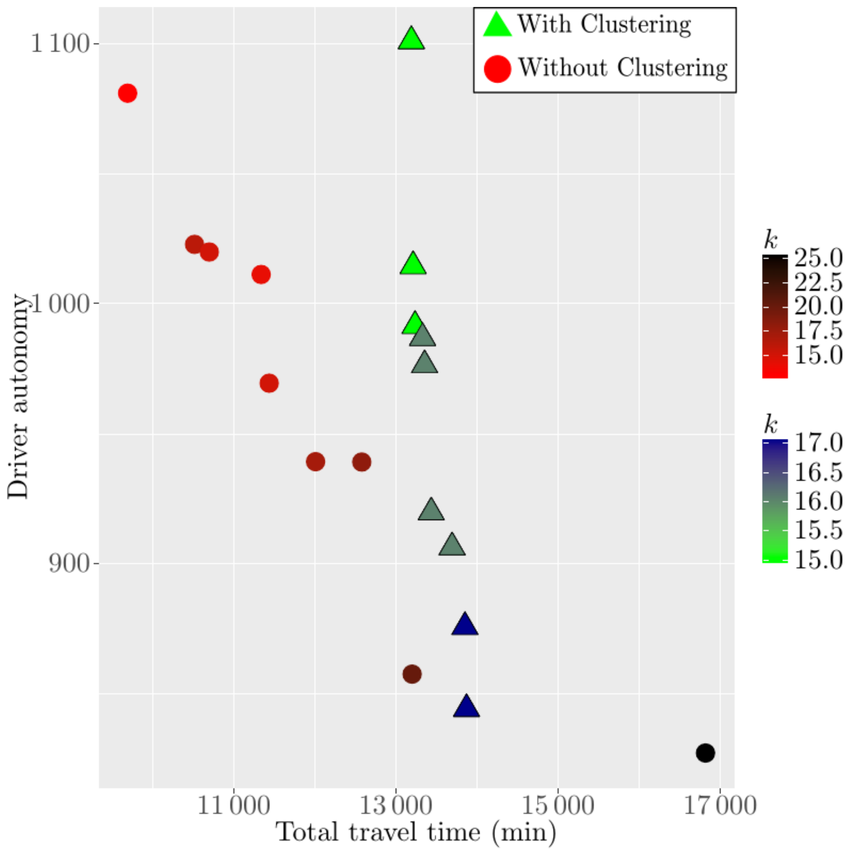


FIGURE 13.2: *The non-dominated fronts generated by the MACO algorithm with respect to the Western Cape TVRPGC instance and the same test instance clustered into ten customer clusters.*

13.4 Discussion of results

The true complexity of the TVRPGC was once again highlighted during the implementation of the MACO algorithm in the context of the Western Cape TVRPGC instance. The MACO algorithm required over eight hours to perform a 1 000 search iterations, with the global cross-docking model feature being responsible for a large portion of the computational burden due to the nested nature of the tier visitation constraint. A facility may have its specimens collected by a vehicle which consolidates the specimen at a different facility. This different facility may also be visited by a vehicle that collects and consolidates the pathological specimens at a different facility. This procedure may be repeated several times for each facility until the pathological specimens eventually reach a facility of an appropriate tier.

The MACO algorithm was typically able to achieve a relatively consistent range of compromise solutions with respect to both total travel time and the number of vehicles, but failed to offer a wide range of possible driver autonomy values, with the non-dominated front containing a limited spread in this regard. This is possibly a consequence of including the parameter ℓ_b in the route construction phase of the search process. The value assigned to the parameter ℓ_b is independent of previous route constructions and therefore a solution consisting of routes that exhibit small driver autonomy values may contain a route that has a relatively large driver autonomy value, resulting in a poor overall driver autonomy objective function value.

The MACO algorithm of Chapter 10 does not *explicitly* model the minimisation of the number of vehicles. The algorithm was nevertheless able to uncover a diverse set of solutions with respect to the number of vehicles. The combinatorial complexity of adding more objectives to an already difficult problem, will inevitably result in an inherent decline in solution quality and so it is promising to note that it is not explicitly required to implement the minimisation of the number of vehicles.

The numerous constraints of the TVRPGC proved too troublesome to accommodate in restrictive form during the solution search process, as a considerable number of solutions would have to be discarded due to constraint violations. The MACO algorithm designed in Chapter 10 for the TVRPGC is still in its infancy, however, and therefore requires considerable effort in terms of expansion and improvement. Accordingly, future solution implementations should aim to incorporate more involved constraint handling techniques and penalty functions in an attempt to better guide the MACO algorithm through its exploration of the solution space. Future solution implementations may also focus on the improvement of the sequence-fix function described in §11.5 — it may need to incorporate a forward looking intuition during the alterations of routes, as the best sequence fixing solution during the current iteration may be a poor solution when resolving other sequence infeasibilities (the moves are not independent of one another). The algorithm should also be tested against a more tier-diverse TVRPGC instance as the George and Western Cape instances consist primarily of facilities of tier type zero, resulting in a limited number of facilities that may act as consolidation points. More tier-diverse instances may be able to benefit more significantly from the cross-docking capabilities of the MACO algorithm.

The MACO algorithm incorporating a clustering phase with respect to the Western Cape TVRPGC instance was able to yield considerably more competitive solutions than those yielded by the same MACO algorithm when applied to the George TVRPGC instance. As previously mentioned, this may be attributed to the clusters each containing more facilities. The combinatorial complexity and corresponding combinatorial explosion of the TVRPGC may, however, also be responsible for the relative competitiveness in solution quality with respect to incorporating a clustering phase as opposed to considering the instance in its entirety. The industry partner associated with this case study does, in fact, provide pathology healthcare services throughout South Africa, and its national network contains more than 6 000 facilities. It is anticipated that the clustering component would be an integral component of the solution procedure if the full national instance were to be considered. Furthermore, there are prevalent differences in the customer locations due to the provincial organisation of the industry partner. Its provincial departments function relatively independently from the national overarching organisation, possibly allowing for high-quality clusters to be generated by the clustering algorithm.

13.5 Chapter summary

The MACO algorithm of §10.4 was applied in this chapter to the Western Cape TVRPGC instance of Chapter 12. The experimental setup of this application was described in §11.1.

The MACO algorithm was able to achieve a considerable improvement in solution quality with respect to the three objective function values of §9.3.3 over those of the current industry partner's vehicle routing implementation. The results were elaborated upon and translated into potential financial savings in §13.2, with the largest potential financial improvement being experienced in the form of a reduction of total travel time due to the incorporation of the global cross-docking component.

Similarly, the TVRPGC instance was additionally partitioned into ten clusters by means of the clustering methodology described in §8.5. The results obtained with and without incorporating a clustering phase in the TVRPGC solution approach were compared in respect of solution quality and computation time in §13.3. Finally, the results obtained in §13.2 and §13.3 were discussed briefly in §13.4, with a focus on possible future algorithmic improvements of the MACO algorithm in respect of its application to larger instances of the TVRPGC.

Part V

Conclusion

CHAPTER 14

Summary of Dissertation Contributions

Contents

This chapter consists of two sections. A summary of the research documented in this dissertation is provided in §14.1. This is followed in §14.2 by an appraisal of the contributions made in this dissertation.

14.1 Dissertation summary

In the introductory chapter to this dissertation, a brief description of the benefits of healthcare to civilisation was presented, drawing attention to the large discrepancies of resources available in this respect between developed and developing nations. An informal problem description of the novel vehicle routing problem considered in this dissertation, called the TVRPGC, was presented in the second section of the chapter. This was followed by an elucidation of the objectives that were to be pursued during the course of work towards the dissertation, as well as a delimitation of the scope of the research undertaken in this dissertation. The chapter concluded with a brief description of the organisation of the material in the dissertation.

The main body of this dissertation comprised twelve further chapters (up to the chapter preceding the current chapter). The twelve chapters were partitioned into three parts. The first such part was a literature review and consisted of three chapters. A general introduction to and brief history of the VRP was presented in the first chapter of Part I, Chapter 2. The classical VRP was discussed in some detail and the numerous variations on the VRP that prevail in the literature today were also reviewed. These variations were further elaborated upon with respect to the underlying network structure, the type of transportation requests accommodated, the intra-route constraints enforced, the vehicle fleet composition available, numerous inter-route constraints and finally the optimisation objectives pursued.

The second chapter of Part I, Chapter 3, was devoted to a literature study of specifically the CVRP — the archetypal VRP — as this problem forms the foundation on which a model for the TVRPGC was established later in the dissertation, in fulfilment of Dissertation Objective I of §1.3. Numerous solution approaches that have been proposed in the literature for solving the CVRP were described in some detail. The chapter opened with a presentation of the most popular model formulations of the CVRP available in the literature. This was followed by a review of various classical and modern exact solution approaches that have been proposed for solving these mathematical models. The discussion then progressed towards approximate solution approaches that have been applied to the CVRP, beginning with a number of classical heuristics and ending with more powerful metaheuristics and hybrids of these. The closing section of the chapter was devoted to a discussion on appropriate measures of the quality of solutions returned by these approximate solution approaches.

The literature on the problem of data clustering was reviewed in the final chapter of Part I, Chapter 4. This review was included because one of the objectives in this dissertation was to determine the desirability of partitioning large instances of the CVRP (and the subsequent TVRPGC) into smaller, more manageable instances by following a clustering approach with respect to the customers. The various clustering approaches available in the literature were elaborated upon. The important features of any clustering method were described, including the determination of the number of clusters to implement, validation of the resulting clustering and a methodology for the comparison of the quality of results returned by various clustering algorithms. The chapter closed with a review of admissibility criteria that have been suggested in the literature for assessing the sensitivity and stability of clustering algorithm results.

Part II of this dissertation contained four chapters which were devoted to the formulation of the CVRP, its relevant solution methodologies and finally the use of clustering algorithms to partition the customer sets of large, real-life CVRP instances into smaller, more manageable sub-problems. This part stands in fulfilment of Dissertation Objective II of §1.3. The first of these four chapters, Chapter 5, was devoted to the establishment of a suitable mathematical model for the CVRP which could be used as the foundation upon which to build a model for the TVRPGC. The chapter contained a brief model assumptions section, a derivation of the particular mathematical model for the CVRP adopted in the dissertation and a validation of the model in respect of a well-known benchmark test instance. The results obtained by and the viability of an exact model solution approach were further discussed within the context of three CVRP instances of varying complexity, highlighting the need for an approximate model solution approach.

Two metaheuristics were subsequently employed in this dissertation for solving the CVRP approximately. Descriptions of these two methods, namely an ACO algorithm and a GA, were included in the second chapter of Part II, Chapter 6. Implementation details pertaining to the ACO algorithm were presented, and this was accompanied by descriptions of its key components (its pheromone updating mechanisms, its heurisitc matrix initialisation and its tour refinement methods). The ACO implementation included slight variations on pheromone update and heuristic matrix initialisation methods from the literature. A selection of these components were elaborated upon by means of pseudo-code descriptions, highlighting the particular algorithmic implementation process that was followed to solve instances of the CVRP. The GA, as well as its constituent components (the method of chromosome representation adopted, the method of population initialisation and chromosome selection employed, and the nature of the various crossover and mutation operators included), was described in a similar style.

The third chapter of Part II, Chapter 7, contained a report on an extensive set of algorithmic parameter evaluation experiments performed in respect of both the GA and the ACO algorithm of Chapter 6. These experiments took the form of a parameter sensitivity analysis and were performed in the context of the same three benchmark instances as those considered in Chapter 5. The results returned by the two algorithms upon implementation of the best-suited parameter values emanating from the parameter evaluation experiments were subsequently compared, and this comparison was followed by a brief discussion of the relative algorithmic performances.

A clustering approach toward solving large CVRP instances was proposed and tested in the final chapter of part II, Chapter 8. Key features of the clustering algorithm employed (such as the determination of the number of clusters and the validation of the clustering technique) were presented. The inclusion of this clustering phase within the CVRP solution procedure was validated in the context of the same benchmark CVRP instances as before and the results were briefly discussed.

Part III was the heart of the dissertation and was devoted to the mathematical modelling of and solution approaches designed for the TVRPGC. The first chapter of Part III, Chapter 9, saw the establishment of a formal mathematical model for the TVRPGC in the form of a tri-objective combinatorial optimisation problem, in fulfilment of Dissertation Objective III of §1.3. The mathematical model formulation was derived in general and then validated in respect of a small, hypothetical test instance. Thorough descriptions were included of the various model parameters, model variables, model objectives and constraints employed. The model's versatility and potential in terms of global cross-docking capability were highlighted. The need for an approximate model solution approach was finally established in view of the model complexity in the closing sections of the chapter, in fulfilment of Dissertation Objective V of §1.3.

A newly designed tri-objective ant colony optimisation algorithm was proposed in the second chapter of Part II, Chapter 10, for solving instances of the TVRPGC approximately, in fulfilment of Dissertation Objective VI of §1.3. The focus in the chapter was on descriptions of the key aspects of the algorithm (such as its pheromone update mechanisms, its route construction methods, and its method of archiving solutions).

A parameter sensitivity analysis was carried out in respect of a real-life regional instance of the TVRPGC in the final chapter of Part III, Chapter 11. The test instance was based on the vehicle routing of the pathology healthcare service provider within the George region of the South African Southern Cape, called the George TVRPGC instance, in partial fulfilment of Dissertation Objectives IV and VII §1.3. The results of this analysis were presented and discussed, after which a suitable set of parameter values was established for inclusion in the solution methodology of Chapter 10. As was done in Chapter 8 for the CVRP, the suitability of including a clustering phase in the TVRPGC solution methodology was also evaluated in respect of the George TVRPGC instance, in partial fulfilment of Dissertation Objective VIII of §1.3. The chapter closed with a discussion on the quality of the results returned by the MACO algorithm in respect of the TVRPGC.

Part IV of this dissertation consisted of a further two chapters focused on the application of the MACO algorithm of Chapter 10 to a real-life instance of the TVRPGC. The first chapter of this part, Chapter 12, contained a detailed description of the vehicle routes implemented for pathological specimen collection and delivery by a real pathology healthcare service provider in the South African Western Cape, in fulfilment of Dissertation Objective IV §1.3. All the relevant information pertaining to this particular instance of the TVRPGC, called the Western Cape TVRPGC instance, was presented in such a manner so as to allow future researchers to replicate the study while still maintaining client confidentiality.

The second chapter of the part, Chapter 13, was dedicated to the application of the MACO algorithm to the Western Cape TVRPGC instance, in further fulfilment of Dissertation Objective VIII of §1.3. The results of this case study were elaborated upon and translated into potential financial benefits. The trade-off between solution quality and the computational expense of incorporating a clustering phase in the solution procedure was also investigated within the context of the Western Cape TVRPGC instance in final fulfilment of Dissertation Objective VIII. The chapter closed with a discussion of the quality of the results obtained by the MACO algorithm in respect of the Western Cape TVRPGC instance as well as possible improvements that may be made to the MACO algorithm so as to be able to solve larger instances of the TVRPGC.

14.2 Appraisal of dissertation contributions

The main contributions of this dissertation are eight-fold. This section contains a documentation and appraisal of these contributions.

Contribution 1 *A thorough overview of the numerous solution approaches adopted in the literature with respect to generating solutions to instances of the CVRP.*

The overarching aim of this dissertation was to provide a suitable solution methodology for a new variant of the classical VRP aimed at the resolution of the pathological specimen transportation requirements of a tiered pathology service provider. The requirements of this transportation problem do not coincide with any of the numerous VRP formulations in the literature. In this dissertation, both an exact solution approach and an approximate solution approach were developed for the TVRPGC, with both solution methodologies heavily drawing inspiration from similar techniques reported in the literature for the archetypal CVRP. This required a thorough literature review of the available CVRP model formulations and model solution approaches.

Contribution 2 *An evaluation of several GA operators that have reportedly yielded high-quality results in the literature within the context of the CVRP.*

Numerous suggestions as to crossover and mutation operators may be found in the literature for GAs designed to solve CVRP instances. A thorough parameter and operator sensitivity analysis was performed in this dissertation with respect to the more popular of these operators available in the literature. The result was a recommendation as to the best combination of these operators as well as which of the operators are considered more competitive with respect to solution quality when applied in isolation to three well-known CVRP benchmark instances.

Contribution 3 *An investigation into the desirability of incorporating a clustering phase in the standard approximate solution methodology for the CVRP.*

Despite the relatively poor results obtained when incorporating a clustering of customers phase in the solution approaches implemented in this dissertation when solving instances of the CVRP, such an endeavour leads to significantly reduced model solution times. The clustering methodology employed in this dissertation obtained high-quality clusters and is flexible enough to be transferred to several other fields of research.

Contribution 4 *A formalisation of the specimen transportation requirements of a pathology healthcare service provider and its corresponding tiered-facility pathological facility network.*

Several meetings with personnel from the pathology healthcare service provider associated with the case study of Chapter 12 were held in an attempt to capture all the subtle requirements and objectives that should be incorporated in a pathological specimen transportation network serving the organisation. This was a difficult process since the transportation network employed by the organisation is operated by several independent logistics companies. Prior to the aforementioned meetings there was no common agreement between the management at the organisation's department for the Western Cape region of the required specimen transportation operations in its network. After these meetings, however, detailed requirements for the operation of this transportation problem, called the TVRPGC, could be formalised.

Contribution 5 *The establishment of a formal mathematical model and an accompanying exact solution methodology for the TVRPGC.*

The introduction of a new VRP variant requires a formal mathematical model that describes all the requirements of the network in an unambiguous manner so as to avoid misinterpretation of the problem specifics by future researchers. The most important contribution of this dissertation was the establishment of a validated mathematical model for the TVRPGC containing all the constraints and objectives required for specimen collection and delivery in a tiered pathological facility network. This model was submitted for publication to the European Journal of Operational Research [404].

Contribution 6 *The establishment of an approximate solution methodology for the TVRPGC problem, capable of finding high-quality solutions to realistic problem instances.*

The swarm-intelligence based method of MACO was implemented as an approximate TVRPGC solution approach in this dissertation. The method of MACO is flexible and may be employed to solve TVRPGC instances of varying complexity. The MACO algorithm of Chapter 10 incorporated several novel features in its development, most of which arose from the combinatorial complexity of the TVRPGC. After performing an extensive parameter sensitivity analysis to determine the best configuration of algorithmic parameter values, the MACO algorithm was applied to a real-life TVRPGC instance within the Western Cape region.

Contribution 7 *A cost-benefit analysis of adopting the vehicle routing suggested by the MACO algorithm as opposed to the current practice implemented by a real pathology healthcare service provider.*

The MACO algorithm of Chapter 10 was implemented with respect to the Western Cape TVRPGC instance for which data were provided by a major South African pathology healthcare service provider. The MACO algorithm was able to generate a non-dominated front consisting of nine high-quality solutions, from which the management of the pathology healthcare service provider in question would be able to choose an alternative, based on its subjective preference in respect of trade-offs realised between the three objectives pursued in this dissertation. The benefits, translated into financial savings, were presented with respect to one of the solutions generated within this non-dominated front. The potential financial savings resulting from the particular vehicle routing recommendation were considerable.

Contribution 8 *An investigation into the desirability of incorporating a clustering phase in the approximate solution methodology established for the TVRPGC.*

The combinatorial complexity of the TVRPGC, and more specifically, the global cross-docking component of the problem lends itself to incorporating a clustering phase within the approximate solution approach of Contribution 6. The MACO algorithm incorporating a clustering phase was able to generate solutions of a relatively high quality in a fraction of the computation time required by the MACO algorithm when solving the Western Cape TVRPGC instance in its entirety. This clustering approach is predicted to be of real practical value for future routing endeavours related to the national TVRPGC instance of the participating pathology service provider, whose entire transportation network contains more than 6 000 facilities.

A research paper based on Contributions 6–8 is currently in preparation and will be submitted for possible publication in Computers and Industrial Engineering.

CHAPTER 15

Possible Future Work

Contents

In the words of Leanardo da Vinci "*L'arte non mai finita, solo abbandonata*[1]" [54]. A number of aspects were identified during the process of completing the research reported in this dissertation as areas of possible improvement and enhancement of the modelling approach adopted. This chapter contains a summary of these aspects in the form of suggestions with respect to possible future research which may be pursued as follow-up work to the contributions of this dissertation, in fulfilment of Dissertation Objective IX of §1.3.

15.1 Consider a larger TVRPGC instance

In this study, a novel variant of the well-known CVRP was proposed in the form of the TVRPGC. A mathematical model for the TVRPGC was validated against a small, hypothetical test-instance by which it was possible to demonstrate the model's flexibility as well as potential in respect of global cross-docking and tiered facility considerations.

The approximate solution approach of Chapter 10 was, however, tested in the context of the Western Cape TVRPGC instance for which data were provided by a national pathology health-care service provider. A more desirable validation approach would be to apply the MACO algorithm to the national tiered facility transportation network of this pathology healthcare

[1] Art is never finished, only abandoned.

service provider so that the performance of the algorithm may be compared against the current national industry *status quo* as there does not currently exist an academic standard for the TVRPGC. The aforementioned national TVRPGC instance would offer more complexity with respect to the number of customers and the road network.

15.2 Introduce benchmark instances for the TVRPGC

Related to the suggestion of the previous section, a suite of TVRPGC benchmark instances is, in fact, required in order to enable future researchers to compare and standardise their TVRPGC-related research. The typical approach would be to modify industry data so as to obtain such test instances. Due to the sensitive nature of pathology healthcare service provider data pertaining to patient specimens, more hypothetical test instances may, however, have to be settled upon.

The design of TVRPGC test instances should accommodate a potential benefit in respect of global cross-docking as this is the defining feature of the TVRPGC. This would require intelligent spacing of customers in terms of their relative positioning with respect to higher tiered facilities and the distribution of the tiers would have to be chosen judiciously so as to encourage global cross-docking which, for larger instances, may only be realisable through extensive testing of the proposed instances. The diversity of tiers should also be encouraged as to fully utilise the global cross-docking component, because it is anticipated that real-life TVRPGC instances may lack the required diversity with respect to the number of facilities that may serve as consolidation points.

TVRPGC test instances should also offer a diverse range of complexity with respect to the number of customers, the locations of the customers and the required amount of global cross-docking in order to be considered suitable as benchmarks. The TVRPGC instance should finally also include complete information specification such as the time windows and vehicle arrival capacities of facilities.

15.3 Design and implement a decision support system

The mathematical model of Chapter 9 was implemented in both CPLEX (for implementation of the exact solution approach) and in R (for implementation of the approximate solution approach). These implementations are, however, not very generic and a number of changes would need to be made manually to the physical code of the implementations in order to accommodate different test instances.

It is therefore suggested that a generic, user-friendly *decision support system* (DSS), capable of providing vehicle routing recommendations aimed at pursuing trade-offs between minimising total travel time, minimising driver autonomy levels and minimising the number of vehicles utilised, should be developed. This DSS should draw on the techniques researched in Chapters 10 and 11, and its working should be based on the model formulation of Chapter 9.

The user of such a DSS should be able to import the data and system specifications of a particular TVRPGC instance via Microsoft Excel spreadsheets, which may then be accessed by the DSS in order to generate vehicle routes. The output of the DSS should be several vehicle routes from which the decision maker may select one according to his or her personal preferences in respect of the realisation of an acceptable trade-off between the aforementioned objectives. Such a DSS may potentially be very profitable for personnel at the pathology healthcare service provider (or any other organisation that implements tiered facilities within its transportation network or would benefit from the incorporation of a global cross-docking component within its transportation network).

15.4 Improve constraint handling techniques

As mentioned in §7.6, the approximate solution methods of Chapter 6 may potentially also benefit from the consideration of infeasible solutions during their execution. There are numerous ways in which such a feature may be incorporated into a metaheuristic search process, with the most common approach being to implement a penalty function. The capabilities of traversing infeasible regions of the solution space is expected to aid a search algorithm in a tightly constrained CVRP instance with respect to providing high-quality solutions. Accordingly, the approximate solution approaches of Chapter 6 may benefit considerably with the inclusion of a penalty function with respect to capacity constraint violations.

Since the mathematical model proposed in Chapter 9 was the first of its kind, much effort was expended to formalise the constraints in the form of linear inequalities. There may, however, exist more powerful and elegant mathematical model formulations for the TVRPGC. The constraint handling techniques with respect to the exact solution approach may also be improved upon so as to reduce the initial number of constraints during the BaB process and adding the constraints not yet considered in an intelligent manner only when they are required so as to reduce the computational burden.

Alternatively, a completely different exact modelling approach may be undertaken, such as implementing a constraint programming exact solution approach, for example, as opposed to the linear integer programming methodology adopted in this dissertation.

15.5 Explicitly model individual specimen constraints

As mentioned in §1.4, certain scope limitations were adopted during the course of the research presented in this dissertation. One such scope limitation was the exclusion of specimen-specific modelling considerations. As a result, certain features of pathological specimens were not modelled explicitly within the TVRPGC model proposed in this dissertation, such as the deterioration over time of individual specimens or processing and storage requirements of individual specimens. This simplifying scope delimitation was adopted based on the magnitude of possible model variations — to model such variations in an already complex combinatorial problem was deemed unwise. Building on the mathematical model foundation laid in this dissertation, future model variations may, however, allow for such intricate detail to be included explicitly as constraints.

15.6 Incorporate goal-programming techniques

The exact solution approach adopted in this dissertation involved fixing certain objectives throughout the solution search. The three objectives pursued in this dissertation may, however, be considered simultaneously if a goal-programming methodology were to be adopted during the search process. Due to the combinatorial complexity and the relatively large computational burden associated with considering only a single objective, goal-programming was not considered in this dissertation.

15.7 Model the stochastic nature of specimen demand

Due to scope limitations, the final modelling consideration excluded was the stochastic nature of demand present within the transportation network of the model. It was assumed that the demand exhibited by customers and their respective locations are known *a priori*. In some instances, facilities may, however, not experience a demand for specimen collection or conversely an otherwise unconsidered facility may unexpectedly experience demand for specimen collection. Future work with respect to this model feature may involve including a noise calibration threshold function in the approximate solution approach (*i.e.* being able to determine when enough change has been introduced into the system input values so as to warrant altering the solution being implemented) or to consider demand fluctuations within the transportation network in a stochastic manner.

15.8 Model the number of vehicles within the MACO

The MACO algorithm of Chapter 10 does not explicitly consider the number of vehicles utilised as an objective when constructing solutions, although, the archiving component of the algorithm does accommodate the number of vehicles utilised as being essentially variable. Future approximate solution techniques for the TVRPGC may explicitly incorporate minimising the number of vehicles used. This was an objective of the pathology healthcare service provider (and was also included as an explicit objective in the mathematical model of Chapter 9). More variety in the solutions presented (with respect to the number of vehicles implemented) may indeed be beneficial to the decision makers of the organisation.

15.9 Improve modelling of the sequence-fix function

As mentioned in §11.5, the MACO algorithm proposed in Chapter 10 is still in its infancy and requires a considerable amount of effort to be expended in pursuit of the improvement of its operators. One such avenue of interest is the modelling of the sequence-fix function, which forms a core component within the MACO algorithm. Future endeavours may aim to incorporate a forward-looking intuition into the sequence-fix function, as certain operations may be beneficial during the current iteration but may, however, lead to solutions of a poor quality during later iterations.

15.10 Improve modelling of the lower-bound parameter

The MACO algorithm of Chapter 10 was unable to generate a wide range of solutions with respect to the driver autonomy objective, as previously mentioned in §11.5. The parameter ℓ_b may be responsible for this shortcoming as it does not consider previous route constructions in the assignment of its values. Future research may benefit from incorporating a backward-looking intuition into the parameter value assignment with respect to the driver autonomy objective pursued in this dissertation.

15.11 Develop an additional approximate solution approach

As mentioned, the results generated by the MACO algorithm of Chapter 10 are difficult to validate as there are no established benchmarks for the TVRPGC (due to the novelty of the problem). Hence there are considerable uncertainties related to the quality of the results generated by the MACO algorithm.

The results returned by the MACO algorithm would benefit from being compared with results returned by an alternative approximate solution approach for the TVRPGC. The results presented in Chapter 7 suggest that GAs are well suited to solving VRPs. It is therefore suggested that a multi-objective GA be tailor-designed for the TVRPGC. This would provide an additional validation mechanism for the results generated by both algorithms.

References

[1] Achuthan NR, Caccetta L & Hill SP, 1996, *A new subtour elimination constraint for the vehicle routing problem*, European Journal of Operational Research, **91(3)**, pp. 573–586.

[2] Achuthan NR, Caccetta L & Hill SP, 2003, *An improved branch-and-cut algorithm for the capacitated vehicle routing problem*, Transportation Science, **37(2)**, pp. 153–169.

[3] Aggarwal CC & Reddy CK, 2013, *Data clustering: Algorithms and applications*, CRC Press, Boca Raton (FL).

[4] Ai TJ & Kachitvichyanukul V, 2009, *A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery*, Computers and Operations Research, **36(5)**, pp. 1693–1702.

[5] Alander JT, 1992, *On optimal population size of genetic algorithms*, Proceedings of the 6th Annual Conference of Computer Systems and Software Engineering, The Hague, pp. 65–70.

[6] Alpert CJ & Kahng AB, 1997, *Splitting an ordering into a partition to minimize diameter*, Journal of Classification, **14(1)**, pp. 51–74.

[7] Alrabea A, Senthilkumar A, Al-Shalabi H & Bader A, 2013, *Enhancing k-means algorithm with initial cluster centers derived from data partitioning along the data axis with PCA*, Journal of Advances in Computer Networks, **1(2)**, pp. 137–142.

[8] Altinkemer K & Gavish B, 1991, *Parallel savings based heuristics for the delivery problem*, Operations Research, **39(3)**, pp. 456–469.

[9] AmPath, 2015, *Ampath Pathologists*, [Online], [accessed 10th September 2017], Available at: `https://www.ampath.co.za/`.

[10] Ancer, J, 2015, *Crisis in laboratory threaten's the nations health*, [Online], [Accessed 28th February 2015], Available at: `http://mg.co.za/article/2015-01-09-crisis-in-laboratory-threatens-the-nations-health`.

[11] Angelo JS & Barbosa HJ, 2011, *On ant colony optimization algorithms for multiobjective problems*, pp. 53–75 in Ostfeld A (Ed), *Ant colony optimisation methods and applications*, Intech, Rijeka.

[12] Applegate DL, Bixby RE, Chvatal V & Cook WJ, 2011, *The traveling salesman problem: A computational study*, Princeton University Press, Princeton (NJ).

[13] Araque JR, Hall LA & Magnanti TL, 1990, *Capacitated trees, capacitated routing, and associated polyhedra*, MIT Press, Cambridge (MA).

[14] Archetti C, Feillet D, Hertz A & Speranza MG, 2009, *The capacitated team orienteering and profitable tour problems*, Journal of the Operational Research Society, **60(6)**, pp. 831–842.

[15] Archetti C, Hertz A & Speranza MG, 2007, *Metaheuristics for the team orienteering problem*, Journal of Heuristics, **13(1)**, pp. 49–76.

[16] Arimond G & Elfessi A, 2001, *A clustering method for categorical data in tourism market segmentation research*, Journal of Travel Research, **39(4)**, pp. 391–397.

[17] Auger A, Bader J, Brockhoff D & Zitzler E, 2009, *Theory of the hypervolume indicator: Optimal μ-distributions and the choice of the reference point*, Proceedings of the 10th ACM SIGEVO Workshop on Foundations of Genetic Algorithms, Orlando (FL), pp. 87–102.

[18] Augerat P, 1995, *Approche polyèdrale du problème de tournées de véhicules*, PhD Thesis, Institut National Polytechnique de Grenoble, Grenoble.

[19] Babu GP & Murty MN, 1993, *A near-optimal initial seed value selection in k-means means algorithm using a genetic algorithm*, Pattern Recognition Letters, **14(10)**, pp. 763–769.

[20] Baeza-Yates RA, 1992, *Introduction to data structures and algorithms related to information retrieval*, pp. 18–33 in Baeza-Yates RA & Frakes WB (Eds), *Information retrieval: Data structures and algorithms*, University of Arizona Press, Tucson (AZ).

[21] Bai L, Liang J, Sui C & Dang C, 2013, *Fast global k-means clustering based on local geometrical information*, Information Sciences, **245**, pp. 168–180.

[22] Bailey D, 1989, *A historical survey of solution by functional iteration*, Mathematics Magazine, **62(3)**, pp. 155–166.

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

[24] Baker FB & Hubert LJ, 1975, *Measuring the power of hierarchical cluster analysis*, Journal of the American Statistical Association, **70(349)**, pp. 31–38.

[25] Baker FB & Hubert LJ, 1976, *A graph-theoretic approach to goodness-of-fit in complete link hierarchical clustering*, Journal of the American Statistical Association, **71(356)**, pp. 870–878.

[26] Baldacci R, Battarra M & Vigo D, 2008, *Routing a heterogeneous fleet of vehicles*, pp. 3–27 in Golden BL, Raghavan S & Amin WE (Eds), *The vehicle routing problem: Latest advances and new challenges*, Springer, New York (NY).

[27] Baldacci R, Christofides N & Mingozzi A, 2008, *An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts*, Mathematical Programming, **115(2)**, pp. 351–385.

[28] Baldacci R, Mingozzi A & Roberti R, 2011, *New route relaxation and pricing strategies for the vehicle routing problem*, Operations Research, **59(5)**, pp. 1269–1283.

[29] Balinski ML & Quandt RE, 1964, *On an integer program for a delivery problem*, Operations Research, **12(2)**, pp. 300–304.

[30] Ball GH & Hall DJ, 1965, *ISODATA, a novel method of data analysis and pattern classification*, (Unpublished) Technical Report AD0699616, Stanford Research Institution, Stanford (CA).

[31] Bandyopadhyay S, 2005, *Simulated annealing using a reversible jump Markov chain Monte Carlo algorithm for fuzzy clustering*, IEEE Transactions on Knowledge and Data Engineering, **17(4)**, pp. 479–490.

[32] Barán B & Schaerer M, 2003, *A multiobjective ant colony system for vehicle routing problem with time windows*, Proceedings of the Applied Informatics Conference of 2003, Innsbruck, pp. 97–102.

[33] Barton F, 2016, *Western Cape Green Point Business Manager at NHLS*, [Personal Communication], Contactable at francois.barton@nhls.ac.za.

[34] Basak J & Krishnapuram R, 2005, *Interpretable hierarchical clustering by constructing an unsupervised decision tree*, IEEE Transactions on Knowledge and Data Engineering, **17(1)**, pp. 121–132.

[35] Basak J & Krishnapuram R, 2002, *Personalized product recommendation*, US Patent Application 10/138,857.

[36] Baxter J, 1984, *Depot location: A technique for the avoidance of local optima*, European Journal of Operational Research, **18(2)**, pp. 208–214.

[37] Beale E, 1969, *Euclidean cluster analysis*, Scientific Control Systems Limited, London.

[38] Beasley J & Christofides N, 1997, *Vehicle routing with a sparse feasibility graph*, European Journal of Operational Research, **98(3)**, pp. 499–511.

[39] Beasley JE, 1983, *Route-first cluster-second methods for vehicle routing*, Omega, **11(4)**, pp. 403–408.

[40] Bechikh S, Said LB & Ghédira K, 2011, *Searching for knee regions of the Pareto front using mobile reference points*, Soft Computing, **15(9)**, pp. 1807–1823.

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

[42] Berger J & Barkaoui M, 2003, *A new hybrid genetic algorithm for the capacitated vehicle routing problem*, Journal of the Operational Research Society, **54(12)**, pp. 1254–1262.

[43] Berkhin P, 2006, *A survey of clustering data mining techniques*, pp. 25–71 in Kogan J, Nicholas C & Teboulle M (Eds), *Grouping multidimensional data*, Springer, Berlin.

[44] Berry MJ & Linoff G, 1997, *Data mining techniques: For marketing, sales, and customer support*, John Wiley & Sons, New York (NY).

[45] BestLogisticsGuide, 2007, *Logistics history shows the benefit and importance of logistics*, [Online], [accessed 11th October 2016], Available at: http://www.bestlogisticsguide.com/logistics-history.html.

[46] Bhattacherjee V, Mukhopadhyay P, Singh S, Johnson C, Philipose JT, Warner CP, Greene RM & Pisano MM, 2007, *Neural crest and mesoderm lineage dependent gene expression in orofacial development*, Differentiation, **75(5)**, pp. 463–477.

[47] Blum C, 2005, *Ant colony optimization: Introduction and recent trends*, Physics of Life Reviews, **2(4)**, pp. 353–373.

[48] Boffey B, García FRF, Laporte G, Mesa JA & Pelegrín BP, 1995, *Multiobjective routing problems*, Top, **3(2)**, pp. 167–220.

[49] Boland N, Dethridge J & Dumitrescu I, 2006, *Accelerated label setting algorithms for the elementary resource constrained shortest path problem*, Operations Research Letters, **34(1)**, pp. 58–68.

[50] Bonabeau E, Dorigo M & Theraulaz G, 1999, *Swarm intelligence: From natural to artificial systems*, Oxford University Press, Oxford.

[51] Bonabeau E, Dorigo M & Theraulaz G, 2000, *Inspiration for optimization from social insect behaviour*, Nature, **406(6791)**, pp. 39–42.

[52] Bowerman R, Hall B & Calamai P, 1995, *A multi-objective optimization approach to urban school bus routing: Formulation and solution method*, Transportation Research Part A: Policy and Practice, **29(2)**, pp. 107–123.

[53] Bradley P, Bennett K & Demiriz A, 2000, *Constrained k-means clustering*, (Unpublished) Technical Report MSR-TR-2000-65, Microsoft Research.

[54] BrainyQuotes, 2015, *Leonardo da Vinci Quotes*, [Online] , [accessed 17th November 2016], Available at: `https://www.brainyquote.com/citation/quotes/quotes/l/leonardoda380290.html?ct=Leonardo+da+Vinci`.

[55] Bramel J & Simchi-Levi D, 1995, *A location based heuristic for general routing problems*, Operations Research, **43(4)**, pp. 649–660.

[56] Bramel J & Simchi-Levi D, 2001, *Set-covering-based algorithms for the capacitated VRP*, pp. 85–108 in Toth P & Vigo D (Eds), *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia (PA).

[57] Bräysy O, 2001, *Local search and variable neighborhood search algorithms for the vehicle routing problem with time windows*, PhD Thesis, University of Vaasa, Vaasa.

[58] Bräysy O & Gendreau M, 2005, *Vehicle routing problem with time windows, part I: Route construction and local search algorithms*, Transportation Science, **39(1)**, pp. 104–118.

[59] Brock G, Pihur V, Datta S & Datta S, 2011, *clValid, an R package for cluster validation*, Journal of Statistical Software, **25**, pp. 1–22.

[60] Brockhoff D, Wagner T & Trautmann H, 2012, *On the properties of the R2 indicator*, Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, Philadelphia (PA), pp. 465–472.

[61] Brown DE, 1998, *The regional crime analysis program (RECAP): A framework for mining data to catch criminals*, Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics 1998, pp. 2848–2853.

[62] Brown DE & Huntley CL, 1992, *A practical application of simulated annealing to clustering*, Pattern Recognition, **25(4)**, pp. 401–412.

[63] Brusco MJ & Stahl S, 2006, *Branch-and-bound applications in combinatorial data analysis*, Springer, Berlin.

[64] Bullnheimer B, Hartl RF & Strauss C, 1997, *A new rank based version of the Ant System: A computational study, Working Paper No. 1*, SFB Adaptive Information Systems and Modelling in Economics and Management Science.

[65] Bullnheimer B, Hartl RF & Strauss C, 1999, *An improved ant system algorithm for the vehicle routing problem*, Annals of Operations Research, **89**, pp. 319–328.

[66] Bullnheimer B, Hartl RF & Strauss C, 1999, *Applying the ant system to the vehicle routing problem*, pp. 285–296 in Voss S, Martello S, Osman I & Roucairol C (Eds), *Meta-heuristics: Advances and trends in local search paradigms for optimization*, Springer, Boston (MA).

[67] Burke EK, Gustafson S & Kendall G, 2004, *Diversity in genetic programming: An analysis of measures and correlation with fitness*, IEEE Transactions on Evolutionary Computation, **8(1)**, pp. 47–62.

[68] BusinessTech, 2015, *How much money your life is actually worth in SA*, [Online], [accessed 11th October 2016], Available at: `http://businesstech.co.za/news/general/102381/how-much-money-your-life-is-actually-worth/`.

[69] Caliński T & Harabasz J, 1974, *A dendrite method for cluster analysis*, Communications in Statistics — Theory and Methods, **3(1)**, pp. 1–27.

[70] Campbell AM, Clarke LW & Savelsbergh MW, 2001, *Inventory routing in practice*, pp. 309–330 in Toth P & Vigo D (Eds), *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia (PA).

[71] Castro RM, Coates MJ & Nowak RD, 2004, *Likelihood based hierarchical clustering*, IEEE Transactions on Signal Processing, **52(8)**, pp. 2308–2321.

[72] Celebi ME & Kingravi HA, 2015, *Linear, deterministic, and order-invariant initialization methods for the k-means clustering algorithm*, pp. 79–98 in Celebi ME (Ed), *Partitional clustering algorithms*, Springer, Cham.

[73] Chabrier A, 2006, *Vehicle routing problem with elementary shortest path based column generation*, Computers and Operations Research, **33(10)**, pp. 2972–2990.

[74] Chandran N, Narendran T & Ganesh K, 2006, *A clustering approach to solve the multiple travelling salesmen problem*, International Journal of Industrial and Systems Engineering, **1(3)**, pp. 372–387.

[75] Chao IM, 2002, *A tabu search method for the truck and trailer routing problem*, Computers and Operations Research, **29(1)**, pp. 33–51.

[76] Chapelle O, Schölkopf B & Zien A, 2006, *Semi-supervised learning, ser. Adaptive computation and machine learning*, MIT Press, Cambridge (MA).

[77] Charrad M, Ghazzali N, Boiteau V, Niknafs A & Charrad MM, 2014, *NbClust package: Finding the relevant number of clusters in a dataset*, Journal of Statistical Software, **61**, pp. 1–36.

[78] Chen AL, Yang GK & Wu ZM, 2006, *Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem*, Journal of Zhejiang University Science A, **7(4)**, pp. 607–614.

[79] Chen P, Huang HK & Dong XY, 2010, *Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem*, Expert Systems with Applications, **37(2)**, pp. 1620–1627.

[80] Cheng Y, 1995, *Mean shift, mode seeking, and clustering*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **17(8)**, pp. 790–799.

[81] Christofides N, 1976, *The vehicle routing problem*, Revue Francaise d'Automatique, Informatique et Recherche Operationelle, **10**, pp. 55–70.

[82] Christofides N & Eilon S, 1969, *An algorithm for the vehicle-dispatching problem*, Operations Research, **20(3)**, pp. 309–318.

[83] Christofides N, Mingozzi A & Toth P, 1979, *The vehicle routing problem*, Wiley, New York (NY).

[84] Christofides N, Mingozzi A & Toth P, 1981, *Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations*, Mathematical Programming, **20(1)**, pp. 255–282.

[85] Christofides N, Mingozzi A & Toth P, 1981, *State-space relaxation procedures for the computation of bounds to routing problems*, Networks, **11(2)**, pp. 145–164.

[86] Chvátal V, 1973, *Edmonds polytopes and weakly hamiltonian graphs*, Mathematical Programming, **5(1)**, pp. 29–40.

[87] Clarke GU & Wright JW, 1964, *Scheduling of vehicles from a central depot to a number of delivery points*, Operations Research, **12(4)**, pp. 568–581.

[88] Climer S & Zhang W, 2004, *Take a walk and cluster genes: A TSP-based approach to optimal rearrangement clustering*, Proceedings of the 21st International Conference on Machine Learning, Atlanta (GA), pp. 22–30.

[89] Climer S & Zhang W, 2006, *Rearrangement clustering: Pitfalls, remedies, and applications*, Journal of Machine Learning Research, **7**, pp. 919–943.

[90] Coello CAC & Lamont GB, 2004, *Applications of multi-objective evolutionary algorithms*, World Scientific, Singapore.

[91] Coello CAC, Van Veldhuizen DA & Lamont GB, 2002, *Evolutionary algorithms for solving multi-objective problems*, Springer, New York (NY).

[92] Coles C, 2016, *Operational productivity and performance in English NHS acute hopsitals: Unwarranted variations*, [Online], [Accessed 22nd May 2016], Available at:: `https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/499229/Operational_productivity_A.pdf`.

[93] Comaniciu D & Meer P, 1999, *Mean shift analysis and applications*, Proceedings of the 7th IEEE International Conference on Computer Vision, Kerkyra, pp. 1197–1203.

[94] Contardo C, Cordeau J & Gendron B, 2011, *A branch-and-cut-and-price algorithm for the capacitated location-routing problem*, (Unpublished) Technical Report CIRRELT-2011-44, Universit de Montral, Montreal.

[95] Contardo C, Cordeau J.-F & Gendron B, 2013, *An exact algorithm based on cut-and-column generation for the capacitated location-routing problem*, INFORMS Journal on Computing, **26(1)**, pp. 88–102.

[96] Contardo C & Martinelli R, 2014, *A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints*, Discrete Optimization, **12**, pp. 129–146.

[97] Corberán Á & Laporte G, 2013, *Arc routing: Problems, methods and applications*, Society for Industrial and Applied Mathematics, Philadelphia (PA).

[98] Cordeau JF, Gendreau M & Laporte G, 1997, *A tabu search heuristic for periodic and multi-depot vehicle routing problems*, Networks, **30(2)**, pp. 105–119.

[99] Cordeau JF & Maischberger M, 2012, *A parallel iterated tabu search heuristic for vehicle routing problems*, Computers and Operations Research, **39(9)**, pp. 2033–2050.

[100] Cordeau JF, Laporte G & Mercier A, 2001, *A unified tabu search heuristic for vehicle routing problems with time windows*, Journal of the Operational Research Society, **52(8)**, pp. 928–936.

[101] Cortes C & Vapnik V, 1995, *Support-vector networks*, Machine Learning, **20(3)**, pp. 273–297.

[102] Cowgill MC, Harvey RJ & Watson LT, 1999, *A genetic algorithm approach to cluster analysis*, Computers and Mathematics with Applications, **37(7)**, pp. 99–108.

[103] Crainic TG, Perboli G, Mancini S & Tadei R, 2010, *Two-echelon vehicle routing problem: A satellite location analysis*, Procedia — Social and Behavioral Sciences, **2(3)**, pp. 5944–5955.

[104] Crainic TG & Toulouse M, 2010, *Parallel meta-heuristics*, pp. 497–541 in Gendreau M & Potvin J (Eds), *Handbook of metaheuristics*, Springer, Boston (MA).

[105] Crevier B, Cordeau JF & Laporte G, 2007, *The multi-depot vehicle routing problem with inter-depot routes*, European Journal of Operational Research, **176(2)**, pp. 756–773.

[106] Croes GA, 1958, *A method for solving traveling-salesman problems*, Operations Research, **6(6)**, pp. 791–812.

[107] Dantzig GB & Ramser JH, 1959, *The truck dispatching problem*, Management Science, **6(1)**, pp. 80–91.

[108] Datta S & Datta S, 2003, *Comparisons and validation of statistical clustering techniques for microarray gene expression data*, Bioinformatics, **19(4)**, pp. 459–466.

[109] Davies DL & Bouldin DW, 1979, *A cluster separation measure*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **(2)**, pp. 224–227.

[110] Day WH, 1992, *Complexity theory: An introduction for practitioners of classification*, World Scientific Publishers, River Branch (NJ).

[111] De Amorim RC & Mirkin B, 2012, *Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering*, Pattern Recognition, **45(3)**, pp. 1061–1075.

[112] De Aragao MP & Uchoa E, 2003, *Integer program reformulation for robust branch-and-cut-and-price algorithms*, Proceedings of the Mathematical Program: A Conference in Honour of Nelson Maculan, Rio de Janeiro, pp. 56–61.

[113] De Lima IX, 2015, *Capacitated Vehicle Routing Problem Library*, [Online], [accessed 19th October 2016], Available at: `http://vrp.atd-lab.inf.puc-rio.br/index.php/en/`.

[114] De Sauvages FB & Daniel CF, 1790, *Nosologia methodica*, De Tourned Brothers, Amsterdam.

[115] Deb K, 2001, *Multi-objective optimization using evolutionary algorithms*, John Wiley & Sons, Hoboken (NJ).

[116] Deb K, Pratap A, Agarwal S & Meyarivan T, 2002, *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation, **6(2)**, pp. 182–197.

[117] Delgado M, Skármeta AG & Barberá HM, 1997, *A tabu search approach to the fuzzy clustering problem*, Proceedings of the 6th IEEE International Conference on Fuzzy Systems, Barcelona, pp. 125–130.

[118] Dembélé D & Kastner P, 2003, *Fuzzy C-means method for clustering microarray data*, Bioinformatics, **19(8)**, pp. 973–980.

[119] Derigs U, Gottlieb J, Kalkoff J, Piesche M, Rothlauf F & Vogel U, 2011, *Vehicle routing with compartments: Applications, modelling and heuristics*, OR Spectrum, **33(4)**, pp. 885–914.

[120] Desaulniers G, Desrosiers J, Erdmann A, Solomon M, Soumis F, Toth P & Vigo D, 2000, *The VRP with pickup and delivery*, Soceity for Industrial and Applied Mathematics, Philadelphia (PA).

[121] Desrochers M, 1988, *An algorithm for the shortest path problem with resource constraints*, (Unpublished) Technical Report 421A, Centre de Reserche sur les Transports, University de Móntreal, Móntreal.

[122] Desrochers M, Desrosiers J & Solomon M, 1992, *A new optimization algorithm for the vehicle routing problem with time windows*, Operations Research, **40(2)**, pp. 342–354.

[123] Desrochers M & Verhoog T, 1989, *A matching based savings algorithm for the vehicle routing problem*, (Unpublished) Technical Report G-59-04, Les Cahiers du GERAD, University de Montreal, Montreal.

[124] Desrosiers J, Dumas Y, Solomon MM & Soumis F, 1995, *Time constrained routing and scheduling*, Handbooks in Operations Research and Management Science, **8**, pp. 35–139.

[125] Desrosiers J, Soumis F & Desrochers M, 1984, *Routing with time windows by column generation*, Networks, **14(4)**, pp. 545–565.

[126] Dhillon IS & Modha DS, 2000, *Large-scale parallel data mining*, Lecture Notes in Artificial Intelligence, **1759**, pp. 245–260.

[127] DHL, 2007, *The great logistics success story*, [Online], [accessed 11th October 2016], Available at: `https://www.dhl-discoverlogistics.com/cms/en/course/origin/historical_development.jsp`.

[128] Diaz-Gomez PA & Hougen DF, 2007, *Initial population for genetic algorithms: A metric approach*, Proceedings of the 2007 International Conference on Genetic and Evolutionary Methods, Las Vegas (NV), pp. 43–49.

[129] Ding Q, Hu X, Sun L & Wang Y, 2012, *An improved ant colony optimization and its application to vehicle routing problem with time windows*, Neurocomputing, **98**, pp. 101–107.

[130] Disney SM & Towill DR, 2003, *Vendor-managed inventory and bullwhip reduction in a two-level supply chain*, International Journal of Operations and Production Management, **23(6)**, pp. 625–651.

[131] Doerner K, Focke A & Gutjahr WJ, 2007, *Multicriteria tour planning for mobile healthcare facilities in a developing country*, European Journal of Operational Research, **179(3)**, pp. 1078–1096.

[132] Doerner K, Gutjahr WJ, Hartl RF, Strauss C & Stummer C, 2004, *Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection*, Annals of Operations Research, **131(1-4)**, pp. 79–99.

[133] Doerner K, Hartl RF & Reimann M, 2001, *Are COMPETants more competent for problem solving? The case of a multiple objective transportation problem*, European Journal of Operations Research, **16(30)**, pp. 115–141.

[134] Dondo R & Cerdá J, 2007, *A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows*, European Journal of Operational Research, **176(3)**, pp. 1478–1507.

[135] Dondo R, Méndez CA & Cerdá J, 2011, *The multi-echelon vehicle routing problem with cross docking in supply chain management*, Computers and Chemical Engineering, **35(12)**, pp. 3002–3024.

[136] Dopazo J & Carazo JM, 1997, *Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree*, Journal of Molecular Evolution, **44(2)**, pp. 226–233.

[137] Dorigo M, 1992, *Optimization, learning and natural algorithms*, PhD Thesis, Politecnico di Milano, Milan.

[138] Dorigo M & Gambardella LM, 1997, *Ant colony system: A cooperative learning approach to the traveling salesman problem*, IEEE Transactions on Evolutionary Computation, **1(1)**, pp. 53–66.

[139] Dorigo M, Maniezzo V & Colorni A, 1996, *Ant system: Optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, **26(1)**, pp. 29–41.

[140] Dorigo M & Stützle T, 2009, *Ant colony optimization: Overview and recent advances*, pp. 251–285 in Glover F & Kochenberger G (Eds), *Handbook of metaheuristics*, Kluwer Academic Press, Norwell (MA).

[141] Drexl M, 2011, *Branch-and-price and heuristic column generation for the generalized truck-and-trailer routing problem*, Revista de Métodos Cuantitativos para la Economía y la Empresa, **12**, pp. 5–38.

[142] Drexl M, 2012, *Synchronization in vehicle routing — A survey of VRPs with multiple synchronization constraints*, Transportation Science, **46(3)**, pp. 297–316.

[143] Drexl M, 2013, *Applications of the vehicle routing problem with trailers and transshipments*, European Journal of Operational Research, **227(2)**, pp. 275–283.

[144] Drexl M, Rieck J, Sigl T & Press B, 2013, *Simultaneous vehicle and crew routing and scheduling for partial and full-load long-distance road transport*, Business Research, **6(2)**, pp. 242–264.

[145] Dror M, 2000, *Arc routing: Theory, solutions, and applications*, Springer Science & Business Media, New York (NY).

[146] Dror M & Trudeau P, 1989, *Savings by split delivery routing*, Transportation Science, **141(23)**, pp. 141–145.

[147] Du Merle O, Villeneuve D, Desrosiers J & Hansen P, 1999, *Stabilized column generation*, Discrete Mathematics, **194(1)**, pp. 229–237.

[148] Dubes RC, 1987, *How many clusters are best? An experiment*, Pattern Recognition, **20(6)**, pp. 645–663.

[149] Duda RO & Hart PE, 1973, *Pattern classification and scene analysis*, Wiley & Sons New York (NY).

[150] Dueck G, 1993, *New optimization heuristics: The great deluge algorithm and the record-to-record travel*, Journal of Computational Physics, **104(1)**, pp. 86–92.

[151] Dueck G & Scheuer T, 1990, *Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing*, Journal of Computational Physics, **90(1)**, pp. 161–175.

[152] Duin RP, Fred AL, Loog M & Pkekalska E, 2012, *Mode seeking clustering by KNN and mean shift evaluated*, pp. 51–59 in Gimelfarb G, Hancock E, Imiya A, Kuijper A, Kudo M, Omachi S, Windeat T & Yamada K (Eds), *Structural, syntactic, and statistical pattern recognition*, Springer, Berlin.

[153] Dunn JC, 1974, *Well-separated clusters and optimal fuzzy partitions*, Journal of Cybernetics, **4(1)**, pp. 95–104.

[154] Eisenbrand F, 2007, *Gomory-Chvátal cutting planes and the elementary closure of polyhedra*, PhD Thesis, Saarland University, Saarbrúcken.

[155] Erdoğan S & Miller-Hooks E, 2012, *A green vehicle routing problem*, Transportation Research Part E: Logistics and Transportation Review, **48(1)**, pp. 100–114.

[156] Feillet D, Dejax P & Gendreau M, 2005, *Traveling salesman problems with profits*, Transportation Science, **39(2)**, pp. 188–205.

[157] Feng P, Xiao-Ting L, Qian Z, Wei-Xing L & Qi G, 2013, *Analysis of standard particle swarm optimization algorithm based on Markov chain*, Acta Automatica Sinica, **39(4)**, pp. 381–389.

[158] Ferguson TS, 1973, *A Bayesian analysis of some nonparametric problems*, Annals of Statistics, pp. 209–230.

[159] Fidanova S & Marinov P, 2013, *Number of ants versus number of iterations on ant colony optimization algorithm for wireless sensor layout*, Proceedings of the Workshop of ICT for New Materials and Nanotechnologies, Bankya, pp. 90–93.

[160] Figueiredo MA & Jain AK, 2002, *Unsupervised learning of finite mixture models*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **24(3)**, pp. 381–396.

[161] Fischetti M & Toth P, 1988, *A new dominance procedure for combinatorial optimization problems*, Operations Research Letters, **7(4)**, pp. 181–187.

[162] Fischetti M & Toth P, 1989, *An additive bounding procedure for combinatorial optimization problems*, Operations Research, **37(2)**, pp. 319–328.

[163] Fisher L & Van Ness JW, 1971, *Admissible clustering procedures*, Biometrika, **58(1)**, pp. 91–104.

[164] Fisher ML, 1994, *Optimal solution of vehicle routing problems using minimum k-trees*, Operations Research, **42(4)**, pp. 626–642.

[165] Fisher ML & Jaikumar R, 1981, *A generalized assignment heuristic for vehicle routing*, Networks, **11(2)**, pp. 109–124.

[166] Fisher RA, 1950, *Contributions to mathematical statistics*, John Wiley and Sons, Hoboken (NJ).

[167] Fogel DB, 1994, *An introduction to simulated evolutionary optimization*, IEEE Transactions on Neural Networks, **5(1)**, pp. 3–14.

[168] Fonseca CM, Knowles JD, Thiele L & Zitzler E, 2005, *A tutorial on the performance assessment of stochastic multiobjective optimizers*, Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization, Guanajuato, pp. 240–252.

[169] Foster BA & Ryan DM, 1976, *An integer programming approach to the vehicle scheduling problem*, Operational Research Quarterly, **27**, pp. 367–384.

[170] Fraley C & Raftery AE, 2002, *Model-based clustering, discriminant analysis, and density estimation*, Journal of the American Statistical Association, **97(458)**, pp. 611–631.

[171] Frey T & Van Groenewoud H, 1972, *A cluster analysis of the D2 matrix of white spruce stands in Saskatchewan based on the maximum-minimum principle*, Journal of Ecology, pp. 873–886.

[172] Friedman HP & Rubin J, 1967, *On some invariant criteria for grouping data*, Journal of the American Statistical Association, **62(320)**, pp. 1159–1178.

[173] Fu L & Medico E, 2007, *FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data*, Bioinformatics, **8(1)**, pp. 1–3.

[174] Fukasawa R, Longo H, Lysgaard J, De Aragao MP, Reis M, Uchoa E & Werneck RF, 2006, *Robust branch and cut and price for the capacitated vehicle routing problem*, Mathematical Programming, **106(3)**, pp. 491–511.

[175] Fukunaga K & Hostetler LD, 1975, *The estimation of the gradient of a density function, with applications in pattern recognition*, IEEE Transactions on Information Theory, **21(1)**, pp. 32–40.

[176] García-Martínez C, Cordón O & Herrera F, 2007, *A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP*, European Journal of Operational Research, **180(1)**, pp. 116–148.

[177] Gary Jarrett P, 1998, *Logistics in the health care industry*, International Journal of Physical Distribution and Logistics Management, **28(9/10)**, pp. 741–772.

[178] Gaskell T, 1967, *Bases for vehicle fleet scheduling*, Operations Research Quartely, **18(3)**, pp. 281–295.

[179] Gee E, 2010, *Encyclopedia of death and dying*, [Online], [accessed 10th October 2016], Available from: `http://www.deathreference.com/Ke-Ma/Life-Expectancy.html`.

[180] Geetha S, Poonthalir G & Vanathi P, 2009, *Improved k-means algorithm for capacitated clustering problem*, Journal of Computer Science, **8(4)**, pp. 52–59.

[181] Geiger M, 2003, *A computational study of genetic crossover operators for multi-objective vehicle routing problem wih soft time windows*, (Unpublished) Technical Report CoRR-2008, Deutscher Universitats-Verlag, Hessen.

[182] Geiger MJ, 2008, *Genetic algorithms for multiple objective vehicle routing*, Proceedings of the 4th Metaheuristic International Conference, Porto, pp. 348–353.

[183] Gen M & Cheng R, 1997, *Genetic algorithms & engineering design*, Wiley & Sons, New York (NY).

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

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

[186] Gendreau M, Iori M, Laporte G & Martello S, 2006, *A tabu search algorithm for a routing and container loading problem*, Transportation Science, **40(3)**, pp. 342–350.

[187] Gendreau M, Laporte G & Semet F, 1997, *The covering tour problem*, Operations Research, **45(4)**, pp. 568–576.

[188] Geoffrion AM, 1968, *Proper efficiency and the theory of vector maximization*, Journal of Mathematical Analysis and Applications, **22(3)**, pp. 618–630.

[189] Giannikos I, 1998, *A multiobjective programming model for locating treatment sites and routing hazardous wastes*, European Journal of Operational Research, **104(2)**, pp. 333–342.

[190] Gillett BE & Miller LR, 1974, *A heuristic algorithm for the vehicle-dispatch problem*, Operations Research, **22(2)**, pp. 340–349.

[191] Glover F, 1989, *Tabu search — Part I*, ORSA Journal on Computing, **1(3)**, pp. 190–206.

[192] Goel A & Meisel F, 2013, *Workforce routing and scheduling for electricity network maintenance with downtime minimization*, European Journal of Operational Research, **231(1)**, pp. 210–228.

[193] Goksal FP, Karaoglan I & Altiparmak F, 2013, *A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery*, Computers and Industrial Engineering, **65(1)**, pp. 39–53.

[194] Golberg DE, 1989, *Genetic algorithms in search, optimization, and machine learning*, Addion Wesley, Boston (MA).

[195] Goldberg DE & Holland JH, 1988, *Genetic algorithms and machine learning*, Machine Learning, **3(2)**, pp. 95–99.

[196] Golden B, Wasil E, Kelly J & Chao I, 2012, *Metaheuristics in vehicle routing*, pp. 33–54 in Craignic T & Laporte G (Eds), *Fleet management and logistics*, Kluwer Academic Press, Boston (MA).

[197] Golden BL, Magnanti TL & Nguyen HQ, 1977, *Implementing vehicle routing algorithms*, Networks, **7(2)**, pp. 113–148.

[198] Golden BL, Raghavan S & Wasil EA, 2008, *The vehicle routing problem: Latest advances and new challenges*, Springer Science & Business Media, New York (NY).

[199] Golden BL, Wasil EA, Kelly JP & Chao IM, 1998, *The impact of metaheuristics on solving the vehicle routing problem: Algorithms, problem sets, and computational results*, pp. 33–56 in Crainic TG & Laporte G (Eds), *Fleet management and logistics*, Springer Science & Business Media, New York (NY).

[200] Gomory RE, 1958, *An algorithm for integer solutions to linear programs*, pp. 211–215 in Bellman R & Hall J (Eds), *Combinatorial Analysis Symposia in Applied Mathematics X*, American Mathematical Society, Providence.

[201] Goodall DW, 1966, *A new similarity index based on probability*, Biometrics, **22**, pp. 882–907.

[202] Google, 2017, *Google maps for every platform*, [Online], [accessed 2nd May 2016], Available at: `https://developers.google.com/maps/`.

[203] Gower JC & Ross G, 1969, *Minimum spanning trees and single linkage cluster analysis*, Applied Statistics, **18(1)**, pp. 54–64.

[204] Gravel M, Price WL & Gagné C, 2002, *Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic*, European Journal of Operational Research, **143(1)**, pp. 218–229.

[205] Gray RM, 2011, *Entropy and information theory*, Springer Science & Business Media, Berlin.

[206] Gribkovskaia I, Halskau Ø, Laporte G & Vlček M, 2007, *General solutions to the single vehicle routing problem with pickups and deliveries*, European Journal of Operational Research, **180(2)**, pp. 568–584.

[207] Groër C, Golden B & Wasil E, 2010, *A library of local search heuristics for the vehicle routing problem*, Mathematical Programming Computation, **2(2)**, pp. 79–101.

[208] Groër C, Golden B & Wasil E, 2011, *A parallel algorithm for the vehicle routing problem*, INFORMS Journal on Computing, **23(2)**, pp. 315–330.

[209] Grötschel M & Padberg MW, 1979, *On the symmetric travelling salesman problem I: Inequalities*, Mathematical Programming, **16(1)**, pp. 265–280.

[210] Guan M, 1984, *On the windy postman problem*, Discrete Applied Mathematics, **9(1)**, pp. 41–46.

[211] Haase K, Desaulniers G & Desrosiers J, 2001, *Simultaneous vehicle and crew scheduling in urban mass transit systems*, Transportation Science, **35(3)**, pp. 286–303.

[212] Hachicha M, Hodgson MJ, Laporte G & Semet F, 2000, *Heuristics for the multi-vehicle covering tour problem*, Computers and Operations Research, **27(1)**, pp. 29–42.

[213] Hadjiconstantinou E, Christofides N & Mingozzi A, 1995, *A new exact algorithm for the vehicle routing problem based on q-paths and k-shortest paths relaxations*, Annals of Operations Research, **61(1)**, pp. 21–43.

[214] Halkidi M, Batistakis Y & Vazirgiannis M, 2002, *Cluster validity methods: Part I*, ACM Sigmod Record, **31(2)**, pp. 40–45.

[215] Halkidi M & Vazirgiannis M, 2001, *Clustering validity assessment: Finding the optimal partitioning of a data set*, Proceedings of the 2001 IEEE International Conference on Data Mining, San Jose (CA), pp. 187–194.

[216] Halkidi M, Vazirgiannis M & Batistakis Y, 2000, *Quality scheme assessment in the clustering process*, Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Lyon, pp. 265–276.

[217] Hall LO, Özyurt IB & Bezdek JC, 1999, *Clustering with a genetically optimized approach*, IEEE Transactions on Evolutionary Computation, **3(2)**, pp. 103–112.

[218] Handl J, Knowles J & Kell DB, 2005, *Computational cluster validation in post-genomic data analysis*, Bioinformatics, **21(15)**, pp. 3201–3212.

[219] Hansen M & Yu B, 2000, *Wavelet thresholding via MDL for natural images*, IEEE Transactions on Information Theory, **46(5)**, pp. 1778–1788.

[220] Hansen MP & Jaszkiewicz A, 1998, *Evaluating the quality of approximations to the non-dominated set*, Technical University of Denmark, Lyngby.

[221] Harik GR & Lobo FG, 1999, *A parameter-less genetic algorithm*, Proceedings of the Genetic and Evolutionary Computation Conference, San Mateo (CA), pp. 258–267.

[222] Hartigan JA, 1973, *Clustering*, Annual Review of Biophysics and Bioengineering, **2(1)**, pp. 81–102.

[223] Hartigan JA, 1975, *Clustering algorithms*, Wiley, Hoboken (NJ).

[224] Hartuv E & Shamir R, 2000, *A clustering algorithm based on graph connectivity*, Information Processing Letters, **76(4-6)**, pp. 175–181.

[225] Heilporn G, Cordeau JF & Laporte G, 2010, *The delivery man problem with time windows*, Discrete Optimization, **7(4)**, pp. 269–282.

[226] Held M & Karp RM, 1971, *The traveling-salesman problem and minimum spanning trees: Part II*, Mathematical Programming, **1(1)**, pp. 6–25.

[227] Hempsch C & Irnich S, 2008, *Vehicle routing problems with inter-tour resource constraints*, pp. 421–444 in Golden B, Raghavan S & Wasil E (Eds), *The vehicle routing problem: Latest advances and new challenges*, Springer Science & Business Media, New York (NY).

[228] Herrero J, Valencia A & Dopazo J, 2001, *A hierarchical unsupervised growing neural network for clustering gene expression patterns*, Bioinformatics, **17(2)**, pp. 126–136.

[229] Hertzsprung E, 1922, *Mean colour equivalents and hypothetical angular semi-diameters of 734 stars brighter than the fifth magnitude and within 95° of the North Pole*, Annalen van de Sterrewacht te Leiden, **14**, pp. A1–A55.

[230] Ho W, Ho GT, Ji P & Lau HC, 2008, *A hybrid genetic algorithm for the multi-depot vehicle routing problem*, Engineering Applications of Artificial Intelligence, **21(4)**, pp. 548–557.

[231] Hofmann T & Buhmann JM, 1997, *Pairwise data clustering by deterministic annealing*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **19(1)**, pp. 1–14.

[232] Holland JH & Reitman JS, 1977, *Cognitive systems based on adaptive algorithms*, ACM Sigart Bulletin, **(63)**, pp. 49–49.

[233] Hoshino EA & De Souza CC, 2008, *Column generation algorithms for the capacitated m-ring-star problem*, Proceedings of the 14th International Conference on Computing and Combinatorics, Berlin, pp. 631–641.

[234] Hu ZH, Zhao Y & Choi TM, 2013, *Vehicle routing problem for fashion supply chains with cross-docking*, Mathematical Problems in Engineering, **2013**, pp. 1–10.

[235] IBM, 2013, *IBM ILOG CPLEX optmisation studio*, [Online], [accessed 5th October 2016], Available at: http://www.ibm.com/support/knowledgecenter/SSSA5P_12.5.1/maps/ic-homepage.html.

[236] InfoPlease, 2015, *Life expectancy for Countries, 2015*, [Online], [accessed 10th October 2016], Available at: http://www.infoplease.com/world/statistics/life-expectancy-country.html.

[237] Ioachim I, Desrosiers J, Dumas Y, Solomon MM & Villeneuve D, 1995, *A request clustering algorithm for door-to-door handicapped transportation*, Transportation Science, **29(1)**, pp. 63–78.

[238] Iori M, Salazar-González JJ & Vigo D, 2007, *An exact approach for the vehicle routing problem with two-dimensional loading constraints*, Transportation Science, **41(2)**, pp. 253–264.

[239] Irnich S, Desaulniers G, Desrosiers J & Hadjar A, 2010, *Path-reduced costs for eliminating arcs in routing and scheduling*, INFORMS Journal on Computing, **22(2)**, pp. 297–313.

[240] Irnich S & Villeneuve D, 2006, *The shortest-path problem with resource constraints and k-cycle elimination for k greater or equal to 3*, INFORMS Journal on Computing, **18(3)**, pp. 391–406.

[241] Jacobs F, 2007, *The patients per doctor map of the world*, [Online], [accessed 11th October 2016], Available at: http://bigthink.com/strange-maps/185-the-patients-per-doctor-map-of-the-world.

[242] Jain AK, 2010, *Data clustering: 50 years beyond K-means*, Pattern Recognition Letters, **31(8)**, pp. 651–666.

[243] Jain AK & Dubes RC, 1988, *Algorithms for clustering data*, Prentice-Hall Publishing, Upper Saddle River (NJ).

[244] Jain AK, Murty MN & Flynn PJ, 1999, *Data clustering: A review*, ACM Computing Surveys, **31(3)**, pp. 264–323.

[245] Jepsen M, Petersen B, Spoorendonk S & Pisinger D, 2008, *Subset-row inequalities applied to the vehicle-routing problem with time windows*, Operations Research, **56(2)**, pp. 497–511.

[246] Jin J, Crainic TG & Løkketangen A, 2012, *A parallel multi-neighborhood cooperative tabu search for capacitated vehicle routing problems*, European Journal of Operational Research, **222(3)**, pp. 441–451.

[247] Johnson DS & McGeoch LA, 1997, *The traveling salesman problem: A case study in local optimization*, Local Search in Combinatorial Optimization, **1**, pp. 215–310.

[248] Johnson D, Krishnan S, Chhugani J, Kumar S & Venkatasubramanian S, 2004, *Compressing large boolean matrices using reordering techniques*, Proceedings of the 13th International Conference on Very Large Data Bases — Volume, Lausanne, pp. 13–23.

[249] Jozefowiez N, Semet F & Talbi EG, 2007, *The bi-objective covering tour problem*, Computers and Operations Research, **34(7)**, pp. 1929–1942.

[250] Jozefowiez N, Semet F & Talbi EG, 2008, *Multi-objective vehicle routing problems*, European Journal of Operational Research, **189(2)**, pp. 293–309.

[251] Kalcsics J, Nickel S & Schroder M, 2005, *Towards a unified territorial design approach: Applications, algorithms and GIS integration*, Top, **13(1)**, pp. 1–56.

[252] Kallehauge B, 2008, *Formulations and exact algorithms for the vehicle routing problem with time windows*, Computers and Operations Research, **35(7)**, pp. 2307–2330.

[253] Kaufman LR & Rousseeuw PJ, 1987, *Clustering by means of medoids*, pp. 405–416 in Dodge Y (Ed), *Statistical data analysis based on the L1 norm*, North-Holland, Amsterdam.

[254] Kaufman LR & Rousseeuw PJ, 2009, *Finding groups in data: An introduction to cluster analysis, 2nd Edition*, John Wiley & Sons, Hoboken (NJ).

[255] Kawamura H, Yamamoto M, Mitamura T, Suzuki K & Ohuchi A, 1998, *Cooperative search based on pheromone communication for vehicle routing problems*, IEICE Traansactions on Fundamentals of Electronics, Communications and Computer Sciences, **81(6)**, pp. 1089–1096.

[256] Kenndy J & Eberhart R, 1995, *Particle swarm optimization*, Proceedings of the IEEE International Conference on Neural Networks, Perth, pp. 1942–1948.

[257] King B, 1967, *Step-wise clustering procedures*, Journal of the American Statistical Association, **62(317)**, pp. 86–101.

[258] Kirkpatrick S, 1984, *Optimization by simulated annealing: Quantitative studies*, Journal of Statistical Physics, **34(5-6)**, pp. 975–986.

[259] Kleinberg J, 2003, *An impossibility theorem for clustering*, pp. 463–470 in Becker S, Thrun S & Obermayer K (Eds), *Advances in Neural Information Processing Systems 15*, MIT Press, Cambridge (MA).

[260] Kolen AW, Rinnooy Kan A & Trienekens H, 1987, *Vehicle routing with time windows*, Operations Research, **35(2)**, pp. 266–273.

[261] Kollat JB & Reed PM, 2005, *The value of online adaptive search: A performance comparison of NSGAII, ε-NSGAII and ε-MOEA*, Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization, Guanajuato, pp. 386–398.

[262] Kraemer HC, 1982, *Biserial correlation*, pp. 276–280 in Kotz S & Johnson N (Eds), *Encyclopedia of statistical sciences*, Wiley & Sons, New York (NY).

[263] Krishna K & Murty MN, 1999, *Genetic k-means algorithm*, Proceedings of the IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, San Diego (CA), pp. 433–439.

[264] Kruskal JB, 1964, *Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis*, Psychometrika, **29(1)**, pp. 1–27.

[265] Krzanowski WJ & Lai Y, 1988, *A criterion for determining the number of groups in a data set using sum-of-squares clustering*, Biometrics, **44(1)**, pp. 23–34.

[266] Kytöjoki J, Nuortio T, Bräysy O & Gendreau M, 2007, *An efficient variable neighborhood search heuristic for very large scale vehicle routing problems*, Computers and Operations Research, **34(9)**, pp. 2743–2757.

[267] Lacomme P, Prins C & Sevaux M, 2006, *A genetic algorithm for a bi-objective capacitated arc routing problem*, Computers and Operations Research, **33(12)**, pp. 3473–3493.

[268] Lagus K, Honkela T, Kaski S & Kohonen T, 1996, *Self-organizing maps of document collections: A new approach to interactive exploration*, Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland (OR), pp. 238–243.

[269] Lambert T, 2016, *A brief history of medicine*, [Online], [accessed 10th October 2016], Available at: `http://www.localhistories.org/medicine.html`.

[270] Lance GN & Williams WT, 1966, *Computer programs for hierarchical polythetic classification*, The Computer Journal, **9(1)**, pp. 60–64.

[271] Lance GN & Williams WT, 1967, *Mixed-data classificatory programs I — Agglomerative systems*, Australian Computer Journal, **1(1)**, pp. 15–20.

[272] Lancet, 2017, *Welcome to Lancet Laboratories*, [Online], [accessed 10th September 2017], Available at: `http://www.lancet.co.za/`.

[273] Lange T, Roth V, Braun ML & Buhmann JM, 2004, *Stability-based validation of clustering solutions*, Neural Computation, **16(6)**, pp. 1299–1323.

[274] Laporte G, Mercure H & Nobert Y, 1986, *An exact algorithm for the asymmetrical capacitated vehicle routing problem*, Networks, **16(1)**, pp. 33–46.

[275] Laporte G & Nobert Y, 1983, *A branch and bound algorithm for the capacitated vehicle routing problem*, Operations Research Spektrum, **5(2)**, pp. 77–85.

[276] Laporte G, Nobert Y & Desrochers M, 1985, *Optimal routing under capacity and distance restrictions*, Operations Research, **33(5)**, pp. 1050–1073.

[277] Larranaga P, Kuijpers CM, Murga RH, Inza I & Dizdarevic S, 1999, *Genetic algorithms for the travelling salesman problem: A review of representations and operators*, Artificial Intelligence Review, **13(2)**, pp. 129–170.

[278] Le Bouthillier A, Crainic TG & Kropf P, 2005, *A guided cooperative search for the vehicle routing problem with time windows*, IEEE Transactions on Intelligent Systems, **20(4)**, pp. 36–42.

[279] Lebart L, Piron A, Lebart M, Morineau A & Piron M, 2000, *Statistique exploratoire multidimensionnelle*, Dunod, Paris.

[280] Lee CY, Lee ZJ, Lin SW & Ying KC, 2010, *An enhanced ant colony optimization (EACO) applied to capacitated vehicle routing problem*, Applied Intelligence, **32(1)**, pp. 88–95.

[281] Lee HL, Padmanabhan V & Whang S, 1997, *The bullwhip effect in supply chains*, Sloan Management Review, **38(3)**, pp. 93–102.

[282] Lee TR & Ueng JH, 1999, *A study of vehicle routing problems with load-balancing*, International Journal of Physical Distribution and Logistics Management, **29(10)**, pp. 646–657.

[283] Letchford AN, Eglese RW & Lysgaard J, 2002, *Multistars, partial multistars and the capacitated vehicle routing problem*, Mathematical Programming, **94(1)**, pp. 21–40.

[284] Letchford AN & Salazar-González JJ, 2006, *Projection results for vehicle routing*, Mathematical Programming, **105(2-3)**, pp. 251–274.

[285] Leung Y, Zhang JS & Xu ZB, 2000, *Clustering by scale-space filtering*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **22(12)**, pp. 1396–1410.

[286] Li C & Biswas G, 2002, *Unsupervised learning with mixed numeric and nominal data*, IEEE Transactions on Knowledge and Data Engineering, **14(4)**, pp. 673–690.

[287] Li F, Golden B & Wasil E, 2005, *Very large-scale vehicle routing: New test problems, algorithms, and results*, Computers and Operations Research, **32(5)**, pp. 1165–1179.

[288] Liao CJ, Lin Y & Shih SC, 2010, *Vehicle routing with cross-docking in the supply chain*, Expert Systems with Applications, **37(10)**, pp. 6868–6873.

[289] Lin CR, Liu KH & Chen MS, 2005, *Dual clustering: Integrating data clustering over optimization and constraint domains*, IEEE Transactions on Knowledge and Data Engineering, **17(5)**, pp. 628–637.

[290] Lin S, 1965, *Computer solutions of the traveling salesman problem*, Bell Systems Technical Journal, **44(10)**, pp. 2245–2269.

[291] Lin S & Kernighan BW, 1973, *An effective heuristic algorithm for the traveling-salesman problem*, Operations Research, **21(2)**, pp. 498–516.

[292] Linnaeus C, 1793, *Systema naturae per regna tria naturae secundum classes, ordines, genera, species,cum characteribus, differentiis, synonymis, locis*, Edition, Holmiae: Impensis Laurentii Salvii.

[293] Little JD, Murty KG, Sweeney DW & Karel C, 1963, *An algorithm for the traveling salesman problem*, Operations Research, **11(6)**, pp. 972–989.

[294] Liu CL, 1968, *Introduction to combinatorial mathematics*, McGraw-Hill, New York (NY).

[295] Lozano JA & Larranaga P, 1999, *Applying genetic algorithms to search for the best hierarchical clustering of a dataset*, Pattern Recognition Letters, **20(9)**, pp. 911–918.

[296] Lysgaard J, 2003, *CVRPSEP: A package of separation routines for the capacitated vehicle routing problem*, (Unpublished) Technical Report, Department of Management Science and Logistics, Aarhus School of Business, Aarhus.

[297] Lysgaard J, Letchford AN & Eglese RW, 2004, *A new branch-and-cut algorithm for the capacitated vehicle routing problem*, Mathematical Programming, **100(2)**, pp. 423–445.

[298] MacQueen J, 1967, *Some methods for classification and analysis of multivariate observations*, Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Oakland (CA), pp. 281–297.

[299] Marinakis Y, Marinaki M & Dounias G, 2010, *A hybrid particle swarm optimization algorithm for the vehicle routing problem*, Engineering Applications of Artificial Intelligence, **23(4)**, pp. 463–472.

[300] Marler RT & Arora JS, 2004, *Survey of multi-objective optimization methods for engineering*, Structural and Multidisciplinary Optimization, **26(6)**, pp. 369–395.

[301] Marriott F, 1971, *Practical problems in a method of cluster analysis*, Biometrics, **27(3)**, pp. 501–514.

[302] Martinhon C, Lucena A & Maculan N, 2000, *A relax and cut algorithm for the vehicle routing problem*, European Journal of Operational Research, **35(6)**, pp. 56–71.

[303] Maulik U & Bandyopadhyay S, 2000, *Genetic algorithm-based clustering technique*, Pattern Recognition, **33(9)**, pp. 1455–1465.

[304] McClain JO & Rao VR, 1975, *Clustisz: A program to test for the quality of clustering of a set of objects*, Journal of Marketing Research, **12(4)**, pp. 456–460.

[305] McNaughton-Smith P, Williams W, Dale M & Mockett L, 1964, *Dissimilarity analysis*, Nature, **202**, pp. 1034–1035.

[306] Mester D & Bräysy O, 2005, *Active guided evolution strategies for large-scale vehicle routing problems with time windows*, Computers and Operations Research, **32(6)**, pp. 1593–1614.

[307] Michael RA, 1973, *Cluster analysis for applications*, Academic Press, Cambridge (MA).

[308] Michalewicz Z, 1996, *Introduction*, pp. 1–10 in Michalewicz Z (Ed), *Genetic algorithms + data structures = evolution programs*, Springer, Berlin.

[309] Miliotis P, 1976, *Integer programming approaches to the travelling salesman problem*, Mathematical Programming, **10(1)**, pp. 367–378.

[310] Miller CE, Tucker AW & Zemlin RA, 1960, *Integer programming formulations and traveling salesman problems*, Journal of the Association for Computing Machinery, **7(4)**, pp. 326–329.

[311] Miller DL, 1995, *A matching based exact algorithm for capacitated vehicle routing problems*, ORSA Journal on Computing, **7(1)**, pp. 1–9.

[312] Milligan GW & Cooper MC, 1985, *An examination of procedures for determining the number of clusters in a data set*, Psychometrika, **50(2)**, pp. 159–179.

[313] Min H, 1989, *The multiple vehicle routing problem with simultaneous delivery and pick-up points*, Transportation Research Part A: General, **23(5)**, pp. 377–386.

[314] Minella G, Ruiz R & Ciavotta M, 2008, *A review and evaluation of multiobjective algorithms for the flowshop scheduling problem*, INFORMS Journal on Computing, **20(3)**, pp. 451–471.

[315] Mitchell TM, 1997, *Machine learning*, McGraw Hill, Burr Ridge (IL).

[316] Mladenović N & Hansen P, 1997, *Variable neighbourhood search*, Computers and Operations Research, **24**, pp. 1097–1100.

[317] Mole R & Jameson S, 1976, *A sequential route-building algorithm employing a generalised savings criterion*, Operational Research Quarterly, **27(503)**, pp. 503–511.

[318] Muter İ, Birbil SI & Sahin G, 2010, *Combination of metaheuristic and exact algorithms for solving set covering-type optimization problems*, INFORMS Journal on Computing, **22(4)**, pp. 603–619.

[319] Naddef D & Rinaldi G, 2001, *Branch-and-cut algorithms for the capacitated VRP*, pp. 53–84 in Toth P & Vigo D (Eds), *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia (PA).

[320] Naddef D & Rinaldi G, 2002, *Branch-and-cut algorithms for the capacitated VRP*, pp. 53–84 in Toth P & Vigo D (Eds), *The vehicle routing problem*, Sociery for Industrial and Applied Mathematics, Philadelphia (PA).

[321] Nagata Y & Bräysy O, 2009, *Edge assembly-based memetic algorithm for the capacitated vehicle routing problem*, Networks, **54(4)**, pp. 205–215.

[322] Nagy G, 1968, *State of the art in pattern recognition*, IEEE Transactions on Computers, New York (NY), **56(5)**, pp. 836–864.

[323] Nazif H & Lee LS, 2012, *Optimised crossover genetic algorithm for capacitated vehicle routing problem*, Applied Mathematical Modelling, **36(5)**, pp. 2110–2117.

[324] Nelson MD, Nygard KE, Griffin JH & Shreve WE, 1985, *Implementation techniques for the vehicle routing problem*, Computers and Operations Research, **12(3)**, pp. 273–283.

[325] Ngueveu SU, Prins C & Calvo RW, 2010, *An effective memetic algorithm for the cumulative capacitated vehicle routing problem*, Computers and Operations Research, **37(11)**, pp. 1877–1885.

[326] NHLS, 2014, *National Health Laboratory Service*, [Online], [accessed 28th February 2015], Available at: `http://www.nhls.ac.za/?page=about_us&id=16`.

[327] Nicoară ES, 2007, *Performance measures for multi-objective optimization algorithms*, (Unpublished) Technical Report LIX(2007), Gaza din Ploeisti, Seria Matematica — Informatica — Fizica, University of Ploeisti, Bucharest.

[328] Ombuki B, Ross BJ & Hanshar F, 2006, *Multi-objective genetic algorithms for vehicle routing problem with time windows*, Applied Intelligence, **24(1)**, pp. 17–30.

[329] OPSI, 2017, *Leaders in transport planning & fleet optimisation*, [Online], [accessed 2nd May 2016], Available at: `http://www.opsisystems.com/company_overview`.

[330] Orloff C, 1974, *Routing a fleet of M vehicles from a central facility*, Networks, **4(2)**, pp. 147–162.

[331] Orwell G, 1945, *Animal farm: A fairy story*, Secker and Warburg, London.

[332] Osman IH, 1993, *Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem*, Annals of Operations Research, **41(4)**, pp. 421–451.

[333] OSRM, 2017, *Open source routing machine*, [Online], [accessed 2nd May 2016], Available at: `http://www.http://project-osrm.org/`.

[334] Pacheco J & Martí R, 2006, *Tabu search for a multi-objective routing problem*, Journal of the Operational Research Society, **57(1)**, pp. 29–37.

[335] Paessens H, 1988, *The savings algorithm for the vehicle routing problem*, European Journal of Operational Research, **34(3)**, pp. 336–344.

[336] Paquete LF, 2005, *Stochastic local search algorithms for multiobjective combinatorial optimization: Methods and analysis*, IOS Press, Amsterdam.

[337] Paquete L & Stützle T, 2003, *A two-phase local search for the biobjective traveling salesman problem*, Proceedings of the Conference on Evolutionary Multi-Criterion Optimization, Faro, pp. 479–493.

[338] Pathcare, 2017, *PathCare is a people-focused, technology-driven laboratory testing services pathology practice*, [Online], [accessed 10th September 2017], Available at: `https://www.pathcare.co.za/`.

[339] Patil H, 1986, *On the structure of k-trees*, Journal of Combinatorics, Information and System Sciences, **11(2-4)**, pp. 57–64.

[340] Pecin D, Pessoa A, Poggi M & Uchoa E, 2014, *Improved branch-cut-and-price for capacitated vehicle routing*, pp. 393–403 in Lee J & Vygen J (Eds), *Integer programming and combinatorial optimization*, Springer, Berlin.

[341] Pecin D, Poggi M & Martinelli R, 2013, *Efficient elementary and restricted non-elementary route pricing*, (Unpublished) Technical Report 11, Pontfical Catholic University of Rio de Janeioro, Rio de Janeiro.

[342] Pelikan M, Goldberg DE & Cantú-Paz E, 2000, *Bayesian optimization algorithm, population sizing, and time to convergence*, Proceedings of the 9th International Conference on Genetic and Evolutionary Computation, Las Vegas (NV), pp. 275–282.

[343] Perboli G, Tadei R & Vigo D, 2011, *The two-echelon capacitated vehicle routing problem: Models and math-based heuristics*, Transportation Science, **45(3)**, pp. 364–380.

[344] Pessoa AA, Sadykov R, Uchoa E & Vanderbeck F, 2013, *In-out separation and column generation stabilization by dual price smoothing*, Proceedings of the 12th International Symposium on Experimental Algorithms, Rome, pp. 354–365.

[345] Pessoa A, De Aragão MP & Uchoa E, 2007, *A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem*, pp. 150–160 in Demetrescus C (Ed), *Experimental algorithms*, Springer, Berlin.

[346] Pessoa A, De Aragão MP & Uchoa E, 2008, *Robust branch-cut-and-price algorithms for vehicle routing problems*, pp. 297–325 in Golden B, Raghavan S & Wasil EA (Eds), *The vehicle routing problem: Latest advances and new challenges*, Springer, Berlin.

[347] Pessoa A, Uchoa E & De Aragão MP, 2009, *A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem*, Networks, **54(4)**, pp. 167–177.

[348] Pierre DM & Zakaria N, 2014, *Partially optimized cyclic shift crossover for multi-objective genetic algorithms for the multi-objective vehicle routing problem with time-windows*, Proceedings of the IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making, Orlando (FL), pp. 106–115.

[349] Pisinger D & Ropke S, 2007, *A general heuristic for vehicle routing problems*, Computers and Operations Research, **34(8)**, pp. 2403–2435.

[350] Potvin JY & Naud MA, 2011, *Tabu search with ejection chains for the vehicle routing problem with private fleet and common carrier*, Journal of the Operational Research Society, **62(2)**, pp. 326–336.

[351] Prins C, 2004, *A simple and effective evolutionary algorithm for the vehicle routing problem*, Computers and Operations Research, **31(12)**, pp. 1985–2002.

[352] Prins C, 2009, *A GRASP evolutionary local search hybrid for the vehicle routing problem*, pp. 35–53 in Periera FB & Tavares J (Eds), *Bio-inspired algorithms for the vehicle routing problem*, Springer, Berlin.

[353] Puljić K & Manger R, 2013, *Comparison of eight evolutionary crossover operators for the vehicle routing problem*, Mathematical Communications, **18(2)**, pp. 359–375.

[354] Pureza V & Franca P, 1991, *Vehicle routing problems via tabu search metaheuristic*, (Unpublished) Technical Report CRT-747, Centre De Recherche Sur Les Transports Publication, Montreal.

[355] Quinlan JR, 2014, *C4.5: Programs for machine learning*, Morgan Kaufman, Burlington (MA).

[356] Rademeyer AL, 2014, *Algorithmic approaches to solving multi-period sales force and delivery vehicle master routing problems*, PhD Thesis, University of the Witwatersrand, Johannesburg.

[357] Rahoual M, Kitoun B, Mabed MH, Bachelet V & Benameur F, 2001, *Multicriteria genetic algorithms for the vehicle routing problem with time windows*, Proceedings of the 4th Metaheuristics International Conference, Porto, pp. 527–532.

[358] Ralphs TK, Kopman L, Pulleyblank WR & Trotter LE, 2003, *On the capacitated vehicle routing problem*, Mathematical Programming, **94(2-3)**, pp. 343–359.

[359] Rand GK, 2009, *The life and times of the savings method for vehicle routing problems*, ORiON, **25(2)**, pp. 125–145.

[360] Raquel CR & Naval Jr PC, 2005, *An effective use of crowding distance in multiobjective particle swarm optimization*, Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, San Diego (CA), pp. 257–264.

[361] Rasmussen CE, 1999, *The infinite Gaussian mixture model*, Proceedings of the 12th Neural Information Processing Systems Conference, Denver (CO), pp. 554–560.

[362] Ratkowsky D & Lance G, 1978, *A criterion for determining the number of groups in a classification*, Australian Computer Journal, **10(3)**, pp. 115–117.

[363] Reed M, Yiannakou A & Evering R, 2014, *An ant colony algorithm for the multi-compartment vehicle routing problem*, Applied Soft Computing, **15**, pp. 169–176.

[364] Reimann M, Doerner K & Hartl RF, 2004, *D-ants: Savings based ants divide and conquer the vehicle routing problem*, Computers and Operations Research, **31(4)**, pp. 563–591.

[365] Renaud J, Boctor FF & Laporte G, 1996, *A fast composite heuristic for the symmetric traveling salesman problem*, INFORMS Journal on Computing, **8(2)**, pp. 134–143.

[366] Renaud J, Boctor FF & Laporte G, 1996, *An improved petal heuristic for the vehicle routeing problem*, Journal of the Operational Research Society, pp. 329–336.

[367] Renaud J, Laporte G & Boctor FF, 1996, *A tabu search heuristic for the multi-depot vehicle routing problem*, Computers and Operations Research, **23(3)**, pp. 229–235.

[368] Resende MG & Ribeiro CC, 2010, *Greedy randomized adaptive search procedures: Advances, hybridizations, and applications*, pp. 283–319 in Gendreau N & Potvin J (Eds), *Handbook of metaheuristics*, Springer, Berlin.

[369] Reyes-Sierra M & Coello CC, 2006, *Multi-objective particle swarm optimizers: A survey of the state-of-the-art*, International Journal of Computational Intelligence Research, **2(3)**, pp. 287–308.

[370] Ribeiro R & Ramalhinho Dias Lourenço H, 2001, *A multi-objective model for a multi-period distribution management problem*, Proceedings of the 4th Metaheuristic International Conference, Porto, pp. 91–102.

[371] Rieck K, Trinius P, Willems C & Holz T, 2011, *Automatic analysis of malware behavior using machine learning*, Journal of Computer Security, **19(4)**, pp. 639–668.

[372] Riera-Ledesma J & Salazar-Gonzalez JJ, 2005, *The biobjective travelling purchaser problem*, European Journal of Operational Research, **160(3)**, pp. 599–613.

[373] Righini G & Salani M, 2006, *Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints*, Discrete Optimization, **3(3)**, pp. 255–273.

[374] Righini G & Salani M, 2008, *New dynamic programming algorithms for the resource constrained elementary shortest path problem*, Networks, **51(3)**, pp. 155–170.

[375] Robust F, Daganzo CF & Souleyrette RR, 1990, *Implementing vehicle routing models*, Transportation Research Part B: Methodological, **24(4)**, pp. 263–286.

[376] Rochat Y & Taillard ÉD, 1995, *Probabilistic diversification and intensification in local search for vehicle routing*, Journal of Heuristics, **1(1)**, pp. 147–167.

[377] Rohlf FJ, 1974, *Methods of comparing classifications*, Annual Review of Ecology and Systematics, **5**, pp. 101–113.

[378] Røpke S, 2012, *Branching decisions in branch-and-cut-and-price algorithms for vehicle routing problems*, Presentation at Column Generation 2012, Bromont, Available at: https://www.gerad.ca/colloques/ColumnGeneration2012/presentations/session7/Ropke.pdf.

[379] Rousseeuw PJ, 1987, *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*, Journal of Computational and Applied Mathematics, **20**, pp. 53–65.

[380] Ryan DM, Hjorring C & Glover F, 1993, *Extensions of the petal method for vehicle routing*, Journal of the Operational Research Society, **44(3)**, pp. 289–296.

[381] Sakamoto Y, Ishiguro M & Kitagawa G, 1986, *Akaike information criterion statistics*, Springer, New York (NY).

[382] Salazar-Aguilar MA, Langevin A & Laporte G, 2012, *Synchronized arc routing for snow plowing operations*, Computers and Operations Research, **39(7)**, pp. 1432–1440.

[383] Salhi S & Nagy G, 1999, *A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling*, Journal of the Operational Research Society, **50(10)**, pp. 1034–1042.

[384] Santos FA, Mateus GR & Da Cunha AS, 2011, *A novel column generation algorithm for the vehicle routing problem with cross-docking*, pp. 412–425 in Pahl J, Reiners T & Voss S (Eds), *Network Optimization*, Springer, Berlin.

[385] Santos HG, Ochi LS, Marinho EH & Drummond LMA, 2006, *Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem*, Neurocomputing, **70(1)**, pp. 70–77.

[386] Sarle WS, 1983, *Cubic clustering criterion*, SAS Institute, Toronto.

[387] SARS, 2017, *Rates per kilometer*, [Online], [accessed 22nd October 2017], Available at: http://www.sars.gov.za/Tax-Rates/Employers/Pages/Rates-per-kilometer.aspx.

[388] Schaffer JD, 1985, *Multiple objective optimization with vector evaluated genetic algorithms*, Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh (PA), pp. 93–100.

[389] Schlünz EB, Bokov PM & Van Vuuren JH, 2014, *Research reactor in-core fuel management optimisation using the multiobjective cross-entropy method*, Proceedings of the 3rd Annual International Conference on Reactor Physics, Kyoto.

[390] Schmitt I & John L, 1994, *An empirical computational study of genetic algorithms to solve order based problems: An emphasis on TSP and VRPTC*, PhD Thesis, Memphis State University, Memphis (TN).

[391] Schneider M, Stenger A & Goeke D, 2014, *The electric vehicle-routing problem with time windows and recharging stations*, Transportation Science, **48(4)**, pp. 500–520.

[392] Schwarz G, 1978, *Estimating the dimension of a model*, Annals of Statistics, **6(2)**, pp. 461–464.

[393] Scott AJ & Symons MJ, 1971, *Clustering methods based on likelihood ratio criteria*, Biometrics, **27(2)**, pp. 387–397.

[394] Scott G, Clark D & Pham T, 2001, *A genetic clustering algorithm guided by a descent algorithm*, Proceedings of the 2001 Congress on Evolutionary Computation, pp. 734–740.

[395] Selim SZ & Alsultan K, 1991, *A simulated annealing algorithm for the clustering problem*, Pattern Recognition, **24(10)**, pp. 1003–1008.

[396] Services NHL, 2008, *Consultation on technical and operational recommendation for clinical laboratory testing harmonization and standardization*, [Online], [accessed 11th May 2015], Available from: http://www.who.int/healthsystems/round11_9.pdf.

[397] Sessomboon W, Watanabe K, Irohara T & Yoshimoto K, 1998, *A study on multi-objective vehicle routing problem considering customer satisfaction with due-time (the creation of pareto optimal solutions by hybrid genetic algorithm)*, Transactions of the Japan Society of Mechanical Engineers, **64**, pp. 1108–1115.

[398] Shamir O & Tishby N, 2007, *Cluster stability for finite samples*, Proceedings of the 21st Annual Conference on Neural Information Processing Systems, Vancouver, pp. 150–158.

[399] Shannon CE, 2001, *A mathematical theory of communication*, Mobile Computing and Communications Review, **5(1)**, pp. 3–55.

[400] Shaw P, 1997, *A new local search algorithm providing high quality solutions to vehicle routing problems*, (Unpublished) Technical Report, APES Group, Department of Computer Science, University of Strathclyde, Glasgow.

[401] El-Sherbeny N, 2001, *Resolution of a vehicle routing problem with multi-objective simulated annealing method*, PhD Thesis, Université de Mons-Hainaut, Mons.

[402] Silveira J, 2006, *A brief history of health and medicine*, [Online], [accessed 10th October 2016], Available from: http://www.backwoodshome.com/a-brief-history-of-health-and-medicine/.

[403] Smith A, 2017, *TVRPGC travel times*, [Online], [accessed 10th September 2017], Available at: https://drive.google.com/drive/folders/0BwVjk2uwIjGlcUZYSmxJOGhwWjQ.

[404] Smith A, Toth P & Van Vuuren JH, *A multi-tiered vehicle routing problem with global cross-docking*, European Journal of Operational Research, *submitted*.

[405] Smith KI, Everson RM & Fieldsend JE, 2004, *Dominance measures for multi-objective simulated annealing*, Proceedings of the Congress on Evolutionary Computation, Portland (OR), pp. 23–30.

[406] Sneath PH & Sokal RR, 1973, *Numerical taxonomy: The principles and practice of numerical classification*, WH Freeman and Company, San Francisco (CA).

[407] Sokal RR, 1958, *A statistical method for evaluating systematic relationships*, University of Kansas Science Bulletin, **38**, pp. 1409–1438.

[408] Steinhaus H, 1956, *Sur la division des corp materiels en parties*, Bull. Acad. Polon. Sci, **1(804)**, pp. 801–804.

[409] Subramanian A, Uchoa E & Ochi LS, 2013, *A hybrid algorithm for a class of vehicle routing problems*, Computers and Operations Research, **40(10)**, pp. 2519–2531.

[410] Al-Sultan KS, 1995, *A tabu search approach to the clustering problem*, Pattern Recognition, **28(9)**, pp. 1443–1451.

[411] Sung CS & Jin HW, 2000, *A tabu-search-based heuristic for clustering*, Pattern Recognition, **33(5)**, pp. 849–858.

[412] Surya Sahoo P, 2004, *Balanced clustering algorithms for improving shapes on vehicle routing problems*, Proceedings of the Institute of Industrial Engineering Annual Conference, Norcross, pp. 1–6.

[413] Taillard ÉD, Gambardella LM, Gendreau M & Potvin JY, 2001, *Adaptive memory programming: A unified view of metaheuristics*, European Journal of Operational Research, **135(1)**, pp. 1–16.

[414] Tan KC, Chew Y & Lee LH, 2006, *A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems*, European Journal of Operational Research, **172(3)**, pp. 855–885.

[415] Tan WF, Lee LS, Abdul Majid Z & Seow HV, 2012, *Ant colony optimization for capacitated vehicle routing problem*, Journal of Computer Science, **8(6)**, pp. 846–852.

[416] Tang L & Wang X, 2006, *Iterated local search algorithm based on very large-scale neighborhood for prize-collecting vehicle routing problem*, International Journal of Advanced Manufacturing Technology, **29(11-12)**, pp. 1246–1258.

[417] Tarantilis CD, 2005, *Solving the vehicle routing problem with adaptive memory programming methodology*, Computers and Operations Research, **32(9)**, pp. 2309–2327.

[418] Tarantilis CD & Kiranoudis CT, 2002, *BoneRoute: An adaptive memory-based method for effective fleet management*, Annals of Operations Research, **115(1-4)**, pp. 227–241.

[419] Tasan AS & Gen M, 2012, *A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries*, Computers and Industrial Engineering, **62(3)**, pp. 755–761.

[420] Theodoridis S & Koutroumbas K, 1999, *Pattern recognition, Academic Press*, New York (NY).

[421] Theurkauff I, Cottin-Bizonne C, Palacci J, Ybert C & Bocquet L, 2012, *Dynamic clustering in active colloidal suspensions with chemical signaling*, Physical Review Letters, **108(26)**, pp. 268–303.

[422] Thompson PM & Psaraftis HN, 1993, *Cyclic transfer algorithm for multivehicle routing and scheduling problems*, Operations Research, **41(5)**, pp. 935–946.

[423] Tibshirani R, Walther G & Hastie T, 2001, *Estimating the number of clusters in a data set via the gap statistic*, Journal of the Royal Statistical Society: Series B (Statistical Methodology), **63(2)**, pp. 411–423.

[424] Tillman FA, 1969, *The multiple terminal delivery problem with probabilistic demands*, Transportation Science, **3(3)**, pp. 192–204.

[425] Toth P & Vigo D, 2001, *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia (PA).

[426] Toth P & Vigo D, 2001, *VRP with backhauls*, pp. 195–224 in Toth P & Vigo D (Eds), *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia (PA).

[427] Toth P & Vigo D, 2003, *The granular tabu search and its application to the vehicle-routing problem*, INFORMS Journal on Computing, **15(4)**, pp. 333–346.

[428] Toth P & Vigo D, 2014, *Vehicle routing: Problems, methods, and applications*, Society for Industrial and Applied Mathematics, Phildelphia (PA).

[429] Tragantalerngsak S, Holt J & Rönnqvist M, 2000, *An exact method for the two-echelon, single-source, capacitated facility location problem*, European Journal of Operational Research, **123(3)**, pp. 473–489.

[430] Tryon RC, 1939, *Cluster analysis: Correlation profile and orthometric (factor) analysis for the isolation of unities in mind and personality*, Edwards Brothers Malloy, Ann Arbor (MI).

[431] Tseng LY & Yang SB, 2001, *A genetic approach to the automatic clustering problem*, Pattern Recognition, **34(2)**, pp. 415–424.

[432] Tung AK, Han J, Lakshmanan LV & Ng RT, 2001, *Constraint-based clustering in large databases*, pp. 405–419 in Van den Bussche J & Vianu V (Eds), *Database Theory — ICDT 2001*, Springer, Berlin.

[433] UPS, 2015, *Driving success: Why the UPS model for managing 103 500 drivers is a competitive advantage*, [Online], [accessed 27th May 2015], Available from: `http://pressroom.ups.com/mediakits/popups/factsheet/0,1889,1201,00.html`.

[434] Van Breedam A, 1994, *An analysis of the behavior of heuristics for the vehicle routing problem for a selection of problems with vehicle-related, customer-related, and time-related constraints*, PhD Thesis, University of Antwerp, Antwerp.

[435] Vidal T, Crainic TG, Gendreau M, Lahrichi N & Rei W, 2012, *A hybrid genetic algorithm for multidepot and periodic vehicle routing problems*, Operations Research, **60(3)**, pp. 611–624.

[436] Vidal T, Crainic TG, Gendreau M & Prins C, 2014, *A unified solution framework for multi-attribute vehicle routing problems*, European Journal of Operational Research, **234(3)**, pp. 658–673.

[437] Vincent FY & Lin SW, 2014, *Multi-start simulated annealing heuristic for the location routing problem with simultaneous pickup and delivery*, Applied Soft Computing, **24**, pp. 284–290.

[438] Von Luxburg U & Ben-David S, 2005, *Towards a statistical theory of clustering*, Proceedings of the Pascal Workshop on Statistics and Optimization of Clustering, Tubingen, pp. 20–26.

[439] Wagstaff KL, 2006, *Value, cost, and sharing: Open issues in constrained clustering*, Springer, Berlin.

[440] Wallace CS & Freeman PR, 1987, *Estimation and inference by compact coding*, Journal of the Royal Statistical Society, Series B (Methodological), **49(2)**, pp. 240–265.

[441] Wallace CS & Boulton DM, 1968, *An information measure for classification*, The Computer Journal, **11(2)**, pp. 185–194.

[442] Walshaw C, 2002, *A multilevel approach to the travelling salesman problem*, Operations Research, **50(5)**, pp. 862–877.

[443] Wang CH & Lu JZ, 2009, *A hybrid genetic algorithm that optimizes capacitated vehicle routing problems*, Expert Systems with Applications, **36(2)**, pp. 2921–2936.

[444] Wang X, Choi TM, Liu H & Yue X, 2016, *Novel ant colony optimization methods for simplifying solution construction in vehicle routing problems*, IEEE Transactions on Intelligent Transportation Systems, **17(11)**, pp. 3132–3141.

[445] Ward Jr JH, 1963, *Hierarchical grouping to optimize an objective function*, Journal of the American Statistical Association, **58(301)**, pp. 236–244.

[446] Wark P & Holt J, 1994, *A repeated matching heuristic for the vehicle routeing problem*, Journal of the Operational Research Society, **45(10)**, pp. 1156–1167.

[447] Wen M, Larsen J, Clausen J, Cordeau JF & Laporte G, 2009, *Vehicle routing with cross-docking*, Journal of the Operational Research Society, **60(12)**, pp. 1708–1718.

[448] While L, Hingston P, Barone L & Huband S, 2006, *A faster algorithm for calculating hypervolume*, IEEE Transactions on Evolutionary Computation, **10(1)**, pp. 29–38.

[449] WHO, 2015, *New report shows that 400 million do not have access to essential health services*, [Online], [accessed 11th October 2016], Available at: http://www.who.int/mediacentre/news/releases/2015/uhc-report/en/.

[450] Wikipedia, 2016, *List of countries by life expectancy: Revision history*, [Online], [accessed 10th October 2016], Available at: https://en.wikipedia.org/w/index.php?title=List_of_countries_by_life_expectancy&action=history.

[451] Wikipedia, 2017, *HIV/AIDS in South Africa*, [Online], [accessed 5nd May 2016], Available at:: https://en.wikipedia.org/wiki/HIV/AIDS_in_South_Africa.

[452] Willard J, 1989, *Vehicle routing using r-optimal tabu search*, Master's Thesis, Imperial College, London.

[453] Williams J, Umaru F, Edgil D & Kuritsky J, 2016, *Progress in harmonizing tiered HIV laboratory systems: Challeges and oppurtunities in 8 African countries*, [Online], [accessed 22nd May 2016], Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5042701/.

[454] Wøhlk S, 2008, *A decade of capacitated arc routing*, pp. 29–48 in Golden BL, Raghavan S & Wasil EA (Eds), *The vehicle routing problem: Latest advances and new challenges*, Springer, Berlin.

[455] Wolfram-MathWorld, 2016, *Planar graphs*, [Online], [accessed 7th November 2016], Available from: http://mathworld.wolfram.com/PlanarGraph.html.

[456] Wolfram-MathWorld, 2017, *Wolfram Alpha*, [Online], [accessed 7th November 2017], Available from: https://www.wolframalpha.com/.

[457] Wren A & Holliday A, 1972, *Computer scheduling of vehicles from one or more depots to a number of delivery points*, Journal of Operations Research Society, **27(503)**, pp. 333–344.

[458] Wu X, Zhu X, Wu GQ & Ding W, 2014, *Data mining with big data*, IEEE Transactions on Knowledge and Data Engineering, **26(1)**, pp. 97–107.

[459] Xie XL & Beni G, 1991, *A validity measure for fuzzy clustering*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **13(8)**, pp. 841–847.

[460] Xing EP, Ng AY, Jordan MI & Russell S, 2003, *Distance metric learning with application to clustering with side-information*, Advances in Neural Information Processing Systems, **15**, pp. 505–512.

[461] Xu J & Kelly JP, 1996, *A network flow-based tabu search heuristic for the vehicle routing problem*, Transportation Science, **30(4)**, pp. 379–393.

[462] Xu R & Wunsch D, 2005, *Survey of clustering algorithms*, IEEE Transactions on Neural Networks, **16(3)**, pp. 645–678.

[463] Yang XS, 2010, *Engineering optimization: An introduction with metaheuristic applications*, John Wiley & Sons, Hoboken (NJ).

[464] Yu TL, Sastry K, Goldberg DE & Pelikan M, 2006, *Population sizing for entropy-based model building in genetic algorithms*, (Unpublished) Technical Report IlliGAL20, Illinois Genetic Algorithms Laboratory, University of Illnois, Urbana (IL).

[465] Yücenur GN & Demirel NÇ, 2011, *A new geometric shape-based genetic clustering algorithm for the multi-depot vehicle routing problem*, Expert Systems with Applications, **38(9)**, pp. 11859–11865.

[466] Zachariadis EE & Kiranoudis CT, 2010, *A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem*, Computers and Operations Research, **37(12)**, pp. 2089–2105.

[467] Zadeh LA, 1978, *Fuzzy sets as a basis for a theory of possibility*, Fuzzy Sets and Systems, **1(1)**, pp. 3–28.

[468] Zhang T, Ramakrishnan R & Livny M, 1997, *BIRCH: A new data clustering algorithm and its applications*, Data Mining and Knowledge Discovery, **1(2)**, pp. 141–182.

[469] Zitzler E, Brockhoff D & Thiele L, 2007, *The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration*, Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization, Matsushima, pp. 862–876.

[470] Zitzler E & Thiele L, 1999, *Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach*, IEEE Transactions on Evolutionary Computation, **3(4)**, pp. 257–271.

[471] Zitzler E, Thiele L, Laumanns M, Fonseca CM & Da Fonseca VG, 2003, *Performance assessment of multiobjective optimizers: An analysis and review*, IEEE Transactions on Evolutionary Computation, **7(2)**, pp. 117–132.

[472] Zografos KG & Androutsopoulos KN, 2004, *A heuristic algorithm for solving hazardous materials distribution problems*, European Journal of Operational Research, **152(2)**, pp. 507–519.

APPENDIX A

Christofides and Eilon Benchmark Test Instances

This appendix contains specifications for three well-known CVRP benchmark instances of Christofides and Eilon [82] considered in this dissertation when performing sensitivity analyses and producing the results reported in Chapter 7.

A.1 The E-n22-k4 CVRP test instance

Data pertaining to the E-n22-k4 CVRP instance of Christofides and Eilon [82] are provided in Table A.1. These data include the coordinates of the 22 customers in the Euclidean plane, as well as the demand of each customer.

Table A.1: *Location and demand for each customer in benchmark instance E-n22-k4 [82].*

Customer	X-coordinate	Y-coordinate	Demand
1	145	215	0
2	151	264	1 100
3	159	261	700
4	130	254	800
5	128	252	1 400
6	163	247	2 100
7	146	246	400
8	161	242	800
9	142	239	100
10	163	236	500
11	148	232	600
12	128	231	1 200
13	156	217	1 300
14	129	214	1 300
15	146	208	300
16	164	208	900
17	141	206	2 100
18	147	193	1 000
19	164	193	900

Customer	X-coordinate	Y-coordinate	Demand
20	129	189	2 500
21	155	185	1 800
22	139	182	700

A.2 The E-n51-k5 CVRP test instance

Data pertaining to the E-n51-k5 CVRP instance of Christofides and Eilon [82] are provided in Table A.2. These data include the coordinates of the 51 customers in the Euclidean plane, as well as the demand of each customer.

Table A.2: *Location and demand for each customer in benchmark instance E-n51-k8 [82].*

Customer	X-coordinate	Y-coordinate	Demand
1	30	40	0
2	37	52	7
3	49	49	30
4	52	64	16
5	20	26	9
6	40	30	21
7	21	47	15
8	17	63	19
9	31	62	23
10	52	33	11
11	51	21	5
12	42	41	19
13	31	32	29
14	5	25	23
15	12	42	21
16	36	16	10
17	52	41	15
18	27	23	3
19	17	33	41
20	13	13	9
21	57	58	28
22	62	42	8
23	42	57	8
24	16	57	16
25	8	52	10
26	7	38	28
27	27	68	7
28	30	48	15
29	43	67	14
30	58	48	6
31	58	27	19
32	37	69	11
33	38	46	12
34	46	10	23
35	61	33	26

Customer	X-coordinate	Y-coordinate	Demand
36	62	63	17
37	63	69	6
38	32	22	9
39	45	35	15
40	59	15	14
41	5	6	7
42	10	17	27
43	21	10	13
44	5	64	11
45	30	15	16
46	39	10	10
47	32	39	5
48	25	32	25
49	25	55	17
50	48	28	18
51	56	37	10

A.3 The E-n76-k8 CVRP test instance

Data pertaining to the E-n76-k8 CVRP instance of Christofides and Eilon [82] are provided in Table A.3. These data include the coordinates of the 76 customers in the Euclidean plane, as well as the demand of each customer.

TABLE A.3: *Location and demand for each customer in benchmark instance E-n76-k8 [82].*

Customer	X-coordinate	Y-coordinate	Demand
1	40	40	0
2	22	22	18
3	36	26	26
4	21	45	11
5	45	35	30
6	55	20	21
7	33	34	19
8	50	50	15
9	55	45	16
10	26	59	29
11	40	66	26
12	55	65	37
13	35	51	16
14	62	35	12
15	62	57	31
16	62	24	8
17	21	36	19
18	33	44	20
19	9	56	13
20	62	48	15
21	66	14	22
22	44	13	28

Customer	X-coordinate	Y-coordinate	Demand
23	26	13	12
24	11	28	6
25	7	43	27
26	17	64	14
27	41	46	18
28	55	34	17
29	35	16	29
30	52	26	13
31	43	26	22
32	31	76	25
33	22	53	28
34	26	29	27
35	50	40	19
36	55	50	10
37	54	10	12
38	60	15	14
39	47	66	24
40	30	60	16
41	30	50	33
42	12	17	15
43	15	14	11
44	16	19	18
45	21	48	17
46	50	30	21
47	51	42	27
48	50	15	19
49	48	21	20
50	12	38	5
51	15	56	22
52	29	39	12
53	54	38	19
54	55	57	22
55	67	41	16
56	10	70	7
57	6	25	26
58	65	27	14
59	40	60	21
60	70	64	24
61	64	4	13
62	36	6	15
63	30	20	18
64	20	30	11
65	15	5	28
66	50	70	9
67	57	72	37
68	45	42	30
69	38	33	10
70	50	4	8
71	66	8	11

Customer	X-coordinate	Y-coordinate	Demand
72	59	5	3
73	35	60	1
74	27	24	6
75	40	20	10
76	40	37	20

APPENDIX B

Vehicle Routes for the George TVRPGC Instance

This appendix contains the vehicle routes returned by the MACO algorithm of §10.4 in tabular form for the George TVRPGC instance. The vehicle routes correspond to the solutions depicted in objective function space in Figure 11.2. The three values highlighted in boldface in each table correspond the respective objective function values for each solution.

Table B.1: *The five vehicle routes proposed by the MACO algorithm in Solution 1 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 17, 23, 24, 43, 16, 15, 14, 12, 2, 3, 5, 1, 8, 6, 10, 7, 9, 4, 62, 11, 18, 19, 38, 54, 52, 53, 51, 65, 58, 61, 55, 56, 57, 20, 67, 68	681.91
2	68, 21, 31, 47, 42, 41, 45, 46, 44, 29, 30, 28, 27, 25, 26, 59, 60, 37, 36, 34, 35, 33, 67, 68	612.45
3	68, 22, 32, 40, 64, 50, 67, 68	699.46
4	68, 48, 49, 39, 13, 67, 68	**705.75**
5	68, 67, 63, 66, 68	622.41
	Total	**3 321.98**

Table B.2: *The five vehicle routes proposed by the MACO algorithm in Solution 2 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 8, 6, 10, 9, 1, 4, 21, 31, 43, 47, 46, 45, 42, 63, 41, 44, 30, 27, 28, 25, 26, 58, 59, 60, 61, 65, 55, 56, 57, 67, 68	640.37
2	68, 20, 23, 15, 16, 22, 19, 38, 37, 36, 34, 33, 35, 40, 64, 50, 51, 53, 52, 54, 67, 68	**707.11**
3	68, 17, 24, 14, 29, 32, 2, 5, 7, 3, 62, 11, 12, 13, 18, 67, 68	436.99
4	68, 49, 39, 48, 67, 68	655.75
5	68, 67, 66, 68	611.94
	Total	**3 052.16**

Table B.3: *The five vehicle routes proposed by the MACO algorithm in Solution 3 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 56, 55, 65, 54, 52, 53, 64, 40, 35, 33, 34, 36, 37, 45, 67, 68	**740.46**
2	68, 20, 17, 23, 24, 15, 16, 22, 14, 11, 2, 5, 4, 9, 7, 10, 6, 8, 1, 3, 62, 12, 13, 32, 31, 46, 42, 63, 41, 44, 47, 43, 21, 18, 19, 38, 57, 50, 51, 67, 68	596.11
3	68, 61, 60, 59, 58, 26, 25, 27, 28, 30, 29, 67, 68	447.52
4	68, 49, 39, 48, 67, 68	655.75
5	68, 67, 66, 68	611.94
	Total	**3 051.77**

Table B.4: *The four vehicle routes proposed by the MACO algorithm in Solution 4 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 20, 19, 38, 61, 60, 59, 58, 26, 25, 28, 27, 30, 29, 44, 46, 45, 41, 63, 42, 47, 43, 31, 21, 13, 12, 2, 5, 1, 10, 6, 8, 7, 9, 4, 3, 62, 11, 14, 22, 16, 15, 24, 23, 17, 18, 67, 68	**775.85**
2	68, 37, 36, 34, 33, 35, 40, 64, 50, 51, 53, 52, 54, 65, 55, 56, 57, 67, 68	690.35
3	68, 49, 39, 48, 32, 67, 68	744.95
4	68, 67, 66, 68	611.94
	Total	**2 823.09**

Table B.5: *The four vehicle routes proposed by the MACO algorithm in Solution 5 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 38, 37, 36, 34, 33, 35, 40, 64, 50, 51, 53, 52, 54, 57, 56, 55, 65, 19, 20, 67, 68	692.62
2	68, 17, 23, 24, 15, 16, 22, 14, 26, 25, 28, 27, 30, 29, 44, 41, 63, 42, 46, 45, 47, 43, 32, 31, 21, 11, 2, 4, 9, 7, 10, 6, 8, 1, 5, 3, 62, 12, 13, 18, 67, 68	656.68
3	68, 61, 60, 59, 58, 39, 49, 48, 67, 68	**801.38**
4	68, 67, 66, 68	611.94
	Total	**2 762.63**

Table B.6: *The four vehicle routes proposed by the MACO algorithm in Solution 6 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 54, 52, 53, 51, 50, 64, 39, 49, 48, 57, 56, 55, 65, 19, 20, 67, 68	717.85
2	68, 18, 17, 14, 21, 13, 12, 11, 62, 2, 3, 5, 4, 1, 9, 8, 6, 10, 7, 37, 36, 34, 33, 35, 40, 67, 68	**867.90**
3	68, 24, 23, 22, 16, 15, 32, 31, 43, 47, 45, 46, 42, 63, 41, 44, 29, 30, 27, 28, 25, 26, 58, 59, 60, 61, 38, 67, 68	555.76
4	68, 67, 66, 68	611.94
	Total	**2 748.46**

Table B.7: *The four vehicle routes proposed by the MACO algorithm in Solution 7 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 20, 18, 17, 23, 24, 22, 16, 15, 14, 21, 31, 43, 47, 42, 63, 41, 45, 46, 44, 29, 30, 28, 27, 26, 25, 58, 59, 60, 61, 48, 49, 50, 51, 53, 52, 54, 57, 56, 55, 65, 38, 19, 67, 68	726.47
2	68, 32, 13, 12, 11, 62, 2, 3, 5, 4, 1, 9, 10, 6, 8, 7, 37, 36, 34, 35, 33, 67, 68	459.27
3	68, 64, 40, 39, 67, 68	**908.78**
4	68, 67, 66, 68	611.94
	Total	**2 706.46**

Table B.8: *The four vehicle routes proposed by the MACO algorithm in Solution 8 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 20, 38, 19, 31, 32, 43, 47, 46, 45, 42, 63, 41, 44, 27, 28, 25, 26, 30, 29, 21, 13, 15, 16, 22, 14, 24, 23, 17, 18, 67, 68	463.13
2	68, 57, 56, 55, 58, 59, 60, 61, 65, 54, 52, 53, 51, 50, 35, 33, 34, 36, 37, 7, 8, 6, 10, 9, 1, 4, 5, 3, 2, 62, 11, 12, 67, 68	668.63
3	68, 48, 49, 39, 40, 64, 67, 68	**911.08**
4	68, 67, 66, 68	611.94
	Total	**2 654.77**

Table B.9: *The four vehicle routes proposed by the MACO algorithm in Solution 9 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 64, 40, 39, 49, 48, 57, 19, 20, 67, 68	**913.79**
2	68, 38, 65, 56, 55, 54, 52, 53, 51, 50, 35, 33, 34, 36, 37, 7, 10, 6, 8, 9, 1, 4, 5, 3, 2, 62, 11, 12, 13, 21, 44, 41, 63, 42, 46, 45, 47, 43, 32, 14, 22, 16, 15, 24, 23, 17, 18, 67, 68	659.93
3	68, 61, 60, 59, 58, 26, 25, 27, 28, 30, 29, 31, 67, 68	451.42
4	68, 67, 66, 68	611.94
	Total	**2 637.08**

Table B.10: *The four vehicle routes proposed by the MACO algorithm in Solution 10 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 20, 19, 57, 56, 55, 61, 60, 59, 58, 26, 25, 28, 27, 30, 29, 44, 41, 63, 42, 45, 46, 47, 43, 32, 31, 21, 12, 11, 13, 14, 15, 16, 22, 24, 23, 17, 18, 67, 68	696.04
2	68, 54, 52, 53, 51, 50, 40, 64, 39, 49, 48, 65, 38, 67, 68	**952.74**
3	68, 33, 35, 34, 36, 37, 7, 10, 6, 8, 9, 1, 4, 5, 3, 2, 62, 67, 68	313.10
4	68, 67, 66, 68	611.94
	Total	**2 573.82**

Table B.11: *The three vehicle routes proposed by the MACO algorithm in Solution 11 of Figure 11.2 for the George TVRPGC instance.*

Route	Customers	Autonomy
1	68, 20, 19, 65, 55, 56, 57, 48, 49, 39, 40, 64, 53, 52, 54, 38, 18, 21, 67, 68	**965.14**
2	68, 13, 12, 11, 62, 2, 3, 5, 4, 1, 9, 10, 6, 8, 7, 37, 36, 34, 33, 35, 50, 51, 61, 60, 59, 58, 26, 25, 28, 27, 30, 29, 44, 14, 22, 16, 15, 24, 23, 17, 67, 68	846.99
3	68, 41, 63, 42, 46, 45, 47, 43, 32, 31, 67, 66, 68	746.52
	Total	**2 558.65**

APPENDIX C

Vehicle Routes for the Western Cape TVRPGC Instance

This appendix contains the vehicle routes returned by the MACO algorithm of §10.4 in tabular form for the Western Cape TVRPGC instance. The vehicle routes correspond to the solutions depicted in objective function space in Figure 13.1. The three values highlighted in boldface in each table correspond the respective objective function values for each solution.

TABLE C.1: *The twenty five vehicle routes proposed by the MACO algorithm in Solution 1 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	386, 174, 369, 184, 185, 294, 300, 299, 305, 308, 297, 309, 310, 377, 360, 288, 74, 287, 282, 286, 283, 252, 292, 212, 303, 207, 371, 213, 304, 290, 3, 376, 257, 206, 279, 280, 256, 198, 197, 196, 210, 194, 5, 262, 366, 277, 193, 190, 192, 191, 260, 239, 276, 272, 273, 274, 367, 269, 258, 243, 244, 242, 291, 268, 56, 240, 176, 377, 386	499.00
2	385, 202, 201, 281, 203, 189, 295, 9, 209, 204, 238, 235, 236, 4, 222, 223, 230, 10, 226, 227, 11, 225, 224, 221, 220, 219, 237, 217, 216, 255, 205, 265, 208, 296, 284, 285, 253, 254, 356, 361, 250, 251, 289, 264, 267, 377, 385	575.25
3	384, 75, 52, 53, 54, 49, 372, 57, 50, 18, 48, 45, 41, 39, 42, 40, 38, 29, 34, 37, 36, 70, 80, 76, 71, 373, 72, 377, 384	585.26
4	384, 307, 306, 322, 293, 318, 81, 78, 77, 84, 79, 69, 68, 67, 97, 100, 99, 66, 63, 65, 96, 85, 86, 89, 88, 91, 92, 105, 109, 62, 83, 60, 59, 82, 367, 384	546.53
5	384, 211, 6, 259, 127, 119, 362, 374, 128, 200, 7, 199, 278, 263, 195, 325, 377, 384	783.63
6	384, 245, 64, 95, 94, 93, 355, 104, 103, 110, 102, 101, 108, 106, 367, 384	610.93
7	382, 163, 162, 164, 167, 148, 130, 131, 141, 123, 122, 117, 147, 146, 144, 143, 365, 382	718.25
8	382, 152, 156, 155, 154, 140, 374, 382	600.16
9	384, 313, 188, 317, 316, 323, 339, 370, 342, 354, 352, 321, 312, 311, 320, 315, 319, 377, 384	603.14
10	378, 270, 271, 175, 178, 179, 331, 330, 333, 334, 336, 337, 23, 24, 335, 30, 332, 32, 33, 338, 180, 177, 182, 173, 181, 172, 183, 186, 73, 377, 378	599.47
11	384, 215, 241, 314, 246, 298, 90, 345, 351, 353, 358, 357, 249, 326, 8, 248, 247, 266, 367, 384	566.85
12	385, 25, 27, 26, 22, 136, 137, 139, 151, 363, 125, 124, 129, 365, 165, 377, 385	749.02
13	383, 232, 2, 231, 233, 376, 383	707.48
14	387, 350, 347, 346, 348, 376, 387	768.47
15	380, 111, 120, 116, 118, 114, 115, 113, 112, 121, 187, 214, 367, 380	764.46
16	383, 166, 149, 364, 383	594.07
17	388, 344, 349, 343, 341, 340, 360, 388	754.00
18	385, 329, 368, 327, 31, 17, 20, 16, 19, 28, 15, 14, 12, 21, 13, 46, 377, 385	641.04
19	386, 87, 35, 43, 359, 47, 51, 55, 58, 328, 369, 386	**827.25**
20	385, 275, 150, 364, 385	795.41
21	387, 171, 170, 169, 168, 261, 229, 1, 228, 234, 218, 376, 387	678.54
22	380, 145, 159, 158, 160, 161, 157, 153, 142, 126, 132, 134, 133, 374, 380	617.21
23	388, 98, 138, 135, 302, 301, 360, 388	806.15
24	381, 324, 44, 372, 381	636.70
25	384, 107, 61, 364, 375, 384	790.58
	Total	**16 818.83**

Table C.2: *The twenty vehicle routes proposed by the MACO algorithm in Solution 2 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	384, 194, 325, 211, 210, 196, 197, 195, 280, 281, 203, 189, 295, 266, 267, 206, 249, 8, 326, 248, 247, 319, 315, 320, 324, 287, 74, 282, 3, 303, 375, 212, 292, 207, 296, 208, 261, 371, 213, 304, 230, 229, 231, 1, 2, 290, 232, 233, 234, 222, 227, 226, 10, 11, 225, 224, 221, 220, 219, 4, 236, 235, 237, 238, 217, 216, 218, 257, 265, 205, 255, 204, 209, 9, 264, 202, 7, 200, 199, 263, 256, 198, 6, 259, 258, 291, 268, 314, 246, 309, 310, 12, 298, 242, 241, 215, 240, 175, 272, 271, 273, 270, 274, 367, 187, 214, 269, 260, 275, 276, 239, 191, 190, 277, 192, 262, 5, 375, 384	621.94
2	386, 30, 33, 32, 73, 186, 183, 172, 181, 184, 62, 109, 101, 102, 361, 108, 107, 106, 178, 174, 369, 176, 177, 180, 375, 386	608.66
3	388, 340, 341, 343, 342, 355, 356, 104, 103, 110, 105, 93, 92, 88, 87, 89, 373, 388	753.74
4	378, 13, 15, 25, 26, 27, 136, 135, 21, 22, 23, 24, 337, 377, 378	491.28
5	381, 151, 154, 139, 140, 138, 137, 14, 168, 169, 170, 171, 165, 365, 381	577.36
6	385, 366, 193, 279, 278, 201, 313, 311, 312, 360, 317, 316, 188, 377, 321, 323, 288, 253, 254, 252, 284, 283, 285, 286, 289, 251, 250, 61, 83, 60, 59, 82, 63, 245, 66, 99, 98, 100, 81, 71, 373, 52, 53, 54, 57, 372, 49, 58, 72, 182, 173, 185, 306, 307, 308, 305, 322, 294, 318, 293, 302, 301, 300, 299, 297, 56, 243, 244, 376, 385	508.80
7	378, 17, 155, 152, 16, 20, 18, 372, 378	558.30
8	378, 84, 76, 77, 78, 79, 80, 70, 69, 68, 67, 97, 34, 38, 42, 39, 41, 46, 45, 359, 47, 48, 51, 55, 373, 378	572.20
9	385, 223, 228, 75, 65, 64, 95, 94, 96, 85, 86, 37, 36, 35, 40, 43, 44, 29, 50, 19, 28, 332, 31, 329, 328, 368, 327, 330, 331, 338, 179, 375, 385	772.58
10	382, 114, 111, 119, 120, 116, 118, 117, 147, 146, 365, 382	638.26
11	387, 129, 148, 167, 130, 365, 374, 128, 127, 133, 363, 375, 387	823.57
12	385, 131, 123, 142, 141, 153, 157, 369, 385	679.84
13	380, 121, 122, 134, 124, 132, 126, 125, 156, 164, 162, 163, 161, 160, 158, 145, 143, 144, 115, 113, 112, 374, 380	696.34
14	387, 345, 344, 370, 339, 358, 357, 375, 387	568.93
15	385, 346, 347, 376, 385	643.16
16	382, 150, 149, 159, 166, 365, 382	826.15
17	386, 333, 334, 336, 335, 90, 91, 352, 353, 369, 386	593.83
18	385, 351, 350, 348, 376, 385	634.76
19	381, 362, 364, 375, 381	**857.57**
20	88, 354, 349, 360, 388	772.94
	Total	**13 200.20**

Table C.3: *The eighteen vehicle routes proposed by the MACO algorithm in Solution 3 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	385, 274, 276, 239, 275, 260, 269, 191, 192, 262, 366, 5, 194, 193, 277, 325, 195, 7, 201, 281, 203, 189, 295, 266, 202, 247, 248, 8, 326, 249, 206, 267, 264, 265, 257, 376, 3, 212, 303, 207, 292, 282, 287, 324, 320, 315, 311, 312, 313, 377, 360, 323, 321, 310, 246, 309, 299, 300, 301, 302, 293, 318, 322, 294, 100, 98, 67, 68, 69, 70, 80, 79, 373, 71, 81, 84, 78, 77, 76, 55, 51, 48, 17, 15, 13, 12, 14, 16, 20, 18, 50, 372, 57, 49, 54, 58, 52, 72, 73, 186, 183, 172, 181, 185, 173, 182, 175, 272, 271, 273, 270, 244, 243, 242, 241, 215, 240, 56, 298, 297, 314, 268, 291, 196, 210, 211, 6, 259, 258, 214, 187, 376, 385	685.00
2	382, 167, 166, 165, 365, 171, 170, 169, 168, 137, 138, 140, 376, 382	912.54
3	378, 288, 289, 82, 59, 60, 83, 61, 62, 356, 355, 104, 103, 110, 102, 109, 101, 361, 108, 107, 106, 250, 251, 376, 378	595.20
4	382, 145, 143, 144, 146, 147, 148, 129, 130, 162, 163, 161, 160, 365, 382	557.62
5	387, 174, 178, 290, 232, 2, 1, 233, 176, 335, 24, 23, 22, 135, 136, 26, 27, 25, 337, 336, 334, 333, 328, 368, 327, 329, 330, 180, 177, 376, 387	623.38
6	383, 285, 286, 372, 383	684.16
7	380, 121, 122, 21, 131, 128, 127, 133, 134, 124, 123, 112, 119, 117, 118, 116, 120, 111, 114, 115, 113, 374, 380	715.58
8	381, 9, 222, 4, 236, 235, 32, 372, 381	635.76
9	378, 66, 245, 63, 65, 64, 105, 93, 92, 90, 91, 88, 89, 87, 86, 85, 37, 36, 35, 34, 38, 42, 39, 41, 46, 359, 47, 372, 378	720.07
10	378, 252, 253, 254, 283, 284, 371, 213, 261, 208, 296, 231, 304, 229, 230, 10, 227, 226, 11, 225, 224, 223, 228, 221, 220, 219, 237, 234, 218, 238, 217, 216, 255, 205, 209, 204, 190, 367, 338, 31, 33, 19, 28, 30, 332, 179, 369, 184, 376, 378	590.57
11	385, 305, 75, 97, 29, 43, 44, 40, 96, 95, 352, 354, 357, 358, 99, 306, 307, 308, 188, 317, 316, 74, 319, 200, 199, 278, 279, 263, 280, 256, 198, 197, 366, 385	637.67
12	378, 149, 364, 376, 378	805.96
13	381, 152, 157, 153, 142, 141, 132, 126, 125, 158, 159, 164, 139, 154, 156, 155, 151, 374, 381	564.59
14	388, 340, 341, 345, 348, 346, 347, 349, 350, 351, 344, 360, 388	**931.08**
15	387, 353, 339, 370, 343, 342, 376, 387	562.63
16	385, 53, 150, 45, 372, 385	825.88
17	386, 331, 94, 362, 374, 375, 386	939.12
18	386, 363, 376, 386	592.09
	Total	**12 578.89**

TABLE C.4: *The seventeen vehicle routes proposed by the MACO algorithm in Solution 4 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	385, 176, 177, 180, 330, 329, 327, 368, 331, 338, 179, 178, 369, 184, 181, 185, 173, 182, 294, 322, 318, 293, 302, 301, 300, 299, 309, 246, 310, 321, 317, 316, 74, 287, 324, 320, 315, 247, 319, 314, 268, 291, 297, 298, 56, 242, 243, 244, 258, 259, 6, 210, 196, 197, 198, 280, 263, 279, 278, 199, 200, 7, 281, 201, 202, 295, 189, 203, 266, 267, 206, 282, 286, 284, 283, 252, 3, 212, 303, 207, 296, 208, 261, 371, 213, 230, 229, 304, 231, 234, 233, 1, 2, 232, 290, 205, 255, 257, 265, 264, 204, 209, 9, 216, 217, 238, 218, 235, 236, 4, 219, 220, 221, 222, 228, 223, 10, 226, 227, 11, 225, 224, 190, 193, 5, 366, 262, 375, 385	623.22
2	379, 101, 109, 102, 110, 103, 104, 105, 93, 241, 215, 240, 175, 272, 271, 273, 270, 274, 367, 269, 260, 275, 239, 191, 192, 313, 311, 312, 377, 360, 323, 288, 289, 251, 250, 106, 107, 108, 371, 379	627.95
3	386, 174, 172, 183, 306, 305, 307, 308, 188, 75, 186, 73, 72, 81, 71, 373, 41, 46, 47, 48, 51, 55, 58, 54, 49, 372, 375, 386	586.59
4	380, 121, 122, 123, 131, 152, 151, 155, 156, 157, 153, 141, 142, 144, 146, 147, 117, 118, 119, 111, 114, 115, 113, 112, 374, 380	595.70
5	385, 276, 194, 325, 277, 195, 211, 124, 134, 133, 127, 374, 130, 128, 214, 187, 375, 385	659.31
6	384, 256, 248, 249, 326, 8, 292, 253, 254, 82, 59, 60, 83, 61, 62, 63, 245, 65, 64, 66, 99, 100, 98, 67, 68, 69, 70, 80, 79, 84, 78, 77, 76, 19, 28, 15, 13, 12, 53, 52, 367, 384	561.15
7	388, 340, 341, 343, 342, 355, 356, 354, 352, 353, 358, 357, 339, 345, 351, 344, 360, 388	923.05
8	382, 160, 161, 163, 162, 164, 167, 166, 165, 365, 143, 145, 371, 382	**939.25**
9	384, 125, 126, 362, 363, 371, 384	716.57
10	387, 332, 237, 285, 361, 92, 90, 91, 88, 89, 87, 86, 85, 40, 43, 44, 29, 42, 39, 38, 34, 35, 37, 36, 96, 94, 95, 97, 371, 387	712.43
11	378, 32, 33, 31, 30, 328, 333, 334, 335, 336, 337, 24, 27, 26, 17, 16, 20, 18, 50, 57, 372, 378	473.24
12	388, 348, 346, 347, 349, 350, 360, 388	764.92
13	380, 154, 139, 140, 138, 137, 168, 169, 170, 171, 158, 148, 129, 374, 380	655.42
14	386, 359, 45, 14, 25, 23, 136, 135, 21, 22, 369, 386	847.78
15	382, 150, 120, 116, 132, 374, 382	749.49
16	383, 149, 159, 374, 383	658.44
17	378, 370, 364, 376, 378	914.88
	Total	**12 009.38**

TABLE C.5: *The fifteen vehicle routes proposed by the MACO algorithm in Solution 5 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	384, 5, 262, 192, 191, 190, 277, 193, 325, 195, 263, 197, 196, 198, 256, 319, 247, 248, 8, 326, 249, 206, 3, 290, 232, 2, 1, 233, 231, 223, 228, 222, 235, 236, 4, 219, 220, 221, 224, 225, 11, 10, 226, 227, 230, 229, 304, 371, 213, 261, 208, 296, 207, 375, 303, 212, 252, 283, 284, 286, 285, 282, 74, 287, 324, 320, 315, 311, 313, 372, 377, 321, 323, 360, 310, 182, 173, 185, 184, 181, 172, 183, 186, 73, 33, 31, 32, 30, 332, 330, 331, 338, 179, 180, 177, 369, 174, 178, 175, 272, 271, 273, 270, 274, 276, 239, 260, 269, 367, 187, 214, 258, 259, 6, 210, 211, 194, 371, 384	754.29
2	381, 151, 7, 124, 125, 126, 132, 134, 133, 127, 128, 130, 122, 113, 111, 119, 117, 120, 116, 118, 115, 112, 123, 131, 142, 141, 153, 157, 155, 156, 152, 374, 381	**969.43**
3	378, 58, 54, 49, 57, 50, 18, 20, 16, 17, 161, 160, 158, 159, 45, 46, 47, 48, 51, 55, 373, 378	762.52
4	384, 291, 242, 243, 240, 215, 241, 56, 297, 298, 309, 299, 300, 301, 302, 293, 318, 294, 322, 305, 308, 307, 306, 75, 81, 71, 373, 84, 76, 77, 78, 79, 80, 70, 69, 68, 67, 97, 98, 100, 99, 82, 59, 60, 83, 61, 62, 361, 108, 107, 106, 250, 251, 289, 288, 188, 317, 316, 312, 246, 314, 268, 279, 278, 199, 200, 202, 266, 267, 264, 265, 257, 292, 254, 253, 234, 237, 218, 217, 238, 255, 205, 9, 209, 204, 203, 189, 295, 201, 281, 280, 375, 384	615.45
5	384, 353, 352, 354, 351, 348, 345, 341, 340, 343, 370, 344, 339, 371, 384	839.59
6	379, 101, 109, 14, 15, 12, 13, 21, 22, 24, 26, 27, 25, 28, 19, 53, 52, 360, 379	650.46
7	383, 171, 170, 169, 168, 137, 138, 140, 139, 154, 121, 148, 164, 162, 163, 167, 166, 165, 374, 383	644.97
8	383, 150, 145, 143, 144, 146, 147, 129, 374, 383	652.61
9	378, 38, 41, 39, 42, 40, 43, 44, 29, 34, 35, 36, 37, 85, 86, 87, 89, 88, 91, 90, 92, 93, 105, 104, 103, 110, 102, 361, 378	566.95
10	387, 244, 275, 366, 216, 63, 245, 65, 64, 94, 95, 96, 72, 23, 136, 135, 336, 176, 371, 387	629.54
11	379, 329, 66, 357, 358, 342, 355, 356, 360, 379	847.10
12	386, 335, 337, 334, 333, 328, 327, 363, 362, 374, 376, 386	912.37
13	380, 114, 365, 364, 371, 380	872.29
14	386, 350, 349, 347, 346, 369, 386	872.74
15	387, 368, 359, 149, 371, 387	846.70
	Total	**11 436.99**

Table C.6: *The fourteen vehicle routes proposed by the MACO algorithm in Solution 6 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	387, 174, 178, 176, 177, 180, 338, 331, 330, 368, 328, 333, 334, 335, 336, 337, 25, 27, 26, 23, 22, 21, 13, 12, 14, 17, 16, 20, 18, 50, 57, 372, 49, 58, 54, 53, 52, 76, 77, 78, 84, 79, 80, 70, 69, 68, 67, 98, 100, 373, 71, 81, 75, 306, 307, 308, 305, 322, 294, 182, 173, 185, 181, 184, 375, 387	725.33
2	380, 112, 113, 115, 114, 111, 119, 120, 116, 118, 117, 147, 146, 144, 143, 145, 160, 161, 163, 162, 164, 365, 171, 170, 169, 168, 137, 372, 380	950.59
3	385, 247, 248, 8, 326, 249, 206, 287, 74, 360, 323, 288, 289, 92, 91, 90, 93, 105, 104, 103, 110, 102, 109, 101, 361, 108, 107, 106, 61, 83, 60, 59, 82, 250, 251, 254, 253, 252, 284, 283, 286, 282, 208, 261, 296, 207, 303, 375, 212, 3, 376, 257, 265, 264, 267, 266, 295, 202, 366, 385	574.54
4	386, 179, 369, 240, 215, 241, 242, 291, 268, 313, 321, 66, 99, 318, 293, 302, 301, 300, 299, 309, 246, 310, 188, 316, 317, 312, 311, 315, 320, 324, 285, 213, 371, 292, 290, 232, 2, 1, 233, 234, 228, 10, 223, 222, 219, 235, 236, 4, 237, 218, 238, 217, 216, 255, 205, 204, 209, 9, 203, 189, 281, 201, 7, 200, 199, 278, 279, 280, 195, 256, 263, 198, 197, 196, 211, 210, 194, 277, 325, 193, 190, 191, 192, 5, 366, 262, 6, 259, 258, 214, 187, 367, 269, 260, 275, 239, 274, 270, 273, 271, 272, 175, 377, 386	587.01
5	381, 151, 138, 24, 28, 15, 19, 48, 46, 47, 11, 136, 135, 140, 139, 154, 374, 381	908.99
6	378, 339, 345, 351, 344, 340, 341, 343, 342, 354, 355, 356, 377, 378	719.94
7	387, 276, 225, 226, 227, 230, 229, 304, 231, 319, 314, 97, 94, 96, 86, 88, 358, 357, 89, 87, 85, 36, 37, 35, 34, 38, 39, 41, 42, 40, 43, 44, 29, 298, 297, 56, 243, 244, 367, 387	665.72
8	383, 166, 165, 374, 127, 133, 134, 132, 126, 125, 124, 123, 131, 141, 142, 152, 363, 155, 156, 157, 153, 128, 130, 129, 148, 167, 158, 159, 375, 383	**1 011.22**
9	384, 329, 327, 30, 332, 33, 31, 32, 73, 72, 186, 183, 172, 64, 65, 352, 353, 370, 63, 245, 62, 95, 372, 384	867.71
10	385, 224, 221, 220, 55, 51, 45, 149, 359, 372, 385	782.64
11	386, 346, 347, 348, 369, 386	766.29
12	380, 121, 122, 150, 364, 372, 380	992.53
13	382, 350, 349, 372, 382	941.87
14	379, 362, 372, 379	843.70
	Total	**11 338.12**

TABLE C.7: *The fifteen vehicle routes proposed by the MACO algorithm in Solution 7 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	385, 366, 5, 262, 191, 190, 238, 217, 216, 218, 237, 235, 236, 4, 219, 220, 221, 224, 223, 228, 222, 234, 233, 1, 304, 230, 229, 231, 2, 232, 290, 303, 375, 212, 292, 207, 371, 252, 253, 254, 82, 59, 60, 83, 62, 61, 250, 323, 360, 310, 309, 298, 297, 314, 246, 188, 377, 313, 315, 320, 324, 311, 312, 317, 316, 287, 74, 282, 285, 286, 283, 284, 3, 257, 265, 205, 255, 9, 209, 204, 264, 267, 266, 206, 249, 326, 8, 248, 319, 247, 203, 189, 295, 202, 200, 7, 201, 281, 280, 263, 279, 278, 199, 256, 198, 197, 196, 210, 211, 195, 325, 277, 193, 194, 6, 259, 258, 244, 243, 242, 56, 241, 215, 240, 175, 272, 271, 273, 270, 276, 239, 275, 260, 269, 274, 376, 385	555.68
2	378, 106, 108, 107, 361, 101, 109, 102, 110, 103, 104, 105, 93, 92, 91, 75, 81, 71, 373, 54, 53, 52, 72, 375, 378	608.02
3	385, 214, 187, 225, 10, 226, 227, 11, 192, 291, 268, 321, 299, 300, 301, 302, 293, 318, 294, 322, 305, 308, 307, 306, 99, 182, 173, 185, 184, 181, 172, 183, 186, 73, 33, 31, 32, 30, 332, 330, 331, 338, 179, 180, 177, 176, 178, 174, 369, 367, 368, 375, 385	638.66
4	381, 151, 22, 21, 13, 12, 15, 19, 28, 25, 26, 27, 24, 23, 154, 374, 381	578.64
5	378, 51, 49, 372, 57, 50, 17, 14, 16, 20, 18, 48, 47, 46, 41, 39, 42, 38, 29, 34, 35, 36, 37, 85, 86, 363, 374, 375, 378	1 013.06
6	388, 344, 351, 348, 345, 339, 353, 352, 354, 356, 355, 342, 343, 341, 340, 360, 388	947.17
7	385, 296, 208, 261, 213, 288, 289, 251, 245, 63, 66, 64, 65, 94, 95, 96, 40, 44, 43, 55, 58, 76, 77, 78, 84, 79, 80, 70, 69, 68, 67, 97, 98, 100, 328, 368, 385	538.94
8	386, 327, 329, 90, 358, 357, 88, 89, 87, 335, 337, 336, 334, 333, 369, 386	694.74
9	380, 121, 123, 131, 138, 152, 155, 156, 157, 153, 142, 141, 129, 130, 128, 127, 133, 134, 132, 126, 125, 124, 122, 112, 113, 115, 114, 111, 120, 117, 118, 119, 116, 374, 380	649.44
10	383, 171, 168, 169, 170, 165, 166, 167, 148, 147, 146, 144, 143, 145, 160, 161, 163, 162, 164, 374, 383	572.58
11	381, 139, 140, 136, 135, 137, 365, 362, 375, 381	**1 019.85**
12	387, 370, 346, 347, 349, 350, 375, 387	805.75
13	385, 45, 159, 158, 359, 372, 385	691.55
14	382, 150, 149, 365, 382	821.79
15	378, 364, 375, 378	562.15
	Total	**10 698.07**

TABLE C.8: *The sixteen vehicle routes proposed by the MACO algorithm in Solution 8 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	384, 5, 187, 214, 367, 274, 276, 270, 273, 271, 272, 178, 174, 369, 179, 338, 330, 329, 327, 368, 328, 333, 334, 335, 331, 176, 175, 258, 259, 6, 210, 211, 196, 197, 256, 279, 278, 199, 200, 7, 202, 247, 248, 8, 326, 249, 206, 264, 265, 255, 205, 290, 232, 2, 1, 233, 234, 231, 304, 229, 230, 10, 227, 226, 11, 223, 222, 235, 236, 4, 237, 217, 216, 204, 209, 9, 203, 189, 295, 266, 267, 201, 281, 280, 325, 193, 190, 277, 194, 371, 384	578.91
2	382, 160, 163, 162, 164, 167, 158, 159, 45, 359, 47, 372, 382	809.91
3	378, 12, 13, 15, 25, 26, 27, 135, 22, 23, 24, 269, 260, 275, 239, 191, 192, 262, 366, 198, 263, 195, 238, 220, 219, 218, 257, 3, 303, 212, 292, 207, 371, 284, 252, 375, 378	523.41
4	378, 308, 307, 299, 300, 301, 302, 293, 318, 294, 322, 305, 306, 182, 75, 73, 72, 52, 76, 77, 78, 79, 84, 71, 373, 81, 100, 99, 66, 64, 65, 245, 63, 82, 250, 251, 289, 288, 74, 286, 285, 283, 296, 208, 261, 213, 254, 253, 282, 287, 324, 320, 315, 319, 314, 268, 291, 242, 243, 244, 240, 215, 241, 56, 297, 298, 309, 310, 246, 313, 311, 312, 377, 317, 316, 323, 377, 378	487.69
5	388, 19, 17, 14, 16, 54, 58, 372, 388	601.33
6	381, 152, 155, 156, 157, 153, 141, 142, 124, 125, 126, 132, 134, 133, 127, 374, 130, 129, 148, 365, 165, 166, 161, 128, 121, 112, 113, 362, 122, 123, 131, 154, 151, 375, 381	**1 022.79**
7	387, 184, 93, 91, 90, 357, 358, 353, 352, 339, 344, 340, 370, 92, 94, 95, 96, 97, 98, 67, 68, 69, 70, 80, 360, 371, 387	593.46
8	378, 185, 173, 186, 183, 172, 181, 177, 180, 221, 106, 107, 108, 109, 101, 361, 102, 110, 103, 104, 105, 62, 61, 83, 59, 60, 371, 378	535.61
9	386, 31, 33, 32, 48, 46, 41, 39, 42, 38, 40, 55, 51, 372, 57, 50, 18, 20, 28, 337, 336, 30, 332, 375, 386	645.74
10	380, 115, 114, 111, 119, 120, 116, 118, 117, 147, 146, 144, 145, 143, 168, 21, 140, 374, 380	719.24
11	384, 224, 225, 228, 321, 53, 49, 29, 44, 43, 34, 35, 37, 36, 85, 86, 87, 89, 88, 354, 188, 377, 384	653.34
12	388, 351, 350, 349, 347, 346, 348, 345, 341, 343, 342, 355, 361, 388	968.60
13	387, 364, 363, 139, 138, 137, 136, 371, 387	822.75
14	380, 169, 170, 171, 365, 380	345.60
15	379, 356, 376, 379	362.89
16	382, 149, 150, 365, 382	845.12
	Total	**10 516.37**

Table C.9: *The thirteen vehicle routes proposed by the MACO algorithm in Solution 9 of Figure 13.1 for the Western Cape TVRPGC instance.*

Route	Customers	Autonomy
1	387, 174, 178, 180, 177, 176, 175, 272, 271, 273, 270, 274, 276, 239, 275, 260, 269, 367, 187, 214, 258, 259, 6, 191, 190, 193, 325, 277, 194, 5, 262, 192, 366, 210, 211, 196, 197, 195, 9, 209, 204, 216, 217, 238, 218, 237, 4, 236, 235, 222, 219, 220, 221, 224, 225, 11, 227, 226, 10, 223, 228, 229, 230, 304, 231, 234, 233, 1, 2, 232, 290, 205, 255, 265, 257, 376, 3, 212, 303, 207, 296, 208, 261, 371, 213, 292, 253, 254, 252, 283, 284, 286, 285, 282, 74, 251, 250, 61, 83, 60, 59, 82, 63, 245, 92, 65, 64, 66, 310, 246, 188, 377, 313, 311, 312, 317, 316, 287, 324, 320, 315, 319, 247, 248, 8, 326, 249, 206, 264, 267, 266, 203, 189, 295, 202, 201, 281, 7, 200, 199, 278, 279, 280, 263, 256, 198, 268, 314, 291, 242, 56, 241, 215, 240, 367, 387	670.92
2	384, 372, 57, 18, 20, 16, 17, 19, 28, 15, 14, 12, 13, 21, 22, 23, 26, 27, 25, 24, 337, 336, 335, 334, 333, 328, 368, 327, 329, 330, 331, 338, 179, 376, 384	659.32
3	380, 374, 130, 129, 148, 167, 166, 165, 365, 171, 170, 169, 168, 137, 138, 140, 139, 154, 151, 363, 152, 155, 156, 376, 380	**1 060.16**
4	388, 342, 339, 345, 351, 350, 349, 347, 346, 348, 344, 340, 360, 388	1 025.14
5	380, 113, 115, 114, 111, 119, 118, 116, 120, 117, 147, 146, 144, 143, 145, 160, 161, 163, 162, 164, 128, 127, 133, 134, 132, 126, 125, 124, 131, 141, 142, 153, 123, 122, 121, 374, 380	687.67
6	388, 99, 100, 98, 67, 68, 72, 73, 186, 183, 172, 181, 369, 184, 185, 173, 182, 75, 294, 322, 305, 308, 307, 306, 376, 388	617.74
7	378, 32, 50, 51, 48, 47, 46, 41, 39, 42, 38, 29, 44, 43, 40, 34, 35, 55, 58, 49, 54, 53, 52, 76, 77, 78, 84, 69, 70, 80, 79, 373, 71, 81, 375, 378	488.65
8	387, 244, 243, 360, 299, 300, 301, 302, 293, 318, 97, 96, 95, 94, 93, 105, 104, 356, 355, 289, 288, 323, 321, 309, 298, 297, 376, 387	629.07
9	379, 102, 110, 103, 91, 90, 353, 358, 357, 88, 89, 87, 86, 85, 62, 106, 108, 107, 109, 101, 360, 379	742.79
10	384, 361, 354, 343, 341, 370, 352, 37, 36, 376, 384	715.14
11	384, 33, 31, 30, 332, 136, 135, 157, 112, 362, 158, 159, 376, 384	848.09
12	383, 45, 359, 149, 364, 383	722.24
13	385, 364, 150, 376, 385	803.14
	Total	**9 670.06**