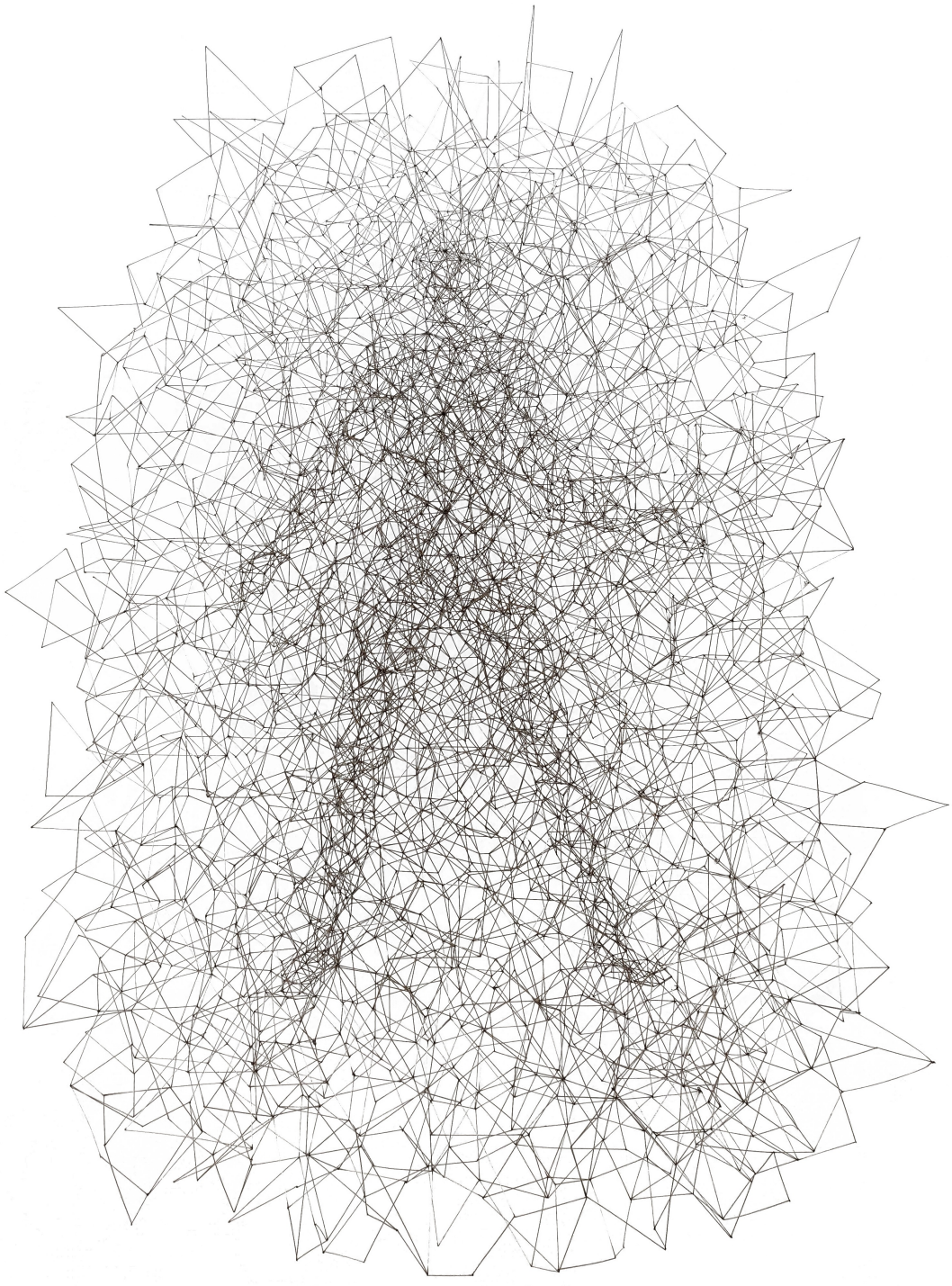
THE UNIVERSITY
of EDINBURGH

The Probabilistic Travelling Salesman

Off the Beaten Track



Antony Gormley, Mass (2006). Sculpture (302 x 206 x 102 cm) crafted from a network of stainless steel bars. Photo taken at Museum Voorlinden, Wassenaar (The Netherlands).

The
Probabilistic Travelling Salesman
Off the Beaten Track

The Impact of

Clustering, Construction, and Correlation

Pascal Leonardus Johannes Wissink

A thesis submitted in fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

to the

UNIVERSITY OF EDINBURGH

2019

THE UNIVERSITY *of* EDINBURGH

Board of Examiners

External examiner: Prof T.S. Chang

Internal examiner: Dr M. Tomasella

Supervion

Principal supervisor: Prof J. Ouenniche

Assistant supervisor: Prof T.W. Archibald

Edinburgh, United Kingdom

Email: Pascal.Wissink@ed.ac.uk

This dissertation was orally examined by a viva voce thesis defence on 23 November 2018 at the University of Edinburgh, College of Arts, Humanities and Social Sciences, Business School, 29 Buccleuch Place, Edinburgh EH8 9JS, UK. A catalogue record is available from the University of Edinburgh library. This thesis is submitted in partial fulfilment for the Doctor of Philosophy (PhD) degree in Management, specialisation in Management Science and Business Economics, at the University of Edinburgh.

Declaration

I declare that this thesis has been composed solely by myself and that it has not been submitted, either in whole or in part, in any previous application for a degree. Except where otherwise acknowledged, the work presented is entirely my own.

Signed,

(Pascal L.J. Wissink)

For Agnes

Acknowledgements

This PhD thesis is the result of a long journey, which I would not have been able to complete without the help of my family, friends, colleagues, and many others. First of all, I would like to thank my principal supervisor, Professor Jamal Ouenniche. Thank you for your guidance, patience, and numerous suggestions. Jamal's expertise and critical look did not only have a great impact on the quality of my dissertation, but also accelerated my academic development. I sincerely owe him a debt of gratitude for his support throughout my PhD journey. I would also like to thank my second supervisor, Professor Thomas Archibald. His comments and recommendations greatly improved the quality of my thesis. Thank you, Professor Tsung-Sheng Chang and Dr Maurizio Tomasella, for serving as my examiners.

I wish to thank the College of Arts, Humanities and Social Sciences for their generous financial support. Likewise, I owe a lot to the Business School staff. The academics of the Management Science & Business Economics group did not only point out exciting new research avenues to explore, but also gave me helpful comments and suggestions. The administrative staff of the Business School, and in particular, the staff of the Postgraduate Research Office were of great help throughout my PhD. A special mention goes to Susan Keatinge, as I am greatly indebted for her support during the organisation of the World Class Workshops on Vehicle Routing, among many other things. This thesis also benefits from the numerous conferences and workshops the Business School enabled me to attend. I would like to thank the participants of the JOPT, IFORS, INFORMS, and ICSP conferences for their valuable feedback.

I would not have been able to complete my PhD without the friends who supported me throughout my long journey. I especially owe a lot to my friends in Edinburgh, from the Business School and beyond, the majority of whom even after five years of discussion still insist that my PhD research is about traffic lights. (Although Edinburgh's lights are, by all means, a suboptimal nuisance.) In alphabetical order, I wish to thank Skarleth Carrales, George Ferns, Denis Fryrych, Marian Gatzweiler, Mona Hamid, Maeve Howe, Dora Jandrić, Jakov Jandrić, Stephen Kay, Mahmoud Khalik, Benjamin Koeck, Christian König, Prasanna Kumar, Aliette Lambert, Mareike Lührs, Marcel Lukas, Ben Marder, the guy who looks like Freddie Mercury, Rachel Moss, Amin Nazifi, Alexandra Rome, Matteo Ronzani, Ben Sila, Siobhan Williams, Alessa Witt, and probably a lot of other people whose names I forgot to include in this list. I owe you a lot for your support, inspiration, and unforgettable times in Edinburgh. I also owe a lot my friends in the Netherlands, who continued to support me even in the most difficult times while I was away. A very special and big thanks to Maarten Jansen, Yuchen Li, Frederik Mannaerts, Jacqueline Schijven, and Dennis van de Werken. Two of my friends deserve a special mention for not only giving their friendship, but also valuable input on my PhD. Betty Johanna Garzon Rozo and Bart Keijsers, thanks a million!

Most of all, thank you to my family for their love and support. Agnes, Jan, Ivo and Ilse, you are the best! I could not have done this without you. Mauritia, I owe you a lot for your love and encouragement on the home stretch of my PhD.

Finally, I wish to extend my gratitude to Wim Struik for inspiring me to study mathematics.

This thesis is dedicated to my mum, Agnes, who I greatly admire for her positive spirit and encouragement, and for the unconditional love and support she gave me throughout my PhD journey.

Abstract

Same-day parcel deliveries, grocery order collection in warehouses and food delivery by bicycles are but a few recent examples that illustrate the increased complexity of the routing problems our society is facing nowadays. The increasingly complex nature of routing problems is in part caused by a tremendous growth of stochastic involvement. Travel times between customers are usually uncertain, their demands may change, and their presence at the time of delivery is often unknown. Accounting for such uncertainties plays an important role in the design of efficient routes. The body of literature dedicated to the performance of solution methods in such stochastic settings is therefore growing rapidly. However, only few studies approach the uncertainty retained by routing problems from the opposite perspective.

By revisiting one of the earliest accounts in the field of stochastic routing, the probabilistic travelling salesman problem, this thesis investigates the impact of uncertainty in customer presence 'off the beaten track': How the problem characteristics of a fundamental stochastic routing problem poses challenges to the empirical performance of solution methods. The probabilistic travelling salesman problem is concerned with finding the shortest route along a set of customers in a graph, while their presence is stochastic. That is, each customer in the graph only requires a visit with a certain probability. Therefore, the subset of customers that eventually requires a visit is unknown. Since it is well-known that the optimal solution of this problem may differ from its deterministic counterpart, the travelling salesman problem, many problem-specific solution methods have been developed

to cope with the stochastic nature of the probabilistic travelling salesman problem. But only little empirical research has focused on the opposite perspective: How problem characteristics affect the performance of solution methods.

Starting from scratch, a first study reviews and compares solution methods for the probabilistic travelling salesman problem. It provides a comprehensive overview of the most important probabilistic methodological adaptations of exact methods, construction heuristics, improvement heuristics and metaheuristics, and discusses their strategies, key features and performance. An in-depth analysis evaluates the construction phase of a selection of several solution methods under different circumstances. The application of sufficiently explorative heuristics in the improvement phase of deterministic routing problems has led to the commonly held belief that advantages accrued by construction heuristics vanish in the final solution. Consequently, the construction phase also remained largely unexplored in the probabilistic travelling salesman problem. I put the hypothesis to the test that initial solutions found by construction heuristics contribute to the quality of final solutions for the probabilistic travelling salesman problem. Empirical research reveals that the spatial configuration of customers in the problem is an essential determinant in the extent of the contribution. Specifically, the presence of clustering of customers across the plane is one of the main drivers of the relative performance of construction heuristics.

Building on this insight, a follow-up study proposes a new hyperheuristic framework that generates new construction heuristics tailored to any given problem setting. This hyperheuristic framework, named GENS-H, is the first reported application of hyperheuristics in the stochastic routing domain that explicitly takes problem characteristics into account. GENS-H relies on the Generic Savings procedure, a solution approach that unifies a class of construction heuristics, namely savings-based procedures, into a single generalised and parameterised framework. GENS embeds existing savings procedures as special instances, but is also capable of producing numerous new procedures on demand by plugging

in different parameter configurations. Notwithstanding its potential, the goal of an included GENS-H benchmark study is not predominantly to outperform state-of-the-art methods, but rather to demonstrate the added value of pursuing an approach where problem characteristics — and specifically, the clustering of customers — are taken into consideration. Extensive numerical tests support this premise.

The notion that spatial proximity in the form of customer clustering plays a quintessential role in the characterisation of a problem promotes the idea that dependency through other, potentially also nonspatial, interactions between customers could be an overlooked trait in models for stochastic routing. That is, the assumption made by the probabilistic travelling salesman problem that customers are independent can be regarded as an oversimplification. Drawing on concepts in computational sociology and financial mathematics, the last study of this thesis proposes a generalised version of the probabilistic travelling salesman problem that incorporates a hands-on way to model dependencies between customers who are seemingly related. This new stochastic combinatorial optimisation problem, the correlated probabilistic travelling salesman problem, ventures into an unexplored territory of stochastic routing where the probabilistic requirements to interact with customers are not isolated endeavours any more. More specifically, the correlated probabilistic travelling salesman problem integrates a heterogeneous set-up that allows one to specify correlations between the stochastic requirements of visiting a group or cluster of customers, with the ability to reduce to the original probabilistic travelling salesman problem in case dependence is absent. This implementation has many practical applications in solving real-world routing problems where, for example, similar behaviour in customer presence can be observed among different customer segments. This study also demonstrates that good solutions for the deterministic version of the travelling salesman problem and the probabilistic travelling salesman problem do not necessarily coincide with good solutions for the correlated adaptation. As such, the development of new, customised, solution methods for the correlated probabilistic travelling salesman is needed.

Lay summary

The planners of food delivery services on bicycles, same-day parcel deliveries by web shops, and grocery order collection in warehouses all face a similar challenge: What is the shortest possible route along all pickup and delivery points such that each point is visited exactly once? These so-called routing problems are ever more often surrounded by uncertainty. For example, there is a chance that a customer decides to cancel a delivery just before it ships, or that a driver experiences a delay en route and needs to skip a group of pickup points. The mathematical problem central to this thesis, where one is concerned with finding the shortest route along a group of points, but where each point only has a certain probability of actually requiring a visit – and thus may be skipped when the visit requirements become apparent – is known as the probabilistic travelling salesman problem.

Because designing routes along a set of points manually becomes a tedious task when the number of points becomes large, most studies about the probabilistic travelling salesman problem focus on finding new solution methods that are able to construct routes in increasingly optimal (i.e., even shorter) and increasingly efficient (i.e., even faster) ways. To investigate whether a new solution method achieves this goal, researchers typically resort to testing their methods on a set of standard routing problems under fixed settings. Under identical conditions, the best route found by a new solution method can be compared directly to the best routes found by existing methods. It should then be easy to spot whether a new method manages to outperform existing methods based on criteria for the total expected distance that needs to be travelled to complete a route, or

the computation time taken by a solution method to find a route. However, by only looking at performance criteria under identical settings, researchers tend to overlook the determining characteristics concealed by a problem that could be even bigger influencers of the performance of their solution methods. To understand why certain solution methods perform better than others, it is therefore crucial to gain a deeper understanding of what characteristics really typify a problem. In a series of three separate studies, this thesis investigates the probabilistic travelling salesman problem 'off the beaten track' by turning the perspective around: How do the characteristics of a problem affect the performance of solution methods?

The first study of this thesis reviews and compares different solution methods for the probabilistic travelling salesman problem. It attempts to classify the solution methods into a number of categories, and identify their key features, strategies, and performance. A more detailed comparative analysis included in the study addresses the impact of problem characteristics on the construction phase of solution methods for the probabilistic travelling salesman problem. Solution methods for routing problems typically advance in two phases: A construction phase and an improvement phase. In the construction phase, the method constructs a first, complete route along the points. The improvement phase that follows then improves this route. The construction phase of routing problems under uncertainty has not received a lot of attention due to the commonly held belief that – for routing problems without uncertainty – advantages accrued in the initial route ultimately vanish after completing the improvement phase. I put this hypothesis to the test for the probabilistic travelling salesman problem in a comparative analysis of several solution methods. The results reveal that, from a selection of problem characteristics, the spatial configuration of the points is an essential determinant in the contribution of the construction phase to the length of the final route. Specifically, the way points are clustered together in some problems, while points are more evenly divided in other problems, is one of the main drivers of the performance of the considered solution methods.

The insight that the characteristics of a problem play an important role in the performance of solution methods for the probabilistic travelling salesman problem calls for a problem-specific approach to construct routes. A follow-up study proposes a new framework, called GENS-H, that caters for this demand. GENS-H searches over all the defining features of a generalised class of solution methods, and picks those components of the solution methods that provide the best match for a given problem setting. In doing so, GENS-H is the first reported application in the routing domain under uncertainty where a tailor-made solution method is automatically generated for a given set of problem characteristics. Notwithstanding its potential, the goal of a subsequent experimental comparison of GENS-H versus other solution methods is not predominantly to find even better routes, but rather to demonstrate the added value of pursuing an approach where problem characteristics – and specifically, the clustering of points – are taken into consideration. Extensive numerical tests support this premise.

The notion that the clustering (or spatial proximity) of points takes a prominent role in the characterisation of routing problems under uncertainty promotes the idea that an independent treatment of each point is perhaps an oversimplification of real-world situations. In reality, points can be related, because they share similar characteristics that affect their mutual uncertainty surrounding a visit. For example, the likelihood of a student being at home in a neighbourhood with a high student population could share resemblance with the likelihood of neighbouring students being at home around the same time. A delivery company may wish to account for such similarities in its determination of a good tour through this neighbourhood.

Unfortunately, the default assumption made in the probabilistic travelling salesman problem is that all points that need to be visited are completely unrelated. In fact, the entire calculation of the expected length of a route under uncertainty relies for a great deal on the assumption that the need to visit one point is independent of the need to visit another point. Drawing from concepts in computational sociology and financial mathematics, the last study of this thesis

therefore proposes a generalised version of the probabilistic travelling salesman problem that incorporates a hands-on way to model dependencies between points that are seemingly related. This new problem, named the correlated probabilistic travelling salesman problem, ventures into an unexplored territory of routing problems where the uncertain requirement to interact with a point is not an isolated endeavour any more. By specifying the degree of relatedness between any group of points on a simple scale, an implementer of the problem is able to calculate the expected length of a route while explicitly accounting for potential dependencies in the requirements to pay each point a visit. The study also demonstrates that a good solution for the probabilistic travelling salesman problem – without the ability to model relations between points – does not necessarily coincide with a good solution of the version of the problem in which points are related. As such, the development of new, customised, solution methods for the correlated probabilistic travelling salesman is needed.

Contents

List of Tables

List of Figures

1 Introduction

With the surge of e-commerce, parcel delivery companies are seeing an incredible growth in delivery volumes. The daily average parcel volume of the world's leading parcel delivery company, United Parcel Service (UPS), increased by 55% since the dot-com bubble up until the end of 2017 (Figure 1.1), with total global volumes expected to double in the next decade (Joerss et al., 2016). The rise of e-commerce sparked a trend of declining bulk deliveries to retailers, and growing single item deliveries to individual consumers. This has led to a tremendous growth in the number of delivery points (Wright, 2014). Simultaneously, same-day and instant deliveries are projected to grow to 25% of the total share of deliveries by 2025 (Joerss et al., 2016). The parcel delivery business is thus not only booming, but also getting increasingly complex in nature. In particular, uncertainty takes an ever more prominent role in parcel delivery services, as stochastic demands, travel times, deadlines, customer presence, and order cancellations become more important factors to account for. The combination of a rapidly increasing number of delivery points, and simultaneous increase in the stochastic nature of deliveries, made companies rethink their routing strategies (Wright, 2014). Specifically, as Henry Maier, CEO of FedEx Ground, put it in an interview with the Financial Times: "*The challenge across the industry is managing the stops*" (Wright, 2014).

In the light of these developments, it seems natural to revisit the old routing problems that lie at the heart of parcel delivery challenges from new perspectives. The routing problem central to this thesis, that directly addresses the combined challenge of finding the shortest route along a number of delivery points,

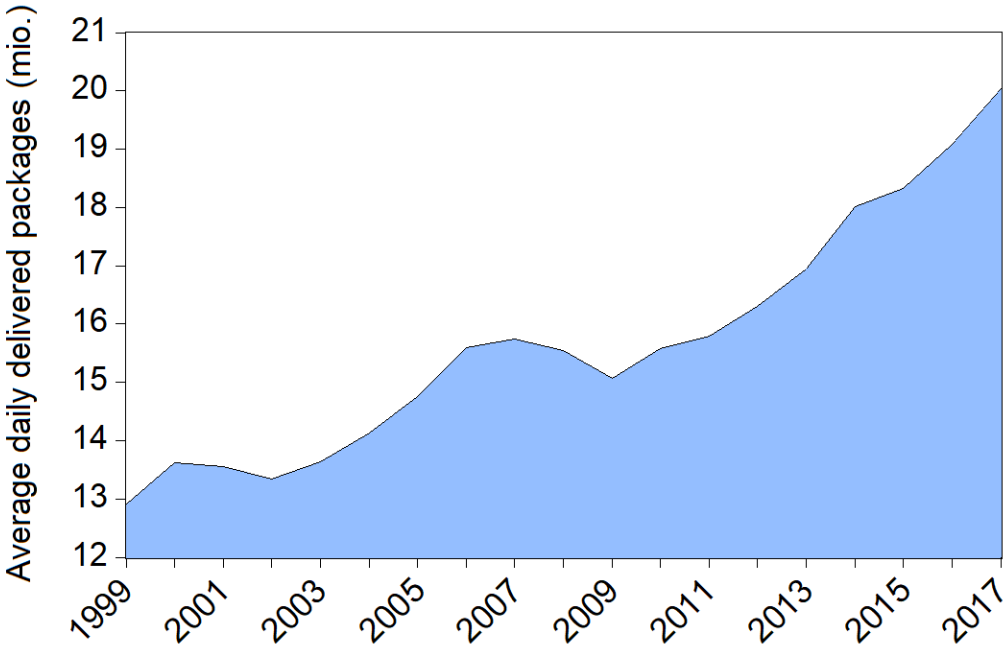


Figure 1.1: The daily global average delivery volume of UPS in millions in the period 1999-2017. The figures used to produce this graph were sourced from UPS Annual Reports, http://investors.ups.com/financials/annual-reports. [Accessed: 10 July 2018.]

while accounting for the uncertainty surrounding their visit requirements, is the probabilistic travelling salesman problem (PTSP). The PTSP was proposed by Jaillet (1985) as a way to introduce stochastic customer presence in the existing deterministic travelling salesman problem (TSP). It is therefore also sometimes referred to as the TSP with stochastic customers (TSPSC). Ever since its introduction in 1985, researchers have attempted to find new methods and improve existing methods to solve the PTSP. Although there is a vast body of literature dedicated to the performance of solution methods for a given stochastic customer setting (e.g., Laporte et al., 1994; Bianchi et al., 2002b; Marinakis and Marinaki, 2010), only few studies pay attention to how the problem-specific setting impacts the performance of solution methods.

In a series of studies, this thesis investigates the impact of uncertainty in customer presence 'off the beaten track': Namely, it investigates how characteristics of a fundamental stochastic routing problem, the PTSP, challenge the empirical performance of solution methods. It should be stressed that the applications

proposed in this thesis are not limited to the PTSP, but can be generalised to develop approaches for other stochastic routing problems, and in particular, the vehicle routing problem with stochastic customers (Jaillet, 1985, 1987; Jaillet and Odoni, 1988). The studies are drafted up in separate chapters, unified by a common problem statement. The three studies are preceded by a literature survey. The literature survey targets problem-specific solution methods for a the entire stochastic routing domain, in order to position solution methods for the PTSP in a broader setting and reflect on potential similarities and differences. In contrast to the consecutive chapters that comprise the subsequent chapters, the literature survey does not specifically address the performance of solution methods.

A first study, "*A comparative analysis of solution methods for the PTSP*", reviews and compares the solution methods designed for the PTSP. It considers a wide range of solution classes: exact methods (e.g., the integer L-shaped method by Laporte et al., 1994), construction heuristics (e.g., the spacefilling curves of Bartholdi and Platzman, 1982), classical improvement heuristics (e.g., probabilistic 2-opt by Bianchi et al., 2005), and metaheuristics (such as the hybrid multi-swarm particle swarm optimisation by Marinakis and Marinaki, 2010). For each solution class, the study covers the full range of corresponding solution methods. It discusses their mechanisms, key characteristics, limitations, and performance. The sections pertaining to the different solution classes are concluded with a comparative analysis of the discussed methods. As such, it provides a critical comparative analysis of the PTSP-specific methods discussed in the preceding literature review.

The study specifically zooms in on the construction phase for a selection of solution methods, with an empirical evaluation under different circumstances. The application of sufficiently explorative heuristics in the improvement phase of deterministic routing problems has led to the commonly held belief that advantages accrued by construction heuristics vanish in the final solution. Consequently, the construction phase also remained largely unexplored in the PTSP. Empirical research reveals that in a number of cases, the spatial configuration of customers

in the problem is an essential determinant of the performance of construction heuristics. Specifically, the presence of clustering or sparsity of customers across the plane is one of the main drivers of the relative performance. This study also puts the hypothesis to the test that initial solutions found by construction heuristics contribute to the quality of final solutions for the probabilistic travelling salesman problem. Empirical research with a state-of-the-art improvement heuristic, 2.5-opt-EEais (Birattari et al., 2008), suggests that this is indeed the case. Moreover, the findings on clustering or sparsity remain valid, even after improvement.

The second study of this thesis, "*A hyper-heuristic driven construction framework*", proposes a new hyper-heuristic framework that generates tailor-made heuristics for any given problem setting. This hyperheuristic framework, called GENS-H, unifies a class of construction heuristics into a single generalised and parameterised framework. Specifically, GENS-H builds on the insight from the first study that the performance differences among solution methods rely to a large extent on the differences in problem configurations. The hyperheuristic framework of GENS-H is meant to produce good and consistent results regardless of the spatial configuration. Although hyperheuristics (Burke et al., 2013) have been around since 2000 (Cowling et al., 2000), to the best of my knowledge, hyperheuristics have not been applied to stochastic routing problems before. This study is therefore the first reported application of hyper-heuristics in the stochastic routing domain that explicitly takes the problem characteristics into account.

GENS-H is inspired by the graph-based hyper-heuristic (Qu and Burke, 2009) for timetabling problems. The proposed adaptation of the graph-based hyper-heuristic in this study is tested on GENS, the Generic Savings procedure. The GENS procedure generalises a class of construction heuristics for the PTSP, viz. probabilistic adaptations of the well-known Clarke and Wright (1964) savings procedure for the TSP. The GENS procedure does not only embed the existing savings-based procedures as special instances, but is also capable of producing a wide range of alternative construction heuristics. Alternative instances of the

savings-based procedure may be obtained by plugging different combinations of parameter values into the GENS procedure. In a nutshell, GENS-H embeds the GENS procedure as a class of low-level heuristics, and subsequently uses a set of metaheuristics as high-level heuristics to tune its parameters. Whereas the solution space of the low-level heuristic is confined to the PTSP instance that is to be solved, the solution space of the high-level heuristics targets the parameters pertaining to the flexible components of the generalised low-level heuristic, i.e., GENS.

Notwithstanding its potential, the goal of the GENS-H benchmark study included in the second study is not predominantly to outperform state-of-the-art methods, but rather to demonstrate the added value of pursuing an approach where problem characteristics — and specifically, the clustering of customers — are taken into consideration. Moreover, the empirical evaluation reveals that GENS-H is able to outperform the individual low-level heuristics it is based upon in almost every instance.

The most attractive feature of GENS-H is its ability to unveil new heuristics. In many cases, manually selecting the best possible combination of heuristical building blocks to cope with specific problem features of the PTSP can be difficult, and guidelines are missing. GENS-H is capable of automatically selecting the most desirable combination of heuristical building blocks in order to cope with the specific characteristics of the problem instance that needs to be solved. The selection of particular building blocks in specific scenarios gives useful insights about which combination of building blocks give the best and most consistent results. It therefore provides a tool for the formulation of developers guidelines for new algorithms.

The notion that spatial proximity in the form of customer clustering plays an essential role in the characterisation of a problem promotes the idea that dependency through other, potentially also nonspatial, interactions between customers could be an overlooked trait in models for stochastic routing. In

reality, points can be related because they share similar characteristics that affect their mutual uncertainty surrounding a potential visit. For example, the need to pick up a specific item of a customer order at one point may depend on the uncertainty surrounding the availability of a related item at another pick-up point. Or the chance that the salesman decides to skip a delivery point in a remote area if he faces a delay, could depend on the chance that he also needs to pay several neighbouring (i.e., geographically related) points a visit. In fact, modelling dependencies can have many practical benefits when solving real-world routing problems where in-group favouritism (i.e., biasing customers in the same group), target marketing (i.e., accounting for behavioural similarities among customers), or chained deliveries (i.e., accounting for the joint visit requirements of related customers) play a role.

The assumption made by the PTSP that customers are independent can therefore be regarded as an oversimplification. Because the calculation of the expected length of a PTSP tour relies on the assumption that the need to visit one point is independent of the need to visit another point, the objective function is in need of an upgrade in order to accommodate potential dependencies.

Drawing from concepts in computational sociology and financial mathematics, the last study of this thesis, "*The correlated PTSP*", proposes a generalised version of the PTSP that incorporates a hands-on way to model dependencies between customers who are seemingly related. This new stochastic combinatorial optimisation problem, termed the correlated probabilistic travelling salesman problem (CPTSP), ventures into an unexplored territory of stochastic routing where the probabilistic requirements to interact with customers are not isolated endeavours any more. By integrating the Bahadur (1961) model and discrete vine pair copula constructions (Panagiotelis et al., 2012), the CPTSP adopts a powerful and flexible implementation. More specifically, the heterogeneous CPTSP only requires the specification of the marginal probabilities describing the stochastic visit requirement of each customer, and a correlation matrix describing the pairwise

dependence between groups or individual pairs of customers. In addition, the CPTSP bears the ability to reduce to the original PTSP in case dependence is absent.

Underscoring its practical relevance, this study also demonstrates that good solutions for the deterministic version of the TSP and the PTSP do not necessarily coincide with good solutions for the correlated adaptation. In fact, an extensive subsequent empirical analysis on both homogeneous and heterogeneous reveals that dependence can have a significant impact on the value of the objective function. Consequently, the relative performance of solution methods depends on the degree of the customer dependence. Therefore, the development of new, customised, solution methods for the correlated probabilistic travelling salesman is desired. In order to cater for such demand, the study concludes with a correlated probabilistic ant colony system, an adaptation of the probabilistic ant colony system (Bianchi et al., 2002b) that is currently regarded as one of the best-performing state-of-the-art metaheuristics for the PTSP.

The remainder of this thesis is organised as follows. Chapter 2 discusses the literature relevant for the stochastic routing domain. Chapters 4–6 discuss the studies treated in this introduction in the same order as they appear above. Chapter 7 concludes with a summary of the contributions, limitations, and suggestions for further research.

2 | A survey of solution methods for stochastic routing problems

Stochastic routing problems are essentially the same as any other class of routing problems, but instead of having predefined fixed components, some of the components of stochastic routing problems are only known with some probability. That is, some components are *stochastic* rather than *deterministic*. Dealing with this uncertainty often leads to a totally different outcome. Although in some cases we can simply modify the existing solution approaches that deal with the deterministic variant of the problem, in many cases, deterministic solution methods do not perform nearly as well as specially designed solution approaches.

This chapter surveys the most important problem-specific solution methods that have been designed for stochastic routing problems. More specifically, it attempts to combine and update the relevant parts of the surveys by Gendreau et al. (1996b), Cordeau et al. (2007), Bianchi et al. (2009), Pillac et al. (2013), Henchiri et al. (2014), and Gendreau et al. (2014) on stochastic routing. It classifies existing literature on stochastic routing problems into two main categories, namely the stochastic travelling salesman problem (STSP) and the stochastic vehicle routing problem (SVRP).

Stochastic routing is a relatively young area of research, and can be traced back to the introduction of the vehicle routing problem with stochastic demands (VRPSD) in 1969 (Tillman, 1969). The field was developed as a straightforward

extension from deterministic routing problems, most notably, the TSP (Dantzig et al., 1954) and the vehicle routing problem (Dantzig and Ramser, 1959), aided by the simultaneous development of stochastic programming (Dantzig, 1955) and stochastic-dynamic programming (Bellman, 1957). Stochastic routing problems are among the most studied topics of the overarching mathematical optimisation branch known as stochastic mixed integer programming (SMIP). The most common stochastic components encountered in stochastic routing literature are stochastic demands (Tillman, 1969), stochastic travel or service times (Leipälä, 1978; Kao, 1978), and stochastic customer presences (Jaillet, 1985). As routing problems are getting increasingly complex in nature (see chapter 1), the study of stochastic components has also gained in popularity – see Figure 2.1. The next section provides an in-depth discussion of the general structure of routing problems and these components.

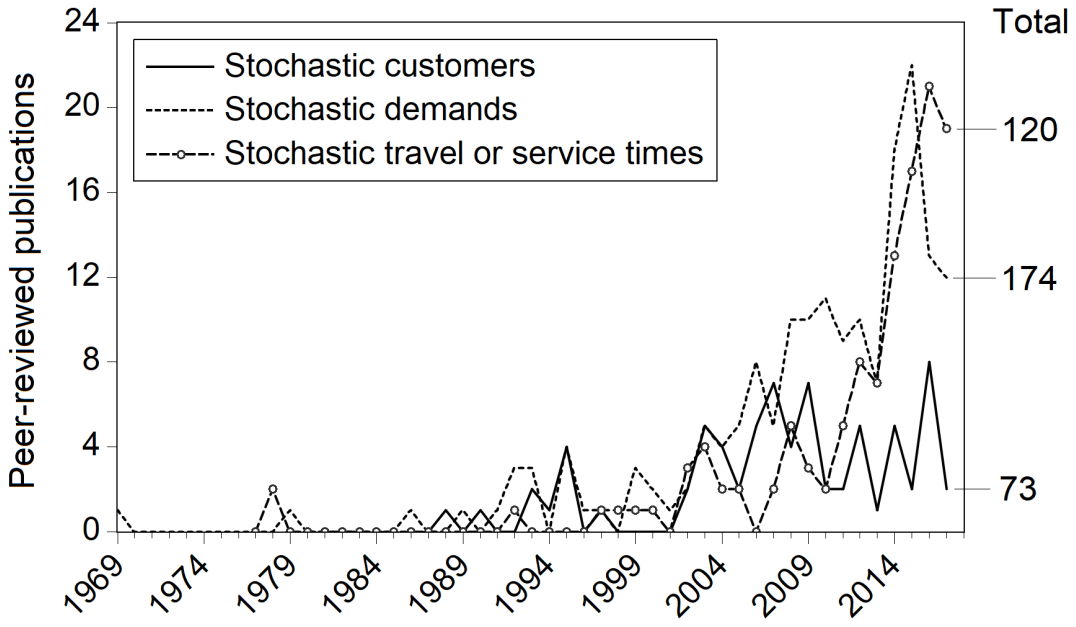


Figure 2.1: Number of peer-reviewed publications on stochastic routing by stochastic component from 1969–2018 (50 years). The lists with publications used to produce these graphs were sourced from Web of Science, Clarivate Analytics, https://www.webofknowledge.com. [Accessed: 9 November 2018.]

The subsequent sections of this chapter treat the literature pertaining to the distinct stochastic components. However, note that it is not uncommon to combine

multiple stochastic components in a single stochastic routing problem. Notable examples include the vehicle routing problem (VRP) with stochastic customers and demands (Bertsimas, 1992), the courier delivery problem with stochastic customers and service times (Sungur et al., 2010), and the VRP with stochastic travel times and demands (Lee et al., 2012). In some cases, such combinations are the result of the observation that certain stochastic components can be generalised to another. For example, the VRP with stochastic customers (VRPSC) may be viewed as a VRP with unitary stochastic demands. In case a customer has no demand he is skipped, akin the VRPSC (Jaillet, 1985). A more elaborate description of the mechanisms integrated by each problem is given in the remaining sections of this chapter.

Due to the similarities in their SMIP formulations and stochastic component linkages, it is perhaps not surprising that different stochastic routing problems share very similar or even the same solution methods. For example, the integer L-shaped method (Laporte and Louveaux, 1993) is shared by many stochastic routing problems as the first solution method to solve small instances to optimality. Hence, it is crucial to gain a complete overview of the most important developments of solution methods in the entire stochastic routing domain, in order to grasp how contributions to specific problems relate to other developments in the field. Specifically, by surveying the literature on the most important solution methods in the entire stochastic routing domain, this chapter attempts to clarify how my contributions to the PTSP – the subject of the remaining chapters in this thesis – relate to the advancement of the greater stochastic routing domain.

Solution methods for routing problems are traditionally divided in two major categories: Exact methods and heuristic methods. Whereas exact methods solve a problem to optimality, heuristic methods, or simply heuristics, do not guarantee to solve problems to a global optimum. Instead, this latter category aims to strike a balance between solution quality and computational efficiency. Since many of the problems in the routing domain are NP complete, the computational time is often rapidly increasing as the size of a problem grows. Because the problems

in the stochastic routing subdomain add even more complexity to this already challenging field, most of the proposed solution-specific methods belong to the heuristic category.

Heuristics can be further divided into classical heuristics and metaheuristics. Classical heuristics are solution methods that have been developed specially for the specific problem that needs to be solved. As a result, classical heuristics are often limited-purpose solution methods; That is, these methods deal with only a very specific set-up of the problem and cannot always be easily adapted to another, related problem. Metaheuristics, on the other hand, are designed as general solution frameworks, which can be readily adjusted to deal with any problem of interest accordingly. This make metaheuristics a popular class of solution methods to study when coping with stochastic routing problems. Metaheuristics and classical heuristics can also be used in conjunction with each other. This chapter attempts to categorise the solution methods for every stochastic routing problem into one of the exact method, classical heuristic, or metaheuristic classes explained above.

The remainder of this chapter is structured as follows. First, section 2.1 attempts to give a general introduction to routing problems as a way to familiarise the reader with the various concepts in stochastic routing. The STSP, consisting of the two principal subcategories where either customers are stochastic (section 2.2.1) or travel times are stochastic (section 2.2.2), is discussed in paragraph 2.2. The other main category, the SVRP, is discussed in section 2.3. This category is also further divided into a number of subcategories that typify their most frequently encountered stochastic components. These include the vehicle routing problem with stochastic demands (section 2.3.1) and the vehicle routing problem with stochastic customers (section 2.3.2). Section 2.4 concludes.

2.1 The general structure of stochastic routing problems

The general structure of a routing problem is as follows. Consider one or more depots, and one or more vehicles leaving from the depot. In some cases, however, the depot may also be absent, in which case a random point of departure may be chosen. Every vehicle has to complete a tour along a set of the customers (nodes, cities), each of which may or may not require a visit, service and the delivery of goods (often referred to as demand). Accordingly, there may be costs associated with (i) the travelling from one customer (or depot) to another and (ii) servicing a customer. In many cases, the travel distance and the costs of traversing the path from one customer to another are assumed to be proportional. Additionally, extra costs, often expressed in terms of penalties, may be incurred whenever (iii) a customer is not serviced in time (Laporte et al., 1992); (iv) the demand of a customer exceeds the capacity of the vehicle (Laporte et al., 1989); and (v) the travel time required to visit all customers in a given tour exceeds the maximum allowed time (Taş et al., 2014).

Many variations of these features are possible; the features given above merely serve to set out an example of the most common variations found in literature. For example, in many cases, most of the previously described penalty cases (iii–v) are hard constraints rather than feasible (yet unfavorable) scenarios. This means that every vehicle must adhere to these conditions in order to obtain a possible solution. Other constraints that are frequently encountered in literature include (vi) vehicles being prohibited from taking subtours, i.e., a partial tour that returns to the depot before visiting every city; and (vii) every customer must be visited once and only once in any given instance of the solution.

This general set-up of a deterministic routing problem can be converted into a stochastic routing problem by transforming one or more features (i–vii) to a probabilistic setting. Common stochastic features include (i*) stochastic customers:

the presence of each customer is given by some probability; (ii*) stochastic demands: the demand of each customer is uncertain; (iii*) stochastic travel time: the time or costs required to go from one customer to another is unknown; or (iv*) stochastic service times: the time it takes to service a customer is unknown. It is also possible to combine one or more of these stochastic features with one another. Moreover, variations on these features are also possible. This leads to a large number of alternative problem descriptions by combining any of the stochastic variantions (i* – iv*) with a variation on the set-up of the problem (i – vii). The objective function consequently varies from one problem to another, and the costs may also vary as a function of the actual outcome of the stochastic components of the problem.

There are traditionally two ways to formulate a stochastic routing problem: As a chance constraint programming (CCP) formulation, or as a stochastic programming with recourse (SPR) formulation. In the CCP formulation, the constraints that incorporate stochastic components are rewritten in a way such that the probability of exceeding that constraint is bounded by a certain threshold. The CCP formulation does not invalidate the objective function if the constraint is nevertheless violated, which means that a certain percentage of the solutions may actually not be feasible.

In the general SPR formulation, a first-stage solution known as an *a priori* tour is designed that finds the optimal tour along all customers. More specifically, the first stage solution minimises the expected value of the length of the tour by integrating the stochastic components into the objective function. Then, in the second stage of the solution, the outcome of the stochastic variables is revealed and a recourse policy is exercised to deal with this outcome. That is, given that all customers are visited in the same order as they appear in the a priori tour, the recourse policy sets a specific instruction to deal with the deviations from the full deterministic setting. That means, skipping the absent customers in a tour with stochastic customers,

or performing a trip back to the depot once the vehicle capacity has nearly been reached in a problem with stochastic demands.

The CCP formulation is generally found easier to solve than the SPR formulation of a problem, but does not allow for the same degree of flexibility or realism as the SPR formulation. Therefore, most researchers prefer the SPR formulation over the CCP formulation.

A number of alternative problem formulations have also recently gained some attention. These include describing the problem as a Markov process (M) (e.g., Secomandi and Margot, 2009) and rewriting the problem as a dynamic programming formulation (DP) (e.g., Errico et al., 2016). In the latter case, the problem is described as a recursive formulation rather than a 'fully specified' formulation.

2.2 The stochastic traveling salesman problem

In the TSP, a salesman has to visit a set of customers that are connected to each other through a set of arcs that indicate possible ways to travel. Each arc comes with an associated cost of traversing it. Furthermore, each arc can be traversed only once, and each customer also requires exactly one visit. The objective is to find a route along all customers that minimises the total costs of travel. Although there are many different versions of the TSP, almost all theoretical-oriented papers assume the absence of a depot. The two most common stochastic variants of the TSP integrate stochastic customers or stochastic travel times in their set-up.

2.2.1 The TSP with stochastic customers

The TSPSC is a variant on the TSP where each customer requires a visit only with some known probability. As such, only a subset of customers may be present on any given instance of the problem. TSPSCs are divided into two categories;

namely, homogeneous TSPSCs and heterogeneous TSPSCs. Whereas homogeneous TSPSCs assume that all customers have identical probabilities of being realised, heterogeneous TSPSCs relax this assumption. The SPR formulation of the TSPSC is also more commonly referred to as the PTSP (see chapter 3).

The TSPSC was first introduced by Jaillet (1985). Extensions of the TSPSC to cases with deadlines (Campbell and Thomas, 2008b), pick-up and deliveries (Beraldi et al., 2005; Ho and Haugland, 2011), clustered set-ups (Tang and Miller-Hooks, 2007), time windows (Voccia et al., 2013) and profits (Zhang et al., 2018) were subsequently also proposed. Ever since 1985, researchers have attempted to develop both exact as well as approximate (i.e., heuristic) methods to solve the TSPSC and its variations. A first attempt to solve the problem using traditional, deterministic solution approaches was made by Jaillet (1985) himself. He uses an enumerative exact algorithm to solve the SPR formulation of the TSPSC to optimality. Jaillet (1985) also proposes a number of heuristics, but does not actually make an attempt to solve the TSPSC using any of these methods.

Another exact solution approach for solving stochastic combinatorial optimisation problems was introduced by Laporte and Louveaux (1993). They propose an exact solution method known as the integer L-shaped algorithm. The framework is subsequently extended and applied to the TSPSC by Laporte et al. (1994). Rosenow (1999), Mahfoudh et al. (2015) and Amar et al. (2017) propose exact branch and bound algorithms for the TSPSC, and demonstrate their practical relevance for small problem instances.

Most of the other solution methods proposed in literature for the TSPSC are heuristics. In a chronological order of events, Jézéquel (1985) first extends the theoretical results from Jaillet (1985) to an empirical setting and applies heuristical solution methods based on the Nearest Neighbor principle and the Clarke and Wright (1964) savings algorithm. Another attempt using similar heuristics is made by Rossi and Gavioli (1987). Bertsimas (1988) and Bertsimas and Howell (1993)

continue in the line of research by deriving additional properties and results for the TSPSC, and subsequently propose heuristics that embed space filling curves to solve the TSPSC. In addition, Bertsimas (1988) applies a local probabilistic 2-opt edge interchange and 1-shift search algorithms. However, these were found to be incorrect by Bianchi and Knowles (2002), and subsequently corrected in Bianchi et al. (2005). More recent advances in an attempt to extend these solution methods to the heterogeneous case are made by Bianchi and Gambardella (2007). Li (2013) and Li (2017) propose an alternative local search procedure, the so-called simulation-based multi-start search algorithm.

Apart from exact and heuristic solution procedures, metaheuristic solution approaches have also been proposed. Among the most popular metaheuristics for TSPSCs are ant colony optimisation (Bianchi et al., 2002a; Branke and Guntsch, 2004; Birattari et al., 2005, 2007; Balaprakash et al., 2007; Gambardella et al., 2012; Weyland et al., 2014), and particle swarm optimisation (Marinakis and Marinaki, 2009, 2010; Marinakis et al., 2015a). Other metaheuristics that also received some attention include simulated annealing (Bellalouna, 1993; Bowler et al., 2003; Gutjahr, 2004; Balaprakash et al., 2007), tabu search (Bellalouna, 1993; Beraldi et al., 2005), and memetic algorithms (Balaprakash et al., 2007, 2010). Besides these pure metaheuristics, a smaller group of literature also focuses on hybrid metaheuristics. These include the combination of scatter search with the nearest neighbour rule (Liu, 2007), ant colony optimisation with local search (Bianchi et al., 2006a), and particle swarm optimisation with simulated annealing (Cabrera et al., 2012). Table 2.1 provides an overview of the most important solution methods that have been proposed to solve the TSPSC.

2.2.2 The TSP with stochastic travel times

The TSP with stochastic travel times (TSPST) is a stochastic variant of the TSP where the travel times (i.e. the costs associated with traversing an edge) are merely known with some probability. As a result, the costs that make up the objective function of the problem are stochastic as well. The problem was described by

Author(s)	Year	Model	Model type	Solution type	Solution approaches
Jaillet	1985	SPR	Homogeneous	Exact, heuristics	Exact enumerative approaches, savings algorithm, nearest neighbour algorithm, hill-climbing heuristics, spacefilling curves, partitioning approaches, minimum spanning trees
Jézéquel	1985	SPR	Homogeneous	Heuristics	Savings algorithm, nearest neighbour algorithm
Rossi and Gavioli	1987	SPR	Homogeneous	Heuristics	Savings algorithm, nearest neighbour algorithm
Bertsimas	1988	SPR	Heterogeneous	Heuristics	TSP algorithms, radial sort, spacefilling curve heuristic, partitioning heuristic, nearest neighbour algorithm
Chervi	1990	SPR	Homogeneous	Metaheuristics	Stochastic annealing, local search
Bertsimas and Howell	1993	SPR	Heterogeneous	Heuristics	Christofides heuristic, spacefilling curve heuristic, nearest neighbour algorithm, 2-opt algorithm, 2-p-opt algorithm, 3-opt algorithm, 1-shift algorithm
Laporte and Louveaux	1993	SPR	Theoretical	Exact	Integer L-shaped algorithm
Bellalouna	1993	SPR	Homogeneous	Metaheuristics	Simulated annealing, 1-shift, tabu search
Laporte, Louveaux, and Mercure	1994	SPR	Heterogeneous	Exact	Integer L-shaped algorithm
Rosenow	1999	SPR	Homogeneous	Exact	Branch-and-bound
Bianchi, Gambardella, and Dorigo	2002a	SPR	Homogeneous	Metaheuristics	Ant colony optimisation
Bowler, Fink, and Ball	2003	SPR	Homogeneous	Metaheuristics	Simulated annealing
Branke and Guntsch	2004	SPR	Heterogeneous	Metaheuristics	Ant colony optimisation, Hilbertsorting, 1-shift, depth-based heuristics, angle-sort heuristics
Gutjahr	2004	SPR	Heterogeneous	Metaheuristics	Ant colony optimisation, simulated annealing
Tang and Miller-Hooks	2004	SPR	Heterogeneous	Metaheuristics	Simulated annealing, local search: k-opt, or-opt
Bianchi, Knowles, and Bowler	2005	SPR	Homogeneous	Heuristics	Local search: 2-p-opt, 1-shift
Birattari, Balaprakash, and Dorigo	2005	SPR	Homogeneous	Metaheuristics	Ant colony optimisation, F-race
Bianchi	2006	SPR	Heterogeneous	Metaheuristics	Ant colony optimisation, nearest neighbour algorithm
Bianchi, Birattari, Chiarandini, Manfrin, Mastrolilli, Paquete, Rossi-Doria, and Schiavinotto	2006a	SPR	Heterogeneous	Metaheuristics	Hybrid ant colony optimisation+local search
Bianchi and Gambardella	2007	SPR	Heterogeneous	Metaheuristics	Ant colony optimisation, 1-shift
Bianchi and Campbell	2007	SPR	Heterogeneous	Heuristics	Local search: 2-p-opt, 1-shift
Balaprakash, Birattari, Stützle, and Dorigo	2007	SPR	Homogeneous	Metaheuristics	Iterated local search algorithms, simulated annealing, ant colony optimisation, memetic algorithm
Liu	2007	SPR	Heterogeneous	Metaheuristics	Hybrid scatter search+nearest neighbor rule
Marinakis, Migdalas, and Pardalos	2008	SPR	Heterogeneous	Metaheuristics	ENS-GRASP
Birattari, Balaprakash, Stützle, and Dorigo	2008	SPR	Heterogeneous	Heuristics	Local search: 2.5-opt
Balaprakash, Birattari, Stützle, Yuan, and Dorigo	2009b	SPR	Heterogeneous	Metaheuristics	Ant colony optimisation, local search

Table 2.1: Summary and classification of literature about solution approaches for the TSPSC.

Author(s)	Year	Model	Model type	Solution type	Solution approaches
Marinakis and Marinaki	2009	SPR	Heterogeneous	Metaheuristics	Honey Bees Mating Optimisation algorithm
Weyland, Bianchi, and Gambardella	2009b	SPR	Heterogeneous	Metaheuristics	Random restart local search, iterated local search, 1-shift-delta, 3-opt, 2.5-opt
Balaprakash, Birattari, Stützle, and Dorigo	2010	SPR	Heterogeneous	Metaheuristics	Iterated local search, memetic algorithm, simulated annealing
Marinakis and Marinaki	2010	SPR	Heterogeneous	Metaheuristics	Particle Swarm Optimisation
Liu	2010	SPR	Heterogeneous	Metaheuristics	Nearest neighbour algorithms, genetic algorithm
Li	2013	SPR	Heterogeneous	Heuristics	Multi-start local search
Mahfoudh, Khaznaji, and Bellalouna	2015	SPR	Homogeneous	Exact	Branch-and-bound
Weiler, Biesinger, Hu, and Raidl	2015	SPR	Homogeneous	Metaheuristics	Insertion heuristics, variable neighbourhood search
Weyland, Montemanni, and Gambardella	2015	SPR	Homogeneous	Metaheuristics	GPGPU metaheuristic framework, local search
Amar, Khaznaji, and Bellalouna	2017	SPR	Homogeneous	Exact	Branch-and-bound
Marinakis, Marinaki, and Migdalas	2017	SPR	Heterogeneous	Metaheuristics	Bumble Bee Optimisation algorithm

Table 2.1: Summary and classification of literature about solution approaches for the TSPSC (*continued*).

Leipälä (1978) and first solved to optimality using a branch-and-cut algorithm by Laporte et al. (1992). Since a great deal of the set-up of the VRP with stochastic travel times (VRPST) is essentially identical to the TSPST, most literature does not make any specific distinction between these problems. Following this line of thought, the papers described in this paragraph are not exclusively limited to the TSPST, but also often covers the VRPST, and vice versa.

The first among the ones to solve the TSPST using heuristics is Kao (1978), who proposes a preference order dynamic program along with an implicit enumeration method. Carraway et al. (1989) develop a heuristic that generalises the dynamic programming (DP) approach devised by Kao (1978) and solves the TSPST subsequently. More recent attempts to solve the TSPST using dynamic programming have been made by Secomandi (2003), who embeds a cyclic heuristic, and Jula et al. (2006) and Chang et al. (2009), who also consider stochastic service times besides stochastic travel times. Taş et al. (2014) apply a branch-and-price approach to the VRPST with soft time windows, and Errico et al. (2016) use a branch-and-cut algorithm to solve the VRP with hard time windows

and stochastic service times. A dynamic formulation of the robust VRP with both stochastic demand and travel times was proposed and solved by Lee et al. (2012).

Heuristic approaches that do not rely on dynamic programming include Kenyon and Morton (2003) and Verweij et al. (2003). Kenyon and Morton (2003) integrate a branch-and-cut framework in a Monte Carlo simulation heuristic, and Verweij et al. (2003) apply sample average approximation methods.

A slightly different version of the TSPST adds multiple depots to the original set-up of the problem (m-TSPST). In this version, the salesman has to return to the same depot it initially left from after completing a (sub)tour along a subset of customers. Lambert et al. (1993) propose an adaptation of the Clarke and Wright (1964) savings algorithm in a multi-depot setting of the TSPST (m-TSPST). Laporte et al. (1992) propose a chance constrained programming (CCP) formulation along with two SPR formulations, and extend the m-TSPST with stochastic service times.

Author(s)	Year	Model	Solution type	Solution approaches
Leipälä	1978	–	Theoretical	Nearest neighbor rule
Kao	1978	DP	Heuristics	Preference order dynamic program with branch-and-bound, implicit enumeration algorithm
Carraway, Morin, and Moskowitz	1989	DP	Heuristics	Preference order dynamic program
Laporte, Louveaux, and Mercure	1992	CCP, SPR	Exact	Branch-and-cut algorithm (early adaptation of the integer L-shaped method)
Lambert, Laporte, and Louveaux	1993	CCP	Heuristics	Savings-based algorithm
Kenyon and Morton	2003	CCP, SPR	Heuristics	Branch-and-cut with Monte Carlo
Secomandi	2003	DP	Heuristics	Cyclic heuristic
Verweij, Ahmed, Kleywegt, Nemhauser, and Shapiro	2003	SPR	Heuristics	Sample average approximation method
Gutjahr	2004	SPR	Metaheuristics	Simulated annealing, ant colony optimisation
Jula, Dessouky, and Ioannou	2006	DP	Heuristics	Dynamic programming
Chang, Wan, and Ooi	2009	DP	Exact	FAN algorithm, dynamic programming
Jabali, Van Woensel, De Kok, Lecluyse, and Peremans	2009	DP	Metaheuristics	Tabu search
Li, Tian, and Leung	2010	CCP, SPR	Metaheuristics	Tabu search
Lee, Lee, and Park	2012	DP	Exact	Dynamic Programming
Lei, Laporte, and Guo	2012	SPR	Metaheuristics	Variable neighbourhood search
Zhang, Chaovalitwongse, and Zhang	2012	CCP	Metaheuristics	Scatter search, genetic algorithm
Taş, Dellaert, Van Woensel, and De Kok	2013	SPR	Metaheuristics	Tabu search
Taş, Gendreau, Dellaert, Van Woensel, and De Kok	2014	SPR	Exact	Branch-and-price
Errico, Desaulniers, Gendreau, Rei, and Rousseau	2016	DP	Exact	Branch-and-cut

Table 2.2: Summary and classification of literature about solution approaches for routing problems with stochastic travel times.

Only few papers address the application of metaheuristics to the TSPST. Gutjahr (2004) apply simulated annealing and ant colony optimisation to the TSPST. Li et al. (2010), Jabali et al. (2009), and Taş et al. (2013) investigate the application of tabu search to solve the TSPST and two time-constrained variants of the VRPST, respectively. Lei et al. (2012) applies a variable neighbourhood search metaheuristic to solve the capacitated VRP with stochastic service times, and Zhang et al. (2012) apply scatter search and a genetic algorithm to the VRPST with simultaneous pick-ups and deliveries. Table 2.2 gives an overview of the solution methods that address the TSPST and VRPST.

2.3 The stochastic vehicle routing problem

The stochastic vehicle routing problem (SVRP) is a generalisation of the STSP, and vice versa, the STSP may be regarded as a special case of the SVRP. In the SVRP, a depot is added to the graph of customers. This depot dispatches one or more vehicles from which the customers will receive their service. A vehicle must always return to the depot, and consequently, it must form a complete tour among the customers it serves. Furthermore, a vehicle is subject to a nonnegative demand capacity, which it may not exceed. Every customer discloses some demand that needs to be satisfied by a vehicle. As such, the demand of any customer cannot exceed the capacity of the largest vehicle. Most often, all vehicles are assumed to have identical capacities, and the total number of vehicles that can be dispatched is limited. The objective is to find a set of tours for the smallest number of possible vehicles such that all customers are served against minimum costs.

Stochastic variations take one or more of the components of this basic set-up as uncertain: That can be either uncertainty in demand, the presence of the customers, travel (or service) times, or a combination of one or more of these. Some of the solution methods for the STSP can be generalised to also tackle some of the stochastic features of the SVRP. This section only deals with the literature that specifically addresses SVRPs.

2.3.1 The vehicle routing problem with stochastic demands

In the vehicle routing problem with stochastic demands (VRPSD), the demands of the customers are random variables. Usually, the demands are considered to be independent and identically distributed (i.i.d.) over the set of customers. The VRPSD is by far the most described variant of all stochastic routing problems. Therefore, an extensive body of literature is devoted to the VRPSD. The literature is summarised chronologically in Table 2.3.

The VRPSD was first introduced in a multi-depot context by Tillman (1969). But it was not solved to optimality using exact methods until 1989, when Laporte et al. (1989) proposed a branch-and-cut algorithm that relaxes some of the constraints. An optimal solution approach that received more attention is the integer L-shaped algorithm by Laporte and Louveaux (1993). Many applications of this algorithm to the VRPSD have been studied ever since, for example, by Séguin (1994), Gendreau et al. (1995), Laporte and Louveaux (1998), Hjorring and Holt (1999) and Laporte et al. (2002). Another optimal solution approach was presented by Dror (1993), who proposes to rewrite the problem into a deterministic formulation and solve the problem to optimality using traditional deterministic methods. Other exact algorithms that were developed more recently include Sungur et al. (2008), who analyse a robust solution approach for the VRPSD, Christiansen and Lysgaard (2007) and Christiansen et al. (2009), who study branch-and-price algorithms. Jabali et al. (2012) propose an formulation for the multi-VRPSD and use the integer L-shaped algorithm to solve it. A dynamic formulation of the robust VRP with both stochastic demand and travel times was proposed and solved by Lee et al. (2012). Gauvin et al. (2014) apply a branch-cut-and-price algorithm to the VRPSD.

The first heuristic to solve the VRPSD is put forward by Tillman (1969), the founder of the VRPSD. Tillman (1969) describes a heuristic based on the Clarke and Wright (1964) savings algorithm in his paper, where penalties are incurred whenever vehicles are almost empty or filled over capacity. But a series of papers by Stewart and Golden (Golden and Stewart, 1978; Golden and

Author(s)	Year	Model	Distribution	Solution type	Solution approaches
Tillman	1969	–	P	Heuristics	Savings-based heuristic
Golden and Stewart	1978	CCP	P	Heuristics	Savings-based heuristic
Golden and Yee	1979	CCP	B, NB, G	Heuristics	Savings-based heuristic
Stewart	1981	CCP, SPR	N	Heuristics	Savings-based heuristic, Lagrangian method
Stewart and Golden	1983	CCP, SPR	N	Heuristics	Savings-based heuristic, Lagrangian method
Dror and Trudeau	1986	CCP, SPR	U, N	Heuristics	Savings-based heuristic
Bertsimas	1988	–	–	Heuristics	Cyclic heuristic, greedy heuristic
Laporte, Louveaux, and Mercure	1989	CCP	U	Exact	Branch-and-cut algorithm
Bertsimas, Jaillet, and Odoni	1990	–	–	Heuristics	Greedy heuristic
Bertsimas	1992	–	–	Heuristics	Cyclic heuristic
Bouzaïene-Ayari, Dror, and Laporte	1992	–	–	Heuristics	Savings-based heuristic
Teodorović and Pavković	1992	CCP	U	Metaheuristic	Simulated annealing
Dror	1993	SPR, M	–	Exact	Deterministic solution approaches
Laporte and Louveaux	1993	SPR	–	Exact	Integer L-shaped method
Séguin	1994	SPR	U	Exact	Integer L-shaped method
Bertsimas, Chervi, and Peterson	1995	–	U, N	Heuristics	Cyclic heuristic, savings-based heuristic, clustering heuristic
Gendreau, Laporte, and Séguin	1995	SPR	U	Exact	Integer L-shaped method
Laporte and Louveaux	1998	SPR	–	Exact	Integer L-shaped method
Secomandi et al.	1998	M, DP	–	Exact, heuristics	Exact dynamic programming algorithm, nearest neighbour rule, rollout algorithms, check-point heuristic
Hjorring and Holt	1999	SPR	U	Exact	Integer L-shaped method
Secomandi	2000	DP	U	Heuristics	Optimistic approximate algorithm, rollout algorithm, nearest neighbour rule
Yang, Mathur, and Ballou	2000	DP	T	Heuristics	Savings-based heuristic, route-and-cluster heuristics
Secomandi	2001	DP	U	Heuristics	Rollout algorithm
Laporte, Louveaux, and Van Hamme	2002	SPR	P, N	Exact	Integer L-shaped method
Secomandi	2003	DP	U	Heuristics	Rollout algorithm with cyclic heuristic
Bianchi, Birattari, Chiarandini, Manfrin, Mastrolilli, Paquete, Rossi-Doria, and Schiavinotto	2004	DP	U	Metaheuristic	Iterated local search, tabu search, simulated annealing, ant colony optimisation, evolutionary algorithms
Mak and Guo	2004	SPR	U	Metaheuristic	Genetic algorithm
Chepuri and Homem-de Mello	2005	–	G	Heuristics	Cross-entropy method, sample average approximation method
Bianchi, Birattari, Chiarandini, Manfrin, Mastrolilli, Paquete, Rossi-Doria, and Schiavinotto	2006a	DP	U	Metaheuristic	Hybrids: Local search+evolutionary algorithms, local search+ant colony optimisation, local search+simulated annealing, local search+tabu search

Table 2.3: Summary and classification of literature about solution approaches for the VRPSD. Model: Chance Constraint Programming (*CCP*), Stochastic Programming with Recourse (*SPR*), Markovian model (*M*), Robust model (*R*), Dynamic Programming (*DP*); Distribution: Distribution of the demand in the empirical setting (if any/if reported): Poisson (*P*), Bernoulli (*Ber*), Binomial (*B*), Negative binomial (*NB*), Normal (*N*), Uniform (*U*), Gamma (*G*), Triangular distribution (*T*), Lognormal distribution (*LN*).

Author(s)	Year	Model	Distribution	Solution type	Solution approaches
Lu, Wu, and Zhang	2006	–	–	Metaheuristic	Particle swarm optimisation, genetic algorithm
Lu and Tan	2006	–	–	Metaheuristic	Particle swarm optimisation
Ak and Erera	2007	CCP, SPR	U	Metaheuristic	Tabu search
Christiansen and Lysgaard	2007	SPR	P	Exact	Branch-and-price algorithm
Haugland, Ho, and Laporte	2007	SPR	B	Metaheuristic	Tabu search
Hvattum, Løkketangen, and Laporte	2007	SPR, DP	U	Heuristics	Branch-and-regret heuristic
Tan, Cheong, and Goh	2007	SPR	N	Metaheuristic	Evolutionary algorithm
Zhao	2007	CCP	N	Metaheuristic	Particle swarm optimisation
Ismail and Irhamah	2008	SPR	U	Metaheuristic	Genetic algorithm, tabu search, hybrid genetic algorithm+tabu search
Peng and Zhu	2008	DP	U	Metaheuristic	Particle swarm optimisation
Novoa and Storer	2009	DP	U	Heuristics	Rollout algorithm with look-ahead policies
Secomandi and Margot	2009	M, DP	U	Heuristic	Rollout algorithm
Shen, Ordónez, and Dessouky	2009	CCP	LN	Metaheuristic	Tabu search
Sungur, Ordónez, and Dessouky	2008	R	–	Exact	Robust optimisation approach
Irhamah and Ismail	2009	SPR	U	Metaheuristic	Genetic algorithms
Christiansen, Lysgaard, and Wøhlk	2009	SPR	P	Exact	Branch-and-price algorithm
Mendoza, Castanier, Guéret, Medaglia, and Velasco	2010	SPR	N	Metaheuristic	Memetic algorithms
Rei, Gendreau, and Soriano	2010	SPR	N	Heuristics	Hybrid monte-carlo branching algorithm, integer L-shaped method, Or-opt
Mendoza, Castanier, Guéret, Medaglia, and Velasco	2011	SPR	N	Heuristics	Savings-based heuristic, look-ahead heuristic, 2-opt algorithm
Shanmugam, Ganesan, and Vanathi	2011	DP	U	Metaheuristic	Genetic algorithm, particle swarm optimisation
Lee, Lee, and Park	2012	DP	–	Exact	Dynamic programming algorithm
Jabali, Gendreau, and Laporte	2012	SPR	U	Exact	Integer L-shaped method
Marinakis, Iordanidou, and Marinaki	2013	SPR	–	Exact	Particle swarm optimisation
Gauvin, Desaulniers, and Gendreau	2014	SPR	P	Exact	Branch-cut-and-price
Marinakis, Marinaki, and Spanou	2015b	SPR	–	Metaheuristic	Memetic algorithms, variable neighbourhood search

Table 2.3: Summary and classification of literature about solution approaches for the VRPSD *(Continued)*. Model: Chance Constraint Programming (*CCP*), Stochastic Programming with Recourse (*SPR*), Markovian model (*M*), Robust model (*R*), Dynamic Programming (*DP*); Distribution: Distribution of the demand in the empirical setting (if any/if reported): Poisson (*P*), Bernoulli (*Ber*), Binomial (*B*), Negative binomial (*NB*), Normal (*N*), Uniform (*U*), Gamma (*G*), Triangular distribution (*T*), Lognormal distribution (*LN*).

Yee, 1979; Stewart and Golden, 1980, 1983) and Bertsimas (Bertsimas, 1988, 1992; Bertsimas et al., 1995) gave way to much of the later literature that appeared on the VRPSD. Whereas Bertsimas (1988) derives a set of results on

the bounds and properties of the VRPSD, Stewart and Golden (1983) propose a CCP formulation and two SPR formulations for the VRPSD. Furthermore, Stewart and Golden (1983) also develop another savings-based heuristic, along with a heuristic based on Lagrangian relaxation that first appeared in Stewart (1981). Many other savings-based heuristics and applications beside the ones by Tillman (1969) and Stewart and Golden (1983) have been studied since the problem was first introduced, including the variants by Golden and Stewart (1978), Stewart and Golden (1980), Stewart et al. (1982), Dror and Trudeau (1986), Bouzaïene-Ayari et al. (1992) and Mendoza et al. (2011). Greedy heuristics are studied in Bertsimas (1988) and Bertsimas et al. (1990), along with cyclic heuristic procedures in Bertsimas (1992). A number of alternative heuristics that integrate less common methods were studied more recently, including a route-and-cluster heuristic (Yang et al., 2000), a cross-entropy heuristic (Chepuri and Homem-de Mello, 2005), a branch-and-regret heuristic (Hvattum et al., 2007), a local-branching with Monte Carlo sampling heuristic (Rei et al., 2010) and a look-ahead heuristic (Mendoza et al., 2011).

Apart from exact and heuristic solution procedures, a number of pure and hybrid metaheuristic solution approaches have also been proposed. Among the most extensively studied pure metaheuristics are genetic algorithms (Mak and Guo, 2004; Ismail and Irhamah, 2008; Irhamah and Ismail, 2009; Shanmugam et al., 2011) and particle swarm optimisation (Lu et al., 2006; Lu and Tan, 2006; Zhao, 2007; Peng and Zhu, 2008; Shanmugam et al., 2011). A number of dynamic and neurodynamic programming solution approaches in the context of VRPSD are addressed in a series of papers by Secomandi (Secomandi et al., 1998; Secomandi, 2000, 2001, 2003; Secomandi and Margot, 2009; Novoa and Storer, 2009). A paper by Bianchi et al. (2004) evaluates a whole range of metaheuristics for the VRPSD, including tabu search, evolutionary computation and simulated annealing, local search and ant colony optimisation. Other efforts that study these metaheuristics are made by Ak and Erera (2007), Haugland et al. (2007), Teodorović and Pavković (1992), and Tan et al. (2007). Shen et al. (2009) also study the performance of

tabu search, but in a setting where not only demands but also travel times are stochastic. Mendoza et al. (2010) and Marinakis et al. (2015b) study memetic algorithms. Finally, Marinakis et al. (2013) propose a particle swarm optimisation metaheuristic for the VRPSD.

Hybrid metaheuristics have also recently gained some popularity. In particular, Bianchi et al. (2006a) address multiple hybrid metaheuristics, including hybrids between local search methods and evolutionary algorithms, ant colony optimisation, simulated annealing and tabu search. Other hybrid metaheuristics are considered by Ismail and Irhamah (2008), who develop a hybrid version of a genetic algorithm with tabu search, and Rei et al. (2010), who develop a hybrid metaheuristic by combining local search with Monte Carlo sampling.

2.3.2 The vehicle routing problem with stochastic customers

The VRPSC is a direct extension of the TSPSC, in which all the customers are present with some probability. But now there are multiple vehicles to account for, which all have a predetermined capacity. Because the VRPSC can essentially be regarded as a generalisation of the TSPSC, many of the methods discussed in section 2.2.1 also directly apply to the VRPSC. This section is only concerned with the literature that specifically addresses the VRPSC.

The VRPSC was first introduced together with the TSPSC in the context of unit demands by Jaillet (1985). The problem is formalised by two more papers by Jaillet (1987) and Jaillet and Odoni (1988). Bertsimas (1988) discusses additional bounds and properties for the VRPSC. The unit demand assumption in these papers implies that every customer has a demand of either 1 or 0, in the latter case of which the customer is simply not present. The VRPSC is therefore also closely related to the VRPSD. The VRPSC was not discussed in a general demand context until the paper by Waters (1989). Because the problem definitions of the VRPSC, the TPSC and the VRPSD largely overlap, only a small set of papers specifically targets the case of the VRP where only customers are stochastic.

The first to derive an exact method are Séguin (1994) and Gendreau et al. (1995), who develop an exact integer L-shaped method for the case in which both customers and demands are stochastic. Jézéquel (1985) discusses a number of TSP based heuristics to solve the VRPSC with unit demands. Bertsimas considers a cyclic heuristic for the VRPSC in Bertsimas (1988) and Bertsimas (1992). Another heuristic, called the reoptimisation heuristic, is developed by Benton and Rossetti (1992) in a setting where also both the customers and the demands are uncertain. Tabu search metaheuristics for the stochastic customers and demands case are investigated by Gendreau et al. (1996a) and Balaprakash (2010). In addition, Balaprakash (2010) studies metaheuristics based on an ant colony system and local search for the VRPSCD. More recently, Balaprakash et al. (2015) apply a set of metaheuristics enhanced by empirical estimation. The solution approaches for the VRPSC are summarised in table 2.4.

Author(s)	Year	Model	Model type	Stoch. comp.	Solution type	Solution approaches
Jaillet	1985	SPR	Homogeneous	C,CD	Heuristics	PTSP-based heuristics: VRPSC with unit demands
Jézéquel	1985	SPR	Homogeneous	C,CD	Heuristics	TSP-based heuristics
Bertsimas	1988	SPR	Heterogeneous	CD	Heuristics	Cyclic heuristic
Waters	1989	SPR	Heterogeneous	CD	Heuristics	Savings-based heuristic
Benton and Rossetti	1992	–	Heterogeneous	CD	Heuristics	Reoptimisation heuristic
Bertsimas	1992	SPR	Heterogeneous	C,CD	Heuristics	Cyclic heuristics
Séguin	1994	SPR	Heterogeneous	CD	Exact, metaheuristic	Integer L-shaped method, tabu search
Gendreau, Laporte, and Séguin	1995	SPR	Heterogeneous	CD	Exact	Integer L-shaped method
Gendreau, Laporte, and Séguin	1996a	–	Heterogeneous	CD	Metaheuristic	Tabu search
Balaprakash	2010	SPR	Heterogeneous	CD	Metaheuristic	Local search, tabu search, ant colony optimization
Balaprakash, Birattari, Stützle, and Dorigo	2015	SPR	Heterogeneous	CD	Metaheuristic	Local search, tabu search, random restart local search, iterated local search, memetic algorithms, ant colony optimisation

Table 2.4: Summary and classification of literature about solution approaches for the vehicle routing problem with stochastic customers (VRPSC). Stochastic components (*stoch. comp.*): Stochastic customers (*C*) or stochastic customers and demands (*CD*).

2.4 Conclusion

The literature covering stochastic routing problems has gained ample attention in the past decades. A large body of literature is devoted to the VRPSD, although

many advances have also been made covering the TSPSC and VRPSC. In that respect, stochastic routing problems in which travel times are random seems to be the relatively most uncovered category. It should be recognised, however, that many problem descriptions do not deal with any specific instance of a stochastic routing problem in particular, but can be transformed into another stochastic routing problem after only slight modifications. That is, in some cases the problem descriptions largely overlap. This is for example the case for the TSPSC and the VRPSC, but also for the VRPSD and the VRPSC. The descriptions of these problems can be regarded as generalisations of one another, which opens up the opportunity of solving different stochastic routing problems using the same (or very similar) solution frameworks.

Furthermore, a pattern emerges among the literature covering stochastic routing problems. Most of the earliest literature describes heuristic solution procedures. Among the most popular heuristics are algorithms based on Clarke and Wright's (1964) savings approach, the nearest neighbor principle and cyclic heuristics. The introduction of the integer L-shaped method by Laporte and Louveaux (1993) led to the introduction of exact methods to the field of stochastic routing, although only few alternative exact methods have since been proposed. Metaheuristics have gained popularity only more recently. Most adaptations revolve around a small set of alternatives, in particular, tabu search, local search, particle swarm optimisation, ant colony optimisation, evolutionary algorithms and genetic algorithms. Hybrid metaheuristics have been addressed almost exclusively in the context of VRPSDs, in which a local search algorithm is combined with another metaheuristic.

3 | The Probabilistic Travelling Salesman Problem

The PTSP is "*perhaps the most fundamental stochastic routing problem that can be defined*" (Powell et al., 1995, p. 149). It is derived from the TSP (Dantzig et al., 1954), the latter being one of the most extensively studied classical uncapacitated node routing problems under deterministic and static conditions. Similar to the TSP, the objective of the PTSP is to determine a Hamiltonian cycle, called a tour, along a fixed set of customers in a graph whilst minimising the costs of travel (see chapter 2).

The difference between the TSP and PTSP is defined by the presence of the customers in the graph: Whereas the TSP assumes that all customers in the graph are always present in every instance of the problem (i.e., with probability 1), the PTSP relaxes this assumption. Instead, the PTSP assumes that every customer is only present with some predetermined and known probability associated with every customer in the graph. Therefore, only a subset of the customers may be present in any given instance of the problem. Although this makes the PTSP a fascinating problem from a mathematical perspective, it is also notoriously difficult to solve.

The PTSP was first introduced by Jaillet (1985) in his PhD thesis. He uses a star-shaped example to illustrate that solutions that are good for a TSP can be poor fit for a PTSP, and vice versa. As a consequence, methods that perform well for deterministic problems may perform poorly in the context of the PTSP. Not

surprisingly, many different problem-specific solution approaches have since been proposed, both exact and heuristic in nature. Chapter 4 gives a description of these solution methods.

Sparked by the revelation of differences in good TSP versus PTSP solutions, research has not only restricted its attention to cases that demonstrate the PTSP's theoretical relevance. Soon after its introduction it was found that the PTSP also has many practical applications. For example, the PTSP arises in real-life situations such as the daily delivery of parcels (Jaillet and Odoni, 1988; Tang and Miller-Hooks, 2007), meals on wheels (Bartholdi et al., 1983), money collection by central banks (Bertsimas, 1988; Chervi, 1990), job scheduling (Bertsimas et al., 1990) and online grocery services (Campbell, 2006). With the rise of e-commerce and the increase of parcel delivery points characterised by uncertain customer presences (see chapter 1), the PTSP will remain a hot topic with ample practical applications in the foreseeable future. Or as Gendreau et al. (2016) put it, in a recent invited article for the 50th anniversary of Transportation Research about 'Future Research Directions in Stochastic Vehicle Routing':

> *"The presence of customers is pivotal when planning tactical routes.* [...] *This presence is influenced by a number of temporal factors and customer characteristics.* [...] *If during a planning period (e.g., a day) certain epochs are substantially different with respect to the likelihood of a customer's presence, then this should be captured."*
>
> — Gendreau, Jabali, and Rei (2016, p. 1169)

Many of the aforementioned theoretical and practical oriented papers focus on a specific formulation of the PTSP, namely homogeneous PTSPs and heterogeneous PTSPs (section 2.2.1). This chapter addresses both the heterogeneous and homogeneous versions of the PTSP.

The remainder of this chapter is organised as follows. Section 3.1 states the

formal definition and solution framework of the PTSP. Section 3.2 explains how to calculate the length of a PTSP tour. Section 3.3 elaborates upon this by developing an alternative representation that might be more appealing for serving several analytical purposes. Section 3.4 discusses how the expected length calculation is integrated in the objective function, and presents the complete and relaxed stochastic integer linear programs that describe the PTSP. Section 3.5 summarises the main results of this chapter.

3.1 A two-stage stochastic programming with recourse model

In environments characterised by uncertainty, managers could opt for either reactive or proactive planning of operations. As far as most practical settings are concerned, the proactive approach is more desirable. Under this approach, the TSPSC is addressed in two stages.

In the first stage, a so-called *a priori tour* is constructed. More specifically, a Hamiltonian salesman tour covering all customers – and potentially also a depot – is determined without prior knowledge of the realisation of customer orders. In fact, only the probability of each customer requesting a visit is known, e.g., based on historical data, so as to minimise the expected total distance travelled by the salesman.

In the second stage of the problem, it is revealed which customers require a visit and which customers do not. Using this information, a so-called *a posteriori tour* can be constructed. The a posteriori tour is a modified version of the a priori tour where those customers who do not request a visit, referred to as absent customers, are subject to a recourse action. Generally speaking, a *recourse policy* is a method that describes what action – i.e., the *recourse action* – should be taken in order to deal with the a posteriori outcomes of the stochastic variables. The recourse policy

of the PTSP prescribes that all customers should be visited in the same order as they appear in the a priori tour, performing the recourse action of skipping any absent customers that are encountered along the way. The tour that results from this strategy is the a posteriori tour.

Observe that it is the a posteriori tour that is implemented or given to the driver. Therefore, the actual costs of the tour should be determined with respect to the a posteriori outcome of the second stage, i.e., the tour that results when visit requests materialise. The difficulty arises here that, because of the stochastic nature of the problem, these visit requests are still unknown when the a priori tour needs to be determined in the first stage of the problem. Therefore, instead of directly minimising the length of the a posteriori tour, the PTSP is concerned with finding an a priori tour that carries the highest likelihood of minimising the average length of all possible a posteriori outcomes. The objective function of the PTSP reflects the difficulties that arise in this challenge, by explicitly accounting for the uncertainties in the customer requests. Specifically, the PTSP is concerned with finding, among all possible a priori tours, the a priori tour of which the *expected length* is minimal.

The general class of approaches that deal with such a multi-stage process is also more commonly referred to as *stochastic programming with recourse* (SPR). The PTSP can be regarded as the SPR-variant of the more general problem class of TSPSCs, that adopts the specific recourse policy of skipping over absent customers. To the best of my knowledge, no other variants of the TSPSC exist; therefore, the terms TSPSC and PTSP are often used interchangeably in the literature.

One should be able to compute the expected length of the a priori tour in order to evaluate the quality of a proposed (either intermediate or final) solution. Because of the differences with respect to the length calculation of its deterministic counterpart, the ability to evaluate the expected length is also crucial for the performance of solution procedures. In fact, many deterministic solution approaches are not very suited to construct tours for the PTSP. The complex

probabilistic nature of the PTSP has given rise to its own class of problem-specific solution procedures, as well as stochastic adaptations of existing deterministic approaches. Such procedures often rely on the repeated evaluation of the expected length of an a priori tour, or even integrate entire parts of this expression, in order to find a good solution. The next section discusses the derivation of the expected length of an a priori tour, as it is so central to the objective of finding and evaluating solutions.

3.2 The length of an a priori tour

The PTSP could be described in many different but equivalent ways. Hereafter, I present and contrast the original description provided by Jaillet (1985), and the extended variant provided by Bianchi (2003).

Let $G = (N, E, D)$ denote an undirected, complete and deterministically weighted graph with a set of nodes $N = \{1, \ldots, n\}$ representing n customers (or cities), the set of edges $E = \{(i, j) : i, j \in N, i \neq j\}$ representing the travel links between the customers, and the entries of the symmetric distance matrix $D = (d_{ij}; i, j \in N)$ representing the costs incurred or the distances required to travel over the edges. In the PTSP, the presence of any given customer is stochastic; therefore, every node $i \in N$ only requires a visit with some known probability. As a result, generally only a subset of customers $S \subseteq N$ requires a visit.

Because of this stochastic nature, the realisation of the subset $S \subseteq N$ of customers that require a visit is unknown. Therefore, the objective in the TSP with stochastic customers is to minimise the expected length of the a priori tour $\tau = (1, \ldots n)$ along every single customer in N, rather than minimise the unknown length of a realised a posteriori tour over S. The expected length of the a priori tour should account for the uncertainty in the stochastic customer presence. More formally, the PTSP defined on graph G requires the determination of a minimum

expected length Hamiltonian tour over the full set of customers N while accounting for the likelihood that only a random realisation S eventually requires a visit.

Let p_i denote the marginal probability corresponding to a positive response of the Bernoulli random variable $Y_i; i \in N$. Furthermore, let $\mathbf{Y} = (Y_1, \ldots, Y_n)$ be a vector of Bernoulli random variables with realisations $\mathbf{y} = (y_1, \ldots, y_n)$, such that each random variable $Y_i \in \mathbf{Y}$ is linked to a response $y_i \in \mathbf{y}$. A random variable takes the response $Y_i = 1$ if customer i is present and $Y_i = 0$ if node i is absent. Therefore, each variable Y_i satisfies

$$g(Y_i) = p_i^{y_i}(1-p_i)^{1-y_i}, \tag{3.1}$$

where

$$p_i = \mathbb{E}(Y_i) = \Pr[Y_i = 1],$$
$$1 - p_i = 1 - \Pr[Y_i = 1] = \Pr[Y_i = 0].$$

Under the assumption that the realisations of nodes are independent and the presence of each customer follows the distribution above, the probability that a particular set of customers S requires a visit is given by

$$p(S) = \prod_{i \in S} p_i \prod_{i \in N-S} (1-p_i).$$

Note that $p(S)$ could be written as $P(S|N \setminus S) \times P(N \setminus S)$ and $P(S|N \setminus S) = P(S)$. Furthermore, $p(S) = p(S) \times P(N \setminus S)$ which equals $\prod_{i \in S} p_i \prod_{i \in N-S}(1-p_i)$ as a consequence of the independence of customers in S and $N \setminus S$.

In a stochastic programming with recourse modelling and solution framework, the expected length of any given tour along a subset S of actual customers to visit can be obtained by weighing every possible subset S on τ with its probability of occurring. Hence, under the assumption that the realisations of subsets of actual customers to visit S are independent on one hand, and a SPR modelling and solution framework is used on the other hand, the expected length of any given tour along

the subset S of actual customers to visit can be calculated using the law of total expectation (Bianchi, 2003):

$$\mathbb{E}[L(\tau)] = \sum_{S \subseteq N} p(S) L(\tau|S), \tag{3.2}$$

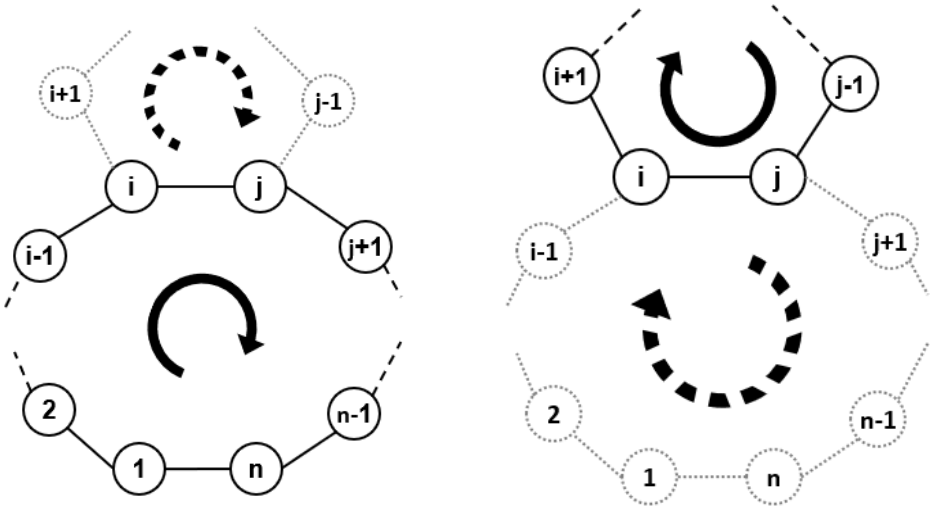
where τ denotes a Hamiltonian tour on S, $L(\tau)$ denotes the length of τ, $L(\tau|S)$ denotes the length of a Hamiltonian tour defined on S so that customers are visited in the same order as they appear in the a priori tour, and $p(S)$ denotes the probability of the subset S occurring (Bianchi, 2003).

Recall that, within a SPR modelling and solution framework, a common solution to deal with the stochastic nature of the PTSP is to first construct an a priori tour and then modify it once the actual customers to visit are known. In the first stage, an a priori tour along all customers in N is constructed. In the second stage, the tour is modified to account for only those customers that require a visit. That is, when the set S of customers that actually require a visit is revealed, the customers that do not require a visit (i.e., $N \setminus S$) are simply skipped, and those in S must be visited in the same order as they appear in the a priori tour. The resulting tour is called the a posteriori tour. The recourse policy of the PTSP is retained throughout this thesis regardless of the probability definition.

Notice that the probability that customers i and j are revealed where i is visited prior to j, and intermediate customers along the path $i+1, \ldots, j-1$ in the a priori tour do not require any visit (see Figure 3.1a for a graphical illustration of this scenario), is given by:

$$p_i p_j \prod_{k=i+1}^{j-1} (1 - p_k).$$

In fact, if event "customers i and j are revealed" and event "customers $i+1, \ldots, j-1$ are not revealed" are simultaneously taking place, then the probability of these events to jointly take place is the product of the probabilities of these individual events. Assuming that revealed and unrevealed customers are independent, the formula above follows.



(a) Visiting j after i (b) Visiting i after j

Figure 3.1: Visiting customers i and j in a tour ($i < j$).

Alternatively, customers i and j could be visited in the reverse order, in which case the probability that customers i and j are revealed where j is visited prior to i, and intermediate customers along the path $j+1,\ldots,n,1,\ldots,i-1$ in the a priori tour do not require any visit (see Figure 3.1b for a graphical illustration), is given by

$$p_i p_j \prod_{k=j+1}^{n} (1-p_k) \prod_{l=1}^{i-1} (1-p_l).$$

Therefore, the expected length of a random Hamiltonian tour τ, $E[L(\tau)] = \sum_{S\subseteq N} p(S)L(\tau|S)$, could be written as follows:

$$\mathbb{E}[L(\tau)] = \sum_{i=1}^{n} \sum_{j=i+1}^{n-1} \left[p_i p_j \prod_{k=i+1}^{j-1} (1-p_k) \right] d_{ij} + \sum_{j=1}^{n} \sum_{i=j+1}^{n-1} \left[p_j p_i \prod_{k=j+1}^{n} (1-p_k) \prod_{l=1}^{i-1} (1-p_l) \right] d_{ji}, \quad (3.3)$$

where $d_{ij} \in D$ denotes costs associated with traversing edge $(i,j) \in E$.

For the homogeneous PTSP with not only independent, but also identically

distributed (i.i.d.) Bernoulli pmfs ($p_i = p \forall i$), expression (3.3) reduces to:

$$\mathbb{E}[L(\tau)] = p^2 \sum_{r=0}^{n-2} (1-p)^r \sum_{j=1}^{n} d_{j,(j+1+r) \bmod n}, \tag{3.4}$$

where *mod* denotes the modulus operator. It provides a cleaner expression of the expected distance, whereby the requirement to loop over every individual node probability is dropped.

3.3 Weight-form notation

Jaillet (1985) derives various mathematical formulations for the PTSP under different scenarios. The differences between these formulations arise from the characteristics that describe the nodes. That is, Jaillet (1985) discerns deterministic nodes from stochastic nodes, which he refers to as black nodes and white nodes, respectively. A black node identifies a customer that always requires a visit; in stochastic terms, it is a customer that bears a probability $p_i = 1$ of requiring a visit. Observe that if all nodes in the graph G are black nodes, then the problem reduces to the usual (i.e. deterministic) TSP. A white node is a stochastic node in the sense I described before: It only requires a visit with some predefined probability p_i, and may or may not require a visit upon disclosing the a posteriori tour.

Since τ is a random Hamiltonian tour, any edge could be included in τ with a probability that depends on whether the edge is traversed in one direction or the other computed with reference to the a priori tour. Note that (3.3) could therefore be rewritten as:

$$\mathbb{E}[L(\tau)] = \sum_{(i,j) \in E} w_{ij} d_{ij} + \sum_{(j,i) \in E} w_{ji} d_{ji}, \tag{3.5}$$

where the weights

$$w_{ij} = \left[p_i p_j \prod_{k=i+1}^{j-1} (1-p_k) \right]$$

and

$$w_{ji} = \left[p_i p_j \prod_{k=j+1}^{n} (1-p_k) \prod_{l=1}^{i-1} (1-p_l) \right]$$

represent the probabilities that nodes i and j are present and the intermediary nodes $i+1, \ldots, j-1$ are not.

Now, let the following definitions apply to the remainder of this section:

- n^S denotes the number of stochastic nodes;
- n^D denotes the number of deterministic nodes in the graph such that $n^S + n^D = n$;
- W denotes a random variable with a general probability density function that represents the number of stochastic nodes that are present in the a posteriori tour;
- k denotes the number of nodes that do not require a visit in the a posteriori tour and therefore are skipped in an a priori tour ($k \leq n^S$);
- r denotes the number of nodes that are skipped in the a priori tour when one visits $j+r+1$ immediately after j ($r \leq k$);
- $k-r$ denotes the remaining number of skipped nodes in the a priori tour.

The first two notations follow from dividing the set of nodes N with cardinality n into two subsets, namely N^D, representing the set of deterministic customers (i.e., those customers who require a visit with probability 1) with cardinality n^D, and N^S, representing the set of stochastic customers (i.e., those customers who require a visit with a probability less than 1) with cardinality n^S.

Then the total number of scenarios where k out of n^S stochastic nodes do not require service by the a posteriori tour is $\binom{n^S}{k}$. Additionally, the total number of scenarios where there are exactly r skipped nodes between nodes j and $j+r+1$ is equal to $\binom{n^S-2-r}{k-r}$. Therefore, the probability of arc $(j, j+r+1)$ being present in any realisation of the problem is proportional to the cardinality of the subset of the total number of scenarios where k nodes do not require service and consists of exactly r skipped nodes between nodes j and $j+r+1$; that is, $\binom{n^S-2-r}{k-r} / \binom{n^S}{k}$. Note

that this probability is only defined for $k \geq r$; otherwise, it is zero. Consequently, $\alpha_r = \sum_{k=r}^{n^S-2} \left(\binom{n^S-2-r}{k-r} / \binom{n^S}{k} \right) \Pr\left(W = n^S - k\right)$ is the probability that a scenario takes place where there are exactly r skipped nodes between nodes j and $j+r+1$ in an a priori tour over all nodes regardless of the specification of the total number of skipped nodes k. Similarly, in the special cases that there is only one stochastic customer present or there are no stochastic customers present, we find $\alpha_{n^S-1} = 1/n^S \Pr(W = 1)$ and $\alpha_{n^S} = \Pr(W = 0)$, respectively.

Jaillet (1985) refers to the α_rs as weights. In his dissertation, he presents the PTSP in a slightly different way, which could be more appealing to practitioners. That is, the expected length of a Hamiltonian cycle or TSP tour, say τ, over the set of nodes N could be computed as follows:

$$\mathbb{E}[L(\tau)] = \sum_{r=0}^{n} \alpha_r L_{n^D}^{(r)}. \tag{3.6}$$

$L_{n^D}^{(r)}$ denotes the length of a realised tour involving n^D deterministic customers and skipping r nodes in the a priori tour. It is computed as follows:

$$L_{n^D}^{(r)} = \sum_{j=1}^{n} d_{j,j+r+1} | N_{j,j+r+1}^{D},$$

where the distance between nodes j and $j+r+1$ depends on whether there are deterministic nodes between them or not (i.e., set $N_{j,j+r+1}^{D}$ is empty or not) and is calculated accordingly; that is, $d_{j,j+r+1} | N_{j,j+r+1}^{D}$ is either the direct travel distance between j and $j+r+1$, or the indirect travel distance between j and $j+r+1$ going through whatever intermediate deterministic nodes there are between them.

In mathematical terms,

$$d_{j,j+r+1}|N^D_{j,j+r+1} = \begin{cases} d_{j,j+r+1} & \forall r \in \{0,1,\ldots,n^S-2\} \quad \text{if } |N^D_{j,j+r+1}| = 0 \text{ and } n^D \geq 0; \\ d_{j,j+r+1} & \forall r = n^S-1 \quad \text{if } |N^D_{j,j+r+1}| = 0 \text{ and } n^D \geq 1; \\ d_{j,j+r+1} & \forall r = n^S \quad \text{if } |N^D_{j,j+r+1}| = 0 \text{ and } n^D \geq 2; \\ \sum_{e=0}^{s} d_{k_e,k_{e+1}} & \forall r \in \{0,1,\ldots,n^S-2\} \quad \text{if } |N^D_{j,j+r+1}| > 0 \text{ and } n^D \geq 0; \\ \sum_{e=0}^{s} d_{k_e,k_{e+1}} & \forall r = n^S-1 \quad \text{if } |N^D_{j,j+r+1}| > 0 \text{ and } n^D \geq 1; \\ \sum_{e=0}^{s} d_{k_e,k_{e+1}} & \forall r = n^S \quad \text{if } |N^D_{j,j+r+1}| > 0 \text{ and } n^D \geq 2; \\ 0 & \text{otherwise,} \end{cases}$$

where $k_0 = j$, $k_{s+1} = j + r + 1$ and $k_1, k_2, \ldots, k_s$ is the sequence of deterministic nodes between j and $j + r + 1$, with $s \leq \min\{r, n^D\}$. In sum, the expected length of a Hamiltonian cycle is a weighted sum of $L^{(r)}_{n^D}$ where the weighting scheme consists of the probabilities of scenarios skipping r nodes.

3.4 A stochastic integer linear program

Laporte et al. (1994) adopt the set-up of Jaillet (1985) and amend his formulations to develop problem-specific solution procedures. The mathematical program can be described as follows.

Let $\mathbf{x}$ denote the vector of binary variables, where $x_{ij} = 1$ if the edge $(i,j) \in E$ connecting customers i and j is selected (i.e., $(i,j) \in \tau$), and $x_{ij} = 0$ otherwise. The binary variables are collected in a vector $\mathbf{x}$, such that $x_{ij} \in \mathbf{x}; i,j \in N$. Then

$$\min_{\mathbf{x}} \mathbb{E}_{\mathbf{y}}\left[L\left(\tau\left(\mathbf{y}|\mathbf{x}\right)\right)\right]$$

describes the objective of minimising the expected length of an a posteriori tour $\tau(\mathbf{y}|\mathbf{x})$ over a set of edges $\mathbf{x}$. Note that for a given set of a priori selected edges $\mathbf{x}$, the a posteriori tour $\tau(\mathbf{y}|\mathbf{x})$ only depends on the a posteriori revealed nodes $\mathbf{y}$.

Relaxing the objective function above yields a more accessible form that can be solved with traditional linear programming approaches. To accomplish this, observe that the expected length of the tour $\mathbb{E}_{\mathbf{y}}[L(\tau(\mathbf{y}|\mathbf{x}))]$ can be decomposed into two separate parts: the length of the part in the a priori tour $d\mathbf{x}$ that also needs to be visited in the a posteriori tour, and the expected length of the part in the a priori tour that will be skipped in the a posteriori tour as a result of performing a recourse action. Let us denote the length of the tour part that is skipped by $L(R(\mathbf{y}|\mathbf{x}))$, and its expectation by $\mathbb{E}_{\mathbf{y}}[L(R(\mathbf{y}|\mathbf{x}))]$, such that $\mathbb{E}_{\mathbf{y}}[L(\tau(\mathbf{y}|\mathbf{x}))] = d\mathbf{x} - \mathbb{E}_{\mathbf{y}}[L(R(\mathbf{y}|\mathbf{x}))]$. Replacing $-\mathbb{E}_{\mathbf{y}}[L(R(\mathbf{y}|\mathbf{x}))]$ by an approximation γ bounded from below by $L \leq \gamma$ gives the following result:

$$\min_{\mathbf{x}} d\mathbf{x} + \gamma.$$

Also taking the usual other TSP constraints also into account, the full formulations of both programs read as follows (Laporte et al., 1994).

Stochastic integer linear program

$$\min_{x} \mathbb{E}_{\mathbf{y}}[L(\tau(\mathbf{y}|\mathbf{x}))]$$

$$\text{s.t.} \quad \sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \qquad \forall k \in N$$

$$\sum_{\substack{(i,j)\in E;\\ i<j}} x_{ij} \leq |S| - 1 \qquad S \subset N; 3 \leq |S| \leq n-3$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in E$$

Relaxed linear problem

$$\min_{\mathbf{x}} d\mathbf{x} + \gamma$$

$$\text{s.t.} \quad \sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \qquad \forall k \in N$$

$$0 \leq x_{ij} \leq 1 \qquad (i,j) \in E$$

$$L \leq \gamma$$

The second and third lines of the unrelaxed formulation on the left ensure a node degree of 2 for all nodes in the graph ('degree equation'), and prohibit the formation of any subtours in the graph ('subtour elimination'), respectively. The last constraint ensures that integrality is maintained ('integrality constraint'). The relaxed problem on the right is obtained by removing the subtour-breaking

and integrality constraints, in addition to the relaxations of the objective function described above.

3.5 Conclusion

This chapter discusses the set-up and mathematical definition of the PTSP. The PTSP is an elegant problem, yet notoriously difficult to solve. The PTSP arises in many theoretical and real-world situations, and as such, exhibits both theoretical and practical relevance. The two-stage SPR formulation ensures that the PTSP embodies a sufficiently realistic representation of the more general problem class of TSPSCs. The mathematical definition of the PTSP is characterised by the presence of independently distributed Bernoulli random variables, representing the feature of stochastic customer presence or absence that distinguishes the PTSP from the TSP. In effect, the objective function of the PTSP evaluates to the *expected* length of a tour, rather than a deterministic (or fixed) length. This objective function can be written in a number of alternative yet equivalent ways, each one potentially appealing to different practitioners. This chapter first derives the general, heterogeneous PTSP set-up, and subsequently describes the specialisation to the homogeneous PTSP case. It then proceeds to describing a weight-form notation of the objective function, that may be particularly useful for analytical purposes. Both forms of the objective function may be integrated by the full SMIP, which takes the basic set-up of the mathematical program of the TSP, and replaces the objective function by its stochastic PTSP counterpart.

4 | A comparative analysis of solution methods for the PTSP

Since the introduction of the PTSP in 1985, a growing body of literature has since been devoted to understanding and solving the problem. There are some papers that specifically address the properties and bounds of the PTSP, such as Berman and Simchi-Levi (1988), Bertsimas (1988) and Jaillet (1993). But most papers about the PTSP attempt to actually solve the problem by proposing a solution procedure. These procedures deal almost exclusively with heuristic and metaheuristic methods, apart from a notable exception by Laporte et al. (1994), who propose an exact solution method. But to the best of my knowledge, a comprehensive overview along with a critical comparison of the most important solution methods for the PTSP has not been the topic of any paper.[1] This study attempts to close this gap.

Many different problem-specific solution approaches have been proposed for the PTSP, both exact and heuristic in nature. Apart from a notable exception known as the integer L-shaped method (section 4.2.2), there are few competitors of exact methods. Due to the combinatorial nature of the PTSP, solving the problem to optimality is in many cases computationally unaffordable in practice. As a result, most research has instead focused on heuristics.

[1] The book "The vehicle routing problem: Latest advances and challenges" by Golden et al. (2008) is probably the closest body of literature. However, their work is not limited to the PTSP, but instead mainly focuses on deterministic and dynamic routing problems. The article by Campbell and Thomas (2008a) (contained within Golden et al., 2008) provides a notable exception, but only focuses on approximate evaluation techniques (see section 4.4.4). Also, an article by Weiler et al. (2015) on a small selection of construction heuristic approaches for the PTSP appeared shortly after this chapter was written.

Early papers mainly focused on construction heuristics; e.g., savings-based procedures (section 4.3.1), nearest neighbour approaches (section 4.3.2), methods based on sorting (sections 4.3.4 and 4.3.3), and the Concorde TSP solver[2] (Bianchi, 2006). Several empirical tests on the performance of these construction heuristics have been conducted on subsets of these methods (Bertsimas et al., 1990; Bertsimas and Howell, 1993; Bertsimas et al., 1995; Bianchi et al., 2005; Bianchi, 2006; Bianchi and Gambardella, 2007; Balaprakash et al., 2010) – some of which suggest that Supersavings procedures perform better than other construction heuristics in most cases.

Interest has since gradually shifted towards improvement methods such as stochastic adaptations of k-opt (sections 4.4.1 and 4.4.3) and 1-shift (section 4.4.2). More recently, the trend shifted towards an almost exclusive domination by metaheuristics, both local search metaheuristics (section 4.5.1) and population search metaheuristics (section 4.5.2). The current state-of-the-art focuses on metaheuristic approaches such as ant colonly optimisation (section 4.5.2), particle swarm optimisation (section 4.5.2) and memetic algorithms (section 4.5.2). As a result of the shift from construction heuristics to improvement methods, the former category has been relatively under-researched in recent years.

The layout of this chapter follows the classification of those methods into these important solution categories: exact methods, heuristic methods and metaheuristic methods. There is also a section that briefly discusses the use of deterministic methods for the PTSP, although problem-inspecific methods are beyond the scope of this study. Even though metaheuristic methods can arguably be regarded a special subclass of heuristic methods, the extensive body of literature that appeared on metaheuristic methods for the PTSP alone made it deserve its own section.

A more general overview that classifies solution methods for the PTSP into a number of subcategories is shown in figure 4.1. The black boxes are the categories

[2]Note that, although regarded to be an exact method for the deterministic TSP, Concorde should be regarded as a heuristic for the PTSP due to its suboptimality in this context.

addressed by the five sections in this chapter. As this figure shows, the PTSP can be addressed using three different kind of approaches: Deterministic solution approaches, which originate from the deterministic TSP, stochastic approaches, which have been specifically designed to address the PTSP, and re-optimisation approaches, which do not target the a priori tour but the a posteriori tour. Note, however, that features from deterministic solution approaches are also often integrated – and in effect, are also part of the discussion – of many stochastic approaches.

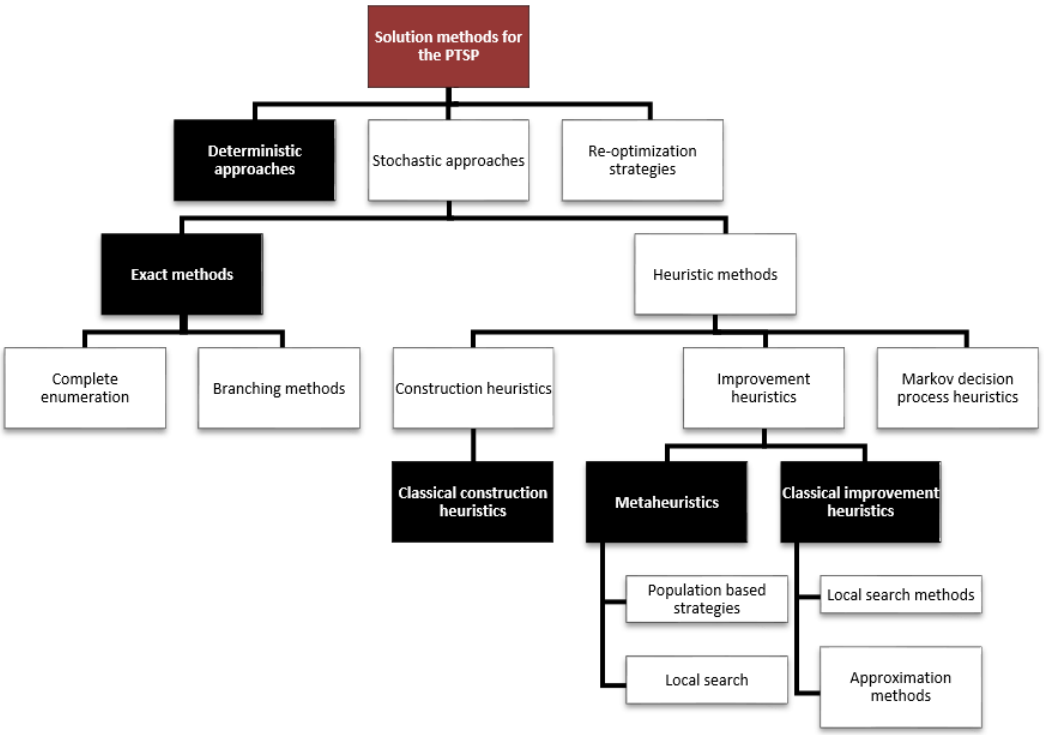


Figure 4.1: A global overview of the most important solution classes for the PTSP.

Stochastic approaches can be further subdivided into two main solution classes, namely exact methods (i.e., complete enumeration and branching methods) that solve the PTSP to optimality, and heuristic methods (i.e., construction heuristics, improvement heuristics and markov decision process heuristics) that solve the PTSP only approximately (cf. chapter 2). Metaheuristics can be regarded as a special subclass of heuristic methods that can incorporate both features from construction heuristics and improvement heuristics. More precisely, most

metaheuristics that were developed for the PTSP are solution frameworks that rely mostly on existing deterministic approaches or existing construction heuristics for the construction of the initial solution. These metaheuristics then proceed to applying framework-specific improvement actions to refurbish that initial solution. It is the latter improvement step of the metaheuristic framework that distinguishes it from other classical heuristics. Reasoning along this notion, the heuristic methods solution class has been separated into two categories concentrating on *construction* heuristics and *improvement* heuristics, whereby the latter category is split into *classical* improvement heuristics and metaheuristics.

An extensive literature review on solution methods for stochastic TSPs (including the PTSP) is the topic of chapter 2; as such, that is where I refer the reader to for an extensive discussion of the literature addressing the broader domain. This study may be regarded as an extensive review of the solution methods that are specific to the PTSP. This study should not only give a comprehensive overview of the literature that appeared on each method, but also aims to provide a systematic review and comparison of the methods in each solution category.

Furthermore, this study also discusses an in-depth analysis of construction heuristics for the PTSP in sections 4.3 and 4.4. For various reasons (substantiated in the appropriate sections), construction heuristics for the PTSP only received little attention. An analysis aims to shed light on the impact of characteristics, and in particular, customer clustering, on the empirical performance of construction heuristics. This empirical analysis is conducted both before and after applying a state-of-the-art improvement heuristic, so as to verify whether the observations about the impact of characteristic still remain valid. In addition, the analysis of construction heuristics after undergoing improvement aims to provide empirical evidence to the debate of whether starting from a good initial solution contributes to the quality of the final solution.

Section 4.1 treats the application of deterministic approaches for the PTSP.

Section 4.2 discusses a number of exact methods. Construction heuristics are the topic of section 4.3, followed by a discussion of improvement heuristics in section 4.4. Section 4.5 discusses a number of metaheuristics that have been proposed for the PTSP. Finally, section 4.6 concludes.

4.1 Deterministic approaches

Deterministic approaches are solution methods that originate from the literature on the deterministic TSP. Nonetheless, they can also be used to generate solutions for the PTSP – both to initialise an a priori tour, as well as to improve upon an existing solution. The use of deterministic approaches is not uncommon in the literature on PTSPs.

The purposes of applying deterministic approaches to PTSPs are manifolds: as initial solution generators which solutions serve as input to stochastic improvement methods, as benchmark methods for comparisons against stochastic solution methods, as an integral part of stochastic solution methods, or as hybrid extensions to stochastic (meta-)heuristic methods. Deterministic approaches that can be found in the literature on PTSPs include exact methods such as Concorde (Bianchi and Gambardella, 2007; Balaprakash et al., 2010), construction methods such as the Christofides heuristic, the savings algorithm and nearest neighbour algorithm (Jaillet, 1985; Bertsimas, 1988; Lu, 2001; Balaprakash, 2005), and improvement methods such as 1-shift and the k-opt interchange mechanisms (Jaillet, 1985; Bianchi et al., 2005; Weyland et al., 2009a; Liu, 2010).

A number of researchers (Bertsimas, 1988; Bianchi, 2006; Bianchi and Gambardella, 2007; Balaprakash et al., 2010) find that deterministic approaches perform reasonably well when the level of randomness of the problem is low (i.e., when the probabilities of the nodes are high and the number of stochastic customers is low). In fact, the Concorde algorithm (Applegate et al., 2006) is even found to outperform PTSP-specific heuristics when the level of stochasticity

is low ($p \geq 0.9$; see Balaprakash et al., 2010, for details). In most stochastic settings, however, deterministic approaches are outperformed by their probabilistic adaptations described in the next sections. This observation can be explained by the notion that good solution methods for the TSP can be arbitrary bad for the PTSP (Jaillet, 1985, see also section 4.3.4). Because of this reason, and to keep the discussion of this chapter tractable, the remainder of this study is concerned with probabilistic solution methods that were specifically designed for PTSPs. The reader is referred to Applegate et al. (2006) for a more thorough discussion of deterministic approaches. Deterministic approaches for the PTSP may nevertheless be worth a further study.

4.2 Exact methods

Exact methods are solution approaches that aim to find optimal solutions for the problem under consideration. In this sense, 'optimal' refers to a global (as opposed to a local) optimum: There is no better solution possible for the given configuration. In terms of accuracy, exact methods should therefore be preferred over heuristic solution approaches, which only attempt to find approximate solutions. However, the high complexity in which most exact methods operate restrict their use to only small to medium sized problems. To avoid long running times, researchers therefore often resort to heuristic solution procedures when the number of nodes is large.

Observe that the stochastic nature of the PTSP makes it impossible to derive an exact method which guarantees an optimal solution for every generated instance of the a posteriori tour. That is, an optimal solution of an a priori problem does not guarantee an optimal solution of every a posteriori generated instance because of the randomness that comes with the problem definition. Optimality of the a posteriori problem in terms of the a priori problem would require perfect foresight of the outcomes of these random variables. Therefore, although a method may be optimal in the a priori-sense, it may still not be optimal in a posteriori-sense. As

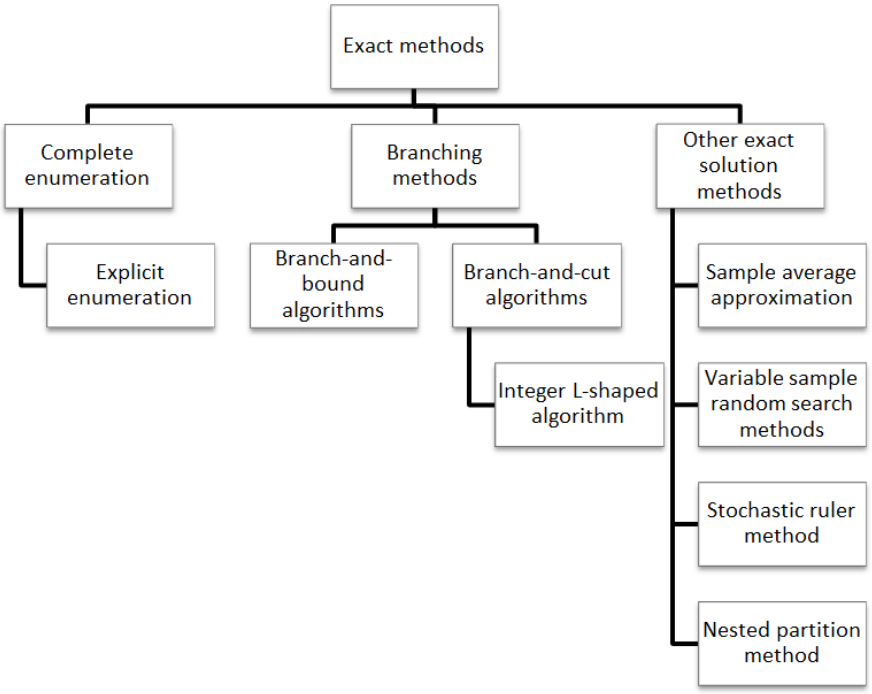


Figure 4.2: Overview of the most important exact methods for the PTSP classified according to their solution strategy.

a result, exact methods for the PTSP do not guarantee to always find a better a posteriori solution than a heuristic method in every generated case.

Nevertheless, the a priori solutions generated by exact methods are ascertained to outperform the a priori approximate solutions generated by heuristics on average, and, by the law of large numbers, are also certain to outperform the a posteriori performance of heuristics in the long run. The solution class of exact methods for PTSPs can be roughly divided into two main categories: Complete enumeration and branching algorithms (i.e., branch-and-bound, branch-and-cut). This chapter describes the two categories in the context of PTSPs. Figure 4.2 shows the classification of exact methods in greater detail.

4.2.1 Explicit enumeration

Explicit enumeration involves the evaluation of every single possible a priori tour in the graph, and returning the one with the minimum expected length. As one might

suspect, this is a very tedious task, and in practice often infeasible when the number of nodes becomes large. Jaillet (1985) shows that the computation of a single a priori tour can be done in $O(n^2)$ steps using the formula for the expected length (3.3). A complete enumerative approach that considers every possible outcome of the tour individually would take $O(n2^n)$ steps. Furthermore, the number of possible a priori tours in the TSP is equal to $(n-1)!$. Hence, the total computational effort involved in explicit enumeration of the PTSP is equal to $O(n!)$. This rules out explicit enumeration as an exact method from a practical perspective in most cases.

4.2.2 Branch-and-cut for the PTSP: the integer L-shaped method

In 1993, Laporte and Louveaux (1993) developed an exact solution procedure for generic stochastic integer programs with recourse known as the integer L-shaped method. The integer L-shaped method can essentially be regarded as the stochastic variant of the branch-and-cut approach. Laporte et al. (1994) adopted and extended the integer L-shaped method to solve the PTSP to optimality. Their method can be briefly summarised as follows.

Starting with the relaxed stochastic integer linear program described in section 3.4, initialise the search tree with a node 0 and an upper bound set to infinity ($\bar{z} = \infty$). Solve the problem with any appropriate linear programming approach of your choice (e.g. Simplex), and plug the resulting preliminary optimal solution $(\mathbf{x}, \gamma)$ into the relaxed objective function $d\mathbf{x}+\gamma$. Test the resulting objective value against the current upper bound $\bar{z}$, prune the node if it is higher, and check for any violated subtour-breaking constraints if it is lower. If a subtour-breaking constraint is indeed violated, the problem should be augmented in the usual fashion of branch-and-cut (see Laporte and Louveaux, 1993). Similarly, if an integrality constraint is violated, we should branch the node in the usual way of branch-and-cut. Continue repeating these steps until a node cannot be branched any further.

Up to this point, the algorithm is exactly the same as the branch-and-bound stage of the deterministic version of branch-and-cut. However, the next steps

are slightly different. That is, once a node is encountered that meets all the subtour-breaking and integrality constraints, we now compute the expected value for the excess a priori tour length $-\mathbb{E}_y[R(\mathbf{y}|\mathbf{x})]$ that results from taking recourse actions. This value can be obtained by plugging the current optimal values for $\mathbf{x}$ of the concerning node into the formula.[3]

Now, once the value for $-\mathbb{E}_y[R(\mathbf{y}|\mathbf{x})]$ has been obtained, compute the value of the *exact* objective function $z = d\mathbf{x} - \mathbb{E}_y[R(\mathbf{y}|\mathbf{x})]$ of the *unrelaxed* problem. If the newly obtained value z is smaller than the current upper bound $\bar{z}$ for that node, update the upper bound with the appropriate 'exact' value: $\bar{z} := z$. Additionally, if the approximation γ derived previously for the node turns out to be larger than (or equal to) the exact value for $-\mathbb{E}_y[R(\mathbf{y}|\mathbf{x})]$, prune that node and return to the branching process; otherwise, impose a so-called optimality cut for γ in terms of $\mathbb{E}_y[R(\mathbf{y}|\mathbf{x})]$ and the lower bound L and update the program. Return to the calculation of the relaxed objective function $d\mathbf{x} + \gamma$ using the newly obtained constraint from the optimality cut and continue until no more nodes are left.

In essence, the integer L-shaped method is a decomposition approach that treats the expected value resulting from the recourse action separately. By doing so, the branching procedure is only concerned with the straightforward calculation of the LP-relaxed problem without stochastic components. As a result, it does not have to worry about the more complex computation of the exact value for the expected length of the full a posteriori tour. The optimality cuts are imposed on the approximation of the expected value, and are expressed in terms of a repeatedly updated lower bound and exact value of the problem. This way, the number of computations of the exact expected value of the problem is kept to a bare minimum, i.e., computed only when strictly necessary, thus speeding up the entire process. Nevertheless, the integer L-shaped method has only been able to successfully solve homogeneous instances of up to c. 50 nodes with low randomness (Laporte et al.,

[3] See Laporte et al. (1994) for a detailed description on how to compute $\mathbb{E}_y[R(\mathbf{u}|\mathbf{x})]$.

1994). For more complex problems, the method has difficulties to converge, accompanied by increasingly large computation times.

4.2.3 Branch-and-bound

Another exact algorithm to solve the PTSP is based on the well-known branch-and-bound approach for solving deterministic mathematical programs. Observe that the branch-and-cut approach described in section 4.2.2 also relies on a branch-and-bound scheme for its initial phase, but that this phase does not differ from the traditional branch-and-bound method that is known to solve the deterministic version of the problem. Hence, we need to take additional efforts in order to make the branch-and-bound algorithm fit for solving the PTSP.

Rosenow (1999) develops an exact branch-and-bound algorithm for the PTSP. Continuing along the same lines as the deterministic variant of branch-and-bound, her approach requires branching on all possible subproblems in every subsequent step of the algorithm, and selecting the node with the lowest bound to determine the next node to branch on. More specifically, for every step i of the algorithms there are $n-i$ nodes to branch on, one corresponding to every move of going from one node in the tour to another node not yet included in the tour. The associated lower bounds of each of these moves can be calculated in a recursive way using a formula derived from the expected length of the tour (3.3). The node that attains the lowest bound of all nodes in that step is selected as the next node to branch on. This process is continued for all steps of the algorithm until the tree hits the last node: The lower bound of this node can be shown to be exactly equal to the expected length of the a priori tour constructed through these nodes. Rosenow (1999) notes that the lower bounds can be further improved by enhancing their calculations within the branching scheme by an additional reduced minimum spanning tree bound.

Despite its guarantee to find an optimal solution for any number of nodes, the branch-and-bound method did not deserve the same amount of attention

as the integer L-shaped method. This can be attributed to the fact that the integer L-shaped method of Laporte and Louveaux (1993) can be adapted to a wide range of stochastic programming problems including the PTSP, whereas the branch-and-bound algorithm by Rosenow (1999) is designed to deal specifically with the PTSP alone. But more importantly, the branch-and-bound approach suggested by Rosenow (1999) takes a total of $(n-1)!$ evaluations. In effect, it attains the same computational complexity as the explicit enumeration strategy. To state it in the words of Rosenow herself, *"obviously this is absurd, no one can use such a strategy"* (Rosenow, 1999, p. 171).

Mahfoudh et al. (2015) and Amar et al. (2017) recently propose alternative branch-and-bound implementations, with mechanisms that operate analogous to the method describe above. Supported by merely thin experimental evidence, Mahfoudh et al. (2015) claim that their algorithm is only able to run in $O(n^2)$. They apply their algorithm to a PTSP instance of 6 nodes, and report findings in line with the results reported by Laporte et al. (1994). Amar et al. (2017) report successful applications of their implementation for instances of up to 18 nodes. The performance of their proposed implementation, however, seems to explode for instances with more than 17 nodes. The lack of empirical evidence for instances greater than 18 nodes, along with only poor theoretical evidence of the computational time, do not offer substantial support for the practical usefulness of these algorithms.

4.2.4 A brief note on dynamic programming for the PTSP

For many deterministic variants of routing problems, including the TSP, an exact algorithm based on dynamic programming is among the biggest competitors for solving it to optimality. However, Jaillet (1985) shows that dynamic programming approaches cannot be used to solve the PTSP to optimality.

More precisely, a dynamic programming approach requires a recursive expression that meets Bellman's principle of optimality (Bellman, 1957), which

in turn requires the ability to decompose the problem into different stages. Jaillet (1985) points out that a decomposition of the PTSP does not guarantee that in any stage of the decomposition the solution of the subproblem is also optimal for the problem as a whole. Therefore, an optimal solution for the PTSP can only be obtained by considering the whole problem. Consequently, any stage-decomposition approach – including dynamic programming – cannot be used to solve the PTSP to optimality.

4.2.5 Other exact solution methods

Balaprakash (2005) mentions a number of alternative exact methods for the PTSP including sample average approximation, variable sample random search methods, the stochastic ruler method and the nested partition method. Some of these methods (such as sample average approximation) only generate a sample of a posteriori tours and solve these instances to optimality using exact deterministic methods rather than exact stochastic methods, and are therefore not necessarily also optimal with respect to the a priori stochastic problem. To the best of my knowledge, none of the other approaches have ever been applied to successfully solve the PTSP to optimality, and are therefore excluded from this study. An example of sample average approximation as part of (meta)heuristic methods (e.g. 2.5-opt-EEs and 2.5-opt-EEais) can be found in sections 4.3 and 4.5; the conceptual framework of sample average approximation, which is basically an approximate evaluative approach, is outlined in more detail in section 4.4.4.

As a final note, observe that there is a simple alternative strategy to the two phase SPR design of the PTSP that *does* guarantee optimality at every instance of the problem. This can be accomplished by performing a complete re-optimisation of the problem after the presence of the customers has been revealed: the so-called re-optimisation strategy (Jaillet, 1985). Instead of solving the PTSP in terms of the a priori objective function (3.3), re-optimisation simply solves the (deterministic) TSP after the customers have been revealed. Observe from equation (3.2) that the number of outcomes of the problem is exponential, which makes a complete

analysis of every possible solution beforehand practically impossible. Moreover, the strategy of daily re-optimisation obviously comes at much higher costs than any of the solution approaches discussed in this chapter, and may also be subject to hidden costs (Benton and Rossetti, 1992). Several analyses of the re-optimisation strategy for the PTSP in comparison with a priori evaluation are carried out by Chervi (1990), Bertsimas (1988), Bertsimas et al. (1990) and Bertsimas and Howell (1993). Re-optimisation strategies are beyond the scope of this study.

4.2.6 A comparison of exact methods

Method	Strategy	Key characteristic(s)	Limitations	Performance
Integer L-shaped method	Branch-and-cut	⋆ Solves the PTSP to optimality in a finite number of steps ⋆ Best method we have for solving the PTSP to optimality	⋆ Empirical tests show that it can only solve small to medium sized problem instances (up to c. 30-50 nodes) ⋆ Only performs well when the level of randomness in the problem is low (i.e. low number of white nodes, high node probabilities)	Slow
Explicit enumeration	Complete enumeration	⋆ Solves the PTSP to optimality	⋆ Slow and exhaustive ⋆ Impractical; Can only be used for a (very) small number of nodes	Extremely slow
Branch-and-bound	Branch-and-bound	⋆ Solves the PTSP to optimality	⋆ Slow and exhaustive ⋆ Impractical; Can only be used for a (very) small number of nodes	Moderate* to extremely slow

Table 4.1: A comparison of various characteristics of exact methods for the PTSP. Computational performance markings: Constant $O(1)$: Extremely fast; Linear $O(n)$: Very fast; Loglinear $O(n \log n)$: fast; Quadratic $O(n^2)$: Moderate; Polynomial $O(n^c)$: slow; Exponential $O(c^n)$: very slow; Factorial $O(n!)$: Extremely slow. * The claim for $O(n^2)$ computation time is only supported by thin evidence from a single and very limited empirical experiment.

Table 4.1 provides a comparison of exact solution approaches for the PTSP. The integer L-shaped method by Laporte et al. (1994) clearly stands out as the current best (and only viable) approach of solving the PTSP to optimality. It is important to realise, however, that many exact stochastic methods are left unexplored in the context of the PTSP, as already pointed out in section 4.2.5. Some of these methods offer potential for even better results, and may be worth a further study.

4.3 Construction heuristics

The study of heuristics is a relatively young area of research. Its origins are generally believed to date back to the 1940s, when Pólya (1945) introduced procedures that simplified the search for a solution to mathematical problems. Although earlier adaptations of the concepts embedded by heuristics were already common in other fields of science prior to the 1940s (Hertwig and Pachur, 2015), the introduction of linear programming in the 1950s led to the actual breakthrough of heuristics in the field of combinatorial optimisation. Ever since, heuristics gained widespread attention among mathematicians and computer scientists.

Heuristics can roughly be divided into two main categories, namely construction heuristics (Figure 4.3) and improvement heuristics. Whereas construction heuristics are used to build a tour, improvement heuristics can be used to improve an existing tour. For this reason, these two subcategories often bear very different characteristics. Construction heuristics are the topic of this section. Improvement heuristics are discussed in the next section.

4.3.1 Savings algorithms

The savings algorithms that are designed for the PTSP originate from the Clarke and Wright (1964) savings algorithm. The concept of the savings algorithm is based on iteratively adding new customers to a subtour based on a selection strategy known as the 'savings criterion', until all customers in the graph have been added. The savings criterion entails inserting a candidate customer in a way such that the cost of inserting that customer into the existing tour is minimised.

The Supersavings-based procedures all share a common basic feature, namely the merging of subtours based on Supersavings, or expected cost figures (Jaillet, 1985). Supersavings-based procedures start by forming subtours of a given size 3; that is, subtours that leave the depot, visit 2 nodes, and get back to the depot. Then, for each pair of subtours, these procedures consider the possible ways of merging

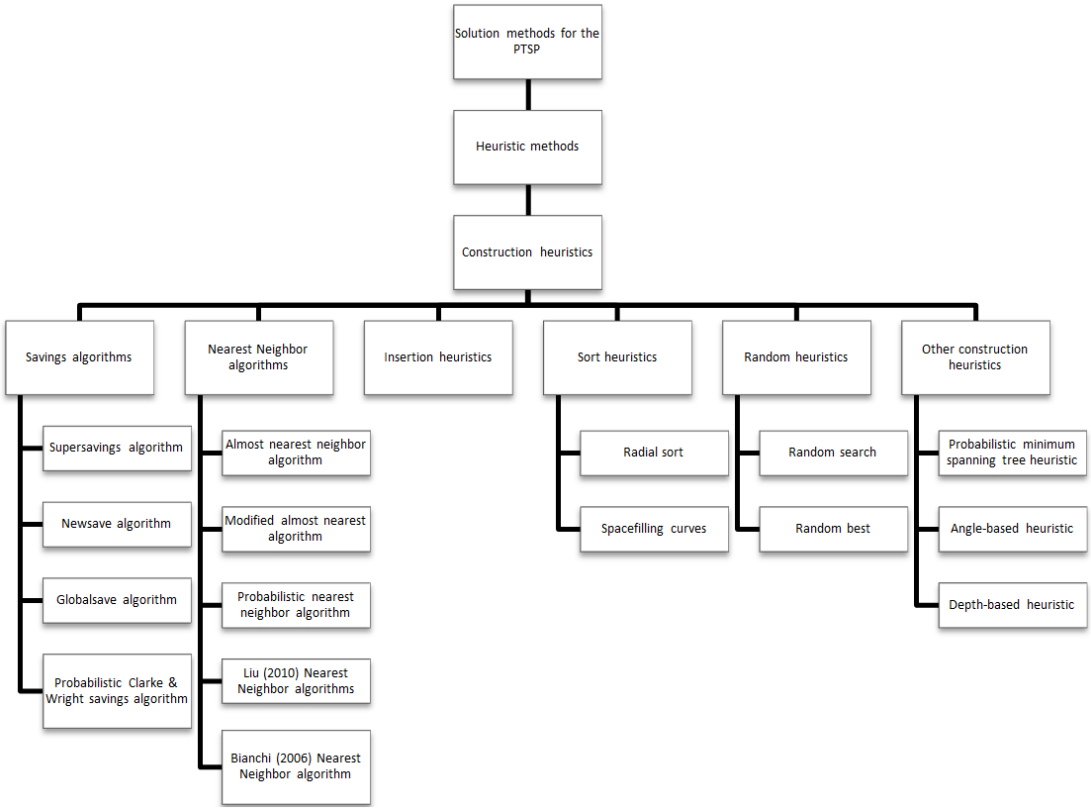
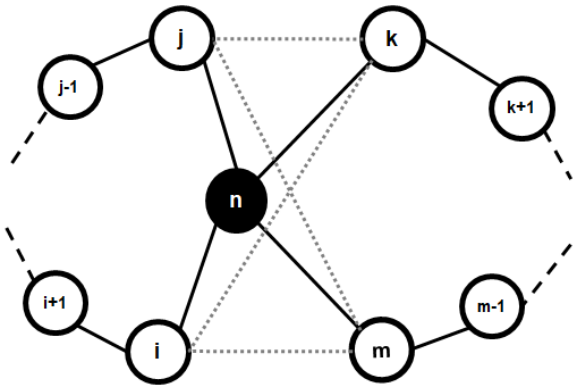
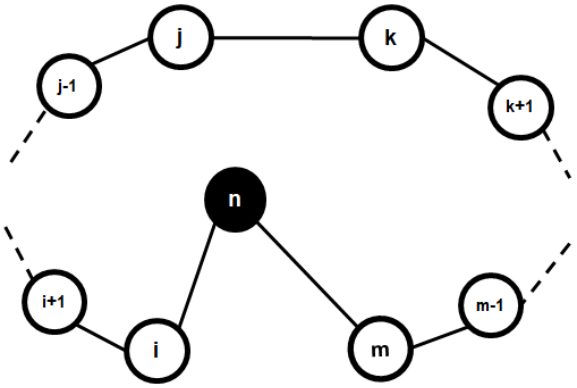


Figure 4.3: The most important construction heuristics for the PTSP classified according to their solution strategy.

any two subtours by connecting the two customers, say i and j, that are the end nodes of the subtour: The last customers of the subtours before the tour goes back to the depot. Alternatively, the end points can also be the first customers that appear in the subtour straight after leaving the depot, or a combination of a first and a last customer of both subtours. Therefore, there are at most four possible ways of merging any pair of subtours -– see Figure 4.4 for an illustrative graph of the merging operation. The expected savings associated with each merge possibility are calculated and stored in a list in descending order. The Supersavings-based procedure starts merging a pair of subtours from the current solution that results in the largest possible saving, and reduces the savings list to reflect the new (partial) solution. It continues to form larger subtours until a complete tour along all nodes is formed.



(a) Two subtours and their candidate merging points.



(b) Subtour after merging nodes j and k corresponding to the largest expected savings.

Figure 4.4: Illustration of a subtour merging operation

The variants of the savings algorithm for the PTSP differ from the TSP in the sense that they rely on the expected savings, rather than the 'usual' deterministic savings. For example, in the figure above, connecting customers i and j in a deterministic sense would result in a direct saving of $s_{ij} = d_{in} + d_{nj} - d_{ij}$: The costs of not having to return to the depot n after visiting customers i and j, diminished by the extra costs of going straight from customer i to j. In contrast, the expected savings that result from the PTSP analysis are equal to (Jaillet, 1985): $\mathbb{E}[s_{ij}] = p^2 s_{ij} + p^2(1-p)(s_{i-1,j} + s_{i,j+1})$, where s_{ij} takes the same form as above and the node probabilities are assumed to be homogeneous for simplicity. Observe that the expected savings do not only take the node probabilities p of customers i and j of the immediate savings s_{ij} into account, but also rely on the presence and associated savings of the other nodes in the subtour $s_{i-1,j}$ and $s_{i,j+1}$. The reason is very simple: If either of the nodes i and j would not be present, the direct savings s_{ij}

of going straight from node i to node j cannot be realised. Consequently, we need to rely on the savings $s_{i-1,j}$ of going from node $i-1$ to node j (in case node i is absent with probability $1-p$ and the other nodes are both present with probabilities p) or the savings of going from node i to node $j+1$ (in case node j is absent with probability $1-p$ and the other nodes are both present with probabilities p) instead.[4] Jaillet (1985) referred to these expected savings as *supersavings*.

Supersavings algorithm

The savings algorithms that are designed for the PTSP come in a number of different flavours, but all rely on the same principles described above. Jaillet (1985) was the first to propose a savings algorithm for the PTSP, which he labelled the supersavings algorithm. It is based on forming increasingly larger initial subtours as a starting point for the eventual merging procedure. Observe that the case of initial subtours of size 3 can easily be amended to subtours of any size g, for $0 \leq g \leq n-1$, using the (deterministic) Clarke and Wright (1964) savings algorithm. For every subtour of size g, the calculation of the supersavings associated with joining two end nodes should be adjusted accordingly (for details, see Jaillet, 1985).

Jaillet (1985) proposes to repeatedly construct initial subtours of size g up to a certain point $g^* \leq n-1$. For each of these starting points, calculate the supersavings and merge subtours until only one large tour remains. The "best" (sic) out of the g^*+1 tours that are formed should be returned as the choice for the a priori tour.

Newsave and globalsave

Jézéquel (1985) was the first to implement the supersavings algorithm by Jaillet (1985) and to perform an empirical analysis. Furthermore, he made two

[4] Note that we do not have to account for the event of both customers being absent, as the savings $s_{i-1,j+1}$ associated with the link of end nodes $i-1$ and $j+1$ should be captured by another savings calculation.

suggestions for alterations of the supersavings approach, which are labelled the newsave and globalsave algorithms.

The newsave algorithm is essentially the same as the supersavings algorithm, but instead of building larger tours by merging any two candidate subtours, only one subtour is selected as the 'base tour'. This base tour is then grown by repeatedly merging it with the other subtours in the graph.

The globalsave algorithm, on the other hand, does not attempt to solve the problem for any specific value of the (homogeneous) node probability, but instead solves the problem for a whole range of different node probabilities. A tour is constructed using a good deterministic TSP algorithm for each probability, and subsequently fed into the newsave algorithm. If, for any node probability, the resulting tour from the newsave algorithm has a smaller expected length than the expected length of the tour already associated with that node probability, the TSP tour is replaced by the newsave tour. This process is repeated for every TSP tour in the array of probabilities such that a whole range of tours emerges, one for each evaluated probability.

Probabilistic Clarke and Wright (1964) savings algorithm

Another extension to the supersavings algorithm of Jaillet (1985) was made by Rossi and Gavioli (1987) and is called the probabilistic Clarke and Wright (1964) savings algorithm (PCW). Unlike the supersavings algorithm, Rossi and Gavioli (1987) fix the point of initial subtour formations to $g^* = 2$. Additionally, instead of using a fixed list of supersavings as in Jaillet (1985) and Jézéquel (1985), their algorithm forces the savings list to update after subtours have been merged. By also accounting for the new customers in the larger subtours in the expected length of another possible merging through (3.4) the accuracy of the algorithm is increased.

4.3.2 Nearest neighbour algorithms

Nearest neighbour algorithms are algorithms which repeatedly add new customers to a path based on the nearest neighbour principle, until a complete tour is formed. The nearest neighbour principle entails adding the customer to the path who is currently the closest to either of the two end points. Although the algorithm is relatively fast, its results are often outperformed by other construction heuristics (Jézéquel, 1985; Rossi and Gavioli, 1987).

Almost nearest neighbour algorithm (type I)

The first nearest neighbour algorithm developed for the PTSP was proposed by Jaillet (1985) and is referred to as the Almost Nearest Neighbour Algorithm (abbreviated ANNA or ANNA0) or Probabilistic Nearest Neighbour (PNN or PNN1) algorithm. It starts out from a single black node – the depot – and selects the closest customer which is then connected to the depot. This customer is added based on the minimisation of the increase in path length in deterministic sense; that is, it is not concerned with any of the customer probabilities. The next nearest customer is subsequently also added to the path by connecting it to the previously selected customer, but now based on the expected length that is added to the tour rather than the deterministic length. More precisely, it selects the next customer based on an approximation of the increase in expected length using the weight-form notation (3.6), up to a certain level $h^* \leq n-1$, such that $h = \{1, 2, \ldots, h^*\}$ and $h^* \leq n-1$. Note that this is similar to the optimisation level g^* that is employed for the savings algorithms (section 4.3.1). This process is repeated until no more nodes can be added to the path. The last node of the path is then connected back to the depot such that a tour is formed. In the last stage, the best out of the h^* tours is selected, and returned as the final tour of the algorithm.

Almost nearest neighbour algorithm (type II)

Slight modifications to the ANNA0 algorithm were proposed by Jaillet (1985) and

subsequently tested in an empirical setting by Jézéquel (1985). This modification consists of first growing two trees of size k out of the depot, where the customers of these two trees are both added based on the increase in expected length in deterministic sense. Jaillet (1985) argues that the customers that are closer to the depot are weighted more heavily by the expected length in weight-form notation (3.6), and should be minimised first in order to obtain a good initial solution for the path. The algorithm is labelled ANNA1, where the suffix 1 is added as a way of distinguishing it from the original heuristic, ANNA0.

Probabilistic nearest neighbour algorithm

Rossi and Gavioli (1987) propose a minor correction to the formula for the increase in expected length employed by the original ANNA0/PNN1 heuristic of Jaillet (1985), such that it also accommodates for the possibility that the path is closed after a newly added customer. They label the heuristic PNN2 and test it empirically against the ANNA0/PNN1 heuristic. Despite the correction, the PNN2 heuristic does not outperform PNN1 because it tends to put extra weight on the nodes closer to the depot, causing the algorithm to perform less accurate.

Liu (2010) and Bianchi (2006) nearest neighbour algorithms

More recent probabilistic adaptations of the deterministic nearest neighbour algorithm include the nearest neighbour algorithms by Liu (2010), who investigate initial solution generators for genetic algorithms, and Bianchi (2006), who investigates the nearest neighbour algorithm as an initial solution generator for the ant colony optimisation metaheuristic. Both rely on deterministic implementations of the nearest neighbour principle, but account for the stochastic elements in the problem by giving it a slightly random twist; a rough outline of both methods is as follows.

Liu (2010) proposes two nearest neighbour algorithms, type 1 and type 2. Starting from a common depot, his type 1 nearest neighbour algorithm adds

customers in a deterministic way both to the beginning and the end of the tour. That is, it grows two trees out of the depot, as in ANNA1. Unlike the deterministic algorithm, however, the nearest and second nearest neighbours are randomly selected for insertion with probabilities 0.9 and 0.1, respectively. When all customers are added to either of the two paths, the end points are connected to form a tour. The Liu (2010) type 2 nearest neighbour algorithm adds nearest and second-nearest customers in the same random way as the type 1 algorithm, but the tour is now grown from two ends relative to a randomly selected point from the centre of the tour. This process is continued until both ends reach the depot.

Finally, the nearest neighbour implementation of Bianchi (2006) constructs n tours using the deterministic nearest neighbour algorithm, each tour starting from a different customer $i = \{1, \ldots, n\}$ in the graph. After every possible tour has been constructed, the algorithm selects the tour that yields the minimum expected length according to (3.3). For this reason, I refer to this algorithm to iterated nearest neighbour (iterated NN) in the remainder of this chapter.

4.3.3 Spacefilling curves

Spacefilling curves build tours by projecting the plane of the graph onto the unit square $[0,1]^2$, and dividing the square into ever converging smaller regions. Each node within the graph is assigned to a specific region within the unit square, after which each region (and as such, also the nodes within each region) are visited in a predetermined sequence. Spacefilling curves rely on the notion that hyperdimensional points within a hypercube $[0,1]^d$ can be mapped on the unit interval $[0,1]$ using Cantor functions. The spacefilling curve that is typically selected to construct tours for the PTSP is a variant of the Sierpinski curve (Sierpiński, 1912). It was originally developed by Bartholdi and Platzman (1982) for the TSP, and introduced to the PTSP by Bertsimas (1988). Despite its original development for the deterministic version of the TSP, spacefilling curves also perform relatively well for the PTSP.

The spacefilling curve can be described as follows. Given a graph G of nodes with coordinates (x_i, y_i), plot the graph onto a two-dimensional Euclidean plane. Next, scale the plane to the unit square $[0,1]^2$ according to the relative distances of the nodes in the plane. This square forms the base of the problem.

The unit square is subsequently divided into four equally sized regions, also known as quadrants. Each of these quadrants is visited in a fixed sequence within the larger square, starting from the bottom left hand square (quadrant 0), to the top left hand corner (quadrant 1), followed by the top right hand corner (quadrant 2) and finally the bottom right hand corner (quadrant 3). Apart from the sequence in which these quadrants are visited, each quadrant is also characterised by a rotation, i.e. a direction in which it is visited. Starting from quadrant 0, the nodes within this quadrant are visited in a clockwise direction. After this step, the reference point of the turning direction is turned by 90°(clockwise), and the algorithm continues on to the next quadrant. This process is illustrated by figure 4.5.

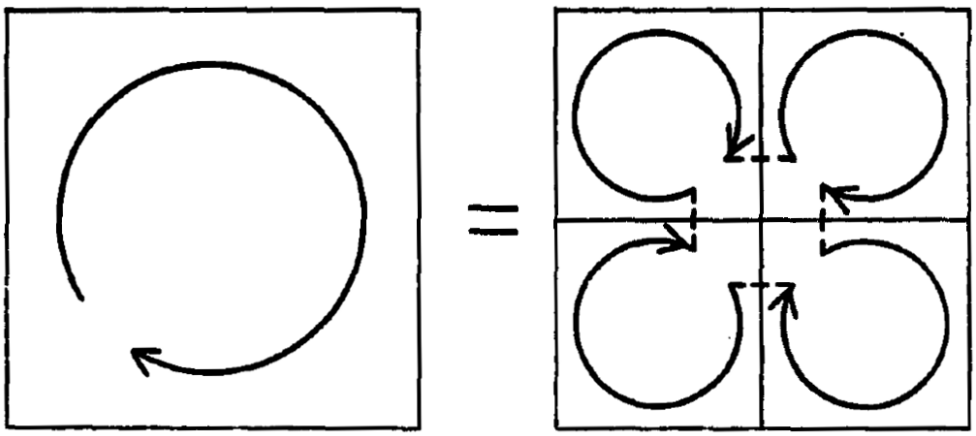


Figure 4.5: Quadrants and rotation of spacefilling curves. Adapted from *An $O(n \log n)$ planar travelling salesman heuristic based on spacefilling curves* by Bartholdi and Platzman (1982), Operations Research Letters 1(4), p. 122. Copyright 1982 by Elsevier, North-Holland.

The quadrants into the unit subsquare can be subdivided into even smaller subsquares, each with their own position within the larger square and rotation

relative to the larger square. Repeated division of each subsquare into four smaller squares in the same way as we did for the larger square is a recursive process, and enhances the accuracy of the overall algorithm. Each node within the graph is assigned to a subsquare using the same recursion, which, for example, may look like this: $q = (3, 0, 2, 1)$. It tells us to which of the four quadrants (0, 1, 2 or 3) the node is assigned in the largest unit square (namely quadrant 3, the bottom right hand corner), then to which of the four smaller quadrants within the larger subsquare it is assigned (namely quadrant 0, the bottom left hand corner), then the quadrant within that smaller subsquare, and so on and so forth. This continues on to the final level of convergence (in our example, a convergence level of 4), which indicates the overall precision of the algorithm. The higher the level of convergence, the more subsquares are used for the largest unit square, thus increasing the accuracy of the actual descriptive location of a customer within the larger unit square. The repeated division of squares into smaller and smaller subsquares is illustrated in figure 4.6.

After this iterative process, the customers are visited in the sequence that was determined by the direction for each of the subsquares as outlined above. This is accomplished by expressing the customer locations (x_i, y_i) in terms of the quadrants as previously illustrated, and subsequently mapping the descriptive customer locations onto the unit interval. Sorting the mappings in ascending order and visiting the customers according to their value in the sorted list yields a path along all customers. If more than one customer is assigned to the same subsquare (i.e. they both have the same value in the list), either the precision of the algorithm should be increased or the customers will be visited in random order within that subsquare. After all customers are visited, the path is closed by connecting the last customer back to the first customer in the path. The resulting tour may look something like the tour in figure 4.7. The exact formula used for the recursion is described in full detail by Bartholdi and Platzman (1982).

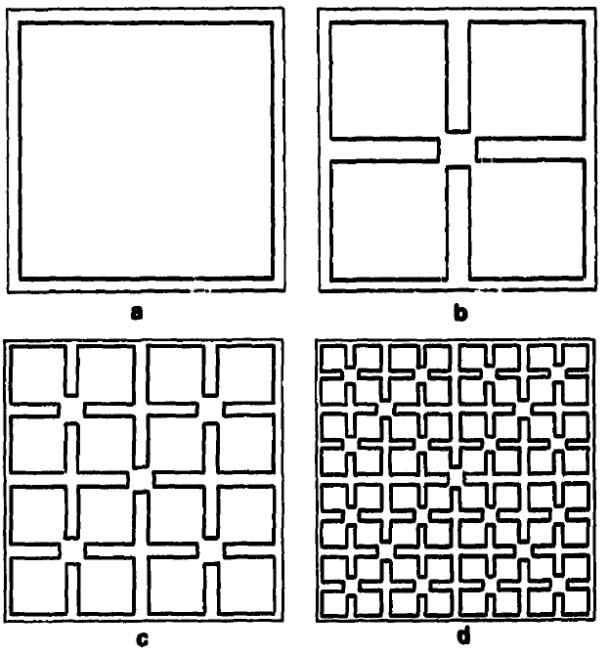


Figure 4.6: Increasing convergence of a spacefilling curve. Adapted from *An $O(n \log n)$ planar travelling salesman heuristic based on spacefilling curves* by Bartholdi and Platzman (1982), Operations Research Letters 1(4), p. 123. Copyright 1982 by Elsevier, North-Holland.

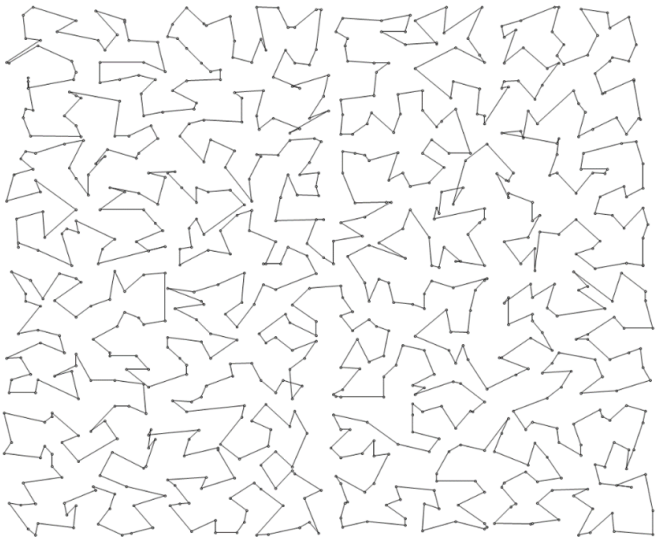


Figure 4.7: Illustration of a tour generated by the spacefilling curve heuristic for a TSPLIB instance consisting of 783 nodes (rat783).

4.3.4 Radial sort heuristic

The radial sort heuristic, or simply radial sort, sorts the customers according to their relative angle with respect to the center of mass of the plane, and then visits the customers in increasing angle. The first application of the radial sort heuristic to the PTSP dates back to Bertsimas (1988), called *angular sort*, and is likely to be inspired by Jaillet (1985) illustration that deterministic TSP solutions can be arbitrarily bad solutions for the PTSP (see figure 4.8). Using a dodecagon outline of 24 nodes, Jaillet (1985) shows that the expected length of tour (a) is up to 31% larger than the expected length of tour (b) in Figure 4.8 for the PTSP with homogeneous node probabilities equal to $p = 0.5$.

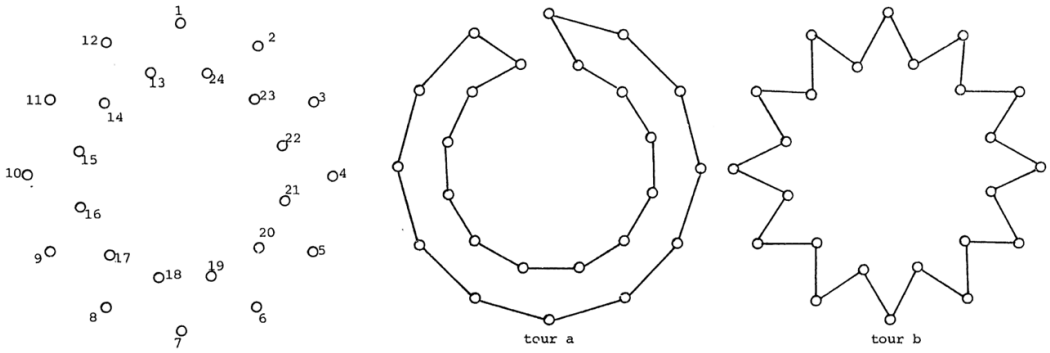


Figure 4.8: Good TSP versus PTSP tours for a dodecagon outline of 24 nodes. Adapted from *Probabilistic traveling salesman problems* by Jaillet (1985), Ph.D. thesis, MIT, Operations Research Center Technical Report 185, p. 33. Copyright 1985 by P.A. Jaillet.

In short, a center of mass is calculated as a reference point for sorting the customers in the graph by averaging over all x_i and y_i coordinates of the customers. The customers in the graph are connected to each other by increasing angle with respect to the center of mass using the arctangent function. That is, the first customer in the tour will be the customer that has the smallest angle with respect to an imaginary radial line that departs from the center of mass with degree 0. The second customer in the tour will be the next customer (excluding the first customer that is already in the tour) that forms the smallest angle with the radial line, and

is connected by a line segment to the first customer. This process is continued in a clockwise direction until all customers are connected to each other, forming a Hamiltonian path. The path is closed by connecting the last customer in the path (the one that has the largest angle with respect to the center of mass) back to the first customer (that has the smallest angle with respect to the center of mass). As an illustration, figure 4.9 shows the tour obtained by connecting 2424 cities in the Netherlands using this heuristic.

The radial sort heuristic bears some similarities to the spacefilling curve heuristic, in the sense that both heuristics do not require any computation of the expected length of a tour in order to form a tour along the customers in the graph. Furthermore, the tours that are formed using both heuristics rely predominantly on the x and y coordinates of the customers in the plane.

As Bertsimas (1988) already points out, the general performance of the radial sort heuristic relies strongly on the probability with which the customers are present. Radial sort only works well for a very small range of node probabilities $p \lessapprox 0.1$ and depends on the dispersion of the customers in the plane, as optimal solutions to the PTSP only tend to be *approximately* star-shaped for low probabilities, and not *exactly* star-shaped.

4.3.5 Random tour algorithms

The random tour algorithm, or random search algorithm generates a random tour. Starting out from a random node in the graph, this is achieved this by randomly selecting new customers from the subset of customers that are not already included in the path. The randomly selected customer is appended to the end of the path and the process is repeated until no more customers remain. At the final step, the first customer is added back to the end of the path such that a complete tour is formed.

Random best heuristic

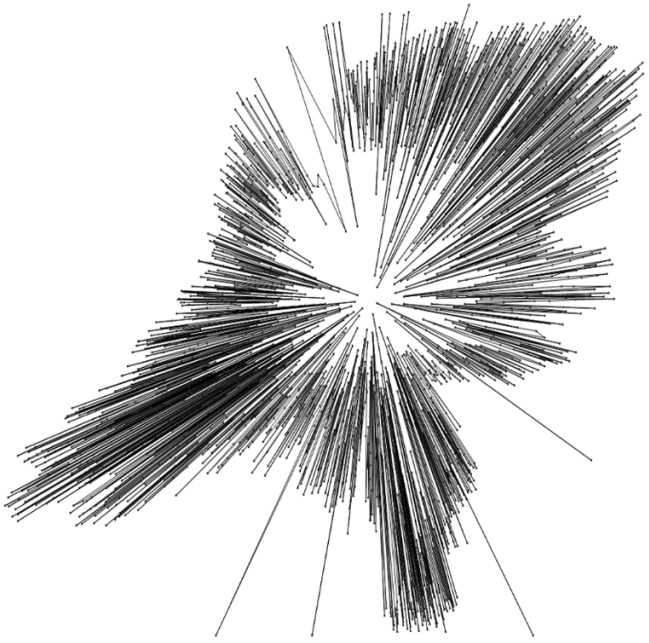


Figure 4.9: Tour along 2424 cities in the Netherlands constructed by the radial sort heuristic.

An alternative version of the random tour algorithm is known as the random best heuristic (Bianchi et al., 2002a). It repeatedly performs the random tour algorithm above to generate a set of random tours instead of only one. It then carries on to calculating the expected lengths of all these tours, and returns the one that has the smallest expected length.

Although random tour construction procedures are extremely fast, their performance is generally very poor. For that reason, random tours are rarely used as generators of a final solution and benchmarking competing methods. Surprisingly, it is not uncommon to use a randomly generated tour as an initial solution for an improvement method (e.g. Bertsimas and Howell, 1993; Bianchi, 2006; Liu, 2010).

4.3.6 Insertion heuristics

Insertion heuristics insert new customers into a subtour based on some insertion criterion. For example, nearest insertion inserts the nearest customer (i.e. the

customer having the lowest arc costs to any of the customers already in the subtour) into the subtour such that the increase in length is minimised. Farthest insertion, on the other hand, inserts customers that are the farthest away from any customer currently already in the subtour, in a way such that the costs associated with inserting the new customer into the existing subtour is minimised. Other insertion heuristics include the Mole and Jameson (1976) sequential insertion heuristic, the Christofides (1976) heuristic, the max-min insertion procedure and the T-Set first insertion heuristics (Rosenkrantz et al., 1974).

Although these heuristics can be easily adapted to exploit the additional expected length formulas also used by (for example) the savings algorithms, attempts to extend these heuristics to the PTSP case were only made recently (Weiler et al., 2015). Their results appeared shortly after completing this chapter. A more formal and extensive variant was also proposed earlier by Tang and Miller-Hooks (2007). They adapt three insertion heuristics for the generalised travelling salesman problem with stochastic customers (PGTSP) based on the increase in expected length rather than deterministic length. These heuristics include the smallest insertion heuristic, the T-set first insertion heuristic, and the max-min insertion procedure. Because of the conceptual differences between the PGTSP and the PTSP, these heuristics cannot be readily applied to the PTSP and are therefore omitted here. Moreover, literature on the PTSP, such as Bertsimas and Howell (1993), Balaprakash (2005) and Bianchi (2006), merely report deterministic variants of insertion heuristics.

4.3.7 Other construction heuristics

Other construction heuristics for the PTSP include a range of less well-known probabilistic adaptations of deterministic TSP algorithms, as well as purpose-designed construction heuristics. Some of these heuristics have only been discussed theoretically, and have not been exposed to any empirical testing (e.g. the probabilistic minimum spanning tree heuristic by Jaillet 1985; the partitioning heuristic by Bertsimas 1988; and the cluster and sweep heuristic by

Lu 2001). Others have been merely part of another main framework to serve as a construction phase and have not been extensively tested separately (e.g. the pure greedy algorithm for ENS-GRASP by Marinakis et al. 2008; and the depth-based and angle-based heuristics for ACO by Branke and Guntsch 2004).

Because of the limited occurrences of these heuristics in the PTSP literature and to retain the focus of this chapter to only the most important heuristics, I omit their descriptions here – please see the original papers for an elaborate discussion. An investigation of these heuristics in an empirical setting may be worth a future study.

4.3.8 A comparison of construction heuristics

Table 4.2 provides a comparison of various features of construction heuristics. The supersavings-based heuristics seem to provide relatively the best results of all construction heuristics.

It is important to realise that, with a few minor exceptions, almost all significant contributions to the area of construction methods for the PTSP have taken place in the 1980s and 1990s. No major new methods or alterations were proposed afterwards, such that many of the construction heuristics are not up-to-date with respect to the most recent developments and alternative implementations of the PTSP in the literature.

Perhaps as a consequence, also most of the classical improvement heuristics discussed in the next section use deterministic construction heuristics, precalculated solutions from deterministic TSPs, or random tour generators in order to obtain initial solutions rather than stochastic construction methods. The only construction heuristic that is still frequently mentioned as an initial solution generator for improvement methods is the spacefilling curve heuristic, and random search. This is an interesting observation because spacefilling curves and random search do not rely on the objective function, unlike most of the other construction heuristics. But an extensive comparative analysis of

Method	Strategy	Key characteristic(s)	Limitations	CP	EP
Supersavings algorithms	Clarke and Wright (1964) savings approach	⋆ Probabilistic adaptation of the traditional savings algorithm ⋆ Hands-on; conceptually simple, robust ⋆ Well-tried method for TSP-related problems	⋆ Requires the construction of multiple tours (i.e. $g^* + 1$) instead of only one ⋆ Requires repeated calculation of the a priori tour length; can be slow for large problems	Slow	1
Probabilistic/Almost nearest neighbour	Nearest neighbour principle	⋆ Probabilistic adaptation of the nearest neighbour algorithm ⋆ Simple concept, relatively fast, and can be effective for some scenarios	⋆ Accuracy strongly depends on the dispersion of the nodes in the graph and the customer probabilities	Slow	3
Spacefilling curves	Sierpinski curves	⋆ Adaptation of spacefilling curves to the PTSP ⋆ Operates independent of the expected length of the tour ⋆ Fast ⋆ Works surprisingly well	⋆ Independence w.r.t. the objective function does not guarantee (convergence to) optimality ⋆ Balance between computational effort and accuracy strongly depends on the customer dispersion; Results can be poor if customers are clustered	Fast	2
Radial sort	Radial sorting	⋆ Adaptation of radial sort for the PTSP ⋆ Operates independent of the expected length of the tour ⋆ Fast	⋆ Independence w.r.t. the objective function does not guarantee (convergence to) optimality ⋆ Quality of the solutions depends strongly on the probability of the nodes and node dispersion; works only well for small node probabilities	Very fast	4
Random search		⋆ Generates a random tour ⋆ Fast	⋆ Solutions can be arbitrarily bad	Extremely fast	5

Table 4.2: A comparison of heuristic methods for the PTSP. Some properties may also apply to extensions of methods, such as the properties for Newsave as an extension to the Supersavings method, etc. CP: Computational performance; EP: (relative) empirical performance. Computational performance markings: Constant $O(1)$: Extremely fast; Linear $O(n)$: Very fast; Loglinear $O(n \log n)$: fast; Quadratic $O(n^2)$: Moderate; Polynomial $O(n^c)$: slow; Exponential $O(c^n)$: very slow; Factorial $O(n!)$: Extremely slow. Relative empirical performance based on empirical studies performed by Jézéquel (1985), Rossi and Gavioli (1987), Bertsimas (1988), complemented by own computations. Relative performances are based on average performance measures over the full range of probabilities. That is, it attempts to give an overall idea of the methods; some methods may perform better than others for other input values.

construction heuristics has, as far as I am aware, not been conducted. Because of this research gap, I perform an additional empirical performance analysis of the construction heuristics described in this section. The good empirical performance of some construction heuristics suggest great potential as an alterative initial solution generators. As such, updated versions of these approaches to more general situations may lead to new insights. Additionally, the effect of a good starting solution produced by such a state-of-the-art construction heuristic on the computational performance and accuracy of classical improvement heuristics is also worth a study.

Besides testing on the usual characteristics, viz. the number of nodes and the customer probabilities, I also test for a third, often omitted characteristic: customer *clustering*. Clustering refers to the spatial phenomenon in which the graph is split up in several subsets – so-called clusters – of customers, such that the maximum distance needed to connect different parts of the cluster is proportionally smaller than the maximum distance required to connect different clusters. *Sparsity* refers to the opposite phenomenon, viz. a situation in which no subsets can be defined such that the maximum distance required to connect parts of a cluster differs substantially from connecting different clusters. It is synonymous with the situation in which customers are *uniformly divided* (or *evenly divided*) across the plane. The term *dispersion* refers to the quantity measure of sparsity: A high (degree of) dispersion refers to the situation in which customers are mostly uniformly divided across the plane, whereas a low (degree of) dispersion refers to the situation in which obvious clusters can be defined. The review of the key strategies of the construction heuristics in this section promotes the idea that for some construction heuristics, clustering may have a significant impact on the performance. I will therefore pay specific attention to the impact of clustering among different heuristics.

The algorithms described in this section are tested on a range of TSPLIB instances inspired by common choices in previous literature (e.g., Bianchi et al.,

2002a; Balaprakash et al., 2010; Marinakis and Marinaki, 2010). These TSPLIB instances are:

eil51 51 nodes, relatively uniformly distributed;
kroA100 100 nodes, small number of slightly clustered regions;
eil101 101 nodes, relatively uniformly distributed;
ch150 150 nodes, relatively uniformly distributed;
d198 198 nodes, small number of highly clustered regions;
pr439 439 nodes, moderate number of highly clustered regions;
rat783 783 nodes, relatively uniformly distributed;
pr1002 1002 nodes, large number of moderately clustered regions;
fl1400 1400 nodes, small number of highly clustered regions.

The arc costs correspond to Euclidean distances between the nodes. The algorithms are implemented in the C# programming language and run on an Intel Core i3 machine (4 threads) at 1.4 GHz with 4 GBs of RAM. Each algorithm is repeatedly tested on a homogeneous probability set for $p = \{0.1, 0.2, \ldots, 0.9\}$. The following parameter settings are adopted: Level of convergence (savings algorithms, nearest neighbour algorithms): 3; number of tours for random best tour algorithm: 10; level of convergence (spacefilling curves): 20.

Table 4.3 reports the average relative performances of the construction heuristics. The reported figures correspond to the overall average expected length versus the optimal TSP tour per instance $E[L(\rho)]/L(\rho^*_{\text{TSP}}) - 1$, such that a lower figure implies a better performance. The best performing construction heuristic per characteristic is highlighted in boldface typesetting. The results for a low number of nodes are given by the averages of instances eil51, kroA100, eil101, ch150, and d198; high numbers of nodes by instances pr439, rat783, pr1002, and fl1400. Low clustering (high dispersion) is represented by instances eil51, eil101, ch150, rat783; High clustering (low dispersion) by kroA100, d198, pr439, pr1002, fl1400. Low probabilities are all probabilities that satisfy $p < 0.5$; high probabilities are the ones that are $p \geq 0.5$. In the table, result sets with shared characteristics are

Nodes	Clustering	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2	ANNA0
Low	Low	Low	48.74%	44.83%	-42.32%	-43.94%	-42.89%	-42.89%	-36.77%	-36.10%	-35.06%
		High	311.91%	279.07%	-2.31%	**-7.68%**	-7.10%	-7.10%	4.59%	3.99%	8.34%
		Total	194.94%	174.96%	-20.10%	**-23.80%**	-23.01%	-23.01%	-13.79%	-13.82%	-10.95%
	High	Low	164.83%	149.74%	-29.48%	-28.39%	**-30.07%**	**-30.07%**	-24.64%	-23.90%	-23.58%
		High	610.61%	557.96%	0.75%	-1.73%	**-3.29%**	**-3.29%**	7.03%	7.11%	8.47%
		Total	412.48%	376.53%	-12.68%	-13.58%	**-15.20%**	**-15.20%**	-7.04%	-6.67%	-5.78%
	Total	Low	95.18%	86.79%	-37.19%	-37.72%	**-37.77%**	**-37.77%**	-31.92%	-31.22%	-30.47%
		High	431.39%	390.62%	-1.09%	-5.30%	**-5.58%**	**-5.58%**	5.57%	5.24%	8.39%
		Total	281.96%	255.59%	-17.13%	-19.71%	**-19.88%**	**-19.88%**	-11.09%	-10.96%	-8.88%
High	Low	Low	410.22%	404.11%	-33.05%	**-36.66%**	-34.90%	-34.90%	-27.20%	-27.19%	-27.70%
		High	1324.39%	1276.76%	6.06%	**-2.22%**	**-2.22%**	**-2.22%**	10.15%	10.16%	10.86%
		Total	918.09%	888.91%	-11.32%	**-17.53%**	-16.74%	-16.74%	-6.45%	-6.44%	-6.28%
	High	Low	956.18%	940.48%	-25.39%	**-31.41%**	-30.47%	-29.15%	-24.18%	-24.05%	-21.11%
		High	2873.56%	2773.51%	12.63%	**-1.04%**	-0.99%	1.21%	12.90%	12.91%	13.96%
		Total	2021.39%	1958.83%	-4.27%	**-14.54%**	-14.09%	-12.28%	-3.58%	-3.52%	-1.62%
	Total	Low	819.69%	806.39%	-29.22%	**-32.72%**	-31.57%	-32.02%	-24.93%	-24.83%	-22.75%
		High	2486.27%	2399.32%	9.35%	**-1.34%**	-1.30%	-0.50%	12.21%	12.22%	13.18%
		Total	1745.57%	1691.35%	-7.79%	**-15.28%**	-14.75%	-14.51%	-4.30%	-4.25%	-2.79%
Total	Low	Low	139.11%	134.65%	-40.00%	**-42.12%**	-40.89%	-40.89%	-34.38%	-33.87%	-33.22%
		High	565.03%	528.49%	-0.22%	**-6.32%**	-5.88%	-5.88%	5.98%	5.54%	8.97%
		Total	375.73%	353.45%	-17.90%	**-22.23%**	-21.44%	-21.44%	-11.96%	-11.98%	-9.78%
	High	Low	639.64%	624.18%	-28.12%	-30.20%	**-30.31%**	-29.77%	-24.36%	-23.99%	-22.10%
		High	1968.38%	1887.29%	4.71%	-1.32%	**-1.91%**	-1.79%	10.55%	10.59%	11.76%
		Total	1377.83%	1325.91%	-9.88%	-14.15%	**-14.53%**	-14.22%	-4.97%	-4.78%	-3.29%
	Total	Low	417.18%	406.61%	-34.91%	-35.50%	-35.01%	**-36.12%**	-28.82%	-28.38%	-27.04%
		High	1344.67%	1283.38%	1.89%	-3.54%	-3.67%	**-4.13%**	8.52%	8.34%	10.52%
		Total	932.45%	893.70%	-14.46%	-17.74%	-17.60%	**-18.35%**	-8.07%	-7.98%	-6.17%

Nodes	Clustering	Probability	ANNA1	Iterated NN	Random NN (type I)	Random NN (type II)	Radial sort	Spacefilling curves	Farthest insertion	Nearest insertion	All
Low	Low	Low	-31.63%	-38.30%	-28.14%	-27.56%	-37.46%	**-44.36%**	-43.46%	-41.02%	-28.14%
		High	8.29%	-0.75%	16.49%	20.35%	48.07%	-2.20%	-3.86%	-0.38%	39.39%
		Total	-9.45%	-17.44%	-3.35%	-0.94%	10.06%	-20.94%	-21.46%	-18.45%	9.38%
	High	Low	-22.26%	-24.03%	-19.66%	-13.65%	-3.89%	-16.16%	-28.76%	-24.21%	-1.66%
		High	7.53%	5.50%	19.39%	28.16%	99.89%	23.76%	-0.06%	5.35%	80.77%
		Total	-5.71%	-7.62%	2.04%	9.57%	53.77%	6.01%	-12.81%	-7.79%	44.14%
	Total	Low	-27.88%	-32.60%	-24.75%	-22.00%	-24.03%	-33.08%	-37.58%	-34.30%	-17.55%
		High	7.99%	1.75%	17.65%	23.47%	68.80%	8.18%	-2.34%	1.91%	55.95%
		Total	-7.96%	-13.52%	-1.19%	3.26%	27.54%	-10.16%	-18.00%	-14.18%	23.28%
High	Low	Low	-23.25%	-24.64%	-18.97%	-8.95%	62.62%	-32.78%	-24.85%	-31.09%	28.87%
		High	9.94%	8.48%	21.50%	27.82%	348.61%	17.54%	12.70%	7.88%	181.54%
		Total	-4.81%	-6.24%	3.52%	11.48%	221.51%	-4.82%	-3.99%	-9.44%	113.69%
	High	Low	-18.53%	-24.64%	-11.55%	-4.50%	44.48%	-23.73%	-26.46%	-25.82%	98.41%
		High	13.99%	5.97%	32.70%	39.72%	278.63%	20.66%	5.65%	7.22%	358.24%
		Total	-0.46%	-7.63%	13.03%	20.07%	174.57%	0.93%	-8.62%	-7.46%	242.76%
	Total	Low	-19.71%	-24.64%	-13.41%	-5.61%	49.02%	-25.99%	-26.06%	-27.14%	81.02%
		High	12.98%	7.22%	29.90%	36.74%	296.13%	19.88%	7.41%	7.39%	314.06%
		Total	-1.55%	-6.94%	10.65%	17.92%	186.30%	-0.51%	-7.46%	-7.96%	210.49%
Total	Low	Low	-29.54%	-34.89%	-25.85%	-22.90%	-12.44%	-41.46%	-38.81%	-38.54%	-13.89%
		High	8.71%	1.55%	17.75%	22.22%	123.21%	2.74%	0.28%	1.68%	74.93%
		Total	-8.29%	-14.64%	-1.63%	2.16%	62.92%	-16.91%	-17.09%	-16.19%	35.46%
	High	Low	-20.02%	-24.24%	-14.79%	-8.16%	25.13%	-20.70%	-27.38%	-25.17%	58.38%
		High	11.41%	5.66%	27.38%	35.09%	207.13%	21.90%	3.37%	6.47%	247.25%
		Total	-2.56%	-7.63%	8.63%	15.87%	126.25%	2.97%	-10.30%	-7.59%	163.31%
	Total	Low	-24.25%	-30.32%	-19.71%	-14.71%	8.44%	-29.93%	-32.46%	-31.12%	26.26%
		High	10.21%	3.31%	23.10%	29.37%	169.83%	13.38%	2.00%	4.34%	170.66%
		Total	-5.11%	-11.64%	4.07%	9.78%	98.10%	-5.87%	-13.32%	-11.42%	110.83%

Table 4.3: Average deviations from the optimal TSP tour lengths $E[L(\rho)]/L(\rho^*_{\text{TSP}})-1$ for 17 construction heuristics across 9 different TSPLIB instances.

averaged out in order to keep results concise and tractable — see Appendix A for the full set of results.

Not surprisingly, the Random search heuristic performs worst out of all 17 tested construction methods, followed closely by the random best heuristic. The latter result can be explained by the mechanism of the Random Best heuristic: It repeatedly constructs a random tour and then picks the best out of the set of constructed tours. Hence, for each tour there is approximately a 50% chance of improving over the initial random tour. The improvement itself, however, will not be large, as even the best tour is still completely random. For that reason the Random search and Random Best heuristics are the only heuristics that clearly stand out from the other heuristics in terms of performance; the performance confidence intervals (not reported here) of every other heuristic overlaps with the performance of at least one other heuristic.

The supersavings-based and insertion-based heuristics constitute the top of the performance list. Jézéquel (1985) and Rossi and Gavioli (1987) already demonstrated that supersavings-based approaches tend to outperform the nearest neighbour-based algorithms, which is also reflected by the results. It is surprising though that two of the other top performing heuristics, i.e., Farthest Insertion and Nearest Insertion, are heuristics that do not rely on the objective function of the PTSP. More specifically, the implementations of Farthest Insertion and Nearest Insertion reported here do not take the stochastic conditions of the problem instances – i.e., the customer probabilities – into account, but merely rely on the (deterministic) distance matrix for the construction of solutions. This could imply that, on average, a reasonably good solution can be obtained without paying much attention to the underlying structure of the problem.

The supersavings-based and insertion-based heuristics are followed by heuristics that are mainly based on nearest neighbour mechanisms. The iterated NN performs best out of all of these heuristics, followed by PNN1 and its modified

version PNN2. The modified ANNA0 and its extension into ANNA1 follow at close distance. The nearest neighbour algorithms that rely on randomness in their construction phase, i.e. the Random NN (type I) and Random NN (type II) algorithms by Liu (2010) reside at the bottom of the list.

To investigate the behaviour of the heuristics in different settings, additional analyses are carried out by splitting up the probability range into low probabilities ($p < 0.5$) and high probabilities ($p \geq 0.5$), a high number of customers ($n \leq 198$) and a high number of customers ($n > 198$), relatively uniformly distributed instances (eil51, eil101, ch150, and rat783) and relatively clustered instances (kroA100, d198, pr439, pr1002, and fl1400).

Confirming the findings of Bertsimas (1988), the results for low probabilities show a significant improvement of performance of the Radial Sort heuristic for low probabilities. This is not surprising, as Bertsimas (1988) himself already pointed out that optimal solutions for the PTSP are approximately star-shaped only for low probabilities ($p \leq 0.1$) such that its performance tends to deteriorate for higher probabilities. As a result, Radial Sort does not perform very well on average. The relative performances of the other heuristic remain relatively stable across the low probability range. Compared to the low probability range, the performances for the high probabilities reveal that the absolute performance increase is relatively small compared to the low probability range. In addition, the relative performances of the heuristics are rather close. Although this makes it easier for a heuristic to improve over another heuristic, the actual difference in performance is only marginal.

Going from a uniform to a clustered problem instance when the number of nodes is low, the relative performance of all heuristics remains the same. Moreover, when the number of nodes increases from low to high, the relative performance ranks of the Supersavings, Newsave and Nearest Insertion heuristics are not affected. On the other hand, the performance ranks of the Spacefilling Curve heuristic and Farthest Insertion decreases.

Increasing the degree of clustering for a large number of customers, the relative performance ranks of Supersavings, Newsave and Globalsave remain relatively stable. The relative performance ranks of nearest neighbour-based heuristics and the Spacefilling Curve heuristic decreases, whereas the Farthest Insertion and Nearest Insertion heuristics both increase. Furthermore, whereas the performance of latter two heuristics decrease for a small number of nodes when increasing the degree of clustering, the performance increases for a high number of nodes when increasing the degree of clustering.

Finally, increasing the clustered instances from a low to a high number of customers does not affect the relative performance ranks of the Supersavings, Newsave and Farthest Insertion heuristics. The relative performance of the Spacefilling Curve heuristic, on the other hand, decreases, and the relative performance of Nearest Insertion increases.

In a nutshell, the observations show that the relative performance ranks (i.e., the performance marks relative to those of other construction heuristics) of the supersavings-based algorithms, remain relatively the stable across all problem instances. In this respect, it makes sense that the more 'restricted' Newsave case performs slightly worse than the Supersavings algorithm. Overall, their performances are similar and seldom differ.

In conclusion, the Supersavings-based heuristics seem to provide consistent and good results across almost all problem instances. In contrast, the relative performance of the spacefilling curve heuristic and the Nearest Insertion heuristic seem to be influenced by the number of nodes and/or the node dispersion. The additional analyses show that the Spacefilling Curve heuristic is mostly affected by the dispersion of the nodes (performing better as problems are less clustered), whereas Nearest Insertion is affected by both the number of nodes as well as the dispersion (performing better for a high number of nodes or highly clustered problems). The notion that both the performance mark and the absolute

performance of algorithms are affected by the dispersion of the problem gives reason to investigate and exploit this trait even further in the quest of finding even better performing heuristics.

4.4 Classical improvement heuristics

Improvement heuristics are, as its name already implies, heuristics that improve upon an existing solution. Consequently, improvement heuristics require an initial solution generated by a construction heuristic as their input. Using a combination of explorative and exploitative strategies, improvement heuristics attempt to restructure the existing solution so as to find an even shorter tour. Improvement heuristics can be divided into a number of subcategories. Figure 4.10 provides an overview. This section addresses the improvement heuristics for the PTSP per subcategory.

4.4.1 Probabilistic 2-opt

Given a Hamiltonian tour along all customers in a graph, k-opt interchange heuristics rely on the concept of swapping k arcs in the existing tour and testing whether these moves improve upon the current tour length. If so, the arc swaps are carried out and the tour length is updated accordingly; if not, the tour remains unchanged. The most common k-opt interchange heuristics for the PTSP are 2-opt and 3-opt (e.g. Jaillet, 1985; Bertsimas and Howell, 1993; Bertsimas et al., 1995; Campbell and Thomas, 2008b; Balaprakash et al., 2010). But sometimes also alternative schemes such as 2.5-opt (Weyland et al., 2009a) and Or-opt (Tang and Miller-Hooks, 2004) are considered.

k-Opt for the PTSP

The first mention of the use of k-opt interchange heuristics for the PTSP appeared in Jaillet (1985). Jaillet (1985) proposes a general framework that bears large

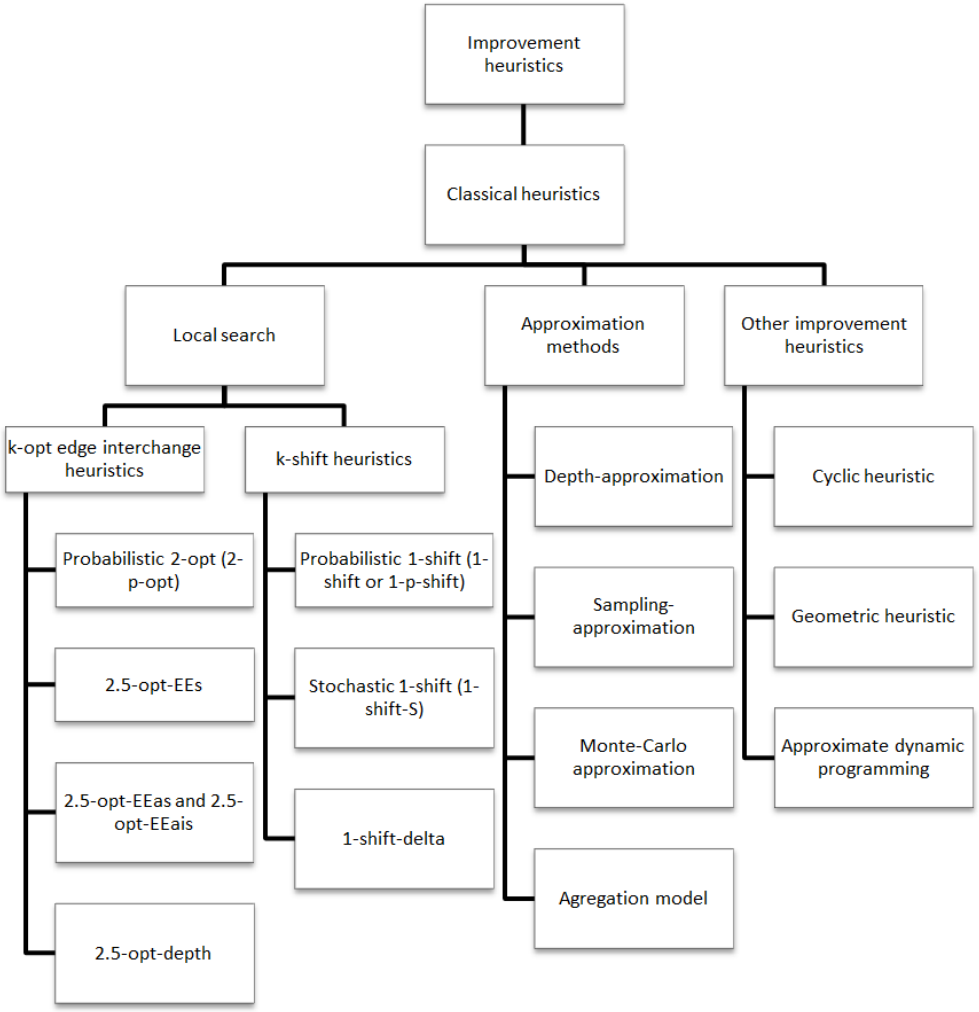


Figure 4.10: Overview of the most important improvement heuristics for the PTSP classified by their solution strategy.

similarities to the solution scheme of the supersavings algorithm (see section 4.3.1), and demonstrates that a k-opt interchange mechanism takes $O(cn^k)$ polynomial time for some constant factor c. This method relies on an initial subtour with a common depot (i.e., a black node) of size c. For a more general, complete Hamiltonian a priori tour along all nodes in the graph irrespective of the number of black and white nodes, Jaillet (1985) argues that an interchange of k arcs with any k other arcs in the tour requires a complete recalculation of the length of the tour to evaluate its merit. Hence, k-opt would take $O(n^2)$ computational effort for each considered move.

2-p-opt

In spite of this observation, Bertsimas (1988) made a first attempt of transforming the deterministic 2-opt algorithm to the general probabilistic context, called 2-p-opt. It relies on the concept that one only needs to know the difference between the tour lengths before and after reversing a tour segment $(i, i+1, \ldots, j)$ in the a priori tour, rather than re-evaluating the entire tour length. However, Bianchi and Knowles (2002) prove the method to be incorrect; thereby invalidating some of the earlier results reported by Bertsimas (1988), Bertsimas et al. (1990), Bertsimas and Howell (1993), Chervi (1990), and Bertsimas et al. (1995).

Bianchi et al. (2005) subsequently propose a correction to the 2-p-opt algorithm for the homogeneous PTSP. (See Bianchi and Campbell, 2007, for the extension to the heterogeneous case.) Their implementation involves a two-stage process, where only interchanges of directly neighbouring customers are considered in the first stage, and the resulting neighbouring solutions are interchanged in the second stage. This method can be briefly summarised as follows.

Given an initial tour, phase one computes the expected increase or decrease in length after swapping two directly neighbouring customers i and $i+1$ for all $i \in \{1, \ldots, n\}$. It mainly operates on two matrices that investigate the effect of swapping nodes i and $i+1$ on the rest of the a priori tour. If the move is favourable, that is, if swapping the two consecutive customers i and $i+1$ results in a decrease of the overall tour length, the swap is carried out and the a priori tour is updated accordingly. The main idea behind the first phase is that most improvements over the initial tour can be made by simply reordering the sequence of the customers in the tour. One does not need to consider any significant arc swaps, thereby greatly reducing the computational efficiency of the algorithm.

The second phase of 2-p-opt takes the result of the first phase as input, and considers major alterations within the tour. Starting at any node i in the current tour, it considers all possible arc swaps of $(i, i+1)$ with $(j, j+1)$ for $j > i+1$ and

calculates the resulting increase or decrease in expected tour length. This is done by first evaluating the nodes j that are closest to node i in the tour, and subsequently moving away further in the tour and considering larger alterations. Each arc swap is evaluated using the principles proposed by Bertsimas (1988). That is, it computes the impact of reversing tour segment $(i, i+1, \ldots, j)$ – referred to as 'inside' – on the rest of the tour – referred to as 'outside', and then tests whether the reversion led to a decrease of expected length. If a swap of arcs is profitable, the move is carried out, and the algorithm then returns to the first phase. Because a more favourable ordered sequence of customers in the new tour may exist, every single customer swap within the tour needs to be re-evaluated. The algorithm is stopped if all possible arc swaps have been considered, or if some user-defined stopping criterion has been met.

The concept of this mechanism is to consider only moves between relatively small arc segments at first, and re-evaluating all the smaller arc segments as we proceed. Larger alterations are left for a later stage, which may not even be necessary after the arc swaps of smaller lengths have been carried out. Furthermore, by returning to the first phase – which relies on two precalculated results after every change within the second phase – we can greatly reduce the computational effort of the algorithm. Contrary to Jaillet's (1985) belief that k-opt interchange procedures require a complete recomputation of the expected tour length after every move, the 2-p-opt scheme only requires the computation of the difference in a priori length after every move. This reduces the computational effort of 2-p-opt from $O(n^4)$ to $O(n^2)$.

4.4.2 Probabilistic 1-shift

The 1-shift algorithm, or simply 1-shift, reinserts a customer at another location in the a priori tour. More specifically, given an a priori tour $\tau = (1, 2, 3, \ldots, i-1, i, i+1, \ldots, j-1, j, j+1, \ldots, n, 1)$, 1-shift inserts a customer i to the position in the tour directly after another location j, $i < j$. All intermediate nodes

$(i+1,\ldots,j)$ are moved one step backwards, such that the resulting tour becomes $\tau = (1,2,3,\ldots,i-1,i+1,\ldots,j-1,j,i,j+1,\ldots,n,1)$.

1-p-shift

The first 1-shift algorithm for the PTSP, named 1-p-shift, was introduced by Bertsimas (1988) along with the 2-p-opt procedure. Because the Bertsimas (1988) version of 1-p-shift relies on some of the same notions as the 2-p-opt scheme, the algorithmic procedure was invalidated in the same paper by Bianchi and Knowles (2002) that invalidated the 2-p-opt algorithm.

Bianchi et al. (2005) propose a correction to the probabilistic 1-shift algorithm of Bertsimas (1988). It proceeds along the same lines as the first phase of the 2-p-opt approach (see section 4.4.1), swapping immediate neighbours in the tour. In the second phase of the algorithm, node positions j farther away from the candidate node i are considered. If reinserting customer i at location $j+1$ is a worthwhile endeavour (i.e. the expected length that results from reinsertion is smaller than the current length), customer i is reinserted at that position in the tour and the algorithm returns to the first phase. This process is repeated until no more moves are left that could result in a better solution: a local optimum. As in 2-p-opt, only the differences in expected length of the a priori tour are required in order to check if reinserting customers is worth the effort, thereby greatly enhancing the computational efficiency of the algorithm.

Beraldi et al. (2005) independently develop a similar strategy as Bianchi et al. (2005) for the PTSP with pickup and deliveries, called neighbourhood search. As pointed out by Campbell (2006), their method can also be applied to the 'basic' PTSP. Neighbourhood search is essentially the same as the 1-shift procedure described above, where the first phase of the algorithm (swapping neighbouring customers) is referred to as a *1-move*.

1-shift-S and 1-shift-delta

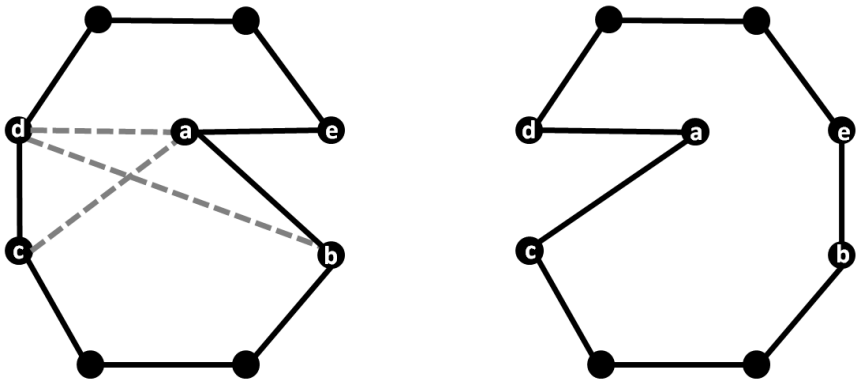
Other variations of 1-shift include an implementation for the heterogeneous PTSP (Bianchi and Campbell, 2007), 1-shift-S (Bianchi and Gambardella, 2007) and 1-shift-delta (Weyland et al., 2009a). The 1-shift algorithm for the heterogeneous PTSP is a straightforward extension of the 1-shift algorithm for the homogeneous PTSP. The 1-shift-S algorithm replaces the expression to calculate the difference in expected tour lengths before and after reinserting a customer by an approximation, thereby decreasing the overall computational effort of the overall algorithm from $O(n^2)$ to $O(cn)$. The 1-shift-delta algorithm precalculates the possible resulting differences in a priori solution costs for all customers, such that the checks required to investigate the profitability of reinserting a customer can be carried out even faster.

4.4.3 Probabilistic 2.5-opt

The 2.5-opt algorithm (Bentley, 1992) is an algorithm that combines the 2-opt edge interchange mechanism from section 4.4.1 with the 1-shift reinsertion procedure of section 4.4.2. Probabilistic implementations of the 2.5-opt algorithm for the PTSP have gained a reasonable amount of attention lately (e.g. Birattari et al., 2008; Balaprakash et al., 2009a; Weyland et al., 2009a, 2013a) and also seem to perform relatively well in combination with metaheuristics (e.g. Balaprakash et al., 2007, 2009b, 2010).

Bentley (1992) explains that the 2.5-opt algorithm can essentially be regarded as a cross-over between the 2-opt and 3-opt algorithms. Whereas 2-opt is concerned with exchanging arcs between only 4 nodes, the 3-opt algorithm is concerned with arcs between 6 nodes; the 2.5-opt algorithm is concerned with arcs between 5 nodes. It recombines the structure of a given tour by carrying out a 2-opt exchange between two arcs and simultaneously reinserting another node – similar to 1-shift – in between any of the two arcs 2-opt is operating on.

In a nutshell, the 2.5-opt algorithm operates as follows (see Figure 4.11). Given a tour and two edges (a, b) and (c, d) within that tour, the 2-opt move considers an



(a) Possibilities of recombination and reinsertion

(b) 2.5-opt move

Figure 4.11: Illustration of the 2.5-opt move according to Bentley (1992): Recombining edges (a, b) and (c, d), in addition to checking the insertion of customer a in between customers c and d (figure a), leads to the tour in figure (b).

exchange of edges (a, b) and (c, d) (see Figure 4.11a). But in addition, the 2.5-opt move also checks a possible 1-shift move that re-inserts any customer of the 2-opt move, say customer a, in between the customers of the other edge (c, d), resulting in the tour illustrated in Figure 4.11b.

2.5-opt-EEs

Out of all probabilistic implementations of the 2.5-opt algorithm, the 2.5-opt-EEs algorithm (Birattari et al., 2008) is by far the most widespread adaptation. The 2.5-opt-EEs algorithm is a special implementation of 2.5-opt that adopts the 2.5-opt algorithm, but extends it with a set of computational enhancements that account for the stochastic nature of the PTSP. More precisely, the 2.5-opt-EEs algorithm consists of two separate elements: The deterministic 2.5-opt approach, and a stochastic feature called *empirical estimation* (EE) and *speedup* (s) (Birattari et al., 2008). Because of its recent popularity and easy integration within other improvement frameworks for the PTSP, 2.5-opt-EEs is also referred to as the *state-of-the-art*

iterative improvement algorithm (Balaprakash et al., 2007; Weyland et al., 2012, 2013b).

EEs basically comprises a fast way to empirically evaluate the current solution quality. That is, to evaluate the merit of performing a particular 2.5-opt move, it is checked against a sample of generated a posteriori tours for the problem at hand. This check is carried out by employing a sophisticated method known as delta evaluations (due to Chervi, 1990): Rather than computing the entire solution costs of the a posteriori tour for a given realisation of the a priori tour, a delta evaluation expresses the costs in terms of the differences between the current and resulting a posteriori tours for a given change of the edges in the a priori tour. That way, a complete re-evaluation of the costs for each realisation of the a posteriori tour in terms of the a priori solution is unnecessary, leading to great improvements in the computational speed of the algorithm. The costs produced by each delta evaluation – one for each a posteriori tour in the sample – are eventually averaged over the total number of a posteriori tours in the sample. If the costs are negative, the move is beneficial and carried out accordingly.

Observe that the empirical evaluation of the 2.5-opt algorithm is performed against a sample of a posteriori tours, rather than calculations of the expected difference in length of the a priori tour. Because such checks do not require any computations involving the (relatively time-consuming) evaluation of stochastic terms according to Jaillet's (1985) objective function, this evaluation method effectively reduces the overall computational speed. On the other hand, the number of a posteriori tours required to obtain a representative sample for the problem depends on the number of nodes in the graph, as well as their associated probabilities. The sample size may therefore greatly affect the overall accuracy of the 2.5-opt-EEs algorithm (Birattari et al., 2008).

2.5-opt-EEas and 2.5-opt-EEais

Based on the latter observation, 2.5-opt-EEs was subjected to two notable

extensions: adaptive sampling (2.5-opt-EEas) and importance sampling (2.5-opt-EEais). Adaptive sampling was suggested by Balaprakash et al. (2009a) as a way to increase the number of generated a posteriori samples for low customer probabilities. The other feature, importance sampling, was suggested in the same paper by Balaprakash et al. (2009a) and implemented on top of adaptive sampling. Importance sampling biases the delta evaluations that are used to compute the merit of every move in such a way that it results more often in a nonzero cost difference. More specifically, it imposes a positive bias on the customer probabilities such that these customers occur more often in the sample of the a posteriori generated solutions. Balaprakash et al. (2009a) report significant improvements of 2.5-opt-EEas and 2.5-opt-EEais over 2.5-opt-EEs for low customer probabilities.

2.5-opt-sampling, 2.5-opt-depth and 2.5-opt-threshold

Weyland et al. (2009a) propose three alternative probabilistic 2.5-opt interchange mechanisms based on 2.5-opt, called 2.5-opt-sampling, 2.5-opt-depth and 2.5-opt-threshold.

2.5-opt-sampling simply takes the 2.5-opt-EEs approach from Birattari et al. (2008), and speeds up the algorithm by calculating the values associated with visiting the customers directly before and after every customer at the initialisation phase of the algorithm. By updating these values after every move that is made instead of recalculating them, the computational effort can be reduced to some extent.

The 2.5-opt-depth algorithm also applies the 2.5-opt algorithm of Bentley (1992), but uses the expected length function (3.4) by Jaillet (1985) to evaluate the merit of a possible move instead of using the emperical estimation and speedup technique of Birattari et al. (2008). There is one notable difference, however, between using the full expression for the expected length (3.4) and the implementation by Weyland et al. (2009a). As Birattari et al. (2008) already

observed, the evaluation of this function can be imprecise and computationally exhaustive when the number of nodes is large. For that reason, 2.5-opt-depth only calculates the expected length for the edges in the tour that carry a cost lower than some predetermined boundary. This calculation is further amended to compute only the differences resulting from a move performed by 2.5-opt, rather than evaluating the full functions, similar to the delta evaluations of 2.5-opt-EEs.

2.5-opt-threshold is essentially the same as 2.5-opt-depth, but instead of putting a boundary on the costs of the edges that are involved in the computation of the expected length, 2.5-opt-threshold puts a threshold on the customer probabilities that are involved in the computation of the expected length. That is, the algorithm only includes the customers whose probabilities exceed a certain threshold into the computation of the expected length. Similar to 2.5-opt-EEs and 2.5-opt-depth, 2.5-opt-threshold also uses delta evaluations to assess the gains of a possible move. A combination of the 2.5-opt-depth and 2.5-opt-threshold algorithms, in which first 2.5-opt-depth is carried out and then 2.5-opt-threshold, is called 2.5-opt-combined. 2.5-opt-combined seems to work particularly well for relatively low customer probabilities.

4.4.4 Approximate evaluation

Besides focusing purely on runtime reducing techniques such as EEs and depth sampling for new heuristics, researchers have also given some attention to ways of speeding up the computations performed by existing heuristics. This is often accomplished by manipulating the PTSP objective function (3.3). A manipulation should lead to a simplification of the objective function such that it is easier to evaluate by the heuristic that uses it, thereby reducing the overall computational effort of the algorithm. In turn, a simplification of the objective function results in an approximation to the exact solutions generated by (3.3), hence the term ‘approximate evaluation’. Examples of approximate evaluation for the PTSP can be found in Bianchi (2003), Branke and Guntsch (2004), Tang and Miller-Hooks

(2004), Bianchi (2006), Meier and Clausen (2015) Li (2013); Li (2017), and Weyland et al. (2013a).

In brief, an approximate objective function is an equation that is only approximately valid, and which accuracy is considered to be sufficient for most computational purposes. Ideally, an approximate objective function has the ability to convert back into the actual objective function by setting a user-controlled parameter to a specific value; that way, the desired level of precision can be readily adjusted. Observe that approximate objective functions can generally be used in conjunction with any kind of heuristic irrespective of its class as long as it incorporates an evaluation of the PTSP objective function. This section is dedicated to the discussion of improvement heuristics that integrate such approximate objective function.

Depth-approximation

The depth-approximation of Bianchi (2003) is essentially the same as the depth-approximation used by 2.5-opt-depth by Weyland et al. (2009a) described in section 4.4.3. That is, it only calculates the PTSP objective function (3.4) for edge distances up to a certain length, as longer edges are less likely to be present in the a posteriori solution. The solutions generated by the depth-approximation of Bianchi (2003) are extensively tested and compared with the results from sampling-approximation. Branke and Guntsch (2004) propose an almost similar depth-approximation as Bianchi (2003) in the context of the ACO metaheuristics (see section 4.5.2). They add to the discussion by noticing that the evaluation of the approximate objective function instead of the exact objective function effectively succeeds in selecting the actual best solution found so far with high probability.

Tang and Miller-Hooks (2004) observe that the depth-approximations only perform well when the customer probabilities are high. In order to account for lower probabilities, their depth-approximation integrates a penalty term which magnitude depends on the customer probabilities (and optionally also a set of

other factors). Unlike the depth-approximations by Bianchi (2003) and Branke and Guntsch (2004), the depth-approximation by Tang and Miller-Hooks (2004) is specifically designed to deal with the expected increase in length from adding a customer to the tour (similar to delta evaluations), as opposed to evaluating the full length of the tour. They show that their depth-approximation is highly effective for the deterministic versions of the 2-opt and Or-opt algorithms applied to the PTSP.

Finally, the depth-approximation and threshold-approximation schemes which formed the inspiration for the 2.5-opt-depth and 2.5-opt-threshold algorithms can be found in Weyland et al. (2009b). Observe that threshold-approximation is essentially equivalent to depth-approximation, as both approximations are based on the principle of cutting off the calculation (3.4) beyond a certain point that depends on the decreasing probability of increasing arc costs.

Sampling-approximation of the a priori tour

Sampling-approximation takes the objective function in subset notation (3.2) and theorises that, by the central-limit theorem, the average of all possible lengths of the a priori tour $L(\rho|S_i)$ given a subset of nodes S_i of size $i, i = 1,\ldots,N$ should converge to the actual length of the objective function for N large. Consequently, sampling-approximation samples an independent number of node sets S_i of increasing size i, up to a predetermined size $N \leq n$, and averages the resulting expected length calculations up to N for a given a priori tour. This way, the calculation of the objective function (3.2) takes only $N \times n$ computations instead of n^2.

Sampling-approximation of the a posteriori tour

Weyland et al. (2013a) formalises the ideas that are used by the delta evaluations in 2.5-opt-EEs (Birattari et al., 2008) and 2.5-opt-EEais (Balaprakash et al., 2009a) for the PTSP with deadlines. Weyland et al. (2013a) remark that averaging over a sample of Monte Carlo generated a posteriori solutions is essentially the same

as using an approximation of the a priori objective function. The extensions proposed by Balaprakash et al. (2009a) merely serve to improve these approximate evaluation approaches with adaptive sampling (2.5-opt-EEas) and importance sampling (2.5-opt-EEais). Balaprakash (2010) and Balaprakash et al. (2009a) also integrate the same Monte Carlo sampling-approximation in an ACO metaheuristic.

Bear in mind that, although both sampling-approximation of the a priori tour and sampling-approximation of the a posteriori retour rely on a sampling approach, these are very different evaluation methods. Whereas 'regular' sampling-approximation uses the a priori tour to obtain an approximation, Monte Carlo sampling-approximation uses a set of a posteriori tours to obtain an approximation. To complicate matters even further, both approaches can be integrated in the same kind of methods such as 2-p-opt or ACO. This can be confusing, as some authors do not explicitly distinguish between the two sampling approaches and simply refer to their implementation as "a sampling-approximation". One should therefore pay careful attention to the details of the implementation in order to find out which one of the two sampling-approximation has been used.

TSP approximation

TSP approximation is a rather simple one, surveyed by Bianchi et al. (2006b): It replaces the objective function (3.4) by the deterministic objective function of the TSP, thereby reducing the computational effort from $O(n^2)$ to $O(1)$. Observe that TSP approximation may suffer from large inaccuracies, especially when the customer probabilities are low.

Aggregation model

Though formally not an approximate evaluation technique, Campbell (2006) suggests to use an aggregation model that can also be used to reduce the runtime of PTSP algorithms. Her approach is similar to approximate evaluation, in the

sense that it also requires one to rewrite the objective function (3.3). More precisely, the objective function of the aggregation model is rewritten in a way such that it can be used to 'aggregate' the customers of the graph. The model then clusters the customers into a number of regions, after which one of the aforementioned heuristics can be used to solve the proposed objective function for the resulting subproblems. Solving the PTSP as a set of subproblems rather than one large problem could reduce the computational effort involved, as demonstrated by computational experiments (Campbell, 2006). The aggregation model of Campbell (2006) may regarded to be a part of the larger class of stochastic partitioning methods (see Bianchi et al., 2006b).

Note that any form of approximate evaluation is not a self-contained solution method. Therefore, an in-depth description is omitted from this chapter – see the references above for the full details on their implementations. An empirical study that compares the different approximate objective functions is left for further study.

4.4.5 Other improvement methods

Apart from the improvement methods discussed in this section, a number of alternative improvement methods have been proposed to solve other variants of the PTSP. For example, Powell et al. (1995) develop a cyclic heuristic for the m-PTSP and Chervi (1990) develops a geometric heuristic for the same problem. As an alternative to exact dynamic programming for the PTSP (see section 4.2.4), Chervi (1990) also develops an approximate dynamic programming approach in the same paper. Unfortunately, these methods were not tested empirically. As a result, their performance lack supporting evidence.

4.4.6 A comparison of improvement heuristics

Table 4.4 provides a comparison of various features of classical improvement heuristics. The 2.5-opt-EEs heuristic, along with its extensions 2.5-opt-EEas and 2.5-opt-EEais, are the most widely adopted methods. These methods do not only

produce good results, but are also faster than their direct competitor, 2-p-opt. It is somewhat remarkable that the probabilistic adaptations of 1-shift did not deserve the same amount of attention in the literature as the 2-p-opt and 2.5-opt-EEs approaches, despite its overall better performance than 2-p-opt. Finally, also the integration of various approximate evaluation procedures into improvement heuristics have gained a decent amount of popularity recently and leave room for further investigation.

The application of sufficiently explorative heuristics in the improvement phase of *deterministic* routing problems has led to the commonly held belief that advantages accrued by construction heuristics vanish in the final solution. Consequently, altough the empirical performance of improvement heuristics (e.g. Bianchi et al., 2005; Birattari et al., 2008; Balaprakash et al., 2009b; Weyland et al., 2009a) received ample attention – as summarised by Table 4.4 – the impact of the preceding construction phase remained largely unexplored in the probabilistic travelling salesman problem. The analysis of construction heuristics after improvement contributes to the understanding of the performance of other heuristics in stochastic routing problems, since the quality of the solution produced by a construction heuristic may also affect the quality and computation time of improvement heuristics and metaheuristics. After all, both types of heuristic classes rely on an initial solution generated by a construction heuristic as their input. Recent evidence by Liu (2010) suggests that deterministic initial solution generators with random components can affect the computation time of Genetic Algorithms (GA) by more than 50% for heterogeneous PTSP instances. In addition, Liu (2010) shows that the solution quality produced by the GA metaheuristic for homogeneous PTSP instances is also influenced by the choice of the construction heuristic, and that the performance depends on the characteristics of the problem instances.

Table 4.5 summarises the performance of the construction heuristics solutions from section 4.3 after applying 2.5-opt-EEais. As one of the top performing

Method	Strategy	Key characteristic(s)	Limitations	CP	EP
2-p-opt	2-opt	⋆ Probabilistic adaptation of the 2-opt exchange algorithm ⋆ Reliable improvement scheme, has proven itself well for TSP	⋆ Number of edge interchanges, and therefore the overall computational effort, strongly depends on the on the quality of the initial solution ⋆ Imposes constraints on the order of which customers are evaluated ⋆ Requires high precision arithmetics for distant edge interchanges when the number of nodes is large	Moderate	3*
1-shift/1-p-shift 1-shift-S	1-shift	⋆ Probabilistic adaptation of the 1-shift algorithm ⋆ Reliable improvement scheme, has proven itself well for TSP	⋆ Number of node shifts, and therefore the overall computational effort, strongly depends on the on the quality of the initial solution ⋆ Algorithm tends to get stuck in local optima ⋆ Imposes constraints on the order of which customers are evaluated	Moderate (1-shift) to very fast (1-shift-S)	2*
2.5-opt-EEs	2.5-opt	⋆ Probabilistic implementation of the 2.5-opt algorithm using empirical delta evaluations ⋆ State-of-the art algorithm, sophisticated concept ⋆ Tests against historical samples of the a posteriori tour; implicit learning ⋆ Flexible ⋆ Computationally fast	⋆ Number of edge interchanges, and therefore the overall computational effort, depends on the on the quality of the initial solution ⋆ Performance strongly depends on the probability of the nodes and the number of generated a posteriori instances used for the empirical evaluation	Very fast	1*
2.5-opt-depth 2.5-opt-threshold 2.5-opt-combined	2.5-opt	⋆ Probabilistic adaptation of 2.5-opt for the PTSP ⋆ Conceptually closer to other probabilistic improvement heuristics (e.g. 2-p-opt) than 2.5-opt-EEs ⋆ Sophisticated calculation of expected length to reduce computational effort	⋆ Number of edge interchanges, and therefore the overall computational effort, depends on the on the quality of the initial solution ⋆ Perfomance strongly depends on the boundaries (threshold) provided; no robust approach to select threshold.	Moderate to very fast (depending on threshold)	N/A

Table 4.4: A comparison of heuristic improvement methods for the PTSP. CP: Computational performance; EP: (relative) empirical performance. Computational performance markings: Constant $O(1)$: Extremely fast; Linear $O(n)$: Very fast; Loglinear $O(n \log n)$: fast; Quadratic $O(n^2)$: Moderate; Polynomial $O(n^c)$: slow; Exponential $O(c^n)$: very slow; Factorial $O(n!)$: Extremely slow. Relative empirical performance based on empirical studies performed by Bianchi et al. (2005), Birattari et al. (2008), Balaprakash et al. (2009b) and Weyland et al. (2009a). Relative performances are based on average performance measures over the full range of probabilities. That is, it attempts to give an overall idea of the methods; some methods may perform better than others for other input values. The empirical performance of the 2.5-opt with approximate evaluation methods were only tested for low probability instances and have been excluded from the table to allow for a fair comparison.

state-of-the-art solution methods, 2.5-opt-EEais is often used in conjunction with construction heuristics and metaheuristics (Birattari et al., 2008; Balaprakash et al., 2010). Albeit differences are less pronounced compared to the empirical performances of bare construction heuristics reported in Table 4.3, there are still significant differences in the overall quality of the solutions of construction heuristics after improvement. Specifically, whereas the savings-based construction heuristics previously showed consistent performance across all characteristics without performance, some of its performance perks dissipate after applying 2.5-opt-EEais. Simultaneously, some of the nearest neighbour heuristics gained some performance at the expense of the savings-based heuristics. Specifically, Iterated NN attains the best overall performance, yet manages to outperform the other heuristics in only 2 out of the 26 cross sections. In addition, insertion-based construction heuristics remain among the best overall performing heuristics.

Comparing the top 50% performing heuristics (see Appendix A for the raw performance data), a good initial solution generated by a construction heuristic – though not guaranteed – is 11.93% more likely to produce a good final solution after applying 2.5-opt-EEais (p-value 0.009). The difference between the best and worst performing heuristic is the most pronounced in the case when the number of nodes, clustering and probability are all high, with a total difference of 29.17% between the average performance of random search and the Supersavings algorithm. On average, the total difference between the top and worst performing heuristics is as high as 7.79%.

Furthermore, the results in Table 4.5 again suggest that the spatial configuration of customers in the problem is an essential determinant of the extent of the contribution. Specifically, the presence of clustering or sparsity of customers across the plane is one of the main drivers of the relative performance of construction heuristics, with the relative performance of all of the supersavings-based heuristics – except for supersavings – improving with increasing clustering. Overall, starting from a good solution when customers are clustered is 17.24% more likely to result

Nodes	Clustering	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2	ANNA0
Low	Low	Low	-46.55%	-46.47%	-47.06%	-47.38%	-46.85%	-46.85%	-46.77%	-46.66%	-46.97%
		High	-10.87%	-11.15%	-10.32%	-11.23%	-10.69%	-10.69%	**-12.19%**	-12.17%	-11.50%
		Total	-26.73%	-26.85%	-26.65%	-27.29%	-26.76%	-26.76%	-27.56%	-27.50%	-27.26%
	High	Low	-33.07%	-32.61%	-36.13%	-32.03%	-36.53%	-36.53%	-33.74%	-33.85%	**-36.78%**
		High	-7.08%	-7.00%	-9.76%	-5.44%	**-9.95%**	**-9.95%**	-9.43%	-9.40%	-9.24%
		Total	-18.63%	-18.38%	-21.48%	-17.26%	**-21.76%**	**-21.76%**	-20.23%	-20.26%	-21.48%
	Total	Low	-41.16%	-40.93%	-42.69%	-41.24%	-42.72%	-42.72%	-41.56%	-41.54%	**-42.89%**
		High	-9.35%	-9.49%	-10.10%	-8.91%	-10.39%	-10.39%	**-11.08%**	-11.06%	-10.60%
		Total	-23.49%	-23.46%	-24.58%	-23.28%	-24.76%	-24.76%	-24.63%	-24.61%	**-24.95%**
High	Low	Low	-41.16%	-41.39%	-39.87%	-41.50%	-40.52%	-40.52%	**-42.07%**	-41.91%	-41.93%
		High	-4.65%	-4.82%	-3.78%	-5.10%	-5.10%	-5.10%	-7.14%	**-7.25%**	-6.90%
		Total	-20.88%	-21.07%	-19.82%	-21.28%	-20.84%	-20.84%	**-22.66%**	**-22.66%**	-22.47%
	High	Low	-25.78%	-25.55%	-26.63%	**-33.56%**	-32.87%	-27.18%	-32.79%	-31.93%	-31.26%
		High	29.14%	27.08%	16.11%	**-0.04%**	0.02%	10.46%	4.61%	5.05%	6.26%
		Total	4.73%	3.69%	-2.89%	**-14.93%**	-14.60%	-6.27%	-12.01%	-11.38%	-10.42%
	Total	Low	-29.62%	-29.51%	-33.25%	**-35.54%**	-34.79%	-33.85%	-35.11%	-34.43%	-33.93%
		High	20.69%	19.11%	6.16%	**-1.30%**	-1.26%	2.68%	1.67%	1.98%	2.97%
		Total	-1.67%	-2.50%	-11.36%	**-16.52%**	-16.16%	-13.55%	-14.68%	-14.20%	-13.43%
Total	Low	Low	-45.20%	-45.20%	-45.27%	-45.91%	-45.27%	-45.27%	-45.60%	-45.47%	-45.71%
		High	-9.31%	-9.57%	-8.69%	-9.69%	-9.29%	-9.29%	-10.92%	**-10.94%**	-10.35%
		Total	-25.26%	-25.40%	-24.95%	-25.79%	-25.28%	-25.28%	**-26.33%**	-26.29%	-26.07%
	High	Low	-28.70%	-28.37%	-32.96%	-32.95%	**-34.34%**	-33.42%	-33.17%	-32.70%	-33.47%
		High	14.65%	13.45%	-1.14%	-2.20%	**-3.97%**	-3.14%	-1.01%	-0.73%	.06%
		Total	-4.61%	-5.14%	-15.28%	-15.86%	**-17.46%**	-16.60%	-15.30%	-14.94%	-14.84%
	Total	Low	-36.03%	-35.85%	-39.99%	-38.71%	-39.20%	**-40.19%**	-38.69%	-38.38%	-38.91%
		High	4.00%	3.22%	-5.45%	-5.53%	-6.33%	-6.66%	-5.41%	-5.27%	-4.57%
		Total	-13.79%	-14.15%	-20.80%	-20.28%	-20.94%	-21.56%	-20.21%	-19.98%	-19.83%

Nodes	Clustering	Probability	ANNA1	Iterated NN	Random NN (type I)	Random NN (type II)	Radial sort	Spacefilling curves	Farthest insertion	Nearest insertion	All
Low	Low	Low	-46.60%	-47.34%	-47.01%	-46.61%	-47.43%	-47.61%	**-47.87%**	-46.56%	-46.98%
		High	-10.96%	-12.02%	-10.56%	-10.06%	-10.38%	-11.73%	-11.40%	-10.49%	-11.08%
		Total	-26.80%	**-27.72%**	-26.76%	-26.30%	-26.84%	-27.68%	-27.61%	-26.52%	-27.04%
	High	Low	-34.82%	-32.78%	-32.77%	-33.10%	-32.18%	-32.50%	-33.57%	-32.77%	-33.87%
		High	-7.81%	-9.25%	-7.65%	-6.19%	-6.46%	-6.63%	-6.11%	-5.47%	-7.81%
		Total	-19.82%	-19.71%	-18.81%	-18.15%	-17.89%	-18.13%	-18.31%	-17.60%	-19.39%
	Total	Low	-41.89%	-41.52%	-41.31%	-41.21%	-41.33%	-41.57%	-42.15%	-41.04%	-41.73%
		High	-9.70%	-10.91%	-9.39%	-8.51%	-8.81%	-9.69%	-9.29%	-8.48%	-9.77%
		Total	-24.01%	-24.51%	-23.58%	-23.04%	-23.26%	-23.86%	-23.89%	-22.95%	-23.98%
High	Low	Low	-39.97%	-40.48%	-41.31%	-41.32%	-41.36%	-41.40%	-40.73%	-41.38%	-41.11%
		High	-6.00%	-7.09%	-6.57%	-6.66%	-5.00%	-3.75%	-3.42%	-4.22%	-5.45%
		Total	-21.10%	-21.93%	-22.01%	-22.07%	-21.16%	-20.48%	-20.01%	-20.74%	-21.29%
	High	Low	-30.29%	-28.97%	-29.74%	-27.82%	-28.56%	-29.82%	-31.73%	-29.59%	-26.41%
		High	6.08%	11.30%	9.96%	11.99%	21.68%	11.22%	4.33%	6.69%	9.22%
		Total	-10.08%	-6.60%	-7.68%	-5.70%	-.65%	-7.02%	-11.70%	-9.43%	-6.61%
	Total	Low	-32.71%	-34.72%	-32.63%	-31.20%	-31.76%	-32.72%	-33.98%	-32.54%	-30.08%
		High	3.06%	2.11%	5.83%	7.33%	15.01%	7.48%	2.39%	3.97%	5.55%
		Total	-12.84%	-14.26%	-11.27%	-9.79%	-5.77%	-10.39%	-13.77%	-12.26%	-10.28%
Total	Low	Low	-44.94%	-45.62%	-45.58%	-45.29%	-45.91%	-46.06%	**-46.08%**	-45.27%	-45.51%
		High	-9.72%	-10.79%	-9.56%	-9.21%	-9.03%	-9.74%	-9.41%	-8.92%	-9.67%
		Total	-25.38%	-26.27%	-25.57%	-25.24%	-25.42%	-25.88%	-25.71%	-25.07%	-25.60%
	High	Low	-32.10%	-31.51%	-30.95%	-29.94%	-30.01%	-30.89%	-32.46%	-30.86%	-29.39%
		High	.52%	-2.40%	2.92%	4.72%	10.43%	4.08%	.15%	1.83%	2.41%
		Total	-13.98%	-15.34%	-12.14%	-10.68%	-7.54%	-11.47%	-14.34%	-12.70%	-11.73%
	Total	Low	-37.81%	-39.58%	-37.45%	-36.76%	-37.07%	-37.63%	-38.52%	-37.26%	-36.55%
		High	-4.03%	**-7.19%**	-2.63%	-1.47%	1.78%	-2.06%	-4.10%	-2.95%	-2.96%
		Total	-19.04%	**-21.58%**	-18.11%	-17.15%	-15.49%	-17.87%	-19.39%	-18.20%	-18.62%

Table 4.5: Average deviations from the optimal TSP tour lengths $E[L(\rho)]/L(\rho^*_{\mathrm{TSP}})-1$ for 17 construction heuristics after being improved by 2.5-opt-EEais across 9 different TSPLIB instances for homogeneous probabilities.

in a good solution (p-value 0.003), whereas starting from a good solution in sparse instances cannot be concluded to be a precedent for a good ultimate solution (5.52% more likely, but with a p-value of 0.188). These results strengthen the observations by Liu (2010) that different problem scenarios may lead to different solution qualities according to the design of the construction heuristics. Overall, these results provide evidence to the hypothesis that the choice and performance of construction heuristics contribute to the overall quality of the solution after improvement.

4.5 Metaheuristics

Metaheuristics differ from classical improvement heuristics by bearing the property of being problem-independent; classical improvement heuristics, on the other hand, are often problem-specific. However, the division between heuristics and metaheuristics is not always as clear-cut as this. Some of the methods described in the previous section may also fit in the class of metaheuristics, and vice versa.

The problem-independency of a metaheuristic is embodied by a general, common, solution framework that is being relied on regardless of the problem it is applied to. However, some of its features can be readily adjusted to cope with the specific nature of a problem accordingly. In contrast to the classical improvement heuristics of section 4.4, metaheuristics also often allow for intermediate inferior (including infeasible) solutions as well as the recombination of several candidate solutions. Consequently, metaheuristics often perform a more thorough search of the solution space than most classical heuristics. This may result in longer computation times, but also in more accurate solutions (Cordeau et al., 2002). At the same time, their problem-independent base set-up also means that optimal solutions are less frequently encountered than by methods that are specifically designed for the problem to be solved.

The latter observation has led to a trend of increasingly sophisticated and

fine-tuned metaheuristics specifically designed for the PTSP, sometimes only reflecting a mere glimpse of the general frameworks that inspired them. This development arguably defies the original definition and intention of metaheuristics, as these frameworks no longer satisfy problem independence. It raises the question whether such metaheuristics are overengineered and overfitted, rather than 'sophisticated' and fine-tuned'. Nonetheless, metaheuristics currently reside among the state-of-the-art and top performing solution methods for the PTSP.

Metaheuristics come in many different flavours, but the strength of most metaheuristics for the PTSP lies in the ability to improve upon an existing solution. For the construction phase, PTSP metaheuristics rely on results of other (mostly deterministic) construction heuristics, or randomly generated initial solutions. There are some notable exceptions of metaheuristics that also construct a solution. These metaheuristics are annotated appropriately within this chapter.

A literature review by Lin et al. (2014) recently divided metaheuristics for routing problems into two main categories, namely local search (figure 4.12) and population search (figure 4.13). Local search updates the current solution by a neighbouring solution, similar to the local search procedures for improvement heuristics. Population search, on the other hand, forms a population of the solutions found so far, and attempts to improve upon these by using the characteristics of the population. I pursue the same division into subcategories in this section.

As a final note, this section only treats the metaheuristics that underwent changes to deal with the specific nature of the PTSP. For the other, more general descriptions of metaheuristics including their applications to the TSP, I refer the reader to Gendreau and Potvin (2010).

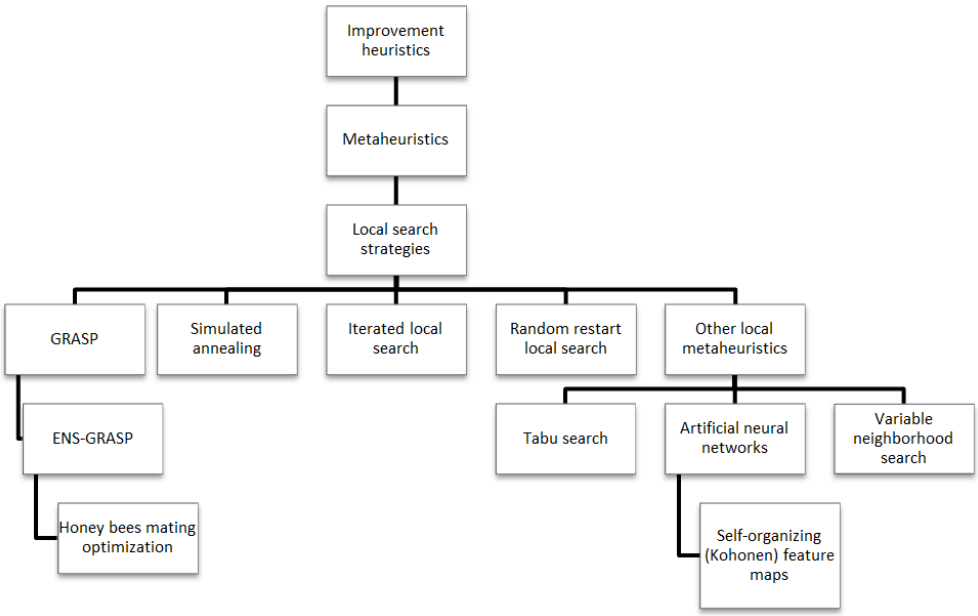


Figure 4.12: Overview of local search metaheuristics for the PTSP classified by their solution strategy.

4.5.1 Local search metaheuristics

Simulated annealing

The simulated annealing metaheuristic, or simply simulated annealing (SA), is a metaheuristic that iteratively improves upon a starting solution by allowing changes to be made with some probability. It takes its concept from annealing, the process of cooling down hot materials. Starting from a hot state, a material cools down into either an amorphous or a crystalline state, depending on the speed of cooling.

SA can be broadly summarised as follows (see Kirkpatrick et al., 1983). Starting from an initial temperature $T = T_0$, the difference in the objective values ΔE of the PTSP (3.4) resulting from a move (e.g. by applying 2-p-opt) determines if the move will be carried out or not. More specifically, if the difference in objective values is positive $\Delta E > 0$, the new neighbour solution is said to improve the objective value, and hence, the move is always carried out. On the other hand, if the difference in objective values is negative $\Delta E < 0$, the move may not directly improve the

objective function. Yet simulated annealing may still carry out the move depending on a probability that is a function of the current temperature T, equal to $e^{-\Delta E/T}$. The difference in objective values ΔE is referred to as the 'energy' of the system. The temperature is repeatedly updated after a number of successful moves have been carried out, by multiplying T by some constant factor α ($0 \leq \alpha \leq 1$). This process is repeated until the temperature reaches a predetermined stopping criterion known as the final temperature. This temperature mimics the state in which the material has cooled down. Alternatively, the final temperature may also be expressed in a fixed number of algorithm iterations.

Observe that many probabilistic improvement procedures discussed in section 4.4 can be integrated into SA. For example, Chervi (1990) applies an SA metaheuristic to the PTSP based on the 2-p-opt procedure described by Bertsimas (1988). This is an example of a straightforward integration of an improvement heuristic into SA which does not require the introduction of new features into the SA metaheuristic. Besides this straightforward adoption of SA for the PTSP, a number of alternative stochastic modifications were also proposed. The remainder of this section discusses these stochastic variations of the SA metaheuristic.

Stochastic simulated annealing

The general structure of *stochastic simulated annealing* (Gutjahr and Pflug, 1996) takes the SA algorithm above and makes a number of slight modifications. Most notably, it uses the a posteriori sampling-approximation (section 4.4.4) to empirically test the profitability of a new candidate solution in each iteration, as opposed to using the actual differences a priori tour lengths implied by the SA approach of, for example, Chervi (1990). Its concept has been tested for the PTSP by Gutjahr (2004).

Bianchi (2006) notices that the approach by Gutjahr and Pflug (1996) to SA can essentially be regarded as a means of perturbation of the energy function ΔE (Gelfand and Mitter, 1989). Perturbation of the energy of the system for

the PTSP entails adding a noise function to the difference in objective functions $\Delta E + \gamma_i$, where γ_i is the outcome of a random variable Γ (or temperature-dependent function) at each algorithm iteration i. According to Gelfand and Mitter (1989), this ensures that SA still converges to some optimal value. A literature review on the exact choice of the perturbation function γ_i for general stochastic combinatorial optimisation problems is performed by Bianchi (2006).

Bowler et al. (2003) propose a variant on stochastic simulated annealing known as *stochastic annealing*. Similar to the stochastic SA approach, it also relies on a replacement of sampling-approximation for the exact PTSP objective function. Their algorithm is based on the limiting behaviour of the PTSP when the number of customers times customer probabilities (np) approach infinity. The error associated with the sampling-approximation is used to regularly update the temperature of their algorithm.

Threshold accepting

In 2004, Tang and Miller-Hooks (2004) propose a stochastic alternative of threshold accepting (TA) as a straightforward extension of deterministic SA. Instead of checking whether the energy from a possible improvement ΔE is positive or negative, threshold accepting checks the energy against a predefined threshold value $-T$ ($T > 0$). Tang and Miller-Hooks (2004) apply their alternative computation of the differences in expected length based on depth-approximation (section 4.4.4) to TA. Unfortunately, Tang and Miller-Hooks (2004) do not report any computational results of their theorised solution scheme.

Empirical estimation enhancements

A number of empirical estimation implementations of SA have been studied by Balaprakash et al. (2007) and Balaprakash (2010). Most notably, Balaprakash et al. (2007) consider the implementation of 2.5-opt-EEais into SA, but report relatively poor results on average for clustered PTSP instances.

GRASP

Greedy randomised adaptive search procedure, or GRASP for short, is a metaheuristic search procedure introduced by Feo and Resende (1995). It is conceptually relatively simple, but proves to be reasonably effective in empirical settings. GRASP consists of a construction phase and an improvement phase. Straightforward implementations of GRASP without any stochastic enhancements for the PTSP include Campbell and Thomas (2008b) and Marinakis et al. (2008).

In the construction phase, an algorithm known as a greedy algorithm is used to construct an initial tour. It consists of both a greedy function and a randomised component to pick the next customer from the set to choose from. The incremental costs associated with inserting a feasible edge into the current path are kept in a so-called restricted candidate list (RCL). In the construction phase, an element is randomly selected from the RCL. If the candidate edge associated with the costs in the RCL joins two customers that have a degree lower than two, they are connected by the edge; if not, another candidate edge is selected from the list. At the end of each iteration in the construction phase, the solution is merged with the costs of the RCL candidate, which concludes the greedy randomised procedure. The RCL is updated accordingly, making the algorithm adaptive to the most recent solution alternatives that are left – hence the terms 'greedy randomised adaptive' in the name of this algorithm.

The second phase of GRASP adds the 'search' term to the name of the algorithm. After the first phase has generated an initial solution, a local search procedure is performed on the solution of the construction phase. This can be any search procedure of choice, from deterministic variants such as 2-opt and 3-opt to probabilistic variants such as 2-p-opt and 1-p-shift.

ENS-GRASP

The most popular implementation of GRASP for the PTSP is known as ENS-GRASP

(Marinakis et al., 2008), where ENS stands for *expanded neighbourhood search*. As in GRASP, ENS-GRASP consists of a construction phase and a local improvement phase.

The construction phase proceeds along the same lines as GRASP. It builds a list of all edges by sorting the edge costs from lowest to highest. However, instead of considering every edge for inclusion, the RCL of ENS-GRASP takes only a subset of the list that corresponds to the D smallest edges. The next steps of the construction phase are identical to the procedure for GRASP outlined above, in which a candidate edge is randomly selected for insertion while accounting for the degree of the customers that are potentially joined through that edge. After the edge is inserted and the solution costs are updated, the RCL is also updated. The RCL update does not only enrich the list with new values on insertion costs, but also inserts an additional member. This member is the next edge with the lowest costs within the sorted list of all possible edges.

The improvement phase applies expanding neighbourhood search (ENS). ENS is another metaheuristic, developed by Marinakis et al. (2005). In a nutshell, it excludes the edges from a local search move (e.g. 2-opt) that have no potential for reducing the objective function. These are all the edges that fall beyond a radius defined by the sum of the costs of the two candidate edges that are selected for removal. ENS fixes the size of this radius, and only searches for local improvements within this region. The radius is repeatedly expanded (e.g. by 10%) to include new edges, until no more improvements can be made. By doing so, ENS greatly increases the search speed of any local search procedure. Marinakis et al. (2008) suggest to use 2-opt and 3-opt for the local searches made by ENS, although this choice can be generalised to any probabilistic interchange mechanisms.

ENS-GRASP has also been the inspiration for a hybrid algorithm, in which Honey bees mating optimisation (HBMO) is combined with ENS-GRASP for the PTSP to form HBMOPTSP (Marinakis and Marinaki, 2009). Despite its name,

HBMOPTSP does not add any special features to cope with the stochastic nature of the PTSP. The performance of HBMOPTSP is further tested for along with a new metaheuristic, the adaptive bumble bees mating optimisation algorithm, for the PTSP in Marinakis et al. (2017).

Random restart local search

Random restart local search algorithm (RRLS) is a direct adaptation of the random best heuristic (section 4.3.5). It functions exactly the same as the random best heuristic, constructing a set of random tours and returning the best among these tours by checking each against the PTSP objective function (3.4). However, unlike the random best heuristic, RRLS also performs local search after each random tour has been generated. Alternative variants of RRLS use more sophisticated deterministic construction heuristics such as the nearest neighbour algorithm (e.g. Balaprakash, 2010) to construct the initial tours.

Straightforward probabilistic adaptations of RRLS are proposed by Balaprakash et al. (2007), Balaprakash (2010) and Balaprakash et al. (2010), who integrate 2.5-opt-EEais (see section 4.4.3) into RRLS (RRLS-EE) and Weyland et al. (2009b); Weyland et al. (2013a) who use 2.5-opt and 3-opt with approximate evaluation techniques (see section 4.4.4). In a number of adaptations, statistical tests (e.g., the student t-test) are used to evaluate the statistical significance of the current solution against the current best solution (Balaprakash et al., 2007; Weyland et al., 2012).

Iterated local search

Iterated local search (ILS; Lourenco et al., 2010) is analogous to RRLS, although ILS often relies on more sophisticated (deterministic) methods to construct an initial solution (e.g. nearest neighbour: Weyland et al., 2009b; Balaprakash, 2010). Moreover, unlike RRLS, ILS perturbs the solution a number of times before it is subjected to a local search improvement algorithm. Perturbation may take place

by reinserting a random number of customers at a different location in the graph using the farthest insertion heuristic (see section 4.3.6) and forming a number of double bridges between some of the customers (Balaprakash, 2010).

The extensions of ILS to the probabilistic case largely resemble the extensions designed for RRLS. More specifically, 2.5-opt-EEais implementations for the local search stage can be found in Balaprakash et al. (2007), Birattari et al. (2008), Balaprakash (2010), Liu (2010) and Balaprakash et al. (2010); approximation evaluation techniques are applied in Weyland et al. (2009b). In addition, Birattari et al. (2008) use a more sophisticated hybrid local search combination of 2.5-opt with ant colony system method for their ILS procedure (see sections 4.4.3 and 4.5.2).

Other local search metaheuristics

Most of the other local search metaheuristics that are applied to the PTSP either use some kind of empirical estimation extension, or an approximate evaluation technique (section 4.4.4). That is, the metaheuristic framework itself is not adapted in any way to cope with the stochastic nature of the PTSP. Examples of such other metaheuristics include tabu search, artificial neural networks and variable neighbourhood search.

Tabu search

Tabu search may be the most unexplored category of metaheuristics for the PTSP. Only deterministic variants of tabu search have been studied in a number of occasions, such as in Bellalouna (1993), Beraldi et al. (2005), Marinakis et al. (2008) and Marinakis and Marinaki (2009); Marinakis and Marinaki (2010). Bianchi (2006, pp. 35–39) dedicates a section of her PhD thesis on a literature review of tabu search for the more general class of stochastic combinatorial optimisation problems.

Artificial neural networks

Another type of metaheuristics that received only little attention in the context of the PTSP are artificial neural networks (ANN). ANN take their name from the networks that are formed within the brain. To the best of my knowledge, only one paper reports on the application of neural networks to the PTSP by Rosenow (1997). Rosenow (1997) uses a variant of ANN that uses self-organising feature maps (Kohonen, 2001) and applies it on the deterministic TSP and PTSP. Empirical results demonstrate that the performance of self-organising maps as a heuristic to construct solutions for the PTSP is relatively poor, as even TSP solutions provide better results. Comparisons against other solution methods are not provided.

Variable neighbourhood search

Gutjahr et al. (2007) develop a general stochastic combinatorial optimisation metaheuristic called stochastic variable neighbourhood search (S-VNS) based on the variable neighbourhood search (VNS) metaheuristic. They demonstrate how S-VNS can be applied to perform a project portfolio analysis. Ghiani et al. (2012) combines GRASP with VNS for the dynamic and stochastic TSP. Voccia et al. (2013) develop yet another VNS metaheuristic and apply it to the PTSP with deadlines. Unfortunately, none of these implementations feature stochastic enhancements for the PTSP.

GPGPU

Weyland et al. (2013b) propose a general framework known as General Purpose Computing on Graphics Processing Units (GPGPUs) that can be integrated into metaheuristics to target stochastic combinatorial optimisation problems. They also provide results for the PTSP with deadlines by integrating GPGPU into the RRLS metaheuristic (topic of section 4.5.1). GPGPU may also be worth further study in the context of the simple PTSP.

4.5.2 Population search metaheuristics

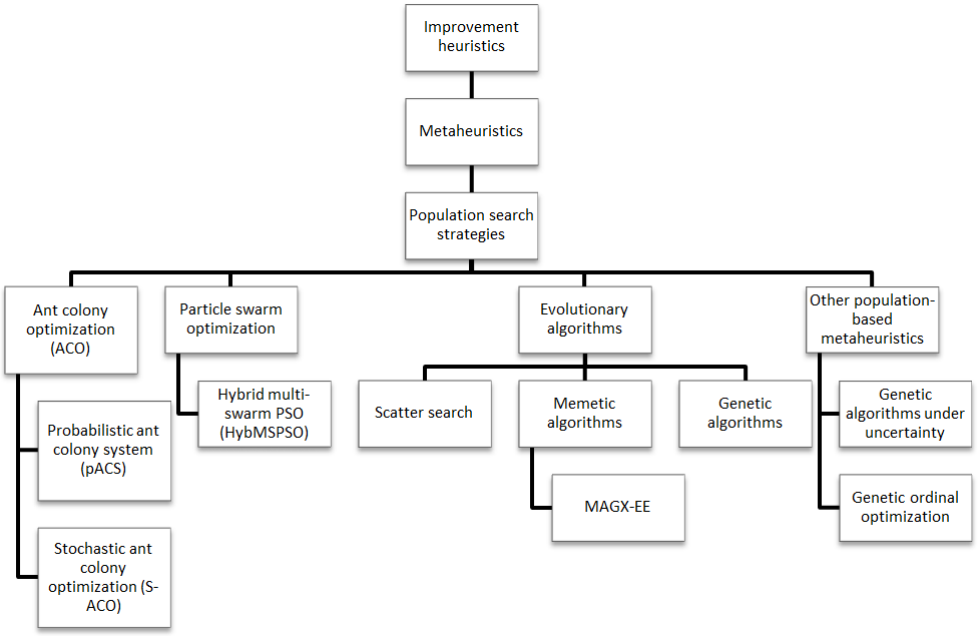


Figure 4.13: Overview of the most important population-based search metaheuristics for the PTSP classified by their solution strategy.

Ant colony optimisation

In contrast to the metaheuristics described in the previous section, ant colony optimisation (ACO) is a metaheuristic that focuses on the construction phase of the solution generation. That is, ACO iteratively builds new candidate solutions from scratch, rather than merely improving upon existing solutions. The ACO metaheuristic is by far the most discussed variant of all metaheuristics for the PTSP. Two notable variants of ACO have been developed for the PTSP, namely probabilistic ant colony system (pACS) and stochastic ant colony optimisation (S-ACO), along with an extension, known as F-Race.

In ACO, a set of artificial ants are set out from their colony in order to find a stochastic solution of the problem at hand. The ants are in search of a better solution by following so-called pheromone trails: these are trails that are laid out by other ants that have already constructed a solution in a previous iteration, and

contain information about the quality of that solution. All subsequent ants follow these trails in order to find an improving solution, mostly obtained by performing local search. Once a new solution has been obtained, the pheromone trails are updated accordingly and the process is repeated until some stopping criterion is met.

Probabilistic ant colony system

Bianchi et al. (2002a); Bianchi et al. (2002b); Bianchi (2003) are the first to develop a probabilistic alternative for the ACO metaheuristic for the PTSP, called probabilistic ant colony scheme (pACS). It is based on ant colony scheme (ACS) by Dorigo and Gambardella (1997), with a set of minor adjustments to account for the probabilistic elements in the PTSP. In essence, the pACS metaheuristic continues in the ordinary way of the ACO procedure outlined above, but it skips the local search procedure after the construction phase of each iterative update procedure. Instead, it goes straight to the pheromone update process.

The construction phase of pACS proceeds along the following lines. First, each arc in the graph is assigned two values: the heuristic information η_{ij}, which is simply the inverse of the arc costs d_{ij}^{-1}, and the quality of the pheromone θ_{ij}. Next, m ants are set out to construct a tour. Their starting points may be chosen randomly in the graph. Each ant $k = 1, \ldots, m$ then constructs a tour according to the same rules. Given the current customer i, the next customer j to be visited is either the 'best' customer in the set of unvisited customers, or a randomly selected customer in the set of unvisited customers. Whether the next customer is the best or a random customer depends on a predetermined probability q_0: with probability q_0 the best customer is selected, and with probability $1 - q_0$ a random customer is selected. The 'best' customer is the customer out of the set of unvisited customers that maximises the quality of the available information of any arc $\eta_{ij}\theta_{ij}^{\beta}$. Here, β is a user-controlled parameter that can be used to tune the proportional influence of the information contained by the pheromone trail θ_{ij} relative to the information

contained by the heuristic information η_{ij}. Similarly, the probability associated with selecting a customer j in case a random customer should be selected is also proportional to this information quality.

The pheromone trails are updated according to a simple weighing scheme. In the tour construction phase, pACS weighs the value θ_{ij} of the current pheromone trail against a user-specified input parameter θ_0. For example, in the empirical analysis by Bianchi et al. (2002a) θ_0 is set equal to the inverse of the number of nodes multiplied by the expected length of a (deterministic) nearest neighbour solution. According to Bianchi et al. (2002a), the general idea behind the weighing scheme of the pheromone updates is to bias the edges for an ant k that have not been used so far by other ants.

Once every ant has completed a tour along all customers, pACS carries out another pheromone trail update with only the arcs belonging to the best tour found so far. This update uses the same weighing scheme, but relies on a trade-off between the current value of θ_{ij} and the inverse value of the objective function (3.4) of the best tour found so far. The value is used for every subsequent pheromone update in the next iteration of the pACS algorithm, such that only the best information about good quality solutions is available to each ant in the future iterations of the construction process. The pACS algorithm terminates when some stopping criterion is met (e.g., a maximum number of iterations has been reached). The resulting solution is the tour that yields the minimum expected length.

Hybridisations of the ant colony system and the probabilistic ant colony system

Because the pACS algorithm requires the repeated evaluation of the PTSP objective function, Bianchi (2006) proposes to improve the computational speed by replacing the actual objective function (3.4) by an approximate one (see section 4.4.4). Bianchi (2006) reports variable results, but on average, finds a balanced reduction in runtime against only constant small drops in accuracy of about 2% using a priori sampling-approximation.

Bianchi (2006) also investigates hybrid methods that, unlike the original pACS approach, integrate a local search method after the tours have been constructed. Variations include both combinations of pACS with 2-p-opt and 1-p-shift (see sections 4.4.1 and 4.4.2, respectively), but also combinations that use the TSP and sampling approximations on top of that. Bianchi and Gambardella (2007) report a comprehensive summary of these results.

Birattari et al. (2008) propose a hybridisation of the 2.5-opt approach (section 4.4.3) into ACS and compare its empirical results with 2.5-opt-EEs. Similarly, Gambardella et al. (2012) and Weyland et al. (2014) propose to integrate a deterministic 2.5-opt variation named 2.5-opt-combined into ACS, resulting in enhanced ACS (EACS). As pACS is derived from ACS, these methods are conceptually the same. However, ACS uses the deterministic objective function of the TSP instead of the probabilistic objective function of the PTSP to evaluate solutions. Moreover, ACS uses deterministic rules to update the pheromone trails (see Dorigo and Gambardella, 1997) and is therefore also different from pACS with TSP-approximation (Bianchi, 2006). In effect, neither ACS nor 2.5-opt-ACS are probabilistic adaptations of solution methods.

Balaprakash et al. (2007); Balaprakash et al. (2009b) propose to integrate 2.5-opt-EEs (see section 4.4.3) into pACS, resulting in a method named pACS-2.5-opt-EEs. Furthermore, they compare the results with a straightforward extension of the deterministic version of ACS with a posteriori approximate-evaluations of the objective function (see section 4.4.4), named ACS-EE. Similar extensions, also considered by Balaprakash et al. (2009b), of ACS with a posteriori approximate evaluations include max-min ACS (MMAS-EE) and best-worst ACS (BWAS-EE). These were also tested empirically against an exhaustive list of other metaheuristics by Balaprakash (2010), Balaprakash et al. (2010), Liu (2010) and Marinakis and Marinaki (2010).

Stochastic extensions of deterministic ACO

Soon after Bianchi et al. (2002a) introduced pACS, Branke and Guntsch (2003); Branke and Guntsch (2004) developed stochastic extensions of the deterministic ant colony optimisation algorithm. More specific, these extensions target the heuristic information η_{ij} used by ACO. Branke and Guntsch (2003) argue that taking the inverse of the arc costs d_{ij} is an inaccurate way of dealing with the actual heuristic information contained by these arcs, because it does not account for the stochastic presence of each of these arcs. In effect, Branke and Guntsch (2003) propose two alternative functions for the heuristic information: A depth-based heuristic and an angle-based heuristic.

The depth-based heuristic is a straightforward adaptation of the deterministic arc costs into a probabilistic one. That is, instead of taking $\eta_{ij} = 1/d_{ij}$ as in pACS, the depth heuristic sets $\eta_{ij} = 1/d^*_{ij}$ where d^*_{ij} involves a probabilistic calculation of the arc costs that takes the presence probabilities of the customers into account (Branke and Guntsch, 2003). However, since the calculation of probabilistic terms take $O(n^2)$ computational effort (Jaillet, 1985), the total complexity of the ACO metaheuristic increases from $O(n^2)$ to $O(n^3)$. The idea of Branke and Guntsch (2003) to calculate the expected values for d^*_{ij} only up to a certain point (hence the name depth-based heuristic, see section 4.4.4) may therefore seem computationally attractive, but is soon abandoned in a later paper Branke and Guntsch (2004). Rather than using a depth-based calculation, Branke and Guntsch (2004) propose to use an alternative approach in which the algorithm keeps track of the list of expected distances and recursively updates these values. Branke and Guntsch (2004) argue that his method is not only faster than employing depth-based calculations, but also more accurate.

The angle-based heuristic by Branke and Guntsch (2003) is similar to the radial sort heuristic (section 4.3.4). But rather than picking a centre of mass, the angle-based heuristic uses the angle between the last traversed edge and the potential next customer to decide how the ant should proceed. It attempts to favour customers that maintain a sharp angle with respect to the last edge, inspired by the

star-shaped example of optimal TSP versus PTSP solutions (section 4.3.4). Because only making sharp turns is arguably a bit radical, the procedure is configured in Branke and Guntsch (2004) to be less aggressive when customers are closer to each other, and more aggressive when heterogeneous customer probabilities of close neighbours are lower.

Stochastic ACO and ACO/F-race

The concept of stochastic ACO (S-ACO) was introduced by Gutjahr (2003) and applied to the PTSP in Gutjahr (2004). S-ACO can be regarded as an integrated a posteriori sample-approximation procedure for ACO. That is, S-ACO proceeds along the same lines as ACO, but instead of checking the tours generated by each ant against the PTSP objective function, the tours are checked against a random generated realisation ("scenario") of the a posteriori tour. The resulting best tour of the ACO iteration is then repeatedly checked against a number of additional a posteriori samples generated by the PTSP. Only if the best tour of the iteration is on average better than the existing one for all the generated samples of the a posteriori tour, it replaces the previous best tour. The adaptive S-ACO, abbreviated S-ACOa by Gutjahr (2004) extends S-ACO by applying statistical tests to determine the best among two candidate solutions. S-ACO is further analysed for the PTSP in Balaprakash (2005).

An alternative selection scheme based on the empirical estimation procedure of Gutjahr (2004) has been developed by Balaprakash (2005). It replaces the normal comparison of a posteriori generated scenarios by taking the average by a more sophisticated tuning scheme, called F-race. The integration of F-race into S-ACO leads to an algorithm called ACO/F-race. Full details of the ACO/F-race algorithm can be found in Balaprakash (2005) along with a set of improvements in Balaprakash (2010). An empirical analysis of ACO for the PTSP is also performed by Birattari et al. (2005) and Birattari et al. (2007).

Evolutionary algorithms

Evolutionary algorithms (EAs) are another class of metaheuristic improvement algorithms for combinatorial optimisation problems. The general procedure of evolutionary algorithms is illustrated in figure 4.14 and consists of four components: an initialisation component, a local search component, a solution selection component and an edge recombination component. Bianchi (2006) elaborates on the literature of evolutionary computation for stochastic combinatorial optimisation problems.

Although conceptually slightly distinct, the term evolutionary algorithms is often also used to describe related, or descending, algorithms, such as memetic algorithms and genetic algorithms. Out of these subcategories, memetic algorithms is the most widespread variant for the PTSP.

Liu (2008a) memetic algorithm for the PTSP

Liu (2008a) is the first to consider an adaptation of a memetic algorithm for the PTSP that adapts the EA scheme. It can be briefly summarised as follows.

The initialisation phases is simple: A tour is generated using a nearest neighbour algorithm according to the type 2 principle formalised by Liu (2010) (see section 4.3.2). This method is repeated a fixed number of times to obtain a population of initial solutions. The subsequent improvement phase consists of a deterministic local search heuristic, either 1-shift or 2-opt, which is selected at random with probability 0.5. The reason for the randomly selection is to increase the diversity of the overall process. Instead of using the actual PTSP objective function to evaluate the efficiency of a move obtained by 1-shift or 2-opt, a depth approximation scheme is used instead (section 4.4.4). This local search procedure is repeated for a predefined number of times for each solution. Only the best solutions resulting from the local search is kept for future evaluations with the algorithm.

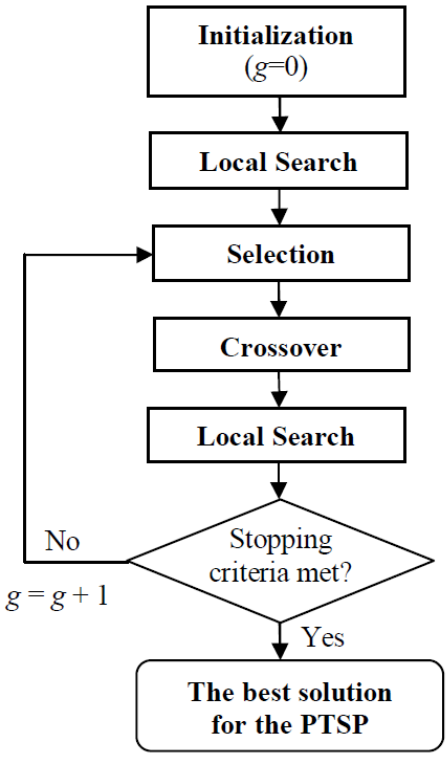
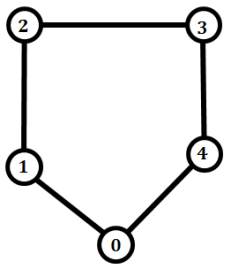


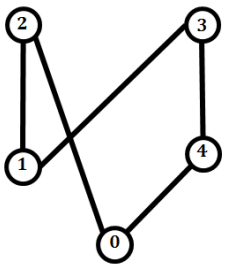
Figure 4.14: Algorithmic procedure of an evolutionary algorithm. Adapted from *A memetic algorithm for the probabilistic traveling salesman problem* by Liu (2008a), 2008 IEEE Congress on Evolutionary Computing, p. 148. Copyright 2008 by Liu.

The next step of MA is an iterative improvement procedure that produces an offspring of the current population of the problem. In the selection phase of this iterative improvement procedure, the solution that is used to create an offspring is selected. This is accomplished by applying a selection scheme. The memetic algorithm by Liu (2008a) uses three schemes: fitness-proportionate selection, which selects a solution with a probability relative to the average quality of the rest of the solutions in the population; tournament selection, which uses fitness-proportionate to select two solutions and picks the best one; and elitism selection, which keeps a predetermined number of only the best solutions found so far.

The next phase, called crossover, uses the edge recombination strategy from genetic algorithms to create new solutions. This strategy is illustrated in Figure



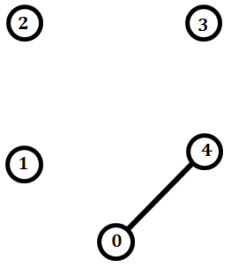
(a) Tour configuration 1



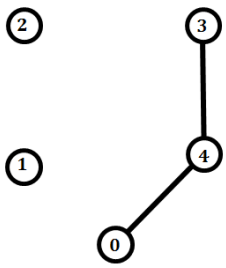
(b) Tour configuration 2

Initial edge list	
0	1,2,4
1	0,2,3
2	0,1,3
3	1,2,4
4	0,3

(c) Initial edge list



(d) Edge recombination I



(e) Edge recombination II

Active edge list		
	ER I	ER II
0	~~1,2,4~~	
1	2,3	2,3
2	1,3	1,3
3	1,2,4	1,2
4	3	~~3~~

(f) Active edge list

Figure 4.15: Illustration of the edge recombination procedure by Liu (2008a).

4.15. Given a population of initial solutions (Figures 4.15a and 4.15b), an initial edge list is constructed (Figure 4.15c) that contains all edges incident to a node in every solution. In the next step, a new solution is constructed starting from a random customer in the tour (customer 0 in Figure 4.15a) based on the active edges in the list. This is achieved by removing all instances of the current customer (i.e., customer 0) in the active edge list, and choosing the next customer among the possibilities to go to starting from the current customer (i.e., customers 1, 2 and 4) that has the smallest number of active edges (customer 4; Figure 4.15d). The previous customer (customer 0) is removed from the list (Figure 4.15f) and the process is repeated (Figure 4.15e) until a complete tour is formed.

Finally, this tour is subjected again to local search, in the same way as described above to further improve upon the current solution. The entire process of selection, edge recombination and local search is repeated until some stopping criterion is met.

MAGX-EE

MAGX-EE is another extension of the memetic algorithm for the PTSP developed by Balaprakash (2010) and Balaprakash et al. (2010). The MAGX-EE is inspired by the MAGX memetic algorithm from Merz and Freisleben (2001), with a set of probabilistic enhancements taken from 2.5-opt-EEais.

Similar to the memetic algorithm of Liu (2008a), MAGX-EE proceeds along the same lines as the procedure outlined in figure 4.14. But in contrast to the memetic algorithm of Liu (2008a), the initialisation phase uses the greedy randomised approach to construction solutions (see GRASP, section 4.5.1) and the local search procedure uses the 2.5-opt-EEais local improvement heuristic (section 4.4.3). Additionally, the edge recombination scheme in the crossover phase is replaced by the perturbation scheme of ILS-EE (section 4.5.1), adding double bridges to a subset of random customers within the population of current solutions.

Applications of MAGX-EE to the PTSP are part of the discussions in Balaprakash (2010) and Balaprakash et al. (2010). Liu (2010) also integrates the nearest neighbour-based initial solution generator of his Liu (2008a) MA into MAGX-EE and tests it against various other metaheuristics. A straightforward deterministic application of MA to the PTSP can be found in Balaprakash et al. (2007).

Genetic algorithms

Genetic algorithms, as a subclass of evolutionary algorithms, was developed by Holland (1975) and also received some attention in the context of PTSPs. Rosenow (1999) introduces the first genetic algorithm for the PTSP and compares it to the results of the exact branch-and-bound approach discussed in section 4.2.3.

As an initial first population generator, Rosenow (1999) applies the random tour heuristic described in section 4.3.5. For the second population, she uses an (unspecified) greedy algorithm which make up 75%-90% of the tours of the second

trial. The remaining proportion of the tours for the second population are generated randomly. In the second phase of the algorithm, edge recombination is applied to two randomly chosen parents of the two initial populations, creating an offspring of half the size. The PTSP objective function is used to compare the final solutions.

Liu (2008b) proposes another genetic algorithm for the PTSP. He essentially relabelled the memetic algorithm (Liu, 2008a) as a genetic algorithm, but added an additional number of selection schemes to the algorithm. In another paper, Liu (2010) considers a number of different initial solution generators – including the type 1 and type 2 nearest neighbour algorithms described in section 4.3.2 – to further investigate the performance of his algorithm. Recently, Smith and Chen (2016) propose genetic minimum matrix search, a metaheuristic which relies on a dimension reduction technique of the edge matrix incorporated into a genetic algorithm.

Scatter search

Liu (2007) develops some hybrid versions of scatter search with the deterministic nearest neighbour rule, threshold accepting (see section 4.5.1) and edge recombination, and applies these to the PTSP. There are no special elements that account for the stochastic features of the PTSP. Liu (2008c) develops another framework which incorporates depth-approximation (section 4.4.4) into scatter search.

Particle swarm optimisation

Particle swarm optimisation (PSO) is a nature-inspired metaheuristic based on the movement of individuals in a swarm. Each individual, or particle, within the swarm has two characteristics that describe it: A position and a velocity. The fitness, i.e., solution quality, of the particles determines which particles are selected for further improvement.

Hybrid multi-swarm particle swarm optimisation

A sophisticated PSO algorithm for the PTSP is developed by Marinakis and Marinaki (2010). Their algorithm is basically a hybrid extension of the ENS-GRASP metaheuristic described in section 4.5.1. Furthermore, instead of relying on only one swarm in the evaluation of solutions as in the classical PSO, their approach relies on multiple swarms – hence the term hybrid multi-swarm PSO, or HybMSPSO.

The initial population of HybMSPSO is generated by ENS-GRASP. Using the pure greedy algorithm and a selection scheme with a restricted candidate list, a number of initial solutions are generated and subsequently submitted to ENS (see section 4.5.1). The solutions are referred to as *particles*; a set of solutions as a *swarm*. The particles are subjected to the next phase of HybMSPSO: Multi-swarm PSO.

The multi-swarm PSO algorithm by Marinakis and Marinaki (2010) starts out by constructing a symmetric position matrix A_{jl} ('adjacency matrix') for every particle. A_{jl} contains the connections with all the other customers in the particle. That is, a particle is characterised by a set of customers with respect to the initial solutions, and the binary elements $a_{jl} \in A_{jl}$ of the adjacency matrix indicate whether a connection is present or not. The binary elements a_{jl} take the value 1 if a connection between customer j and l within the particle exists, and 0 otherwise. The position (i.e., route) s_{ij} of a particle itself corresponding to the encoded matrix A_{jl} is given by a simple path along the customers within the particle, where $i, 1 \leq i \leq N$ is the subscript that denotes the particle number and $j, 1 \leq j \leq N$ denotes the customer number of interest.

The velocity v_{ij} of each particle is related to the probabilities that 0 and 1 occur within matrix A_{jl}. In short, the velocities are recursively updated as a random weighted function of the previous velocity, the last known position s_{ij}, the previous best positions of the particle p_{ij}, and the previous best position of all particles in the swarm p_{gj}. The weights in this function are predefined 'acceleration' constants (corresponding to the position variables) and an 'inertia weight' (corresponding

Method	Strategy	Key characteristic(s)	Limitations	EP
pACS ACS-EE S-ACO	Ant colony optimisation	⋆ Probabilistic adaptation of ACS for the PTSP ⋆ Thorough search of the solution space ⋆ Biases solutions that have not yet been explored	⋆ Requires repeated calculation of the objective function for each iteration; may be slow	2
Stochastic SA	Simulated annealing	⋆ Stochastic adaptation of SA for the PTSP ⋆ Thorough search of the solution space; allows for non-improving solutions ⋆ Can be fast, depending on the implementation	⋆ Takes many iterations before it may convergence ⋆ Performs relatively poor on average	6
ENS-GRASP	GRASP	⋆ Extension of GRASP for the PTSP ⋆ Sophisticated integration of a neighbourhood sorting approach, conceptually robust	⋆ Computational set-up for ENS procedure is quite demanding ⋆ Usually stopped by maximum runtime criterion rather than convergence to an optimal solution	5
HybMSPSO	Particle swarm optimisation	⋆ Adaptation of a hybrid multi-swarm particle swarm optimisation for the PTSP ⋆ Combines the best assets from other PTSP-based metaheuristics ⋆ Solid estimation scheme	⋆ Complex computational set-up ⋆ Conceptually more complex than direct competitors ⋆ Requires repeated comparison of the objective function ⋆ Usually stopped by maximum runtime criterion rather than convergence to an optimal solution	4
MAGX-EE	Memetic algorithm	⋆ Stochastic extension of MA ⋆ Relatively straightforward iterative improvement process ⋆ Easy to implement	⋆ Requires repeated comparison of the objective function ⋆ Usually stopped by maximum runtime criterion rather than convergence to an optimal solution	1
ILS-EE	Iterated local search	⋆ Probabilistic adaptation of ILS ⋆ Easy set-up ⋆ Conceptually simple	⋆ Choice of perturbation function strongly affects results; may not explore entire solution space	3
RRLS-EE	Random restart local search	⋆ Probabilistic adaptation of RRLS ⋆ Easy set-up ⋆ Conceptually very straightforward	⋆ Computational speed depends strongly on quality of initial solution ⋆ May provide very poor results, relies severely on underlying improvement heuristic	N/A

Table 4.6: A comparison of various characteristics of metaheuristic methods for the PTSP. EP: (relative) empirical performance. Relative empirical performance based on empirical studies performed by Marinakis and Marinaki (2010), Balaprakash et al. (2009b) and Balaprakash et al. (2010). Note that performance may be greatly enhanced by using hybrid extensions (e.g. pACS+1-shift instead of simply pACS). In case of multiple stochastic variants, the best possible result among the simple implementations of the metaheuristics is reported. Relative performances are based on average performance measures over the full range of probabilities. That is, it attempts to give an overall idea of the methods; some methods may perform better than others for other input values. Empirical results from RR-LS have been excluded due to limited data availability.

to the velocity variable). The latter weight is defined as a declining function of the performance of the other particles and the number of iterations, such that the algorithm becomes increasingly exploitative (as opposed to explorative) as time progresses. The positions s_{ij} with respect to the adjacency matrix A_{jl} are updated in a similar fashion, by applying a sigmoid function which transforms the velocities v_{ij} of the particle into probability space $[0, 1]$. The full details of the involved computations can be found in Marinakis and Marinaki (2010).

The essence of the multi-swarm PSO process is to find a local optimum for each of the initial swarms generated by the ENS-GRASP procedure. The best particles within each swarm are combined into a new swarm, which is again subjected to the PSO procedure. The whole multi-swarm PSO algorithm is repeated a number of times, and the best solution with respect to the expected length function (3.3) is returned as the final result.

Implementations of HybMSPSO can be found in Marinakis and Marinaki (2009), who apply a PSO algorithm without stochastic enhancements to the PTSP, along with the HybMSPSO procedure of Marinakis and Marinaki (2010). Marinakis and Marinaki (2009) test HybMSPSO empirically against various other metaheuristics. Balaprakash (2010); Balaprakash et al. (2010) and Liu (2010) test HybMSPSO against various probabilistic implementations of the ACO metaheuristic. Cabrera et al. (2012) propose an alternative hybrid implementation of PSO with SA. Marinakis et al. (2015a) propose a tuning algorithm for the parameters of multi-swarm PSO.

Other population search metaheuristics

Yoshitomi et al. (2000) describe genetic algorithms under uncertainty, and briefly mention the possibility of applying GAs to stochastic TSPs. Genetic ordinal optimisation, another genetic algorithm alternative, is the topic of study in Zhang and Wang (2004). Zhang and Wang (2004) apply their method to the stochastic variant of the TSP where travel time instead of customer presence is uncertain. As

both papers do not describe applications to the PTSP, these papers are beyond the scope of this chapter.

4.5.3 A comparison of metaheuristics

Table 4.6 provides a comparison of the various features of metaheuristics. The computational performance of metaheuristics depends largely on their exact implementation, i.e. the (classical) improvement heuristics that are integrated into the metaheuristic frameworks. Additionally, maximum runtime is often used as a stopping criterion for metaheuristic implementations of the PTSP, instead of awaiting convergence.

Among all metaheuristics, the probabilistic adaptations of ant colony optimisation algorithms have deserved the most attention – and as such, have been subjected to the most thorough testing. More recent studies expose that MAGX-EE and ILS-EE can be considered as direct competitors of the ACO-based metaheuristics. It is important to realise, however, that different settings show contrasting results with respect to the best performing metaheuristic. Many different factors, such as the number of customers, the dispersion of the nodes in the graph and the customer presence probabilities may have an influence on the performance of these metaheuristics. Empirical studies of metaheuristics, however, are abundant. Therefore, the reader is referred to the analyses of Bianchi (2006),Balaprakash et al. (2010), and Marinakis and Marinaki (2010) for extensive bechmark tests and empirical comparisons.

4.6 Conclusion

This chapter provides a comprehensive overview of the most important solution methods for the PTSP. It identifies several solution categories, namely exact methods, construction heuristics, improvement heuristics, and metaheuristics. The solution methods within these categories are described in this study.

With respect to exact methods, only one method really stands out: The integer L-shaped method by Laporte et al. (1994). Competitor methods rely on factorial computation effort, and are therefore not practically viable. Other exact methods for stochastic combinatorial optimisation problems that may also result in optimal solutions for the PTSP have not yet been explored, neither theoretically nor empirically. Because the integer L-shaped method is found to be successful for problem instances of up to only c. 50 nodes, other exact methods may be worth exploring. In addition, more sophisticated solution schemes that rely on a posteriori optimisations than the re-optimisation approach (section 4.2.5) are among the future research opportunities. For example, one could think of recombining a pool of several optimal a posteriori candidate solutions into an a priori solution that maximizes the likelihood of being the a priori global optimum.

Classical heuristic methods can be roughly divided into two classes, namely construction heuristics and improvement heuristics. As to the construction heuristics, the supersavings-based heuristic seems to provide relatively the best results. The farthest insertion heuristic follows within close distance. This is an interesting observation, because farthest insertion does not take the stochastic nature of the customer presence into account, unlike most of the other construction heuristics. In addition, the probabilistic adaptations of the savings algorithms seem to be still in its infancy, despite having been around since 1985. Generalisations of the supersavings approach to adapt to new problem settings may lead to new insights.

With regard to the classical improvement methods, the 2.5-opt-EEs heuristic, along with its extensions 2.5-opt-EEas and 2.5-opt-EEais, are among the most reliable methods. These methods do not only produce good results, but are also faster than direct competitors such as 2-p-opt. It is interesting that the probabilistic adaptations of 1-shift did not deserve the same amount of attention as the 2-p-opt and 2.5-opt-EEs approaches, despite its alleged overall better performance than 2-p-opt. Empirical estimation extensions of 1-shift, similar to the

empirical estimation scheme of 2.5-opt to 2.5-opt-EEs, may be worth exploring. In addition, other approximate evaluation schemes than the EEs scheme of 2.5-opt, e.g. 2.5-opt-threshold, have only been tested for low customer probabilities, which leaves another research gap to be explored.

The last category of solution methods discussed in this study are the metaheuristic methods. Metaheuristics can be divided into two main subcategories: Local search metaheuristics and population-based search metaheuristics. Among the metaheuristics, the probabilistic adaptations of ant colony optimisation algorithms deserved the most attention in the context of the PTSP. More recent studies expose that memetic algorithms, in particular MAGX-EE, and iterative local search algorithms, namely ILS-EE, can be considered as direct competitors of the ACO-based metaheuristics. It is important to realise, however, that different settings show contrasting results with respect to the best performing metaheuristic. Many different factors, such as the node probabilities, the number of customers, the dispersion of the nodes in the graph and the customer presence probabilities, seem to affect the eventual outcome of these metaheuristics. Moreover, empirical settings where other factors, such as the number of deterministic customers, play a role are still left to be investigated. All in all, this leaves a great deal of opportunities for future research.

5 | A hyper-heuristic driven construction framework for the PTSP

This chapter proposes a new construction heuristic for the PTSP based on expected savings. I call this heuristic the Generic Savings (GENS) procedure. It is a generalisation of the Supersavings procedure by Jaillet (1985), the Newsave and Globalsave procedures by Jézéquel (1985), and the Probabilistic Clarke and Wright procedure by Rossi and Gavioli (1987), obtained by abstracting common properties from these specific instances. As a result, Supersavings-based procedures are special instances of GENS, which can be obtained by plugging in certain combinations of parameter choices. More generally, the savings algorithms for the PTSP all originate from the Clarke and Wright savings algorithm (Clarke and Wright, 1964) and differ from those designed for the TSP in that they rely on the expected savings, also referred to as Supersavings, rather than (the usual) deterministic savings (Jaillet, 1985).

GENS comes with a set of attractive features. First, it unifies the main design elements of several construction heuristics into one generalised and parameterised framework. Second, it provides a new flexible framework that has the ability to unveil new Supersavings-based algorithms. Last, the empirical results suggest that it has the ability to construct high quality solutions regardless of the characteristics of the underlying problem instance. More specifically, due to the ability of GENS to control its behavior by changing its parameters, GENS can be adapted and

applied to a wide array of different PTSP scenarios. Using an effective parameter tuning algorithm, GENS should theoretically be able to find solutions that are at least as good as the current best fitting Supersavings-based procedures in any setting. In the worst-case scenario, the parameter combination corresponding to the best performing Supersavings-based procedure is selected for GENS, resulting in identical solutions.

In this chapter, I propose to implement GENS within a hyper-heuristic framework to determine the values of its parameters, resulting in a solution framework that I refer to as GENS-H. GENS-H is benchmarked against various other construction heuristics specially designed for the PTSP, namely, supersavings procedures (Jaillet, 1985; Jézéquel, 1985; Rossi and Gavioli, 1987), the radial sort (also referred to as the angular sort) procedure (Bertsimas, 1988), the spacefilling curve procedure (Bartholdi and Platzman, 1982), insertion procedures (Mole and Jameson, 1976), nearest neighbour procedures (Jaillet, 1985; Jézéquel, 1985; Rossi and Gavioli, 1987; Bianchi, 2006; Liu, 2010), and random tour procedures (Bianchi et al., 2002a). Note that, to the best of my knowledge, this study is the first to propose a hyper-heuristic to tune the parameters of an algorithm in the context of stochastic routing problems. So far, hyper-heuristics have merely been applied to the domain of a handful of deterministic and dynamic vehicle routing problems (e.g. Keller and Poli, 2007a,b, 2008a,b,c; Pisinger and Ropke, 2007; Garrido and Castro, 2009; Garrido and Riff, 2010) — see Burke et al. (2013) for an overview. I argue that hyper-heuristics also have a lot of potential in other areas of routing, including stochastic routing. The promising results from the empirical analysis on the performance of GENS-H for the PTSP (section 5.2) support this belief.

Note that the main contribution of this study is methodological. That is, the aim of this study is to demonstrate the power of generalising classes of solution methods and applying hyperheuristics to the stochastic routing domain. As such, the results of the comparative analysis reported in this chapter primarily serve to illustrate

the empirical potential of pursuing such an approach, rather than to outperform state-of-the-art methods.

The remainder of this chapter is structured as follows. Section 5.1 presents the methodological contributions to the PTSP. It is followed by an empirical analysis in section 5.2 along with a discussion of results. Finally, section 5.3 concludes.

5.1 An automated construction framework based on expected savings

In this section, I propose a parameterised and generalised solution framework for the PTSP called the Generic Savings (GENS) procedure along with a hyper-heuristic framework (GENS-H) for its implementation.

5.1.1 The Generic Savings (GENS) procedure

This section proposes a new construction heuristic for the PTSP called GENS. GENS is a parameterised generic construction method based on Supersavings, and a generalisation of the Supersavings-based procedures proposed for the PTSP (see section 4.3.1): the Supersavings procedure, the Newsave procedure, the Globalsave procedure, and the Probabilistic Clarke & Wright procedure (see Appendix B for their pseudo-codes). Observe that, although the Supersavings-based procedures differ slightly from each other in set-up and implementation, the merging operation that is at the heart of each procedure is the same. Since the GENS procedure is a generalisation of the Supersavings-based procedures, the subtour merging operation is also one if its main characteristics.

The description of GENS requires a number of decisions to be made for implementation customisation – see Algorithm 1 for pseudo-code. These decisions represent the parameters of GENS and are discussed hereafter.

γ is the degree of approximation of the actual expected cost function of the

PTSP. The repeated evaluatiuon of the PTSP objective function (3.4) can be a time-consuming task. As the contribution of the terms $p^2(1-p)^r \sum_{j=1}^{n} d_{j,(j+1+r) \bmod n}$ in (3.4) to the total expected length becomes smaller for increasing r, one may consider pruning the first summation of (3.4) beyond some level $\gamma \leq n-2$. Although this induces an error with respect to the true expected length, these approximations may still be sufficiently accurate to compare intermediate candidate solutions.

Coverage Probabilities Criterion This variable is concerned with whether to compute a single a priori PTSP tour or several ones, or equivalently, specify a single homogeneous probability of occurrence of nodes or a set of such probabilities — often referred to as coverage probabilities. I introduce a binary variable, $CoverageProbabilities\_Criterion$, to specify this choice. $CoverageProbabilities\_Criterion = 0$ if a single coverage probability of occurrence of nodes is specified, and $CoverageProbabilities\_Criterion = 1$ if $k > 1$ coverage probabilities $\{\pi_1, \ldots, \pi_i, \ldots, \pi_k\}$ are specified. In the latter case, each tour is also cross-examined with the tours resulting from other probabilities.

Algorithm 1 Pseudo-code of the GENS Procedure

Input:

1: $N \leftarrow$ Set of nodes representing customers and the depot;
2: $D \leftarrow$ Distance matrix between each pair of nodes;
3: $\gamma \leftarrow$ Degree of approximation of the actual expected cost function of the PTSP;
4: $CoverageProbabilities\_Criterion \leftarrow 0$ or 1;
5: $InitialBestSolution\_Criterion \leftarrow 0$ or 1;
6: $Embryo\_Criterion \leftarrow 0$ or 1;
7: $Merge\_Criterion \leftarrow 0$ or 1;
8: $SavingsList\_Criterion \leftarrow 0$ or 1;
9: $\pi \leftarrow \{\pi_1, \ldots, \pi_k\}$;
10: If $Merge\_Criterion = 1$, set $NumberOfSubtoursToMerge$ and θ; // See $Merge\_Criterion$ description

Algorithm 1 Pseudo-code of the GENS Procedure (continued)

Initialisation step:

// Initialise the solution(s) for all $\pi_i \in \pi$ according to the choice of $InitialBestSolution\_Criterion$

if $InitialBestSolution\_Criterion = 0$ **then**

    $\rho_i^{best} \leftarrow \emptyset$; // For all $i = 1, \ldots, k$

    $E[L(\rho_i^{best}, \pi_i)] \leftarrow +\infty$; // For all $i = 1, \ldots, k$

else

    $\rho \leftarrow TSP\_Solution(N, C)$; // Build a deterministic TSP solution

    $\rho_i^{best} \leftarrow \rho$; // For all $i = 1, \ldots, k$

    $E[L(\rho_i^{best}, \pi_i)] \leftarrow E[L(\rho, \pi_i)]$; // For all $i = 1, \ldots, k$

end if

$ClassicalSavingsList \leftarrow Build\_Classical\_SavingsList()$;

Iterative step:

for $i \leftarrow 1$ to k **do** // Loop over the probabilities

    for $g \leftarrow 2$ to γ **do** // Loop over the initial subtour sizes

        $\rho_{(i,g)} \leftarrow Construct\_Initial\_Solution(g, ClassicalSavingsList)$;

        $SupersavingsList \leftarrow Build\_Supersavings\_List(\rho_{(i,g)}, \pi_i)$;

        if $Embryo\_Criterion = 1$ **then**

            $\rho_{(i,g)} \leftarrow Initialise\_Embryo(\rho_{(i,g)}, ClassicalSavingsList)$;

        end if

        // Continue merging until one complete tour is obtained

        while number of subtours in $\rho_{(i,g)} > 1$ **do**

            // Select the candidate subtours for the merge operation

            $Subtours \leftarrow Select\_Merge\_Candidates(\rho_{(i,g)}, SupersavingsList, Merge\_Criterion, NumberOfSubtoursToMerge, \theta)$;

            $\rho_{(i,g)} \leftarrow Merge(Subtours)$;

            $SupersavingsList \leftarrow Reduce(SupersavingsList, \rho_{(i,g)})$;

Algorithm 1 Pseudo-code of the GENS Procedure (continued)

```
            if SavingsList_Criterion=1 then
                SupersavingsList ← Update(SupersavingsList, ρ_(i,g));
            end if
        end while
        // Update the best solution(s) known so far by cross-checking with other
           probabilities
        for j ← 1 to k do
            if E[L(ρ_(i,g), π_j)] < E[L(ρ_j^best, π_j)] then
                ρ_j^best ← ρ_(i,g);
            end if
        end for
    end for
end for
```

Output:

An a priori PTSP tour or k a priori PTSP tours $\rho^{best} = (\rho_1^{best}, \ldots, \rho_k^{best})$ each associated with a different coverage probability $(\pi_1, \ldots, \pi_k)$, along with its (their) expected costs $E[L(\rho_i^{best})]$, $i = 1, \ldots, k$.

Initial Best Solution Criterion The third decision that one has to make is concerned with the initialisation of the best solution found so far for each coverage probability. I introduce a binary variable, $InitialBestSolution\_Criterion$, to specify how this decision is made.

$InitialBestSolution\_Criterion = 0$ if the best solution found so far for each coverage probability is initialised to an empty tour and the corresponding expected cost to $+\infty$, and $InitialBestSolution\_Criterion = 1$ if the best solution found so far for each coverage probability is initialised as follows: Construct a Hamiltonian tour for the deterministic TSP version of the problem, ρ, using an appropriate TSP construction heuristic, optionally improved by a local search mechanism. Then compute $E[L(\rho, \pi_i)]$, the expected length of ρ for each π_i, $i = (1, \ldots, k)$, using (3.4). Initialise the

parameter ρ_i^{best}, which keeps track of the best a priori tour found so far for coverage probability π_i, to ρ for all $i = (1, \ldots, k)$. Initialise its corresponding expected length, denoted $E[L(\rho_i^{best}, \pi_i)]$, to $E[L(\rho, \pi_i)]$ for all i.

Embryo Criterion Another decision that one has to make is concerned with the choice of whether to make use of an embryo – two subtours merged using the deterministic Clarke and Wright (1964) savings approach – as an initial largest subtour to expand further or not. Here, the term *largest subtour* is defined as the subtour that has the largest number of included edges. A binary variable $Embryo\_Criterion$ is introduced to represent this choice.

$Embryo\_Criterion = 0$ if no embryo is used to initialise the largest subtour, and $Embryo\_Criterion = 1$ if an embryo is used to initialise the current largest subtour.

Merge Criterion The next decision that one has to make is concerned with the choice of the criterion according to which the subtours to be merged are chosen. There are many possible ways of choosing which couple of subtours to merge. The Supersavings procedures proposed so far consider merging either *any* pair of subtours with maximum deterministic or stochastic savings, or merging the current largest subtour and any other subtour corresponding to a maximum deterministic or stochastic savings. Moreover, the existing savings heuristics only merge one pair of subtours at a time. This limitation is relaxed in GENS by allowing any number of subtours to be merged at a time. All possible ways of merging subtours are explored either in a deterministic and exhaustive way, or in a probabilistic way. The binary variable $Merge\_Criterion$ is introduced to specify this choice, and should be supplied along with a numerical value for another decision variable $NumberOfSubtoursToMerge$, the number of subtours to be merged at a time.

$Merge\_Criterion = 0$ if $NumberOfSubtoursToMerge$ subtours with maximum deterministic or stochastic savings are chosen for a merge operation, and $Merge\_Criterion = 1$, if, amongst the subtours with maximum deterministic or stochastic savings, the θ proportion of

GENS parameters	Supersavings	Newsave	Globalsave	PCW
Value of γ	$n-1$	$n-1$	$n-1$	2
$CoverageProbabilities\_Criterion$	0	0	1	0
$Merge\_Criterion$	0	1	1	0
$NumberOfSubtoursToMerge$	2	2	2	2
Value of θ	–	0	0	–
$SavingsList\_Criterion$	0	0	0	1
$Embryo\_Criterion$	0	1	1	0
$InitialBestSolution\_Criterion$	0	0	1	0

Table 5.1: Comparative Analysis of Savings Procedures Designs

$NumberOfSubtoursToMerge$ smallest subtours are chosen to merge with the $1-\theta$ proportion of $NumberOfSubtoursToMerge$ largest subtours. Here, the term *smallest subtours* refers to the subtours that have the least number of included edges.

Savings List Criterion The final decision that one has to make is concerned with the choice of the criterion according to which the list of stochastic savings is to be maintained. The Supersavings procedures proposed so far consider either reducing the list of savings after each merge operation without updating the savings values, or updating both the savings list and its content after each merge operation. The binary variable $SavingsList\_Criterion$ is introduced to specify this choice.

$SavingsList\_Criterion = 0$ if the list is only reduced after each merge operation, and $SavingsList\_Criterion = 1$ if the list is both reduced and its entries updated after each merge operation.

Note that the Supersavings procedure of Jaillet (1985), the Newsave procedure of Jézéquel (1985), the Globalsave procedure of Jézéquel (1985), and the Probabilistic Clarke & Wright procedure of Rossi and Gavioli (1987) are all special instances of the GENS procedure (cf. pseudocode 1) — see Table 5.1 for a snapshot of their comparative design analysis.

5.1.2 GENS-H: A hyper-heuristic framework for automating the implementation of GENS

In practice, experienced analysts might supply a good set of parameters to run GENS. I refer to this approach as the *manual* implementation of GENS. One however might be better off optimizing the choice of GENS parameters in an objective fashion. I refer to this approach as the *automated* implementation of GENS. In this section, I propose a hyper-heuristic framework for the automated implementation of GENS – see Algorithm 2 for pseudo-code.

Hyper-heuristics are high-level heuristics that manage a set of low-level heuristics in order to find or design a good solution method for a combinatorial optimisation problem by only making use of limited problem-specific information (Chaklevitch and Cowling, 2008). Hyper-heuristics can be roughly divided into two main categories, namely, heuristic selection methodologies, and heuristic generation methodologies (Burke et al., 2013). Whereas heuristic selection methodologies are used to select a set or sequence of appropriate low-level heuristics from a range of candidate heuristics, heuristic generation methodologies are used to generate new heuristics from a set of candidate components. Heuristic selection methodologies, the focus of this study, can be further separated into hyper-heuristics that act on low-level construction heuristics, and hyper-heuristics that act on low-level improvement or perturbative heuristics (Burke et al., 2013). Hyper-heuristics are found to be attractive because of their ability to adapt to the underlying problem — possibly through learning mechanisms — without performing an exhaustive search across the entire low-level search space. Contrary to their deterministic counterparts, I argue that stochastic routing problems may especially benefit from the hyper-heuristic adaptivity due to the typical lack of a known exact solution, thus requiring a more sophisticated exploration of the low-level search space in order to obtain a good solution.

Although hyper-heuristics are primarily designed to select an appropriate set or

Algorithm 2 Pseudo-code of the GENS-H Procedure

Input:

$HL \leftarrow$ Choice of the high level heuristic (RDM, ILS, VNS or TS);
$N \leftarrow$ Set of nodes representing customers and the depot;
$C \leftarrow$ Cost or distance matrix between each pair of nodes;
$\pi \leftarrow$ Set of coverage probabilities $\{\pi_1, \ldots, \pi_k\}$;
$max\_HL\_iterations \leftarrow$ Maximum number of evaluations by the high-level heuristic;
$max_RS_iterations \leftarrow$Number of iterations before the search is restarted if $HL = ILS$ or $HL = VNS$;

Initialisation step:

$\zeta_0 \leftarrow$ Generate_Random_Parameter_Set();
$\rho_0 \leftarrow GENS(\zeta_0, N, C, \pi)$;
$\zeta^{best} \leftarrow \zeta_0, \zeta \leftarrow \zeta_0$;
$\rho^{best} \leftarrow \rho_0, \rho \leftarrow \rho_0$;
if $HL = TS$ **then**
    $Tabu\_List \leftarrow \emptyset$; // Initialises tabu list to an empty list
else if $HL = ILS$ or $HL = VNS$ **then**
    $RS\_iterations \leftarrow 0$;
end if
$HL\_iterations \leftarrow 0$;
$stop \leftarrow false$;
$paramsToChange \leftarrow 2$; // Defines neighbourhood structure

Iterative step:

// Repeat until a stopping criterion is met
while $stop = false$ **do**
    // Apply the move operator to the current parameter set while possibly accounting for already explored moves
    if $HL = TS$ **then**
        $\zeta' \leftarrow M(\zeta, Tabu\_List, paramsToChange)$;

Algorithm 2 Pseudo-code of the GENS-H Procedure (continued)

else
  $\zeta' \leftarrow M(\zeta, paramsToChange)$;
end if
if $HL = ILS$ or $HL = VNS$ **then**
  $RS\_iterations \leftarrow RS\_iterations + 1$;
end if
$\rho' \leftarrow GENS(\zeta', N, C, \pi)$;
$Accept\_Move \leftarrow Move\_Acceptance\_Criterion(HL, \rho, \rho')$;

// Test the candidate tour against the current tour using the move acceptance criterion defined by the high-level heuristic
if $Accept\_Move = true$ **then**
  $\rho \leftarrow \rho'$;
  $\zeta \leftarrow \zeta'$;
end if

// Update the best solution found so far, if necessary
if $E[L(\rho')] < E[L(\rho^{best})]$ **then**
  $\rho^{best} \leftarrow \rho'$;
  $\zeta^{best} \leftarrow \zeta'$;
end if

// Update high-level heuristic-specific features
if $RS\_Iterations = max\_RS\_iterations$ or $Accept\_Move = false$ **then**
  if $HL = ILS$ **then**
    $\zeta \leftarrow \zeta_0$; // Restart the search
    $RS\_iterations \leftarrow 0$;
  else if $HL = VNS$ **then**
    $\zeta \leftarrow \zeta_0$; // Restart the search

Algorithm 2 Pseudo-code of the GENS-H Procedure (continued)

52: $paramsToChange \leftarrow Update\_Parameters\_To\_Change(...$
$paramsToChange)$;
53: $RS\_iterations \leftarrow 0$;
54: **end if**
55: **end if**
56: // Update the stopping criterion according to the criterion defined by the high-level heuristic
57: $HL\_iterations \leftarrow HL\_iterations + 1$;
58: $stop \leftarrow Update\_Stopping\_Criterion(HL\_iterations, max\_HL\_iterations, Accept\_Move)$;
59: **end while**

Output:

60: The results from the high-level heuristic search procedure:
61: A set of parameter values ζ^{best} corresponding to the best parameter configuration of GENS found during the search;
62: The results from the low-level heuristic, namely GENS:
63: An a priori PTSP tour or k a priori PTSP tours ρ_i^{best} $(i = 1,\ldots,k)$, each associated with a different coverage probability, along with its (their) expected costs $E[L(\rho_i^{best})]$, $i = 1,\ldots,k$.

sequence of low-level heuristics, they can also be used to tune parameters. For this purpose, instead of acting on a search space consisting of low-level heuristics, the hyper-heuristic must now act on a search space consisting of the parameter values of GENS. From a pure hyper-heuristic perspective, this implies that we treat every heuristic instance that results from passing a particular set of parameter values to GENS as a separate low-level constructive heuristic. As a result, the hyper-heuristic acts on the parameter space of GENS by applying a move operator M to its current

values. More specifically, given the vector of parameters of GENS,

$$\begin{aligned}\zeta = (&CoverageProbabilities\_Criterion,\\ &InitialBestSolution\_Criterion,\\ &Embryo\_Criterion,\\ &Merge\_Criterion,\\ &SavingsList\_Criterion,\\ &\gamma, \theta, NumberOfSubtoursToMerge),\end{aligned}$$

the move operator randomly chooses a subset of two or more of the GENS parameters and changes its current values; i.e., it applies M to ζ, denoted $M(\zeta)$. In case a binary parameter in ζ is selected, its current setting is simply switched by M from 0 to 1 or 1 to 0. In case an integer variable in ζ is selected by M, its current value is increased or decreased so as to reflect a single step in the parameter space. More specifically, if the parameter γ or $NumberOfSubtoursToMerge$ is selected, its current value is simply increased or decreased by 1 while accounting for the bounds of the selected parameter. In case θ — the proportion of smallest number of subtours to merge among $NumberOfSubtoursToMerge$ -– is selected, the increase (decrease) required for a single step is chosen so as to reflect exactly one extra (one less) number of small subtours to merge among the current value of $NumberOfSubtoursToMerge$. The neighbourhood N of ζ, denoted $N(\zeta)$, constitutes all parameter values that can be reached by applying M to ζ.

The Random descent method, Iterated Local Search (Lourenco et al., 2010), Tabu Search (Glover, 1989) and Variable Neighbourhood Search (Mladenović and Hansen, 1997) have been selected as high-level heuristics for the analysis. Each high-level heuristic is applied separately to the low-level heuristic, GENS; consequently, each high-level heuristic generates its own set of results. The high-level methods are implemented as follows.

Random Descent Method (RDM) RDM searches the parameter space in a random

fashion. The move it carries out consists of randomly changing the values of any subset of parameters in ζ by a single step size. The move is accepted if the resulting solution is a better neighbour with respect to the PTSP objective function; thus, the search might get stuck in a local optimum. The search stops if the selected move does not improve the current solution, or after a pre-specified number of iterations is reached.

Iterated Local Search (ILS) To avoid getting stuck in a local optimum, ILS repeatedly perturbs the seed solution and applies RDM to locally improve it until a pre-specified number of iterations is reached. Hence, its move also consists of randomly selecting a subset of parameters in ζ, and changing their values according to the move defined above. The acceptance of the moves is handled by RDM.

Tabu Search (TS) TS makes use of the same neighbourhood structure as RDM and ILS in that its moves involve changing a subset of parameters in ζ at a time. TS is equipped with a search memory known as a tabu list along with an aspiration criterion to prevent the search process from remaining stuck in a local optimum. In sum, recent moves are added to the tabu list to prevent them from being used in the short run unless the aspiration criterion overrides this restriction. The algorithm stops after a pre-specified number of iterations is reached.

Variable Neighbourhood Search (VNS) To avoid getting stuck in a local optimum, VNS systematically changes neighbourhood structures during a local search process. The implementation of VNS uses moves that entail changing anywhere between 2 and 5 parameters of ζ at a time; hence it uses four neighbourhood structures. The search with each neighbourhood structure is performed using RDM; thus, the acceptance of moves is handled by RDM. In sum, VNS changes the neighbourhood structure each time it encounters a local optimum. It starts out with moves that change only two parameters at a time and increases, if necessary, such number of changes one extra parameter at a time to five parameters, thereby exploring increasingly larger neighbourhoods. The choice of neighbourhood structure is reset to

the simplest one once an improved solution is found. The algorithm stops after a pre-specified number of iterations is reached.

The high-level heuristics listed above are all incorporated into a generic hyper-heuristic framework, which I refer to as GENS-H – See Algorithm 2 for pseudo-code. Note that this presentation of GENS-H is inspired by Qu and Burke (2009) and is used for compactness purposes only. A schematic representation of GENS-H is given in Figure 5.1.

5.2 Results

5.2.1 Implementation

For the experimental set-up I choose several instances from the TSPLIB library ranging from 51 to 783 customers with different degrees of dispersion across the plane. The experiments are run on a set of homogeneous customer probabilities $\pi_i = \{0.1, 0.2, \ldots, 0.9\}$ with a depot at node 1 for every instance. That is, $p_1 = 1$, whereas $p_j = \pi_i$ for the remaining nodes $j \in \{2, \ldots, n\}$. Euclidean distances between customers are maintained as the costs of travel between two customers throughout the experiments. All heuristics are programmed in C# and experiments are run on an Intel(R) Core(TM) i5-3470 at 3.20GHz computer.

The following benchmark heuristics were included in the experiments: the Supersavings algorithm (Jaillet, 1985), the Newsave and Globalsave algorithms (Jézéquel, 1985), the Probabilistic Clarke-Wright (PCW) (Rossi and Gavioli, 1987), radial sort or angular sort (Bertsimas, 1988), the spacefilling curve (SFC) heuristic (Bartholdi and Platzman, 1982; Bertsimas, 1988), farthest insertion and nearest insertion (Mole and Jameson, 1976), the almost nearest neighbour algorithm or probabilistic nearest neighbour algorithm (Jaillet, 1985; Jézéquel, 1985) along with an extension (Jaillet, 1985; Jézéquel, 1985) and a modification (Rossi and Gavioli, 1987), a deterministic nearest neighbour algorithm based on expected lengths (Bianchi, 2006), two different nearest neighbour algorithms with random

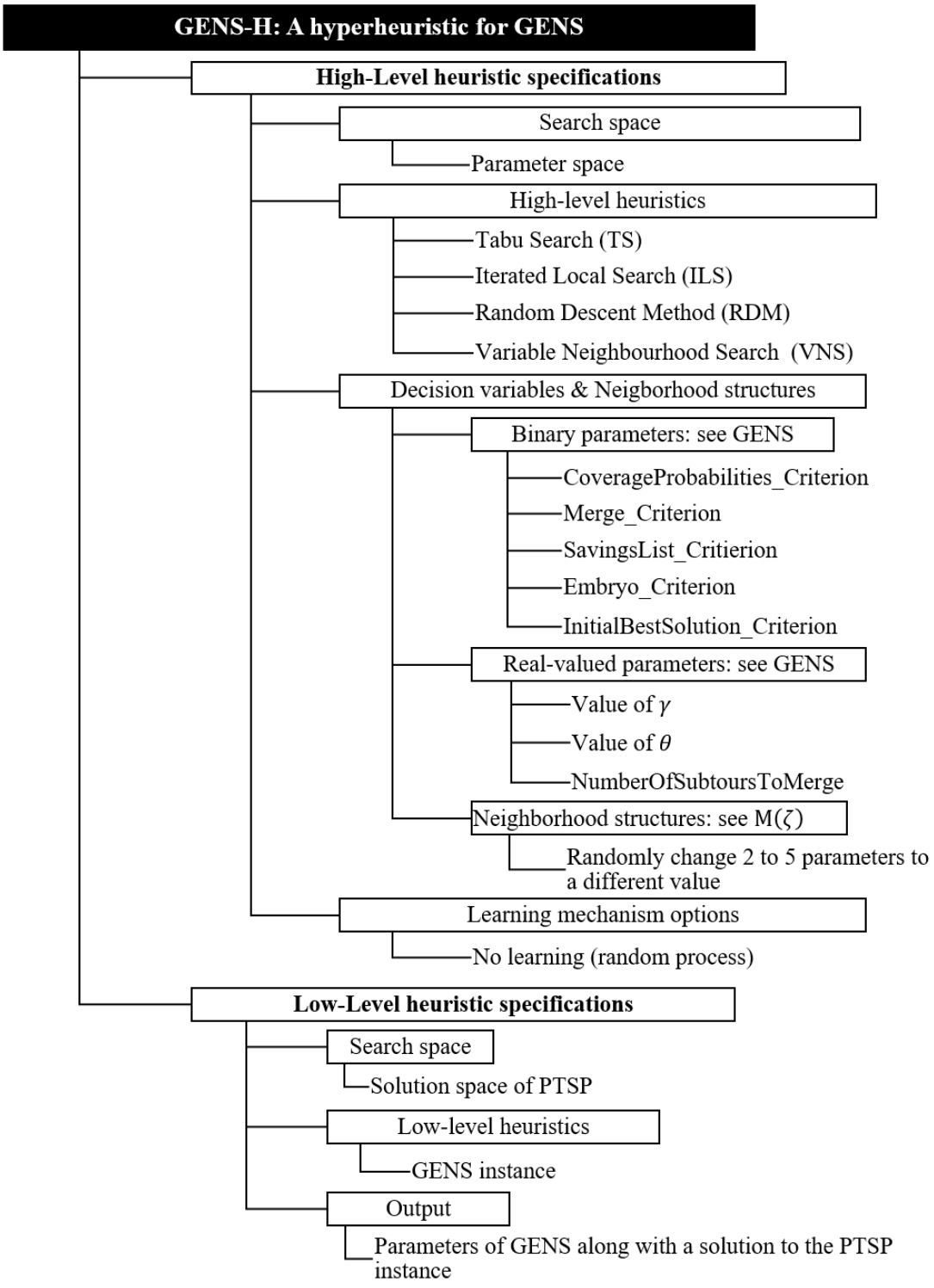


Figure 5.1: A schematic representation of of the different features and components constituting GENS-H.

additions (Liu, 2010), a random tour, and the random best algorithm (Bianchi et al., 2002a).

I restrict the parameter space of the Supersavings-based procedures to $2 \leq \gamma \leq 6$, and $2 \leq NumberOfSubtoursToMerge \leq \min(\frac{n-1}{g-1}, 10)$ based on initial experiments. I also restrict the subset of parameters to change at a time for the high-level heuristic of GENS-H to 2 for RDM, ILS and TS. When $InitialBestSolution\_Criterion$ is set to 1, one could initialise ρ_i^{best} with any tour construction heuristic; however, in the experiments it is initialised to the tour produced by farthest insertion for each π_i. The choice for farthest insertion is motivated by the empirical performance of this heuristic.

Note that if GENS-H chooses to set $InitialBestSolution\_Criterion$ to 1, the expected length found should never exceed the expected length produced by farthest insertion. In addition, the generic design of GENS and its ability to transform into any of the special cases represented by the set of Supersavings-based procedures (cf. Table 5.1) implies that – in theory – the result produced by GENS-H should not be worse than the best result produced among the set of Supersavings-based procedures. That is, in the worst-case scenario GENS-H could simply pick the parameter values that lead to one of the special instances of GENS (see Table 5.1), thus generating the same solution.

Observe that this is merely a theoretical observation — if the high-level search heuristic is not explorative enough, results could potentially be worse. Moreover, because of the mutual exclusivity between the search space of the high-level heuristic and the search space of the low-level heuristic, it is hard to define what constitutes 'explorative': Exploring a large neighbourhood in the high-level search space, e.g., changing many parameters at the same time, does not necessarily imply exploring a large neighbourhood in the low-level search space. For example, it is not guaranteed that changing 4 parameters of ζ at a time would also produce a more explorative search of the low-level space than changing only 1 parameter. On

the one hand, an exhaustive search of all possible high-level parameter choices would be explorative from both the high-level and low-level perspectives, yet computationally unattractive. On the other hand, pinpointing a particular subset of parameters upfront that is believed to result in a 'sufficiently' extensive exploration of the low-level search space would be computationally more attractive, but would require a priori knowledge of GENS-H' behaviour.

5.2.2 GENS-H versus other Supersavings-based Procedures

Table 5.2 reports the expected lengths of the PTSP tours calculated according to (3.4) for a range of coverage probabilities obtained with GENS-H, where RDM, ILS, TS and VNS are used as high-level heuristics to guide the search for an optimal or near-optimal vector of parameters for GENS, and the existing supersavings-based procedures; that is, Supersavings algorithm (Sup) of Jaillet (1985), the Newsave (New) and Globalsave (Glo) algorithms of Jézéquel (1985), and the Probabilistic Clarke-Wright savings algorithm (PCW) of Rossi and Gavioli (1987). The best (in casu: smallest) solution in terms of expected length (3.4) is highlighted in boldface typesetting.

The empirical results, as outlined in Table 5.2, demonstrate that, in general, GENS either outperforms the existing Supersavings heuristics or delivers the same solution. In fact, for some combinations of coverage probabilities and TSP problem instances, GENS delivers the same solution as the existing Supersavings heuristics. However, in most cases, there is clear evidence that GENS delivers a better solution when fed with the right vector of parameters. Note that any solution produced by GENS, under any high-level heuristic for guiding the search in its parameter space, that outperforms existing Supersavings heuristics uses a vector of parameters that is different from the ones used by the existing Supersavings heuristics. Therefore, the implementation of GENS within a hyper-heuristic framework helps to unveil new variants of supersavings heuristics. In sum, the results suggest that the optimisation of the parameters of generic procedures such as GENS under a hyper-heuristic framework is a promising research avenue. As to the relative

Instance	p	RDM	TS	ILS	VNS	Sup	New	Glo	PCW
eil51	0.1	139.41	**139.17**	**139.17**	140.28	142.53	142.47	142.47	**139.17**
	0.2	**205.00**	**205.00**	**205.00**	**205.00**	211.52	212.25	212.25	211.09
	0.3	**253.52**	**253.52**	**253.52**	**253.52**	265.24	264.79	264.79	261.73
	0.4	**295.62**	**295.62**	**295.62**	**295.62**	306.44	306.18	306.18	309.14
	0.5	**331.74**	**331.74**	**331.74**	**331.74**	338.46	338.46	338.46	356.93
	0.6	**358.05**	**358.05**	**358.05**	**358.05**	365.97	365.90	365.90	382.69
	0.7	**382.47**	**382.47**	**382.47**	**382.47**	390.58	390.98	390.98	406.78
	0.8	**405.69**	**405.69**	**405.69**	**405.69**	413.37	413.37	413.37	440.15
	0.9	**428.10**	**428.10**	**428.10**	**428.10**	**428.10**	433.93	433.93	438.78
kroA100	0.1	**9,507.26**	**9,507.26**	9,530.55	9,568.61	9,627.95	9,745.24	9,745.24	9,528.58
	0.2	13,334.84	**12,448.99**	12,602.01	12,597.66	12,644.61	12,796.30	12,796.30	12,637.34
	0.3	14,845.51	14,845.51	**14,788.62**	14,845.51	14,845.51	14,845.51	14,845.51	14,954.17
	0.4	**16,462.33**	**16,462.33**	**16,462.33**	**16,462.33**	**16,462.33**	**16,462.33**	**16,462.33**	17,411.11
	0.5	**17,832.62**	**17,832.62**	**17,832.62**	**17,832.62**	**17,832.62**	**17,832.62**	**17,832.62**	18,794.16
	0.6	20,480.26	**19,043.42**	**19,043.42**	**19,043.42**	**19,043.42**	**19,043.42**	**19,043.42**	21,292.17
	0.7	**20,144.44**	**20,144.44**	**20,144.44**	**20,144.44**	**20,144.44**	**20,144.44**	**20,144.44**	22,193.95
	0.8	**21,166.81**	**21,166.81**	**21,166.81**	**21,166.81**	**21,166.81**	**21,166.81**	**21,166.81**	22,257.04
	0.9	**22,130.46**	**22,130.46**	**22,130.46**	**22,130.46**	**22,130.46**	**22,130.46**	**22,130.46**	24,013.15
eil101	0.1	**211.28**	**211.28**	213.96	213.96	213.96	218.15	218.15	212.78
	0.2	311.99	**305.04**	**305.04**	**305.04**	307.42	311.32	311.32	321.54
	0.3	**375.72**	**375.72**	**375.72**	**375.72**	377.21	386.03	386.03	410.47
	0.4	454.45	**435.43**	**435.43**	**435.43**	436.02	452.22	452.22	474.02
	0.5	**487.18**	**487.18**	**487.18**	**487.18**	487.38	511.27	511.27	529.01
	0.6	**535.07**	**535.07**	**535.07**	**535.07**	551.12	551.12	551.12	583.95
	0.7	597.65	**580.63**	**580.63**	**580.63**	584.75	584.75	584.75	620.90
	0.8	**615.95**	**615.95**	**615.95**	**615.95**	**615.95**	**615.95**	**615.95**	**657.19**
	0.9	**645.54**	**645.54**	**645.54**	**645.54**	**645.54**	**645.54**	**645.54**	**686.71**
ch150	0.1	2,679.96	**2,674.33**	2,674.68	2,674.68	2,710.32	2,704.02	2,704.02	2,680.02
	0.2	**3,617.91**	**3,617.91**	3,667.64	**3,617.91**	3,751.20	3,813.04	3,813.04	3,806.19
	0.3	4,617.97	**4,366.72**	**4,366.72**	4,380.13	4,380.13	4,609.58	4,609.58	4,558.31
	0.4	**4,935.82**	5,012.95	**4,935.82**	**4,935.82**	5,033.66	5,237.22	5,237.22	5,297.65
	0.5	5,455.49	**5,442.57**	**5,442.57**	**5,442.57**	5,462.39	5,616.73	5,616.73	5,636.69
	0.6	6,250.98	**5,860.31**	**5,860.31**	**5,860.31**	**5,860.31**	5,939.19	5,939.19	6,111.20
	0.7	**6,226.45**	**6,226.45**	**6,226.45**	**6,226.45**	**6,226.45**	**6,226.45**	**6,226.45**	6,742.86
	0.8	6,967.19	**6,490.05**	**6,490.05**	**6,490.05**	**6,490.05**	**6,490.05**	**6,490.05**	6,978.08
	0.9	**6,736.36**	**6,736.36**	**6,736.36**	**6,736.36**	**6,736.36**	**6,736.36**	**6,736.36**	7,206.12
d198	0.1	**9,975.09**	**9,975.09**	**9,975.09**	9,998.01	10,394.98	10,057.57	10,057.57	10,065.02
	0.2	11,509.65	11,595.24	**11,509.52**	11,522.53	12,330.03	11,738.08	11,738.08	11,706.78
	0.3	12,694.47	12,567.12	**12,565.37**	**12,565.37**	13,565.81	12,831.03	12,831.03	12,883.30
	0.4	**13,450.52**	13,508.32	13,467.84	13,458.76	14,379.29	13,718.92	13,718.92	13,935.93
	0.5	14,276.34	**14,242.28**	14,245.08	14,245.08	15,060.10	14,472.66	14,472.66	14,552.57
	0.6	14,999.38	15,001.82	**14,911.64**	**14,911.64**	15,653.93	15,130.64	15,130.64	15,171.76
	0.7	15,593.96	15,593.96	**15,521.57**	**15,521.57**	16,185.78	15,716.36	15,716.36	15,680.48
	0.8	16,126.30	16,126.30	**16,091.42**	**16,091.42**	16,670.62	16,185.92	16,185.92	16,362.47
	0.9	**16,608.87**	**16,608.87**	**16,608.87**	**16,608.87**	17,117.95	16,714.89	16,714.89	16,731.11
pr439	0.1	**52,693.03**	**52,693.03**	**52,693.03**	**52,693.03**	53,805.25	55,132.29	55,132.29	54,532.01
	0.2	**69,836.51**	**69,836.51**	**69,836.51**	**69,836.51**	71,147.48	73,067.56	73,067.56	76,017.11
	0.3	**80,900.61**	**80,900.61**	**80,900.61**	**80,900.61**	83,772.12	83,772.12	83,772.12	89,936.49
	0.4	**89,619.79**	**89,619.79**	**89,619.79**	**89,619.79**	91,894.78	91,894.78	91,894.78	99,487.35
	0.5	101,542.04	96,918.32	96,918.32	**96,826.79**	98,508.70	98,508.70	98,508.70	108,144.50
	0.6	103,263.13	**102,935.88**	103,263.13	103,263.13	103,263.13	104,135.87	104,135.87	114,673.57
	0.7	108,934.76	108,934.76	**108,404.18**	**108,404.18**	109,061.95	109,061.95	109,061.95	123,601.59
	0.8	113,456.13	**113,416.38**	113,456.13	113,456.13	113,456.13	113,456.13	113,456.13	126,346.08
	0.9	117,425.73	117,425.73	**117,401.73**	117,425.73	117,425.73	117,425.73	117,425.73	131,019.85

Table 5.2: Results for GENS-H versus Supersavings-based construction heuristics

Instance	p	RDM	TS	ILS	VNS	Sup	New	Glo	PCW
rat783	0.1	**3,718.96**	**3,718.96**	**3,718.96**	**3,718.96**	**3,718.96**	3,968.24	3,968.24	3,816.42
	0.2	**5,186.55**	5,193.31	**5,186.55**	**5,186.55**	5,223.53	5,452.21	5,452.21	5,528.47
	0.3	**6,207.47**	**6,207.47**	**6,207.47**	**6,207.47**	6,260.59	6,396.80	6,396.80	6,661.05
	0.4	**7,004.56**	**7,004.56**	**7,004.56**	**7,004.56**	7,102.20	7,114.06	7,114.06	7,577.88
	0.5	**7,651.45**	**7,651.45**	**7,651.45**	**7,651.45**	7,702.97	7,702.97	7,702.97	8,195.21
	0.6	**8,208.54**	**8,208.54**	**8,208.54**	**8,208.54**	8,208.54	8,208.54	8,208.54	8,933.89
	0.7	**8,655.53**	**8,655.53**	**8,655.53**	**8,655.53**	8,655.53	8,655.53	8,655.53	9,387.41
	0.8	**9,058.67**	**9,058.67**	**9,058.67**	**9,058.67**	**9,058.67**	**9,857.86**	**9,058.67**	9,857.86
	0.9	10,820.10	**9,427.16**	**9,427.16**	**9,427.16**	9,427.16	9,427.16	9,427.16	10,324.69

Table 5.2: Results for GENS-H versus Supersavings-based construction heuristics (continued).

performance of the high-level heuristics used for implementing GENS-H, ILS, TS and VNS seem to outperform RDM on some combinations of coverage probabilities and problems, which could be explained by the design limitations of classical local search algorithms such as RDM. However, there is no clear winner amongst ILS, TS and VNS.

Last, but not least, I would like to point out that the implementation decisions of high-level heuristics such as the stopping criteria or the degree of sophistication of neighbourhood structure(s) could affect the performance of GENS when these decisions lead to premature convergence or limit the degree of diversification of the search process.

5.2.3 GENS-H versus procedures not based on Supersavings

This section reports on the empirical results of benchmarking GENS-H against various construction heuristics that are not based on the concept of savings; that is, radial sort (RS) heuristic (Bertsimas, 1988), the spacefilling curve (SFC) heuristic (Bartholdi and Platzman, 1982; Bertsimas, 1988), and the farthest insertion (FI) and nearest insertion (NI) heuristics (Mole and Jameson, 1976). The results of other benchmarked methods, including the almost nearest neighbour algorithm or probabilistic nearest neighbour algorithm (Jaillet, 1985; Jézéquel, 1985), an extension (Jaillet, 1985; Jézéquel, 1985) and a modification (Rossi and Gavioli, 1987), a deterministic nearest neighbour algorithm based on expected lengths

(Bianchi, 2006), two different nearest neighbour algorithms with random additions (Liu, 2010), and the random best algorithm (Bianchi et al., 2002a) can be found in Appendix C.

Table 5.3 reports a sample of the empirical results on the performance of GENS-H versus RS, SFC, FI and NI for TSPLIB instances ranging from 51 to 783 nodes, and over a homogeneous probability set $p = \{0.1, \ldots, 0.9\}$. These tables provide the expected lengths of PTSP tours obtained with GENS-H and the above mentioned benchmarks, where the best solution is highlighted in boldface typesetting.

Our results demonstrate again that, in general, the different implementations of GENS-H either outperform construction heuristics that are not based on the concept of savings or deliver the same solution. There are, however, a few exceptions. First, the radial sort heuristic outperforms GENS-H for coverage probabilities 0.1 and 0.2 on TSPLIB instance eil51, and for coverage probability 0.1 on TSPLIB instance eil101. Second, the spacefilling curve heuristic outperforms GENS-H for coverage probabilities 0.3, 0.4, and 0.5 on instance eil51 and for coverage probability 0.1 on instance ch150. These exceptions can be explained as follows: Radial sort is often found to produce good results for $\pi \leq 0.1$ due to the close resemblance of its solution to the near-optimal star-shape for low probabilities (Bertsimas, 1988). Its performance therefore does not only strongly depend on the customer's probability of requiring a visit, but also on the dispersion of customers on the plane. The results confirm that the results of radial sort quickly deteriorate as probabilities get larger, or when the customers in the plane are highly clustered.

The spacefilling curve heuristic also suffers from the latter problem, as it relies on recursively partitioning the graph into smaller subsquares and visiting the customers in these subsquares in a predetermined sequence. Hence, for larger problems or highly clustered ones, the likelihood that two or more nodes fall within the same subsquare increases. This makes it harder for the spacefilling curve

Instance	p	RDM	TS	ILS	VNS	RS	SFC	FI	NI
eil51	0.1	139.41	139.17	139.17	140.28	**137.45**	138.61	140.83	139.93
	0.2	205	205	205	205	**201.1**	202.07	205	207.03
	0.3	253.52	253.52	253.52	253.52	256.09	**250.67**	253.52	260.1
	0.4	295.62	295.62	295.62	295.62	310.73	**291.64**	295.62	305.64
	0.5	331.74	331.74	331.74	331.74	366.25	**327.05**	333.38	345.37
	0.6	**358.05**	**358.05**	**358.05**	**358.05**	422.67	358.32	367.67	380.59
	0.7	**382.47**	**382.47**	**382.47**	**382.47**	479.87	386.57	399.06	412.4
	0.8	**405.69**	**405.69**	**405.69**	**405.69**	537.76	412.59	428.08	441.69
	0.9	**428.1**	**428.1**	**428.1**	**428.1**	596.39	436.94	455.24	469.08
kroA100	0.1	**9,507.26**	**9,507.26**	9,530.55	9,568.61	10,229.52	10,418.99	10,060.99	10,458.90
	0.2	13,334.84	**12,448.99**	12,602.01	12,597.66	14,842.76	14,351.79	13,334.84	14,454.10
	0.3	14,845.51	14,845.51	**14,788.62**	14,845.51	19,266.97	17,179.86	15,663.53	17,071.02
	0.4	**16,462.33**	**16,462.33**	**16,462.33**	**16,462.33**	23,752.76	19,517.59	17,539.45	18,976.19
	0.5	**17,832.62**	**17,832.62**	**17,832.62**	**17,832.62**	28,292.07	21,580.36	19,114.86	20,488.50
	0.6	20,480.26	**19,043.42**	**19,043.42**	**19,043.42**	32,842.98	23,457.55	20,480.26	21,764.94
	0.7	**20,144.44**	**20,144.44**	**20,144.44**	**20,144.44**	37,365.06	25,194.45	21,697.56	22,887.96
	0.8	**21,166.81**	**21,166.81**	**21,166.81**	**21,166.81**	41,829.72	26,819.86	22,808.93	23,906.01
	0.9	**22,130.46**	**22,130.46**	**22,130.46**	**22,130.46**	46,223.03	28,354.81	23,842.05	24,850.31
eil101	0.1	211.28	211.28	213.96	213.96	**206.2**	211.35	214.19	231.73
	0.2	311.99	**305.04**	**305.04**	**305.04**	311.49	305.75	311.99	337.38
	0.3	**375.72**	**375.72**	**375.72**	**375.72**	417.74	380.26	390.01	412.19
	0.4	454.45	**435.43**	**435.43**	**435.43**	526.94	444.38	454.45	471.72
	0.5	**487.18**	**487.18**	**487.18**	**487.18**	638.08	502.31	508.76	522.64
	0.6	**535.07**	**535.07**	**535.07**	**535.07**	750.25	555.97	555.76	567.91
	0.7	597.65	**580.63**	**580.63**	**580.63**	862.73	606.3	597.65	608.97
	0.8	**615.95**	**615.95**	**615.95**	**615.95**	975.03	653.76	636	646.57
	0.9	**645.54**	**645.54**	**645.54**	**645.54**	1086.82	698.61	671.88	681.14
ch150	0.1	2,679.96	2,674.33	2,674.68	2,674.68	2,688.75	**2,656.48**	2,679.96	2,815.21
	0.2	**3,617.91**	**3,617.91**	3,667.64	**3,617.91**	4,161.40	3,699.41	3,779.99	3,956.34
	0.3	4,617.97	**4,366.72**	**4,366.72**	4,380.13	5,726.72	4,530.53	4,617.97	4,832.11
	0.4	**4,935.82**	5,012.95	**4,935.82**	**4,935.82**	7,364.75	5,245.12	5,275.01	5,530.29
	0.5	5,455.49	**5,442.57**	**5,442.57**	**5,442.57**	9,064.72	5,886.21	5,806.91	6,101.33
	0.6	6,250.98	**5,860.31**	**5,860.31**	**5,860.31**	10,820.91	6,476.48	6,250.98	6,580.89
	0.7	**6,226.45**	**6,226.45**	**6,226.45**	**6,226.45**	12,629.71	7,029.98	6,632.15	6,994.45
	0.8	6,967.19	**6,490.05**	**6,490.05**	**6,490.05**	14,489.55	7,556.56	6,967.19	7,359.77
	0.9	**6,736.36**	**6,736.36**	**6,736.36**	**6,736.36**	16,401.76	8,063.46	7,267.72	7,688.93
d198	0.1	**9,975.09**	**9,975.09**	**9,975.09**	9,998.01	11,724.83	12,147.77	10,076.65	10,573.61
	0.2	**11,509.65**	11,595.24	**11,509.52**	11,522.53	15,776.01	14,539.12	11,622.32	12,218.67
	0.3	12,694.47	12,567.12	**12,565.37**	**12,565.37**	19,691.88	16,141.29	12,694.47	13,368.51
	0.4	**13,450.52**	13,508.32	13,467.84	13,458.76	23,647.24	17,433.97	13,577.29	14,315.25
	0.5	14,276.34	**14,242.28**	14,245.08	14,245.08	27,625.48	18,552.04	14,336.11	15,119.02
	0.6	14,999.38	15,001.82	**14,911.64**	**14,911.64**	31,591.43	19,563.43	15,001.82	15,814.35
	0.7	15,593.96	15,593.96	**15,521.57**	**15,521.57**	35,507.30	20,506.86	15,593.96	16,425.28
	0.8	16,126.30	16,126.30	**16,091.42**	**16,091.42**	39,335.76	21,405.23	16,126.30	16,969.30
	0.9	**16,608.87**	**16,608.87**	**16,608.87**	**16,608.87**	43,042.32	22,272.12	**16,608.87**	17,459.18
pr439	0.1	**52,693.03**	**52,693.03**	**52,693.03**	**52,693.03**	61,861.19	57,301.42	**52,693.03**	60,514.33
	0.2	**69,836.51**	**69,836.51**	**69,836.51**	**69,836.51**	109,309.21	79,405.23	72,099.86	80,116.83
	0.3	**80,900.61**	**80,900.61**	**80,900.61**	**80,900.61**	157,763.21	94,867.57	84,919.85	92,960.84
	0.4	**89,619.79**	**89,619.79**	**89,619.79**	**89,619.79**	206,947.14	107,296.94	94,274.82	102,357.55
	0.5	101,542.04	96,918.32	96,918.32	**96,826.79**	256,800.83	117,956.13	101,542.04	109,703.15
	0.6	103,263.13	**102,935.88**	103,263.13	103,263.13	307,307.47	127,436.34	107,444.47	115,700.42
	0.7	108,934.76	108,934.76	**108,404.18**	**108,404.18**	358,439.26	136,076.19	112,402.51	120,761.62
	0.8	113,456.13	**113,416.38**	113,456.13	113,456.13	410,145.78	144,086.87	116,678.57	125,150.49
	0.9	117,425.73	117,425.73	**117,401.73**	117,425.73	462,374.57	151,602.30	120,445.28	129,042.63

Table 5.3: Results for GENS-H versus construction heuristics not based on savings.

Instance	p	RDM	TS	ILS	VNS	RS	SFC	FI	NI
rat783	0.1	**3,718.96**	**3,718.96**	**3,718.96**	**3,718.96**	5,968.34	3,821.64	4,497.01	4,164.90
	0.2	**5,186.55**	5,193.31	**5,186.55**	**5,186.55**	11,486.29	5,419.28	6,299.11	5,676.65
	0.3	**6,207.47**	**6,207.47**	**6,207.47**	**6,207.47**	17,098.42	6,678.97	7,422.78	6,777.86
	0.4	**7,004.56**	**7,004.56**	**7,004.56**	**7,004.56**	22,728.66	7,758.51	8,252.07	7,651.79
	0.5	**7,651.45**	**7,651.45**	**7,651.45**	**7,651.45**	28,351.37	8,717.84	8,919.42	8,379.72
	0.6	**8,208.54**	**8,208.54**	**8,208.54**	**8,208.54**	33,955.30	9,591.23	9,484.19	9,006.18
	0.7	**8,655.53**	**8,655.53**	**8,655.53**	**8,655.53**	39,534.46	10,400.84	9,977.83	9,558.11
	0.8	**9,058.67**	**9,058.67**	**9,058.67**	**9,058.67**	45,084.46	11,161.09	10,419.10	10,053.05
	0.9	10,820.10	**9,427.16**	**9,427.16**	**9,427.16**	50,599.51	11,881.18	10,820.10	10,503.13

Table 5.3: Results for GENS-H versus construction heuristics not based on savings (continued).

heuristic to determine a low cost tour along these customers, thereby deteriorating its performance. The empirical results support this hypothesis.

As to the relative performance of the high-level heuristics used for implementing GENS-H, metaheuristics seem to outperform classical local search on some combinations of coverage probabilities and problems, which could be explained by the design limitations of such classical local search algorithms. However, amongst ILS, TS and VNS, there is no clear winner.

5.2.4 Parameter choices

The results of the previous subsections illustrate the ability of GENS-H to adapt to a wide variety of different scenarios depending on the underlying problem structure. I also demonstrate that, in the worst case scenario, GENS-H tends to produce a solution that is at least as good as the solution produced by one of the 'special cases' embodied by the Supersavings approaches (Table 5.1). This subsection illustrates how GENS-H achieves these results by a case study on the parameter choices made for TSPLIB instance eil101.

Table 5.4 illustrates the parameter choices made by GENS-H for eil101 versus the parameter choices representing the special cases Supersavings (Sup), Newsave (New), Globalsave (Glo) and the Probablistic Clarke-Wright Savings Procedure (PCW) over the full set of homogeneous probabilities

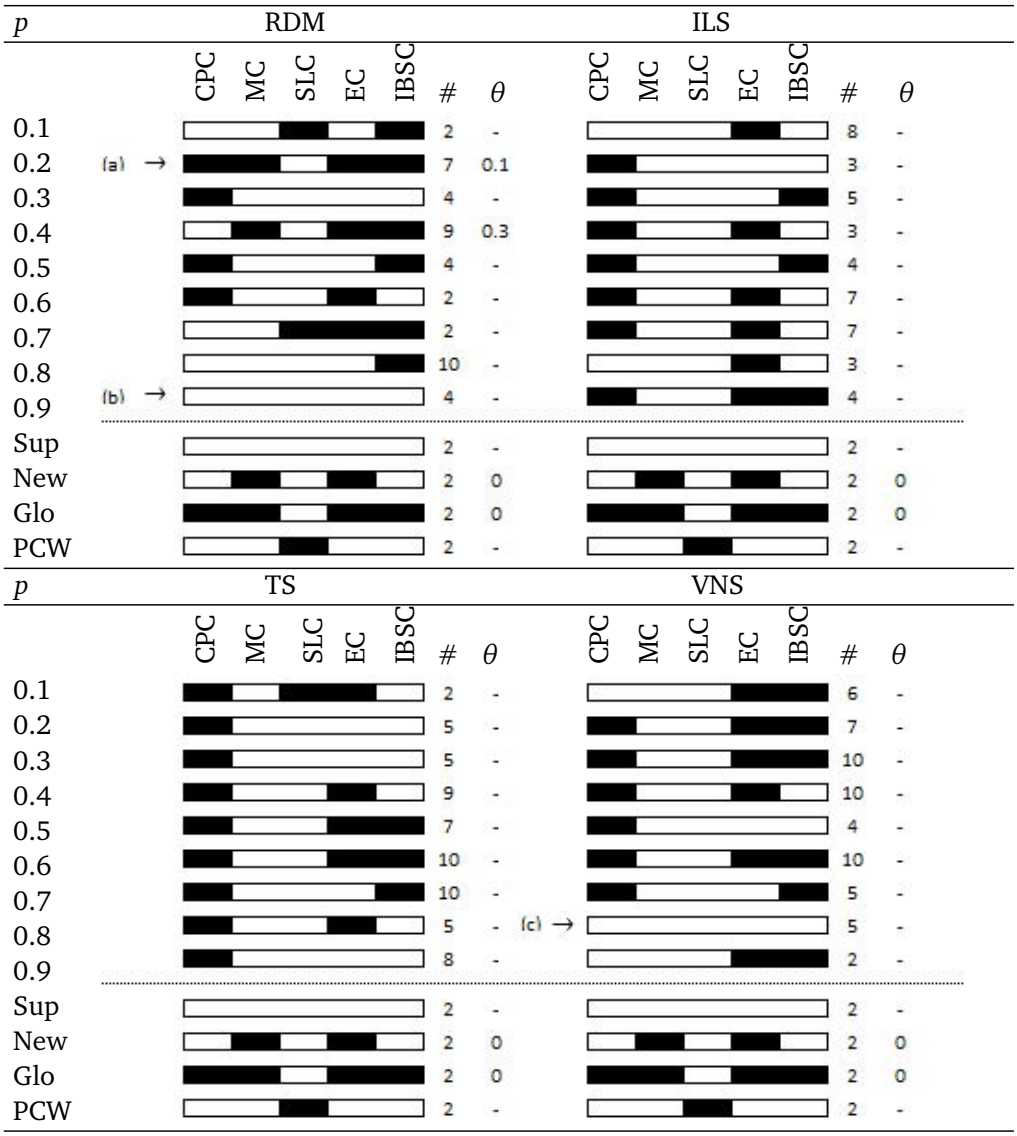


Table 5.4: GENS-H parameter choices for 4 different high-level heuristics RDM, ILS, TS and VNS applied to TSPLIB instance eil101.

$p = \{0.1, \ldots, 0.9\}$. The boxes representing the string of binary parameter choices for *CoverageProbabilities_Criterion* (CPC), *Merge_Criterion* (MC), *SavingsList_Criterion* (SLC), *Embryo_Criterion* (EC) and *Initial BestSolution_Criterion* (IBSC) are blacked out if the solution produced by GENS-H was generated with a parameter choice of 1 for that particular variable, or white if it was generated with a parameter choice of 0. The two figures following the string of binary variables, # and θ, represent the number of subtours merged at a time (*NumberOfSubtoursToMerge*) and the θ proportion of smallest number of subtours to merge among *NumberOfSubtoursTo Merge*, respectively. If the

coloured string representing the variable choices made by a GENS-H instance matches the colour pattern of one of the special cases presented just below the GENS-H choices in Table 5.4, and the values of # and θ are the same (if applicable), GENS-H picked a mechanism that corresponds to one of the special Supersavings cases. In these cases it should therefore also produce the same solution.

I distinguish 3 separate GENS-H cases annotated (a) – (c) in Table 5.4 that (partially) match the generics of other Supersavings-based approaches. The binary string of case (a), GENS-H for $p = 0.2$ using the RDM high-level heuristic, matches that of the binary pattern for the Globalsave algorithm. Consequently, the mechanisms used to obtain the solutions for GENS-H using RDM and Globalsave are largely the same. Yet the resulting solutions slightly differ: Whereas GENS-H using RDM managed to produce a solution with a corresponding expected length of 311.99, Globalsave produced a slightly better solution of length 311.32 (see Table 5.2). This difference can be attributed to other factors that play a role in constructing the solution, namely the number of subtours merged in each merging operation, and the selection of the amount of small and large subtours among the number of subtours to merge. Whereas Globalsave prescribes merging to the largest subtour out of 2 selected tours in each iteration of its merging procedure (#=2, θ=0), GENS-H with RDM favored an approach that merges the 6 largest subtours to the 1 smallest subtour in each merging operation (#=7, θ=0.1). As a result, the tours produced by both methods slightly differ; more precisely, closer inspection reveals that the tours produced by both methods are exactly the same apart from 2 arcs. In this case, the RDM high-level heuristic is not explorative enough, resulting in a slightly worse solution than Globalsave.

Cases (b) and (c), representing GENS-H with RDM and GENS-H with VNS for probabilities $p = 0.9$ and $p = 0.8$ respectively, match the binary string pattern of the Supersavings approach. These cases exemplify the ability of GENS-H to adapt one of the special cases representing another Supersavings-based approach

(see Table 5.1). Please observe that the parameter choices # and θ are irrelevant for the given set of binary choices (cf. the pseudo-codes by plugging in relevant parameter choices in Algorithm 1), hence the merging operations reduce to the same procedure as the Supersavings approach.

In the majority of cases, however, GENS-H produces a parameter set that is different from one of the special cases reported in Table 5.1. These parameter selections have the potential to result in different tours and associated expected lengths than the ones produced by the other Supersavings-based approaches. This observation continues to hold for the other tested TSPLIB problem instances (not reported here). This behavior indicates that GENS-H tends to explore a wider range of possible solutions than the existing Supersaving-based approaches, resulting in an overall better performance than the other benchmarked methods. It also demonstrates the ability of a hyper-heuristic, in our case GENS-H, to generate entirely new construction heuristics from a generalised framework using parameter selection mechanisms. The flexibility of such an approach offers a degree of adaptivity to the underlying problem structure, being more explorative and thus having a higher potential to result in a better solution.

5.3 Conclusions

This chapter proposes a new construction method for the probabilistic travelling salesman problem, called GENS, based on the concept of Supersavings. This new heuristic is a parameterised procedure that could be regarded as a generalisation of the existing Supersavings-based construction heuristics including Supersavings, Newsave, Globalsave and the Probabilistic Clarke & Wright savings procedure. GENS embeds the existing Supersavings-based methods as special instances and reduces to one of the cases embodied by this class of heuristics for specific parameter choices. The implementation of GENS within a hyper-heuristic framework, referred to as GENS-H, enables us to adapt GENS to a wide variety of PTSP scenarios. More specifically, the proposed hyper-heuristic framework of GENS-H is used to guide

the search for an optimal or near-optimal set of parameters in GENS' parameters space.

Empirical results demonstrate that GENS-H is able to unveil completely new Supersavings-based construction heuristics. The results also suggest that, in general, the newly discovered heuristics by GENS-H proved to be superior to its benchmark methods. Furthermore, the results suggest that GENS-H is robust to both coverage probabilities and the structure of TSP instances. In fact, in contrast to the tested benchmark construction heuristics which are prone to deterioration when the characteristics of the routing problem change (e.g. coverage probability, size, or the clustering of customers in the plane), GENS-H delivers consistent and high-end solutions under almost all settings. As a consequence, I argue that the quality and competitiveness of most traditional construction methods for the PTSP are context-dependent. Finally, the empirical results provide evidence that the design of parameterised heuristics and their implementation within hyper-heuristic frameworks is an interesting research avenue that needs further attention. This may be especially true for areas such as stochastic routing where exact solutions are rare and alternative explorative approaches can be promising.

6 The correlated PTSP

Ever since Jaillet (1985) introduced the PTSP in 1985, routing problems with stochastic customer presence gained widespread attention. A significant part of the literature focusses on the simple PTSP and its extension to the VRP, the PVRP (Bertsimas, 1988; Jaillet and Odoni, 1988). However, a number of alternative versions of the PTSP were also developed. These include the probabilistic pickup and delivery travelling salesman problem (Beraldi et al., 2005), probabilistic generalised travelling salesman problem (Tang and Miller-Hooks, 2007), the PTSP with deadlines (Campbell and Thomas, 2008b), the PTSP with time windows (Voccia et al., 2013), and the TSP with profits and stochastic customers (Zhang et al., 2018).

Whereas extensions of the PTSP and other stochastic routing problems are abundant, only few attempts were made to capture dependencies between the components of stochastic routing problems. Some notable exceptions include Golden and Yee (1979) and Stewart and Golden (1983), who consider correlation among the stochastic demands in the vehicle routing problem. More recently, Samaranayake et al. (2012) propose a class of algorithms to solve routing over networks with correlated travel times. And Jaillet et al. (2016) consider a class of routing problems with correlated stochastic travel times and deadlines. But in spite of the notion that the PTSP formulation can also be generalised to capture dependencies between the stochastic presence of the customers (Jaillet, 1985, p. 45), to the best of my knowledge, so far no attempts have been made to explore this avenue. Instead, research has always worked under the assumption that the

presence of the customers are drawn from independently distributed probability mass functions.

This chapter proposes a method to model dependencies in the stochastic customer presences of the PTSP using correlation structures. I refer to this new stochastic combinatorial optimisation problem, a generalisation of the PTSP, as the correlated probabilistic traveling salesman problem (CPTSP). I develop a flexible objective function for the problem that only requires the specification of the marginal probabilities describing the stochastic presence of each customer, and a correlation matrix describing the pairwise dependence between customers. In addition, the proposed implementation satisfies a number of other attractive properties with respect to the computational efficiency and the ability to reduce to the familiar objective function for the PTSP in the absence of dependence.

The CPTSP is tested under various conditions: with homogeneous and heterogeneous customer dependencies, for a wide range of customer presence probabilities, and on a number of instances of varying sizes. I highlight its practical relevance by showing that a good solution for the PTSP does not necessarily coincide with a good solution for the CPTSP. Building on this observation, I also show that the relative performance of solution methods differs depending on the degree of dependence. Here, the degree of dependence refers to the values in the range $[-1, +1]$ of the Pearson correlation coefficient, where -1 denotes a perfect negative relationship between the need to visit two stochastic customers, and $+1$ denotes a perfect positive relationship between the need to visit two stochastic customers.

This study contributes to the PTSP literature by challenging the notion of independence among stochastic customers. That is, the PTSP has traditionally been set up with independently distributed Bernoulli variables as a straightforward way to model the probabilities associated with the binary outcome of customer presence (success) and customer absence (failure) (Jaillet, 1988). The independence

assumption of the Bernoulli variables offers computational advantages over dependence models, as the joint probabilities of several customers is simply given by the product of each customer's marginal probability of presence or absence. However, in reality the need to visit one customer may very well depend on the presence of one or more other customers.

The practical relevance of considering dependency among stochastic customers extends to a number of cases. In fact, Gendreau et al. (2016) note that:

> *"Customer characteristics have a definite influence on its presence on a given epoch in the planning period, e.g., domestic customers are more likely to be present in the late afternoon when compared to normal workday hours. [...] Once identified, influential customer characteristics can be used to map correlations between clusters of customers. [...] mapping correlated behavior, will warrant more accurate representations of the stochastic phenomena at hand."*
>
> — Gendreau, Jabali, and Rei (2016, p. 1169)

That is, correlations can be used to reflect similarities in behaviour and opposite behaviour between (clusters of) stochastic customers. Grouping individuals in clusters for routing purposes is not new; see Laporte and Palekar (2002) for a list of applications of the clustered TSP. Moreover, clustering is found to have an impact on the performance of PTSP solutions (see chapter 4, and Bianchi et al., 2002a). In the context of the CPTSP, the following examples are meant to illustrate situations where different forms of dependence between stochastic customers could play a role in the determination of a good tour.

- **Example 1: Positive in-cluster correlation.** A meal-kit delivery service that mainly serves students, young professionals, and pensioners notices that customers from the same customer segment tend to be at home around the same time. Moreover, it finds that customers of a particular segment typically reside in the same neighbourhood. The company introduces a

positive correlation between customers in the same neighbourhoods based on customer segmentation.

- **Example 2: Negative in-cluster correlation.** A parcel delivery company believes that, during its typical operating hours, the presence of domestic customers is inversely related to the opening hours of businesses in the area. In order to account for this relationship without the need for re-optimisation, the parcel delivery company decides to set a negative correlation between the likelihood of a business customer's presence and the presences of domestic customers in its vicinity.
- **Example 3: Different in-cluster correlation and out-of-cluster correlation.** A company that delivers raw materials to customers along the border of two countries is taxed for some of it goods every time it crosses the border. Therefore, it prefers to serve all customers in one country first before serving customers in another country, but only if the detour to serve the customers who require service on a particular day is not too large. The company sets a positive correlation between neighbouring customers in the same country that diminishes with increasing distance. In addition, it sets a negative correlation between neighbouring customers of different countries, that gradually vanishes with increasing distance.
- **Example 4: Correlations not based on spatial clusters.** A factory producing electronic devices routinely picks up components from different suppliers to assemble the modules needed for the end product. In some cases, components for different modules can be picked up from the same supplier. However, due to limited supplies, changing demands, and varying production schedules, the availability of each component – and hence, the need to visit a certain supplier – is subject to uncertainty. The factory finds it more cost-effective to only pick up certain components from its suppliers, when a complete module can be assembled with those components. Therefore, it sets a positive correlation between suppliers who produce components that go into the same module.

In the examples above, the CPTSP allows us to account for similarities and

discrepancies among customers of the same cluster and customers of other clusters in the pre-determination of a good tour.

The remainder of this chapter is organised as follows. Section 6.1 reviews the literature relevant for binary dependency modelling. Section 6.2 presents the mathematical set-up of the CPTSP under various conditions and derives expressions for the expected length. Section 6.3 discusses the properties of the models previously derived and discusses the practical implications. The CPTSP is subsequently tested in a range of different settings in section 6.4, and elaborates on the empirical findings. Section 6.5 proposes the correlated probabilistic ant colony system metaheuristics, a problem-specific method to cope with the correlated stochastic nature of the CPTSP. Finally, Section 6.6 concludes.

6.1 An overview of multivariate probability models for discrete outcomes

In order to model the joint probability between customers when the binary outcome of each customer's presence is potentially dependent, one must resort to multivariate discrete probability models. Multivariate probability models for discrete outcomes can be roughly divided into two categories: Marginal models and conditional models (for an overview see, for example, Joe, 1997; Molenberghs and Verbeke, 2005; Agresti, 2013). A third category is formed by a hybrid combination of these two, called mixed marginal-conditional models.

As the name implies, marginal models allow for the explicit specification of the marginal probabilities of a set of discrete random variables. This property offers substantial ease from the point of view of an implementer. Among the most popular traditional marginal models are the Bahadur model, multivariate probit models, and the Dale model. The Bahadur model (Bahadur, 1961) employs a straightforward expansion of the independence model with a dependence term

composed of correlation coefficients. Its closed-form expression is easy to interpret and allows for tractability of the results. On the downside, its parameter space is found to be restrictive (Declerck et al., 1998). Multivariate probit models (Molenberghs and Verbeke, 2005) and the Dale model (Dale, 1986) study the relationship between binary variables and a set of regressors which are believed to influence the outcomes. Whereas the multivariate probit model assumes that the probability associated with a set of binary dependent variables follows a normal cumulative distribution function (cdf) around a linear combination of the regressors, the Dale model and its multivariate extension (Molenberghs and Lesaffre, 1994) rely on the decomposition of joint probabilities into 'main effects', described by marginal probabilities, and 'interactions', described by cross ratios. The joint probability of the multivariate probit model does not have a closed-form expression, but requires the estimation of a multidimensional integral which scales with the number of variables (Hajivassiliou and Ruud, 1994). The Dale model, on the other hand, does not require any numeric evaluation, since it embeds the closed-form Plackett distribution (Plackett, 1965) as its underlying probability distribution. The multivariate Plackett distribution (Molenberghs and Lesaffre, 1994) is an extension of the bivariate Plackett distribution. It describes joint probabilities in a recursive fashion, where each joint probability is calculated from previously calculated lower dimensional probabilities. Similar in spirit, Teugels (1990) also proposes a set of link functions that recursively relies on lower order results to derive the joint probability of higher order discrete outcomes.

Generalised estimating equations (Liang and Zeger, 1986) are a special class of marginal models. Generalised estimating equations (GEE) provide an alternative way of estimating the relationship between discrete outcomes and a set of explanatory variables compared to the previously described marginal models, which are all based on maximum likelihood estimation. Among the most studied GEE are Prentice (1988), who extends the original concept of GEE by Liang and Zeger (1986) with a method to estimate joint probabilities and pairwise correlations between variables, and Lipsitz et al. (1991), who propose an

alternative estimation method of the GEE by Prentice (1988) using odd ratios. GEE do not make assumptions regarding the underlying distribution, which is why GEE are unsuitable for models that require the explicit specification of a joint probability mass function (pmf).

Copulas (Nelsen, 2006) are a class of general multivariate dependency models that have become increasingly popular in recent years. Copulas rely on the notion that the cdf of any set of random variables can be transformed to uniformly distributed marginal density functions. Therefore, copulas are a special class of marginal models. Copulas are abundant in quantitative finance literature (see, for example, Cherubini et al., 2004, for an overview), but relatively rare in routing and transportation literature. Although copulas rely on the assumption that their marginals are continuous, some efforts were made to apply copulas to binary variables (Lee, 1993) and other types of discrete data (Genest and Nešlehová, 2007). For example, the multivariate Gaussian distribution may be regarded a special case of a Gaussian copula that can also applied to discrete data (Dai et al., 2013). However, simply applying copulas to discrete sets of variables does not come at zero costs. More specifically, the Fréchet upper and lower bounds (Fréchet, 1951) of the association measures used by copulas (e.g., Kendall's tau, Spearman's rho) are all expressed in terms of their marginals. This, in turn, implies that one can only choose from a limited number of values to express the association structure, analogous to the parameter restrictions on the Bahadur and Ising models. This problem can be circumvented by introducing latent variables so that an unrestricted continuous distribution can be approximated from its discrete counterpart (Smith and Khaled, 2012; Smith et al., 2012). However, like most other copula-based approaches, this method does not easily scale to higher dimensions. Panagiotelis et al. (2012) therefore propose a method to specifically cope with high-dimensional discrete data, by decomposing a copula into a set of bivariate copulas in a discrete vine (D-vine) structure. This so-called D-vine pair copula construction (PCC) does not suffer from limitations on the number of dimensions. Panagiotelis et al. (2012) demonstrate how D-vine PCCs can be used to estimate high dimensional sets of

binary data with Gaussian, Gumbel and Clayton copulas. The Clayton copula (Clayton, 1978), directly derived from the Cox model, and the Gumbel copula (Gumbel, 1960) benefit from a closed-form expression, as do many other types of Archimedean copulas. However, both copulas are restricted to only positive association parameters. On the contrary, whereas elliptical copulas typically do not admit a closed-form expression, this copula family does not put restrictions on the bounds of the association parameter.

Conditional models (Agresti, 2013) are a separate category of discrete probability models that estimate joint probabilities given the outcomes of other observations of the same unit. In contrast to marginal models, conditional models do not require the explicit specification of the marginal probabilities of each outcome. Conditional models usually take the form of a log-linear model. This form was popularised since the introduction of the Cox (1972) model. The Cox model forms the basis of many other models that have been developed since, most notably the quadratic simplifications of the model by Zhao and Prentice (1990) and Molenberghs and Ryan (1999). The Pólya urn model (Mahmoud, 2008) and its related Beta-binomial (Prentice, 1986) and Dirichlet-multinomial (Aerts et al., 2002) distributions, are considered to be another subclass of conditional models where each probability is sequentially updated conditional on previous draws. One major disadvantage of conditional models, in contrast to marginal models, is the lack of reproducibility (Liang et al., 1992; Fitzmaurice et al., 1993). That is, the marginalisation of the joint pmf of a conditional model over a random subset of discrete variables does not lead to a pmf from the same distribution family.

Mixed marginal-conditional models (Molenberghs and Verbeke, 2005) extend the conditional models of Cox (1972), Zhao and Prentice (1990), Fitzmaurice and Laird (1993) and others with link functions of, among others, McCullagh and Nelder (1989) to allow for the explicit specification of marginal probabilities. Pursuing such an approach for the quadratic conditional model results in a class of models related to Ising model (Ising, 1925; Kruis and Maris, 2016). The Ising

model has its roots in statistical physics to make inferences about the dipole states of atomic spins in magnetic fields. Dai et al. (2013) show that a generalisation of the Ising model, that can be regarded as an analogous mixed marginal-conditional version of the Cox model, results in a multivariate Bernoulli distribution. In contrast to the conditional models, the marginal distributions of the Ising model and multivariate Bernoulli distribution do not suffer from the lack of reproducibility (Dai et al., 2013, p. 11). However, both models involve the calculation of a so-called log-partition function that requires the calculation of a sum over all possible binary outcomes. Moreover, the number of parameters that needs to be specified in the multivariate Bernoulli distribution grows exponentially with the number of binary outcomes. Whereas the implementation of the Ising model only requires the specification of the pairwise probabilities, deriving such probabilities from their marginal probabilities using, for example, the link functions by Teugels (1990), induces bounds on the admitted range of correlation coefficients.

6.2 Methods

6.2.1 Basic model requirements

This section discusses various alternatives for the joint probability mass function (pmf) and their corresponding expressions for the expected length. A suitable joint pmf is a prerequisite for modelling dependence between customers. I restrict the attention to the independent probability model of the PTSP (section 6.2.3), the Bahadur model (section 6.2.4), and PCCs (section 6.2.5). A suitable joint pmf for the CPTSP should satisfy the following properties.

(i) The joint pmf is separable in individual Bernoulli distributed marginal probabilities and their pairwise correlations;

(ii) There is no need to specify all joint probabilities: Higher order probabilities result directly from marginal probabilities and their dependency formulation;

(iii) The joint pmf reduces to the pmf of the PTSP in case dependence is absent;

(iv) The joint probability allows for reproducibility, i.e., the marginalisation of the joint pmf over a random subset of customer presences leads to a joint probability mass function from the same distribution family;

(v) The joint pmf can be computed in polynomial time.

This desire for these properties is motivated by a number of reasons.

Property (i) follows primarily from a practitioner's perspective. Joint probabilities reflect complicated relationships between customer interactions and can be difficult to estimate upfront. In contrast to conditional models, which possibly rely on successive model outcomes, marginal models support the possibility to specify individual customer probabilities rather than only joint probabilities. Also GEE, which do not make any distributional assumptions, are for this reason also not considered here.

Property (ii) follows from a modelling perspective. Pre-specifying or computing all $2^n - 1$ joint probabilities becomes a tedious task when the number of customers n becomes large. Essentially, this property prohibits model choices that require computations of the same order of magnitude as full enumeration. The Bahadur model and PCCs only require the specification of marginal probabilities and pairwise interactions, and are therefore very attractive choices from a computational perspective. On the contrary, some other types of marginal models, including the suggested approaches based on odds ratios, e.g., the multivariate Plackett distribution and the Dale model, but also the link functions described by Teugels (1990), the Ising model, the multivariate Bernoulli model, and the copula approach by Lee (1993), are deemed to be inappropriate choices.

Property (iii) is self-explanatory and should allow for a one-to-one comparison of the results with the PTSP. The mathematical set-up of the Bahdur model and copulas allow for a straightforward reduction to the pmf of the PTSP in the case dependence is absent (see sections 6.2.4 and 6.2.5), unlike most types of conditional models and mixed marginal-conditional models.

Property (iv) ensures that the joint probabilty mass specification of a subset of customers does not depend on the full dependency structure, such that excluding a subset of customers in the original problem still yields an outcome that can be compared directly to outcomes of the complete problem. The property also aids the derivation of an objective function that can be solved in polynomial time due to the separability of probability terms. This property excludes GEE, and certain types of conditional and mixed marginal-conditional models.

Property (v) also follows from a practitioner's perspective. It excludes models that become computationally intensive or inaccurate when the number of dimensions grows. For example, it prohibits models that involve the numeric evaluation of multidimensional integrals, such as the multivariate probit model and certain formulations of discrete copula models.

6.2.2 Mathematical preliminaries

In this section, I propose a framework for calculating the expected length of a tour under general (i.e., distribution independent) conditions. In order to derive a mathematical framework that is eventually also able to accommodate the model requirements described in the previous section, I initially discard the original mathematical definition of the PTSP described in chapter 3 to start our journey from scratch. A new framework is then gradually built by taking a bottom-up approach, where useful concepts are borrowed from the original PTSP framework whenever I deem them useful. The notation introduced in chapter 3 still remains valid.

Let us start by the basis. For a given tour τ on G, its deterministic length $L(\tau)$ is given by the sum over the distances d_{ij} of the involved edges:

$$L(\tau) = \sum_{(i,j)\in E} d_{ij} x_{ij}, \tag{6.1}$$

The TSP is concerned with the minimisation of (6.1) subject to a degree equation,

a subtour elimination constraint and an integrality constraint (see section 3.4 and Applegate et al., 2006).

Remember from sections 3.1 – 3.2 that only a subset $S \subseteq N$ of the nodes requires a visit when the presences of the customers in the TSP are stochastic, such that the edges in (6.1) that need to be traversed, expressed through x_{ij}, are a function of the customer presences in N. Let us assume that the original recourse policy of the PTSP, i.e., skipping customers who are absent in the second stage of the problem, will be retained regardless of the pmf definition. With this in mind, the length of a (realised) a posteriori tour can be rewritten in terms of the Bernoulli distributed variables $\mathbf{Y} = (Y_1, \ldots, Y_n)$ as

$$L(\tau|\mathbf{y}) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} f_a(\mathbf{y}_{ij}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} f_b(\mathbf{y}_{ji}), \tag{6.2}$$

where

$$f_a(\mathbf{y}_{ij}) = y_i y_j \prod_{k=i+1}^{j-1} (1 - y_k), \qquad \text{and} \qquad f_b(\mathbf{y}_{ji}) = y_j y_i \prod_{k=j+1}^{n} (1 - y_k) \prod_{l=1}^{i-1} (1 - y_l).$$

That is, given the realisations $\mathbf{y}$ of the nodes in τ, the length of a tour is given by the sum over the distances of all edge combinations $(i, j) \in E$. However, each edge $(i, j) \in \tau = (1, \ldots, n)$ needs to be traversed if and only if both customers i and j require a visit (i.e., $y_i = y_j = 1$), while any intermediate customers $i, \ldots, j$ on the path $a = (i, \ldots, j)$, $i < j$, or intermediate customers $j + 1, \ldots, n, 1, \ldots, i - 1$ on the path $b = (j, \ldots, n, 1, \ldots, i)$, $i > j$, do not require a visit (i.e., $y_r = 0, r \in \{a, b\} \setminus \{i, j\}$). The functions $f_a(\cdot)$ and $f_b(\cdot)$ reflect the need of traversing paths a and b as a function of the realisations $\mathbf{y}_{ij} \subseteq \mathbf{y}$ and $\mathbf{y}_{ji} \subseteq \mathbf{y}$ of the customers on paths a and b, respectively, in the specified manner. The term $L(\tau|\mathbf{y})$ denotes the corresponding total deterministic distance – or length – of the tour τ on a subset of E for a given vector of realisations $\mathbf{y}$.

Similar to the TSP, a salesman faced with stochastic customer conditions still seeks to minimise the length of the tour (6.2) he has to complete. Akin the

reasoning in section 3.2, because in the TSPSC the realisations of the variables $\mathbf{y}$ are unknown – expressing the composition of the associated set S of customers who require a visit – our aim in this problem is to minimise the *expected* length of the tour in (6.2) instead. Taking the expectation of (6.2) over every possible combination of the stochastic elements $y_k \in \mathbf{y}$ (cf. equation (3.1)) results in

$$\mathbb{E}[L(\tau)] = \sum_{y_k \in \{0,1\}^n} L(\tau|\mathbf{y}) p(\mathbf{y}). \tag{6.3}$$

The ability to decompose the length of a tour (6.3) in a sum of $n(n-1)$ independent terms, analogous to (6.2), is retained when we change the perspective from an a posteriori to an a priori length calculation. This is due to the independence between any (deterministic) distance $d_{s,t}$ and the probability $p_u(\mathbf{y}_{st})$ associated with traversing that edge (s,t). Therefore,

$$\begin{aligned}
\mathbb{E}[L(\tau)] &= \sum_{y_k \in \{0,1\}^n} L(\tau|\mathbf{y}) p(\mathbf{y}) \\
&= \sum_{y_k \in \{0,1\}^n} \left(\sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} f_a(\mathbf{y}_{ij}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} f_b(\mathbf{y}_{ji}) \right) p(\mathbf{y}) \\
&= \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} \sum_{y_k \in \{0,1\}^n} f_a(\mathbf{y}_{ij}) p(\mathbf{y}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} \sum_{y_k \in \{0,1\}^n} f_b(\mathbf{y}_{ji}) p(\mathbf{y})
\end{aligned} \tag{6.4}$$

where $p(\mathbf{y})$ denotes the joint probability of the outcomes $\mathbf{y}$ occurring.

Furthermore, observe that $f_a(\mathbf{y}_{ij}) = 0$ and $f_b(\mathbf{y}_{ji}) = 0$ for any realisation other than $\mathbf{y}_{ij} := (1,0,\ldots,0,1)_{|a|}$ and $\mathbf{y}_{ji} := (1,0,\ldots,0,1)_{|b|}$. Therefore, any terms with outcomes of $\mathbf{y}$ that deviate from $\mathbf{y}_{ij}$ or $\mathbf{y}_{ji}$ vanish from (6.4):

$$\mathbb{E}[L(\tau)] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} p_a(\mathbf{y}_{ij}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} p_b(\mathbf{y}_{ji}), \tag{6.5}$$

where $p_a(\mathbf{y}_{ij})$ and $p_b(\mathbf{y}_{ji})$ are the marginal probabilities that can be obtained by marginalising the joint pmf $p(\mathbf{y})$ over all $y_r \notin a$ and $y_s \notin b$, respectively, while the

customers on paths a and b take the outcomes of the forms $\mathbf{y}_{ij}$ and $\mathbf{y}_{ji}$. . That is,

$$p_a(\mathbf{y}_{ij}) = \sum_{\substack{y_r \in \{0,1\}^{n-|a|} \\ y_r \notin a}} p(Y_1 = y_1, \ldots, Y_{i-1} = y_{i-1}, Y_i = 1, \\ Y_{i+1} = 0, \ldots, Y_{j-1} = 0, Y_j = 1, Y_{j+1} = y_{j+1}, \ldots, Y_n = y_n), \tag{6.6}$$

and

$$p_b(\mathbf{y}_{ji}) = \sum_{\substack{y_s \in \{0,1\}^{n-|b|} \\ y_s \notin b}} p(Y_1 = 0, \ldots, Y_{i-1} = 0, Y_i = 1, \\ Y_{i+1} = y_{i+1}, \ldots, Y_{j-1} = y_{j-1}, Y_j = 1, Y_{j+1} = 0, \ldots, Y_n = 0). \tag{6.7}$$

Observe that the binary responses of each marginal variable $Y_i \in \mathbf{Y}$ are captured by the pmf of a Bernoulli distribution (see (3.1) in Chapter 3). The remainder of this section is concerned with finding a suitable pmf $p(\mathbf{y})$ for the objective function (6.5), given that the marginal probability associated with each random variable Y_i is Bernoulli distributed.

6.2.3 Independence

In chapter 3 we learnt that under the assumption of independence, the joint probability $p^{\text{i.d.}\cdot}(\mathbf{y})$ that the customers in subset S require a visit and the customers of its complement $N \setminus S$ do not, is given by:

$$p^{\text{i.d.}\cdot}(\mathbf{y}) = \prod_{i \in N} p_i^{y_i} (1 - p_i)^{1 - y_i} = \prod_{i \in S} p_i \prod_{i \in N \setminus S} (1 - p_i). \tag{6.8}$$

Since the probabilities associated with visiting each customer are independent, the joint probability is simply given by the product of the involved marginal probabilities.

Let $p_a^{\text{i.d.}\cdot}(\mathbf{y}_{ij})$ and $p_b^{\text{i.d.}\cdot}(\mathbf{y}_{ji})$ denote the probabilities (6.6) and (6.7) associated with paths a and b in the case of independence, respectively. As the sum of the Bernoulli pmf over the outcomes is given by $\sum_{y_i \in Y_i} g(Y_i) = 1$, the probabilities $p_a^{\text{i.d.}\cdot}(\mathbf{y}_{ij})$ and

$p_b^{\text{i.d.}}(\mathbf{y}_{ji})$ follow directly from (6.8):

$$\begin{aligned} p_a^{\text{i.d.}}(\mathbf{y}_{ij}) &= p^{\text{i.d.}}(Y_i = 1, Y_{i+1} = 0, \ldots, Y_{j-1} = 0, Y_j = 1) \\ &= \prod_{r \in a} p_r^{y_r} (1 - p_r)^{1 - y_r} = p_i p_j \prod_{k=i+1}^{j-1} (1 - p_k), \end{aligned}$$

and

$$\begin{aligned} p_b^{\text{i.d.}}(\mathbf{y}_{ji}) &= p^{\text{i.d.}}(Y_1 = 0, \ldots, Y_{i-1} = 0, Y_i = 1, Y_j = 1, Y_{j+1} = 0 \ldots, Y_n = 0) \\ &= \prod_{r \in b} p_r^{y_r} (1 - p_r)^{1 - y_r} = p_j p_i \prod_{k=j+1}^{n} (1 - p_k) \prod_{l=1}^{i-1} (1 - p_l). \end{aligned}$$

In other words, the joint probabilities associated with paths a and b are independent of the outcomes of the customers beyond paths a and b. Furthermore, the independence property allows us to rewrite $p_a^{\text{i.d.}}(\mathbf{y}_{ij})$ and $p_b^{\text{i.d.}}(\mathbf{y}_{ji})$ as products of their marginal pmfs.

The expected length $\mathbb{E}\left[L\left(\tau^{\text{i.d.}}\right)\right]$ of the Hamiltonian tour $\tau^{\text{i.d.}} = (1, \ldots, n)$ under the assumption of independence can be written as the composite of these probability expressions, by substituting $p_a(\mathbf{y}_{ij})$ by $p_a^{\text{i.d.}}(\mathbf{y}_{ij})$, and $p_b(\mathbf{y}_{ji})$ by $p_b^{\text{i.d.}}(\mathbf{y}_{ji})$ in (6.5). Doing so yields the result:

$$\mathbb{E}\left[L\left(\tau^{\text{i.d.}}\right)\right] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} p_a^{\text{i.d.}}(\mathbf{y}_{ij}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} p_b^{\text{i.d.}}(\mathbf{y}_{ji}). \tag{6.9}$$

Note that this objective function is identical to the original PTSP objective function described in chapter 3, by observing that $p_a^{\text{i.d.}}(\mathbf{y}_{ij})$ and $p_b^{\text{i.d.}}(\mathbf{y}_{ji})$ are identical to the weight expressions w_{ij} and w_{ji} in (3.6). Therefore, (6.9) also reduces to (3.4) in the case of independent and identically distributed (i.i.d.) pmfs. The heterogeneous PTSP objective function (6.9) can be evaluated in $O(n^2)$ computational time, compared to $O(n)$ for the TSP objective function (6.1).

6.2.4 Bahadur model

The Bahadur model (Bahadur, 1961), also known as the Bahadur-Lazarsfeld expansion, is perhaps the most widely known multivariate distribution for modelling dependencies between Bernoulli variables. Bahadur (1961) uses that the dependency between a variable Y_i and another variable $Y_j, i \neq j$, may be represented by a marginal correlation coefficient

$$\rho_{ij} \equiv \text{Corr}(Y_i, Y_j) = \mathbb{E}(z_i z_j), \tag{6.10}$$

where $z_i = \frac{y_i - p_i}{\sqrt{p_i(1-p_i)}}$. Higher order correlations between more than two variables, e.g. ρ_{ijk} describing the joint correlation between the variables Y_i, Y_j and Y_k, assume a similar form. In general, the following relatinship holds: $\rho_{1,2,\dots,n} = \mathbb{E}(z_1 z_2 \dots z_n)$.

Given these relationships, the joint probability distribution $p^{\text{B}^*}(\mathbf{y}|\mathbf{R}^*)$ under the Bahadur model is given by

$$\begin{aligned} p^{\text{B}^*}(\mathbf{y}|\mathbf{R}^*) = &\prod_{i \in N} p_i^{y_i}(1-p_i)^{1-y_i} \\ &\times \left(1 + \sum_{i<j} \rho_{ij} z_i z_j + \sum_{i<j<k} \rho_{ijk} z_i z_j z_k + \dots + \rho_{12\dots n} z_1 z_2 \dots z_n\right), \end{aligned} \tag{6.11}$$

where $\mathbf{R}^*$ denotes an n^n correlation matrix capturing the correlation coefficients $\rho_{12}, \rho_{12\dots n}$. Essentially, (6.11) can be thought of as the same independently distributed probability terms as for the PTSP (6.8), but with a corrective action for the correlations expressed through the second term on the right hand side. In fact, note that when all off-diagonal correlation entries $\rho_{ij}, \rho_{ijk}, \dots, \rho_{12\dots n}$ are set to zero such that $\mathbf{R}^*$ is equal to an $n \times n$ identity matrix $\mathbf{I}^*$, the term within parentheses on the right hand side in (6.11) vanishes. Consequently, (6.11) reduces to the probability distribution of independently Bernoulli distributed random variables (6.8). Mathematically speaking, we have that $p^{\text{B}^*}(\mathbf{y}|\mathbf{I}^*) = p^{i.d.}(\mathbf{y})$.

To allow for tractable results, I restrict our attention to second order terms ρ_{ij}

and set all the remaining higher order terms $\rho_{ijk}, \ldots, \rho_{12\ldots n}$ to zero. The second order correlation coefficients are collected in an $n \times n$ correlation matrix $\mathbf{R}$. This results in a Bahadur model of the second order:

$$p^{\mathrm{B}}(\mathbf{y}|\mathbf{R}) = \prod_{i \in N} p_i^{y_i} (1-p_i)^{1-y_i} \left(1 + \sum_{i<j} \rho_{ij} z_i z_j \right). \tag{6.12}$$

Pruning the summation of all correlation terms beyond the second order increases the ease of implementation from a practitioners perspective, because now only pairwise correlations need to be specified. Moreover, the computational burden is significantly reduced. However, disregarding higher order terms does not come completely at zero costs. A number of practical implications are discussed in paragraph 6.3.

It can be demonstrated that the marginal distribution of the Bahadur models in (6.11) and (6.12) with respect to a subset of Bernoulli random variable realisations $\mathbf{y}_M \subseteq \mathbf{y}$, corresponding to a subset of nodes $M \subseteq N$, results in a pmf of the same density family. That is, it satisfies the reproducibility property (iv) described in section 6.2.1. In mathematical terms:

$$\sum_{\mathbf{y}_M = \{0,1\}^M} p^{\mathrm{B}}(\mathbf{y}) = p^{\mathrm{B}}(\mathbf{y}_{M^C}),$$

where M^C denotes the complement of the set M, such that $\mathbf{y} = \mathbf{y}_M \cup \mathbf{y}_{M^C}$. The reproducibility property is required to decompose the full joint pmf $p^{\mathrm{B}}(\mathbf{y}|\mathbf{R})$ in a series of individual joint pmfs $p_a^{\mathrm{B}}(\mathbf{y}_{ij}|\mathbf{R})$ and $p_b^{\mathrm{B}}(\mathbf{y}_{ji}|\mathbf{R})$, corresponding to the probability of traversing a path a or b, respectively, under the Bahadur model. Consequently, the path probabilities are given by

$$p_a^{\mathrm{B}}(\mathbf{y}_{ij}|\mathbf{R}) := \sum_{y_r \notin a} p^{\mathrm{B}}(\mathbf{y}|\mathbf{R}) = p^{\mathrm{B}}(\mathbf{y}_{ij}|\mathbf{R}) = \left[p_i p_j \prod_{r=i+1}^{j-1} (1-p_r) \right] \left(1 + \sum_{k=i}^{j-1} \sum_{l=k+1}^{j} \rho_{kl} z_k z_l \right),$$

and

$$p_b^{\mathrm{B}}(\mathbf{y}_{ji}|\mathbf{R}) := \sum_{y_s \notin b} p^{\mathrm{B}}(\mathbf{y}|\mathbf{R}) = p^{\mathrm{B}}(\mathbf{y}_{ji}|\mathbf{R})$$
$$= \left[p_i p_j \prod_{k=j+1}^{n} (1-p_k) \prod_{l=1}^{i-1} (1-p_l) \right] \left(1 + \sum_{k=i}^{j-1} \sum_{l=k+1}^{j} \rho_{lk} z_l z_k \right).$$

Setting $p_a(\mathbf{y}_{ij}) = p_a^{\mathrm{B}}(\mathbf{y}_{ij}|\mathbf{R})$ and $p_b(\mathbf{y}_{ji}) = p_b^{\mathrm{B}}(\mathbf{y}_{ji}|\mathbf{R})$ in equation (6.5) results in an expression for the expected length of a Hamiltonian tour $\tau^{\mathrm{B}} = (1, \ldots, n)$ with Bahadur-correlated customer probabilities:

$$\mathbb{E}[L(\tau^{\mathrm{B}})] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} p_a^{\mathrm{B}}(\mathbf{y}_{ij}|\mathbf{R}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} p_b^{\mathrm{B}}(\mathbf{y}_{ji}|\mathbf{R}) \tag{6.13}$$

This equation can be thought of as the objective function of the PTSP with correlated stochastic customer presences according to the Bahadur model, or simply Bahadur-CPTSP. Identical to the PTSP, the goal in the Bahadur-CPTSP is to minimise (6.13) with respect to the tour τ^{B}. It exhibits the same mathematical form as the PTSP objective function in (6.9), but now each probability term is weighed by the correlation coefficients associated with the customers on path a or b. In fact, if all second order cross correlations $\rho_{st} = 0 \ \forall s, t$ such that $\mathbf{R} = \mathbf{I}$ with $\mathbf{I}$ the identity matrix, we have that $p_a^{\mathrm{B}}(\mathbf{y}_{ij}|\mathbf{I}) = p_a^{\mathrm{i.d.}}(\mathbf{y}_{ij})$ and $p_b^{\mathrm{B}}(\mathbf{y}_{ji}|\mathbf{I}) = p_b^{\mathrm{i.d.}}(\mathbf{y}_{ji})$. In that case, equation (6.13) reduces to the expected length of an a priori tour in the familiar independent PTSP sense (6.9). As a consequence of the extra correlation term in (6.13), the complexity increases from $O(n^2)$ for the PTSP objective function to $O(n^4)$ for the Bahadur-CPTSP objective function.

In the fully homogeneous case, i.e., with equiprobable $p_i = p \forall i$ and equicorrelation $\rho_{ij} = \rho \forall i \neq j$ terms, our expression for the expected length of an a priori tour $\tau^{i.d.b.}$ with identically distributed Bahadur-correlated (i.d.b.) customer

probabilities reduces to

$$\mathbb{E}[L(\tau^{\text{i.d.b.}})] = p^2 \sum_{r=0}^{n-2} (1-p)^r \left(1 + \frac{\rho}{p(1-p)} \left(1 - 2(r+1)p + \frac{1}{2}(r+1)(r+2)p^2\right)\right) \times \sum_{j=1}^{n} d_{j,(j+1+r) \bmod n}. \tag{6.14}$$

Again, this formula attains the same mathematical form as the objective function of the homogeneous PTSP, but with an additional corrective term to account for the correlation in the stochastic need to visit customers. The homogeneous PTSP objective funtion (3.4) can be obtained by setting $\rho = 0$. Observe that the computational burden of the homogeneous CPTSP-Bahadur variant above is drastically reduced compared to the objective function of the heterogeneous CPTSP-Bahadur objective function (6.13), namely $O(n^2)$ instead of $O(n^4)$ – the same complexity as the PTSP objective function (3.3).

6.2.5 Discrete-Vine Pair Copula Constructions

Copulas rely on the notion that the cdf of every continuous random variable has a uniform distribution on the interval $[0,1]$ that can be obtained by probability integral transformation. A copula uses a multivariate probability distribution to describe the relationship between several of these transformed cdfs. In fact, Sklar's (1959) theorem states that every multivariate cdf can be rewritten in terms of its marginal probabilities and a copula function.

Consider a set of random variables $U = (U_1 = F_1(x_1), U_2 = F_2(x_2), \ldots, U_n = F_n(x_n))$ with uniformly distributed marginal probability density functions, i.e., $u_i \sim U_i(0,1)$. The function $F_i(x_i)$ denotes the cdf of the random variable X_i with realisation x_i, i.e., $F_i(x_i) = \Pr(X_i \leq x_i)$. Given a set of cdfs $(F_1(X_1), \ldots, F_n(X_n))$ and a correlation matrix $\mathbf{R}$ describing their degree of association, a copula describes the joint cumulative distribution in terms of its probability integral transformed set U:

$$C(u_1, u_2, \ldots, u_n|\mathbf{R}) = \Pr(U_1 \leq u_1, U_2 \leq u_2, \ldots, U_n \leq u_n|\mathbf{R}). \tag{6.15}$$

Observe that, using the inverse probability transformation $X_i = F^{-1}(U_i)$, a copula (6.15) can also be expressed in in terms of the original random variables $(X_1,\ldots,X_n)$:

$$C(u_1,u_2,\ldots,u_n|\mathbf{R}) = \Pr(X_1 \leq F^{-1}(u_1), X_2 \leq F^{-1}(u_2),\ldots,X_n \leq F^{-1}(u_n)|\mathbf{R}). \quad (6.16)$$

Since the customer probabilities in the PTSP are given by discrete Bernoulli distributed random variables, using a copula – which formally requires continuous variables as input – may not seem immediately obvious. However, applying copulas to discrete valued variables has become increasingly common in the last decade (Joe, 1997; Song, 2007; Nikoloulopoulos, 2013). Although there are a number of limitations to using discrete variables, other useful properties of the copula still remain valid (Genest and Nešlehová, 2007).

Using finite differences, one is able to express the pmf of any discrete variable in terms of cdfs as

$$\Pr(Y_1 = y_1,\ldots,Y_n = y_n) = \sum_{k_1=0,1} \cdots \sum_{k_n=0,1} (-1)^{k_1+\cdots+k_n} \Pr(Y_1 \leq y_1 - k_1,\ldots,Y_n - k_n).$$

By substituting the copula (6.15) for the probability term above, it follows that (Song, 2007):

$$\begin{aligned} p^{\mathrm{C}}(\mathbf{y}|\mathbf{R}) &\equiv \Pr(Y_1 = y_1,\ldots,Y_n = y_n|\mathbf{R}) \\ &= \sum_{k_1=0,1} \cdots \sum_{k_n=0,1} (-1)^{k_1+\cdots+k_n} C(u_1^{k_1},\ldots,u_n^{k_n}|\mathbf{R}), \end{aligned} \quad (6.17)$$

where $u_k^{k_1} = F_k(y_k)$ and $u_k^{k_2} = F_k(y_k - 1)$ for any countable discrete random variable

Y_k. Hence, if Y_k is a Bernoulli distributed random variable with cdf

$$F_k(y_k) = \begin{cases} 0 & \text{if } y_k < 0, \\ 1 - p_k & \text{if } 0 \leq y_k < 1, \\ 1 & \text{if } y_k \geq 1, \end{cases} \tag{6.18}$$

then (6.17) may be evaluated by substituting the Bernoulli cdf for $u_k^{k_j} \forall j, k$, where $j = \{1, 2\}$ and $k = \{1, \ldots, n\}$.

Copulas enjoy a number of advantageous properties (Nelsen, 2006) that aid the verification of the requirements postulated in section 6.2.1. Firstly, copulas satisfy $C(u_1, \ldots, u_{i-1}, 1, u_{i+1}, \ldots, u_n) = C(u_1, \ldots, u_{i-1}, u_{i+1}, \ldots, u_n)$ and $C(u_1, \ldots, u_{i-1}, 0, u_{i+1}, \ldots, u_n) = 0$ for all $i \in \{1, \ldots, n\}$. Therefore, marginalising the full pmf (6.17) over the binary outcomes of the Bernoulli distribution leads to expressions that are only concerned with the random variables of the marginalised subset. It is then easy to verify that (6.17) satisfies the reproducibility property (iv). Consequently, the marginal probabilities (6.6) and (6.7) for the pmf (6.17) are given by

$$p_a^{\mathrm{C}}(\mathbf{y}_{ij}|\mathbf{R}) = p^{\mathrm{C}}(Y_i = 1, Y_{i+1} = 0, \ldots, Y_{j-1} = 0, Y_j = 1|\mathbf{R}), \tag{6.19}$$

and

$$p_b^{\mathrm{C}}(\mathbf{y}_{ji}|\mathbf{R}) = p^{\mathrm{C}}(Y_1 = 0, \ldots, Y_{i-1} = 0, Y_i = 1, Y_j = 1, Y_{j+1} = 0 \ldots, Y_n = 0|\mathbf{R}). \tag{6.20}$$

This implies that even though customers on a path, let us say c, may be correlated with customers not on path c, their dependence will have no consequences for the probability associated with the need to traverse the customers on path c. Therefore, the contribution of any path c to the total expected distance (6.5) is independent of the customers that lie beyond path c.

Secondly, for the correlation matrix $\mathbf{R} = \mathbf{I}$ it follows that, by definition, copulas

satisfy $C(u_1, u_2, \ldots, u_n) = u_1 u_2 \ldots u_n$. Using (6.18), it is then easy to show that

$$p^{\mathrm{C}}(\mathbf{y}|\mathbf{I}) = \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} (-1)^{k_1+\cdots+k_n} u_{1k_1} \times \cdots \times u_{nk_n} = \prod_{i=1}^{n} p_i^{y_i} (1-p_i)^{1-y_i}. \quad (6.21)$$

This is consistent with the multivariate pmf of independently distributed Bernoulli variables (6.8), i.e., $p^{\mathrm{C}}(\mathbf{y}|\mathbf{I}) = p^{\mathrm{i.d.}}(\mathbf{y})$. In similar fashion, it follows that $p_a^{\mathrm{C}}(\mathbf{y}_{ij}|\mathbf{I}) = p_a^{\mathrm{i.d.}}(\mathbf{y}_{ij})$ and $p_b^{\mathrm{C}}(\mathbf{y}_{ji}|\mathbf{I}) = p_b^{\mathrm{i.d.}}(\mathbf{y}_{ji})$. This observation aids the verification of property (iii).

Observe that (6.17) grows exponentially with the number of dimensions. Therefore, this formulation fails to satisfy property (v). If one were to implement (6.17) to calculate the expected length of a tour, only low-dimensional instances can be calculated. In fact, in such case one might as well resort to full enumeration, which requires the same computational effort. In alternative formulations involving elliptical copulas, one could resort to computing the rectangle probabilities instead (Panagiotelis et al., 2012). Such formulations typically drastically reduce the computational burden compared to the finite differences approach (6.17). However, it requires the evaluation of an n-dimensional integral, which becomes tedious for higher dimensions (e.g., Smith and Khaled, 2012).

Panagiotelis et al. (2012) propose a solution based on a discrete vine (D-vine) decomposition of the multidimensional pmf (6.17) into $2n(n-1)$ bivariate copulas, termed D-vine pair copula constructions (PCCs). It relies on the decomposition of the joint probability into a series of conditional probabilities:

$$\begin{aligned} p^{\mathrm{C}}(\mathbf{y}|\mathbf{R}) = \Pr(Y_1 = y_1, \ldots, Y_n = y_n|\mathbf{R}) &= \Pr(Y_1 = y_1|Y_2 = y_2, \ldots, Y_n = y_n, \mathbf{R}) \\ &\times \Pr(Y_2 = y_2|Y_3 = y_3, \ldots, Y_n = y_n, \mathbf{R}) \times \cdots \times \Pr(Y_n = y_n). \end{aligned} \quad (6.22)$$

The conditional probabilities of the form $\Pr(Y_i = y_i|\mathbf{V} = \mathbf{v}, \mathbf{R})$ on the right hand side of (6.22), with $\mathbf{V}$ indicating the set of conditioned variables from $\mathbf{Y}$ and $\mathbf{v}$ its

corresponding subset of realisations from $\mathbf{y}$, can be calculated by:

$$\Pr(Y_i = y_i|\mathbf{V}=\mathbf{v},\mathbf{R}) = \Bigg[\sum_{k_i=0,1} \sum_{k_j=0,1} (-1)^{k_i+k_j} C_{Y_i,V_j|\mathbf{V}_{\setminus j},\rho_{ij}} \Big(F_{Y_i|\mathbf{V}_{\setminus j}}(y_i - k_i|\mathbf{v}_{\setminus j},\mathbf{R}), F_{V_j|\mathbf{V}_{\setminus j}}(v_j - k_j|\mathbf{v}_{\setminus j},\mathbf{R}) \Big) \Bigg] \Big/ \Pr\big(V_j = v_j|\mathbf{V}_{\setminus j} = \mathbf{v}_{\setminus j},\mathbf{R}\big), \tag{6.23}$$

where

$$F_{Y_i|V_j,\mathbf{V}_{\setminus j}}\big(y_i|v_j,\mathbf{v}_{\setminus j},\mathbf{R}\big) = \Big[C_{Y_i,V_j|\mathbf{V}_{\setminus j},\rho_{ih}} \Big(F_{Y_i|\mathbf{V}_{\setminus j}}(y_i|\mathbf{v}_{\setminus j},\mathbf{R}_{\setminus j}), F_{V_j|\mathbf{V}_{\setminus j}}(v_j|\mathbf{v}_{\setminus j},\mathbf{R}_{\setminus i}) \Big) - C_{Y_i,V_j|\mathbf{V}_{\setminus j},\rho_{ih}} \Big(F_{Y_i|\mathbf{V}_{\setminus j}}(y_i|\mathbf{v}_{\setminus j},\mathbf{R}_{\setminus j}), F_{V_j|\mathbf{V}_{\setminus j}}(v_j - 1|\mathbf{v}_{\setminus j},\mathbf{R}_{\setminus i}) \Big) \Big] \Big/ \Pr(Z_k = z_k|\mathbf{Z}_{\setminus k} = \mathbf{z}_{\setminus k},\mathbf{R}). \tag{6.24}$$

I introduced the shorthand notation $F_{A|B}(a|b) = \Pr(A \leq a|B \leq b)$. Furthermore, $C_{A,B|C,D}(u_1,u_2)$ denotes the bivariate copula density for variables A and B conditional on C, evaluated at u_1 and u_2 using a correlation of D. Finally, V_j is used to describe an element of $\mathbf{V}$ and $\mathbf{V}_{\setminus j}$ is its complement. A stepwise procedure to compute (6.22) – (6.24) is described in the appendix of Panagiotelis et al. (2012).

The pmf in expression (6.17) can be rewritten into a D-Vine PCC using expressions (6.22) – (6.24) by plugging in the cdf of the Bernoulli distributed variables (6.18) and correlation matrix $\mathbf{R}$. Then

$$p^{\mathrm{C}}(\mathbf{y}|\mathbf{R}) = \prod_{i=1}^{N} \Pr(Y_i = y_i|\mathbf{Y}_{i+1,n} = \mathbf{y}_{i+1,n},\mathbf{R}), \tag{6.25}$$

where $\mathbf{Y}_{ij} \subseteq \mathbf{Y}$ denotes the vector of Bernoulli distributed variables corresponding to the customers on path $(i,\dots j) \in \tau$ with realisations $\mathbf{y}_{ij} \subseteq \mathbf{y}$ and cdfs $\mathbf{F}_{ij}(\mathbf{y}_{ij}) = \{F_i(y_i),\dots,F_j(y_j)\}$. The probability terms on the right hand side can be calculated with (6.23) and (6.24). Following the same line of reasoning, the marginal

probabilities (6.19) and (6.20) associated with traversing paths a and b are respectively given by

$$p_a^{\mathrm{C}}(\mathbf{y}_{ij}|\mathbf{R}) = \Pr(Y_i = 1|Y_j = 1, \mathbf{Y}_{i+1,j-1} = \mathbf{0}_{j-i-1}, \mathbf{R}) \times \Pr(Y_j = 1|\mathbf{Y}_{i+1,j-1} = \mathbf{0}_{j-i-1}, \mathbf{R}) \times \prod_{r=i+1}^{j-1} \Pr(Y_r = 0|\mathbf{Y}_{r+1,j-1} = \mathbf{0}_{j-r-1}, \mathbf{R}), \quad (6.26)$$

and

$$\begin{aligned} p_b^{\mathrm{C}}(\mathbf{y}_{ji}|\mathbf{R}) = &\Pr(Y_j = 1|Y_i = 1, \mathbf{Y}_{j+1,n} = \mathbf{0}_{n-j}, \mathbf{Y}_{1,i-1} = \mathbf{0}_{i-1}, \mathbf{R}) \\ &\times \Pr(Y_i = 1|\mathbf{Y}_{j+1,n} = \mathbf{0}_{n-j}, \mathbf{Y}_{1,i-1} = \mathbf{0}_{i-1}, \mathbf{R}) \\ &\times \prod_{l=1}^{i-1} \Pr(Y_l = 0|\mathbf{Y}_{l+1,i-1} = \mathbf{0}_{i-l-1}, \mathbf{R}) \\ &\times \prod_{k=j+1}^{n} \Pr(Y_k = 0|\mathbf{Y}_{k+1,n} = \mathbf{0}_{n-k}, \mathbf{Y}_{1,i-1} = \mathbf{0}_{i-1}, \mathbf{R}), \end{aligned} \quad (6.27)$$

where $\mathbf{0}_m$ denotes a $m \times 1$ vector of zeros. These expressions follow directly from applying the decomposition (6.25) to (6.26) for the customers on paths a and b. Again, the conditional probabilities on the right hand sides of (6.26) and (6.27) should be evaluated with (6.23)–(6.24). The last term of (6.26) and last two terms of (6.27) can be obtained computationally efficiently from (6.23), by noting that any copula $C_{A,B|C,D}(u_1, u_2)$ evaluates to 0 if $u_1 \vee u_2 = 0$. Since $F_k(y_k - 1) = 0$ if $y_k = 0$, only one of the four summations of (6.23) needs to be evaluated.

Substituting $p_a(\mathbf{y}_{ij})$ by $p_a^{\mathrm{C}}(\mathbf{y}_{ij}|\mathbf{R})$, and $p_b(\mathbf{y}_{ji})$ by $p_b^{\mathrm{C}}(\mathbf{y}_{ji}|\mathbf{R})$ in (6.5) and expanding the result yields the following equation for the expected length of an a priori tour τ^{C} using PCCs:

$$\mathbb{E}[L(\tau^{\mathrm{C}})] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} p_a^{\mathrm{C}}(\mathbf{y}_{ij}|\mathbf{R}) + \sum_{j=1}^{n} \sum_{i=j+1}^{n} d_{ji} p_b^{\mathrm{C}}(\mathbf{y}_{ji} \quad (6.28)$$

The minimisation of this formula with respect to its tour τ^{C} is an expression for the objective function of the PTSP under correlated stochastic customer conditions using D-Vine PCCs, referred to as Copula-CPTSP in the remainder of this thesis.

By definition (6.21), it follows that $p_a^{\text{C}}(\mathbf{y}_{ij}|\mathbf{I}) = p_a^{\text{i.d.}}(\mathbf{y}_{ij})$ and $p_b^{\text{C}}(\mathbf{y}_{ji}|\mathbf{I}) = p_b^{\text{i.d.}}(\mathbf{y}_{ji})$. Therefore, $\mathbb{E}[L(\tau^{\text{C}})] = \mathbb{E}\left[L\left(\tau^{\text{i.d.}}\right)\right]$ when dependence is absent such that property (iii) is satisfied. As D-Vine PCCs are computed in $O(n^2)$ time (Panagiotelis et al., 2012), the total computational complexity of (6.28) is equal to $O(n^4)$, compared to $O(n^2)$ for the objective function of the PTSP and $O(n^4)$ for the objective function of the heterogeneous Bahadur-CPTSP.

The D-vine PCC approach offers substantial computational benefits over the finite differences approach in the evaluation of general copula expressions. However, one could argue that the special form of the inputs $\mathbf{y}_{ij}$ and $\mathbf{y}_{ji}$ in the D-vine PCC pmf of (6.28) only requires the evaluation of 4 multidimensional copula expressions with the finite differences approach (6.17) for each pair (i, j), rather than $2^n - 1$ calculations. Consequently, if the chosen copula bears a closed-form multidimensional expression (e.g., special cases of Archimedean copulas), the number of calculations required to evaluate a pmf may be further reduced from $2n(n-1)$ to $4n$ with finite differences. Because this observation only holds true for a small selection of copula functions, and is not generalisable to other types of multidimensional copulas, I will not pursue such an approach here.

For the homogeneous CPTSP with equiprobable $p_i = p \forall i$ and equicorrelation $\rho_{ij} = \rho \forall i \neq j$ terms, the expected length of a tour $\tau^{\text{i.d.c.}}$ with identically distributed bivariate copula expressions, (6.28) simplifies to

$$\mathbb{E}[L(\tau^{\text{i.d.c.}})] = \sum_{r=0}^{n-2}\left(\prod_{k=0}^{r-1}\Pr(Y_k = 0|\mathbf{Y}_2 = \mathbf{1}_2, \mathbf{Y}_r = \mathbf{0}_r, \rho)\right) \times \left(1 + C_\rho(1-p, 1-p) - 2(1-p)\right) \times \sum_{j=1}^{n} d_{j,(j+1+r) \bmod n}. \tag{6.29}$$

In this expression, $C_\rho(u_1, u_2)$ denotes a bivariate copula with correlation ρ, $\mathbf{Y}_m$ denotes a $m \times 1$ vector of identically distributed Bernoulli variables with parameter p, and $\mathbf{1}_m$ denotes a $m \times 1$ vector of ones. It bears similarities to the homogeneous PTSP objective function (3.4), by noting that the first term is a conditional dependency term composed of an expanding series of absent customers, analogous

to the second term on the right hand side in (3.4). Furthermore, the second term on the right hand side is the result of expanding $\Pr(Y_1 = 1, Y_2 = 1)$ and reduces to p^2 iff $\rho = 0$. second The exact form of (6.28) depends on the choice of the underlying copula.

6.3 Properties

6.3.1 Limits of the Bahadur-CPTSP probability weight

Analogous to the homogeneous PTSP (3.4), the expected length of a homogeneous Bahadur-CPTSP tour (6.14) is given by the sum of a series of weighted distances. Following the reasoning of the weight-form notation of the expected length of an a priori tour (Jaillet, 1988), the probability weight of homogeneous PTSP distances can be expressed by

$$\alpha_{(r)}^{\text{i.i.d.}} = p^2(1-p)^r \tag{6.30}$$

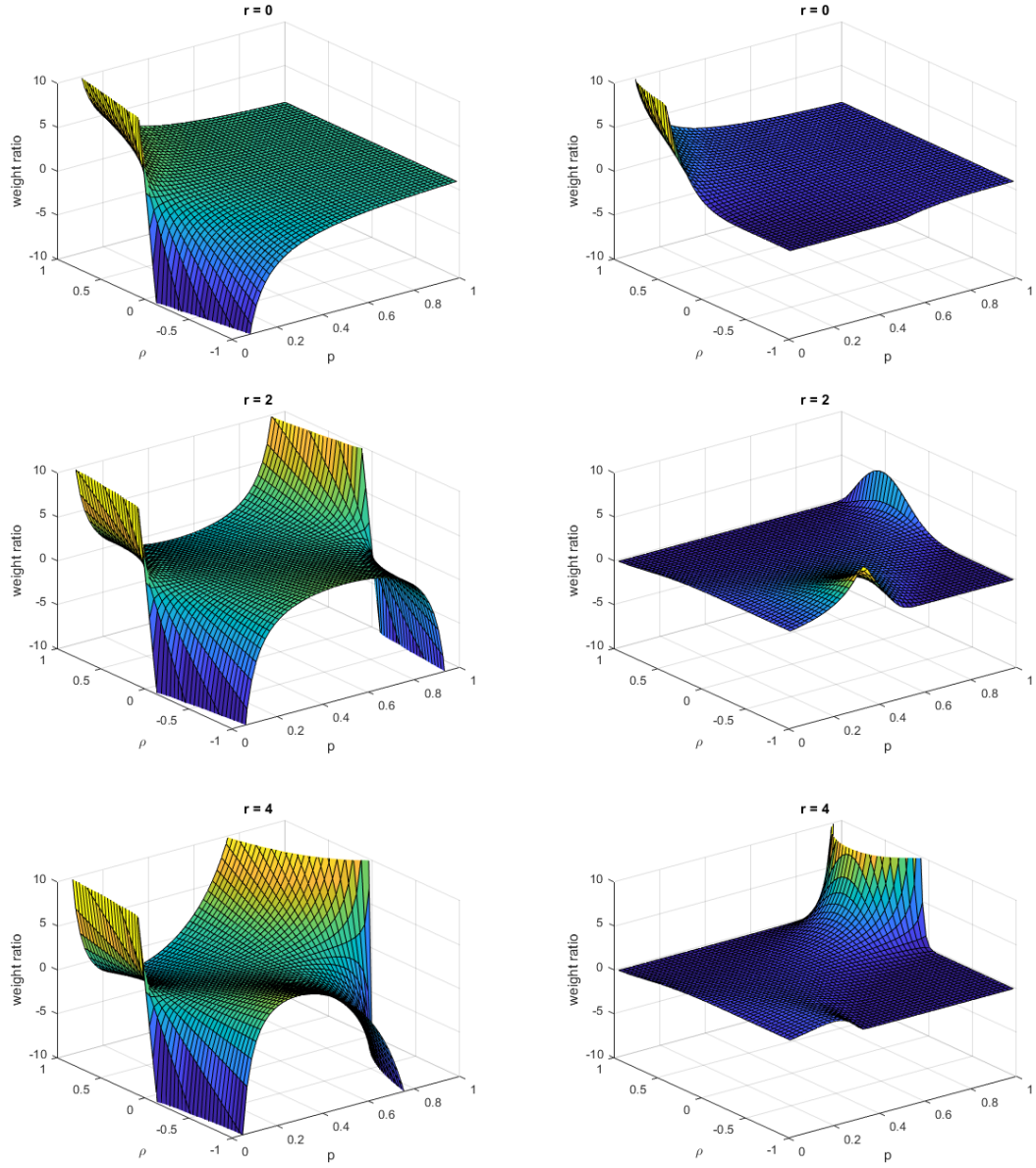
for $0 \leq r \leq n-2$, whereas the probability weight of a homogeneous Bahadur-CPTSP tour attains the form

$$\begin{aligned}\alpha_{(r)}^{\text{B}} &= p^2(1-p)^r \left(1 + \frac{\rho}{p(1-p)}(1-2(r+1)p + \frac{1}{2}(r+1)(r+2)p^2)\right) \\ &\equiv \alpha_{(r)}^{\text{i.i.d.}} \cdot w^{\text{B}}(r,p,\rho).\end{aligned} \tag{6.31}$$

As a result, each individual distance within the a priori tour, $L_\tau^{(r)} := \sum_{j=1}^{n} d(j,(j+1+r) \bmod n)$, is weighed either by (6.30) or (6.31) in the PTSP or Bahadur-CPTSP cases, respectively. The ratio between the PTSP and Bahadur-CPTSP probability weights (6.30) and (6.31) can be expressed as

$$w^{\text{B}}(r,p,\rho) \equiv \frac{\alpha_{(r)}^{\text{B}}}{\alpha_{(r)}^{\text{i.i.d.}}} = 1 + \frac{\rho}{p(1-p)}\left(1-2(r+1)p + \frac{1}{2}(r+1)(r+2)p^2\right). \tag{6.32}$$

Figure 6.1a illustrates $w^{\text{B}}(r,p,\rho)$ for different values of r. For increasing values of r, the tails of the probability weight ratio intensifies. More precisely, the weight ratio



(a) Bahadur probability weight ratios (6.32).

(b) Copula probability weight ratios (6.33) for the Gaussian copula.

Figure 6.1: PTSP to CPTSP probability weight ratios for values of $r = \{0, 2, 4\}$.

$w^{\mathrm{B}}(r,p,\rho)$ is increasing in r, increasing (decreasing) in ρ for positive (negative) values of ρ, and a non-linear function of p (see Table 6.1).

$\mathrm{sign}[\delta w^{\mathrm{B}}(r,p,\rho)/\delta p]$		$r<2$	$r=2$		$r>2$	
		$0\le p\le 1$	$0\le p\le 0.5$	$0.5<p\le 1$	$0<p\le\varphi$	$\varphi<p\le 1$
ρ	(-1, 0]	+	+	−	+	−
	[0, 1)	−	−	+	−	+

Table 6.1: Regimes where the probability weight (6.32) is increasing (+) or decreasing (−) in p for different values of ρ and r. In the probability regimes, $\varphi \equiv \sqrt{\binom{r}{2}}/(\binom{r}{2}-1)-\binom{r}{2}^{-1}$.

Table 6.2 reports the limits of the probability weights (6.30) and (6.31), and probability ratio (6.32) for specific values of r, p and ρ. Combining these characteristics with the results of Table 6.1, it can be observed that the value of the probability ratio (6.32) can be either bigger or smaller than 1 – or put differently, the probability weight of a PTSP distance (6.30) can be either bigger or smaller than the probability weight of a CPTSP distance (6.32) – for different values of r, p and ρ. Consequently, the expected length of the a priori tour with independent customers (3.4) may also be either bigger or smaller than the expected of the same tour in Bahadur sense (6.14) for both positive and negative homogeneous correlation coefficients ρ. Simulations suggest that for almost any choice of homogeneous customer presence probability p, the expected length decreases with increasing ρ for instances of $n \geq 20$ customers.

6.3.2 Limits of the copula-CPTSP probability weight

Unlike the Bahadur-CPTSP expected tour length (6.14), the Copula-CPTSP expected tour length (6.28) does typically not attain a closed-form expression for its probability density function. Only in a few exceptional cases, e.g. specific cases of Archimedean copulas, a closed-form expression may be derived. As a result, also the homogeneous probability weight

$$\alpha^{\mathrm{C}}_{(r)} = \big(1 + C_\rho(1-p,1-p) - 2(1-p)\big) \times \left(\prod_{k=0}^{r-1} \Pr(Y=0|\mathbf{Y}_2=\mathbf{1}_2,\mathbf{Y}_r=\mathbf{0}_r,\rho)\right),$$

	$\alpha_{(r)}^{\text{i.i.d.}}$	$\alpha_{(r)}^{\text{B}}$	$w^{\text{B}}(r,p,\rho)$
$\lim_{r\to 0}$	p^2	$p^2(1+\rho\frac{1-p}{p})$	$1+\rho\frac{1-p}{p}$
$\lim_{r\to\infty}$	0	0	$+\infty$ if $\rho>0$
			1 if $\rho=0$
			$-\infty$ if $\rho<0$
$\lim_{p\to 0}$	0	0	$+\infty$ if $\rho>0$
			1 if $\rho=0$
			$-\infty$ if $\rho<0$
$\lim_{p\to 1}$	1 if $r=0$	1 if $r=0$	1 if $r=0$
	0 if $r\geq 1$	0 if $r\geq 1$	$1-2\rho$ if $r=1$
			$-\infty$ if $r\geq 2$ and $\rho>0$
			1 if $r\geq 2$ and $\rho=0$
			$+\infty$ if $r\geq 2$ and $\rho<0$
$\lim_{\rho\to -1}$	$p^2(1-p)^r$	$p^2(1-p)^r\cdot w^{\text{B}}(r,p,-1)$	$w^{\text{B}}(r,p,-1)$
$\lim_{\rho\to 0}$	$p^2(1-p)^r$	$p^2(1-p)^r$	1
$\lim_{\rho\to 1}$	$p^2(1-p)^r$	$p^2(1-p)^r\cdot w^{\text{B}}(r,p,1)$	$w^{\text{B}}(r,p,1)$

Table 6.2: Limits of the independent, Bahadur and Bahadur:independent probability weights (6.30), (6.31) and (6.32) for boundary values of r, p and ρ.

and the associated probability ratio

$$w^{\text{C}}(r,p,\rho)\equiv\frac{\alpha_{(r)}^{\text{C}}}{\alpha_{(r)}^{\text{i.i.d.}}}=\frac{\left(1+C_\rho(1-p,1-p)-2(1-p)\right)}{p^2(1-p)^r}\times\left(\prod_{k=0}^{r-1}\Pr\left(Y=0|\mathbf{Y}_2=\mathbf{1}_2,\mathbf{Y}_r=\mathbf{0}_r,\rho\right)\right)\tag{6.33}$$

does generally not attain a closed-form expression analogous to (6.31) and (6.32). Figure 6.1b illustrates the simulated weight probability ratios of a Gaussian copula-based probability weight versus the independent probability weight (6.30).

Nonetheless, some general probability results may be derived. First of all, similar to the Bahadur probability weight, as copula probabilities decrease with

increasing dimensions we have that

$$\lim_{r\to 0} \alpha^{\mathrm{C}}_{(r)} = C(p,p) \qquad \text{and} \qquad \lim_{r\to\infty} \alpha^{\mathrm{C}}_{(r)} = 0.$$

Furthermore, since $C(0,u) = C(u,0) = 0$ and $C(1,u) = C(u,1) = u$, we have that

$$\lim_{p\to 0} \alpha^{\mathrm{C}}_{(r)} = 0, \qquad \lim_{p\to 1} \alpha^{\mathrm{C}}_{(r)} = 0 \text{ if } r = 0, \quad \text{and} \qquad \lim_{p\to 1} \alpha^{\mathrm{C}}_{(r)} = 0 \text{ if } r \geq 1.$$

Finally, as demonstrated analytically in paragraph 6.2.5, as with all copulas, for $\mathbf{R} = \mathbf{I}$ the probability expression (6.28) reduces to the probability expression in independent sense:

$$\lim_{\rho\to 0} \alpha^{\mathrm{C}}_{(r)} = p^2(1-p)^r.$$

As a result, we have for any copula that

$$\lim_{\rho\to 0} w^{\mathrm{C}}(r,p,\rho) = 1.$$

6.3.3 Practical implications of the Bahadur-CPTSP

The Bahadur model, as the underlying pmf of the Bahadur-CPTSP objective function, comes with a number of advantages and disadvantages. First, the expected length of a Bahadur-CPTSP a priori tour (6.13) is easy to interpret, and its connection with the expected length of a PTSP a priori tour (6.5) is obvious (property iii). As a result, (6.13) bears little complexity and is easy to implement. Second, due to the fact that the individual terms in the joint pmf can be separated, the model can be described as a function of merely marginal probabilities and pairwise joint interactions (property i). Therefore, a practitioner only needs to specify $n + n(n-1)/2$ terms, rather than the joint probabilities describing all $2^n - 1$ possible individual outcomes (property ii). Lastly, the Bahadur model is reproducible (property iv). That is, marginalising the joint distribution (6.13) with respect to a $|S| \times 1$ subset of $\mathbf{y}$, $|S| < N$, leads to probability distributions of the same type. Other types of marginal models (e.g. log-linear models) may suffer

from the inability of expressing the parameters of a marginalised model in terms of the original marginal moments (Fitzmaurice et al., 1993).

But the Bahadur model also comes with a number of drawbacks. As Bahadur (1961) points out, and further illustrated by Declerck et al. (1998) with a number of empirical results, the second-order correlations of the Bahadur model can only admit a certain range of values. That is, for certain values of r, p and ρ, the probability weight $\alpha^{\mathrm{B}}_{(r)}$ could assume values beyond the permitted range $[0,1]$. This can also be observed from Figure 6.1a, where negative values indicate infeasible regions.

More specifically, it can be demonstrated that correlations should satisfy the inequalities given by

$$-\frac{2}{n(n-1)}\min\left(\frac{p}{1-p},\frac{1-p}{p}\right)\leq\rho\leq\frac{2p(1-p)}{(n-1)p(1-p)+\frac{1}{4}-\gamma_0}, \tag{6.34}$$

where

$$\gamma_0=\min_{z=0}^{n}\{[z-(n-1)p-\frac{1}{2}]^2\},$$

in order to yield a valid probability outcome. Therefore, depending on the size and probability of the problem under consideration, one can only choose from a limited range of correlations describing the joint dependence between customers. For homogeneous probability values in the specific case of the homogeneous Bahadur-CPTSP (6.14), the allowed correlation space is somewhat different due to the fact that we only need to consider outcomes of the form $\mathbf{y}'=(1,0,0,\ldots,0,1)$.

Figure 6.2 shows the resulting boundaries of the (constant) correlation parameter ρ for different values of the homogeneous probability parameter p and the dimensions $3\leq n\leq 30$. The region under the solid line (I) indicates the feasible region, whereas the shaded area (II) indicates the infeasible region. With increasing dimension, the possible correlation range drastically decreases for probabilities $p<0.65$.

These strict correlation bounds can be somewhat relaxed. First of all, increasing the Bahadur model (6.11) beyond the second order allows for a wider range of allowed values. This, however, imposes a huge burden on the ease of implementation of the model, since it would require the specification of all higher order correlations $\rho_{ij\ldots z}$. In practice, these higher order correlations are often unknown. Additionally, the increase in the allowed correlation range is only limited (Declerck et al., 1998).

An alternative solution that allows for a wider range of possible dependence interactions would be to introduce a calculation threshold. This threshold cuts off the summation $r = \{0, \ldots, n-2\}$ of the Bahadur model expansion in the expression for the homogeneous a priori expected length (6.14) at a certain level β, with $0 \leq \beta < n-2$. Cutting off the expected length at predefined level β results in a summation over a truncated range of dimensions $r = 0, \ldots, \beta \leq n-2$, thereby increasing the permitted correlation range (Losee, 1994). Although this seems reasonable from a practitioner's perspective, one should bear in mind that this approach induces an error with respect to the true expected length of the correlated a priori tour. The magnitude of the error depends on the chosen parameter values and the characteristics of the problem at hand. Empirical tests (not reported here) suggest that, one would still require the calculation of $\beta \geq 9$ terms for probabilities $p \leq 0.5$ to maintain reasonable error bounds in small to medium-sized instances. Unfortunately, such values of β only produce marginally improved correlation bounds (see Figure 6.2). Therefore, I do not opt to pursue such an approach in this paper.

6.3.4 Practical implications of the copula-CPTSP

In contrast to the Bahadur-CPTSP, the Copula-CPTSP does not suffer from equally strict correlation bounds. However, the correlations that can be used by the copulas in the Copula-CPTSP model are still limited by the Fréchet-Hoeffding bounds (Embrechts et al., 2002). The exact bounds depend on the chosen copula, and only under specific circumstances bear an analytical expression. For example,

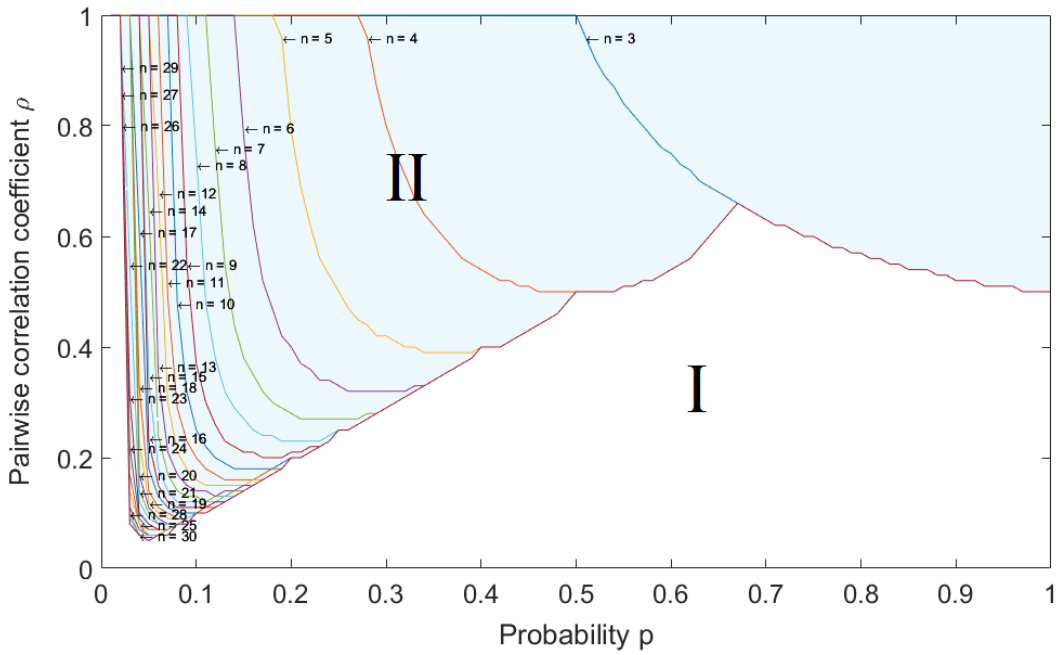


Figure 6.2: Feasible region (I) and infeasible region (II) of the Bahadur model for different values of the homogeneous probability parameter p, correlation parameter ρ, and dimension n.

some types of Archimedean copulas (e.g., Clayton and Gumbel copulas) do not allow negative correlations (i.e., $\rho < 0$). For elliptical copulas (e.g., Gaussian and Student-t copulas), such correlation bounds typically do not exist. For example, for Gaussian copulas we can choose any homogeneous correlation matrix in the range $\mathbf{R} \in [-1, 1]^{n \times n}$.

Although elliptical copulas admit a wide range of values for the correlation coefficients, this does not imply that an implementer can simply pick any heterogeneous correlation value in the range $\rho_{ij} \in [-1, 1]$ to describe the dependency between a pair of customers i and j. Because the correlation matrix $\mathbf{R}$ needs to be symmetric and positive semidefinite, the correlation matrix

$$\mathbf{R} = \begin{pmatrix} 1 & -0.9 & 0 \\ -0.9 & 1 & -0.9 \\ 0 & -0.9 & 1 \end{pmatrix}$$

is not permitted since it does not satisfy the latter property. One should therefore always be careful to check whether the basic conditions of the specified correlation matrix are met before proceeding with the implementation.

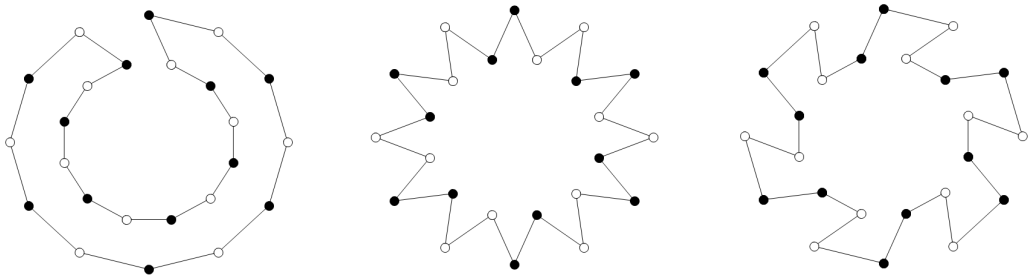
6.4 Computational results

6.4.1 Examples of good TSP, PTSP and CPTSP tours

Jaillet (1985) demonstrates that a good tour for the deterministic TSP does not necessarily need to be a good tour for the PTSP, and vice versa. He uses a 24-node star-shaped example to illustrate the differences. Building on his example, this section demonstrates that a good tour for the TSP or a good tour for the PTSP does not necessarily coincide with a good tour for the CPTSP.

Identical to the problem introduced by Jaillet (1985), let us take a graph consisting of 24 nodes grouped into an inner regular polygon and an outer regular polygon of 12 nodes each. The distance between two successive nodes of the outer 12-gon is 1, whereas the distance between two successive nodes of the inner 12-gon is 0.629. All customers are assigned a homogeneous probability of $p = 0.5$ of requiring a visit. The original problem is expanded with a colour-coding scheme, in which two nodes of the same colour (i.e., any pair of white nodes or any pair of black nodes) are completely independent, but two nodes of a different colour (i.e., any pair consisting of one white node and one black node) bear a positive correlation of 0.9.

Figure 6.3 illustrates what constitutes a good tour for the TSP, PTSP and CPTSP. The objective of the TSP is the minimisation of the length (6.1) with respect to the tour τ. This results in a C-shaped tour (see Figure 6.3a) for the TSP. This is not surprising, since this tour exhibits the closest geometric similarity to a combination of two (mathematically optimal) circle-shaped 12-gons along the inner and outer customers.



(a) Good TSP tour. (b) Good PTSP tour. (c) Good CPTSP tour.

Figure 6.3: Good TSP, PTSP, and CPTSP tours.

In contrast, minimising the expected length of the homogeneous PTSP (3.3) results in a star-shaped tour (see Figure 6.3b) for the chosen probability. The difference with respect to the optimal TSP tour can be explained as follows. In the a posteriori tour of the PTSP, it is likely that only a subset of the 24 nodes requires a visit. On average, $np = 12$ nodes require a visit. It is easy to spot that in case only the nodes on the outer 12-gon or only the nodes on the inner 12-gon require a visit, the tour in Figure 6.3b reduces to one of the two (mathematically optimal) circle-shapes along the customers on either the inner or outer polygon. Since the a posteriori realisation can be any probabilistic combination of these two extreme outcomes, the minimisation of (3.3) results in the star-shaped tour of figure 6.3b.

Taking the example one step further, minimising the expected length of the copula-based CPTSP (6.28) results in a gear-shaped tour (see Figure 6.3c) for our choice of variables. Following the line of thought of the star-shaped example for the PTSP, the high correlation between two nodes of different colours results in a slightly higher preference towards visiting two successive nodes on the inner and outer 12-gons – corresponding to the good TSP solution – whereas the probabilistically independent nature of nodes of the same colour pulls the solution towards the star-shape – corresponding to a good PTSP solution. The combination of the two solutions that minimises the expected length of the Copula-CPTSP a priori tour produces a gear-shaped tour of figure 6.3c.

Length definition	Figure 6.3a	Figure 6.3b (ΔFig. 6.3a)		Figure 6.3c (ΔFig. 6.3a)	
TSP (6.1)	19.561	19.698	(0.7%)	19.623	(0.3%)
PTSP (3.3)	16.109	12.282	(-23.8%)	12.911	(-19.9%)
CPTSP (6.28)	11.480	11.173	(-2.7%)	10.612	(-7.6%)

Table 6.3: (Expected) lengths of the TSP, PTSP and CPTSP tours of Figure 6.3 according to the length definitions of a TSP tour (6.1); a PTSP tour (3.3); and a CPTSP tour (6.28).

Table 6.3 reports the (expected) lengths for the tours corresponding to Figure 6.3. Albeit the CPTSP length of the good PTSP tour (Figure 6.3b) is 2.7% better than the CPTSP length of the good TSP tour (Figure 6.3a), the good CPTSP (Figure 6.3c) tour outperforms the CPTSP length of the good PTSP tour by another 4.9%. Similarly, the reported lengths according to the TSP and PTSP objective function definitions are also consistent with our previous observations. These figures support the notion that neither a good TSP tour nor a good PTSP tour necessarily coincides with a good CPTSP tour.

6.4.2 Implementation

For the remaining empirical experiments in this study, I obtained three instances from TSPLIB[1] named kroA100, d198 and rat783, consisting of 100, 198 and 783 nodes, respectively. These instances bear different characteristics with respect to the number of dimensions and degree of dispersion of the nodes across the plane. For each instance, I specify an $n \times 1$ marginal probability vector describing the stochastic requirements to visit the customers, and an $n \times n$ symmetric and positive semidefinite correlation matrix describing the pairwise correlations between two customers requiring a visit, respectively.

The objective of the computational experiments is not to optimise the expected length of the a priori tour, but rather to show the impact of correlation on the expected length. Therefore, only one tour is generated for each instance, which is subsequently retained throughout the calculations of a particular instance. This

[1] https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.

way all differences in the reported expected lengths can be directly attributed to the impact of correlation, rather than differences in the performance of the algorithm.

I use the Concorde algorithm (Applegate et al., 2006) to generate tours for each instance. Although originally developed as an exact algorithm to find optimal solutions for the TSP, Concorde has also been reported to produce relatively good results in stochastic settings (Bianchi and Gambardella, 2007; Balaprakash et al., 2010). For some instances, the Concorde algorithm produces more than one unique tour, each bearing the same deterministic length. However, these different tours do not necessarily also bear identical PTSP expected tour lengths, nor identical CPTSP expected tour lengths. I did not perform an exhaustive search over all possible tours produced by Concorde to find the best tour in PTSP or CPTSP sense.

Motivated by the desirable model requirements of section 6.2.1, I truncate the Bahadur model after the second order and the copula model after the second tree. That is, I only consider pairwise probabilities and conditional probabilities up to the first order, and consequently, limit ourself to the specification of pairwise correlations only. The conditional dependencies of trees in the Copula-CPTSP after the second tree are set to zero, following the implementation proposed by Panagiotelis et al. (2012). The correlations of the first and second trees are both set to the same value.

6.4.3 Homogeneous correlation

Table 6.4 reports the results for three homogeneous CPTSP instances of different sizes over a (homogeneous) low, medium and high probability set of $p = 0.1$, $p = 0.5$ and $p = 0.9$, respectively. Homogeneous correlations are defined in the range $[-0.9, 0.9]$ with decimal increments, to analyse the impact of correlation under stochastic conditions. The reader is referred to the introduction of this chapter for examples of situations where setting positive or negative correlations might be appropriate. Although completely homogeneous correlations are arguably rarely encountered in practice, these results allow us to study the marginal impact

Instance	Corr.	Bahadur-CPTSP			Copula-CPTSP		
		$p = 0.1$	$p = 0.5$	$p = 0.9$	$p = 0.1$	$p = 0.5$	$p = 0.9$
kroA100	-0.9				8065.851	13160.746	15275.100
	-0.8				8065.714	13120.140	15275.098
	-0.7				8063.133	13085.794	15275.046
	-0.6				8052.088	13051.043	15274.785
	-0.5				8026.741	13014.131	15274.125
	-0.4				7983.497	12973.805	15272.934
	-0.3				7920.296	12927.968	15271.131
	-0.2				7835.453	12874.187	15268.636
	-0.1				7726.804	12809.964	15265.314
Ind.	0.0	7591.124	12732.974	15260.920	7591.124	12732.974	15260.920
	0.1	6394.166	12591.949	15246.154	7423.642	12641.199	15255.032
	0.2		12450.924	15231.388	7217.495	12532.734	15246.934
	0.3		12309.900	15216.623	6962.899	12405.138	15235.496
	0.4		12168.875	15201.857	6645.583	12253.877	15219.073
	0.5		12027.850	15187.091	6243.212	12068.643	15195.425
	0.6				5716.429	11826.096	15161.270
	0.7				4988.799	11477.460	15109.927
	0.8				3931.979	10920.873	15021.756
	0.9				2547.622	9845.772	14828.716
d198	-0.9				8082.826	13057.333	15253.800
	-0.8				8082.693	13018.050	15253.798
	-0.7				8080.182	12984.833	15253.748
	-0.6				8069.445	12951.243	15253.495
	-0.5				8044.765	12915.589	15252.857
	-0.4				8002.516	12876.704	15251.705
	-0.3				7940.448	12832.636	15249.960
	-0.2				7856.582	12781.197	15247.547
	-0.1				7748.423	12720.317	15244.334
Ind.	0.0	7612.421	12648.247	15240.088	7612.421	12648.247	15240.088
	0.1	6364.313	12518.339	15225.828	7443.540	12563.591	15234.408
	0.2		12388.431	15211.567	7234.764	12464.890	15226.628
	0.3		12258.523	15197.307	6976.326	12349.609	15215.731
	0.4		12128.615	15183.047	6654.197	12212.133	15200.301
	0.5		11998.707	15168.786	6246.538	12039.897	15178.434
	0.6				5714.732	11806.663	15147.084
	0.7				4983.237	11461.700	15099.148
	0.8				3924.587	10904.562	15013.670
	0.9				2541.047	9830.690	14822.274

Table 6.4: Expected lengths for the homogeneous CPTSP for equiprobable customer presences of $p = 0.1$, 0.5, and 0.9, and equicorrelations (Corr.) in the range $[-0.9, 0.9]$ with decimal increments.

Instance	Corr.	Bahadur-CPTSP			Copula-CPTSP		
		$p = 0.1$	$p = 0.5$	$p = 0.9$	$p = 0.1$	$p = 0.5$	$p = 0.9$
rat783	-0.9				4183.944	7366.264	8533.400
	-0.8				4183.866	7333.397	8533.398
	-0.7				4182.376	7305.834	8533.356
	-0.6				4175.987	7278.421	8533.145
	-0.5				4161.341	7249.762	8532.610
	-0.4				4136.463	7218.980	8531.646
	-0.3				4100.399	7184.740	8530.187
	-0.2				4052.565	7145.640	8528.173
	-0.1				3992.276	7100.446	8525.508
Ind.	0.0	3918.430	7048.070	8522.023	3918.430	7048.070	8522.023
	0.1	3358.778	6954.896	8510.396	3829.259	6987.404	8517.434
	0.2		6861.723	8498.768	3722.072	6917.044	8511.285
	0.3		6768.549	8487.141	3592.967	6834.963	8502.905
	0.4		6675.375	8475.513	3436.519	6738.015	8491.343
	0.5		6582.201	8463.886	3245.513	6620.945	8475.244
	0.6				3013.444	6474.027	8452.481
	0.7				2723.128	6277.363	8419.150
	0.8				2355.999	5989.405	8365.986
	0.9				1813.976	5530.259	8264.444

Table 6.4: Expected lengths for the homogeneous CPTSP for for equiprobable customer presences of $p = 0.1$, 0.5, and 0.9, and equicorrelations (Corr.) in the range $[-0.9, 0.9]$ with decimal increments *(continued)*.

of correlation on the expected length under controlled (theoretical) conditions. The gaps in the results of the Bahadur-CPTSP are due to the limitations in the correlation bounds.

The expected lengths estimated by the Bahadur-CPTSP are up to 16.28% smaller than the Copula-CPTSP, with differences increasing as either the probability or the size of the problem diminishes. In the case of zero correlation, the expected lengths of the CPTSP are identical to the PTSP. Then, as the correlation gradually increases, the expected lengths diminishes. On the other hand, with increasing probability, the effect of correlation on the total expected length decreases. These result can be explained by weighing the distances by the sum of a series of non-linear probability factors (section 6.3).

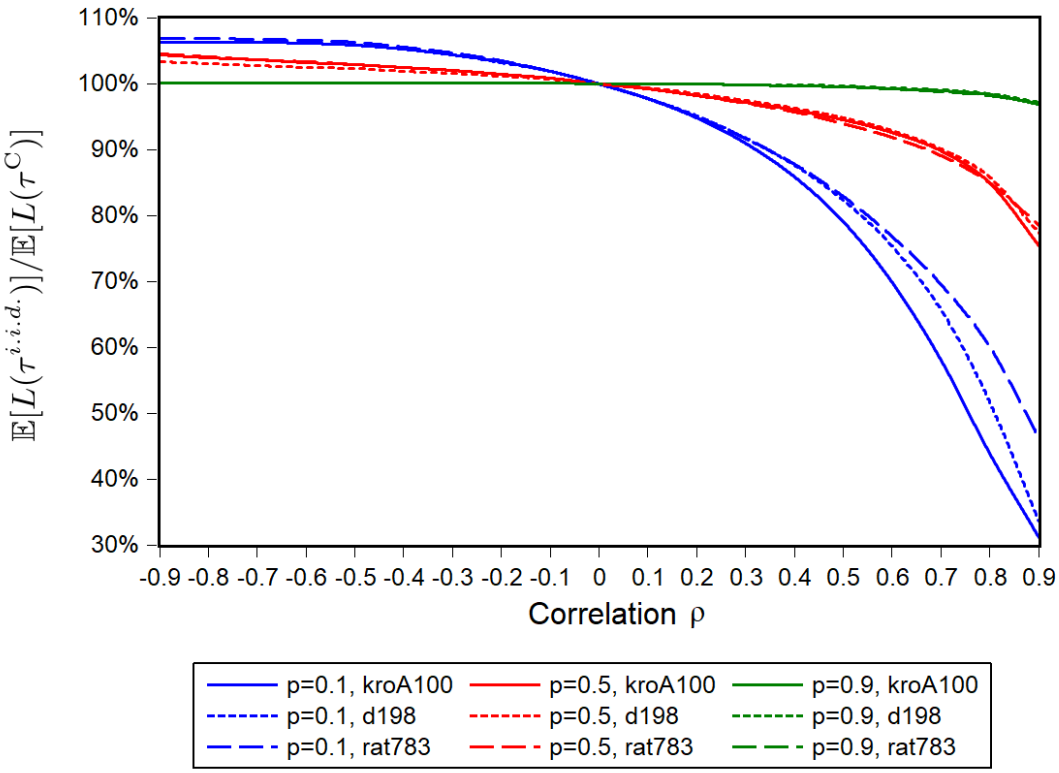


Figure 6.4: Relative expected lengths of homogeneous Copula-CPTSP tours versus independence (100%).

Figure 6.4 shows the results of copula-CPTSP tour lengths in table 6.4 relative to the expected length under independence, i.e., $\mathbb{E}[L(\tau^{i.i.d.})]/\mathbb{E}[L(\tau^{C})]$. Large positive correlations seem to have a more outspoken effect on the relative expected length than large negative correlations. The effect of correlation dissipates as the correlation reaches -0.9, whereas large positive correlations can have a massive effect on the expected length of the reported instances versus complete independence. In general, the effect of correlation on the expected length increases with decreasing probability. More specifically, in the case of $p = 0.9$, correlation seems to have a negligible impact on the expected length of a tour for the negative correlations, but gradually increases to a difference of -3% for positive correlations. In the case of $p = 0.1$, on the other hand, correlation ranges from +6% for large negative correlations, and increases to a whopping -69% for large positive correlations versus the expected length of the PTSP.

On average, the effect of correlation becomes less pronounced as the number of customers grows. However, this statement is not true for all correlation ranges. For example, when the correlation is negative, this trend vanishes. In addition, for expected lengths in the correlation range $(0.0, 0.7]$ if $p = 0.5$ or $p = 0.9$, a clear trend is absent. The expected lengths of the 162 Copula-CPTSP cases in Table 6.4 deviate 6.821% on average from the expected lengths under independence (standard error 0.945%). A student's t-test (T-score 7.238, p-value 0.000) reveals that it is safe to conclude that correlation has a significant effect on the expected length of a tour.

6.4.4 Heterogeneous correlation

The following problem configurations are chosen to study the impact of heterogeneous correlation. First, low, medium, and high homogeneous probabilities $p_i = \{0.1, 0.5, 0.9\} \forall i \in \{1, \ldots, n\}$ are assigned to the customers. Then, for a prespecified number of clusters $K = \{2, 5, 10\}$, the k-means algorithm (Lloyd, 1982) is applied to the set of nodes N resulting in k clusters, $k \in K$. For each number of clusters k, I report the results under the absence of in-cluster correlation, and various degrees of negative and positive correlations between clusters of ±0.1, ±0.5, and ±0.9. In addition, I report the results under the absence of correlation between clusters, and instead let the degree of in-cluster correlation vary by the same degrees. Due to the limited correlation ranges that are admitted in the Bahadur-CPTSP, too many results were found to be invalid. Therefore, the results for the Bahadur-CPTSP have been omitted.

Table 6.5 reports the results of the expected length for several CPTSP instances with heterogeneous correlations. For heterogeneous correlations between customers within clusters, and independence between customers of different clusters, a clear trend of declining expected lengths becomes apparent as the correlation increases. This observation remains valid regardless of cluster size k or probability p_i. Futhermore, for all positive correlations of the kroA100 and rat783 instances, increasing the number of clusters has a positive effect on the

Instance	Corr.		$p = 0.1$			$p = 0.5$			$p = 0.9$		
	In	Out	$k = 2$	$k = 5$	$k = 10$	$k = 2$	$k = 5$	$k = 10$	$k = 2$	$k = 5$	$k = 10$
kroA100	-0.9	0.0	9804.575	9768.804	9744.808	17440.514	17356.331	17342.648	20539.100	20536.320	20537.597
	-0.5	0.0	9759.509	9724.788	9701.973	17161.739	17119.152	17090.098	20537.240	20534.668	20535.810
	-0.1	0.0	9407.736	9392.570	9383.295	16800.174	16787.246	16773.821	20520.503	20519.726	20519.874
	0.1	0.0	9043.635	9067.637	9081.868	16530.794	16545.915	16563.779	20501.363	20502.354	20502.533
	0.5	0.0	7499.666	7833.857	8031.733	15763.082	15866.355	16006.585	20401.665	20415.889	20421.795
	0.9	0.0	3388.345	4511.001	5469.125	13707.040	14631.769	15068.257	20026.912	20151.789	20163.761
	0.0	-0.9	9279.082	9342.574	9373.530	16628.667	16768.006	16832.186	20509.517	20512.312	20511.604
	0.0	-0.5	9276.209	9334.475	9363.495	16654.125	16734.282	16788.451	20509.724	20512.314	20511.700
	0.0	-0.1	9254.785	9272.260	9282.250	16670.865	16685.577	16700.368	20511.505	20512.300	20512.234
	0.0	0.1	9236.103	9215.587	9202.558	16675.736	16662.239	16645.849	20513.165	20512.246	20512.206
	0.0	0.5	9185.508	9051.532	8943.157	16673.578	16620.061	16524.560	20517.625	20512.611	20511.328
	0.0	0.9	9139.989	8901.057	8676.301	16663.434	16576.486	16398.128	20522.594	20503.943	20501.522
d198	-0.9	0.0	8059.603	8037.617	8043.056	13162.128	13160.029	13117.303	15275.150	15275.160	15273.299
	-0.5	0.0	8021.064	8000.144	8005.664	13015.341	13010.839	12983.185	15274.172	15274.177	15272.453
	-0.1	0.0	7725.077	7716.698	7719.686	12810.197	12808.222	12800.668	15265.332	15265.312	15264.784
	0.1	0.0	7425.773	7438.263	7433.342	12641.070	12643.940	12654.340	15255.006	15255.073	15255.681
	0.5	0.0	6265.488	6412.722	6368.161	12072.822	12120.639	12208.289	15195.305	15197.921	15201.832
	0.9	0.0	2820.865	3341.088	3894.562	10055.827	10865.485	11141.446	14846.179	14982.372	14900.183
	0.0	-0.9	7597.583	7635.384	7618.928	12731.307	12744.577	12811.393	15260.859	15260.912	15262.785
	0.0	-0.5	7597.044	7631.396	7616.637	12732.096	12742.250	12780.968	15260.863	15260.916	15262.652
	0.0	-0.1	7592.888	7602.524	7598.732	12732.769	12735.128	12743.733	15260.901	15260.930	15261.457
	0.0	0.1	7589.064	7578.366	7582.228	12733.156	12730.787	12721.738	15260.945	15260.897	15260.285
	0.0	0.5	7577.921	7517.616	7532.971	12733.217	12723.005	12674.767	15261.118	15260.671	15256.753
	0.0	0.9	7568.252	7475.585	7483.985	12733.614	12721.016	12636.131	15258.809	15256.082	15241.859
rat783	-0.9	0.0	4174.445	4162.526	4146.069	7361.709	7356.062	7356.569	8533.220	8532.931	8533.060
	-0.5	0.0	4152.741	4141.760	4126.519	7245.137	7242.058	7239.967	8532.441	8532.172	8532.288
	-0.1	0.0	3989.744	3986.108	3980.924	7098.636	7097.490	7096.539	8525.444	8525.354	8525.373
	0.1	0.0	3832.322	3837.137	3844.042	6989.867	6991.738	6993.378	8517.538	8517.658	8517.665
	0.5	0.0	3271.092	3320.642	3385.771	6641.517	6666.113	6694.610	8477.223	8479.168	8480.672
	0.9	0.0	1959.813	2195.911	2413.305	5642.309	5866.101	6093.345	8285.424	8319.326	8356.305
	0.0	-0.9	3928.253	3943.262	3963.797	7059.771	7070.544	7076.924	8522.239	8522.555	8522.471
	0.0	-0.5	3927.403	3941.080	3959.877	7056.291	7062.726	7067.939	8522.226	8522.520	8522.445
	0.0	-0.1	3921.024	3924.932	3930.464	7050.101	7051.535	7052.740	8522.093	8522.187	8522.174
	0.0	0.1	3915.478	3911.074	3904.676	7045.857	7044.368	7043.124	8521.931	8521.819	8521.824
	0.0	0.5	3900.472	3874.619	3834.220	7035.468	7028.251	7021.842	8521.273	8520.549	8520.468
	0.0	0.9	3884.394	3842.148	3769.533	7025.598	7014.925	7003.405	8519.617	8516.415	8514.838

Table 6.5: Expected lengths for the heterogeneous CPTSP for equiprobable customer presences of $p = \{0.1, 0.5, 0.9\}$, and heterogeneous in-cluster (In) correlation versus between-cluster cluster correlation (Out) of ±0.1, ±0.5, and ±0.9 for cluster sizes of $k = \{2, 5, 10\}$.

expected length. This statement also holds true for all but 3 out of 15 (80%) positive correlation cases in the d198 instance. A negated and slightly less pronounced trend is also visible for the negative in-cluster correlation values, where in 42 out of 54 cases (78%) a higher number of clusters yields a lower expected length.

These results can be explained as follows. A positive (negative) in-cluster correlation can be interpreted as the effect of a customer preferring (disliking)

to be served directly after a customer in the same cluster has been visited. This preference (aversion) is stronger than its preference (aversion) for being served directly after a customer from another cluster has been visited, if the in-cluster correlation is higher (lower) than the correlation between clusters. Moreover, if a tour and the total expected number of customers that requires a visit along that tour remains unchanged, increasing the number of clusters also increases the chance that the salesman visits two successive customers on the tour that are not in the same cluster. As a result, a salesman is increasingly penalised (rewarded) for disregarding increased in-group favouritism.

In contrast to in-cluster correlation, for increasing heterogeneous correlations between clusters, no universal trend of declining expected lengths with increased correlation is visible. For example, in the case of $p = 0.9, k = 2$ of kroA100, expected lengths are even monotonically increasing for higher correlations, and in the case of $p = 0.5, k = 2$ of the d198 instance expected distances remain almost unchanged. But in similar spirit as the in-cluster case, there are trends visible for an increasing number of clusters when the correlation between clusters increases. This behaviour is in almost all cases (102 out of 108 or 94%) the opposite of the trend in changes in in-cluster correlation: As the number of clusters increases, the expected length decreases for positive correlations, and increases for negative correlations. The reasoning follows the same lines as the in-cluster scenario, but negated: A positive (negative) correlation between customers of different clusters has the effect of a customer preferring (disliking) a visit immediately after a customer of another cluster has been visited, over the visit after a customers who is in the same cluster. Hence, if a tour and the total expected number of customers that requires a visit along that tour remains unchanged, the salesman is more likely to leave a cluster more often as the total number of clusters grows. The salesman is thus rewarded (penalised) for taking increased intergroup favouritism into account.

Besides trends between correlation and cluster size, there is also a trend between correlation and probabilities. For the homogeneous probabilities among

the nonzero in-cluster correlation instances, lower probabilities are associated with higher relative deviations from the independence baseline. In other words, in-cluster correlation has a higher impact on the expected length when the probability is small. This is not surprising, given that the marginal effect of probability deviations on the PTSP expected length also increases with decreasing probability (Jaillet, 1985). Since in-cluster correlation can be regarded as a perturbation of the PTSP probability weight, low-probability CPTSP instances are more prone to small deviations in the in-cluster correlation than high-probability instances. The trend that correlation has a higher impact on the expected length for small probabilities is not true for every reported case of table 6.5 for correlations between clusters. However, with 96 out of 108 cases (or 88.9%) still exhibiting this trend it can still be maintained as a general rule of thumb.

6.5 A correlated probabilistic ant colony system

The observation that solutions that are good for the TSP are not necessarily good for the PTSP formed the logic behind some tailor-made PTSP heuristics (see, for example, Branke and Guntsch, 2004). The state-of-the-art for the PTSP includes stochastic adaptations of ant colony optimisation (Bianchi et al., 2002a,b; Gutjahr, 2004), particle swarm optimisation (Marinakis and Marinaki, 2010; Marinakis et al., 2015a), memetic algorithms (Liu, 2008a; Balaprakash et al., 2010), and extensions applying empirical estimation improvement heuristics (Birattari et al., 2008).

In this section, I propose a metaheuristic named the correlated probabilistic ant colony system (cpACS) to solve the CPTSP. cpACS is a straightforward extension of the ant colony system, ACS (Dorigo and Gambardella, 1997), and probabilistic ant colony system, pACS (Bianchi et al., 2002b), metaheuristics for the TSP and PTSP that are known to produce competitive results. The cpACS algorithm is directly benchmarked against state-of-the-art competitors for the TSP and PTSP: Concorde and pACS.

Algorithm 3 presents the pseudo-code for cpACS. First, the farthest-insertion construction heuristic is applied to the graph to initialise the best-so-far solution s_{BSF} and the associated CPTSP expected length $\mathbb{E}[L(s_{BSF})]$ given by (6.28). Each edge (i, j) of the graph is assigned to a heuristic information parameter η_{ij} and a pheromone trail parameter φ_{ij}. The heuristic information parameter is initialised with the inverse distance of each arc $\eta_{ij} = d_{ij}^{-1}$, whereas each pheromone trail parameter φ_{ij} is initialised with φ_0, given by $\varphi_0 = (n \cdot \mathbb{E}[L(s_{BSF})])^{-1}$.

Then, for a set of m ants, each ant constructs its own solution based on the heuristic information and the pheromone trails. If an ant a is at customer i, the ant chooses the next customer j from the set of unvisited customers J^a with probability π_0 according to $\arg\max_{j \in J^a} \eta_{ij}^{\gamma} \varphi_{ij}$. With probability $1-\pi_0$, the ant chooses the next

Algorithm 3 Pseudo-code of the cpACS algorithm

```
1: s_BSF ← Initialise best-so-far solution
2: Initialise parameters
3: while Stopping criterion is not met do
4:     s_BA ← Initialise best ant solution
5:     for each ant do
6:         s_a ← Construct new ant solution
7:         if E[L(s_a)] < E[L(s_BA)] then
8:             Update s_BA with s_a
9:         end if
10:    end for
11:    if E[L(s_BA)] < E[L(s_BSF)] then
12:        Update s_BSF with s_BA
13:    end if
14:    Update pheromone trails
15: end while
```

customer j randomly with a probability of

$$q_{ij}^{a} = \begin{cases} \frac{\eta_{ij}^{\gamma}\varphi_{ij}}{\sum_{k\in J}\eta_{ij}^{\gamma}\varphi_{ij}}, & \text{if } j \in J \\ 0, & \text{otherwise.} \end{cases}$$

After selecting the next cutomer, the pheromone trail φ_{ij} of the traversed edge (i, j) is updated according to $\varphi_{ij} \leftarrow (1-\theta)\varphi_{ij} + \theta\varphi_0$ for some parameter $\theta, 0 \leq \theta \leq 1$.

The ant repeats this procedure until it completes its tour. cpACS then checks the CPTSP expected length of the resulting solution s_a against the current best solution s_{BA} of every other ant, and updates the best ant solution accordingly. After each ant has completed a tour, the overall best ant solution s_{BA} is checked against the best-so-far solution s_{BSF}, which is also updated if necessary. Once the updating procedure has completed, the pheromone trails are updated by applying the rule $\varphi_{ij} \leftarrow (1-\lambda)\varphi_{ij} + \lambda\Delta\varphi_{ij}$, where $\Delta\varphi_{ij} = \mathbb{E}[L(s_{BSF})]$ for some pheromone decay parameter $\lambda, 0 \leq \lambda \leq 1$. Following the suggestions by Bianchi (2006) after extensive parameter tuning experiments of pACS, I set $m = 10, \pi_0 = 0.95, \theta = \lambda = 0.1, \gamma = 3$. I set the stopping criterion to 10,000 iterations.

Essentially, cpACS uses the correlated version of the expected length of the a priori tour (6.28) in the objective function that updates the pheromone trails, whereas pACS uses the PTSP expected length (3.3) and ACS uses the deterministic a priori tour length (6.1). For an elaborate discussion of the algorithm and the implications of integrating stochastic elements, I refer the reader to Bianchi et al. (2002b).

Table 6.6 reports the results of the Concorde, pACS and cpACS algorithms for a case study on a number of homogeneous variations of the d198 problem instance. The reported figures are the best solutions found out of 10 independent runs for Concorde, pACS and cpACS. It can be observed that cpACS outperforms pACS and Concorde for the entire negative correlation range of $p = 0.1$, with improvements of up to 2.09% versus Concorde and 1.09% versus pACS. The cpACS

Instance	Corr.	Concorde			pACS			cpACS		
		$p = 0.1$	$p = 0.5$	$p = 0.9$	$p = 0.1$	$p = 0.5$	$p = 0.9$	$p = 0.1$	$p = 0.5$	$p = 0.9$
d198	-0.9	8082.826	**13057.333**	**15253.800**	8082.295	**13057.333**	**15253.800**	**8061.684**	**13057.333**	**15253.800**
	-0.8	8082.693	**13018.050**	**15253.798**	8082.163	**13018.050**	**15253.798**	**8058.082**	**13018.050**	**15253.798**
	-0.7	8080.182	**12984.833**	**15253.748**	**8079.679**	**12984.833**	**15253.748**	**8079.679**	**12984.833**	**15253.748**
	-0.6	8069.445	**12951.243**	**15253.495**	8069.046	**12951.243**	**15253.495**	**8024.680**	**12951.243**	**15253.495**
	-0.5	8044.765	**12915.589**	**15252.857**	8009.974	**12915.589**	**15252.857**	**7997.287**	**12915.589**	**15252.857**
	-0.4	8002.516	**12876.704**	**15251.705**	8002.516	**12876.704**	**15251.705**	**7873.897**	**12876.704**	**15251.705**
	-0.3	7940.448	**12832.636**	**15249.960**	7860.603	**12832.636**	**15249.960**	**7774.660**	**12832.636**	**15249.960**
	-0.2	7856.582	**12781.197**	**15247.547**	7856.582	**12781.197**	**15247.547**	**7827.644**	**12781.197**	**15247.547**
	-0.1	7748.423	**12720.317**	**15244.334**	**7731.397**	**12720.317**	**15244.334**	7737.602	**12720.317**	**15244.334**
	Ind. 0.0	**7612.421**	**12648.247**	**15240.088**	**7612.421**	**12648.247**	**15240.088**	**7612.421**	**12648.247**	**15240.088**
	0.1	7443.540	**12563.591**	**15234.408**	7421.506	**12563.591**	**15234.408**	**7339.678**	**12563.591**	**15234.408**
	0.2	**7234.764**	**12464.890**	**15226.628**	**7234.764**	**12464.890**	**15226.628**	**7234.764**	**12464.890**	**15226.628**
	0.3	**6976.326**	**12349.609**	**15215.731**	**6976.326**	**12349.609**	**15215.731**	**6976.326**	**12349.609**	**15215.731**
	0.4	**6654.197**	**12212.133**	**15200.301**	**6654.197**	**12212.133**	**15200.301**	**6654.197**	**12212.133**	**15200.301**
	0.5	**6246.538**	**12039.897**	**15178.434**	**6246.538**	**12039.897**	**15178.434**	**6246.538**	**12039.897**	**15178.434**
	0.6	**5714.732**	**11806.663**	**15147.084**	**5714.732**	**11806.663**	**15147.084**	**5714.732**	**11806.663**	**15147.084**
	0.7	**4983.237**	**11461.700**	**15099.148**	**4983.237**	**11461.700**	**15099.148**	**4983.237**	**11461.700**	**15099.148**
	0.8	**3924.587**	**10904.562**	**15013.670**	**3924.587**	**10904.562**	**15013.670**	**3924.587**	**10904.562**	**15013.670**
	0.9	**2541.047**	**9830.690**	**14822.274**	**2541.047**	**9830.690**	**14822.274**	**2541.047**	**9830.690**	**14822.274**

Table 6.6: Expected lengths for Concorde, pACS and cpACS (best out of 10 independent runs) on homogeneous (equiprobable and equicorrelation) instances of d198.

algorithm finds it hard to improve over results produced by Concorde or pACS for any other correlation range, yet underperforms the other algorithms in only one case. This is in line with the observation by Bianchi and Gambardella (2007) that Concorde performs relatively well in moderately stochastic settings. Only in strongly stochastic settings, tailor-made heuristics such as pACS manage to outperform Concorde. These notions are in line with the reported results for cpACS, which proofs to be a good metaheuristic for all settings, but especially when the stochasticity is high.

6.6 Conclusion

This paper introduces the CPTSP, a new problem that accounts for dependencies among the stochastic customer presence in the PTSP. Unlike the PTSP, it does not assume that customers are independent. I considered two methods to account for stochastic customer dependencies, using a Bahadur-Lazarsfeld expansion (Bahadur, 1961) and using D-Vine PCCs (Panagiotelis et al., 2012) as a model for the underlying pmf. I demonstrated that these models exhibit a number of highly

desirable properties from both a modeller's and practitioner's perspective. The resulting Bahadur-CPTSP and Copula-CPTSP only require the specification of marginal probabilities and pairwise correlations, without overintensifying the computational burden. Using a C-shaped, star-shaped and a gear-shaped example, I illustrated that a good TSP tour nor a good PTSP tour necessarily coincides with a good CPTSP tour. Computational experiments suggest that the expected length is a negative function of correlation. The overall effect of correlation on the expected length of the tour diminishes with increasing probability, and a smaller number of customers.

The main contribution of this paper is a method to model dependencies in the customer presence of stochastic routing problems. It uses the PTSP as a mere example, however, it may readily be extended to other types of routing problems. Incorporating dependencies allows for the reflection of customer behaviour, e.g. cases in which customer presence is related to customer segments (example 1) or working hours (example 2); preferences or penalties for serving customers from the same cluster or from different clusters, e.g., cases in which tax needs to be paid when serving customers from another cluster (example 3), or the product space, e.g., cases in which it is necessary to account for the relatedness of products between different suppliers (example 4).

The computational results for a wide range of homogeneous and heterogeneous CPTSP instances support the belief that the expected length of an a priori tour varies a lot with changing problem characteristics. Therefore, new tailor-made approaches are needed in order to solve CPTSP problems more effectively. The cpACS metaheuristic, an extension of the ant colony system, explores the benefits of pursuing such an approach under homogeneous conditions. Our results demonstrate that cpACS outperforms closely related state-of-the-art approaches for the TSP and PTSP in some instances, and rarely underperforms these approaches.

Future research focuses on the integration of correlated stochastic customer

presences in other domains, such as the vehicle routing problem with stochastic customers (Bertsimas, 1988). Another avenue that also requires further exploration includes a step-wise procedure with dynamic updates, such that the probability of visiting the next customer along the route can be conditioned on the presence or absence of the previous customer. Finally, the interaction between the stochastic problem structure and the search for a good solution calls for the investigation of new, tailor-made, algorithms for the CPTSP.

7 | Conclusion

This thesis investigates the challenges of problem-specific characteristics for the PTSP. It consists of three separate studies. The first study compares solution methods for the PTSP. It analyses the main methods for the most important solution classes, namely, exact methods, construction heuristics, classical improvement heuristics, and metaheuristics. I find that construction heuristics received only little attention, though their performance is found to affect the quality of the final solution. Furthermore, I find that the configuration of the problem, and most notably the clustering of customers, plays an important role in the relative performance of solution methods. The second study exploits this notion, by devising a hyper-heuristic framework, named GENS-H, that is capable of selecting and combining the most desirable construction components to specifically target the characteristic features of the instance to be solved. By doing so, GENS-H exposes new construction heuristics on the fly. This leads to a powerful framework, capable of beating its low-level heuristics in nearly all circumstances. The third and final study in this thesis formalises spatial clustering and other possible expressions of customer relatedness as an integral and explicit characteristic of the PTSP. This is accomplished by introducing dependence between the stochastic requirements to visit customers, leading to a new stochastic combinatorial optimisation problem, the CPTSP. Empirical results demonstrate that a good solution for the CPTSP does not necessarily coincide with a good solution for closely problems such as the TSP and the PTSP. The commonality across these studies is marked by the impact that characteristics may have on the performance of solution methods for the TSPSC.

7.1 Contributions

This thesis contributes to the existing literature with empirical studies on problem characteristics for the PTSP. More specifically, it focuses on the impact of clustering, the role of the construction phase, and the introduction of correlation. These three traits remain largely unexplored in the literature. Firstly, although empirical analyses over different probability ranges and instances of increasing size are abundant (e.g., Bianchi, 2006; Balaprakash et al., 2010; Marinakis and Marinaki, 2010), only few studies focus on the combination of these characteristics with clustering groups of customers. A notable exception to this rule is provided by Birattari et al. (2008), who test the state-of-the-art 2.5-opt-EEais improvement algorithm on a set of self-generated problem instances. Secondly, although the construction phase received ample attention in the decade immediately after the PTSP was proposed, its role faded into the background in more recent years. The combination of construction heuristics with recent trends, including state-of-the-art improvement heuristics and hyperheuristics, did not deserve any attention besides the analyses of the GA and VNS metaheuristics by Liu (2010) and Weiler et al. (2015). Finally, the impact of dependencies among customers remained unexplored altogether in the PTSP literature.

Chapter 4 discusses and reviews solution methods for the PTSP, along with a computational and empirical comparison. It contibutes to the existing literature and heuristical comparison surveys (e.g., Gendreau et al., 1996b; Cordeau et al., 2007; Bianchi et al., 2009; Pillac et al., 2013; Henchiri et al., 2014; Weiler et al., 2015) in the stochastic routing domain by providing a comprehensive overview of the most important probabilistically adapted solution methods for the PTSP. In addition, chapter 4 performs an empirical analysis of construction heuristics for the PTSP. This empirical study contributes to the knowledge on performance of solution methods in different settings. In particular, the observation that the degree of clustering of customers across the plane has an impact on the performance of construction heuristics, provides food for thought for researchers

who aim to develop new solution methods. Adjusting the design of solution methods to cope with the problem-specific nature of a particular PTSP instances may benefit the performance. In addition, a follow-up analysis of construction heuristics in combination with the 2.5-opt-EEais state-of-the-art improvement heuristic contributes to the debate on the impact of a good initial solution on the quality of the final solution. In line with the observation by Liu (2010) for GA, I find supporting evidence that the initial solution produced by construction heuristics affects the quality of the final solution by 7.79% on average.

Chapter 5 extends the analysis of construction heuristics to the hyperheuristics domain. It contributes to the existing literature on hyperheuristics (see Burke et al., 2013) by proposing a construction framework, GENS-H, that is capable of automatically generating new (supersavings-based) construction heuristics. In doing so, it is, to the best of my knowledge, the first reported instance of the application of a hyperheuristic construction framework in the field of stochastic routing. The proposed implementation demonstrates the potential of generalising an entire class of solution methods in a way such that underperforming any single of its underlying individual solution methods becomes rare. By doing so, the GENS-H hyperheuristic construction framework contributes to the ongoing quest of finding improved solution methods (e.g., Laporte et al., 1994; Bianchi et al., 2002b; Marinakis and Marinaki, 2010), by offering a powerful tool to cope with the complex problem-specific nature of PTSP instances.

Chapter 6 introduces correlation between the stochastic presence of customers in the PTSP. To the best of my knowledge, the impact of correlation between stochastic customers has never been empirically investigated before in the routing domain. Chapter 6 contributes to the existing literature on new routing models for stochastic customers (e.g., Jaillet, 1985; Beraldi et al., 2005; Campbell and Thomas, 2008b; Voccia et al., 2013) by proposing a new theoretical framework to model dependencies between customers, thereby introducing a new stochastic routing problem, the CPTSP. The CPTSP expands the window of practical applications of the

PTSP to implementations where customer relations play a role. For example, one could think of PTSP settings where the uncertainty surrounding a product drop-off at one site depends on the need to pick up a product at another site. Alternatively, the correlation parameter may be regarded as a means to make spatial relatedness – e.g., clustering – between stochastic customers explicit. An empirical analysis of the CPTSP demonstrates that good solutions for the CPTSP do not necessarily coincide with good solutions for the PTSP, nor with good solutions for the TSP. This observation contributes empirical evidence to the notion that explicitly accounting for (or neglecting) problem characteristics adds to the (in)accuracy of solution methods. Finally, chapter 6 attempts to spark a debate on problem-specific solution methods by contributing a new metaheuristic, cpACS, to the class of ant-based solution approaches for stochastic combinatorial optimisation problems.

7.2 Limitations

A number of shortcomings to the analyses of this thesis can be identified. Overall, this thesis is limited in scope: The empirical studies performed in this thesis merely consider the PTSP, whereas many concepts extend to other stochastic routing problems. Most notably, an investigation of the concepts proposed in this thesis for the VRPSC may lead to new insights. Moreover, with the dawn of e-commerce, routing problems do not only become increasingly stochastic but also increasingly dynamic in nature. The studies in this thesis do not address dynamic problem features. In addition, a number of study-specific limitations can be identified.

The scope of chapter 4 is limited to a discussion of stochastic adaptations of solution methods for the PTSP. It largely omits the treatment of deterministic methods for the PTSP, which in some contexts are nevertheless found to perform reasonably well when the level of stochasticity is low (Bianchi, 2006; Bianchi and Gambardella, 2007; Balaprakash et al., 2010). In addition, chapter 4 reviews empirically tested methods for the PTSP, but does not include a comparison of

these methods against solution methods designed for closely related vehicle routing problems, nor untested methods.

The empirical comparison on construction heuristics investigates the impact of an only limited number of PTSP characteristics. However, the impact of other characteristics, e.g., heterogeneous customer probabilities, could lead to even more insights. As the results from the later CPTSP study illustrate, clustering may have a different effect on heterogeneous instances than on homogeneous instances. The consecutive empirical evaluation is conducted on a wide range of instances from TSPLIB. These instances are widely regarded the standard in the PTSP literature (e.g., Bianchi et al., 2002a; Balaprakash et al., 2010; Marinakis and Marinaki, 2010), which allows one to directly compare the results and to reproduce the reported results. However, the division of the TSPLIB into clustered and evenly divided instances is based on expert judgement, and did not involve the inference of any test statistics or an objective clustering criterion. The use of instance generators such as DIMACS (Johnson et al., 2001), could have provided more objective clustered test cases. Finally, the impact of characteristics on heuristical performance is only tested for construction heuristics in conjunction with the 2.5-opt-EEais improvement heuristic. Although the 2.5-opt-EEais improvement heuristic is considered to be the state-of-the-art, and frequently applied in combination with other top performing (meta)heuristic procedures in the literature (e.g., Balaprakash et al., 2007, 2009b, 2010), a more extensive empirical analysis of construction heuristics in combination with other metaheuristics would have been useful. Such an extensive analysis would provide consistency and robustness support to the results. The future design of metaheuristics may also benefit from such an analysis.

Chapter 5 claims the first account of applying of a hyperheuristic-driven construction framework, GENS-H, in the stochastic routing domain. However, the study could have benefited from benchmark tests of GENS-H against related approaches from other domains, such as tuning algorithms (Bartz-Beielstein and Preuss, 2007; Hutter et al., 2009; Balaprakash et al., 2010). In addition, the

scope of the empirical analysis of high level heuristics in GENS-H is limited to tuning the set of parameters for a single instance during execution. Another, less computationally intensive, class of offline parameter tuning approaches attempts to find the best parameter configuration for an entire set of problem instances. Unfortunately, the GENS-H is not designed to pursue such an approach. As a result, the practical usefulness of tuning parameters on a per-instance basis with GENS-H is relatively low. That is, GENS-H relies on the repeated application of supersavings-based heuristics, each bearing a computational complexity of $O(n^2)$. This implies that the computational effort involved in finding new instance-specific parameter sets is high. This limitation could be overcome by performing a more thorough analysis of the optimal parameters found for instances that bear similar traits, thereby generalising the found parameter configurations to an entire set of instances rather than focusing on a per-instance basis.

Chapter 6 introduces correlation between the stochastic customer visits in the PTSP, and lists a number of practical examples to justify how real-world situations could benefit from such extension. Unfortunately, the scope of my empirical analyses in the CPTSP study is yet limited to modified TSPLIB instances. With applications from mere theoretical situations rather than real-world cases, the practical usefulness and behaviour of the CPTSP remains indefinite. Furthermore, the Bahadur-CPTSP is restricted in its application to only a small range of correlation values (Declerck et al., 1998). The practical usefulness of the Bahadur-CPTSP is therefore fairly limited. In addition, the empirical study of the Copula-CPTSP is limited to numerical evaluations of the Gaussian copula. A benchmark study of the Gaussian Copula implementation with other copula implementation could lead to new insights. Specifically, considering special types of Archimedean copulas may improve the computational attractiveness of the Copula-CPTSP. Because the computational complexity of the currently implemented Gaussian Copula-CPTSP objective function is proportional to $O(n^4)$, a single evaluation of the expected length belongs to the class of polynomial functions. As most state-of-the-art solution methods rely on the repeated evaluation of the

objective function, the current complexity of the Copula-CPTSP imposes a large burden on the computational effort of such methods. Compared to the $O(n^2)$ computational time for the PTSP, it is therefore questionable whether the increase in computation time weighs up to the accuracy gained by recognising dependencies among customers.

7.3 Research opportunities

There are plenty directions for further research. First of all, the results of the empirical analyses about clustering in chapter 4, the construction of solutions with GENS-H in chapter 5, and the role of correlation in chapter 6 could readily be extended to the VRPSC and related stochastic routing problems.

Following the comprehensive review of solution methods in chapter 4, I find that a number of exact solution methods in the stochastic combinatorial optimisation domain (Balaprakash, 2005) are left unexplored in the PTSP. A methodological and empirical analysis of these methods versus the existing state-of-the-art procedures can be beneficial. In addition, the literature reporting empirical results on the performance of solution methods for the PTSP may benefit from a more thorough analysis of statistical moments beyond the mean. That is, the benchmark studies reported in chapter 4 approach the PTSP almost exclusively from an a priori perspective, and evaluate the performance of solution methods by merely considering differences among expected lengths of a priori tours. In doing so, the a posteriori results are largely neglected. The evaluation of other statistical moments beyond the mean (i.e., expected) length of a tour, such as the median, variance, skewness, and kurtosis, broadens our understanding of why solution methods perform better under some circumstances than others.

The application domain of GENS-H in chapter 5 could be generalised to sets of problem instances, rather than individual instances. By finding appropriate parameter configurations for entire problem sets, guidelines can be developed for

the choice of the parameters pertaining to specific problem *characteristics*, rather than parameter choices pertaining to specific problem *instances*. These guidelines may, in turn, provide us with insights beneficial to the design of new solution methods. More specifically, the choice for a specific parameter set is essentially a recommendation of applying a set of construction components to an instance that bears a specific set of problem characteristics. In order to categorise problem instances into classes of characteristics, however, one would first need to devise new statistical methods to determine problem characteristics such as clustering.

Analogous to the follow-up study for the PTSP by Bertsimas and Howell (1993), the analysis of the CPTSP in chapter 6 may benefit from the derivation of boundaries and other theoretical results, along with an extension to the VRP domain. Furthermore, a transformation of the problem from a mere stochastic formulation to a stochastic-dynamic formulation, i.e., by investigating the need to visit a customer conditional on the requirement to visit related customers at every time step, may be worth a further study. Finally, the notion that copulas are frequently encountered in risk management promotes the idea that a risk-based assessment of stochastic tours could be a promising new research avenue. Variations of the Copula-CPTSP in which a driver faces financial incentives and constraints creates a window of opportunity for future research at the brink of routing and finance.

A Performance results of construction heuristics for the PTSP

Tables A.1 and A.2 present the full set of results used for the empirical investigations of construction heuristics in chapter 4. Table A.1 presents the 'plain' results without improvement; table A.2 presents the results of the same set of construction heuristics after applying 2.5-opt-EEais. The results of Probabilistic Savings, Globalsave, and Iterated Nearest Neighbour for pr1002 and fl1400 have been omitted due to their excessively long computation times.

Instance	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2
eil51	0.1	182.559	177.353	139.172	142.487	142.425	142.425	156.680	159.758
	0.2	330.749	340.007	211.095	211.714	212.115	212.115	236.201	240.366
	0.3	525.586	505.318	261.732	265.069	264.615	264.615	284.481	292.001
	0.4	691.715	636.691	309.145	306.436	306.049	306.049	323.153	323.034
	0.5	844.436	758.627	356.930	338.460	338.460	338.460	356.021	361.581
	0.6	1027.464	921.844	382.693	366.160	366.093	366.093	401.933	391.587
	0.7	1138.591	1036.367	406.783	390.985	390.985	390.985	444.414	439.434
	0.8	1356.007	1206.490	440.151	413.373	413.373	413.373	468.256	461.420
	0.9	1439.761	1303.818	438.785	428.105	433.932	433.932	490.163	471.163
kroA100	0.1	18179.13	17654.715	9528.582	9627.948	9745.240	9745.240	10276.261	10792.985
	0.2	35379.318	33672.342	12637.342	12644.613	12796.301	12796.301	13513.931	13513.931
	0.3	51629.821	49380.255	14954.169	14845.509	14845.509	14845.509	15801.046	15801.046
	0.4	70650.334	63895.493	17411.109	16462.330	16462.330	16462.330	17737.082	17737.082
	0.5	83896.738	80300.693	18794.162	17832.623	17832.623	17832.623	19265.121	19265.121
	0.6	103446.373	95531.769	21292.168	19043.418	19043.418	19043.418	21206.457	21206.457
	0.7	117648.388	109932.933	22193.953	20144.444	20144.444	20144.444	22400.153	22400.153
	0.8	134417.78	126430.341	22257.042	21166.812	21166.812	21166.812	24812.554	24812.554
	0.9	168027.705	141500.199	24013.150	22130.456	22130.456	22130.456	25967.871	25967.871

Table A.1: Expected a priori tour lengths for 17 construction heuristics across 9 homogeneous TSPLIB instances.

Instance	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2
eil101	0.1	351.547	346.513	212.776	213.960	218.145	218.145	228.001	230.640
	0.2	692.8	682.476	321.537	307.416	311.322	311.322	350.683	351.673
	0.3	1024.774	1002.029	410.473	377.209	386.030	386.030	436.595	438.736
	0.4	1358.919	1325.131	474.023	436.017	452.221	452.221	512.464	518.812
	0.5	1744.644	1602.223	529.011	487.378	511.268	511.268	544.721	547.189
	0.6	2097.994	1913.596	583.947	551.120	551.120	551.120	610.107	610.107
	0.7	2314.517	2206.835	620.897	584.749	584.749	584.749	691.335	688.277
	0.8	2826.519	2539.583	657.194	615.952	615.952	615.952	731.835	733.099
	0.9	3229.536	2814.592	686.706	645.541	645.541	645.541	766.675	762.587
ch150	0.1	5835.957	5553.001	2680.022	2710.319	2704.022	2704.022	3071.030	3226.596
	0.2	11032.064	10849.580	3806.195	3751.202	3813.041	3813.041	4105.301	4129.895
	0.3	16390.007	15839.102	4558.313	4380.132	4609.581	4609.581	5012.077	5012.077
	0.4	21163.115	20951.509	5297.648	5033.662	5237.215	5237.215	6153.156	6153.577
	0.5	26386.048	24974.664	5636.690	5462.395	5616.729	5616.729	6691.713	6691.713
	0.6	32409.306	31031.155	6111.205	5860.315	5939.185	5939.185	7080.992	7080.992
	0.7	35849.533	35148.040	6742.862	6226.450	6226.450	6226.450	6584.312	6584.312
	0.8	43475.701	40047.716	6978.084	6490.050	6490.050	6490.050	6703.261	6703.261
	0.9	49488.803	44921.540	7206.121	6736.356	6736.356	6736.356	7528.475	7528.475
d198	0.1	22869.012	21989.835	10065.023	10394.979	10057.570	10057.570	10757.482	10768.446
	0.2	40473.148	39848.143	11706.783	12330.028	11738.078	11738.078	12626.683	12755.522
	0.3	60278.348	58004.130	12883.300	13565.815	12831.027	12831.027	14117.399	14341.831
	0.4	80317.263	73379.461	13935.926	14379.294	13718.915	13718.915	15125.013	15308.688
	0.5	98598.551	92123.832	14552.569	15060.098	14472.661	14472.661	15853.639	15868.319
	0.6	112155.694	109460.271	15171.756	15653.932	15130.643	15130.643	16462.956	16481.124
	0.7	134790.095	126058.850	15680.482	16185.778	15716.365	15716.365	16989.189	17012.079
	0.8	161583.562	141219.755	16362.473	16670.616	16185.925	16185.925	17453.162	17482.316
	0.9	163812.425	158848.145	16731.112	17117.949	16714.894	16714.894	17867.447	17904.404
pr439	0.1	195805.38	191764.314	54532.012	53804.284	55132.293	55132.293	58228.585	58683.509
	0.2	381982.064	378116.348	76017.105	71147.481	73067.564	73067.564	75631.478	75631.478
	0.3	576232.751	561410.541	89936.488	83772.115	83772.115	83772.115	88092.983	88112.153
	0.4	755995.018	742435.170	99487.345	91894.780	91894.780	91894.780	103449.391	103449.391
	0.5	957261.301	916362.378	108144.499	98508.705	98508.705	98508.705	109206.887	109206.887
	0.6	1118322.799	1107622.707	114673.571	103263.126	104135.868	104135.868	113862.263	114082.279
	0.7	1352484.576	1273143.587	123601.590	109061.951	109061.951	109061.951	120940.179	120940.179
	0.8	1544288.751	1468080.373	126346.079	113456.133	113456.133	113456.133	124735.723	124735.723
	0.9	1783365.613	1626676.834	131019.853	117425.730	117425.730	117425.730	128154.186	128154.186
rat783	0.1	18358.792	18074.804	3816.415	3718.781	3968.242	3968.242	4728.894	4729.832
	0.2	36151.195	35669.694	5528.472	5223.102	5452.211	5452.211	6001.641	6000.706
	0.3	54169.787	53213.593	6661.053	6265.360	6396.803	6396.803	7098.099	7100.631
	0.4	71039.061	70609.363	7577.883	7102.202	7114.061	7114.061	7815.250	7815.250
	0.5	90678.098	87757.549	8195.208	7702.974	7702.974	7702.974	8655.991	8655.991
	0.6	108435.017	105669.051	8933.885	8208.539	8208.539	8208.539	9194.943	9194.943
	0.7	124238.514	122111.181	9387.410	8655.529	8655.529	8655.529	9725.051	9731.785
	0.8	141659.186	135115.192	9857.861	9058.668	9058.668	9058.668	10193.849	10193.849
	0.9	162146.859	155532.878	10324.688	9427.157	9427.157	9427.157	10728.190	10728.190
pr1002	0.1	655913.765	651375.840	-	120843.600	125974.620	-	143858.054	146376.987
	0.2	1306285.879	1282882.982	-	164171.319	165172.686	-	170184.522	170445.234
	0.3	1949924.689	1915827.950	-	190158.280	190223.859	-	205576.734	205576.734
	0.4	2582229.994	2543868.522	-	209677.672	209677.672	-	224910.092	224910.092
	0.5	3200442.658	3143775.026	-	225951.567	225951.567	-	246048.173	246048.173
	0.6	3883417.147	3765462.197	-	240194.947	240194.947	-	263589.368	263603.407
	0.7	4524646.913	4426210.586	-	253030.760	253030.760	-	291825.305	291825.305
	0.8	5158872.886	4993110.063	-	264828.492	264828.492	-	299312.681	299312.681
	0.9	5681798.451	5589504.140	-	275824.999	275824.999	-	306782.646	306782.646

Table A.1: Expected a priori tour lengths for 17 construction heuristics across 9 homogeneous TSPLIB instances (*continued*).

Instance	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2
fl1400	0.1	170266.524	168643.167	-	10265.069	10837.519	-	12620.371	12626.307
	0.2	336844.285	333348.944	-	13157.489	13764.967	-	15353.861	15355.840
	0.3	512947.299	495923.118	-	15601.245	15601.245	-	17313.759	17315.760
	0.4	667728.077	666556.345	-	17003.417	17003.417	-	18903.366	18903.366
	0.5	836163.632	832429.744	-	18137.890	18137.890	-	21560.413	21560.413
	0.6	1013418.792	989536.410	-	19094.806	19094.806	-	22859.482	22859.482
	0.7	1192591.722	1156092.675	-	19928.043	19928.043	-	24017.562	23999.908
	0.8	1364514.602	1322012.222	-	20672.118	20672.118	-	24882.996	24884.430
	0.9	1558211.78	1472325.085	-	21349.791	21349.791	-	26120.243	26121.952

Table A.1: Expected a priori tour lengths for 17 construction heuristics across 9 homogeneous TSPLIB instances (*continued*).

Instance	Probability	ANNA0	ANNA1	Iterated NN	Random NN (type I)	Random NN (type II)	Radial sort	Spacefilling curves	Farthest insertion	Nearest insertion
eil51	0.1	154.125	157.873	149.334	155.511	157.620	137.454	138.608	140.832	139.933
	0.2	238.462	249.214	230.825	260.781	273.872	201.101	202.071	204.997	207.034
	0.3	288.327	311.445	291.336	329.775	314.437	256.087	250.672	253.523	260.097
	0.4	335.399	355.332	338.343	370.028	406.963	310.731	291.644	295.615	305.640
	0.5	391.717	390.031	375.78	456.917	400.876	366.246	327.052	333.380	345.373
	0.6	421.436	419.175	406.537	472.903	435.252	422.672	358.324	367.668	380.589
	0.7	447.758	441.391	432.312	528.033	468.930	479.869	386.569	399.061	412.403
	0.8	471.747	468.256	455.761	538.469	512.553	537.757	412.591	428.082	441.689
	0.9	494.032	490.163	476.285	506.087	614.749	596.394	436.943	455.239	469.076
kroA100	0.1	10233.025	10741.782	10598.41	10913.298	11456.845	10229.517	10418.987	10060.993	10458.896
	0.2	13975.825	14719.116	14196.1	14934.867	15477.080	14842.759	14351.792	13334.842	14454.099
	0.3	15711.540	17399.976	16568.85	18423.912	19834.248	19266.974	17179.860	15663.529	17071.017
	0.4	18571.973	19432.18	18434.386	19526.709	19258.187	23752.760	19517.594	17539.454	18976.193
	0.5	20078.694	21065.83	20024.254	23614.277	27773.487	28292.074	21580.355	19114.857	20488.503
	0.6	21878.404	21938.258	21443.933	23917.043	22319.284	32842.978	23457.550	20480.259	21764.941
	0.7	23106.780	22980.707	22747.669	24175.148	26298.265	37365.055	25194.446	21697.565	22887.961
	0.8	24449.206	23624.831	23965.391	24359.676	31971.832	41829.716	26819.856	22808.934	23906.011
	0.9	25721.397	25330.205	25114.254	31666.850	29375.900	46223.031	28354.814	23842.050	24850.306
eil101	0.1	234.407	258.033	232.169	249.623	249.862	206.200	211.349	214.192	231.727
	0.2	356.442	376.431	344.727	427.682	404.575	311.494	305.749	311.990	337.378
	0.3	452.916	465.081	431.775	554.319	502.828	417.745	380.260	390.012	412.189
	0.4	525.425	535.838	501.522	545.631	557.342	526.936	444.383	454.455	471.723
	0.5	598.482	595.086	553.825	681.960	675.345	638.082	502.307	508.756	522.635
	0.6	650.974	646.24	598.824	657.930	685.243	750.248	555.969	555.758	567.910
	0.7	696.240	690.065	639.232	758.261	703.506	862.731	606.300	597.652	608.975
	0.8	738.476	711.685	676.344	706.783	839.494	975.026	653.764	636.005	646.572
	0.9	776.169	768.804	711.061	895.221	892.292	1086.818	698.611	671.885	681.141
ch150	0.1	3389.266	3424.072	3028.266	3474.907	3808.744	2688.750	2656.478	2679.955	2815.212
	0.2	4765.425	4806.256	4150.681	4617.860	4545.271	4161.404	3699.410	3779.987	3956.341
	0.3	5463.781	5659.72	4766.371	6376.431	6151.389	5726.722	4530.533	4617.971	4832.115
	0.4	5391.963	6239.145	5236.069	6271.491	6781.927	7364.747	5245.115	5275.014	5530.293
	0.5	6617.096	6664.041	5629.466	6761.854	7013.407	9064.718	5886.212	5806.909	6101.334
	0.6	7133.528	7133.528	5974.717	7278.246	8333.926	10820.908	6476.478	6250.980	6580.888
	0.7	7044.463	7390.992	6285.501	7992.038	7171.896	12629.714	7029.982	6632.147	6994.449
	0.8	7339.915	7669.733	6569.484	7387.651	8923.498	14489.549	7556.558	6967.194	7359.774
	0.9	7921.107	7921.107	6831.522	7903.100	9731.727	16401.764	8063.459	7267.723	7688.930

Table A.1: Expected a priori tour lengths for 17 construction heuristics across 9 homogeneous TSPLIB instances (*continued*).

Instance	Probability	ANNA0	ANNA1	Iterated NN	Random NN (type I)	Random NN (type II)	Radial sort	Spacefilling curves	Farthest insertion	Nearest insertion
d198	0.1	10768.023	10531.209	10655.938	11134.376	11446.576	11724.831	12147.770	10076.655	10573.610
	0.2	12775.864	12586.091	12553.458	12994.962	13199.273	15776.005	14539.119	11622.324	12218.671
	0.3	14304.019	13968.576	13720.592	14805.535	18523.877	19691.883	16141.287	12694.466	13368.513
	0.4	15249.331	14858.684	14632.639	15187.593	16876.564	23647.237	17433.974	13577.286	14315.248
	0.5	16008.525	15853.639	15393.449	18993.266	20521.051	27625.475	18552.041	14336.110	15119.024
	0.6	16649.101	16462.956	16035.201	17992.425	18765.760	31591.427	19563.434	15001.823	15814.348
	0.7	17207.140	16824.397	16588.014	18608.311	19887.532	35507.302	20506.856	15593.964	16425.284
	0.8	17703.257	17453.162	17036.511	18827.241	20510.187	39335.756	21405.227	16126.301	16969.299
	0.9	18149.891	17867.447	17427.66	19263.363	20414.757	43042.325	22272.120	16608.870	17459.181
pr439	0.1	58093.358	65611.698	59141.118	66637.899	77095.137	61861.187	57301.422	52693.028	60514.333
	0.2	83363.198	86420.227	77426.046	87852.656	91884.615	109309.205	79405.228	72099.855	80116.829
	0.3	96787.769	97595.827	89060.674	104893.330	108852.092	157763.206	94867.568	84919.846	92960.842
	0.4	105506.835	105506.835	97556.122	119943.069	130903.566	206947.142	107296.938	94274.823	102357.551
	0.5	111639.437	111620.04	104205.191	135215.376	136008.888	256800.828	117956.130	101542.044	109703.154
	0.6	116685.624	116638.406	109653.747	123148.394	148355.804	307307.474	127436.345	107444.467	115700.422
	0.7	121013.025	120940.179	114259.055	154099.006	159699.580	358439.265	136076.193	112402.511	120761.623
	0.8	124832.244	124735.723	118239.932	167461.191	158638.993	410145.776	144086.867	116678.566	125150.492
	0.9	128272.718	128154.186	121743.735	152702.924	171019.429	462374.572	151602.295	120445.280	129042.630
rat783	0.1	4600.470	4977.066	4741.52	5112.273	5786.793	5968.344	3821.639	4497.006	4164.905
	0.2	6166.911	6458.972	6357.057	6521.896	7828.345	11486.291	5419.278	6299.106	5676.653
	0.3	7028.486	7474.582	7349.278	8115.669	8508.943	17098.422	6678.971	7422.778	6777.862
	0.4	7671.196	8122.864	8095.574	8793.867	9947.810	22728.661	7758.512	8252.069	7651.790
	0.5	8699.381	8790.336	8669.388	9290.300	10722.825	28351.370	8717.843	8919.424	8379.717
	0.6	9351.310	9350.058	9159.041	10107.549	9707.131	33955.302	9591.228	9484.193	9006.182
	0.7	9747.868	9419.288	9593.865	11048.013	11531.197	39534.455	10400.844	9977.832	9558.108
	0.8	10316.682	10180.81	9988.352	11298.103	12125.352	45084.456	11161.090	10419.105	10053.050
	0.9	10694.647	10667.793	10351.214	11754.293	12194.172	50599.513	11881.178	10820.103	10503.131
pr1002	0.1	138165.394	147559.017	-	162872.774	160450.637	184699.957	120343.607	129156.477	123921.463
	0.2	187855.672	194226.784	-	212987.220	212818.326	347907.211	167707.872	176180.679	165004.630
	0.3	216510.583	227067.754	-	253971.311	249963.185	513245.297	204180.304	207345.236	194285.286
	0.4	239433.095	241911.947	-	260240.975	267282.661	679970.414	235044.298	231556.391	217696.475
	0.5	254357.316	259820.887	-	289171.821	315299.564	847380.780	262276.929	251762.648	237369.556
	0.6	271897.820	275983.046	-	304059.846	326502.247	1014834.338	286920.614	269255.617	254400.562
	0.7	289785.867	286219.97	-	343869.244	350102.106	1181840.804	309627.061	284742.852	269460.061
	0.8	296518.020	308745.829	-	342069.297	342560.827	1348096.824	330842.151	298673.590	282995.699
	0.9	312174.857	314650.037	-	349636.052	366489.732	1513473.997	350887.424	311357.593	295322.688
fl1400	0.1	12205.613	13233.629	-	13477.859	21316.055	15140.623	12197.452	12798.116	12306.864
	0.2	14787.395	15942.804	-	17172.514	19089.942	24057.274	15191.927	15068.984	14699.926
	0.3	18116.830	18202.141	-	21843.433	21104.967	33027.235	17420.269	16760.144	16554.017
	0.4	20154.319	19739.198	-	20762.190	23231.054	42049.700	19274.755	18093.830	18085.214
	0.5	21693.935	20909.939	-	24785.550	25803.366	51128.769	20888.148	19184.967	19386.313
	0.6	23159.795	22776.792	-	26687.656	26851.430	60258.552	22322.891	20107.837	20513.914
	0.7	24332.546	23037.766	-	29088.485	27725.609	69427.943	23615.400	20911.774	21503.695
	0.8	25411.602	25627.755	-	29442.936	33431.359	78624.671	24789.772	21630.064	22378.725
	0.9	25666.749	26457.944	-	26548.602	30598.040	87841.836	25862.786	22285.017	23154.379

Table A.1: Expected a priori tour lengths for 17 construction heuristics across 9 homogeneous TSPLIB instances (*continued*).

Instance	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2
eil51	0.1	137.278	136.344	135.841	138.494	138.392	138.392	140.465	137.015
	0.2	197.617	198.561	196.552	195.799	196.033	196.033	199.139	199.249
	0.3	244.62	249.588	243.000	243.647	244.832	244.832	246.014	248.978
	0.4	288.33	294.270	283.470	285.501	287.452	287.452	284.146	295.083
	0.5	319.416	315.761	313.928	318.113	318.113	318.113	320.659	313.940
	0.6	353.812	353.316	342.195	346.969	346.891	346.891	338.991	339.661
	0.7	372.997	371.208	374.697	372.364	373.106	373.106	380.361	375.340
	0.8	401.002	391.449	399.176	397.875	397.875	397.875	396.716	396.884
	0.9	414.483	419.883	425.837	417.694	417.178	417.178	412.689	426.522
kroA100	0.1	9231.651	9291.384	9288.348	9351.961	9309.122	9309.122	9314.835	9282.302
	0.2	11847.439	11839.438	11932.091	11982.913	11805.786	11805.786	11993.867	12110.301
	0.3	14010.43	14240.410	13621.180	14066.908	14066.908	14066.908	14032.011	13882.650
	0.4	15396.697	16491.689	15192.439	15874.193	15874.193	15874.193	15419.671	15448.008
	0.5	17445.15	17097.308	16581.143	17163.192	17163.192	17163.192	16757.035	17071.578
	0.6	18803.274	18482.444	18031.540	18341.583	18341.583	18341.583	18235.299	18036.357
	0.7	18831.798	19825.931	19043.669	19369.967	19369.967	19369.967	19114.302	19451.138
	0.8	19761.086	19830.957	21142.344	20045.746	20045.746	20045.746	20856.164	20225.944
	0.9	21389.855	20654.181	21403.124	20983.083	20983.083	20983.083	20956.033	21100.680
eil101	0.1	202.189	202.303	204.046	204.853	203.233	203.233	197.824	198.128
	0.2	290.047	295.503	286.746	293.875	285.877	285.877	286.462	284.696
	0.3	360.278	367.598	359.754	355.892	360.118	360.118	367.241	361.286
	0.4	441.968	417.428	422.645	414.147	412.874	412.874	419.719	429.877
	0.5	457.865	486.122	471.501	464.616	470.821	470.821	464.751	472.395
	0.6	515.058	523.588	522.342	527.799	527.799	527.799	510.722	510.722
	0.7	571.539	553.491	582.433	561.718	561.718	561.718	546.058	554.056
	0.8	598.491	591.071	609.906	593.777	593.777	593.777	580.492	580.497
	0.9	631.367	618.136	645.874	624.389	624.389	624.389	633.134	615.005
ch150	0.1	2590.173	2585.201	2507.661	2503.425	2520.211	2520.211	2603.731	2501.100
	0.2	3526.228	3638.065	3506.983	3444.699	3528.621	3528.621	3511.468	3511.448
	0.3	4334.455	4204.976	4221.436	4106.942	4201.539	4201.539	4261.077	4261.077
	0.4	4684.964	4722.404	4857.201	4766.410	5004.649	5004.649	4796.416	4796.416
	0.5	5155.613	5305.440	5275.211	5243.240	5497.874	5497.874	5228.888	5228.888
	0.6	5751.95	5652.656	5728.917	5595.094	5794.667	5794.667	5645.950	5645.950
	0.7	6090.154	6027.813	6121.016	6066.154	6066.154	6066.154	5805.824	5805.824
	0.8	6360.882	6384.323	6294.408	6314.123	6314.123	6314.123	6102.275	6102.275
	0.9	6599.173	6486.921	6559.809	6542.111	6542.111	6542.111	6476.931	6476.931
d198	0.1	9916.354	9921.574	9915.537	9926.728	7457.494	7457.494	9929.436	9960.231
	0.2	11495.504	11326.167	9485.405	11438.489	9426.275	9426.275	11679.334	11309.852
	0.3	12281.512	12353.383	10875.435	12571.157	12294.134	12294.134	12624.565	12598.630
	0.4	13358.751	13023.046	13257.214	13843.165	13088.697	13088.697	11779.807	12028.299
	0.5	13735.132	13847.819	12786.897	14584.269	12643.903	12643.903	13108.435	13012.139
	0.6	14277.842	14666.874	13489.945	15173.709	13432.498	13432.498	13695.619	13898.062
	0.7	15349.297	15187.891	14218.630	15654.499	14138.404	14138.404	14376.953	14432.034
	0.8	15396.966	15936.695	15095.984	16148.838	15174.456	15174.456	15006.974	14898.794
	0.9	16514.101	16010.260	15471.162	16551.562	15605.171	15605.171	15613.390	15608.298
pr439	0.1	64868.87	57381.478	50609.565	53489.691	54401.892	54401.892	54685.252	54515.679
	0.2	80876.028	81006.338	72462.327	71576.215	74586.782	74586.782	72877.475	72877.475
	0.3	98791.284	95099.492	89278.527	87126.908	87126.908	87126.908	86079.896	92419.982
	0.4	116727.905	114709.924	102294.093	96170.559	96170.559	96170.559	96001.201	96001.201
	0.5	134693.011	127010.212	112991.720	109240.819	109240.819	109240.819	110630.944	110630.944
	0.6	141942.05	142743.105	117689.510	112824.007	113728.621	113728.621	117230.858	122395.761
	0.7	150498.082	160232.329	125223.295	118251.107	118251.107	118251.107	124769.513	124769.513
	0.8	169573.331	165100.542	129594.561	123329.384	123329.384	123329.384	128656.752	128656.752
	0.9	179931.452	176816.395	136926.052	127636.200	127636.200	127636.200	132167.828	132167.828

Table A.2: Expected a priori tour lengths for 17 construction heuristics after improvement by 2.5-opt-EEais across 9 homogeneous TSPLIB instances.

Instance	Probability	Random search	Random best	Probabilistic Savings	Supersavings	Newsave	Globalsave	PNN1	PNN2
rat783	0.1	3366.336	3396.450	3325.293	3370.875	3359.265	3359.265	3342.464	3310.405
	0.2	4811.741	4742.651	4918.533	4821.446	4877.018	4877.018	4740.069	4798.329
	0.3	5883.345	5800.609	6067.243	5855.477	5963.833	5963.833	5732.429	5761.739
	0.4	6663.34	6706.673	6867.770	6558.133	6751.253	6751.253	6591.553	6591.553
	0.5	7281.616	7279.962	7526.923	7423.900	7423.900	7423.900	7225.896	7225.896
	0.6	7891.904	7958.487	8098.565	7946.477	7946.477	7946.477	7781.638	7781.638
	0.7	8466.103	8521.920	8487.222	8403.715	8403.715	8403.715	8167.723	8116.528
	0.8	8893.47	8938.810	8963.508	8827.375	8827.375	8827.375	8635.548	8635.548
	0.9	9450.844	9206.726	9287.658	9184.294	9184.294	9184.294	9076.966	9076.966
pr1002	0.1	126117.45	123990.364	120818.985	117372.858	117715.529	-	116445.781	121933.758
	0.2	171751.404	170148.704	160167.551	159853.379	161685.585	-	151520.122	150773.329
	0.3	207469.662	207111.535	193857.231	186299.929	187957.525	-	184873.538	184873.538
	0.4	240391.892	244772.844	219397.064	208536.923	208536.923	-	211672.193	211672.193
	0.5	276426.887	280173.983	242516.084	225006.225	225006.225	-	231445.240	231445.240
	0.6	301219.093	306587.621	264424.163	239368.503	239368.503	-	252850.170	254063.542
	0.7	331869.466	347180.760	284419.297	252313.710	252313.710	-	271676.740	271676.740
	0.8	367252.99	366466.857	302657.800	264406.002	264406.002	-	278980.661	278980.661
	0.9	395538.789	388178.345	313269.526	275753.302	275753.302	-	295767.425	295767.425
fl1400	0.1	10412.357	9920.488	9686.782	9405.269	9610.597	-	9799.072	10193.901
	0.2	12733.692	12679.548	12113.538	11703.798	12112.403	-	12535.729	12593.880
	0.3	14674.366	16018.628	13913.695	13900.531	13900.531	-	14135.293	14242.349
	0.4	15690.444	17872.072	15657.805	15363.990	15363.990	-	16088.989	16088.989
	0.5	19425.719	18714.699	17082.859	16609.350	16609.350	-	17838.261	17838.261
	0.6	22864.522	22342.565	18336.377	17753.336	17753.336	-	18716.030	18716.030
	0.7	22692.	20474.574	18925.853	18808.300	18808.300	-	19304.740	19805.893
	0.8	26329.2	22710.013	19952.153	19609.599	19609.599	-	19886.027	19916.430
	0.9	22834.502	23326.749	21279.331	20362.344	20362.344	-	21525.299	21271.678

Table A.2: Expected a priori tour lengths for 17 construction heuristics after improvement by 2.5-opt-EEais across 9 homogeneous TSPLIB instances (*continued*).

Instance	Probability	ANNA0	ANNA1	Iterated NN	Random NN (type I)	Random NN (type II)	Radial sort	Spacefilling curves	Farthest insertion	Nearest insertion
eil51	0.1	140.413	136.246	138.426	136.179	135.898	135.044	133.325	133.538	137.721
	0.2	195.488	199.464	198.59	195.289	199.152	195.610	192.362	192.641	198.661
	0.3	247.157	248.182	252.728	248.191	241.925	243.345	237.407	241.380	247.282
	0.4	289.691	283.168	282.331	288.678	291.201	281.940	274.937	280.298	287.082
	0.5	328.173	327.724	312.289	325.985	311.354	321.506	306.520	311.050	319.186
	0.6	357.904	356.666	342.54	351.525	351.049	357.645	337.069	341.440	339.588
	0.7	376.360	375.066	366.787	372.274	377.068	381.209	361.575	362.467	372.586
	0.8	395.758	396.716	387.722	401.338	398.197	400.939	383.336	393.520	396.496
	0.9	417.051	412.689	419.15	420.507	430.009	421.694	403.822	429.275	431.876
kroA100	0.1	9304.518	9063.808	9322.474	9291.808	9273.473	9319.860	9254.029	9231.833	9309.233
	0.2	12110.301	11915.498	11867.152	11873.506	11853.031	11832.834	12028.397	11937.137	11970.638
	0.3	13882.650	13900.024	13694.144	13796.597	13875.612	13816.794	13937.806	13813.818	13956.429
	0.4	15448.008	15925.546	15249.975	16372.908	15778.588	15714.815	16426.450	15445.307	15771.320
	0.5	17071.578	17074.23	16529.842	17185.682	17035.885	17190.746	17319.851	16702.865	17989.164
	0.6	18036.357	19397.407	18559.475	18258.699	18020.485	18402.770	18811.068	19172.817	19022.827
	0.7	19451.138	20340.361	19328.986	20258.710	20741.711	19364.813	18829.975	20175.830	19929.755
	0.8	20225.944	21153.872	19715.435	20277.944	20849.857	20988.997	20636.074	21147.020	20796.061
	0.9	21100.680	21329.679	21148.475	21740.686	21009.738	21776.090	21536.129	22377.356	21622.178

Table A.2: Expected a priori tour lengths for 17 construction heuristics after improvement by 2.5-opt-EEais across 9 homogeneous TSPLIB instances (*continued*).

Instance	Probability	ANNA0	ANNA1	Iterated NN	Random NN (type I)	Random NN (type II)	Radial sort	Spacefilling curves	Farthest insertion	Nearest insertion
eil101	0.1	197.418	203.26	202.752	203.548	204.088	202.939	201.964	201.053	203.785
	0.2	292.301	285.444	285.457	291.692	292.695	289.488	290.985	287.999	292.849
	0.3	356.145	357.779	349.345	363.595	368.288	353.741	353.165	354.425	361.778
	0.4	423.164	420.564	412.042	411.023	422.167	410.425	414.033	409.882	427.993
	0.5	459.608	464.665	480.792	465.118	497.410	465.393	468.371	471.997	483.847
	0.6	508.089	511.961	527.501	516.398	504.662	529.737	506.498	510.062	521.715
	0.7	564.258	577.857	576.024	558.196	553.071	547.698	552.825	561.032	567.073
	0.8	581.399	592.806	600.98	585.960	601.699	590.453	592.212	603.264	592.317
	0.9	613.099	611.098	636.77	642.710	637.134	636.887	634.402	619.510	620.950
ch150	0.1	2578.448	2581.756	2586.038	2504.781	2563.965	2566.203	2571.124	2572.139	2606.108
	0.2	3676.275	3459.682	3508.824	3467.588	3508.340	3508.607	3557.059	3527.757	3538.474
	0.3	4128.424	4359.024	4136.393	4127.326	4216.268	4163.643	4252.154	4113.978	4228.174
	0.4	4615.160	4997.808	4688.925	4925.931	4868.039	4787.534	4739.995	4624.164	4795.699
	0.5	5108.622	5276.761	5066.393	5248.470	5416.708	5421.539	5330.816	5198.562	5291.681
	0.6	5697.115	5697.115	5439.191	5714.885	5728.637	5737.543	5734.989	5602.381	5701.023
	0.7	6037.860	5992.606	5788.943	6094.762	5998.494	6200.078	6187.756	6051.468	5916.914
	0.8	6228.506	6398.779	6146.88	6379.459	6648.802	6350.278	6446.582	6354.944	6507.343
	0.9	6553.266	6553.266	6403.656	6734.007	6663.503	6442.871	6678.489	6681.231	6820.386
d198	0.1	9950.050	9897.27	10760.909	9944.007	9944.469	9903.670	9910.111	9897.160	9986.260
	0.2	9500.952	11406.237	11317.234	11358.722	11491.334	12343.836	11354.565	11291.085	11418.951
	0.3	10805.949	11165.289	12483.682	12445.981	12263.353	12454.554	12420.732	12214.485	12389.219
	0.4	11925.259	12142.889	13122.718	13058.473	13097.380	13334.349	13227.506	13073.191	13258.710
	0.5	13127.269	13108.435	12776.782	13370.778	13855.111	14104.846	13949.933	13699.168	14006.719
	0.6	13854.623	13695.619	13765.227	13883.079	14613.479	14584.753	14532.691	14373.791	14688.516
	0.7	14484.674	14423.386	15006.717	15023.724	15268.743	15078.805	15257.011	14874.451	15078.645
	0.8	15049.310	15006.974	15177.519	15147.957	15567.336	15610.868	15512.271	15494.368	15596.185
	0.9	15613.390	15613.39	15833.532	15844.917	16320.333	15773.955	16061.670	15880.838	16131.536
pr439	0.1	56127.897	56425.515	53635.721	57191.337	56497.251	53393.061	53328.502	51252.303	58688.399
	0.2	77249.733	75015.894	72075.265	73737.110	80122.807	78166.907	72184.705	71244.368	78231.051
	0.3	84052.546	88019.885	84705.727	90418.865	93547.695	91767.355	90775.457	82960.526	90870.288
	0.4	102480.143	102480.143	94209.092	97678.736	107052.018	106181.487	102543.449	93024.945	96122.778
	0.5	110791.111	114041.2	100366.869	114082.885	115470.632	125906.557	114157.750	100202.565	115928.495
	0.6	121345.148	119190.385	116014.545	124000.319	128088.587	132573.623	122878.099	113918.927	120230.423
	0.7	121395.627	124769.513	122019.85	137655.145	137701.872	149968.193	132749.738	119692.673	128884.825
	0.8	129864.133	128656.752	127750.405	144116.755	143482.128	157421.388	144686.081	124464.586	135118.157
	0.9	132973.588	132167.828	130534.369	145467.951	144940.763	169500.778	142499.396	128906.330	139538.047
rat783	0.1	3370.647	3432.289	3361.722	3386.824	3389.636	3383.320	3374.596	3358.506	3345.183
	0.2	4796.448	5071.211	4917.204	4803.586	4824.106	4797.853	4848.263	4984.506	4844.227
	0.3	5754.314	5893.479	5965.594	5845.552	5847.363	5785.292	5829.550	5893.474	5805.492
	0.4	6534.633	6748.099	6721.291	6637.666	6609.099	6688.285	6588.453	6640.196	6653.037
	0.5	7233.079	7437.935	7217.75	7250.937	7137.574	7319.931	7269.692	7434.106	7340.451
	0.6	7809.498	7882.611	7793.749	7803.945	7909.644	7881.016	7927.101	7996.338	7940.910
	0.7	8275.027	8335.156	8269.876	8365.059	8268.624	8456.037	8593.970	8525.169	8474.339
	0.8	8644.783	8647.218	8689.292	8719.428	8758.720	8897.606	9065.709	9019.180	8963.035
	0.9	9028.027	9083.244	8935.952	8998.830	9021.491	9272.746	9521.563	9547.395	9452.784
pr1002	0.1	115462.182	115203.786	-	123569.152	124054.883	126176.233	119253.207	118321.790	121280.866
	0.2	158102.998	159939.97	-	166739.685	166320.311	167561.808	164190.214	163428.270	162014.553
	0.3	189446.321	190730.873	-	197689.065	195330.809	204965.908	197210.902	196255.287	192493.906
	0.4	216865.303	215338.614	-	215962.251	223446.385	241598.520	226653.359	222919.231	216000.187
	0.5	239522.274	234230.091	-	238154.521	249408.573	274279.450	255581.156	245536.406	236615.250
	0.6	262075.598	256216.3	-	260899.537	266062.590	300528.710	276456.927	265419.061	253665.364
	0.7	275815.499	273073.632	-	277841.878	282910.636	330805.854	298806.081	283455.069	269377.595
	0.8	284116.127	288918.923	-	292468.908	303407.733	358861.430	318456.500	298351.251	282902.719
	0.9	302935.465	301489.863	-	324005.045	315653.219	396864.132	339466.239	310972.224	295219.800
fl1400	0.1	9575.697	10194.696	-	9940.275	10704.412	9960.208	10171.096	10268.125	10532.112
	0.2	12471.664	12465.683	-	12909.289	13410.643	12426.861	13079.572	13191.661	13148.569
	0.3	14521.878	15623.372	-	14531.298	14821.343	13888.166	15094.494	14791.895	15220.425
	0.4	16567.347	16718.347	-	17732.699	16987.302	16901.714	16344.288	16149.131	16607.188
	0.5	17535.630	18038.587	-	18562.456	17323.009	17371.847	17462.266	17339.308	17287.100
	0.6	19155.466	19147.769	-	19072.504	18209.278	19514.257	18737.622	18351.626	18951.347
	0.7	19984.916	20028.895	-	19621.460	20989.877	20419.954	19752.164	19111.762	19822.815
	0.8	21114.475	20272.728	-	20339.270	22833.053	21419.336	19870.812	20119.516	20571.672
	0.9	21299.248	21409.644	-	21218.528	22915.004	21516.875	20949.786	20765.496	21453.807

Table A.2: Expected a priori tour lengths for 17 construction heuristics after improvement by 2.5-opt-EEais across 9 homogeneous TSPLIB instances (*continued*).

B | Pseudo-codes of savings procedures for the PTSP

The so-called Supersavings procedures are variants and adaptations of the original savings procedure proposed by Clarke and Wright (1964) for the deterministic traveling salesman problem (TSP). This section reports the adaptations of this type of construction heuristic to stochastic TSPs. In particular, I present the pseudo-codes of the savings procedures proposed by Jaillet (1985), Jézéquel (1985), and Rossi and Gavioli (1987) -– see Algorithms 4, 5, 6, and 7.

Algorithm 4 Pseudo-code of the Supersavings procedure by Jaillet (1985)

Input: A set of nodes including a depot and several customers, along with their "homogeneous" probability of occurrence, the distance between each pair of nodes, and the degree of approximation of the actual expected cost function of the PTSP γ.

Initialisation step: Initialize the best solution found so far, say ρ^{best}, to an empty tour $\emptyset$.

Iterative step:

1: **for** $g = 2$ to γ **do**
2: Construct a partial initial solution by forming subtours of size g; that is, subtours that leave the depot, visit $g-1$ given nodes, and get back to the depot;
3: For each pair of subtours of length g, consider the possible ways of merging them and construct the list of expected savings accordingly;
4: **while** the number of subtours is larger than 1 **do**
5: Merge the current subtours with the maximum expected savings in the list;
6: Reduce the savings list to reflect the new (partial) solution;
7: **end while**
8: Update the best solution found so far ρ^{best};
9: **end for**

Output: An a priori PTSP tour ρ^{best} along with its expected cost.

Algorithm 5 Pseudo-code of the NewSave Procedure of Jézéquel (1985)

Input: A set of nodes including a depot and several customers, along with their "homogeneous" probability of occurrence, the distance between each pair of nodes, and the degree of approximation of the actual expected cost function of the PTSP γ.

Initialisation step: Initialize the best solution found so far ρ^{best} to an empty tour;

Iterative step:

1: **for** $g = 2$ to γ **do**
2: Construct a partial initial solution by forming subtours of size g; that is, subtours that leave the depot, visit $g-1$ given nodes, and get back to the depot;
3: For each pair of subtours of length g, consider the possible ways of merging them and build the list of expected savings accordingly;
4: Merge the two subtours with the largest deterministic savings and refer to it as the embryo of the PTSP tour – this is the largest subtour computed so far;
5: **while** the number of subtours is larger than 1 **do**
6: Merge the current largest subtour with the subtour that corresponds to the maximum expected savings in the list;
7: Reduce the savings list to reflect the new (partial) solution;
8: **end while**
9: Update the best solution found so far ρ^{best};
10: **end for**

Output: An a priori PTSP tour along with its expected cost.

Algorithm 6 Pseudo-code of the GlobalSave Procedure of Jézéquel (1985)

Input: A set of nodes including a depot and several customers, along with their homogeneous probability of occurrence, the distance between each pair of nodes, and the degree of approximation of the actual expected cost function of the PTSP γ.

Initialisation step: Compute a Hamiltonian tour ρ for the deterministic TSP version of the problem using an appropriate TSP construction heuristic, which may be improved by a local search mechanism. Choose a discrete set of k homogeneous coverage probabilities $\{\pi_1, \ldots, \pi_i, \ldots, \pi_k\}$, compute the expected cost $E[L(\rho, \pi_i)]$ of ρ for each π_i $(i = 1, \ldots, k)$, and initialize the best a priori tour ρ_i^{best} corresponding to coverage probability π_i to ρ.

Iterative step:

1: **for** $i = 1$ to k **do**

2: Construct an a priori tour ρ_i using Newsave procedure with parameters π_i and γ, and compute the expected cost $E[L(\rho_i, \pi_j)]$ of ρ_i for each π_j $(j = 1, \ldots, k; j \neq i)$.

3: Then, compare the expected costs $E[L(\rho_i, \pi_j)]$; $j = 1, \ldots, k$ with the cost of ρ_i^{best} and update ρ_i^{best} accordingly;

4: **end for**

Output: k a priori PTSP tours $\rho_1^{best}, \ldots, \rho_k^{best}$, each associated with a different coverage probability, along with their expected costs.

Algorithm 7 Pseudo-code of the Probabilistic Clarke & Wright procedure of Rossi and Gavioli (1987)

Input: A set of nodes including a depot and several customers, along with their homogeneous probability of occurrence, and the distance between each pair of nodes.

Initialisation step:

1: Construct a partial initial solution by forming subtours of size 2; that is, subtours that leave the depot, visit a single node, and get back to the depot;

2: For each pair of subtours of length 2, consider the two possible ways of connecting them and build the list of expected savings accordingly;

Iterative step:

3: **while** the number of subtours is larger than 1 **do**

4: Merge the two subtours corresponding to the maximum expected savings;

5: Update the entries of the reduced savings list that would involve merging the newly added subtour with the old ones;

6: **end while**

7: Compute the expected cost of the obtained a priori PTSP tour.

Output: An a priori PTSP tour along with its expected cost.

C | Empirical performance of GENS-H versus nearest neighbour-based procedures

Tables C.1 and C.2 report the performance of GENS-H versus a number of construction heuristics inspired by the nearest neighbour principle. The nearest neighbour heuristics have all been adapted to cope with the probabilistic nature of the PTSP.

The Almost Nearest Neighbour Algorithm or Probabilistic Nearest Neighbour type 1 (PNN1) algorithm is based on sequentially adding nodes to a partial tour based on the minimum increase in expected length. It was introduced by Jaillet (1985) and further developed by Rossi and Gavioli (1987) to account for the costs of returning to the depot, resulting in Probabilistic Nearest Neighbour type 2 (PNN2). Jaillet (1985) also proposed a slightly modified version of his PNN1 algorithm in which first a tree of two nodes is grown from the depot deterministically to account for the 'heavy' weighting of nodes close to the depot. I refer to this variant as ANNA. Building upon this idea also led to yet another, generalized, extension of this method, called ANNAe, in which first two trees of size k are deterministically grown from the depot before the other nodes are added in a probabilistic fashion.

In addition to the variants that specifically integrate probabilistic components of the problem at hand, a number of variants of the deterministic nearest neighbour algorithms were also developed for the PTSP. Bianchi (2006) considers a heuristic

that repeatedly constructs a tour based on the nearest neighbour principle in deterministic sense, using each of the N nodes in the graph as a new starting point of the tour. It compares the N resulting tours based on their expected lengths and returns the tour with the smallest length. We refer to this method as Iterative Nearest Neighbour (INN). Liu (2010) considers two types of nearest neighbour algorithms for the PTSP using random addition of the nearest and second nearest nodes from two ends of the depot (Randomized Nearest Neighbour type 1, RNN1) or from two ends growing out of a randomly chosen point farthest from the depot (Randomized Nearest Neighbour type 2, RNN2).

A final method that was considered is not based on the nearest neighbour principle, but generates a set of completely random tours and picks the best one in terms of expected length as its final tour (Bianchi, 2003). This method is referred to as Random Best (RB).

In none of the reported cases in tables C.1 and C.2 the benchmark heuristics are able to outperform the GENS-H algorithms. In fact, in most cases the competing algorithms perform considerably worse than GENS-H. This is in line with previous results on the empirical performance of nearest neighbour algorithms for the PTSP, e.g. Jézéquel (1985), Rossi and Gavioli (1987), Bianchi (2003) and Bianchi (2006).

Instance	p	RDM	TS	ILS	VNS	PNN1	PNN2	ANNA	ANNAe
eil51	0.1	139.41	**139.17**	**139.17**	140.28	156.68	159.76	154.12	157.87
	0.2	**205.00**	**205.00**	**205.00**	**205.00**	236.20	240.37	238.46	249.21
	0.3	**253.52**	**253.52**	**253.52**	**253.52**	284.48	292.00	288.33	311.44
	0.4	**295.62**	**295.62**	**295.62**	**295.62**	323.15	323.03	335.40	355.33
	0.5	**331.74**	**331.74**	**331.74**	**331.74**	356.02	361.58	391.72	390.03
	0.6	**358.05**	**358.05**	**358.05**	**358.05**	401.93	391.59	421.44	419.17
	0.7	**382.47**	**382.47**	**382.47**	**382.47**	444.41	439.43	447.76	441.39
	0.8	**405.69**	**405.69**	**405.69**	**405.69**	468.26	461.42	471.75	468.26
	0.9	**428.10**	**428.10**	**428.10**	**428.10**	490.16	471.16	494.03	490.16
kroA100	0.1	**9507.26**	**9507.26**	9530.55	9568.61	10276.26	10792.98	10233.02	10741.78
	0.2	13334.84	**12448.99**	12602.01	12597.66	13513.93	13513.93	13975.82	14719.12
	0.3	14845.51	14845.51	**14788.62**	14845.51	15801.05	15801.05	15711.54	17399.98
	0.4	**16462.33**	**16462.33**	**16462.33**	**16462.33**	17737.08	17737.08	18571.97	19432.18
	0.5	**17832.62**	**17832.62**	**17832.62**	**17832.62**	19265.12	19265.12	20078.69	21065.83
	0.6	20480.26	**19043.42**	**19043.42**	**19043.42**	21206.46	21206.46	21878.40	21938.26
	0.7	**20144.44**	**20144.44**	**20144.44**	**20144.44**	22400.15	22400.15	23106.78	22980.71
	0.8	**21166.81**	**21166.81**	**21166.81**	**21166.81**	24812.55	24812.55	24449.21	23624.83
	0.9	**22130.46**	**22130.46**	**22130.46**	**22130.46**	25967.87	25967.87	25721.40	25330.20
eil101	0.1	**211.28**	**211.28**	213.96	213.96	228.00	230.64	234.41	258.03
	0.2	311.99	**305.04**	**305.04**	**305.04**	350.68	351.67	356.44	376.43
	0.3	**375.72**	**375.72**	**375.72**	**375.72**	436.59	438.74	452.92	465.08
	0.4	454.45	**435.43**	**435.43**	**435.43**	512.46	518.81	525.43	535.84
	0.5	**487.18**	**487.18**	**487.18**	**487.18**	544.72	547.19	598.48	595.09
	0.6	**535.07**	**535.07**	**535.07**	**535.07**	610.11	610.11	650.97	646.24
	0.7	597.65	**580.63**	**580.63**	**580.63**	691.33	688.28	696.24	690.07
	0.8	**615.95**	**615.95**	**615.95**	**615.95**	731.83	733.10	738.48	711.68
	0.9	**645.54**	**645.54**	**645.54**	**645.54**	766.67	762.59	776.17	768.80
ch150	0.1	2679.96	**2674.33**	2674.68	2674.68	3071.03	3226.60	3389.27	3424.07
	0.2	**3617.91**	**3617.91**	3667.64	**3617.91**	4105.30	4129.89	4765.42	4806.26
	0.3	4617.97	**4366.72**	**4366.72**	4380.13	5012.08	5012.08	5463.78	5659.72
	0.4	**4935.82**	5012.95	**4935.82**	**4935.82**	6153.16	6153.58	5391.96	6239.14
	0.5	5455.49	**5442.57**	**5442.57**	**5442.57**	6691.71	6691.71	6617.10	6664.04
	0.6	6250.98	**5860.31**	**5860.31**	**5860.31**	7080.99	7080.99	7133.53	7133.53
	0.7	**6226.45**	**6226.45**	**6226.45**	**6226.45**	6584.31	6584.31	7044.46	7390.99
	0.8	6967.19	**6490.05**	**6490.05**	**6490.05**	6703.26	6703.26	7339.92	7669.73
	0.9	**6736.36**	**6736.36**	**6736.36**	**6736.36**	7528.47	7528.47	7921.11	7921.11
d198	0.1	**9975.09**	**9975.09**	**9975.09**	9998.01	10757.48	10768.45	10768.02	10531.21
	0.2	11509.65	11595.24	**11509.52**	11522.53	12626.68	12755.52	12775.86	12586.09
	0.3	12694.47	12567.12	**12565.37**	**12565.37**	14117.40	14341.83	14304.02	13968.58
	0.4	**13450.52**	13508.32	13467.84	13458.76	15125.01	15308.69	15249.33	14858.68
	0.5	14276.34	**14242.28**	14245.08	14245.08	15853.64	15868.32	16008.53	15853.64
	0.6	14999.38	15001.82	**14911.64**	**14911.64**	16462.96	16481.12	16649.10	16462.96
	0.7	15593.96	15593.96	**15521.57**	**15521.57**	16989.19	17012.08	17207.14	16824.40
	0.8	16126.30	16126.30	**16091.42**	**16091.42**	17453.16	17482.32	17703.26	17453.16
	0.9	**16608.87**	**16608.87**	**16608.87**	**16608.87**	17867.45	17904.40	18149.89	17867.45

Table C.1: Results for GENS-H versus Nearest Neighbour-based procedures

Instance	p	RDM	TS	ILS	VNS	PNN1	PNN2	ANNA	ANNAe
pr439	0.1	**52693.03**	**52693.03**	**52693.03**	**52693.03**	58228.58	58683.51	58093.36	65611.70
	0.2	**69836.51**	**69836.51**	**69836.51**	**69836.51**	75631.48	75631.48	83363.20	86420.23
	0.3	**80900.61**	**80900.61**	**80900.61**	**80900.61**	88092.98	88112.15	96787.77	97595.83
	0.4	**89619.79**	**89619.79**	**89619.79**	**89619.79**	103449.39	103449.39	105506.83	105506.83
	0.5	101542.04	96918.32	96918.32	**96826.79**	109206.89	109206.89	111639.44	111620.04
	0.6	103263.13	**102935.88**	103263.13	103263.13	113862.26	114082.28	116685.62	116638.41
	0.7	108934.76	108934.76	**108404.18**	**108404.18**	120940.18	120940.18	121013.02	120940.18
	0.8	113456.13	**113416.38**	113456.13	113456.13	124735.72	124735.72	124832.24	124735.72
	0.9	117425.73	117425.73	**117401.73**	117425.73	128154.19	128154.19	128272.72	128154.19
rat783	0.1	**3718.96**	**3718.96**	**3718.96**	**3718.96**	4728.89	4729.83	4600.47	4977.07
	0.2	**5186.55**	5193.31	**5186.55**	**5186.55**	6001.64	6000.71	6166.91	6458.97
	0.3	**6207.47**	**6207.47**	**6207.47**	**6207.47**	7098.10	7100.63	7028.49	7474.58
	0.4	**7004.56**	**7004.56**	**7004.56**	**7004.56**	7815.25	7815.25	7671.20	8122.86
	0.5	**7651.45**	**7651.45**	**7651.45**	**7651.45**	8655.99	8655.99	8699.38	8790.34
	0.6	**8208.54**	**8208.54**	**8208.54**	**8208.54**	9194.94	9194.94	9351.31	9350.06
	0.7	**8655.53**	**8655.53**	**8655.53**	**8655.53**	9725.05	9731.78	9747.87	9419.29
	0.8	**9058.67**	**9058.67**	**9058.67**	**9058.67**	10193.85	10193.85	10316.68	10180.81
	0.9	10820.10	**9427.16**	**9427.16**	**9427.16**	10728.19	10728.19	10694.65	10667.79

Table C.1: Results for GENS-H versus Nearest Neighbour-based procedures (*continued*)

Instance	p	RDM	TS	ILS	VNS	INN	RNN1	RNN2	RB
eil51	0.1	139.41	**139.17**	**139.17**	140.28	149.33	155.51	157.62	177.35
	0.2	**205.00**	**205.00**	**205.00**	**205.00**	230.83	260.78	273.87	340.01
	0.3	**253.52**	**253.52**	**253.52**	**253.52**	291.34	329.77	314.44	505.32
	0.4	**295.62**	**295.62**	**295.62**	**295.62**	338.34	370.03	406.96	636.69
	0.5	**331.74**	**331.74**	**331.74**	**331.74**	375.78	456.92	400.88	758.63
	0.6	**358.05**	**358.05**	**358.05**	**358.05**	406.54	472.90	435.25	921.84
	0.7	**382.47**	**382.47**	**382.47**	**382.47**	432.31	528.03	468.93	1036.37
	0.8	**405.69**	**405.69**	**405.69**	**405.69**	455.76	538.47	512.55	1206.49
	0.9	**428.10**	**428.10**	**428.10**	**428.10**	476.28	506.09	614.75	1303.82
kroA100	0.1	**9507.26**	**9507.26**	9530.55	9568.61	10598.41	10913.30	11456.85	17654.71
	0.2	13334.84	**12448.99**	12602.01	12597.66	14196.10	14934.87	15477.08	33672.34
	0.3	14845.51	14845.51	**14788.62**	14845.51	16568.85	18423.91	19834.25	49380.26
	0.4	**16462.33**	**16462.33**	**16462.33**	**16462.33**	18434.39	19526.71	19258.19	63895.49
	0.5	**17832.62**	**17832.62**	**17832.62**	**17832.62**	20024.25	23614.28	27773.49	80300.69
	0.6	20480.26	**19043.42**	**19043.42**	**19043.42**	21443.93	23917.04	22319.28	95531.77
	0.7	**20144.44**	**20144.44**	**20144.44**	**20144.44**	22747.67	24175.15	26298.26	109932.93
	0.8	**21166.81**	**21166.81**	**21166.81**	**21166.81**	23965.39	24359.68	31971.83	126430.34
	0.9	**22130.46**	**22130.46**	**22130.46**	**22130.46**	25114.25	31666.85	29375.90	141500.20

Table C.2: Results for GENS-H versus Nearest Neighbour-based procedures

Instance	p	RDM	TS	ILS	VNS	INN	RNN1	RNN2	RB
eil101	0.1	**211.28**	**211.28**	213.96	213.96	232.17	249.62	249.86	346.51
	0.2	311.99	**305.04**	**305.04**	**305.04**	344.73	427.68	404.57	682.48
	0.3	**375.72**	**375.72**	**375.72**	**375.72**	431.77	554.32	502.83	1002.03
	0.4	454.45	**435.43**	**435.43**	**435.43**	501.52	545.63	557.34	1325.13
	0.5	**487.18**	**487.18**	**487.18**	**487.18**	553.83	681.96	675.35	1602.22
	0.6	**535.07**	**535.07**	**535.07**	**535.07**	598.82	657.93	685.24	1913.60
	0.7	597.65	**580.63**	**580.63**	**580.63**	639.23	758.26	703.51	2206.83
	0.8	**615.95**	**615.95**	**615.95**	**615.95**	676.34	706.78	839.49	2539.58
	0.9	**645.54**	**645.54**	**645.54**	**645.54**	711.06	895.22	892.29	2814.59
ch150	0.1	2679.96	**2674.33**	2674.68	2674.68	3028.27	3474.91	3808.74	5553.00
	0.2	**3617.91**	**3617.91**	3667.64	**3617.91**	4150.68	4617.86	4545.27	10849.58
	0.3	4617.97	**4366.72**	**4366.72**	4380.13	4766.37	6376.43	6151.39	15839.10
	0.4	**4935.82**	5012.95	**4935.82**	**4935.82**	5236.07	6271.49	6781.93	20951.51
	0.5	5455.49	**5442.57**	**5442.57**	**5442.57**	5629.47	6761.85	7013.41	24974.66
	0.6	6250.98	**5860.31**	**5860.31**	**5860.31**	5974.72	7278.25	8333.93	31031.15
	0.7	**6226.45**	**6226.45**	**6226.45**	**6226.45**	6285.50	7992.04	7171.90	35148.04
	0.8	6967.19	**6490.05**	**6490.05**	**6490.05**	6569.48	7387.65	8923.50	40047.72
	0.9	**6736.36**	**6736.36**	**6736.36**	**6736.36**	6831.52	7903.10	9731.73	44921.54
d198	0.1	**9975.09**	**9975.09**	**9975.09**	9998.01	10655.94	11134.38	11446.58	21989.84
	0.2	11509.65	11595.24	**11509.52**	11522.53	12553.46	12994.96	13199.27	39848.14
	0.3	12694.47	12567.12	**12565.37**	**12565.37**	13720.59	14805.54	18523.88	58004.13
	0.4	**13450.52**	13508.32	13467.84	13458.76	14632.64	15187.59	16876.56	73379.46
	0.5	14276.34	**14242.28**	14245.08	14245.08	15393.45	18993.27	20521.05	92123.83
	0.6	14999.38	15001.82	**14911.64**	**14911.64**	16035.20	17992.43	18765.76	109460.27
	0.7	15593.96	15593.96	**15521.57**	**15521.57**	16588.01	18608.31	19887.53	126058.85
	0.8	16126.30	16126.30	**16091.42**	**16091.42**	17036.51	18827.24	20510.19	141219.76
	0.9	**16608.87**	**16608.87**	**16608.87**	**16608.87**	17427.66	19263.36	20414.76	158848.15
pr439	0.1	**52693.03**	**52693.03**	**52693.03**	**52693.03**	59141.12	66637.90	77095.14	191764.31
	0.2	**69836.51**	**69836.51**	**69836.51**	**69836.51**	77426.05	87852.66	91884.61	378116.35
	0.3	**80900.61**	**80900.61**	**80900.61**	**80900.61**	89060.67	104893.33	108852.09	561410.54
	0.4	**89619.79**	**89619.79**	**89619.79**	**89619.79**	97556.12	119943.07	130903.57	742435.17
	0.5	101542.04	96918.32	96918.32	**96826.79**	104205.19	135215.38	136008.89	916362.38
	0.6	103263.13	**102935.88**	103263.13	103263.13	109653.75	123148.39	148355.80	1107622.71
	0.7	108934.76	108934.76	**108404.18**	**108404.18**	114259.06	154099.01	159699.58	1273143.59
	0.8	113456.13	**113416.38**	113456.13	113456.13	118239.93	167461.19	158638.99	1468080.37
	0.9	117425.73	117425.73	**117401.73**	117425.73	121743.73	152702.92	171019.43	1626676.83
rat783	0.1	**3718.96**	**3718.96**	**3718.96**	**3718.96**	4741.52	5112.27	5786.79	18074.80
	0.2	**5186.55**	5193.31	**5186.55**	**5186.55**	6357.06	6521.90	7828.35	35669.69
	0.3	**6207.47**	**6207.47**	**6207.47**	**6207.47**	7349.28	8115.67	8508.94	53213.59
	0.4	**7004.56**	**7004.56**	**7004.56**	**7004.56**	8095.57	8793.87	9947.81	70609.36
	0.5	**7651.45**	**7651.45**	**7651.45**	**7651.45**	8669.39	9290.30	10722.83	87757.55
	0.6	**8208.54**	**8208.54**	**8208.54**	**8208.54**	9159.04	10107.55	9707.13	105669.05
	0.7	**8655.53**	**8655.53**	**8655.53**	**8655.53**	9593.86	11048.01	11531.20	122111.18
	0.8	**9058.67**	**9058.67**	**9058.67**	**9058.67**	9988.35	11298.10	12125.35	135115.19
	0.9	10820.10	**9427.16**	**9427.16**	**9427.16**	10351.21	11754.29	12194.17	155532.88

Table C.2: Results for GENS-H versus Nearest Neighbour-based procedures (*continued*)

Bibliography

M. Aerts, G. Molenberghs, H. Geys, and L.M. Ryan. *Topics in modelling of clustered data*. Chapman and Hall/CRC, 2002.

A. Agresti. *Categorical Data Analysis*. John Wiley & Sons, 3 edition, 2013.

A. Ak and A.L. Erera. A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation science*, 41(2):222–237, 2007.

A.M. Amar, W. Khaznaji, and M. Bellalouna. An exact resolution for the probabilistic traveling salesman problem under the a priori strategy. *Procedia Computer Science*, 108:1414–1423, 2017. International Conference on Computational Science (ICCS) 2017.

D.L. Applegate, R.E. Bixby, V. Chvatal, and W.J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

R.R. Bahadur. A representation of the joint distribution of responses to *n* dichotomous items. *Studies in item analysis and prediction*, 6:158–168, 1961.

P. Balaprakash. Ant colony optimization under uncertainty. Master's thesis, Université Libre De Bruxelles, Brussels, Belgium, Tech. Rep. TR/IRIDIA/2005-028, 2005.

P. Balaprakash. *Estimation-based metaheuristics for stochastic combinatorial optimization: case studies in stochastic routing problems*. PhD thesis, Université Libre De Bruxelles, Brussels, Belgium, Tech. Rep. TR/IRIDIA/2005-028, 2010.

P. Balaprakash, M. Birattari, Th. Stützle, and M. Dorigo. An experimental study of estimation-based metaheuristics for the probabilistic traveling salesman problem. *Learning and intelligent optimization, LION*, pages 8–12, 2007.

P. Balaprakash, M. Birattari, Th. Stützle, and M. Dorigo. Adaptive sample size and importance sampling in estimation-based local search for the probabilistic traveling salesman problem. *European Journal of Operational Research*, 199(1): 98–110, 2009a.

P. Balaprakash, M. Birattari, Th. Stützle, Z. Yuan, and M. Dorigo. Estimation-based ant colony optimization and local search for the probabilistic traveling salesman problem. *Swarm Intelligence*, 3(3):223–242, 2009b.

P. Balaprakash, M. Birattari, Th. Stützle, and M. Dorigo. Estimation-based metaheuristics for the probabilistic traveling salesman problem. *Computers & operations research*, 37(11):1939–1951, 2010.

P. Balaprakash, M. Birattari, Th. Stützle, and M. Dorigo. Estimation-based metaheuristics for the single vehicle routing problem with stochastic demands and customers. *Computational Optimization and Applications*, 61(2):463–487, 2015.

J.J. Bartholdi and L.K. Platzman. An $O(N \log N)$ planar travelling salesman heuristic based on spacefilling curves. *Operations Research Letters*, 1(4):121–125, 1982.

J.J. Bartholdi, L.K. Platzman, R.L. Collins, and W.H. Warden. A minimal technology routing system for meals on wheels. *Interfaces*, 13(3):1–8, 1983.

Th. Bartz-Beielstein and M. Preuss. Experimental research in evolutionary computation. In *Proceedings of the 9th annual conference companion on Genetic and evolutionary computation*, pages 3001–3020. ACM, 2007.

M. Bellalouna. Problèmes d'optimisation combinatoires probabilistes. Master's thesis, Ecole Nationale des Ponts et Chaussées, 1993.

R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

J.J. Bentley. Fast algorithms for geometric traveling salesman problems. *ORSA Journal on computing*, 4(4):387–411, 1992.

W.C. Benton and M.D. Rossetti. The vehicle scheduling problem with intermittent customer demands. *Computers & Operations Research*, 19(6):521–531, 1992.

P. Beraldi, G. Ghiani, G. Laporte, and R. Musmanno. Efficient neighborhood search for the probabilistic pickup and delivery travelling salesman problem. *Networks*, 45(4):195–198, 2005.

O. Berman and D. Simchi-Levi. Finding the optimal a priori tour and location of a traveling salesman with nonhomogeneous customers. *Transportation Science*, 22 (2):148–154, 1988.

D.J. Bertsimas. *Probabilistic combinatorial optimization problems*. PhD thesis, Massachusetts Institute of Technology, 1988.

D.J. Bertsimas. A vehicle routing problem with stochastic demand. *Operations Research*, 40(3):574–585, 1992.

D.J. Bertsimas and L.H. Howell. Further results on the probabilistic traveling salesman problem. *European Journal of Operational Research*, 65(1):68–95, 1993.

D.J. Bertsimas, P. Jaillet, and A.R. Odoni. A priori optimization. *Operations Research*, 38(6):1019–1033, 1990.

D.J. Bertsimas, Ph. Chervi, and M. Peterson. Computational approaches to stochastic vehicle routing problems. *Transportation science*, 29(4):342–352, 1995.

L. Bianchi. New approaches for solving the probabilistic traveling salesman problem. Master's thesis, Université Libre de Bruxelles, IRIDIA, 2003.

L. Bianchi. *Ant colony optimization and local search for the probabilistic traveling salesman problem: a case study in stochastic combinatorial optimization*. PhD thesis, Université Libre de Bruxelles, 2006.

L. Bianchi and A.M. Campbell. Extension of the 2-p-opt and 1-shift algorithms to the heterogeneous probabilistic traveling salesman problem. *European Journal of Operational Research*, 176(1):131–144, 2007.

L. Bianchi and L.M. Gambardella. Ant colony optimization and local search based on exact and estimated objective values for the probabilistic traveling salesman problem. Technical Report IDSIA-06-07, IDSIA / USI-SUPSI, 2007.

L. Bianchi and J. Knowles. Local search for the probabilistic traveling salesman problem: a proof of the incorrectness of Bertsimas' proposed 2-p-opt and 1-shift algorithms. *IDSIA*, 2002.

L. Bianchi, L.M. Gambardella, and M. Dorigo. Solving the homogeneous probabilistic traveling salesman problem by the aco metaheuristic. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Ant Algorithms*, volume 2463 of *Lecture Notes in Computer Science*, pages 176–187. Springer, 2002a.

L. Bianchi, L.M. Gambardella, and M. Dorigo. An ant colony optimization approach to the probabilistic traveling salesman problem. In J.J.M. Guervós, P. Adamidis, H.G. Beyer, H.P. Schwefel, and José-Luis Fernández-Villacañas, editors, *Parallel Problem Solving from Nature–PPSN VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 883–892. Springer, 2002b.

L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto. Metaheuristics for the vehicle routing problem with stochastic demands. In *Parallel Problem Solving from Nature-PPSN VIII*, pages 450–460. Springer, 2004.

L. Bianchi, J. Knowles, and N. Bowler. Local search for the probabilistic traveling salesman problem: correction to the 2-p-opt and 1-shift algorithms. *European Journal of Operational Research*, 162(1):206–219, 2005.

L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto. Hybrid metaheuristics for the vehicle routing problem with stochastic demands. *Journal of Mathematical Modelling and Algorithms*, 5(1):91–110, 2006a.

L. Bianchi, M. Dorigo, L.M. Gambardella, and W.J. Gutjahr. Metaheuristics in stochastic combinatorial optimization: a survey. *TechReport: Dalle Molle Institute for Artificial Intelligence*, 2006b.

L. Bianchi, M. Dorigo, L.M. Gambardella, and W.J. Gutjahr. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing: an international journal*, 8(2):239–287, 2009.

M. Birattari, P. Balaprakash, and M. Dorigo. ACO/F-Race: Ant colony optimization and racing techniques for combinatorial optimization under uncertainty. In *MIC 2005: The 6th Metaheuristics International Conference*, pages 107–112, 2005.

M. Birattari, P. Balaprakash, and M. Dorigo. The ACO/F-RACE algorithm for combinatorial optimization under uncertainty. In *Metaheuristics*, pages 189–203. Springer, 2007.

M. Birattari, P. Balaprakash, Th. Stützle, and M. Dorigo. Estimation-based local search for stochastic combinatorial optimization using delta evaluations: a case study on the probabilistic traveling salesman problem. *INFORMS Journal on Computing*, 20(4):644–658, 2008.

B. Bouzaïene-Ayari, M. Dror, and G. Laporte. Vehicle routing with stochastic demand and split deliveries. *Foundations of computing and decision sciences*, 18: 63–69, 1992.

N.E. Bowler, Th.M.A. Fink, and R.C. Ball. Characterization of the probabilistic traveling salesman problem. *Physical Review E*, 68(3):036703, 2003.

J. Branke and M. Guntsch. New ideas for applying ant colony optimization to the probabilistic TSP. In *Applications of Evolutionary Computing*, pages 165–175. Springer, 2003.

J. Branke and M. Guntsch. Solving the probabilistic tsp with ant colony optimization. *Journal of Mathematical Modelling and Algorithms*, 3(4):403–425, 2004.

E.K Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and R. Qu. Hyper-heuristics: a survey of the state of the art. *Journal of the Operational Research Society*, 64(12):1695–1724, 2013.

G. Cabrera, S. Roncagliolo, J.P. Riquelme, C. Cubillos, and R. Soto. A hybrid particle swarm optimization-simulated annealing algorithm for the probabilistic travelling salesman problem. *Studies in Informatics and Control*, 21(1):49–58, 2012.

A.M. Campbell. Aggregation for the probabilistic traveling salesman problem. *Computers & Operations Research*, 33(9):2703–2724, 2006.

A.M. Campbell and B.W. Thomas. Challenges and advances in a priori routing. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 123–142. Springer, 2008a.

A.M. Campbell and B.W. Thomas. Probabilistic traveling salesman problem with deadlines. *Transportation Science*, 42(1):1–21, 2008b.

R.L Carraway, Th.L. Morin, and H. Moskowitz. Generalized dynamic programming for stochastic combinatorial optimization. *Operations Research*, 37(5):819–829, 1989.

K. Chaklevitch and P. Cowling. Hyperheuristics: Recent developments. In C. Cotta, M. Sevaux, and K. Sörensen, editors, *Adaptive and Multilevel Metaheuristics*, volume 136 of *Studies in Computational Intelligence*, pages 3–29. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-79437-0.

T.S. Chang, Y.W. Wan, and W.T Ooi. A stochastic dynamic traveling salesman problem with hard time windows. *European Journal of Operational Research*, 198(3):748–759, 2009.

K. Chepuri and T. Homem-de Mello. Solving the vehicle routing problem with stochastic demands using the cross-entropy method. *Annals of Operations Research*, 134(1):153–181, 2005.

U. Cherubini, E. Luciano, and W. Vecchiato. *Copula methods in finance*. John Wiley & Sons, 2004.

Ph. Chervi. *A computational approach to probabilistic vehicle routing problems*. PhD thesis, Massachusetts Institute of Technology, 1990.

C.H. Christiansen and J. Lysgaard. A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, 35 (6):773–781, 2007.

C.H. Christiansen, J. Lysgaard, and S. Wøhlk. A branch-and-price algorithm for the capacitated arc routing problem with stochastic demands. *Operations Research Letters*, 37(6):392–398, 2009.

N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976.

G. Clarke and J.W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

David G. Clayton. A model for association in bivariate life tables and its application in epidemiological studies of familial tendency in chronic disease incidence. *Biometrika*, 65(1):141–151, 1978.

J. Cordeau, M. Gendreau, G. Laporte, J. Potvin, and F. Semet. A guide to vehicle routing heuristics. *Journal of the Operational Research society*, 53(5):512–522, 2002.

J.F. Cordeau, G. Laporte, M.W.P. Savelsbergh, and D. Vigo. Vehicle routing. In C. Barnhart and G. Laporte, editors, *Handbook in Operations Research & Management Science*, volume 14, pages 367–428. Elsevier B.V., 2007.

P. Cowling, G. Kendall, and E. Soubeiga. A hyperheuristic approach to scheduling a sales summit. In *International Conference on the Practice and Theory of Automated Timetabling*, pages 176–190. Springer, 2000.

D.R. Cox. The analysis of multivariate binary data. *Applied statistics*, pages 113–120, 1972.

Bin Dai, Shilin Ding, and Grace Wahba. Multivariate bernoulli distribution. *Bernoulli*, 19(4):1465–1483, 2013.

J.R. Dale. Global cross-ratio models for bivariate, discrete, ordered responses. *Biometrics*, pages 909–917, 1986.

G.B. Dantzig. Linear programming under uncertainty. *Science*, 1955.

G.B. Dantzig and J.H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

G.B. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4):393–410, 1954.

L. Declerck, M. Aerts, and G. Molenberghs. Behaviour of the likelihood ratio test statistic under a bahadur model for exchangeable binary data. *Journal of Statistical Computation and Simulation*, 61(1-2):15–38, 1998.

M. Dorigo and L.M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

M. Dror. Modeling vehicle routing with uncertain demands as a stochastic program: Properties of the corresponding solution. *European Journal of Operational Research*, 64(3):432–441, 1993.

M. Dror and P. Trudeau. Stochastic vehicle routing with modified savings algorithm. *European Journal of Operational Research*, 23(2):228–235, 1986.

Paul Embrechts, Alexander McNeil, and Daniel Straumann. Correlation and dependence in risk management: properties and pitfalls. *Risk management: value at risk and beyond*, 176223, 2002.

F. Errico, G. Desaulniers, M. Gendreau, W. Rei, and L.M. Rousseau. The vehicle routing problem with hard time windows and stochastic service times. *EURO Journal on Transportation and Logistics*, 2016.

Th.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133, 1995.

G.M. Fitzmaurice and N.M Laird. A likelihood-based method for analysing longitudinal binary responses. *Biometrika*, 80(1):141–151, 1993.

G.M. Fitzmaurice, N.M. Laird, and A.G. Rotnitzky. Regression models for discrete longitudinal responses. *Statistical Science*, pages 284–299, 1993.

M. Fréchet. Sur les tableaux de corrélation dont les marges sont dollnes. *Annales de l'Université de Lyon*, 1951.

L.M. Gambardella, R. Montemanni, and D. Weyland. Coupling ant colony systems with strong local searches. *European Journal of Operational Research*, 220(3): 831 – 843, 2012.

P. Garrido and C. Castro. Stable solving of CVRPs using hyperheuristics. In F. Rothlauf, editor, *Genetic and Evolutionary Computation Conference*, pages 255–262. ACM: Montreal, 2009.

P. Garrido and M.C. Riff. An evolutionary hyperheuristic to solve strip-packing problems. In H. Yin, P. Tino, E. Corchado, W. Byrne, and X. Yao, editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2007*, volume 4881 of *Lecture Notes in Computer Science*, pages 406–415. Springer, 2010.

C. Gauvin, G. Desaulniers, and M. Gendreau. A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 50:141 – 153, 2014.

S.B. Gelfand and S.K. Mitter. Simulated annealing with noisy or imprecise energy measurements. *Journal of Optimization Theory and Applications*, 62(1):49–62, 1989.

M. Gendreau and J. Potvin. *Handbook of metaheuristics*, volume 2. Springer, 2010.

M. Gendreau, G. Laporte, and R. Séguin. An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation Science*, 29(2): 143–155, 1995.

M. Gendreau, G. Laporte, and R. Séguin. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477, 1996a.

M. Gendreau, O. Jabali, and W. Rei. Stochastic vehicle routing problems. In *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, chapter 8, pages 213–239. SIAM, 2014.

M. Gendreau, O. Jabali, and W. Rei. 50th anniversary invited article—future research directions in stochastic vehicle routing. *Transportation Science*, 50(4): 1163–1173, 2016.

M.l Gendreau, G. Laporte, and R. Séguin. Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12, 1996b.

C. Genest and J. Nešlehová. A primer on copulas for count data. *ASTIN Bulletin*, 37(2):475–515, 2007.

G. Ghiani, E. Manni, and B.W. Thomas. A comparison of anticipatory algorithms for the dynamic and stochastic traveling salesman problem. *Transportation Science*, 46(3):374–387, 2012.

F. Glover. Tabu search—part i. *ORSA Journal on Computing*, 1(3):190–206, 1989.

B.L Golden and W.R. Stewart. Vehicle routing with probabilistic demands. In *Computer Science and Statistics: Tenth Annual Symposium on the Interface, NBS Special Publication*, volume 503, pages 252–259, 1978.

B.L. Golden and Y.R. Yee. A framework for probabilistic vehicle routing. *AIIE Transactions*, 11(2):109–112, 1979.

B.L. Golden, S. Raghavan, and E.A. Wasil. *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43. Springer, 2008.

Emil J. Gumbel. Bivariate exponential distributions. *Journal of the American Statistical Association*, 55(292):698–707, 1960.

W.J. Gutjahr. A converging aco algorithm for stochastic combinatorial optimization. In *Stochastic algorithms: Foundations and applications*, pages 10–25. Springer, 2003.

W.J. Gutjahr. S-aco: An ant-based approach to combinatorial optimization under uncertainty. In *Ant colony optimization and swarm intelligence*, pages 238–249. Springer, 2004.

W.J. Gutjahr and G.Ch. Pflug. Simulated annealing for noisy cost functions. *Journal of Global Optimization*, 8(1):1–13, 1996.

W.J. Gutjahr, S. Katzensteiner, and P. Reiter. A vns algorithm for noisy problems and its application to project portfolio analysis. In *Stochastic Algorithms: Foundations and Applications*, pages 93–104. Springer, 2007.

V.A. Hajivassiliou and P.A. Ruud. Classical estimation methods for ldv models using simulation. *Handbook of econometrics*, 4:2383–2441, 1994.

D. Haugland, S.C. Ho, and G. Laporte. Designing delivery districts for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 180(3):997–1010, 2007.

A. Henchiri, M. Bellalouna, and W. Khaznaji. A probabilistic traveling salesman problem: a survey. *Annals of Computer Science and Information Systems: Position papers of the 2014 Federated Conference on Computer Science and Information Systems*, 3:55–60, 2014.

R. Hertwig and Th. Pachur. Heuristics, history of. In *International Encyclopedia of the Social & Behavioral Sciences*, pages 829–835. Elsevier, 2015.

C. Hjorring and J. Holt. New optimality cuts for a single-vehicle stochastic routing problem. *Annals of Operations Research*, 86:569–584, 1999.

S.C. Ho and D. Haugland. Local search heuristics for the probabilistic dial-a-ride problem. *OR Spectrum*, 33(4):961–988, October 2011.

J.H. Holland. *Adaption in natural and artificial systems*. The University of Michigan Press, 1975.

F. Hutter, H.H. Hoos, K. Leyton-Brown, and Th. Stützle. ParamILS: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36: 267–306, 2009.

L.M. Hvattum, A. Løkketangen, and G. Laporte. A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems. *Networks: An International Journal*, 49(4):330–340, 2007.

Irhamah and Z. Ismail. A breeder genetic algorithm for vehicle routing problem with stochastic demands. *Journal of Applied Sciences Research*, 5(11):1998–2005, 2009.

E. Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift fur Physik*, 31: 253–258, feb 1925.

Z. Ismail and Irhamah. Solving the vehicle routing problem with stochastic demands via hybrid genetic algorithm-tabu search. *Journal of Mathematics and Statistics*, 4(3):161, 2008.

O. Jabali, T. Van Woensel, A.G. De Kok, C. Lecluyse, and H. Peremans. Time-dependent vehicle routing subject to time delay perturbations. *Iie Transactions*, 41(12):1049–1066, 2009.

O. Jabali, M. Gendreau, and G. Laporte. New valid inequalities for the multi-vehicle routing problem with stochastic demands. Technical Report CIRRELT-2012-58, CIRRELT, 2012.

P. Jaillet. *Probabilistic traveling salesman problems*. PhD thesis, Massachusetts Institute of Technology, 1985.

P. Jaillet. Stochastic routing problems. In *Stochastics in Combinatorial Optimization*, pages 197–213. Word Scientific, 1987.

P. Jaillet. A priori solution of a traveling salesman problem in which a random subset of the customers are visited. *Operations research*, 36(6):929–936, 1988.

P. Jaillet. Analysis of probabilistic combinatorial optimization problems in euclidean spaces. *Mathematics of Operations Research*, 18(1):51–70, 1993.

P. Jaillet and A.R. Odoni. The probabilistic vehicle routing problem. In B.L. Golden and A.A. Assad, editors, *Vehicle routing: methods and studies*, pages 293–318. Elsevier North Holland, Amsterdam, 1988.

Patrick Jaillet, Jin Qi, and Melvyn Sim. Routing optimization under uncertainty. *Operations Research*, 64(1):186–200, 2016.

A. Jézéquel. Probabilistic vehicle routing problems. Master's thesis, Massachusetts Institute of Technology, 1985.

H. Joe. *Multivariate models and multivariate dependence concepts*. CRC Press, 1997.

M. Joerss, J. Schröder, F. Neuhaus, C. Klink, and F. Mann. Parcel delivery: The future of the last mile. *Travel Transport and Logistics*, 2016.

D.S. Johnson, L.A. McGeoch, C. Rego, and F. Glover. 8th dimacs implementation challenge: The traveling salesman problem, 2001. URL http://dimacs.rutgers.edu/archive/Challenges/TSP/. [Retrieved 24 July 2018].

H. Jula, M. Dessouky, and P.A. Ioannou. Truck route planning in nonstationary stochastic networks with time windows at customer locations. *Intelligent Transportation Systems, IEEE Transactions on*, 7(1):51–62, 2006.

E.P. Kao. A preference order dynamic program for a stochastic traveling salesman problem. *Operations Research*, 26(6):1033–1045, 1978.

R.E. Keller and R. Poli. Cost-benefit investigation of a genetic-programming hyperheuristic. In N. Monmarche, E.G. Talbi, P. Collet, M. Schoenauer, and E. Lutton, editors, *International Conference on Artificial Evolution*, pages 13–24. Springer-Verlag: Berlin, Heidelberg, 2007a.

R.E. Keller and R. Poli. Linear genetic programming of parsimonious metaheuristics. In *IEEE Congress on Evolutionary Computation*, pages 4508–4515. IEEE: Singapore, 2007b.

R.E. Keller and R. Poli. Self-adaptive hyperheuristic and greedy search. In *IEEE World Congress on Computational Intelligence*, pages 3801–3808. IEEE: Hong Kong, 2008a.

R.E. Keller and R. Poli. Subheuristic search and scalability in a hyperheuristic. In *Genetic and Evolutionary Computation Conference*, pages 609–610. ACM: Atlanta, 2008b.

R.E. Keller and R. Poli. Toward subheuristic search. In *In: IEEE World Congress on Computational Intelligence*, pages 3148–3155. IEEE: Hong Kong, 2008c.

A.S. Kenyon and D.P. Morton. Stochastic vehicle routing with random travel times. *Transportation Science*, 37(1):69–82, 2003.

S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simmulated annealing. *Science*, 220(4598):671–680, 1983.

T. Kohonen. *Self-organizing maps*, volume 30. Springer, 2001.

Joost Kruis and Gunter Maris. Three representations of the ising model. *Scientific reports*, 6:srep34175, 2016.

V. Lambert, G. Laporte, and F. Louveaux. Designing collection routes through bank branches. *Computers & Operations Research*, 20(7):783–791, 1993.

G. Laporte and F.V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Operations research letters*, 13(3):133–142, 1993.

G. Laporte and F.V. Louveaux. *Solving stochastic routing problems with the integer L-shaped method*. Springer, 1998.

G. Laporte and U. Palekar. Some applications of the clustered travelling salesman problem. *Journal of the Operational Research Society*, 53(9):972–976, 2002.

G. Laporte, F.M. Louveaux, and H. Mercure. Models and exact solutions for a class of stochastic location-routing problems. *European Journal of Operational Research*, 39(1):71–78, 1989.

G. Laporte, F. Louveaux, and H. Mercure. The vehicle routing problem with stochastic travel times. *Transportation science*, 26(3):161–170, 1992.

G. Laporte, F.V. Louveaux, and H. Mercure. A priori optimization of the probabilistic traveling salesman problem. *Operations research*, 42(3):543–549, May–June 1994.

G. Laporte, F.V. Louveaux, and L. Van Hamme. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423, 2002.

A.J. Lee. Generating random binary deviates having fixed marginal distributions and specified degrees of association. *The American Statistician*, 47(3):209–215, 1993.

C. Lee, K. Lee, and S. Park. Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *Journal of the Operational Research Society*, 63 (9):1294–1306, 2012.

H. Lei, G. Laporte, and B. Guo. A generalized variable neighborhood search heuristic for the capacitated vehicle routing problem with stochastic service times. *Top*, 20(1):99–118, 2012.

T. Leipälä. On the solutions of stochastic traveling salesman problems. *European Journal of Operational Research*, 2(4):291–297, 1978.

W. Li. Constructing a solution attractor for the probabilistic traveling salesman problem through simulation. In M. Emmerich, A. Deutz, O. Schütze, Th. Bäck, E. Tantar, A.A. Tantar, P.D. Moral, P. Legrand, P. Bouvry, and C.A. Coello, editors, *EVOLVE – A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, pages 1–15, Heidelberg, 2013. Springer International Publishing.

W. Li. A simulation-based algorithm for the probabilistic traveling salesman problem. In M. Emmerich, A. Deutz, O. Schütze, P. Legrand, P. Bouvry, E. Tantar, and A.A. Tantar, editors, *EVOLVE – A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation VII*, pages 157–183. Springer, 2017.

X. Li, P. Tian, and S.C.H. Leung. Vehicle routing problems with time windows and stochastic travel and service times: models and algorithm. *International Journal of Production Economics*, 125(1):137–145, 2010.

Kung-Yee Liang and Scott L Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73(1):13–22, 1986.

K.Y. Liang, S.L. Zeger, and B. Qaqish. Multivariate regression analyses for categorical data. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 3–40, 1992.

C. Lin, K.L. Choy, G.T.S. Ho, S.H. Chung, and H.Y. Lam. Survey of green vehicle routing problem: past and future trends. *Expert Systems with Applications*, 41 (4):1118–1138, 2014.

Stuart R Lipsitz, Nan M Laird, and David P Harrington. Generalized estimating equations for correlated binary data: using the odds ratio as a measure of association. *Biometrika*, 78(1):153–160, 1991.

Y. Liu. A hybrid scatter search for the probabilistic traveling salesman problem. *Computers & Operations Research*, 34(10):2949–2963, 2007.

Y. Liu. A memetic algorithm for the probabilistic traveling salesman problem. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 146–152. IEEE, 2008a.

Y. Liu. Solving the probabilistic traveling salesman problem based on genetic algorithm with queen selection scheme. In F. Greco, editor, *Travelling Salesman Problem*, pages 157–172. InTech, 2008b.

Y. Liu. Diversified local search strategy under scatter search framework for the probabilistic traveling salesman problem. *European Journal of Operational Research*, 191(2):332–346, 2008c.

Y.H. Liu. Different initial solution generators in genetic algorithms for solving the probabilistic traveling salesman problem. *Applied mathematics and computation*, 216(1):125–137, 2010.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

R.M. Losee. Term dependence: truncating the bahadur lazarsfeld expansion. *Information processing & management*, 30(2):293–303, 1994.

H.R. Lourenco, O.M. Martin, and Th. Stützle. Iterated local search: Framework and applications. In M. Gendreau and Y. Potvin, editors, *Handbook of metaheuristics*, volume 146, pages 363–397. Springer, 2 edition, 2010.

L. Lu and Q. Tan. Hybrid particle swarm optimization algorithm for stochastic vehicle routing problem. *Systems Engineering and Electronics*, 28(2):244–247, 2006.

M. Lu, D. Wu, and J. Zhang. A particle swarm optimization-based approach to tackling simulation optimization of stochastic, large-scale and complex systems. In *Advances in Machine Learning and Cybernetics*, pages 528–537. Springer, 2006.

X. Lu. *Dynamic and stochastic routing optimization: algorithm development and analysis*. PhD thesis, University of California, Irvine, 2001.

S.S. Mahfoudh, W. Khaznaji, and M. Bellalouna. A branch and bound algorithm for the porbabilistic traveling salesman problem. In *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 1–6. IEEE, 2015.

Hosam Mahmoud. *Pólya urn models*. CRC press, 2008.

K.L. Mak and Z.G. Guo. A genetic algorithm for vehicle routing problems with stochastic demand and soft time windows. In *Systems and Information Engineering Design Symposium, 2004. Proceedings of the 2004 IEEE*, pages 183–190. IEEE, 2004.

Y. Marinakis and M. Marinaki. A hybrid honey bees mating optimization algorithm for the probabilistic traveling salesman problem. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pages 1762–1769. IEEE, 2009.

Y. Marinakis and M. Marinaki. A hybrid multi-swarm particle swarm optimization algorithm for the probabilistic traveling salesman problem. *Computers & Operations Research*, 37(3):432–442, 2010.

Y Marinakis, A. Migdalas, and P.M. Pardalos. Expanding neighborhood grasp for the traveling salesman problem. *Computational Optimization and Applications*, 32(3):231–257, 2005.

Y. Marinakis, A. Migdalas, and P.M. Pardalos. Expanding neighborhood search–GRASP for the probabilistic traveling salesman problem. *Optimization Letters*, 2(3):351–361, 2008.

Y. Marinakis, G.R. Iordanidou, and M. Marinaki. Particle swarm optimization for the vehicle routing problem with stochastic demands. *Applied Soft Computing*, 13(4):1693–1704, 2013.

Y. Marinakis, M. Marinaki, and A. Migdalas. Adaptive tunning of all parameters in a multi-swarm particle swarm optimization algorithm: An application to the probabilistic traveling salesman problem. In A. Migdalas and A. Karakitsiou, editors, *Optimization, Control, and Applications in the Information Age*, pages 187–207. Springer, 2015a.

Y. Marinakis, M. Marinaki, and P. Spanou. A memetic differential evolution algorithm for the vehicle routing problem with stochastic demands. In I. Fister and I. Fister Jr., editors, *Adaptation and Hybridization in Computational Intelligence*, pages 185–204. Springer, 2015b.

Y. Marinakis, M. Marinaki, and A. Migdalas. An adaptive bumble bees mating optimization algorithm. *Applied Soft Computing*, 55:13–30, 2017.

P. McCullagh and John A. Nelder. *Generalized Linear Models*, volume 37. CRC Press, 1989.

J.F. Meier and W. Clausen. Solving the probabilistic traveling salesman problem by linearising a quadratic approximation. *Optimization Online*, 2015.

J.E. Mendoza, B. Castanier, C. Guéret, A.L. Medaglia, and N. Velasco. A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898, 2010.

J.E. Mendoza, B. Castanier, C. Guéret, A.L. Medaglia, and N. Velasco. Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45(3):346–363, 2011.

P. Merz and B. Freisleben. Memetic algorithms for the traveling salesman problem. *Complex Systems*, 13(4):297–346, 2001.

N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097 – 1100, 1997.

R.H. Mole and S.R. Jameson. A sequential route-building algorithm employing a generalised savings criterion. *Operational Research Quarterly*, pages 503–511, 1976.

G. Molenberghs and E. Lesaffre. Marginal modeling of correlated ordinal data using a multivariate plackett distribution. *Journal of the American Statistical Association*, 89(426):633–644, 1994.

G. Molenberghs and L.M. Ryan. An exponential family model for clustered multivariate binary data. *Environmetrics*, 10(3):279–300, 1999.

G. Molenberghs and G. Verbeke. *Models for Discrete Longitudinal Data*. Springer, 2005.

R.B. Nelsen. *An Introduction to Copulas*. Springer, 2 edition, 2006.

A.K. Nikoloulopoulos. Copula-based models for multivariate discrete response data. In *Copulae in Mathematical and Quantitative Finance*, chapter 11, pages 231–249. Springer, 2013.

C. Novoa and R. Storer. An approximate dynamic programming approach for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 196(2):509–515, 2009.

A. Panagiotelis, C. Czado, and H. Joe. Pair copula constructions for multivariate discrete data. *Journal of the American Statistical Association*, 107(499): 1063–1072, 2012.

Y. Peng and H. Zhu. Research on vehicle routing problem with stochastic demand and pso-dp algorithm with inver-over operator. *Systems Engineering-Theory & Practice*, 28(10):76–81, 2008.

V. Pillac, M. Gendreau, C. Guéret, and A.L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225:1–11, 2013.

D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers and Operations Research*, 34(8):2403–2435, 2007.

R.L. Plackett. A class of bivariate distributions. *Journal of the American Statistical Association*, 60(310):516–522, 1965.

George Pólya. How to solve it, 1945.

W.B. Powell, P. Jaillet, and A. Odoni. Stochastic and dynamic networks and routing. *Handbooks in operations research and management science*, 8:141–295, 1995.

R.L. Prentice. Binary regression using an extended beta-binomial distribution, with discussion of correlation induced by covariate measurement errors. *Journal of the American Statistical Association*, 81(394):321–327, 1986.

R.L. Prentice. Correlated binary regression with covariates specific to each binary observation. *Biometrics*, pages 1033–1048, 1988.

R. Qu and E.K. Burke. Hybridizations within a graph-based hyper-heuristic framework for university timetabling problems. *Journal of the Operational Research Society*, 60(9):1273–1285, 2009.

Walter Rei, Michel Gendreau, and Patrick Soriano. A hybrid monte carlo local branching algorithm for the single vehicle routing problem with stochastic demands. *Transportation Science*, 44(1):136–146, 2010.

D.J. Rosenkrantz, R.E. Stearns, and P.M. Lewis. Approximate algorithms for the traveling salesperson problem. In *Switching and Automata Theory, 1974., IEEE Conference Record of 15th Annual Symposium on*, pages 33–42. IEEE, 1974.

S. Rosenow. A heuristic for the probabilistic traveling salesman problem. In *Operations Research Proceedings 1996*, pages 117–122. Springer, 1997.

S. Rosenow. Comparison of an exact branch-and-bound and an approximative evolutionary algorithm for the probabilistic traveling salesman problem. In *Operations Research Proceedings 1998*, pages 168–174. Springer, 1999.

F. Rossi and I. Gavioli. Aspects of heuristic methods in the probabilistic traveling salesman problem. *Advanced school on stochastics in combinatorial optimization*, pages 214–227, 1987.

S. Samaranayake, S. Blandin, and A. Bayen. A tractable class of algorithms for reliable routing in stochastic networks. *Transportation Research Part C: Emerging Technologies*, 20(1):199–217, 2012.

N. Secomandi. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27 (11):1201–1225, 2000.

N. Secomandi. A rollout policy for the vehicle routing problem with stochastic demands. *Operations Research*, 49(5):796–802, 2001.

N. Secomandi. Analysis of a rollout approach to sequencing problems with stochastic routing applications. *Journal of Heuristics*, 9(4):321–352, 2003.

N. Secomandi and F. Margot. Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230, 2009.

N. Secomandi et al. *Exact and heuristic dynamic programming algorithms for the vehicle routing problem with stochastic demands*. PhD thesis, University of Houston, 1998.

R. Séguin. *Problemes stochastiques de tournées de véhicules*. PhD thesis, Centre de recherche sur les transports, Université de Montreal, 1994.

G. Shanmugam, P. Ganesan, and P.T. Vanathi. Meta heuristic algorithms for vehicle routing problem with stochastic demands. *Journal of Computer Science*, 7(4): 533, 2011.

Z. Shen, F. Ordónez, and M.M. Dessouky. The stochastic vehicle routing problem for minimum unmet demand. In *Optimization and logistics challenges in the enterprise*, pages 349–371. Springer, 2009.

W. Sierpiński. *Sur une nouvelle courbe continue qui remplit toute une aire plane*. Imprimerie de L'Université, 1912.

M Sklar. Fonctions de repartition an dimensions et leurs marges. *Publ. inst. statist. univ. Paris*, 8:229–231, 1959.

M.M. Smith and Y.S. Chen. A novel evolutionary algorithm for the homogeneous probabilistic traveling salesman problem. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pages 1–6, June 2016.

M.S. Smith and M.A. Khaled. Estimation of copula models with discrete margins via bayesian data augmentation. *Journal of the American Statistical Association*, 107(497):290–303, 2012.

M.S. Smith, Q. Gan, and R.J. Kohn. Modelling dependence using skew t copulas: Bayesian inference and applications. *Journal of Applied Econometrics*, 27(3): 500–522, 2012.

P.X.K. Song. *Correlated data analysis: modeling, analytics, and applications*. Springer Science & Business Media, 2007.

W.R. Stewart. *New algorithms for deterministic and stochastic vehicle routing problems*. PhD thesis, University of Maryland, 1981.

W.R. Stewart and B.L. Golden. A chance-constrained approach to the stochastic vehicle routing problem. In *1980 Northeast AIDS Conference*, pages 33–35, 1980.

W.R. Stewart and B.L. Golden. Stochastic vehicle routing: A comprehensive approach. *European Journal of Operational Research*, 14(4):371–385, 1983.

W.R. Stewart, B.L. Golden, and F. Gheysens. A survey of stochastic vehicle routing. In *Proceedings: 1982 IEEE International Large Scale Systems Symposium*, page 229. IEEE, 1982.

I. Sungur, F. Ordónez, and M. Dessouky. A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *Iie Transactions*, 40(5):509–523, 2008.

I. Sungur, Y. Ren, F. Ordóñez, M. Dessouky, and H. Zhong. A model and algorithm for the courier delivery problem with uncertainty. *Transportation science*, 44(2): 193–205, 2010.

K.C. Tan, C.Y. Cheong, and C.K. Goh. Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813–839, 2007.

H. Tang and E. Miller-Hooks. Approximate procedures for probabilistic traveling salesperson problem. *Transportation Research Record: Journal of the Transportation Research Board*, 1882(1):27–36, 2004.

H. Tang and E. Miller-Hooks. Solving a generalized traveling salesperson problem with stochastic customers. *Computers & operations research*, 34(7):1963–1987, 2007.

D. Taş, N. Dellaert, T. Van Woensel, and T. De Kok. Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research*, 40(1):214–224, 2013.

D. Taş, M. Gendreau, N. Dellaert, T. Van Woensel, and A.G. De Kok. Vehicle routing with soft time windows and stochastic travel times: A column generation and branch-and-price solution approach. *European Journal of Operational Research*, 236(3):789–799, 2014.

D. Teodorović and G. Pavković. A simulated annealing technique approach to the vehicle routing problem in the case of stochastic demand. *Transportation Planning and Technology*, 16(4):261–273, 1992.

J.L. Teugels. Some representations of the multivariate bernoulli and binomial distributions. *Journal of multivariate analysis*, 32(2):256–268, 1990.

F.A. Tillman. The multiple terminal delivery problem with probabilistic demands. *Transportation Science*, 3(3):192–204, 1969.

B. Verweij, S. Ahmed, A.J. Kleywegt, G. Nemhauser, and A. Shapiro. The sample average approximation method applied to stochastic routing problems: a computational study. *Computational Optimization and Applications*, 24(2-3): 289–333, 2003.

S.A. Voccia, A.M. Campbell, and B.W. Thomas. The probabilistic traveling salesman problem with time windows. *EURO Journal on Transportation and Logistics*, 2 (1-2):89–107, 2013.

C.D.J. Waters. Vehicle-scheduling problems with uncertainty and omitted customers. *Journal of the Operational Research Society*, pages 1099–1108, 1989.

C. Weiler, B. Biesinger, B. Hu, and G.R. Raidl. Heuristic approaches for the probabilistic traveling salesman problem. In *EUROCAST 2015: International Conference on Computer Aided Systems Theory*, pages 342–349. Springer, 2015.

D. Weyland, L. Bianchi, and L.M. Gambardella. New approximation-based local search algorithms for the probabilistic traveling salesman problem. In *Computer Aided Systems Theory-EUROCAST 2009*, pages 681–688. Springer, 2009a.

D. Weyland, L. Bianchi, and L.M. Gambardella. New heuristics for the probabilistic traveling salesman problem. In *Proceedings of the VIII metaheuristic international conference (MIC 2009)*, 2009b.

D. Weyland, R. Montemanni, and L.M. Gambardella. Using statistical tests for improving state-of-the-art heuristics for the probabilistic traveling salesman problem with deadlines. In *Computer Aided Systems Theory–EUROCAST 2011*, pages 448–455. Springer, 2012.

D. Weyland, R. Montemanni, and L.M. Gambardella. Heuristics for the probabilistic traveling salesman problem with deadlines based on quasi-parallel monte carlo sampling. *Computers & Operations Research*, 40(7):1661–1670, 2013a.

D. Weyland, R. Montemanni, and L.M. Gambardella. A metaheuristic framework for stochastic combinatorial optimization problems based on GPGPU with a case study on the probabilistic traveling salesman problem with deadlines. *Journal of Parallel and Distributed Computing*, 73(1):74–85, 2013b.

D. Weyland, R. Montemanni, and L.M. Gambardella. An enhanced ant colony system for the probabilistic traveling salesman problem. In G.A. Di Caro and G. Theraulaz, editors, *Bio-Inspired Models of Network, Information, and Computing Systems. BIONETICS 2012. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, volume 134, pages 237–249. Springer, 2014.

R. Wright. UPS and FedEx invest in technology to meet soaring demand, August 2014. URL https://www.ft.com/content/0123be4c-1d78-11e4-8b03-00144feabdc0. [Accessed 10 July 2018].

W.H. Yang, K. Mathur, and R.H. Ballou. Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(1):99–112, 2000.

Y. Yoshitomi, H. Ikenoue, T. Takeba, and S. Tomita. Genetic algorithm in uncertain environments for solving stochastic programming problem. *Journal of the Operations Research Society of Japan-Keiei Kagaku*, 43(2):266–290, 2000.

L. Zhang and L. Wang. Genetic ordinal optimization for stochastic traveling salesman problem. In *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on*, volume 3, pages 2086–2090. IEEE, 2004.

M. Zhang, J. Qin, Y. Yugang, and L. Liang. Traveling salesman problems with profits and stochastic customers. *International Transactions in Operational Research*, 25 (4):1297–1313, 2018.

T. Zhang, W.A. Chaovalitwongse, and Y. Zhang. Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. *Computers & Operations Research*, 39(10):2277–2290, 2012.

L.P. Zhao and R.L. Prentice. Correlated binary regression using a quadratic exponential model. *Biometrika*, 77(3):642–648, 1990.

P. Zhao. Improved particle swarm optimization algorithm for the stochastic loader problem. In *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pages 773–776. IEEE, 2007.