Jorge Lusi Oyola Mendoza

Essays on stochastic and multi-objective capacitated vehicle routing problems

Molde University College
Specialized University in Logistics

PhD theses in Logistics 2016:2

Essays on stochastic and multi-objective capacitated vehicle routing problems

Jorge Luis Oyola Mendoza

A dissertation submitted to Molde University College -
Specialized University in Logistics
for the degree of Philosophiae Doctor

PhD theses in Logistics 2016:2

Molde University College - Specialized University in Logistics

Molde, Norway 2016

Jorge Luis Oyola Mendoza

Essays on stochastic and multi-objective capacitated vehicle routing problems

2016

PhD theses in Logistics 2016:2

Molde University College - Specialized University in Logistics
P.O.Box 2110
NO-6402 Molde, Norway
www.himolde.no

This dissertation can be ordered from Molde University College Library
biblioteket@himolde.no

Printing Molde University College

ISBN: 978-82-7962-209-3
ISSN: 0809-9588

Preface

This document is submitted as partial fulfillment of the requirements for the degree of *Philosophiae Doctor* (PhD) in Logistics at Molde University College - Specialized University in Logistics, Molde, Norway.

This work was conducted from September 2010 until February 2016. Late Professor Arne Løkketangen from Molde University College was the main supervisor until June 2013. Associate Professor Halvard Arntzen from Molde University College was co-supervisor during this period. In June 2013 Professor Arntzen was appointed as main supervisor and Professor David L. Woodruff from University of California, Davis was appointed as co-supervisor.

The main subject of this thesis is multi-objective vehicle routing problems. Both deterministic and stochastic versions of such problems have been studied. Routing planning is considered to be a multi-criteria process, therefore optimizing several meaningful objectives simultaneously may provide decision makers with better evidence to support their choices.

The work presented in this PhD thesis has been evaluated by a committee consisting of Professor Rafael Martí from University of Valencia, Spain, Associate Professor Sin C. Ho from Aarhus University, Denmark, and Associate Professor Arild Hoff from Molde University College - Specialized University in Logistics, Norway.

Acknowledgments

First and foremost I would like to thank my previous supervisor, late Professor Arne Løkketangen. Perhaps he was even more confident than me in the eventual success of this work. He gave me the opportunity to work with him during my master degree program and later during the Phd program. The experience of working with him was invaluable. I am equally thankful to my current supervisor, Associate Professor Halvard Artnzen, for his patience suggestions, constructive criticisms and quest for precision. I owe many thanks to Professor David L. Woodruff whose guidance and encouragement had an unquestionable impact in the completion of this work.

I am also grateful to the library, IT center, faculty members and administrative personnel at Molde University College. The *de facto* coordinator, Irene Sætre, Jens Erik Østergaard, Rickard Romfo, Bente Lindset and Johan Oppen in particular deserve to be mentioned.

I will always appreciate the support received from the Jaime Benítez Tobón Foundation. This achievement still is a result of that support and I will never be able to pay back.

I also wish to thank all who had the (mis)fortune of being part of the A288 club, including the latest members, Kiros, Evellyn and Pipe. But specially to the great team of 2014-2015: Primo, Poncho, Chapulín and The Sundance Kid (aka Butch Cassidy).

I met many people during all this years in Molde. I would like to thank all of them, specially those who despite the fact of knowing me better somehow became my friends. A special mention goes to Yuri Redutskiy, Sergei Teryokhin, Jianyong Jin, Øivind Opdal and Bella Nujen.

In the last stage of this process, my supervisor had limited availability. I wish to extend many thanks to Aksel for, sometimes reluctantly, allowing stochastic time windows for Associate Professor Artnzen to supervise my progress.

Special Thanks go to Jana Hajasova, her unconditional support and friendship are very much appreciated. I would also like to thank *people* who were there for me, even when I was unable to do the same.

Finally I would like to thank Delia Barros for always trying to see the best of me. To my friends and family who even at the distance have made me feel close to them. And to Lidis, of course.

Molde, Norway
April, 2016

Jorge Luis Oyola Mendoza

Contents

Introduction

Introduction

The vehicle routing problem (VRP) is one of the most studied subjects in Operations Research (Braekers et al., 2016). Such interest may be due to its wide range of applications in solving real-life problems. Examples of such applications are: milk collection, milk delivery, distribution of ready-made concrete, furniture distribution, city logistics, retail distribution, green logistics (Koç et al., 2016); cash transportation, garbage collection, social legislation for drivers' working hours, school bus routing, shipment of hazardous material (Braekers et al., 2016); allocation of work-force, vendor-managed distribution systems, courier service, emergency service, taxi cab service, after-sales service, e-commerce (Lin et al., 2014); livestock collection (Oppen and Løkketangen, 2008); beer, wine, and spirits distribution (Erera et al., 2009); road network monitoring (Chen et al., 2014).

The vehicle routing problem (VRP) was proposed in 1959 by Dantzig and Ramser (1959) as a generalization of the traveling salesman problem, though the name used back then was "the truck dispatching problem". Due to the shape of a solution to the problem, the name "clover leaf problem" was also suggested. The term "vehicle routing" did not appear in the literature until the early 1970s (Eksioglu et al., 2009).

The theme of this thesis is the development of solution methods for different versions of stochastic and multi-objective VRPs. The following parts of this introduction define the general concepts used in the thesis, the scientific contribution of this research, the summary of papers that compose it and suggest avenues for further research.

The Vehicle Routing Problem

Different versions of the vehicle routing problem have been proposed, such as the basic version of the capacitated vehicle routing problem (CVRP) (Toth and Vigo, 2002a). Various authors have defined this version of the problem, e.g. Toth and Vigo (1998) and Cordeau et al. (2002), who begin with an undirected graph $G = \{V, E\}$, where $V = \{v_0, v_1, \ldots, v_n\}$ is the vertex set, and $E = \{(v_i, v_j) : v_i, v_j \in V, i < j\}$ is the edge set. v_0 represents the depot, and the other vertices represent the customers, each having a non-negative demand, q_i. The set E has an associated cost matrix c_{ij}, representing the cost of traveling from vertex i to vertex j and $c_{ij} = c_{ji}$, in the symmetric case. A fleet of m vehicles with equal capacity Q is based at the depot. The optimal solution to the problem is the one that minimizes the total routing cost, the total demand of the customers in one route is not greater than Q, every customer is visited once by just one vehicle, and each tour includes the depot.

Several other versions can be found, depending on considerations and/or constraints included in the problem, e.g. the vehicle routing problem with time windows (VRPTW), vehicle routing problem with pickups and deliveries (VRPPD), vehicle routing problem with backhauling (VRPB) and vehicle routing problem with distance constraints (DCVRP) (Toth and Vigo, 2002a).

Multi-objective optimization

More than two thousands years ago Sun Tzu considered planning to be a multi-criteria process, at least in the context of war (Sawyer and Sawyer, 1994). Such philosophy initially applied to warfare has been applied to other areas, such as business and politics, for several decades (Dimovski et al., 2012). In particular, it is considered that transportation planning is intrinsically a multi-objective decision process (Current and Min, 1986). Therefore considering several meaningful objectives simultaneously may lead to better plans.

In a multi-objective optimization problem several functions are optimized (minimized or maximized) subject to the same set of constraints. Without loss of generality, it can be assumed that all objective function are minimized. This problem then can be stated as

$$\min\ F(x) = (f_1(x), f_2(x), \ldots, f_n(x)),\ s.t.\ x \in D, \tag{1}$$

with the number of objective functions being $n \geq 2$; the decision variable vector $x = (x_1, x_2, \ldots, x_r)$; the feasible solution space D; and $F(x)$ is the objective vector (Jozefowiez et al., 2007).

The development of mathematical programming by Kantorovich and Dantzig, in the 1930s and 1940s respectively, provided the right environment for the establishment of multi-objective mathematical programming. Even though mathematical programming does not directly solve multi-objective optimization problems, it was proposed to use weights to combine the objective functions into a weighted single objective function (Köksalan et al., 2011). The multi-objective vehicle routing problem started to gain popularity in the 1980s (Jozefowiez et al., 2008).

The different objective functions in a multi-objective problem are usually conflicting. Because of this, it is common that no single optimal solution is able to minimize all the objectives of the problem. Therefore instead of a single solution, a whole set is found. The objective function values of such set of solutions represent a gradual priority shifting from one of the objectives to the other(s), these solutions are called tradeoff solutions (Collette and Siarry, 2003) or Pareto optimal solutions (Jozefowiez et al., 2007).

The Pareto optimal solutions are defined using the concept of domination. A solution y evaluated in the objective as $F(y) = (f_1(y), f_2(y), \ldots, f_n(y))$, dominates a solution $F(z) = (f_1(z), f_2(z), \ldots, f_n(z))$, if and only if $\forall\ i\ \{1, 2, \ldots, n\}\ f_i(y) \leq f_i(z)$, and $\exists\ j\ \{1, 2, \ldots, n\}$, such that $f_j(y) < f_j(z)$. Which means that the solution z has not a better performance than y in any objective functions, but it has a worse performance in at least one. The set of non-dominated solutions, Pareto set or Pareto optimal solutions define the solution of a multi-objective optimization problem (Jozefowiez et al., 2007).

When the method used for solving the optimization problem does not guarantee an optimum solution, one more concept should be included, the potentially Pareto optimal (PPS). A solution y found by a particular algorithm A is considered potentially Pareto optimal, relative to A, if that algorithm does not find a different solution z that dominates y (Jozefowiez et al., 2007).

A non-dominated solution y is also called efficient (Raith and Ehrgott, 2009). There are two types of efficient solutions:

- Supported efficient solutions, which can be obtained as the optimum of the single objective optimization problem

$$\min_{x \in D} \lambda_1 f_1(x) + \lambda_2 f_2(x) + \ldots + \lambda_n f_n(x) \tag{2}$$

Where $\lambda_1, \lambda_2, \ldots, \lambda_n > 0$. The supported efficient solutions are located in the convex hull of the feasible set in objective space.

- Non-supported efficient solutions are located in the interior of the convex hull of the feasible set in objective space. Because of that they can not be obtained as a solution of Equation (2)

A large number of methods have been proposed and used by different authors in order to solve multi-objective optimization problems (Collette and Siarry, 2003). These methods have been classified using different criteria or approaches (Collette and Siarry, 2003; Jozefowiez et al., 2008). According to Jozefowiez et al. (2008), the multi-objective optimization methods are classified as scalar, Pareto and non-scalar and non-Pareto methods. Scalar methods use a mathematical transformation in the process of solving the optimization problem. The Pareto methods use the concept of Pareto dominance to evaluate the quality of the solutions. Methods that consider each objective separately are classified as non-scalar and non-Pareto.

The Stochastic Vehicle Routing Problem

In some cases not all parameters are known when planning the routes. If some of the parameters are unknown and their values are revealed after the planning decisions are taken, then a different problem arises, the stochastic VRP (SVRP) (Gendreau et al., 1996a; Cordeau et al., 2007). Different stochastic parameters have been studied in the literature, including the presence of customers, the travel times, service times and the demands. The most common version of the SVRP is the one with stochastic demands (Gendreau et al., 1996a).

Stochastic problems can be modeled in two different ways: as a *stochastic program with recourse* (SPR) or as a *chance constrained program* (CCP) (Gendreau et al., 1996a). When the stochastic problem is modeled as CCP, it is ensured that the probability of failure (violation of stochastic constraint) is below a certain level. The cost of such failures is typically ignored (Gendreau et al., 1996a; Tan et al., 2007).

For the case of problems modeled as SPR, route failures are allowed, but the decision maker (DM) must define a *recourse* policy, describing the actions to take in order to repair the solution once a failure has occurred. The cost of the failure, measured as the expected cost of the recourse action, is included into the objective function. Compared to CCP, problems modeled as SPR are more difficult to solve, but objectives are more meaningful (Gendreau et al., 1996a).

The VRP is considered to be a difficult problem to solve (Laporte, 2009). Including stochastic parameters will make it even more difficult. Treating the problem in a multi-objective approach increases the complexity. This creates the need for meta-heuristic methods. It may also, at least partially, explain why the multi-objective approach of the SVRP has received little attention in the literature. A multi-objective approach CVRP with stochastic demands (CVRPSD) was presented in Tan et al. (2007), with stochastic travel times in Russell and Urban (2008), a location, allocation and routing under the risk of disruption in Ahmadi-Javid and Seddighi (2013). Even though the CVRPSD is not explicitly modeled as a multi-objective problem in Juan et al. (2011), the solutions are presented as a tradeoff between the total expected cost and the reliability.

Solution methods for VRPs

Different methods have been used to deal with the many variants of VRP. Some of these methods are listed below, for a more thorough description the reader is referred to Gendreau et al. (1996a); Toth and Vigo (1998, 2002b); Cordeau et al. (2002, 2007); Jozefowiez et al. (2008); Potvin (2009);

Laporte (2009); Gendreau et al. (2014); Oyola et al. (2015); Braekers et al. (2016) and Koç et al. (2016).

Branch-and-cut

A VRP with stochastic travel times is described in (Kenyon and Morton, 2003) and it is solved by using branch-and-cut. A similar approach is used in Adulyasak and Jaillet (2014) to solve the robust and the stochastic approach of the The CVRP with deadlines under travel time uncertainty. Reiter and Gutjahr (2012) uses branch-and-cut for solving the ϵ-constraint formulation of a bi-objective CVRP.

Branch-and-price

A branch-and-price algorithm was used in Christiansen and Lysgaard (2007) for the CVRPSD. The same problem was solved in Gauvin et al. (2014) using a a branch-cut-and-price algorithm. In Taş et al. (2014) a similar approach is used for solving the CVRP with soft time windows and stochastic travel times. A branch-and-price algorithm was also used for solving the robust CVRP with deadlines and travel time/demand uncertainty (Lee et al., 2012).

Integer L-Shaped method

In Hjorring and Holt (1999) and Rei et al. (2007) the L-shaped method was used to solve the single vehicle CVRPSD. It was also used in Laporte et al. (2002) to solve the CVRPSD. It was extended in Jabali et al. (2014) for solving the CVRPSD exactly.

Dynamic programming

A dynamic programming algorithm was used to solve the stochastic CVRP with optimal restocking in Yang et al. (2000). The single vehicle CVRPSD with reoptimization was also solved by using dynamic programming in Secomandi and Margot (2009)

Local search

Several versions of tabu search (Glover, 1989) were proposed in order to deal with VRPs e.g. Potvin et al. (1996); Gendreau et al. (1996b); Badeau et al. (1997); Taillard et al. (1997) and Cordeau et al. (1997). In Cordeau et al. (2001) a unified tabu search (UTS) heuristic was proposed for the CVRP with time windows. A tabu search heuristic for the CVRP with soft time windows and split deliveries was developed in Ho and Haugland (2004). A similar heuristic was used in Brandão (2006) for the CVRP with pickups and deliveries. Ho and Gendreau (2006) used a tabu search with path relinking for the CVRP with duration constraints. A multi-compartment CVRP was described in Oppen and Løkketangen (2008) where a heuristic based on UTS was applied. Hoff et al. (2009) used a tabu search heuristic to find solutions for the CVRP with pickups and deliveries, allowing a subset of costumers to be visited twice.

In Russell and Urban (2008) a tabu search is used to optimize the resulting weighted sum of the different objective functions in a multi-objective VRP with stochastic travel times. Another example of using tabu search in a multi-objective routing problem is found in Pacheco and Martí (2006), where it is used to find solutions to an ϵ-constraint formulation of a bi-objective school bus routing problem.

A tabu search heuristic was used to design delivery districts for the CVRPSD in Haugland et al. (2007). A tabu search procedure is included as part of the methodology used in Sungur et al.

(2010) to create the master plan to the VRP with soft time windows, stochastic service times and probabilistic customers. The CVRP with stochastic travel times, soft time windows and service costs, proposed in Taş et al. (2013), was also solved using a tabu search heuristic. In Li et al. (2010) was used to solve the CVRP with time windows and stochastic travel and service times. In Ak and Erera (2007) the VRPSD with Pair Locally coordinated (PLC) recourse was solved using the same heuristic. In Erera et al. (2010) it was used to find solutions for the vehicle routing problem with stochastic demands and duration constraints (VRPSD-DC).

An adaptive large neighborhood search heuristic was used to solve the CVRP with stochastic demand and time windows in Lei et al. (2011). Lei et al. (2012) proposed a generalized variable neighborhood search to solve the CVRP with stochastic service times.

A location-routing problem with disruption risk was solved using simulated annealing in (Ahmadi-Javid and Seddighi, 2013). The same technique was also used in Goodson et al. (2012) to find solutions to the CVRPSD and in Goodson (2015) to solve the multi-compartment vehicle routing problem with stochastic demands.

Evolutionary algorithms

Evolutionary algorithms have been extensively used for different variant of VRPs. A genetic algorithm was used in Potvin and Bengio (1996) for the CVRP with time windows. A genetic algorithm with new crossover operator was described in Prins (2004) for the CVRP. Tan et al. (2006) used an evolutionary algorithm combined with a local search which is performed at each generation to solve a bi-objective truck and trailer CVRP. In Jozefowiez et al. (2002, 2007, 2008) and Jozefowiez et al. (2009) evolutionary algorithms together with a local search are proposed for the CVRP with route balancing. In Gupta et al. (2010) genetic algorithm is used to deal with a multi-objective VRP with time windows. Four objectives are optimized, the fleet size, total length, average grade of customer satisfaction and total waiting time over the vehicles.

A multi-objective CVRPSD was solved using a evolutionary algorithm in Tan et al. (2007). Sörensen (2006) describes a vehicle routing problem where the objectives are to find a solution close to a given initial solution and minimize the total length. An evolutionary algorithm combined with a tabu search is used to solve this problem.

The multi-compartment CVRPSD was solved using an evolutionary algorithm combined with a local search (memetic algorithm) in Mendoza et al. (2010). A genetic algorithm is used in Ando and Taniguchi (2006) to solve the CVRP with soft time windows and stochastic travel times. In Zhang et al. (2012) a scatter search heuristic was used to solve the stochastic travel-time CVRP with simultaneous pick-ups and deliveries.

A memetic algorithm with population management was used in Sörensen and Sevaux (2009) to deal the CVRP with stochastic demands and travel cost, and the VRP with stochastic customers.

Other nature inspired heuristics

Szeto et al. (2011) uses an artificial bee colony algorithm (swarm-based) for the CVRP. Gambardella et al. (1999) deals with a bi-objective vehicle routing problem with time windows. The number of vehicles and the total travel time are minimized. An ant algorithm is used to solve it. The same type of algorithm is used in Bará and Schaerer (2003) for a vehicle routing problem with time windows, where three objectives are optimized: the number of vehicles, the total travel time (no including waiting time) and the total delivery time (including waiting time).

In Marinakis et al. (2013), The CVRPSD was solved by particle swarm optimization. The single vehicle CVRPSD was solved in Chepuri and Homem-de Mello (2005) using Cross Entropy (CE) method (Rubinstein, 1999).

Constructive heuristics

The multi-compartment CVRPSD was solved using three different constructive heuristics in Mendoza et al. (2011): A stochastic variant of the Clarke-Wright (Clarke and Wright, 1964) heuristic and two versions of a look-ahead heuristic (Voß et al., 2005).

In Juan et al. (2011), the CVRPSD was solved by a multi-start search procedure, combined with the Clarke-Wright heuristic. A multispace sampling heuristic is proposed by Mendoza and Villegas (2013) to find solutions to the CVRPSD. The same problem is solved in Mendoza et al. (2015) using a combination of GRASP (M.G.C. Resende, 2010) and a heuristic concentration (HC). A GRASP heuristic is also used in Villegas et al. (2011) for the truck and trailer routing problem.

Scientific contribution

The thesis consists of this introduction text and five scientific papers, which represent the contribution of this work. The problems considered in this thesis correspond to different versions of VRPs. In all cases where a VRP is studied, it is modeled using several objectives, becoming a multi-objective optimization problem (MOP).

In **paper 1** a deterministic multi-objective VRP is used to illustrate the application of a function that measures the structural differences between two solutions. A scalar method is used to transform the multi-objective problem into several single-objectives. A tabu search heuristic is applied to every single-objective problem.

A multi-objective algorithm is presented in **paper 2** to find solutions to a multi-objective VRP. A Pareto approach is used to deal with the MOP. Results are compared to others obtained from both scalar and Pareto approaches to solve the same problem.

Paper 3 outlines the main contributions done in the last 20 years in the field of SVRP. It describes the different types of problems and the solution methods used to solve them.

The problem from **paper 1** is extended in **paper 4**, modeling the VRP as multi-objective and stochastic. A Pareto approach is proposed to solve the problem and it is compared to a Pareto method initially proposed to deal with a similar problem.

Finally in **paper 5** a known multi-objective stochastic VRP (Russell and Urban, 2008) is further studied. A Pareto approach is used to deal with the problem and it is compared to the scalar approach applied when the problem was introduced.

Summary of the papers

The five papers that constitute this thesis are listed below. The contributions of all co-authors and/or supervisors is stated. If the paper has been published or presented in conferences or workshops, such event is also stated.

Paper 1 – An attribute based similarity function for VRP decision support

Paper 1 is a revised version of a paper included by Associate Professor Johan Oppen in his PhD thesis. Part of the work by the PhD candidate was done in 2010 in connection with a Master degree in logistics. The candidate was responsible for the extension of an existing code and implementing a multi-objective decision support for VRP. The computational experiments were also conducted by the candidate. In this paper, the structural differences of solutions to a bi-objective VRP are suggested to be used as decision criteria. The paper is published in *Decision Making in Manufacturing and Services*, Vol. 6(2), pages 6583, 2012. The paper was presented by the candidate at MIC 2011 (9th Metaheuristics International Conference), Udine, Italy, 2011.

Paper 2 – GRASP-ASP: An algorithm for the CVRP with route balancing

The basic idea of the algorithm in **paper 2** was a contribution of late Professor Arne Løkketangen. The implementation of the algorithm, the computational experiments as well as the writing was done by the PhD candidate. Associate Professor Halvard Arntzen together with Professor Dave L. Woodruff contributed to improve the writing as well as two anonymous referees. In this paper a novel multi-objective heuristic is proposed. The paper was published in *Journal of Heuristics*, Volume 20 (4), pages 361-382, 2014. Preliminary results of this research were presented by the candidate at EURO XXV (25th European Conference on Operational Research), Vilnius, Lithuania, 2012 and by Professor Arne Løkketangen at Optimization Days, Montreal, Canada, 2013.

Paper 3 – The stochastic vehicle routing problem, a literature review

Most of the reviewing of papers included in **paper 3** was done by the PhD candidate. Professor Dave L. Woodruff contributed with valuable suggestions on which papers should be included in the survey together with Associate Professor Halvard Arntzen. The three authors contributed with the writing process. An overview of the SVRP advances in the last 20 years is presented in the paper. It was submitted to *EURO Journal on Transportation and Logistics* and a revision has been recommended. It is currently available at *Optimization Online*, `http://www.optimization-online.org/DB_FILE/2016/01/5299.pdf`, 2015.

Paper 4 – The capacitated vehicle routing problem with route balancing and stochastic demand

The implementation of the algorithm used in **paper 4**, the computational experiments and most of the writing was done by the PhD candidate. Professor Dave L. Woodruff suggested topics that should be covered in the paper. Associate Professor Halvard Arntzen contributed with suggestions about the experiment design. Both Halvard Arntzen and Dave L. Woodruff contributed to improve the writing. In this paper a new version of a multi-objective stochastic VRP is described. Journal submission pending.

Paper 5 – The CVRP with soft time windows and stochastic travel times

The algorithm used in **paper 5** was implemented by the PhD candidate. The computational experiments and the writing was also a contribution of the PhD candidate. Associate Professor Halvard Arntzen contributed with suggestions about the usage of lookup tables and improving the construction of initial solutions and Professor Dave L. Woodruff suggested some characteristics of the algorithm. The final text was improved thanks to Halvard Arntzen as well as Dave L. Woodruff. A new solution approach to a multi-objective stochastic problem is used in this paper. Journal submission pending.

Further research

Given the scant literature that exists on multi-objective stochastic VRP, such a topic becomes an obvious choice for further scientific work. This is especially true because real-life problems are considered to be both stochastic and multi-criteria. The complexity of such problems calls for the use of algorithms able to find good solutions within reasonable computation time. Parallel algorithms warrant testing as an alternative for achieving that goal.

Bibliography

Adulyasak, Y. and Jaillet, P. (2014). Models and algorithms for stochastic and robust vehicle routing with deadlines. *Transportation Science, forthcoming*, pages –.

Ahmadi-Javid, A. and Seddighi, A. H. (2013). A location-routing problem with disruption risk. *Transportation Research Part E: Logistics and Transportation Review*, 53:63 – 82.

Ak, A. and Erera, A. L. (2007). A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation Science*, 41(2):222–237.

Ando, N. and Taniguchi, E. (2006). Travel time reliability in vehicle routing and scheduling with time windows. *Networks and Spatial Economics*, 6(3-4):293–311.

Badeau, P., Guertin, F., Gendreau, M., Potvin, J.-Y., and Taillard, E. (1997). A parallel tabu search heuristic for the vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 5(2):109–122.

Bará, B. and Schaerer, M. (2003). A multiobjective and colony system for vehicle routing problem with time windows. In *Proceedings of the 21st IASTED international Conference. APPLIED INFORMATICS*.

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

Brandão, J. (2006). A new tabu search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, 173(2):540–555.

Chen, L., H, M. H., Langevin, A., and Gendreau, M. (2014). Optimizing road network daily maintenance operations with stochastic service and travel times. *Transportation Research Part E: Logistics and Transportation Review*, 64:88–102.

Chepuri, K. and Homem-de Mello, T. (2005). Solving the vehicle routing problem with stochastic demands using the cross-entropy method. *Annals of Operations Research*, 134(1):153–181.

Christiansen, C. H. and Lysgaard, J. (2007). A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, 35(6):773 – 781.

Clarke, C. and Wright, J. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:432–441.

Collette, Y. and Siarry, P. (2003). *Multiobjective optimization: principles and case studies*. Springer,

Berlin.

Cordeau, J.-F., Gendreau, M., and Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30(2):105–119.

Cordeau, J.-F., Gendreau, M., Laporte, G., Potvin, J.-Y., and Semet, F. (2002). A guide to vehicle routing heuristics. *The Journal of the Operational Research Society*, 53(5):512–522.

Cordeau, J.-F., Laporte, G., and Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *The Journal of the Operational Research Society*, 52(8):928–936.

Cordeau, J.-F., Laporte, G., Savelsbergh, M. W., and Vigo, D. (2007). Chapter 6. Vehicle routing. In Barnhart, C. and Laporte, G., editors, *Transportation*, volume 14, pages 367–428. Elsevier.

Current, J. and Min, H. (1986). Multiobjective design of transportation networks: Taxonomy and annotation. *European Journal of Operational Research*, 26(2):187 – 201.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management Science (pre-1986)*, 6(1):80–91.

Dimovski, V., Maric, M., Uhan, M., Durica, N., and Ferjan, M. (2012). Sun tzu's "The Art of War" and implications for leadership: Theoretical discussion. *Organizacija*, 45(4):151–158.

Eksioglu, B., Vural, A. V., and Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4):1472–1483.

Erera, A. L., Morales, J. C., and Savelsbergh, M. (2010). The vehicle routing problem with stochastic demand and duration constraints. *Transportation Science*, 44(4):474–492.

Erera, A. L., Savelsbergh, M., and Uyar, E. (2009). Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints. *Networks*, 54(4):270–283.

Gambardella, E., Taillard, E., and Agazzi, G. (1999). Macs-vrptw: A multiple ant colony system for vehicle routing problems with time windows. In *New ideas in optimization. Corne, D., Dorigo, M. and Glover, F. (Eds)*. McGraw-Hill.

Gauvin, C., Desaulniers, G., and Gendreau, M. (2014). A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 50(0):141–153.

Gendreau, M., Jabali, O., and Rei, W. (2014). Stochastic vehicle routing problems. In Toth, P. and Vigo, D., editors, *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 213–239. Society for Industrial and Applied Mathematics.

Gendreau, M., Laporte, G., and Séguin, R. (1996a). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12.

Gendreau, M., Laporte, G., and Seguin, R. (1996b). A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477.

Glover, F. (1989). Tabu search - part I. *ORSA Journal on Computing*, 1:190–206.

Goodson, J. C. (2015). A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 241(2):361–369.

Goodson, J. C., Ohlmann, J. W., and Thomas, B. W. (2012). Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand. *European Journal of Operational Research*, 217(2):312 – 323.

Gupta, R., Singh, B., and Pandey, D. (2010). Multi-objective fuzzy vehicle routing problem: a case study. *International Journal of comtemporary mathematical sciences*, 5:1439–1454.

Haugland, D., Ho, S. C., and Laporte, G. (2007). Designing delivery districts for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 180(3):997–1010.

Hjorring, C. and Holt, J. (1999). New optimality cuts for a single-vehicle stochastic routing problem. *Annals of Operations Research*, 86:569–584.

Ho, S. and Haugland, D. (2004). A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. *Computers & Operations Research*, 31(12):1947–1964.

Ho, S. C. and Gendreau, M. (2006). Path relinking for the vehicle routing problem. *Journal of Heuristics*, 12(1-2):55–72.

Hoff, A., Gribkovskaia, I., Laporte, G., and Lkketangen, A. (2009). Lasso solution strategies for the vehicle routing problem with pickups and deliveries. *European Journal of Operational Research*, 192(3):755–766.

Jabali, O., Rei, W., Gendreau, M., and Laporte, G. (2014). Partial-route inequalities for the multi-vehicle routing problem with stochastic demands. *Discrete Applied Mathematics*, 177:121–136.

Jozefowiez, N., Semet, F., and Talbi, E. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2):293–309.

Jozefowiez, N., Semet, F., and Talbi, E. (2009). An evolutionary algorithm for the vehicle routing problem with route balancing. *European Journal of Operational Research*, 195(3):761–769.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2002). *Parallel Problem Solving from Nature VII, Lecture Notes in Computer Science*, volume 2439, chapter Parallel and hybrid models for multi-objective optimization: Application to the vehicle routing problem, pages 271–280. Springer-Verlag.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2007). Target aiming pareto search and its application to the vehicle routing problem with route balancing. *Journal of Heuristics*, 13(5):455–469.

Juan, A., Faulin, J., Grasman, S., Riera, D., Marull, J., and Mendez, C. (2011). Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands. *Transportation Research Part C: Emerging Technologies*, 19(5):751–765. Freight Transportation and Logistics (selected papers from {ODYSSEUS} 2009 - the 4th International Workshop on Freight Transportation and Logistics).

Kenyon, A. S. and Morton, D. P. (2003). Stochastic vehicle routing with random travel times. *Trans-*

portation Science, 37(1):69–82.

Koç, c., Bektaş, T., Jabali, O., and Laporte, G. (2016). Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research*, 249(1):1–21.

Köksalan, M., Wallenius, J., and Zionts, S. (2011). *Multiple criteria decision making: from early history to the 21st century*. World Scientific Publishing, Hackensack, NJ.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416.

Laporte, G., Louveaux, F. V., and Hamme, L. V. (2002). An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423.

Lee, C., Lee, K., and Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *The Journal of the Operational Research Society*, 63(9):1294–1306.

Lei, H., Laporte, G., and Guo, B. (2011). The capacitated vehicle routing problem with stochastic demands and time windows. *Computers & Operations Research*, 38(12):1775–1783.

Lei, H., Laporte, G., and Guo, B. (2012). A generalized variable neighborhood search heuristic for the capacitated vehicle routing problem with stochastic service times. *TOP*, 20(3):99 – 118.

Li, X., Tian, P., and Leung, S. C. (2010). Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics*, 125(1):137–145.

Lin, C., Choy, K., Ho, G., Chung, S., and Lam, H. (2014). Survey of green vehicle routing problem: Past and future trends. *Expert Systems with Applications*, 41(4, Part 1):1118–1138.

Marinakis, Y., Iordanidou, G.-R., and Marinaki, M. (2013). Particle swarm optimization for the vehicle routing problem with stochastic demands. *Applied Soft Computing*, 13(4):1693 – 1704.

Mendoza, J., Rousseau, L.-M., and Villegas, J. (2015). A hybrid metaheuristic for the vehicle routing problem with stochastic demand and duration constraints. *Journal of Heuristics*, pages 1–28.

Mendoza, J. and Villegas, J. (2013). A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. *Optimization Letters*, 7(7):1503–1516.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2011). Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45(3):346–363.

M.G.C. Resende, C. R. (2010). Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau, J.-Y. P., editor, *Handbook of Metaheuristics*, pages 283–319. Springer.

Oppen, J. and Løkketangen, A. (2008). A tabu search approach for the livestock collection problem.

Computers & Operations Research, 35(10):3213 – 3229. Part Special Issue: Search-based Software Engineering.

Oyola, J., Arntzen, H., and Woodruff, D. W. (2015). The stochastic vehicle routing problem, a literature review. *Optimization Online*. `http://www.optimization-online.org/DB_FILE/2016/01/5299.pdf/`.

Pacheco, J. and Martí, R. (2006). Tabu search for a multi-objective routing problem. *The Journal of the Operational Research Society*, 57(1):29–37.

Potvin, J.-Y. (2009). Evolutionary algorithms for vehicle routing. *INFORMS Journal on Computing*, 21(4):518–548.

Potvin, J.-Y. and Bengio, S. (1996). The vehicle routing problem with time windows part II: Genetic search. *INFORMS Journal on Computing*, 8(2):165–172.

Potvin, J.-Y., Kervahut, T., Garcia, B.-L., and Rousseau, J.-M. (1996). The vehicle routing problem with time windows part i: Tabu search. *INFORMS Journal on Computing*, 8(2):165–172.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002.

Raith, A. and Ehrgott, M. (2009). A comparison of solution strategies for biobjective shortest path problems. *Computers & Operations Research*, 36(4):1299–1331.

Rei, W., Gendreau, M., and Soriano, P. (2007). Local branching cuts for the 0-1 integer L-shaped algorithm. *Technical report, CIRRELT-2007-23, Centre interuniversitaire de recherche sur les réseaux d'enterprise, la logistique et le transport*, 44(1):136–146.

Reiter, P. and Gutjahr, W. J. (2012). Exact hybrid algorithms for solving a bi-objective vehicle routing problem. *Central European Journal of Operations Research*, 20(1):19–43.

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1(2):127–190.

Russell, R. A. and Urban, T. L. (2008). Vehicle routing with soft time windows and erlang travel times. *The Journal of the Operational Research Society*, 59(9):1220–1228.

Sawyer, R. and Sawyer, M. (1994). *The Art of War*. Westview Press.

Secomandi, N. and Margot, F. (2009). Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230,.

Sörensen, K. (2006). Route stability in vehicle routing decisions: a bi-objective approach using metaheuristics. *Central European Journal of Operations Research*, 14(2):193–208.

Sörensen, K. and Sevaux, M. (2009). A practical approach for robust and flexible vehicle routing using metaheuristics and monte carlo sampling. *Journal of Mathematical Modelling and Algorithms*, 8(4):387–407.

Sungur, I., Ren, Y., Ordez, F., Dessouky, M., and Zhong, H. (2010). A model and algorithm for the courier delivery problem with uncertainty. *Transportation Science*, 44(2):193–205.

Szeto, W., Wu, Y., and Ho, S. C. (2011). An artificial bee colony algorithm for the capacitated vehicle routing problem. *European Journal of Operational Research*, 215(1):126–135.

Taş, D., Dellaert, N., van Woensel, T., and de Kok, T. (2013). Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research*, 40(1):214 – 224.

Taş, D., Gendreau, M., Dellaert, N., van Woensel, T., and de Kok, A. (2014). 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.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31(2):170–186.

Tan, K., Cheong, C., and Goh, C. (2007). Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813–839.

Tan, K., Chew, Y., and Lee, L. (2006). A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172(3):855 – 885.

Toth, P. and Vigo, D. (1998). Exact algorithms for vehicle routing. In *Crainic, T. and Laporte, G. (Eds.), Fleet management and logistics (pp. 1-31)*. Kluwer Academic Publishers.

Toth, P. and Vigo, D. (2002a). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1-3):487 – 512.

Toth, P. and Vigo, D. (2002b). *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics.

Villegas, J. G., Prins, C., Prodhon, C., Medaglia, A. L., and Velasco, N. (2011). A GRASP with evolutionary path relinking for the truck and trailer routing problem. *Computers & Operations Research*, 38(9):1319–1334.

Voß, S., Fink, A., and Duin, C. (2005). Looking ahead with the pilot method. *Annals of Operations Research*, 136(1):285–302.

Yang, W.-H., Mathur, K., and Ballou, R. H. (2000). Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(1):99–112.

Zhang, T., Chaovalitwongse, W., and Zhang, Y. (2012). Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. *Computers & Operations Research*, 39(10):2277 – 2290.

Paper 1

An Attribute Based Similarity Function for VRP Decision Support

An Attribute Based Similarity Function for VRP Decision Support

Arne Løkketangen[1], Johan Oppen[1], Jorge Oyola[1] and David L. Woodruff[2]

[1]Molde University College, Molde, Norway
[2]Graduate School of Management, UC Davis, Davis Ca 95616 USA

Abstract

When solving problems in the real world using optimization tools, the model solved by the tools is often only an approximation of the underlying, real, problem. In these circumstances, a decision maker (DM) should consider a diverse set of good solutions, not just an optimal solution as produced using the model. On the other hand, the same DM will only be interested in seeing a few of the alternative solutions, and not the plethora of solutions often produced by modern search techniques. There is thus a need to distinguish between good solutions using the attributes of solutions. We develop a distance function of the type proposed in the Psychology literature by Tversky (1977) for the class of VRP problems. We base our difference on the underlying structure of solutions.

A DM is often interested in focusing on a set of solutions fulfilling certain conditions that are of specific importance that day, or in general, like avoiding a certain road due to construction that day. This distance measure can also be used to generate solutions containing these specific classes of attributes, as the normal search process might not supply enough of these interesting solutions. We illustrate the use of the functions in a Multi-objective Decision Support System (DSS) setting, where the DM might want to see the presence (or absence) of certain attributes, and show the importance of identifying solutions not on the Pareto front. Our distance measure can use any attributes of the solutions, not just those defined in the optimization model.

Keywords: Solution Variety, Solution Similarity, Vehicle Routing Problem (VRP), DSS

1 Introduction

The family of Vehicle Routing Problems (VRP), constitute a diverse and at the same time practically important family of problems. Many organizations solve VRPs daily or even more often. Instances of many of the different forms of the VRP are commonly solved manually, or by some tool that is often optimization-based and embedded in a DSS. Common features of the VRP are usually that each instance is defined by a series of n stops that must be served by some number of vehicles using a shared depot where all routes start and end, and that there are resource constraints, the most common being a limited capacity on each vehicle. The goal of the VRP is then to find routes so as to minimize some function that depends on things such as the cost of making deliveries using the chosen routes, the number of vehicles needed, etc. In Toth and Vigo (2002) a variety of extensions to the classical VRP are described, some of which are usually required in order to make the VRP rich enough, and thus suitable for modeling the real world problem at hand. For more information about rich VRPs see, e.g., Hartl et al. (2006); Oppen et al. (2010).

Our goal in this paper is the specification of similarity measures between solutions to a VRP instance. These functions should be based on some measure apart from the objective function value of the solutions. Our function is based on the structural difference between the solutions, because this is what a human dispatcher bases her valuations of difference on. We produce measures

of difference of the type proposed in the Psychology literature by Tversky (1977). We also show how these measures can be extended by including solution attributes not explicitly modeled in the underlying optimization tool.

For convenience, we will define the output of the similarity functions on the range zero to one. This means that they can be subtracted from 1 to yield a distance function. We emphasize that we have no interest in describing optimization methods in this paper, we simply provide means to measure the similarity between two VRP solutions. It should also be stated that because we base our measure on the attributes of stops, arcs and tours of VRP solutions, our measure can handle most extensions to the classical VRP model, thus making it suitable for embedding in a real-world DSS.

Such distance measures can have many applications. The focus of this paper is their use in a DSS to help a decision maker (DM). The DM might want to see multiple good solutions that are different. We presume that an objective function and the problem constraints determine the meaning of "good." We give methods here for quantifying the notion of "different," based on the solution structures. Such a measure, based on the difference between the solution attributes is also of high value in Multi Criteria Optimization, MSO, as shown in Løkketangen and Woodruff (2005).

The opposite use is in plan recovery, where the original plan has to be abandoned due to some unforeseen event, e.g., a breakdown or accident. Here, the DM wants a new plan that deviates as little as possible from the original. There is a similar concern when trying to make robust plans, where small changes in the input should give as little disruption as possible to the original plan, see e.g. Sörensen (2007). If one observes that *similarity* is complimentary to *difference*, the same measure can of course be used for this purpose also.

There are other uses for diversity measures. For example, in search algorithms such as scatter search (Laguna and Martí, 2003) or genetic algorithms (Reeves, 2003) it is assumed that a variety of solutions are available. In some cases, random construction can reasonably be assumed to create variety, but in highly constrained settings it might be important to verify that solutions are in fact mutually distant (see Sörensen and Seveaux (2006)).

In local search based meta-heuristics such as tabu search (Glover and Laguna, 1997), there are several uses. The most obvious is as a diversity measure, to make sure that the search has moved to a significantly different part of the search space. A distance function can also be used in a constructive heuristic, to make sure that the newly constructed solutions are sufficiently different from earlier solutions to justify launching a local search from the new solution.

A DM is often interested to focus on a set of solutions fulfilling certain conditions that are of specific importance that day, or in general, like avoiding a certain road due to road-works that day, or, making sure that the routes are sufficiently similar (for fairness reasons). The normal search process might not supply enough of these interesting solutions. Our distance measure can also be used in a multi-criteria optimization setting, to generate more of solutions containing "sufficient" levels of certain attributes.

Since the words "similarity" and "distance" are complimentary, the literature for computing values for them is intertwined. We will use the terms *similarity* and *distance* function in their broad, intuitive sense.

This introduction is followed by Section 2, which gives examples of other distance measures in the literature. Section 3 gives an introduction to Tversky's similarity measure that we use as a basis for our distance measures, and in Section 4 our distance measures are defined. We then show

examples in Section 5. A description of computational experiments illustrating our methods is in Sections 7 and 6. The final section offers conclusions.

2 Other Distance Measures

The need for measuring distances between solutions is not new, and many metrics have been suggested. The Euclidean distance is, of course, not directly applicable to permutation vectors representing routing order. Our discussion here will be restricted to functions that measure the distance between solutions represented as permutation, as this is the common way to represent solutions to routing problems like the VRP and TSP. We highlight here a few distance measures from the literature, even though they all were designed for other purposes so that differences in the tour structure and other attributes are not addressed properly for use in a DSS. They are typically applied within an algorithm that is searching for good solutions.

Exact Match Distance. This can also be called the *hamming distance* (Hamming, 1950) *for permutations* (see Ronald (1998)); the distance is the number of positions that are different in the two solutions. If S and T are the two solutions, then the distance between them is given by the following formula

$$d_{em}(S,T) = \sum_{i=1}^{N} x_i \text{ where } x_i = \begin{cases} 0 & \text{if S(i) = T(i)} \\ 1 & \text{otherwise} \end{cases} \quad (1)$$

The Deviation Distance. This is also called "Spearman's footrule". The measure here is based on the sum of the total deviation of all items between the two strings. (See Ronald (1998)).

$$d_{dev}(S,T) = \sum_{k=1}^{N} |i-j| \text{ where } T(j) = S(i) = k \quad (2)$$

Bontoux-Feillet Distance. This measure is defined in Bountoux and Feillet (2005) for a Traveling Purchaser Problem. Here the distance between two solutions is defined to be the quotient of the number of markets (i.e. nodes) in the symmetric difference between the two solutions, divided by the number of markets in the union of the two solutions.

The Edit Distance. Given three edit operations (add, delete and substitute). The edit distance (also called the Levenshtein distance) is then the minimum number of operations to transform string S into string T. (See Seveaux and Sörensen (2005), Wagner and Fisher (1974) and Sörensen (2006)) This measure is more a search space distance than a solution space distance, as the edit operations can be considered neighborhood operations.

The Reversal Distance. This is the number of substring reversals required to transform string S into string T. (See Caprara (1999)). This measure is important in molecular biology.

Apart from the Bontoux-Feillet distance, these functions are designed to operate on permutations without regard to their perception by a DM as a VRP. An important objective for our distance function is to be able to produce an explanation of the differences that is understandable to a DM using attributes of a VRP solution. We also go beyond the attributes of solutions that are used in the optimization process. The Bontoux-Feillet Distance is for a single tour, and is similar to the Tversky ratio that we now describe. That distance is specialized for the traveling purchaser problem, while we will introduce a distance function specialized for VRPs.

3 Tversky's similarity measure

We begin our investigation with a ratio version of Tversky's similarity measure (Tversky, 1977). This ratio has been widely used to measure similarity and there is extensive discussion of its properties in the psychology literature (Goertzel, 1997; Medin et al., 1993). It has been used in diverse contexts such as spam detection (Dimmock and Maddison, 2004) and clustering (Ryu and Eick, 2005). For two non-empty sets A and$B$, define

$$J(A,B) \equiv \frac{|A \cap B|}{|A \cap B| + |A - B| + |B - A|}$$

where $|A|$ denotes the cardinality of set A. It is well known that $1 - J(\cdot)$ is a semi-metric; i.e., it satisfies the definition of a metric apart from, perhaps, the triangle inequality.

We will base our similarity measures on generalizations of the ratio given by $J(\cdot)$. Insights into creation of a generalization can be gleaned by considering the literature on two related problems: comparing vectors of categorical data and comparing sets of categorical data. This work has been extended to consider comparisons of vectors of sets of data, which is not our current problem but it is close enough so that solutions to it are instructive. A unifying proposal for vectors of sets and a nice summary of related work is provided by Ruy and Eick (1998).

4 Methods for comparing vectors, sets of vectors and vectors of sets of vectors

Our interest here is strictly in the similarity and difference between solution, not optimization, *per se.* We can assume that all the solutions we look at are good enough, that is, the objective function value is within some fraction of the best found, or some other threshold. Additionally, it should also be stressed that similarity is *n*ot the same as visual pattern matching. This is exemplified in Figure 1, where the two solutions may look very similar, but in fact may be very different in terms of the attributes of interest to the DM.

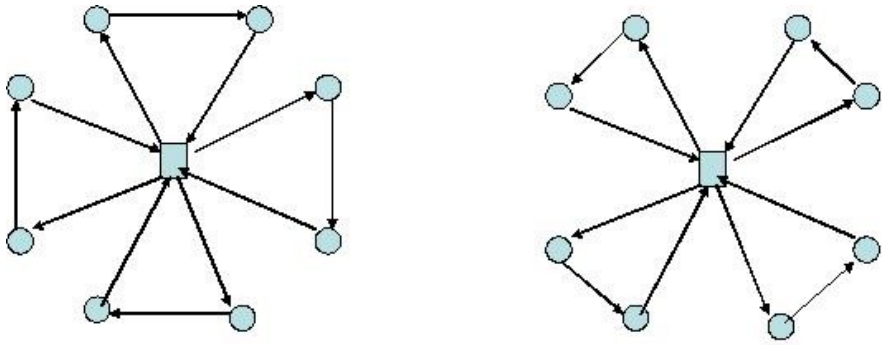

Figure 1: Two Distinct VRP Solutions.

A solution to a VRP instance consists of a collection of vehicle routes that traverse some of the arcs to serve all the stops included in the instance. Each stop, each potential arc, (whether included in the solution or not), and each tour has associated with it a vector of attributes. Given the attribute vectors of two stops, two arcs or two routes, we want to compute the similarity between them. The elements of an attribute vector might be categorical, binary, measured, or might even be sets. A unique identifier of the stop, the arc or the tour needs to be included in the attribute vector and treated as categorical data. This is to avoid a situation where, e.g., two different stops with identical sets of attributes are compared and their similarity is computed to be 1.

Our similarity functions are based on the attributes of the solutions. Before getting into the formalisms, we can get intuition by thinking of VRP solutions from the point of view of each stop. Each stop is on a tour, hence for each stop there is a set of other stops that are on the same tour, a set of arcs that are in the tour and a vector of attributes for the tour itself. Given sets of stops, sets of arcs and attributes of tours for a given stop, we want to compute the similarity between the tour this stop is on in one solution and another. Once we have the similarity for each stop we can compute the similarity between solutions by summing the similarity over the stops.

In order to build up the similarity function, we need functions to find the similarity between vectors, sets of vectors, and vectors of sets of vectors. The vectors correspond to single stops, arcs or tours, the sets of vectors correspond to sets of stops or arcs that share the same tour. Finally, the vector of sets of vectors correspond to a complete solution for a VRP instance, viewed from the point of view of each stop. Løkketangen and Woodruff (2005) developed a method for finding the distance between sets of vectors, where each vector in the set gives the attributes of that member of the set. We use their method as a basis for our similarity function. In order to generalize Tversky's function, Løkketangen and Woodruff generalized the intersection using a function $h(\cdot)$ and the difference using a function $g(\cdot)$; these functions are described in Subsection 4.2. The functions, in turn, require functions to compare attribute vectors which we obtained based on the work of Ryu and Eick; this is described in the next subsection. In Section 5 we give a method of representing VRP solutions so that differences between them can be computed. Subsequent sections give and example and some computational experience.

4.1 Comparing attribute vectors

For two particular values of vector element j, let the function $\eta_j(\cdot)$ take values on $[0, 1]$ corresponding to the dissimilarity between the two attribute values. For a measured attribute, $\eta_j(\cdot)$ should provide a continuous measure of dissimilarity scaled by the variability as measured across all potential measurements or some other set of interest. For example the distance between vectors x and y attributed to element j is,

$$\eta_j(x, y) \equiv \min\left(1, \frac{|x_j - y_j|}{s_j}\right)$$

where s_j is a measure of the dispersion of the values for attribute j; we have in mind the standard deviation, but other measures of dispersion could be used. We assume that $0/0$ is zero. Scaling by the dispersion puts the difference on the same scale in terms of deviation regardless of the original scale of the values. For categorical attributes, including binary attributes, difference is replaced by an indicator of inequality so it will take the value zero or one. If any of the vector elements are sets, then a function such as Tverskys can be used. We then define

$$\delta(x, y; w) \equiv (\sum_{j=1}^{p} \eta_j(x, y) w_j)/(\sum_{j=1}^{p} w_j)$$

to be a measure of dissimilarity between vectors x and y. Here, p is the number of vector elements (attributes) and w is an optional vector of user-specified weights giving the relative importance of the different attributes.

4.2 Comparing sets of vectors

In Løkketangen and Woodruff (2005) functions are developed that allow for a generalization of Tversky difference function given in Section 3 to compare two sets of vectors. Given two sets A and B,

$$g(A, B; w) \equiv \sum_{k \in (A-B)} \sum_{k' \in B} \delta(k, k', w)/|B|$$

provides a generalization of $|A - B|$. Observe that if the vector size is one and all vector elements are treated as categorical variables (i.e., A and B are sets), then $g(A, B; w) = |A - B|$ as shown in Løkketangen and Woodruff (2005). This extends to binary set elements if the range is used to scale δ.

The numerator in Tversky's ratio as given in Section 3 is the cardinality of $A \cap B$. One can make use of the fact that for simple sets, $|A \cap B| = |A| - |A - B| = |B| - |B - A|$. Since, in general, $|A| - g(A, B; w) \neq |B| - g(B, A; w)$, to obtain symmetry we use both in our bounding approximation to $|A \cap B|$, namely

$$h(A, B; w) \equiv (|A| - g(A, B; w) + |B| - g(B, A; w))/2$$

As with $g(\cdot)$, $h(A, B; w) = |A \cap B|$ if A and B are simple sets and w is a vector of ones.

This enables extension of Tversky's similarity ratio to compute dissimilarity between sets of vectors. Define

$$d(A, B; w) \equiv 1 - \frac{h(A, B; w)}{h(A, B; w) + g(A, B; w) + g(B, A; w)} \tag{3}$$

The function $d(\cdot)$ that we have given offers the advantage over the Tversky ratio that instead of simply using the number of elements in the difference sets, the $\delta(\cdot)$ function is used to find out how different they are. In the next section we show how these functions, originally developed for portfolio optimization problems, can be extended to rich VRP solutions.

5 An attribute based measures for similarity between VRP solutions

In a VRP setting, several different attributes for stops, arcs and tours might be of interest for doing comparisons:

- For stops, attributes of interest could be related to accessibility (parking, maneuvering, loading/unloading facilities etc.), time windows, type and amount of load picked up or delivered.

- For arcs, attributes could include length, road quality (number of lanes, type of pavement, average altitude, slope, curves etc.), average travel time, travel time variations (rush hours, ferry routes etc.).

- Tours could be characterized by day/time for the tour, vehicle and driver. In some applications, it might also be of interest to measure the *importance* of the tour. If e.g. production and/or inventory constraints are involved, a tour would be important if the goods picked up on the tour is critical to keep the production process from stopping and less important if the load from the tour is put into inventory for use the next day.

The attributes listed here are meant only as examples; the actual attributes used could be fewer or more and have to be decided based on the application at hand. Note also that a unique identifier

is always included as an attribute, and that many of these attributes might not be used in the optimization model.

Consider the following solution representation that leads to a fairly compact expression of our similarity function. Order the stops (arbitrarily) and index them from 1 to n; we will then use vectors of length n referred to as n-vectors. Given a solution X, we can represent it in a way that is useful for our purpose. Represent X by $\hat{X}$, $\tilde{X}$ and $\bar{X}$, where $\hat{X}$ and $\tilde{X}$ are n-vectors of sets and $\bar{X}$ is an n-vector.

In $\hat{X}$, each vector element i corresponds to a particular stop and gives the attributes of the stops that share the route with i, omitting stop i and the depot.

The arc representation of a solution we will use is analogous: $\tilde{X}$ gives the attributes of the arcs that are on the route of each stop.

The third representation, $\bar{X}_i$, gives the attribute vector of the tour that stop i is on.

We can now give an abstract definition of distance for two VRP solutions represented as X and Y that generalizes the Tversky function:

$$t(X,Y) \equiv \alpha\frac{1}{n}\sum_{i=1}^{n} d(\hat{X}_i,\hat{Y}_i;\hat{w}) + \beta\frac{1}{n}\sum_{i=1}^{n} d(\tilde{X}_i,\tilde{Y}_i;\tilde{w}) + \gamma\frac{1}{n}\sum_{i=1}^{n} \bar{X}_i,\bar{Y}_i;\delta(\bar{w}).$$

where the parameters α, β and γ control the relative importance of stops, arcs and tours, respectively. The sum of these three parameters should equal one to have the function take values on $[0,1]$. The vector $\hat{w}$ gives the weights for the different attributes of stops, while $\tilde{w}$ and $\bar{w}$ give the weights for attributes of arcs and tours, respectively.

6 Example

To illustrate the computations, we will make use of a very small example with five stops and three vehicles. For this example, we associate with each stop three attributes: the stop number, the number of pallets to pick up and a binary variable indicating whether the vehicle has to park in the street or not while loading. With each arc we associate its (origin, destination) pair of nodes, the arc length and number of lanes, and with each route we associate the number of the vehicle performing the route.

We will make use of two solutions:

- E: stops one, four and five are the route for vehicle one; stops two and three are the route for vehicle two.
- F: stops five and three are the route for vehicle one; stops two, one and four are the route for vehicle three.

Table 1 gives the attributes for the stops and the arcs in our small example. Even though some of the arcs are not used in any of the two solutions listed here, their attributes are given because they may be used in some solution. Note also that vehicle one is the only vehicle used in both solutions. The actual tours can be visualized as in Figure 2.

Table 1: Attributes for VRP example

Stop attributes:		
Stop	Pallets	Street parking
1	3	No
2	3	Yes
3	4	No
4	1	Yes
5	5	No

Arc attributes:		
Arc	Length	Lanes
0-1	20 km	2
0-2	10 km	2
0-3	20 km	2
0-4	5 km	1
0-5	10 km	2
1-2	25 km	4
1-3	40 km	2
1-4	15 km	4
1-5	25 km	4
2-3	15 km	4
2-4	15 km	1
2-5	20 km	2
3-4	25 km	1
3-5	20 km	2
4-5	10 km	4

So for the small example given, $\hat{E}_1$ gives the attributes of the stops that share a route with stop one in this solution so it is $\{(4, 1, Yes), (5, 5, No)\}$ because in solution E, stop one shares the route with stops four and five. The other elements of $\hat{E}$ can be constructed in the same way, so $\hat{E}_2 = \{(3, 4, No)\}, \hat{E}_3 = \{(2, 3, Yes)\}, \hat{E}_4 = \{(1, 3, No), (5, 5, No)\}$, and $\hat{E}_5 = \{(1, 3, No), (4, 1, Yes)\}$. This is repeated for $\hat{F}$, so $\hat{F}_1 = \{(2, 3, Yes), (4, 1, Yes)\}, \hat{F}_2 = \{(1, 3, No), (4, 1, Yes)\}, \hat{F}_3 = \{(5, 5, No)\}, \hat{F}_4 = \{(1, 3, No), (2, 3, Yes)\}$, and $\hat{F}_5 = \{(3, 4, No)\}$.

The arc representation of a solution we use is analogous, $\tilde{X}$ gives the attributes of the arcs that are on the route of each stop. If X represents solution E, $\tilde{X}_1 = \{((0-1), 20, 2), ((1-4), 15, 4), ((4-5), 10, 4), ((5-0), 10, 2)\}$ and $\tilde{X}_2 = \{((0-2), 10, 2), ((2-3), 15, 4), ((3-0), 20, 2)\}$.

The third representation, $\bar{X}_i$, gives the attribute vector of the tour that stop i is on. Following our example, it means that, e.g., $\bar{X}_1 = 1$ because we have defined so few attributes for this simple, small example.

Writing down the representations is tedious and doing the calculations by hand is treacherous, but the representations and calculations are easily programmed. To help the reader verify any such programming, we illustrate some of the calculations to complete the example by working backwards down one thread. We start with the distance between the stop attributes for the first stop:

$$d(\hat{E}_1, \hat{F}_1; \hat{w})$$

that creates a need for generalized set differences for stops attributes, e.g.:

$$g(\hat{E}_1, \hat{F}_1; \hat{w}) \equiv \sum_{k \in (\hat{E}_1 - \hat{F}_1)} \sum_{k' \in \hat{F}_1} \delta(k, k', w)/|\hat{F}_1|$$

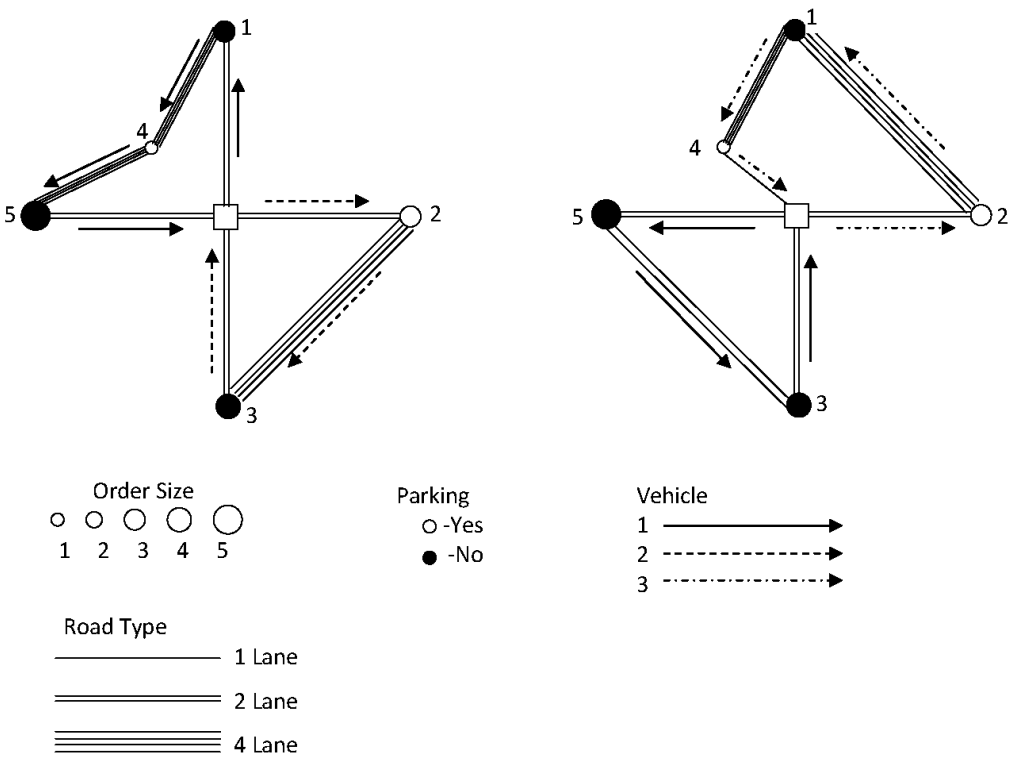

Figure 2: Solutions E and F.

The set $\hat{E}_1 - \hat{F}_1$ has only one member; the set difference is $\{(5, 5, No)\}$ so

$$g(\hat{E}_1, \hat{F}_1; \hat{w}) = \delta((5, 5, No), (2, 3, Yes); \hat{w}) + \delta((5, 5, No), (4, 1, Yes); \hat{w})$$

To compute this, we need to compute δ function values. For example, with an importance weight vector of $\hat{w}$ given by the user:

$$\delta((5, 5, No), (2, 3, Yes); \hat{w}) = \eta_1(5, 2)\hat{w}_1 + \eta_2(5, 3)\hat{w}_3 + \eta_3(No, Yes)\hat{w}_3$$

The η function requires some value for the expected spread of each vector element. The first and third vector elements are categorical so since we do not have a full data set to use estimate a value, we will assume $1/2$ rather than estimating it from the limited data we have. This results in,

$$\eta_3(No, Yes) = \frac{1}{1/2}$$

Since "No" differs from "Yes." The value of $\eta_1(5, 2)$ is also 2 because stop 5 is not the same as stop 2. The number of pallets for stops that we have (namely: 3,3,4,1, and 5) has a sample standard deviation of 1.48 so we use that for illustration purposes. This yields

$$\eta_2(5, 3) = \frac{5 - 3}{1.48}$$

We note that it actually takes less time to write computer code for the distances in general than it takes to work through one full example by hand, even when the example is small.

7 Computational Experiments

This section reports on computational experiments using the distance measure. Our example instance is from http://branchandcut.org/VRP/data (as are all of our examples). It has 5 vehicles and 32 customers. This is a simple example to illustrate calculations has the following attributes. A stop has the customer ID given as a number and the customer demand, normalized by dividing with the capacity of the vehicle. The arcs are represented by the pair of end-point nodes, and the arc length. The tour attributes are just the number of the tour and the set of customers on the tour, as all vehicles are the same, and there are no operational costs associated with a vehicle.

Table 2 shows 4 different, good solutions to the instance A-n32-k5. In Table 3 shows the distances between them assuming equal weights on solution attributes. The 4 solutions are shown graphically in Figures 3 ,4 ,5 and 6. It is evident that solutions A and B are similar. On the other hand, solutions C and D are very different. This is reflected in the pair-wise distances shown in Table 3.

Table 2: Different solutions to the instance A-n32-k5

Solution	Tour	Customers
A	1	30,26,28,18,22,15,29,27
	2	21,31,19,17,14,24,20
	3	8,13,7,16,12
	4	6,3,2,23,4,11,9
	5	5,25,10,1
B	1	30,26,23,28,18,22,15,29,27
	2	21,31,19,17,14,24,20
	3	8,11,13,16,12
	4	7,6,3,2,4,9
	5	5,25,10,1
C	1	30,26,31,19,23,28,18,29,27
	2	12,1,21,14,24,10,20
	3	15,22,8,11,13,16
	4	7,6,3,2,4,9,5
	5	25,17
D	1	29,15,10,25,5,20
	2	7,21,31,19,17,13
	3	23,3,2,6,30
	4	14,28,4,11,8,18,9,22,27
	5	12,1,16,26,24

Table 3: Solution Distances

	A	B	C	D
A		0.18	0.54	0.66
B	0.18		0.42	0.68
C	0.54	0.42		0.76
D	0.66	0.68	0.76	

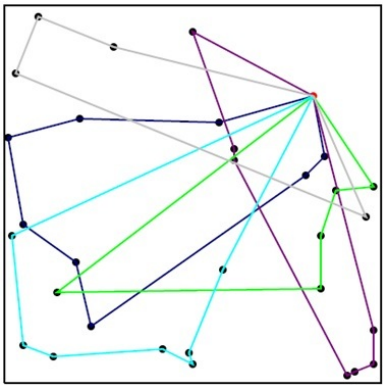

Figure 3: Solution A to instance A-n32-k5

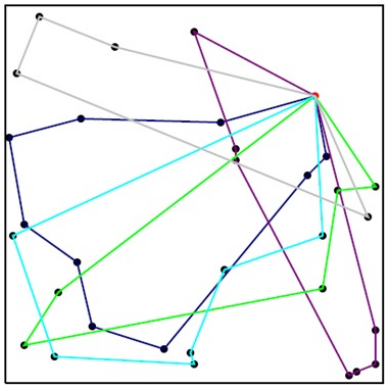

Figure 4: Solution B to instance A-n32-k5

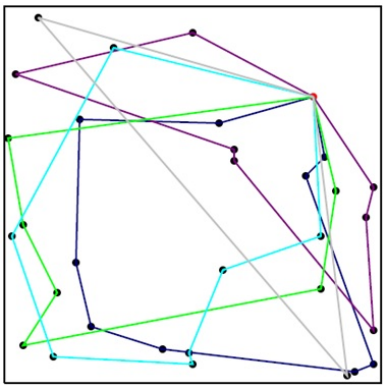

Figure 5: Solution C to instance A-n32-k5

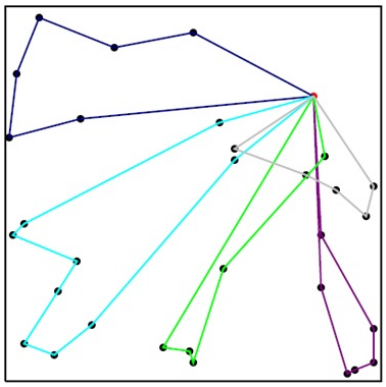

Figure 6: Solution D to instance A-n32-k5

In order to illustrate the usefulness of our distance function $t(X, Y)$ for the VRP, we will look at this in a multi-criteria DSS setting. (For an overview of multi-criteria optimization and VRP, see Jozefowiez et al. (2008). This will show the usefulness of having a distance measure both for selecting between solutions to consider, and for generating solutions containing specific attributes. We show our results on a set of benchmark instances from the literature, modified for our multi-objective setting.

A user might also want to ask: *what makes two solutions different?* and *what are the important stops for determining the distance?*. Because our distance function is decomposable by stop, it is easy to answer these questions. In other words, since $t(X, Y)$ is computed by summing over the stops, the contribution of each stop can be directly determined.

7.1 Solution Distances and λ Distances

Our specific example will be in a DSS where the DM wants to be presented a small set of routes that are good w.r.t. the overall cost, and within the specified route difference threshold. We have implemented this in a solver based on Oppen and Løkketangen (2006) and Cordeau et al. (2001). This solver uses tabu search with an insert neighborhood and infeasibility penalties in the move selection function. The optimization algorithm design has been modified to work in a multi-objective setting, where the final objective is a linear combination of the individual objectives. This means that there is a control parameter λ (between 0 and 1) that gives the blend between the two objectives.

$$Obj_{tot} = \lambda * Obj_1 + (1 - \lambda) * Obj_2$$

This blend may also be regarded as a policy. The goal in a multi-objective optimization setting is often to find the set of solutions that define the Pareto Front, PF (i.e. the set of undominated solutions). In practice there is a limit to both computational time and resources, so every point on the PF cannot be mapped. Also, with the weighted sum of the two objectives, concave regions of the PF may be undetected (Corne et al., 2000). A DM is also in general interested in implementing a policy, and not balance the different criteria dynamically.

A typical result is shown in Figure 7, which was generated from 11 runs of our solver with λ ranging from 0 to 1 in 0.1 increments. We will call this λ-space.

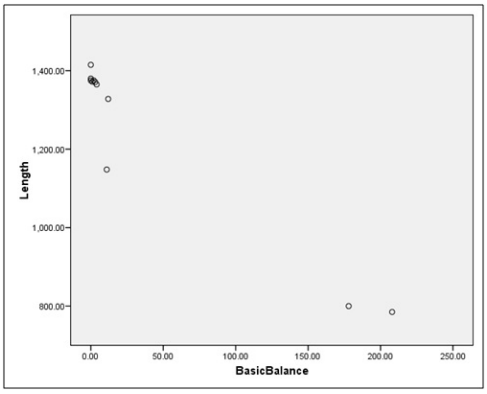

Figure 7: Pareto Front for instance A-n32-k5

It is very easy to imagine that a large distance between solutions in λ-space (measured in λ) corresponds to a large differences between solutions, and vice-versa. This is not necessarily the case. In Table 4 are shown the distances between the solutions E, F and G for instance A-n32-k5. The λ-values used were $E = 0.0$, $F = 0.1$ and $G = 1.0$. The solutions are shown graphically in Figures 8, 9 and 10. The distance between solutions E and F in λ-space is much smaller than the distance between solutions E and G. (0.1 compared to 1.0). Even so, the difference in solution space is larger between solutions E and F, than between solutions F and G. This illustrates the importance of having the difference measure in solution space.

Table 4: Solution Distances – Multi Objective – A-n32-k5

	E	F	G
E		0.76	0.70
F	0.76		0.64
G	0.70	0.64	

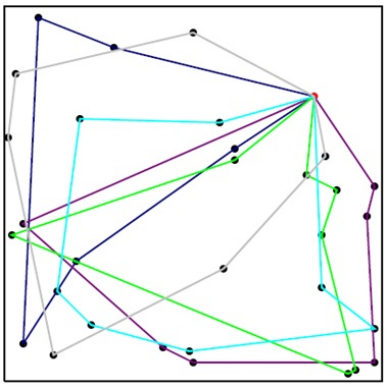

Figure 8: Solution E to instance A-n32-k5

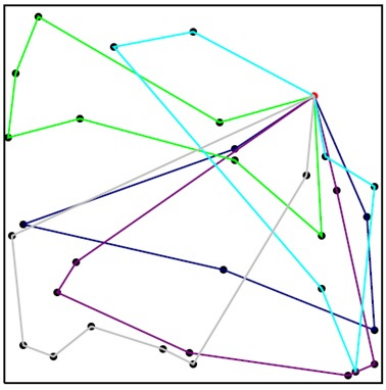

Figure 9: Solution F to instance A-n32-k5

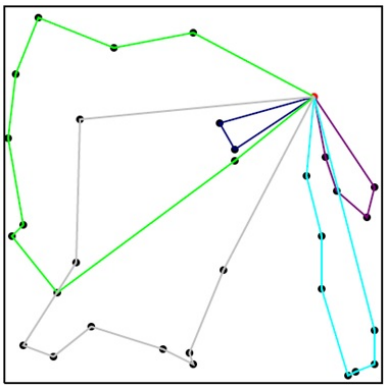

Figure 10: Solution G to instance A-n32-k5

Table 5 shows the correlation between the solution distances for a set of quite different instances from http://branchandcut.org/VRP/data. Both linear (Pearson) and rank (Spearman) correlations are shown. All the numbers are below 0.5, which can be understood as small or medium.

Table 5: correlation between distances in solution space and λ space

Instance	Pearson	Spearman
A-n36-k5	0.37	0.37
A-n37-k6	0.29	0.26
A-n45-k7	0.31	0.29
A-n60-k9	0.41	0.40
A-n80-k10	0.46	0.42
B-n41-k6	0.32	0.30
B-n57-k9	0.42	0.39
P-n70-k10	0.22	0.19
P-n101-k4	0.18	0.16
E-n76-k10	0.37	0.34
G-n262-k25	0.44	0.37
M-n200-k17	0-49	0.44

7.2 Using a Threshold

As stated above, a given value for λ (in our two-objective setting) corresponds to a given policy, or balance between the criteria. Such a policy is often implemented as a threshold value for a given objective. In our example, a threshold on the route balance would correspond to a statement like: “The maximum difference in duration between two routes is 30 minutes.” This corresponds to saying that solutions fulfilling this policy are of interest, and the most desirable are the solutions

on, or near, the PF that fulfill the criteria, and therefore have a “good enough” solution value w.r.t. the overall route length. This thus corresponds to a section of the PF graph. In Figure 11 this is indicated as follows. The overall Pareto Front is indicated by the curved line. The DM indicates preferences using lines on the PF graph. The route balance policy is indicated by the vertical dotted line. The horizontal dotted line indicates the minimum desired solution quality. The set of optimal solutions will then be on the PF between A and B. (If the route balancing objective had been implemented as a constraint, then only the point A will be produced). As we are interested in a portfolio of different, but good, solutions, we will find these between A and B. These solutions are not necessarily positioned only on the PF, but can also be a dominated solution that is close, as indicated by X. We call the set of solutions containing such dominated solutions for the set of *good* solutions.

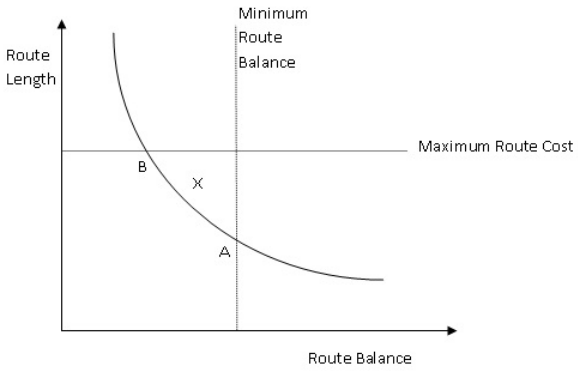

Figure 11: Threshold and the Pareto Front

Table 6 shows the 95% confidence interval for the difference between the average distance obtained from the set of good solutions and similarly from the non-dominated solution set, for a set of the benchmark problems. As can be seen, there is no evidence to say that the average distances between the solutions in the good solutions set are not greater than those in the non-dominated solutions set. The solution having *Best length* corresponds to the point A in figure 11.

Table 6: Solution difference test

	2 Most Different		Best length	
Instance	Lower	Upper	Lower	Upper
A-n36-k5	0.03	0.04	0.02	0.04
A-n37-k6	0.01	0.03	0.01	0.03
A-n45-k7	0.01	0.03	0.00	0.03
A-n60-k9	0.01	0.02	0.00	0.02
A-n80-k10	0.01	0.02	0.01	0.02
B-n41-k6	0.01	0.03	0.01	0.03
B-n57-k9	0.02	0.03	0.01	0.02
P-n70-k10	0.01	0.03	0.01	0.03
P-n101-k4	0.02	0.05	0.02	0.05
E-n76-k10	0.01	0.04	0.00	0.03
G-n262-k25	0.02	0.03	0.02	0.03
M-n200-k17	0.03	0.05	0.03	0.04

We can say that including dominated solutions in the set to give to the DM, will increase the diversity of the presented solutions. This is, according to previous considerations, something that could be of interest for the DM.

The results presented here are intended only as illustrative examples. If attribute weights or the objective function value limit are changed, the results will be different. In practice, attribute weights and other parameters have to be set by the DM based on their preferences for the application at hand.

8 Conclusions

A decision maker (DM) is often more interested in a set of different, good solutions to his problem, rather than just the (optimal) solution produced by a tool, using a simplified model of the underlying real world problem. On the other hand, the same DM will only be interested in seeing a few of the alternative solutions, and not the plethora of solutions often produced by modern search techniques (see Danna and Woodruff (2009)).

We have developed an attribute based distance function for use in a DSS for rich, real world, VRP problems. Based on Tversky's well-known similarity ratio, the function makes use of the tour structure and other attributes of the solution. Of particular importance is that the suggested difference measure also can use solution attributes that are not explicitly modeled in the underlying optimization tool.

We have used this function in a multi-objective setting to produce diverse sets of solutions having specific attributes that are of interest for the DM in a particular planning situation. The examples shown illustrate that using a solution difference measure for screening based on the structure and attributes of the solutions gives more varied solutions for the DM to choose from.

References

Bountoux, B. and Feillet, D. (2005). Ant colony optimization for the traveling purchaser problem. Working paper.

Caprara, A. (1999). Sorting permutations by reversals and eulerian cycle permutations. *SIAM Journal on Discrete Mathematics*, 12(1):91–110.

Cordeau, J.-F., Laporte, G., and Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operations Research Society*, 52:928–936.

Corne, D. W., Knowles, J. D., and Oates, M. J. (2000). *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18–20, 2000 Proceedings*, chapter The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization, pages 839–848. Springer Berlin Heidelberg, Berlin, Heidelberg.

Danna, E. and Woodruff, D. L. (2009). How to select a small set of diverse solutions to mixed integer programming problems. *Operations Research Letters*, 37(4):255–260.

Dimmock, N. and Maddison, I. (2004). Peer-to-peer collaborative spam detection. *Crossroads*, 11:4–4.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Kluwer academic publishers.

Goertzel, B. (1997). *From Complexity to Creativity: Explorations in Evolutionary, Autopoietic, and Cognitive Dynamics*. Kluwer, Dordrecht.

Hamming, R. (1950). Error-detecting and error-correcting codes. *Bell System Technical Journal*, 29(2):147–160.

Hartl, R. F., Hasle, G., and Janssens, G. K. (2006). Special issue on rich vehicle routing problems. *Central European Journal of Operations Research*, 14(2):103–104.

Jozefowiez, N., Semet, F., and Talbi, E. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189(2):293–309.

Laguna, M. and Martí, R. (2003). *Scatter Search: Methodology and Implementations in C*. Kluwer Academic Publishers.

Løkketangen, A. and Woodruff, D. (2005). A distance function to support optimized selection decisions. *Decision Support Systems*, 39:345–354.

Medin, D., Goldstone, R., and Gentner, D. (1993). Respects for similarity. *Psychological Review*, 100:254–278.

Oppen, J., Lkketangen, A., and Desrosiers, J. (2010). Solving a rich vehicle routing and inventory problem using column generation. *Computers & Operations Research*, 37(7):1308 – 1317. Algorithmic and Computational Methods in Retrial Queues.

Oppen, J. and Løkketangen, A. (2006). Arc routing in a node routing environment. *Computers and Operations Research*, 33(4):1033–1055.

Reeves, C. (2003). Genetic algorithms. In Glover, F. and Kochenberger, G., editors, *Handbook of Metaheuristics*, pages 55–82. Kluwer Academic Publishers.

Ronald, S. (1998). More distance functions for order based encodings. In *Proceedings of the IEEE Conference on Evolutionary Computation*, pages 558–563. IEEE Press.

Ruy, T. and Eick, C. (1998). A unified similarity measure for attributes with set or bag of values for database clustering. In *The 6th International Workshop on Rough Sets, Data Mining and Granular Computing (RSDMGrC'98)*.

Ryu, T. and Eick, C. (2005). A database clustering methodology and tool. *Information Sciences*, 171:29–59.

Seveaux, M. and Sörensen, K. (2005). Permutation distance measures for memetic algorithms with population management. In *Extended Abstract Book MIC 2005, Metaheuristic International Conference*.

Sörensen, K. (2006). Distance measures based on the edit distance for permutation type representations. *Journal of Heuristics*. Accepted for publication.

Sörensen, K. (2007). Route stability in vehicle routing decisions: A practical approach using meta-heuristics. *Central European Journal of Operations Research*. Accepted for publication.

Sörensen, K. and Seveaux, M. (2006). MA - PM: Memetic algorithms with population management. *Computers and Operations Research*, 33:1214–1225.

Toth, P. and Vigo, D., editors (2002). *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, Philadelphia.

Tversky, A. (1977). Features of similarity. *Psychological Review*, 84:327–352.

Wagner, R. and Fisher, M. (1974). The string-to-string correction problem. *Journal of the Association for Computing Machinery*, 21:168–173.

Paper 2

GRASP-ASP: An algorithm for the CVRP with route balancing

GRASP-ASP: An algorithm for the CVRP with route balancing

Jorge Oyola[1] and Arne Løkketangen[2]

[1]Molde University College, Molde, Norway
[2]Late of Molde University College, Molde, Norway

February 22, 2016

Abstract

An extension of the capacitated vehicle routing problem (CVRP) is studied in this paper. In this version the difference between the individual route lengths is minimized simultaneously with the total length. The drivers' workload and perhaps, income, may be affected by the route lengths; so adding this objective makes the problem closer to real-life than the original, single-objective problem. A heuristic based on GRASP is used to obtain an approximation of the Pareto set. The proposed heuristic is tested on instances from the literature, obtaining good approximations of the Pareto set.

Keywords: CVRP, Multi-objective optimization, GRASP

1 Introduction

In the capacitated vehicle routing problem (CVRP) the total travel cost is minimized, given a set of customers with non-negative demand, a homogenous fleet of vehicles with limited capacity and a travel cost from node to node (customers and depot). A solution to the CRVP is a set of routes, each one starting and ending at the depot, where each customer is visited exactly once by one vehicle. In an optimal solution, the total cost (length) is minimized (Toth and Vigo, 2002).

Taking into account just the total length may produce substantial differences among the route lengths. In some industries the income of the driver may be affected by the traveled distance. This could be seen as unfair by the drivers, since not all are having the same workload, affecting their welfare. In some settings, drivers are considered to be an arena of competition among the transportation companies, making their welfare a significant issue (Lee and Ueng, 1999).

Including the route balance in the CVRP may describe a problem closer to real-life. This paper deals with a bi-objective vehicle routing problem, where the balance of the route lengths is included as an objective in addition to the traditional minimization of the total length. This approach is relevant since due to drivers agreements, legal restrictions or fairness, the drivers' workload might become an important aspect to be considered by the decision maker when doing the transportation planning. The problem will be considered as a multi-objective problem. The additional objective included in the problem is measured as the difference between the longest and the shortest route. This problem is known as the vehicle routing problem with route balancing (VRPRB) (Jozefowiez et al., 2009, 2007b).

Several approaches have been used to deal with the VRPRB. These approaches involve evolutionary algorithms, either combined with tabu search (Jozefowiez et al., 2002, 2007b) or with additional diversification strategies (Jozefowiez et al., 2009). In this paper, a new algorithm is proposed to

find solutions to the VRPRB, it combines local search and a variant of a greedy randomized adaptive search procedure (GRASP) (Resende and Ribeiro, 2003, 2010). The GRASP metaheuristic has been previously used in solving multi-objective problems, e.g. knapsack and rule selection (Reynolds et al., 2009; Vianna and Arroyo, 2004). However a different approach is used here, where no weights are assigned to the objective functions in the problem during the enumeration of the Pareto set.

Following the spirit of the ruin and recreate heuristic (Schrimpf et al., 2000), the constructive phase of the GRASP procedure starts from a partially built solution. This initial partial solution is obtained using the common parts of two previously found solutions (ruin). A GRASP procedure is then applied to complete the solution (recreate). This process we call the greedy randomized adaptive search procedure with advanced starting point (GRASP-ASP). To the best of the authors' knowledge, the approach used in GRASP-ASP, with no weights and partially built starting solutions has not been used before for solving multi-objective problems.

Heuristics with a similar approach are iterated greedy algorithm (IG) (Jacobs and Brusco, 1995) and tabu search with strategic oscillation (TSSO) (Lozano et al., 2003), for the case of single objective optimization problems. In a multi-objective framework, the restarted iterated Pareto greedy (RIPG) algorithm (Minella et al., 2011) has been used and includes a destructive phase and a constructive one. In IG and TSSO one single solution is obtained starting from a partially built solution. In GRASP-ASP, as well as in RIPG, a set of solutions is obtained starting from a partially built solution. In all these algorithms, the constructive phase starts with a partial solution. In IG, TSSO and RIPG, the partial solution starts after randomly removing elements from a complete solution. In GRASP-ASP, the partial solution is not obtained in a random way, but from the common elements in two complete solutions. In these four algorithms, the construction phase is done using a greedy algorithm. In IG the elements are inserted in the solution having as criteria the best impact on the objective function. TSSO uses a similar approach, but it also has a memory structure to prevent some elements to be part of the new solution. RIPG inserts the elements in a specific order. The first is inserted in every possible position in the partial solution, creating a set of partial solutions, one per each different position. The second element is inserted in every possible position of every partial solution. The process continues until all the elements have been inserted. In GRASP-ASP the construction phase is performed by a GRASP procedure and takes into consideration the impact on both objective functions.

The rest of the paper is organized as follows: in Section 2 the multi-objective optimization problem is introduced; the algorithm used to solve the VRPRB is presented in Section 3; the computational results and the conclusions are presented in Sections 4 and 5 respectively.

2 Multi-objective optimization

In a multi-objective optimization problem (MOP) several functions are optimized (minimized or maximized) subject to the same set of constraints. Without loss of generality the MOP can be treated as a minimization problem, in such case, it can be stated as

$$min\,\{F(x) = (f_1(x), f_2(x), \ldots, f_n(x))\} \tag{1}$$

$$subject \quad to$$

$$x \in D \tag{2}$$

with the number of objective functions being $n \geq 2$; the decision variable vector $x = (x_1, x_2, \ldots, x_r)$; the feasible solution space D; and $F(x)$ the objective vector (Jozefowiez et al., 2007b).

Real-life problems often involve more than one single objective. In fact, it is considered that transportation planning is inherently multi-objective in nature (Current and Min, 1986). This becomes a motivation to include several relevant objectives into the optimization problem, since it is expected that it will become more realistic.

2.1 Pareto optimal solutions

For an optimization problem to be considered multi-objective, the different objectives should be in conflict. Otherwise, the problem may be solved by single objective techniques, since optimizing one of the objectives will improve the other. But because of the conflict among the different objectives, no single solution is able to minimize all the objectives of the problem simultaneously. Instead of that, a solution to a MOP is given by a set of solutions, each of which are not expected to be optimal for all the objectives. These solutions are called tradeoff solutions (Collette and Siarry, 2003). When dealing with a real problem, not all the solutions in the solution set can be implemented, the decision maker should select just one solution, usually after examining the tradeoff solutions.

A solution y with objective function values $(f_1(y), f_2(y), \ldots, f_n(y))$, dominates a solution z, $y \prec z$, if and only if $\forall\ i\ \{1, 2, \ldots, n\}\ f_i(y) \leq f_i(z)$, and $\exists\ j\ \{1, 2, \ldots, n\}$, such that $f_j(y) < f_j(z)$. That is, the solution z does not perform better than y in any objective functions, but it performs worse in at least one. The set of non-dominated solutions is called the Pareto set (PS) or the Pareto optimal solutions and it defines the solution set of a multi-objective optimization problem (Jozefowiez et al., 2007b). If the last condition is not fulfilled, solution z does not perform worse than y in any of the objectives, then it is said that y weakly dominates solution z, $y \preceq z$ (Knowles, 2002).

The method or algorithm used for solving the MOP may not guarantee a set of non-dominated solutions as result. Then the obtained solutions are not Pareto optimal solutions. If this is the case and the algorithm used to solve the MOP does not find a solution z that dominates a solution y, then the latter is considered a *potentially* Pareto optimal solution, relative to the particular algorithm or method that was used to solve the problem (Jozefowiez et al., 2007b). The set of *potentially* Pareto optimal solutions, relative to GRASP-ASP, is an approximation to the Pareto set (PS_{apr}).

2.2 Mathematical model for the VRPRB

A model for the VRPRB is constructed based on the standard model for the VRP (Laporte, 1992), including some different constraints to prevent subtours (Desrochers et al., 1988). The notation used in the model is described here:

Variables	
X_{ijk}	Indicates if vehicle k traverses arc (i, j)
V_{ik}	Cargo that vehicle k carries after departing from customer i
L	Length of the longest tour
S	Length of the shortest tour
Parameters	
$\mathcal{A}$	Set of arcs
$\mathcal{K}$	Set of vehicles
$\mathcal{N}$	Set of customers plus the depot
n	Number of customers
c_{ij}	Distance between customers i and j
d_i	Demand of customer i
Q	Vehicle's capacity

$$min \left\{ \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij} X_{ijk}, \quad L - S \right\} \tag{3}$$

$subject \quad to$

$$\sum_{j\in\mathcal{N}} X_{0jk} \leq 1, \quad \forall k \in \mathcal{K} \tag{4}$$

$$\sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{K}} X_{ijk} = 1, \quad \forall i \in \mathcal{N}\backslash 0 \tag{5}$$

$$\sum_{j\in\mathcal{N}} X_{ijk} = \sum_{j\in\mathcal{N}} X_{jik}, \quad \forall i \in \mathcal{N}, k \in \mathcal{K} \tag{6}$$

$$V_{0k} = \sum_{(i,j)\in\mathcal{A}} d_i X_{ijk}, \quad \forall k \in \mathcal{K} \tag{7}$$

$$V_{jk} \leq V_{ik} - d_j + Q(1 - X_{ijk}), \quad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K} \tag{8}$$

$$V_{0k} \leq Q, \quad \forall k \in \mathcal{K} \tag{9}$$

$$\sum_{(i,j)\in\mathcal{A}} c_{ij} X_{ijk} \leq L, \quad \forall k \in \mathcal{K} \tag{10}$$

$$\sum_{(i,j)\in\mathcal{A}} c_{ij} X_{ijk} \geq S, \quad \forall k \in \mathcal{K} \tag{11}$$

$$X_{ijk} \in \{0, 1\}, \quad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K} \tag{12}$$

$$V_{ik} \geq 0, \quad \forall i \in \mathcal{N}, k \in \mathcal{K} \tag{13}$$

$$L, S \geq 0 \tag{14}$$

Equation (3) shows the two objectives to minimize, total length and route balance. Constraints (4) - (6) ensure that at most k vehicles are used and every customer is visited exactly one time. Constraints (7) computes the initial load on every vehicle. Constraints (8) acts as sub-tour elimination constraints. Constraints (9) impose the capacity limit to each vehicle. Constraints (10) and (11) establish the bounds for the value of the longest and the shortest tour, respectively.

3 The GRASP-ASP algorithm for solving the VRPRB

The proposed algorithm is divided into two phases. The first phase finds the initial set of extremal solutions in the approximation of the Pareto set (best found solutions for each objective when considered individually). In the second phase more solutions are generated starting from the solutions obtained in phase I. It is expected that the approximation of the Pareto set is located between the best solutions found for each of the objectives individually. Solutions in the approximation of the Pareto set represent a tradeoff between two objective functions. Then it could be expected that these solutions share attributes with the two solutions optimizing the different objectives, in a higher or lower degree depending on which objective the tradeoff is favoring. In Phase II, more solutions

are generated from some of the solutions in the *prospects to become part of the approximation of the Pareto set* (PS_{pro}), this set is defined in Section (3.2.1). It starts with the two solutions obtained in Phase I and through an iterative process, the algorithm attempts to improve PS_{pro}, as described in Section 3.2.

A solver was implemented in C++ using an existing code for the CVRP (Oppen and Løkketangen, 2006), which was based on the unified tabu search heuristic for vehicle routing problems with time windows (Cordeau et al., 2001; Glover and Laguna, 1997; Glover, 1989).

3.1 Phase I

Tabu search (Glover and Laguna, 1997; Glover, 1989) has shown to be effective when dealing with VRP (Zachariadis et al., 2009). For Phase I a tabu search algorithm, based on the unified tabu search heuristic for vehicle routing problems with time windows (Cordeau et al., 2001; Oppen and Løkketangen, 2006) is performed for each of the objectives separately in the VRPRB. As a result two solutions are obtained, the one that optimizes the total length (x_l) and the route balance (x_r). These solutions become the initial members of the reference set, which later will include all the non-dominated solutions found during the search in Phase II. A description is shown in Algorithm 3.1.

Algorithm 3.1: PHASE I($f_1, f_2, \ldots, f_m : objective\ functions$)

Let s be an initial solution
$PS \leftarrow \emptyset$
for $i \leftarrow 0$ **to** m
  do $PS \leftarrow PS \cup$ TABU SEARCH(f_i, s)
return (PS)

Algorithm 3.2: TABU SEARCH($f : objective, s : solution$)

Let s^* be the best found solution
Let γ be the diversification function
Let α be the penalization for infeasibility
Let c be the cost function of f
Let $N(s)$ be the neighborhood of solution s
if $feasible(s)$
  then $\begin{cases} s^* \leftarrow s \\ c(s^*) \leftarrow c(s) \end{cases}$
  else $c(s^*) \leftarrow +\infty$
while $stopping\ criterion$
  do $\begin{cases} c(s') \leftarrow +\infty \\ \textbf{for } i \leftarrow 0 \textbf{ to } |N(s)| \\ \quad \textbf{do} \begin{cases} \bar{s} \leftarrow s_i \in N(s) \\ c(\bar{s}) \leftarrow c(\bar{s}) + \gamma(\bar{s}) + \alpha(\bar{s}) \\ \textbf{if } (\bar{s} \textbf{ not } tabu \textbf{ and } c(\bar{s}) < c(s')) \textbf{ or } (c(\bar{s}) < c(s^*) \textbf{ and } feasible(s)) \\ \quad \textbf{then} \begin{cases} s' \leftarrow \bar{s} \\ c(s') \leftarrow c(\bar{c}) \end{cases} \end{cases} \\ twoOptProcedure(s') \\ update(\gamma, \alpha) \\ update(tabulist) \\ update(s^*, c(s^*), s) \end{cases}$
return (s^*)

3.2 Phase II for the GRASP-ASP algorithm

The greedy randomized adaptive search procedure (GRASP) is an iterative metaheuristic (Resende and Ribeiro, 2003, 2010) consisting of two phases: construction and local search.

At every GRASP iteration a solution is built (a repair procedure can be applied in case of infeasibility) and finally a local search is performed, until a local optimum is found. In the single objective approach, the solution is built by including, one at the time, elements to be part of it. At every stage of the construction process, all the candidate attributes that could be included in the solution at the current step are evaluated using a greedy approach. A list, known as the restricted candidate list (RCL), is created with the elements that have a higher performance according to the previous evaluation. The element to be added to the current solution is randomly selected from the RCL (Resende and Ribeiro, 2010). The process is repeated until the solution is completed, updating at every step the evaluation, list of candidates and RCL.

3.2.1 Approximation of the Pareto set

The *prospects* to become part of the approximation of the Pareto set (PS_{pro}), correspond to the set of solutions found by the algorithm that are not dominated by any other solution found during the current search. The cardinality of this set is kept less than or equal to a parameter L_P.

When the cardinality of PS_{pro} exceeds L_P, the additional solutions are removed by using the following simple procedure. Every solution y, except the two extremal solutions, has two neighbors z and w such that for a particular objective i, $f_i(w) < f_i(y) < f_i(z)$. The summation of the Manhattan distance, in the objective space, to both neighbors is computed for every solution in the set of prospects to become part of the approximation of the Pareto set, except by the extremal

solutions. The solution with the smaller total manhattan distance to both neighbors is removed from the set. The procedure is repeated until the number of solutions reaches L_P. The extremal solutions are kept as long as they remain prospects to become part of the approximation to the Pareto set.

The GRASP-ASP procedure builds a set of solutions starting from a half-built solution, instead of an empty solution. The starting solution consists of the common elements of two previously found solutions. These two solutions should not have more than L_A percent of common attributes. At the beginning of phase II, there are just two solutions to be considered, i.e. the extremal solutions found in Phase I. These are the initial set PS_{pro}. Once the procedure is applied to these two solutions, the set is updated with the solutions that have just been built. This process is repeated at most L_N times. If after one of these repetitions, the time consumed by the GRASP-ASP procedure is longer than L_T seconds, the algorithm stops.

At every repetition of the process, every pair of solutions from a subset of PS_{pro} is used in the following way. The set PS_{pro} is updated with the solutions built in the process that are not dominated by any other solution in the set. The cardinality of the subset of solutions used in the GRASP-ASP procedure is defined as no greater than L_G. The solutions to include in such subset are the two extremal plus $L_G - 2$ randomly selected solutions. In case that PS_{pro} has a cardinality less than L_G, every pair of solutions with the percentage of common attributes no greater than L_A, will be subjected to the GRASP-ASP procedure.

Algorithm 3.3 summarizes the procedure. Once the algorithm stops, PS_{pro} becomes the actual PS_{apr}. Figure 1 shows how PS_{pro} improves with the repetitions of the GRASP-ASP procedure.

Algorithm 3.3: PHASE II. GRASP-ASP(s_L, s_B, α)

Let s_L be the best found solution that minimizes the total length
Let s_B be the best found solution that minimizes the route balance
Let α be the algorithm parameter
Let $\mathcal{R}$ be PS_{pro}
Let $\mathcal{G}$ be a subset of PS_{pro}
Let $\mathcal{S}$ be the solutions obtained by GRASP-ASP algorithm
$\mathcal{R} \leftarrow s_L \cup s_B$
while *stopping criterion*
do
- $\mathcal{S} \leftarrow \emptyset$
- **for each** $(s_i, s_j) \in \mathcal{G}$
  - **do**
    - **while** *stopping criterion*
      - **do** $\{\mathcal{S} \leftarrow$ GRASPPROCEDURE(s_i, s_j, α)
- $\mathcal{S} \leftarrow$ LOCALSEARCH$(\mathcal{S}, \mathcal{R})$
- $\mathcal{R} \leftarrow updateProspectSolutions(\mathcal{R}, \mathcal{S})$

return $(\mathcal{R})$

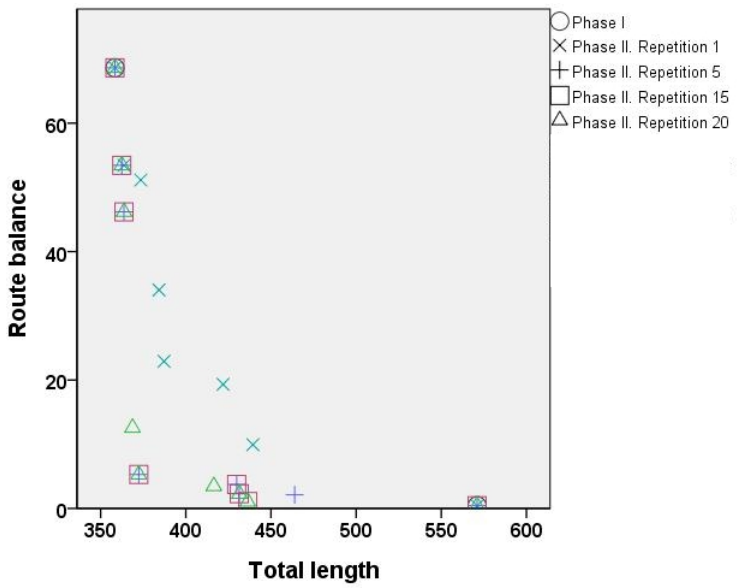

Figure 1: Improvement of the approximation of the Pareto solution with the repetitions of the GRASP-ASP for instance E021-04m

Algorithm 3.4: GRASPPROCEDURE(s_1, s_2, α)

Let $c(a_i)$ be the fitness of attribute a_i
Let Cl be the list of attributes including customers not assigned yet
Let Nr be the number of GRASP-ASP iterations
Let U be the set of solutions built from 2 starting solutions
$A_1 \leftarrow \{(i,k)|(i,k) \in s_1\}$
$A_2 \leftarrow \{(i,k)|(i,k) \in s_2\}$
$\mathcal{U} \leftarrow \emptyset$
$j \leftarrow 0$
while $j < Nr$
do
- $s \leftarrow A_1 \cap A_2$
- $initialize(Cl)$
- $paretoRanking(Cl)$
- **while** $Cl \neq \emptyset$
  - **do**
    - $c_{min} \leftarrow min\{c(a)|a \in Cl\}$
    - $c_{max} \leftarrow max\{c(a)|a \in Cl\}$
    - $RCL \leftarrow \{a \in Cl|c(a) \leq c_{min} + \alpha(c_{max} - c_{min})\}$
    - $\bar{a} \leftarrow random(a)|a \in RCL$
    - $s \leftarrow s \cup \{\bar{a}\}$
    - $update(Cl)$
    - $paretoRanking(Cl)$
- **if** $infeasible(s)$
  - **then** $\{repair(s)$
- **if** $feasible(s)$
  - **then** $\{\mathcal{U} \leftarrow \mathcal{U} \cup s$
- $j \leftarrow j + 1$

return $(\mathcal{U})$

3.2.2 Pareto rank of partial solutions

GRASP-ASP constructs a solution starting from a partially built solution, by adding attributes, one at the time. Every candidate A_j represents the insertion of customer i in a tour k. The selection of the attributes to be added to the partial solution requires an evaluation of all attribute candidates. This evaluation is performed using a Pareto rank (Mateo and Alberto, 2012; Zitzler et al., 2001), which takes into consideration the Pareto dominance concept.

For every candidate or possible insertion A_j, its impact on both objective functions is evaluated. Each candidate A_j is compared with all the others to compute how many of them are dominated by A_j. This corresponds to the *strength value*, $S(j)\ \forall j \in \{1, 2, \ldots, T\}$, where T is equal to the actual number of attribute candidates.

The *raw fitness* is computed for every candidate:

$$R(j) = \sum_{t:A_t \prec A_j} S(t) \quad \forall t \in \{1, 2, \ldots, T\} \tag{15}$$

Non-dominated candidates will have a *raw fitness* equal to zero. It may happen that several candidates have the same *raw fitness*. To be able to rank these candidates a *density* $D(j)$ is computed. $D(j)$ is computed as a decreasing function of the Euclidean distance of the candidate A_j to the k-nearest neighbor. The density will always lie in the interval $(0, 1)$ and it is defined as

$$D(j) = 1/(d_k + 2), \tag{16}$$

where d_k is the distance to the k-nearest neighbor.

The performance of each candidate will be measured by the *fitness* $F(j) = R(j) + D(j)$. The RCL is built with the α fraction of the candidate attributes that show a higher performance (lower fitness).

For every pair of solutions selected for GRASP-ASP a maximum of G_{max} solutions is built. If the initial number of customers to be inserted into the partial built solution is lower than C_{min}, then the number of built solutions will be equal to the factorial of the initial number of customers.

3.2.3 Local search procedure

At the end of each repetition of the GRASP-ASP procedure, a local search is performed to every newly found solution. The objective is to improve the quality of the solution, if possible. This local search is based on the dominance concept. Starting from the initial solution, just moves that lead to a solution that dominates the current solution are accepted. Algorithm 3.5 describes the procedure.

Algorithm 3.5: LOCALPSEARCH($\mathcal{S} : solutions, \mathcal{R} : PS_{pro}$)

$\mathcal{P} \leftarrow \emptyset$
for each $s \in \mathcal{S}$
do
- $localOptimum \leftarrow false$
- **while** $localOptimum = false$
  - **do**
    - $j \leftarrow 0$
    - $moveFound \leftarrow false$
    - **while** $moveFound = false$ **and** $j < |N(s)|$
      - **do**
        - **if** $s_j \prec s$
          - **then** $s \leftarrow s_j$; $moveFound \leftarrow true$
        - $j \leftarrow j + 1$
    - **if** $moveFound = false$
      - **then** $localOptimum \leftarrow true$
- $\mathcal{P} \leftarrow \mathcal{P} \cup s$

return $(\mathcal{P})$

4 Computational results

4.1 Test environment

Tests have been conducted using 20 standard instances: 7 by Christofides, Mingozzi and Toth, E016-03m, E016-05m, E021-04m, E021-06m, E022-06m, E026-08m and E101-10c; 2 by Christofides and Eilon, E051-05e and E101-08e; 6 by Gillett and Miller, E023-05s, E030-04s, E033-05s, E076-07s, E076-08s and E101-14s; 3 by Hadjiconstantinou, Christofides and Mingozzi E031-09h, E036-11h, E041-14h; 1 by Russell E076-14u; and 1 by Rinaldi, Yarrow and Araque E048-04y. These instances are available online (Operations Research Group - Library of Instances, 2012). The instances were selected in a way that different sizes (number of nodes and vehicles) would be included. The size of the instances can be seen in the name: the first number in it shows the number of nodes and the second one represents the number of vehicles in the instance.

The mathematical model presented in Section 2.2 was implemented in GUROBI 5.1, using the *weighted sum of objective function method* (WSO) to solve it. Eleven sets of weights (λ, $1 - \lambda$) were used, from 0 to 1, with 0.1 step, i.e. a weight λ was assigned to the total length and $1 - \lambda$ to the balance. A maximum of eleven solutions per instance are obtained by WSO solved by GUROBI (WSO-G), one per each set of weights, however it may occur that the same solution is obtained when using two different sets of weights.

In three instances WSO-G was able to find solutions within a 0.01% MIP gap (default for GUROBI), for some of the set of weights. In the instances E016-03m and E016-05m, for all sets of weights a solution was found within the default MIP gap, except the set (0, 1), for which after 3 hours the best feasible found solution was not within such MIP gap. For the case of the instance E023-05s a solution within the default MIP gap was reported for the sets (0.8, 0.2), (0.9, 0.1) and (1, 0). For these instances, approximation to the Pareto set found by GRASP-ASP (PS_{apr}(GRASP-ASP)) is compared with a a very close approximation of the PS, since optimal solutions of the WSO are part of the PS(Collette and Siarry, 2003). This provides a good reference set for evaluating the performance of GRASP-ASP.

For the other instances a maximum time of three hours was given to each set of weights, for a total time of 33 hours. A maximum of eleven solutions will be obtained by GUROBI (PS_{apr}(WSO-G)), one per each set of weights. To make a fair comparison if PS_{apr}(GRASP-ASP) has a cardinality bigger than eleven, it is reduced to eleven, using a similar process to the one described in Section 3.2.1 to reduce the cardinality of the PS_{pro} set. Ten different runs of the GRASP-ASP algorithm per instance were performed.

The parameters of the algorithm were tuned by means of preliminary testing. For sensitivity analysis, a selection of instances were run with changes around $\pm$ 10% from the base parameter values. The results were reassuring in terms of stability of the method. The values given to these parameters are:

- Maximum cardinality of the set of prospects to become part of the approximation to the Pareto set. $L_P = 100$
- Maximum percentage of common attributes in two solutions for performing GRASP-ASP on them. $L_A = 90\%$
- Maximum number of times than GRASP-ASP is performed on the set of prospects to become part of the approximation to the Pareto set. $L_N = 20$
- Maximum number of elapsed seconds for starting a new repetition of the GRASP-ASP. $L_T = 1800$
- Maximum cardinality of the subset of prospects to become part of the approximation to the Pareto set used at every GRASP-ASP repetition. $L_G = 15$
- Fraction of the best candidate attributes included in the RCL. $\alpha = 0.2$
- Maximum number of solutions built by GRASP-ASP from a starting solution. $G_{max} = 150$
- Minimum initial number of customers be be inserted in a half built solution, in order to build $Gmax$ solutions by GRASP-ASP. $C_{min} = 6$

All computational experiments were conducted on a computer with processor Intel (R) Xeon (R) CPU E31270 @ 3.40 GHz and 16.0 GB of RAM.

4.2 Evaluation criteria

Measuring the quality of the set of (potentially) Pareto optimal solutions to a multi-objective problem is not straightforward. Several performance metrics have been proposed in the literature. However, no single metric is able to fairly compare two sets of solutions, since all of them present some drawbacks. Consequently more than one metric is required to make a better comparison. Some metrics are: the $\mathcal{S}$ metric , $\mathcal{C}$ metric (Jozefowiez et al., 2009, 2007b; Knowles, 2002; Mateo and Alberto, 2012), the ratio of non-dominated individuals in a set $\mathcal{X}$, $RNI(\mathcal{X})$ (Tan et al., 2006), the total number of solutions (when dealing with exact methods) (Visée et al., 1998), the generational distance (Jozefowiez et al., 2007a; Knowles, 2002; Mateo and Alberto, 2012), the D_2 metric (Ulungu et al., 1999) and hyperarea metric (Collette and Siarry, 2003).

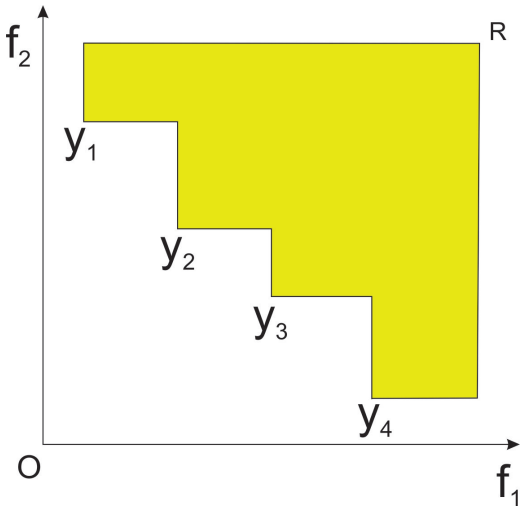

Figure 2: Example of the $\mathcal{S}$ metric

Two of these metrics have been used to evaluate algorithms that attempt to solve the VRPRB (Jozefowiez et al., 2009, 2007b), $\mathcal{C}$ metric and $\mathcal{S}$ metric. For that reason they seem to be a good choice to evaluate the performance of the proposed algorithm. However straight comparison with the results obtained in previous research was not possible since computing $\mathcal{S}$ metric requires knowing the reference points, and the $\mathcal{C}$ metric requires the whole approximation of the Pareto set. This information was not available. The $\mathcal{S}$ metric measures the area dominated by a set of solutions, given a reference point, as Figure 2 shows. The reference points used for computing the metric will be reported, to allow further comparison with other results. For every instance, the worst value found for each objective function was the one selected to be used as reference point. A set $\mathcal{R}$ has a better performance than a set $\mathcal{X}$, if the $\mathcal{S}$ metric of the former one is greater.

Given two sets of solutions $(\mathcal{R}, \mathcal{X})$, the $\mathcal{C}$ metric (Jozefowiez et al., 2009, 2007b; Knowles, 2002; Mateo and Alberto, 2012) measures the ratio of solutions in $\mathcal{X}$ weakly dominated by solutions in $\mathcal{R}$. The metric is not symmetric, so both $\mathcal{C}(\mathcal{R}, \mathcal{X})$ and $\mathcal{C}(\mathcal{X}, \mathcal{R})$ should be computed. This metric is not reliable if the cardinality of the sets is different. In addition it is not able to measure by how much a set outperforms another one (Knowles, 2002). A set $\mathcal{R}$ has a better performance than a set $\mathcal{X}$, if $\mathcal{C}(\mathcal{R}, \mathcal{X})$ is closer to 1 and $\mathcal{C}(\mathcal{X}, \mathcal{R})$ is closer to 0.

4.3 Results

Results comparing GRASP-ASP and WSO-G are presented in Tables 1 and 2. In both cases, the average, minimum and maximum values were computed from the results obtained in the ten different runs. In Table 1, numbers are rounded to the nearest integer. Considering the $\mathcal{S}$ metric, GRASP-ASP algorithm is able to find a better approximation of the Pareto set in 19 out of 20 instances. This is just if the average value is considered; if the maximum value is taken into account, then GRASP-ASP manages to find in every instance at least one approximation of the Pareto set with a better performance.

A comparison using the $\mathcal{C}$ metric is found in Table 2. In average, GRASP-ASP performs better in all instances, however in the case of instance E021-04m, at least one of the sets found by GRASP-ASP is not better than the one found by GUROBI.

Table 1: $\mathcal{S}$ metric performance indicator

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E016-03m	(476.26, 114.56)	WSO-G	22 302	22 302	22 302
		GRASP-ASP	**22 314**	21 771	**22 418**
E016-05m	(499.29, 17.20)	WSO-G	**1 951**	1 951	1 951
		GRASP-ASP	1 885	1 147	**1 977**
E021-04m	(743.59, 119.00)	WSO-G	44 518	44 518	44 518
		GRASP-ASP	**44 712**	44 558	**44 852**
E021-06m	(737.50, 104.00)	WSO-G	28 467	28 467	28 467
		GRASP-ASP	**28 900**	28 494	**29 148**
E022-06m	(1 004.30, 133.00)	WSO-G	59 406	59 406	59 406
		GRASP-ASP	**62 736**	60 512	**63 483**
E023-05S	(1 469.51, 290.11)	WSO-G	206 324	206 324	206 324
		GRASP-ASP	**218 625**	217 432	**219 412**
E026-08m	(935.02, 102.00)	WSO-G	23 856	23 856	23 856
		GRASP-ASP	**27 884**	26 005	**28 528**
E030-04S	(1 716.44, 137.76)	WSO-G	152 771	152 771	152 771
		GRASP-ASP	**157 430**	157 128	**157 690**
E031-09h	(1 102.78, 56.70)	WSO-G	5 219	5 219	5 219
		GRASP-ASP	**21 292**	19 740	**22 617**
E033-05s	(1 766.32, 267.88)	WSO-G	180 801	180 801	180 801
		GRASP-ASP	**206 117**	196 722	**207 823**
E036-11h	(1 181.39, 122.00)	WSO-G	27 724	27 724	27 724
		GRASP-ASP	**51 198**	48 311	**52 361**
E041-14h	(1 404.02, 116.00)	WSO-G	22 390	22 390	22 390
		GRASP-ASP	**47 957**	42 612	**51 911**
E048-04y	(167 771.00, 15 705.00)	WSO-G	19.01×10^8	19.01×10^8	19.01×10^8
		GRASP-ASP	$\mathbf{19.83\times10^8}$	19.78×10^8	$\mathbf{19.87\times10^8}$
E051-05e	(1 495.14, 38.38)	WSO-G	14 450	14 450	14 450
		GRASP-ASP	**36 581**	36 444	**36 788**
E076-07s	(3 318.56, 125.97)	WSO-G	187 882	187 882	187 882
		GRASP-ASP	**325 757**	324 379	**327 001**
E076-08s	(2 365.50, 100.64)	WSO-G	39 166	39 166	39 166
		GRASP-ASP	**154 780**	151 516	**157 341**
E076-14u	(2 156.79, 113.85)	WSO-G	26 848	26 848	26 848
		GRASP-ASP	**66 179**	54 537	**69 366**
E101-08e	(1 698.22, 156.97)	WSO-G	20 492	20 492	20 492
		GRASP-ASP	**123 763**	120 693	**126 331**
E101-10c	(1 539.17, 134.05)	WSO-G	0	0	0
		GRASP-ASP	**75 345**	72 143	**79 956**
E101-14s	(2 518.35, 139.58)	WSO-G	1 591	1 591	1 591
		GRASP-ASP	**151 681**	144 664	**155 792**

Table 2: $\mathcal{C}$ metric performance indicator

Instance	Metric	Average	Min	Max
E016-03m	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.24	0.00	0.57
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.50**	0.20	**0.60**
E016-05m	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.10	0.00	0.33
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.34**	0.20	**0.40**
E021-04m	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.27	0.11	**0.40**
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.32**	0.17	0.33
E021-06m	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.15	0.10	0.26
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.27**	0.14	**0.43**
E022-06m	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.06	0.00	0.55
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.50**	0.29	**0.57**
E023-05S	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.08	0.00	0.09
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.11**	0.00	**0.25**
E026-08m	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.02	0.00	0.18
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.90**	0.60	**1.00**
E030-04S	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.59**	0.50	**0.75**
E031-09h	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E033-05s	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.89**	0.86	**1.00**
E036-11h	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E041-14h	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E048-04y	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E051-05e	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E076-07s	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E076-08s	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E076-14u	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**0.95**	0.50	**1.00**
E101-08e	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E101-10c	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**
E101-14s	$\mathcal{C}$(WSO-G, GRASP-ASP)	0.00	0.00	0.00
	$\mathcal{C}$(GRASP-ASP, WSO-G)	**1.00**	1.00	**1.00**

Table 3: Summary of performance indicators

Instance	$\mathcal{S}$(GRASP-ASP) - $\mathcal{S}$(WSO-G)	$\mathcal{C}$(GRASP-ASP, WSO-G) - $\mathcal{C}$(WSO-G, GRASP-ASP)
E016-03m	12	0.26
E016-05m	-66	0.24
E021-04m	193	0.05
E021-06m	433	0.12
E022-06m	3 329	0.44
E023-05S	12 301	0.03
E026-08m	4 028	0.88
E030-04S	4 659	0.59
E031-09h	16 073	1.00
E033-05S	25 316	0.89
E036-11h	23 474	1.00
E041-14h	25 567	1.00
E048-04y	82 730 000	1.00
E051-05e	22 131	1.00
E076-07s	137 875	1.00
E076-08s	115 615	1.00
E076-14u	39 331	0.95
E101-08e	103 271	1.00
E101-10c	75 345	1.00
E101-14s	150 090	1.00

As an illustration, a summary of the performance indicators is presented in Table 3. The difference between the average $\mathcal{S}$(GRASP-ASP) and the average $\mathcal{S}$(WSO-G) corresponds to column 2, where a positive value shows and average better performance of GRASP-ASP in the tests. Column 3 shows the difference between $\mathcal{C}$(GRASP-ASP, WSO-G) and $\mathcal{C}$(WSO-G, GRASP-ASP). Positive values, specially values closer to one indicates an average better performance of GRASP-ASP in the tests.

Figure 3 shows a comparison between PS_{apr}(WSO-G) and PS_{apr}(GRASP-ASP). Solutions generated by GRASP-ASP can be located in the concavities of the approximation of the Pareto set. On the other hand the WSO cannot find solutions in concavities (Collette and Siarry, 2003), which gives an advantage to GRASP-ASP.

For the instance E101-10c, WSO-G was able to find just two solutions with a very low performance. That explain why the $\mathcal{S}$ metric is equal to zero, as it can be seen in Table 1.

4.4 Comparison with an evolutionary algorithm

An additional comparison is done. An evolutionary algorithm (EA) proposed for the VRPRB (Jozefowiez et al., 2009) was implemented in C++. The original algorithm is presented as parallel, however the implementation done for this testing is serial. The performance should not be affected, since all the features and solutions exchange were included. In principle the only aspect that should be affected by this change is the running time. 8 threads are used in the original version of the algorithm, so the serial version should run 8 times slower. The parameters used for the testings were the same as in the original version of the algorithm.

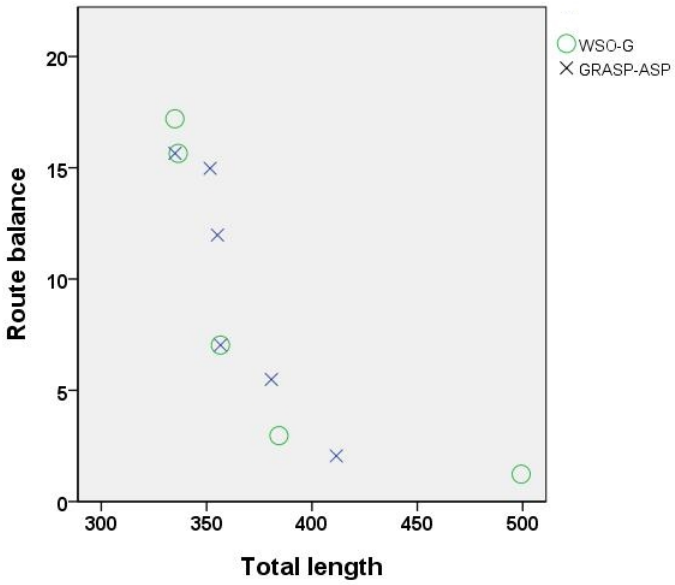

Figure 3: Example of results obtained by GRASP-ASP and WSO-G for instance E016-05m

Ten different runs of the evolutionary algorithm per instance were started. However limited time did not admit all tests to complete. All obtained results are reported, Table 4 shows the comparison between GRASP-ASP and EA. Table 5 shows the comparison regarding the the $\mathcal{C}$ metric. To make a fair comparison, the final approximation of the Pareto set was reduced to 11 solutions. The method used to do that was the *average linking method* (Morse, 1980), which is used in EA to cluster sets of solutions.

A summary of the performance indicators is presented in Table 6. In three instances every indicator shows in average a different result, but even though in average the $\mathcal{S}$ metric indicates that EA performs better, it also shows that GRASP-ASP is able to find the set of solutions with the best performance, as reported in Table 4. In four of the instances GRASP-ASP has a better performance, in average. In eleven instances, EA has a better performance in average, but GRASP-ASP is able to find a set of solutions with the best performance for one them.

The average total running time, as well as the number of tests completed by instance (in parenthesis), are reported in Table 7. There is a clear difference in the running time of GRASP-ASP and EA. For example, average running times for instance E016-03m were 251 and 101 756 seconds for GRASP-ASP and EA respectively. The difference becomes bigger with the instance size, for the instance E076-14u the running times are 5 381 and 1 041 506 seconds.

Table 4: $\mathcal{S}$ metric performance indicator for GRASP-ASP and EA

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E016-03m	(395,36 , 46,53)	EA	**5 224**	5 224	5 228
		GRASP-ASP	5 127	4 589	**5 229**
E016-05m	(421,72 , 17,20)	EA	**776**	776	776
		GRASP-ASP	747	277	**835**
E021-04m	(590,07 , 68,62)	EA	**14 964**	14 937	**15 002**
		GRASP-ASP	14 886	14 754	14 987
E021-06m	(655,09 , 70,22)	EA	**13 337**	13 337	**13 337**
		GRASP-ASP	13 148	13 059	13 240
E022-06m	(785,06 , 92,66)	EA	22 417	22 274	22 495
		GRASP-ASP	**22 493**	20 421	**23 031**
E023-05S	(1270,71 , 290,11)	EA	157 295	154 102	158 728
		GRASP-ASP	**161 419**	160 383	**161 908**
E026-08m	(935,02 , 72,22)	EA	**18 563**	18 453	18 608
		GRASP-ASP	18 134	16 255	**18 778**
E030-04S	(1080,00 , 137,76)	EA	68 077	67 912	68 242
		GRASP-ASP	**69 771**	69 462	**70 042**
E031-09h	(993,74 , 56,70)	EA	**17 938**	17 702	**18 287**
		GRASP-ASP	15 878	14 325	16 743
E033-05S	(1337,07 , 266,66)	EA	89 401	88 660	89 936
		GRASP-ASP	**90 549**	81 166	**92 236**
E036-11h	(1181,37 , 66,52)	EA	**27 730**	27 583	**27 826**
		GRASP-ASP	24 656	21 871	25 681
E041-14h	(1404,02 , 75,70)	EA	**35 067**	34 639	**35 431**
		GRASP-ASP	27 079	21 774	30 401
E048-04y	(90826,00 , 15672,10)	EA	$\mathbf{7.77\times10^8}$	7.76×10^8	$\mathbf{7.77\times10^8}$
		GRASP-ASP	7.74×10^8	7.68×10^8	7.77×10^8
E051-05e	(1033,23 , 21,52)	EA	**10 379**	10 344	10 417
		GRASP-ASP	10 334	10 218	**10 552**
E076-07s	(1594,61 , 123,96)	EA	**109 963**	109 153	**110 772**
		GRASP-ASP	107 004	105 546	108 217
E076-08s	(1526,72 , 100,64)	EA	**77 401**	77 051	**77 852**
		GRASP-ASP	70 484	67 221	73 059
E076-14u	(2156,79 , 113,85)	EA	**121 713**	121 288	**122 138**
		GRASP-ASP	66 179	54 537	69 366
E101-08e	(1698,22 , 104,47)	EA	**88 710**	88 710	**88 710**
		GRASP-ASP	78 383	75 712	81 023

Table 5: $\mathcal{C}$ metric performance indicator for GRASP-ASP and EA

Instance	Metric	Average	Min	Max
E016-03m	$\mathcal{C}$(EA, GRASP-ASP)	0.52	0.14	**0.86**
	$\mathcal{C}$(GRASP-ASP, EA)	**0.60**	0.14	0.71
E016-05m	$\mathcal{C}$(EA, GRASP-ASP)	0.76	0.67	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	**0.80**	0.13	**1.00**
E021-04m	$\mathcal{C}$(EA, GRASP-ASP)	**0.64**	0.44	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.47	0.27	0.82
E021-06m	$\mathcal{C}$(EA, GRASP-ASP)	**0.86**	0.70	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.58	0.45	0.64
E022-06m	$\mathcal{C}$(EA, GRASP-ASP)	0.21	0.09	0.64
	$\mathcal{C}$(GRASP-ASP, EA)	**0.42**	0.09	**0.73**
E023-05S	$\mathcal{C}$(EA, GRASP-ASP)	**0.05**	0.00	0.27
	$\mathcal{C}$(GRASP-ASP, EA)	0.20	0.09	**0.40**
E026-08m	$\mathcal{C}$(EA, GRASP-ASP)	0.36	0.18	0.82
	$\mathcal{C}$(GRASP-ASP, EA)	**0.73**	0.36	**0.91**
E030-04S	$\mathcal{C}$(EA, GRASP-ASP)	0.11	0.09	0.18
	$\mathcal{C}$(GRASP-ASP, EA)	**0.49**	0.27	**0.73**
E031-09h	$\mathcal{C}$(EA, GRASP-ASP)	**0.54**	0.18	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.03	0.00	0.09
E033-05S	$\mathcal{C}$(EA, GRASP-ASP)	0.15	0.09	**0.36**
	$\mathcal{C}$(GRASP-ASP, EA)	**0.22**	0.09	**0.36**
E036-11h	$\mathcal{C}$(EA, GRASP-ASP)	**0.53**	0.18	**0.73**
	$\mathcal{C}$(GRASP-ASP, EA)	0.02	0.00	0.09
E041-14h	$\mathcal{C}$(EA, GRASP-ASP)	**0.92**	0.73	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.00	0.00	0.00
E048-04y	$\mathcal{C}$(EA, GRASP-ASP)	**0.38**	0.09	**0.64**
	$\mathcal{C}$(GRASP-ASP, EA)	0.19	0.00	0.45
E051-05e	$\mathcal{C}$(EA, GRASP-ASP)	**0.29**	0.09	**0.55**
	$\mathcal{C}$(GRASP-ASP, EA)	0.28	0.00	**0.55**
E076-07s	$\mathcal{C}$(EA, GRASP-ASP)	**0.74**	0.45	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.07	0.00	0.18
E076-08s	$\mathcal{C}$(EA, GRASP-ASP)	**0.93**	0.89	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.02	0.00	0.09
E076-14u	$\mathcal{C}$(EA, GRASP-ASP)	**0.99**	0.89	**1.00**
	$\mathcal{C}$(GRASP-ASP, EA)	0.00	0.00	0.00
E101-08e	$\mathcal{C}$(EA, GRASP-ASP)	**0.84**	0.64	**0.91**
	$\mathcal{C}$(GRASP-ASP, EA)	0.00	0.00	0.00

Table 6: Summary of performance indicators for GRASP-ASP and EA

Instance	$\mathcal{S}$(GRASP-ASP) - $\mathcal{S}$(EA)	$\mathcal{C}$(GRASP-ASP, EA) - $\mathcal{C}$(EA, GRASP-ASP)
E016-03m	-97	0.08
E016-05m	-29	0.05
E021-04m	-78	-0.17
E021-06m	-189	-0.27
E022-06m	76	0.21
E023-05S	4 124	0.15
E026-08m	-429	0.37
E030-04S	1 694	0.38
E031-09h	-2 060	-0.50
E033-05S	1 148	0.07
E036-11h	-3 074	-0.52
E041-14h	-7 988	-0.92
E048-04y	-7 322 000	-0.19
E051-05e	-45	-0.01
E076-07s	-2 959	-0.66
E076-08s	-6 917	-0.91
E076-14u	-55 534	-0.99
E101-08e	-10 327	-0.84

Table 7: Average running time (in seconds)

Instance	WSO-G	GRASP-ASP	EA (average time and number of tests)
E016-03m	11 729	251	101 756 (10)
E016-05m	14 373	295	112 648 (8)
E021-04m	118 800	381	152 411 (10)
E021-06m	118 801	391	162 731 (2)
E022-06m	118 801	855	235 828 (6)
E023-05S	89 360	546	411 502 (7)
E026-08m	118 801	1 569	217 385 (7)
E030-04S	118 801	653	744 485 (2)
E031-09h	118 801	2 420	369 201 (4)
E033-05s	118 802	2 121	842 680 (4)
E036-11h	118 803	2 638	428 237 (3)
E041-14h	118 802	2 999	500 168 (5)
E048-04y	118 802	1 987	896 509 (4)
E051-05e	118 802	2 335	570 281 (5)
E076-07s	119 008	4 026	1 142 078 (2)
E076-08s	118 813	4 201	909 250 (3)
E076-14u	10 800	5 381	1 041 506 (2)
E101-08e	119 002	5 043	1 859 921 (1)
E101-10c	118 908	7 081	-
E101-14s	118 908	3 283	-

5 Conclusions and further research

A new bi-objective optimization algorithm was proposed and tested. Compared to a fairly simple, but reasonable solution method (WSO-G), the GRASP-ASP results are clearly superior, measured by $\mathcal{S}$ and $\mathcal{C}$ metrics. In general, GRASP-ASP is able to find a set PS_{apr} that performs better than the set PS_{apr}(WSO-G) found by WSO-G, even for instances where WSO-G finds solutions within 0.01% MIP gap. The running time, in Table 7, is much lower than the time used by WSO-G.

The weighted sum of objective function method does not find *non-supported efficient solutions* i.e. solutions located in concavities of the Pareto set (Visée et al., 1998). In GRASP-ASP no weights are assigned to the objective functions in the MOP, which means that solutions in concavities (non supported) may be eventually found. This can partially explain the fact that the set PS_{apr}(GRASP-ASP) has a better performance than the set PS_{apr}(WSO-G). An additional aspect to consider is that the mixed integer programming (MIP) optimality gap is set equal to 0.01%, which may also explain how GRASP-ASP is able to find a better set of solutions then WSO-G. With a tighter MIP gap, GUROBI often does not terminate.

The fact that the GRASP-ASP procedure is repeated using the updated PS_{pro}, seems to have a positive impact in the quality of the final potentially Pareto set. Here it might have some similarities with evolutionary algorithms, but GRASP-ASP involves less randomness in the process of building the new solutions. Perhaps this can have some connection with the fact that GRASP-ASP is able to find better/similar solutions for some instances using a much shorter running time than an evolutionary algorithm for the VRPRB.

Given the promising results in this paper, a possible direction for further research could be to apply GRASP-ASP to other bi-objective problems, perhaps general multi-objective problems. Another option could be to apply it in an stochastic extension of VRPRB. Even more elaborated methods of clustering could be implemented for reducing/selecting the set of solutions, since the quality of the resulting potentially Pareto solutions may be affected.

Acknowledgement

This paper is dedicated to the memory of Arne Løkketangen. It has benefited from comments made by Halvard Arntzen (Molde University College), David Woodruff (University of California, Davis) and two anonymous referees.

References

Collette, Y. and Siarry, P. (2003). *Multiobjective optimization: principles and case studies*. Springer, Berlin.

Cordeau, J.-F., Laporte, G., and Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *The Journal of the Operational Research Society*, 52(8):928–936.

Current, J. and Min, H. (1986). Multiobjective design of transportation networks: Taxonomy and annotation. *European Journal of Operational Research*, 26(2):187 – 201.

Desrochers, M., Lenstra, J., Savelsbergh, M., and Soumis., F. (1988). Vehicle routing with time windows: optimization and approximation. In Golden, B. L. and Assad, A. A., editors, *Vehicle routing: methods and studies*.

Glover, F. (1989). Tabu search - part I. *ORSA Journal on Computing*, 1:190–206.

Glover, F. and Laguna, M. (1997). *Tabu search*. Kluwer Academic, Boston.

Jacobs, L. W. and Brusco, M. J. (1995). A local-search heuristic for large set-covering problems. *Naval Research Logistics (NRL)*, 42(7):1129–1140.

Jozefowiez, N., Semet, F., and El-Ghazali, T. (2007a). The bi-objective covering tour problem. *Computers & Operations Research*, 34(7):1929–1929.

Jozefowiez, N., Semet, F., and Talbi, E. (2009). An evolutionary algorithm for the vehicle routing problem with route balancing. *European Journal of Operational Research*, 195(3):761–769.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2002). *Parallel Problem Solving from Nature VII, Lecture Notes in Computer Science*, volume 2439, chapter Parallel and hybrid models for multi-objective optimization: Application to the vehicle routing problem, pages 271–280. Springer-Verlag.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2007b). Target aiming pareto search and its application to the vehicle routing problem with route balancing. *Journal of Heuristics*, 13(5):455–469.

Knowles, J. D. (2002). *Local-search and hybrid evolutionary algorithms for Pareto optimization*. PhD thesis, University of Reading.

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345 – 358.

Lee, T.-R. and Ueng, J.-H. (1999). A study of vehicle routing problems with load-balancing. *International Journal of Physical Distribution & Logistics Management*, 29(10):646 – 657.

Lozano, M., Glover, F., García-Martínez, C., Rodríquez, F. J., and Martí, R. (2003). Tabu search with strategic oscillation for the quadratic minimum spanning tree. *IIE Transactions*.

Mateo, P. M. and Alberto, I. (2012). A mutation operator based on a pareto ranking for multi-objective evolutionary algorithms. *Journal of Heuristics*, 18:53–89.

Minella, G., Ruiz, R., and Ciavotta, M. (2011). Restarted iterated pareto greedy algorithm for multi-objective flowshop scheduling problems. *Computers & Operations Research*, (38):1521–1533.

Morse, J. N. (1980). Reducing the size of the nondominated set: pruning by clustering. *Computers & Operations Research*, (7):55–66.

Operations Research Group - Library of Instances (2012). Accessed on 20-06-2012. `http://www.or.deis.unibo.it/research_pages/ORinstances/VRPLIB/VRPLIB.html`.

Oppen, J. and Løkketangen, A. (2006). Arc routing in a node routing environment. *Computers & Operations Research*, 33(4):1033 – 1055. Part Special Issue: Optimization Days 2003.

Resende, M. and Ribeiro, C. (2003). *Handbook of Metaheuristics*, chapter Greedy randomized adaptive search procedures, pages 219–250. Kluwer Academic Publishers.

Resende, M. and Ribeiro, C. (2010). Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau, J.-Y. P., editor, *Handbook of Metaheuristics*, pages 283–319. Springer.

Reynolds, A. P., Corne, D. W., and de la Iglesia, B. (2009). A multiobjective grasp for rule selection. In *Proceedings of the 11th Annual conference on genetic and evolutionary computation (GECCO), Montreal*.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., and Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 2(159):139–171.

Tan, K., Chew, Y., and Lee, L. (2006). A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172(3):855 – 885.

Toth, P. and Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1-3):487 – 512.

Ulungu, E. L., Teghem, J., Fortemps, P. H., and Tuyttens, D. (1999). Mosa method: a tool for solving multiobjective combinatorial optimization problems. *Journal of Multicriteria Decision Analysis*, 8(4):221–221.

Vianna, D. S. and Arroyo, J. E. C. (2004). A grasp algorithm for the multi-objective knapsack problem. In *Proceedings of the 24th International Conference of the Chilean Computer Science Society*.

Visée, M., Teghem, J., Pirlot, M., and Ulungu, E. (1998). Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem. *Journal of Global Optimization*, 12(2):139–155.

Zachariadis, E. E., Tarantilis, C. D., and Kiranoudis, C. T. (2009). A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research*, 195(3):729 – 743.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm. Technical report, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institue of technology (ETH).

Paper 3

The stochastic vehicle routing problem, a literature review

The stochastic vehicle routing problem, a literature review

Jorge Oyola[1], Halvard Arntzen[1] and David L. Woodruff[2]

[1]Molde University College, Molde, Norway
[2]Graduate School of Management, UC Davis, Davis CA, USA

Abstract

Building on the work of Gendreau, Laporte, and Seguin (1996), we review the past 20 years of scientific literature on stochastic vehicle routing problems (SVRP). The numerous variants of the problem that have been studied in the literature are described and categorized. Also a thorough review of solution methods applied to the SVRP is included as an Appendix.

Keywords: Stochastic Vehicle Routing Problem (SVRP), Survey, Solution methods for SVRPs

1 Introduction

Vehicle routing problems concern the challenge of selecting a set of routes for a fleet of vehicles to serve the demands of a set of customers. Almost invariably, the vehicles have limitations on the amount of goods they can carry, and the primary goal of the decision maker (DM) is most often to minimize the total transportation cost. It thus makes sense to use the formulation of the (deterministic) *capacitated vehicle routing problem* (CVRP) as a point of departure for this review of the literature for stochastic variants.

The CVRP is defined over an undirected graph $G(V, E)$, where $V = v_0, \ldots, v_N$ is a set of vertices and $E = (v_i, v_j) : v_i, v_j \in V, i < j$ is a set of edges. There is a symmetric matrix $C = [c_{ij}]$ that correspond to the travel costs along edge (v_i, v_j). Vertex v_0 represents the depot where there is a homogeneous fleet of m vehicles with capacity Q. A set of customers $V \setminus v_0$ with a non-negative known demand d_i must be served. A solution to the CVRP consists of m delivery routes starting and ending at the depot. Each customer must be visited once by exactly one vehicle. The summation of the demands of the customers in the same route, must be less than or equal to the vehicle's capacity. A different approach where the demand corresponds to items that must be collected from the customers leads to an equivalent problem. The classic objective is minimization of total route cost (see, e.g., Toth and Vigo, 2002) but some formulations minimize total route length, total travel time or total cost.

In the real world one or more of the elements of the CVRP are uncertain. In order to model this, one typically allows some parameters in the general formulation to be represented as stochastic. When stochastic data are included into the problem, we have a *stochastic (capacitated) vehicle routing problem* (SVRP or SCVRP). Gendreau et al. (2014) provide a tutorial with a synthesis of some recent literature. A thorough review of the early literature on the SVRP, including a concise description of relevant solution concepts is found in Gendreau et al. (1996a). We provide a brief recap of some details here before launching into a thorough review of papers since then.

Although demand, the presence of customers, travel times, and service times are sometimes modeled as stochastic (Gendreau et al., 1996a; Tan et al., 2007), the most studied version of SVRP is the capacitated vehicle routing problem with stochastic demand (CVRPSD) Cordeau et al. (2007). This is a feature of many real life problems (see, e.g., Yang et al., 2000).

There is a striking difference between deterministic and stochastic VRP formulations: For all SVRP variants, the DM must decide the solution (at least partially) *before* the exact values of all parameters are completely known. In some situations, constraints may be violated when the actual parameter values are realized, e.g. the total realized demand of a planned route may actually exceed the vehicle's capacity. One can say that the solution (or the route) "fails" when it is exercised with the realized data. In a deterministic problem the DM has complete information when making the plans, so there is no similar concept of a solution "failing". There are two common ways of modeling stochastic problems: as a *chance constrained program* (CCP) or as a *stochastic program with recourse* (SPR).

In the case of CCP, the problem is solved ensuring that the probability of route failure is below a certain level and the cost of failures is typically ignored (Gendreau et al., 1996a; Tan et al., 2007). Although chance constrained problems can be formulated with an expected value objective, in the SVRP literature, the objective is typically deterministic. Consider a very abstract formulation where the objective function is $f(x)$ for a decision vector x and constraints are summarized by a set $\mathcal{X}$. We can then write a chance constrained program as:

$$\min_{x} \quad f(x) \text{ subject to } \text{Prob}(x \in \mathcal{X}) \geq 1 - \alpha$$

where the DM provides a parameter value α giving the acceptable probability of failing to meet the constraints. Of course, in less abstract formulations, the specific constraints that are subject to failure are specified.

On the other hand, in SPR, one allows route failures, but the DM must define a *recourse* policy, describing what actions to take in order to repair the solution after a failure. The expected transportation cost (travel cost + recourse policies cost) is optimized. SPR is more difficult to solve, but objectives are more meaningful (Gendreau et al., 1996a).

The recourse policy is a modeling choice leading to different variants of an SVRP formulation. For the CVRPSD, three common recourse policies are (Tan et al., 2007; Secomandi and Margot, 2009):

- The vehicle returns to depot to restock when capacity is attained or exceeded. Service resumes at the customer where route failure occurred. This is known as detour to depot (DTD).

- A preventive restocking can be done *before* a route failure occurs. Obviously, it may be less costly to travel to the depot to restock from the actual location than waiting for a route failure at a location further from the depot.

- After failure or after each customer is served and its demand becomes known, the portion of a route that has not been served is re-optimized. A decision is taken regarding which customer must be visited next, either as part of the regular routing or on the way to replenishment at the depot.

In other SVRP formulations the recourse policy does not involve routing decisions (as above), but a penalty for late/early arrivals or the extra time cost of the driver can be part of the expected cost when time windows and/or stochastic service time are taken into consideration (see, e.g., Li et al., 2010; Taş et al., 2013).

In the presence of stochastic data, any function of the data, such as a classic total cost objective function, will be a random variable, so some choice must be made to form a well-posed objective function. Most of the problems found in the SVRP literature can be cast as two-stage stochastic programming problems that minimize expected value. An abstract formulation (Birge and Louveaux, 1997) is as follows:

$$\min_{x} \quad f(x) + E[Q(x, \boldsymbol{\xi})] \text{ subject to } x \in \mathcal{X}$$

where $Q(x, \xi)$ is the optimal value of the second-stage problem

$$\min_{y} \quad q(y; x, \xi) \text{ subject to } y \in \mathcal{Y}(x, \xi)$$

Here x represents the first stage decisions that must be taken initially, before all information is available. In most formulations these are routing decisions. The function $f(\cdot)$ evaluates the objective function for the first stage portion of the decisions. The random variables that make the problem stochastic are represented by $\boldsymbol{\xi}$. These may be pickup quantities or travel times, etc. In general, $\boldsymbol{\xi}$ and its realizations ξ are vector-valued. The second stage, or *recourse*, decisions are represented by y, which is evaluated using the function $q(\cdot)$ that can be parameterized by x and ξ. For some problems, this is simply a calculation of cost but in other cases a significant minimization is required. In most of the literature $\boldsymbol{\xi}$ is discrete, so the expectation is computed using a sum.

In this abstract formulation we have summarized constraints using the sets $\mathcal{X}$ and $\mathcal{Y}$. Most of the formulations in the literature are constructed so that there is *relatively complete recourse* (Kall and Wallace, 1994), which in this formulation means that $\mathcal{Y}(x, \xi)$ is non-empty for every $x \in \mathcal{X}$ and every ξ with non-zero probability.

A focus of two-stage formulations is the need to compute the first stage decisions, x, with the second stage decisions y used to compute, or estimate, an appropriate second stage expected cost. However, we note that a few papers in the literature seek methods for finding a good policy (dynamic programming approaches) or to provide an algorithm for routing vehicles dynamically (rollout algorithms) in addition to a good first stage decision.

One should note that SVRP papers usually come in one of two standard forms. On one hand there are researchers who are mainly interested in the modeling of SVRPs. They will need to formulate precise mathematical models describing what is understood by a solution x, and algebraic expressions for objective functions and constraints. On the other hand, we see researchers who are most interested in algorithmic issues. In an "algorithmic" paper, the model formulation is often not given algebraically. While input parameters are usually defined, there is often no mathematical formulations of either objectives or constraints, and a "solution" can be defined as "a set of routes" without further precision.

Evaluating the quality (i.e. the objective value) of a solution to a SVRP is not straight forward. Several approaches have been used to tackle this issue. In some cases a simulation is performed to generate a large number of possible realizations (scenarios), and then the solution is evaluated on each realization, getting an estimation of the quality (see, e.g., Juan et al., 2011).

The quality of the solution is sometimes possible to calculate analytically, given certain characteristics of the problem (see, e.g., Laporte et al., 2010). A dynamic programming recursion can be also used for evaluating solutions (see, e.g., Yang et al., 2000).

This survey proceeds in Section 2 with an overview of problem types found in the SVRP literature. In Appendix A we summarize the various solution methods applied for solving SVRP variants. Tables summarizing the literature are presented in Section 3. The paper closes with some conclusions and connections with related literature.

2 Types of problems

The stochasticity can be incorporated in the problem through different aspects and, typically, one or two elements are considered as stochastic. This limitation is likely due to the difficulty of solving a problem where many different parameters are stochastic. A summary of the different problems that have been studied is presented here.

2.1 The CVRP with stochastic demand - CVRPSD

In this version of the SVRP, the customer demands are stochastic and become known only after the routes have been established. The problem is usually modeled as a two-stage SPR.

A study of the basic CVRPSD is found in Laporte et al. (2002) where an SPR formulation is used with recourse action DTD. Two types of demand distributions are considered theoretically, Poisson and normal. Computational tests are done for both cases. The same problem is found in Jabali et al. (2014) where theoretically the demands are treated as independent and identically distributed (IID), however, tests are done using a normal distribution truncated at zero.

In the multi-compartment VRP with stochastic demands (Mendoza et al., 2010, 2011), each customer has a stochastic demand for different products, which follows a known probability distribution. Such products need to be transported in independent compartments in the vehicles. The recourse action is to travel to the depot once the capacity of any of the compartments in the vehicle is reached. Although the problem does not assume a particular probability distribution for the demand, computational tests were performed on instances with demands following a normal probability distribution.

A more recent paper by Goodson (2015) deals with a similar problem, where each route is subject to a route duration limit L. A method for computing the expected cost of the solutions is proposed, however it works only with discrete distributions. Due to this restriction a different method was used in (Mendoza et al., 2010, 2011), since results available for comparison were obtained assuming demands with normal distribution.

The stochastic CVRP with restocking (Yang et al., 2000; Marinakis et al., 2013) gives the option after each visit to choose between visiting the next customer in the route or traveling back to the depot to restock, even if the vehicle capacity has not been reached. This problem was formulated as a SPR in Yang et al. (2000) where the recourse action was to travel to the depot and restock, continuing with the planned route afterwards. The demand was assumed to be discrete, test instances were generated using a discrete triangular distribution. In Secomandi (2003) the single vehicle CVRPSD is studied in two versions, allowing restocking and with no restocking. In both cases the demand is assumed to be discrete, having instances with uniform discrete distribution.

A policy-based solution approach was taken in Marinakis et al. (2013). In this work, route failures are not permitted, assuming that these can be avoided by selecting a threshold value, such that when the residual load in a vehicle is less than or equal to the threshold, the vehicle should go to the depot for preventive restocking. This will only work under bounding conditions on the probability distributions. The only additional assumption regarding the probability distribution of the demands is that they are known and independent. For the computational tests, the demands follow discrete uniform probability distributions.

A different approach to handling the dynamics uses re-optimization; after visiting a customer the driver decides which customer to visit next, either directly or after replenishing at the depot. The decision is taken on basis of the available capacity and the expected demand of the unvisited customers. This problem was studied for the particular case of a single vehicle (Secomandi, 2000, 2001; Secomandi and Margot, 2009), where the probability distributions of the demands are discrete and independent (discrete uniform in one of the cases Secomandi (2000)). The problem was modeled as SPR, the recourse action is DTD. The computational tests are performed over instances with demands following discrete uniform probability distributions.

The single vehicle case of the CVRPSD has also been studied using a regular recourse action DTD, involving two different types of stockout Hjorring and Holt (1999). A *normal stockout*, means the vehicle does not have enough goods to serve a customer. After restocking at the depot, the route is resumed starting with the customer where the failure occurred. An *exact stockout* means the vehicle have just enough goods to satisfy the demand of a particular customer. After restocking it will resume the trip at the next customer in the route. The proposed approach may apply for several probability distributions, but tests are performed using a discrete distribution. The same problem was studied following a different approach Rei et al. (2010), where the demands are considered to have a known probability distribution. The testing was performed on instances with demands that follow normal probability distributions. Another version of the single vehicle case, where demands are a normal random variable truncated at zero, was also studied Rei et al. (2007). In the latter case, if a failure occurs, then partial delivery is performed and the vehicle returns to the depot to restore capacity. In the cases where demands follow a continuous probability distribution, exact stockouts are not considered.

An interesting variant of the single vehicle CVRPSD allows preventive restocking (Bianchi et al., 2005). In this case, the vehicle can travel to depot before the next customer in the route to restock, even if a route failure has not occurred. The demands are modeled as random variables with integer uniform distribution. Two ways of evaluating the objective function are considered, a dynamic programming recursion (Yang et al., 2000) and an approximation with the length of the *a priori* tour, without considering the stockout cost and the preventive restocking cost.

Another variant of the single vehicle CVRPSD was modeled considering that after a failure, no action is taken, and unserved customers will not be serviced (Chepuri and Homem-de Mello, 2005). A penalty must be paid to the customer where the failure occurs and to the other unserved customers in the route, since the vehicle will not resume the route. The authors claim that this is the situation in several industries, where failures may result in lost revenue or emergency deliveries. They were motivated by a liquid air distributor. Demands are considered to follow a gamma distribution. The methods can exploit the special situation where the parameters of the probability distribution are the same for all customers.

Another problem that has been studied is the combination of routing plus clustering (designing delivery districts) (Haugland et al., 2007). In this case the solution to the problem will have m contiguous districts, where all customers in the same district are assigned to the same route. The clustering process is in the first stage; i.e., it is done before the demands are realized. Tests were conducted using data where it was assumed that each customer will order a known minimum

demand d_i plus a stochastic amount which follows a binomial probability distribution. The decision regarding the order in which the customers must be visited in each cluster are taken after the demands become known. However, during the construction of the districts, a restriction for the expected tour length within each district is imposed, otherwise the solution will be a single district. The problem is modeled as SPR. The recourse action is to plan the routing in each district with as many subtours as needed so as to ensure that the total demand on every subtour is less than or equal to the vehicle capacity and the routing cost is minimized.

The basic CVRPSD was extended to include time windows in (Lei et al., 2011). The problem is modeled as an SPR, and two types of failure are considered, the vehicle capacity and the time windows. When there is a violation of the time window for a particular customer, it must be served by an additional single trip, which generates an extra cost. In addition, the vehicle must travel to the depot for restoring capacity whenever it is exceeded. No specific assumption regarding the demand distribution is made and the analysis of the expected cost of the solutions is done for both continuous and discrete cases. For computational testing, demands are generated following Poisson distributions.

Another extension of the CVRPSD can be found in Erera et al. (2010), where each tour duration must be feasible for all demand realizations. The problem, named the vehicle routing problem with stochastic demands and duration constraints (VRPSD-DC), it is modeled as an SPR. The non-splittable detour-to-depot recourse is used i.e. if the demand of customer i is greater than the remaining capacity, the vehicle must travel first to the depot to restock capacity before serving customer i. The demands are assumed to follow a discrete uniform probability distribution. Two alternatives are consider to handle the exact stockouts, either to travel back to the depot and restock capacity or to identify the stockout just after arriving to the next customer. The objective function of the VRPSD-DC is to minimize the sum of the *a priori* total travel time, the expected additional travel time due to recourse actions and a penalty term for using more than m vehicles (in case they are required).

A new recourse strategy for the CVRPSD was proposed in Ak and Erera (2007) called the “pair locally coordinated” (PLC) operating scheme and it is presented as extension of the DTD. The demands are assumed to follow discrete, independent and identical probability distributions. The idea behind PLC is that some (not necessarily all) routes are matched together to create a route pair and routes are in at most one route pair. If a vehicle exceeds its capacity, its partner adds any unserved customer to the end of its route, which operates using the DTD scheme. One of the routes in the pair is labeled “type I route” and the other is “type II”. Routes not in pairs, are also type II. A failure will occur when visiting a customer, if adding its demand, the capacity of the vehicle would be exceeded. Demand cannot be split, so if the capacity is exceeded, the demand is collected after the recourse action: If a vehicle is serving a type I route, once a failure occurs, the vehicle returns to depot and the unserved customers are added to end of the planned route of the vehicle serving its type II route pair. If the vehicle serving a type II route experiences a failure, then it returns to the depot to unload, and then resumes the the route in the first unserved customer of its route. Paired vehicles serving type II routes should wait at the final customer of their route, until the vehicle serving their paired type I route is traveling to the depot. The testing was performed on instances with demands that follows a discrete probability distribution that is the same for all customers.

The concept of PLC (Ak and Erera, 2007) was also used in Zhu et al. (2014) as part of a paired cooperative reoptimization (PCR) for the CVRPSD. This strategy is proposed to be used for a pair of vehicles. The demand is assumed to follow a uniform discrete probability distribution. The problem is modeled as a SPR with DTD as recourse action, in addition partial reoptimization of routes is applied as described in Secomandi and Margot (2009). The two vehicles can communicate and dynamically modify their routes. The information about locations, residual capacities and unvisited

customers is available to both vehicles. The model considers three assumptions: the service time is ignored, vehicles travel at the same speed and do not have idle time. Even though the strategy is proposed for a pair of vehicles, the authors briefly mention that the multivehicle case is solved by clustering the customers into groups and serving every group by a pair of vehicles.

The CVRPSD was formulated as a set partitioning problem and the associated column generation subproblem is solved using a dynamic programming scheme in Christiansen and Lysgaard (2007). In principle the demands can follow any probability distribution with accumulative property (the sum of two or more independent variables follows the same distribution). However, the tests were performed on instances with Poisson demands. The recourse action for the vehicle is to return to depot when a customer's demand is greater than the residual capacity. If the vehicle becomes exactly depleted, it will not go to depot. It continues the route until a customer with positive demand is found, then it goes to the depot to restock.

The same problem was later studied in Goodson et al. (2012), where after generating a set of routes, a set partitioning problem is solved. In that way the best solution that can be constructed using a subset of the routes is found. The problem is also modeled as an SPR and uses DTD as recourse action.

Mendoza and Villegas (2013) worked with the same type of problem with an unlimited fleet of vehicles with capacity Q. The demand follows a known probability distribution, in the tests Poisson is used. The problem is formulated as an SPR and the recourse action is DTD.

The same problem and formulation as in Christiansen and Lysgaard (2007) was recently considered in Gauvin et al. (2014), where the CVRPSD was also formulated as an SPR. The recourse action is DTD, with the particular feature that in case of exact stockout, the vehicle returns to the customer where the failure occurred, after restoring capacity at the depot. The demand is assumed to follow a Poisson distribution.

A different approach that includes a limit on the duration of the routes in the CVRPSD was presented in Mendoza et al. (2015). The problem is modeled as a CCP and as an SPR. In the first case, the probability of the total duration of a route being greater than the maximum duration must be lower than a given threshold. In the second case, the violations to the maximum duration are included as a penalty in the objective function (expected cost of overtime). The objective is to minimize the total expected duration of routes.

The CVRPSD was studied using a different approach in Sungur et al. (2008), as the robust vehicle routing problem (RVRP). A solution that is feasible for all demands that belong to a bounded uncertainty set is said to be robust. Constraints that depend on the uncertainty set replace the connectivity and capacity constraints in the original model. The solution for the RVRP is a route that optimizes the objective function when all uncertain parameters are assumed to have the worst case value. The resulting problem is reported to be not significantly more difficult than solving the deterministic counterpart. It is assumed that the bounded uncertainty set captures all uncertainty of interest and the *a priori* route is feasible for every demand realization within the bounded uncertainty set, so no recourse actions or costs are considered.

Another robust approach can be found in Lee et al. (2012), where the robust CVRP with deadlines and travel time/demand uncertainty is studied. In this version of the problem, there is a deadline assigned to to every customer i. This can be seen as a specific case of time windows, where the earliest starting time is equal to zero. The objective is to minimize the total distance, which is deterministic since there is no uncertainty associated to the distances. The stochastic parameters are the travel time (which may include the service time) and the demand. There are no recourse action in case of failures, since the robustness of a solution is evaluated as the percentage of scenarios

(from a set) in which the solution is feasible. This has some similarity with a CCP. Scenarios are generated assuming that demands follow a normal distribution and travel times follow a distribution based on truncated normal. Such scenarios are used just to evaluate and compare the the robustness of found solutions. During the search a uniform distribution is assumed.

A more recent version of the robust CVRPSD can be found in Gounaris et al. (2013). The probability distribution of the demands is unknown. However, it is assumed that all possible realizations of the demands are known (support). Several deterministic formulations of the CVRP are reformulated into robust CVRP: Two-index vehicle flow (Laporte et al., 1985), Miller-Tucker-Zemlin (MTZ) (Kulkarni and Bhave, 1985), precedence formulation obtained from MTZ (Gounaris et al., 2013), commodity flow (Gouveia, 1995) and vehicle assignment formulation (Golden et al., 1977). Two demand supports are considered, budget and factor support. In the budget support, the customers are partitioned in four geographic areas. Customers demands can deviate from their nominal values at most $\alpha\%$, but the cumulative demand in each area can not exceed the nominal value by more than $\beta\%$. In the factor model support, the demand of a customer depends on the nominal value and an additive disturbance that depend on several independent factors, for the tests, it is a measure of the relative proximity of the customer to the geographical areas. As in the budget support, the cumulative demand in each area can not exceed the nominal value by more than $\beta\%$.

2.2 The capacitated arc routing problem with stochastic demand - CARPSD

The CARPSD was introduced by Fleury et al. (2002), the problem was presented as the stochastic capacitated arc routing problem (SCARP). The problem is defined on an undirected graph, where a set of edges (not necessarily all the edges in the graph) have a nonnegative stochastic demand of items that must be collected and a set of vehicles with identical limited capacity is based at the depot. The problem is modeled as SPR, if total demand of a route is greater than the vehicle capacity, a trip to the depot has to be performed. The objective of SCARP is to find a solution for which the variations due to the random event realizations in the number of trips (and the cost) are minimum. The problem is not solved directly, instead a deterministic model is used to find solutions that are subjected to a sensitivity study by computing estimators of the average total cost and the standard deviation of the total cost. These computations are done by generating different scenarios. Demands of edges that require service are assumed to follow a truncated normal distribution, in a way that demands are greater than zero and less than or equal to the vehicle capacity. A similar problem and models are presented in Fleury et al. (2005b). The SCARP was also studied in Fleury et al. (2004), where a few additional assumptions are considered: the average demand of an edge is small, compared to the vehicle capacity. A trip can not be interrupted more than once. In a robust solution, route failures are not common, if they occur it is more likely to happen just before the last edge to be served. Demands are also assumed to follow a truncated normal distribution.

The same problem was studied in Fleury et al. (2005a). Some theoretical analysis is done on the problem and five variants are considered: the minimization of the average cost, C, of solution to the stochastic problem, minimization of C with bounds on the number of vehicles, minimizing C plus a fixed constant multiplied by the standard deviation of C, minimizing C plus a fixed constant multiplied by the standard deviation of C, minimizing C under the condition that its standard deviation be less than a fixed value,and minimizing C under the conditions that the probability of having a route failure is less than a fixed value, for every route.

In the capacitated arc routing problem with stochastic demands (Laporte et al., 2010) there is a subset of edges with a non-negative demand of items that must be collected, a depot with a set of trucks and a vertex (dump site) that may or may not be different from the depot. The edge demands follow a known probability distribution, it can be either discrete or continuous and it is

distributed uniformly along the edge. The edges can be traversed any number of times, but must be served just once. The recourse action is to travel to the dump site when the capacity of any vehicle is reached and resume its route at the point of failure. Just one failure is allowed per route. Test instances are generated using demands that follow Poisson distributions.

A similar problem was studied in Christiansen et al. (2009), but in this case the demands on the edges are assumed to have a Poisson probability distribution. This implies that if the edge is divided in a number of segments, the demand on each segment will also have a Poisson distribution. The computation of an approximate expected number of failures, consider a range of segments from one to a sufficiently large integer number. The routes start and end at the depot. A failure occurs when the actual accumulated demand exceeds the capacity of the vehicle. It is assumed that the total demand is revealed gradually along the edge and in case of failure the vehicle returns to depot using the end point of the edge that gives it a shorter distance to the depot. Several failures are allowed per route.

2.3 The CVRP with stochastic customers (and demand) - CVRPSCD

The customers' presence has also been modeled as stochastic in several variants. The most interesting formulations are as SPR, where the routing is done for a given set of possible customers (stage 1), and then the presence is revealed, meaning that some customers in the original set have demand 0, and do not need a visit. The recourse action is to modify routes (stage 2). The demand at the present customers can be deterministic, but even more interesting are the formulations where demands are also stochastic, giving the CVRPSCD problem class. Pioneering work on such problems is discussed in Gendreau et al. (1996a), where the CVRPSCD problem is described as "exceedingly difficult".

A further extension of the problem based on a case study, was solved as a dynamic and stochastic problem in Hvattum et al. (2006). It is the case of a distribution company, where the customers can call at any time of the day, in addition there is a stochastic demand associated with the customers. Some of the calls are received before the vehicles are dispatched. The problem was modeled as an SPR, where the recourse action is using new vehicles and/or rearranging the customers already planned in a route. This problem does not have *relatively complete recourse* since feasibility in the first stage, does not imply feasibility on the second stage due to the customers' time windows. The number of customers that appear at each time interval follows a Poisson probability distribution. Every demand already registered in the historical database is assigned an equal probability of reappearing as the demand associated with a new customer.

Another problem dealing with package delivery was presented in Zhong et al. (2007) where strategic and operational (daily) routes are created. Customer requests and locations are not known with certainty when designing the strategic routes. However this information is revealed before vehicles are routed for every operational route. The learning curve (and forgetting curve) of the drivers regarding the different areas is taken into consideration and the time used to serve a set of customers varies from driver to driver. Because of that, the operational routes are designed so that day-to-day variations are minimized. Even though there are no constraints regarding the capacity, there is a time constraint: the vehicle must return to depot within the driver's work shift. Customers are grouped into cells, which are each served by a single driver. Some cells are grouped into core areas that are assigned to the same driver every day. The rest of the cells are not assigned during strategic routing and can be served by any driver in the operational routes. The number of customers in each cell is assumed to follow a normal distribution. The problem of designing the strategic routes is modeled as CCP and the result of it is a nonlinear generalized assignment problem.

A vehicle routing problem with stochastic customers was studied in Sörensen and Sevaux (2009) as an example of a stochastic version of the CVRP. In this case customers need to book the service ahead, and cancellations at a short notice are allowed. This problem is used to test the *flexibility* of solutions, measured as the possibility of being adapted/repaired after cancellations are realized and still having a high performance. An option to deal with this problem is to include all customers in the route, and remove the customers that do not require service, once this information becomes known. Customers need service with 0.5 probability. Customers not requiring service are not visited.

A different version of the CVRPSD with time windows was proposed in Erera et al. (2009). Each day the customers to be visited is a subset of all the customers because the demand quantity, which is uncertain, can be zero. Customers may have one or two time windows. Each day, customers are assigned to two routes: primary and backup. The recourse action is to move customers to a backup route. Customers are visited in the order that they appear in the route and those that do not require a service that day are skipped. The research was motivated by a collaboration with a beer, wine, and spirits distributor. Here are the key ideas of the approach: customers are divided into two sets, regular (with a high probability of placing an order for that day) and irregular, the former are included into the planned routes and the latter are added dynamically to operational routes. Regular customers are assigned to two routes, for each weekday: primary and backup. Customers can be moved to a backup route to regain feasibility or improve costs, this is done every day after demand realizations and the result is the operational routes. Each customer i places an order a given day with probability p_i. The discrete random variable, that represents the quantity that must be delivered to customer i, if an order is placed is q_i. The probability mass function of q_i is known. For each customer the service must start within its time window(s).

2.4 Probabilistic multi-vehicle pickup and delivery problem - MPDP

This problem is described as a fleet of vehicles that must serve a set of customers' requests (Beraldi et al., 2010), where each request specifies an origin and a destination and the origin must be visited before the destination. At the depot there may be cargo transfer between vehicles, so a request can be served by two routes, one for pickup and one for delivery. A fixed number of routes is designed. For each customer a companion is specified that is the customer where the delivery must be sent. There are no restrictions on the capacity and the vehicles perform two routes per day.

In a solution to the deterministic problem, each vertex $V \setminus 0$ is part of a single route, the pickup visit is done before delivery optimizing a given performance indicator. Randomness comes from the fact that a customer may or may not require a service. A Bernoulli random variable is associated to every customer i, it takes value one with probability p_i, if i requires a service, and zero with probability $1 - p_i$ otherwise.

The problem is modeled as a two-stage SPR. In the first stage, m routes are designed, with m equal to twice the number of vehicles, since they perform two routes per day, satisfying that each vertex is visited once and precedence constraints (delivery performed after pickup). In the second stage, after information about the requests is available, the customers are served in the same order as in the *a priori* route. Customers with no service requirement are skipped, which is considered to be the recourse action.

2.5 The CVRP with stochastic travel times (and service times)

Travel time has also been considered as the element that brings stochasticity to the CVRP. A version of the CVRP with soft time windows and stochastic travel times is found in Ando and Taniguchi (2006). In this model a vehicle is allowed to make several routes per day and all goods

from each customer must be loaded at the vehicle at the same time. Total weight of the goods in one route must not exceed the capacity of the vehicle. In addition there is a hard time window for the depot. A triangular distribution is estimated for the travel time using real data. The objective of the problem is to minimize the total cost, which is given by the fixed cost of using vehicles, the operational costs and penalties for arriving outside time windows. The penalty for late/early arrivals can be seen as a recourse. The service time is assumed to be deterministic. Tests are performed for the single vehicle case.

A multi-objective approach to the CVRP with soft time windows and stochastic travel times (SCVRPSTW) is found in Russell and Urban (2008). In SCVRPSTW the demand is known in advance, and there is a deterministic service time and a time windows associated to each customer. Servicing outside the time window is allowed at a cost, either for earliness or lateness. Three objectives are taken into consideration, the minimization of the number of vehicles, the total traveled distance and the total expected penalties for earliness and lateness in the service. Travel times are assumed to follow a shifted gamma, but the analysis and the tests are performed using a special case of gamma, the Erlang distribution. Due to the additive property of gamma distribution, minimizing the total traveled distance is equivalent to minimize the expected travel time. The problem is modeled as a SPR, the recourse being the cost for servicing outside the time windows. The authors indicate that the problem could be modeled as a CCP, however no tests were conducted for that case.

Another version of CVRP with stochastic travel times includes simultaneous pick-ups and deliveries (Zhang et al., 2012), where each customer can have both pick-ups and delivery demands. The vehicle has a maximum travel time B. The problem is modeled as CCP, so in a feasible solution the probability that the vehicle travel time be less than or equal to B must be greater than or equal to a given parameter. Testing is performed on instances with travel time following a normal distribution.

A variant of the problem including soft time windows was modeled as SPR (Taş et al., 2013) with is an extra cost for servicing the customers outside the time windows. In addition there is an overtime cost when the route time is longer than certain value. The recourse cost is given by these penalties. In this formulation the demand is deterministic and the travel time is assumed to follow a Gamma probability distribution. The objective is to minimize the sum of transportation costs and service costs. Transportation costs are the total distance, the fixed cost of using the vehicles and the total expected cost of overtime. Service costs are incurred for early or late arrivals at customers locations. The same problem can be found in a more recent paper by Taş et al. (2014).

A CVRP with time windows and stochastic travel and service times is studied in Li et al. (2010). Here the problem is modeled using both a CCP and an SPR model. In the CCP approach two aspects are considered, the probability of arriving to each customer within the time windows and the probability of finishing a route within certain given time. In the second approach, the expected value of some extra costs is computed: the penalty for arriving after the deadline of the time windows and the cost of the driver overtime. The travel times and service times are assumed to follow a normal probability distribution.

In Kenyon and Morton (2003) an uncapacitated VRP with stochastic travel and service time is described and two different problems are studied. The minimization of the completion time, which is the duration of the longest route, is one of the problems. The second problem is the maximization of the probability of completing the operation within a predefined target time. In the second stage, no route reoptimizations or recourse actions are allowed after the times are realized.

The two models are analyzed theoretically, including the computation of bounds and how is the connection to the deterministic model that uses the expected value of the parameters. In the tests reported, the travel times is assumed to follow a discrete distribution as well as a uniform distribution. Tests with stochastic service times were not reported.

An interesting queueing approach is used by Woensel et al. (2007) to model routing problems with time-dependent travel time. The travel time is assumed to depend on traffic congestion. There is a limit for the maximum length of every route L and a deterministic service time. The expected travel time is calculated analytically together with the variance. The variance allow the evaluation of the risk involved. Time-dependent speeds are obtained using queueing models. Assuming that traffic conditions are stationary, there is a relationship between flow (number of vehicles), density (number of cars on a road segment) and speed. The time horizon is divided into a certain number of discrete periods. A different travel speed is associated to each period. Each segment of the route is considered as a service station where the vehicles arrive at one rate λ and get served at a rate μ. The objective function is to minimize total travel time, subject to capacity and length constraints. A modification of the objective function is also considered by adding the variance of the travel times, multiplied by a factor.

The CVRP with deadlines under travel time uncertainty was modeled by Adulyasak and Jaillet (2014) using two different approaches: as a robust problem and as a stochastic problem. In the stochastic approach the probability distribution of travel time is assumed to be known (Normal in the tests). The objective is to minimize the sum of probability of deadline violations. In the robust approach, on the other hand, the exact probability distribution is unknown but it is described by an interval and a mean. The objective is to optimize a performance measure, the lateness index, which takes the value of zero if the travel time meets the deadline. Both approaches are extended using fixed service times, random service time and by replacing deadline for a soft time windows.

The distance-constrained CVRP with stochastic travel and service times (DCVRPSTT), which was originally introduced by Laporte et al. (1992), is modeled using a different approach in a recent paper by Gómez et al. (2015). This new approach is interesting since no assumptions are made on the probability distribution. The stochastic travel and service times are modeled with Phase-type (PH) distributions (Neuts, 1981), where a random time interval is modeled as being composed of a number of exponentially distributed segments. There exists a PH distribution arbitrarily close to any positive distribution. The objective function is to minimize the total expected duration, subject to a service level condition, where every route must finish before a threshold T with a probability greater than β.

The CVRP with stochastic travel times was studied using a different approach in Solano-Charris et al. (2015), as a robust CVRP. The travel times are modeled by discrete scenarios, which are not associated with probability distributions. The objective of this problem is to minimize the worst cost (total cost of the routes) over all scenarios. A lexicographic approach is used to break ties, ranking the other scenarios from worst to best.

2.6 The VRP with stochastic service times

In some types of services, the variability of the travel time is considered small, compared to the service time. Because of that, the travel time might be considered as a deterministic parameter and the stochasticity of the problem is given by the service time. In Lei et al. (2012) the number of vehicles and their capacity is considered unlimited and a stochastic service time is associated with each customer. There is an overtime cost if the route time exceeds a given value. The sum of travel cost, expected service and overtime cost is minimized. The service time is assumed to follow a continuous probability distribution. During the testing, service times in the instances are assumed to be normally distributed.

2.7 The CARP with stochastic service times

The routing problem in daily road maintenance operations is formulated as a variation of the arc routing problem, where the travel and service times are considered stochastic Chen et al. (2014). Each day some segments of the road network need to be monitored, which is an operation performed using a fleet of vehicles. Each monitoring service is associated with an estimated service time and each segment of the road is associated with a stochastic travel time. The objective is to determine a set of monitoring routes of minimum cost and the total service duration of each vehicle must not exceed a given threshold L. The problem is modeled using both CCP and SPR. In the CCP formulation, the objective is to minimize the total service cost, while the probability of the total service duration being greater than L must be kept below a given value for each route. The total service cost includes the the total fixed cost of using the vehicles and the total deadheading cost (traveling over a segment of road without servicing it). Travel and service times are considered to follow a normal distribution.

In the SPR approach, the objective is to minimize the total service cost and the expected recourse cost. Weights can be assigned to prioritize any of the costs. Two recourse alternatives are considered: when a failure is expected to occur in a particular arc a_{ij} different than the last one, then once the service before the arc a_{ij} is finished, the vehicle travels back to the depot and the rest of the services are reschedule for the next day, at a cost. The other alternative is that the vehicle serves all the arcs that belong to the original route and are located in the shortest path between arc a_{ij} and the depot. In this case the recourse cost includes the penalty for rescheduling some services for the next day, but also the excess duration of the work.

2.8 The VRPSTW with stochastic service times and probabilistic customers

The courier delivery problem (CDP) is presented in Sungur et al. (2010) as a variant of the VRPSTW. In the CDP the customers appear probabilistically and service times are stochastic. The objective is to create regular routes which are later adapted to the demand realizations every day. Delivery requests arrive daily from potential customers. Location and time windows are known, but not the service time. There is a limited number of couriers and a hard time window at the depot. The first goal when solving the problem is to construct the master plan. The second goal is to modify the master plan to construct daily schedules, in a way that number of customers served is maximized, route similarity (with respect to the master plan) is maintained, penalties for earliness and lateness are minimized, as well as the total time (travel, waiting and service time). The objective function is modeled as weighted sum of all these goals. Similarity is measured as the number of customers in a daily route that are within certain distance from any customer in the same master plan route. The recourse action is partial rescheduling. In computational tests, the service times are assumed to follow a lognormal probability distribution, and real-world data is used to introduce uncertainty.

2.9 The CVRP with stochastic demands and travel costs

A robust CVRP is defined by Sörensen and Sevaux (2009) as the problem of finding a solution with a high performance, across most possible outcomes. In such a framework, the CVRP with stochastic demands and travel cost is studied. Demands and travel costs are uniformly distributed. There is a maximum travel cost per route. Some penalties are associated with unsatisfied demand and travel times greater than the maximum. In this problem the regular objective function is replaced by an average cost computed on a set of stochastic parameter realizations. Another way to measure the robustness is the highest cost evaluated on the same set.

2.10 Stochastic multi-objective approaches

The SVRP literature includes some work on problems where the decision maker faces several optimization criteria, with formulations resembling a multi-objective structure.

A CVRP with stochastic demands was formulated assuming a percentage of the vehicle's capacity is reserved as a safety stock in the model of Juan et al. (2011). The routing (stage 1) is done assuming a capacity limit lower than the maximum for the vehicles. Slack capacity can be used to cope with excess in cumulative demands. Two criteria are optimized, one of them is the total expected cost and the other is the reliability, measured as the probability of the solution suffering a route failure. The demands are assumed to follow a known parametric or empirical probability distribution (discrete or continuous), however the testing is performed on instances with demands that have a log-normal probability distribution. The decision maker is provided with a set of solutions that provide a tradeoff between these two criteria. The problem is modeled as SPR: the recourse action is DTD, whenever there is a failure.

An extension of the CVRP was made to include location, allocation and routing decisions under the risk of disruption (Ahmadi-Javid and Seddighi, 2013). In this approach a set of potential producer-distributors is considered. The capacities vary randomly due to disruptions. Because of this, the actual capacity of the producer-distributor is assumed to follow a discrete probability distribution (discrete uniform in the tests). The decision regarding which potential producer-distributors should be opened has to be made. There is a set of customers with known, non-negative demands, each of which is allocated to one producer-distributor. The customers are served by a set of vehicles which might suffer disruptions, so the number of times per year that a vehicle can visit the customers allocated to it follows a discrete probability distribution (Binomial in the tests). The problem is modeled as an SPR. In case of disruptions in the producer-distributor location, a risk mitigation strategy has to be used to satisfy the customers' demand. In the case of vehicle disruptions, another vehicle is dispatched. In both cases the recourse action represents an extra cost. The decision maker is presented with three different solutions, related to three different types of risk policies (moderate, cautious and pessimistic). For each risk policy, there is a different risk measurement, expected cost for moderate, conditional value-at-risk for cautious and worst case for pessimistic.

A multi-objective CVRPSD was initially formulated with three objectives: to minimize travel time, driver remuneration and number of vehicles in Tan et al. (2007). The demands are assumed to follow a normal probability distribution. It was found that two of the objectives, travel time and number of vehicles, are not in conflict, i.e. it is possible to minimize them together. This makes sense since the solution to a CVRPSD with minimum travel distance will have a single vehicle (Yang et al., 2000), and in the motivating case, the travel time is computed as the Euclidean distance. The problem is modeled as an SPR, with recourse action DTD. In addition, there is an extra cost if the route length exceeds a time limit B.

3 Tables

In Table 1 there is a summary of surveyed papers dealing with the CVRPSD, where the demand is assumed to follow a continuous probability distribution. A list of all abbreviations and acronyms used in this section can be found in Table 7.

Table 1: Summary of papers dealing with the CVRPSD with continuous demand distribution

Author	Probability distribution	Recourse action	Solution method / evaluation
Laporte et al. (2002)	Normal	DTD	LSM / AN
Chepuri and Homem-de Mello (2005)	Gamma	Penalty paid to unserved customers	CE / SI
Rei et al. (2007)	Truncated normal	DTD	LBD / AN
Sungur et al. (2008)	Unknown (bounded)	None	B&C / SI
Rei et al. (2010)	Any. Normal reported	DTD	MDLB, LSM + B&C / SI
Mendoza et al. (2010)	Any. Normal reported	DTD	MA / AN
Mendoza et al. (2011)	Any. Normal reported	DTD	SCW, LAH and DP / AN
Lee et al. (2012)	Normal	None	B&P / AN
Jabali et al. (2014)	Truncated normal	DTD	LSM / AN
Goodson (2015)	Any. Normal reported	DTD	SA + CO / AN

In Table 2 there is a summary of surveyed papers dealing with the CVRPSD where the demand is assumed to follow a discrete probability distribution.

Table 2: Summary of papers dealing with the CVRPSD with discrete demand distribution

Author	Probability distribution	Recourse action	Solution method / evaluation
Hjorring and Holt (1999)	Several. Discrete reported	DTD (no while exact stockout)	LSM / AN
Yang et al. (2000)	Discrete. Discrete triangular reported	DTD. Preventive restocking	DP / AN
Secomandi (2000)	Discrete uniform	DTD. ROPT	NDM / AN
Secomandi (2001)	Discrete. Discrete uniform reported	DTD. ROPT	RA / AN
Laporte et al. (2002)	Poisson	DTD	LSM / AN
Secomandi (2003)	Discrete. Discrete uniform reported	DTD. ROPT	RA / AN
Bianchi et al. (2005)	Discrete uniform	DTD. Preventive restocking	SA, TS, ILS, ACO and EA / AN
Haugland et al. (2007)	Binomial. A minimum amount will be demanded	DTD (deterministic)	TS, MSH / AN
Ak and Erera (2007)	Discrete	PLC	TS / AN
Christiansen and Lysgaard (2007)	Any with accumulative property. Poisson reported	DTD. (no while exact stockout)	B&P / AN
Secomandi and Margot (2009)	Discrete. Discrete uniform reported	DTD. ROPT	DP / AN
Erera et al. (2010)	Discrete Uniform	NSDTD	TS / AN
Lei et al. (2011)	Any. Poisson reported	Violation of TW, additional single trip. DTD	ALNS / AN
Goodson et al. (2012)	Poisson	DTD	SA + CO / AN
Mendoza and Villegas (2013)	Any known. Poisson reported	DTD	MSSH / AN
Marinakis et al. (2013)	Any. Discrete uniform reported	None	PSO / AN
Gounaris et al. (2013)	Unknown. Realizations are known	None	B&C / SI
Gauvin et al. (2014)	Poisson	DTD	BC&P / AN
Zhu et al. (2014)	Discrete uniform	DTD. ROPT	DP, RA / AN
Mendoza et al. (2015)	Poisson	DTD / CCP	GRASP + HC / AN

In Table 3 there is a summary of surveyed papers dealing with the CARPSD.

Table 3: Summary of papers dealing with the CARPSD

Author	**Probability distribution**	**Recourse action**	**Solution method / evaluation**
Fleury et al. (2002)	Truncated normal	DTD	GA / SI
Fleury et al. (2004)	Truncated normal	DTD	GA / AN
Fleury et al. (2005b)	Truncated normal	DTD	GA / SI
Fleury et al. (2005a)	Truncated normal	DTD	GA / SI
Christiansen et al. (2009)	Poisson	DTD	B&P / AN
Laporte et al. (2010)	Any. Poisson reported	DTD	ALNS / AN

In Table 4 there is a summary of surveyed papers dealing with the CVRP with stochastic customers and demands.

Table 4: Papers dealing with the CVRP with stochastic customers and demands

Author	**Probability distribution**	**Recourse action**	**Solution method / evaluation**
Hvattum et al. (2006)	Number of customers, Poisson. Demands, discrete uniform	New vehicles / Rearranging customers	PHH / SI
Zhong et al. (2007)	Normal	CCP	TS / AN
Sörensen and Sevaux (2009)	Discrete	Customer not requiring service are skipped	MA / SI
Erera et al. (2009)	Discrete	Move customers to a backup route	LS / SI
Beraldi et al. (2010)[a]	Bernoulli	Customer not requiring service are skipped	LS / AN

[a] Paper deals with the probabilistic multi-vehicle pickup and delivery problem

In Table 5 there is a summary of surveyed papers dealing with the CVRP with stochastic travel time (and service time).

Table 5: Summary of papers dealing with the CVRP with stochastic travel time (and service time)

Author	**Probability distribution**	**Recourse action**	**Solution method / evaluation**
Ando and Taniguchi (2006)	Triangular	TWVC	GA / SI
Woensel et al. (2007)	-	OC	ACO / AN
Zhang et al. (2012)	Normal	CCP	SS / AN
Lee et al. (2012)	Truncated normal	None	B&P / AN
Taş et al. (2013)	Gamma	TWVC. OC	TS / AN
Russell and Urban (2008)	Shifted gamma	TWVC	TS / AN
Taş et al. (2014)	Gamma	TWVC. OC	B&P / AN
Kenyon and Morton (2003) [a]	Discrete, uniform (travel time)	None	B&C / AN
Li et al. (2010) [a]	Normal	TWVC. OC. CCP also considered	TS / SI
Adulyasak and Jaillet (2014)	Known. Normal reported	DVC	B&C / AN

Continued on next page

Table 5 – *Continued from previous page*

Author	Probability distribution	Recourse action	Solution method / evaluation
Chen et al. (2014)[b]	Normal	Reschedule / CCP	ALNS / SI
Gómez et al. (2015) [a]	Modeled using Phase-type distribution	CCP	MSSH / SI
Solano-Charris et al. (2015)	Discrete scenarios	Robust	CW, GRASP, ILS, MS-ILS / AN

[a] Paper deals with the VRP with stochastic travel and service time

[b] Paper deals with the CARP with stochastic travel and service time

In Table 6 there is a summary of surveyed papers dealing with the VRP with a multi-objective approach.

Table 6: Summary of papers dealing with a multi-objective approach

Author	Probability distribution	Recourse action	Solution method / evaluation
Tan et al. (2007)	Normal	DTD. Extra cost for routes longer than a limit value	MOEA / SI
Juan et al. (2011)	Any. Lognormal reported	DTD	MSSP + CW / SI
Ahmadi-Javid and Seddighi (2013)	Discrete. Discrete uniform reported (capacity). Binomial reported (number of visits)	Cost for covering the lack of capacity or hiring extra vehicles	LS / AN

Table 7: Notation

Abbreviation / Acronym	Method
ACO	Ant colony optimization
ALNS	Adaptive large neighborhood search
AN	Analytically
B&C	Branch-and-cut
B&P	Branch-and-price
BC&P	Branch-cut-and-price
CCP	Chance constraint programming
CE	Cross entropy
CO	cyclic-order
CW	Clarke-Wright
DP	Dynamic programming
DTD	Detour to depot
DVC	Deadline violation cost
EA	Evolutionary algorithm
HC	Heuristic concentration
ILS	Iterated local search
LAH	Look ahead heuristic
LBD	Local branching descent
LS	Local search
LSM	L-shaped method
MA	Memetic algorithm
MDLB	Multidescent local branching
MOEA	Multi-objective evolutionary algorithm
MSH	Multi-start heuristic
MS-ILS	Multi-start iterated local search
MSSH	Multispace sampling heuristics
MSSP	Multi-start search procedure
NDM	Neuro-dynamic methodology

Continued on next page

Table 7 – *Continued from previous page*

Abbreviation / Acronym	**Method**
NSDTD	Non-splittable detour to depot
OC	Overtime cost
PHH	Progressive hedging heuristic
PLC	Pair locally coordinated
PSO	Particle swarm optimization
RA	Rollout algorithm
ROPT	Reoptimization
SA	Simulated annealing
SCW	Stochastic Clarke-Wright
SI	Simulation
SS	Scatter search
TS	Tabu search
TW	Time windows
TWVC	Time windows violation cost

4 Conclusions

In order to survey the past 20 years of research on stochastic VRPs, we found it necessary to limit the scope of the survey to what we consider “core” stochastic VRP papers. Needless to say, there is a vast literature on optimization problems relating to VRPs, which is beyond the scope of this survey. Related problem types include Traveling Salesman Problems (TSP), Inventory Routing Problems (IRP), and Fleet Size and Mix VRP (FSMVRP). In all of these classes there are several papers on stochastic variants of the problems. We refer the reader to general surveys by Coelho et al. (2014) and Andersson et al. (2010) on IRP, Hoff et al. (2010) on FSMVRP, Pillac et al. (2013) on dynamic VRP.

An important feature in the stochastic CVRP is the source of stochasticity. This may be the demand, the travel time, the presence of customers and the service time, among others. The CVRPSD (capacitated vehicle routing problem with stochastic demand) has been by far the most studied version of the problem.

Another important distinguishing feature of stochastic vehicle routing problems is the *recourse* policy, which describes the actions to take in order to repair the solution after a failure. Three popular actions are as follows.

- The vehicle returns to depot to restock when capacity is attained or exceeded. Service resumes at the customer where route failure occurred.
- A preventive restocking can be done *before* a route failure occurs.
- Re-optimizing the portion of a route that has not been served after failure or after each customer is served and its demand becomes known.

In other SVRP formulations the recourse policy does not involve routing decisions (as in the DTD case), but a penalty for late/early arrivals or the extra time cost of the driver, can be part of the expected cost when time windows and/or stochastic service time are taken into consideration. Preventative restocking and partial re-optimization seem to be more realistic but require more sophistication. With improving solver technology and ongoing research, we see the field moving more in this direction.

Looking to the future we expect to see increasing interest in multi-stage SVRP, where there are multiple recourse actions. We also foresee more interest in multiple-objectives as well as so-called robust optimization, which guards against a worst-case or bad-case. Of course, it is difficult the predict the future with certainty, hence the need for stochastic programming.

Appendices

A Solution methods

Several approaches have been used for solving (or dealing with) different variants of the stochastic VRP. Here a review of these approaches is presented, trying to cover most of the methods designed to tackle such problems. The major dichotomy in terms of solution methods is of course between exact methods on one side versus heuristic methods on the other.

A.1 Exact methods

Although realistic stochastic VRP problems are often hard to solve, exact methods have been employed with some success. These methods view the problem as special case of an integer, or mixed-integer program and employ some form of branching so that eventually a probably optimal solution will be found. Even when employed on instances that are too large for full convergence, the methods can often find very good solutions.

A.1.1 Integer L-Shaped method

The integer L-shaped method is a branch-and-cut algorithm that can described in seven steps (Hjorring and Holt, 1999; Laporte et al., 2002; Jabali et al., 2014). It can start using a previously found feasible solution (by another heuristic) as in Hjorring and Holt (1999) or with no initial feasible solution as in Laporte et al. (2002) and Jabali et al. (2014). General steps given here are these described in Laporte et al. (2002). The methods operate at each node of the search tree on a subproblem called The "current problem" (CP). Initially it is the result of relaxing integrality, subtour elimination and capacity constraints in the original problem. In addition a lower bound θ replaces the expected recourse cost in the objective function. CP is modified by adding constraints once violations are found.

1. The iteration count is set equal to zero. A new constraint is added to the original problem $\theta \geq L$, with L a lower bound on the expected recourse cost. The objective value of the best found solution is set to ∞. The only pendent node in the search tree corresponds to the initial current problem.

2. A pendent node is selected from the tree. If no pendent node exists, stop.

3. The iteration count is increased by one. The CP is solved, finding an optimal solution to it.

4. If there are any capacity or subtour elimination constraint violations, then at least one constraint is added. Lower bounding functional (LBF) may also be generated and added and the algorithm returns to Step 3. Otherwise, if the objective function of CP is greater than or equal to the objective function of the best solution found so far, the current node is fathomed and the algorithm returns to Step 2. LBFs are constraints that strengthen the lower bound of the recourse cost, and are associated with partial routes, which is to say: routes where a subset of customers are ordered in a particular way.

5. If the solution is not integer, i.e. integrality constraints are violated, branching is done on a fractional variable. The corresponding subproblems are added to the pendent nodes in the search tree and algorithm goes to Step 2.

6. The expected cost of the recourse is computed for the optimal solution of the current problem, and added to the objective function, instead of θ. The objective function value of the CP is compared to the objective function value of the best found solution, from that comparison optimal solution to the CP can become the best found solution.

7. If the lower bound, θ, for the expected recourse cost of the optimal solution to the CP, is greater than or equal to its actual expected recourse cost, the current node is fathomed and the algorithm goes to Step 2. Otherwise an optimality cut is imposed and algorithm goes to Step 3. The optimality cut just forces the algorithm to move to another solution that differs with the current in at least two edges not incident to the depot. The reason for that is that better solutions may exist since the lower bound θ is strictly less than the actual expected recourse cost.

In Hjorring and Holt (1999) the L-shaped method was used to solve the single vehicle CVRPSD. Just one route is evaluated at the time, since the the model is dealing with just one vehicle. A general lower bound for the expected value of the recourse action is imposed as optimality cuts. They use the concept of partial routes, where the same cut will improve the lower bound for all the solutions that share specific sequences of customers. The lower bound for a partial route is equal to the contributions from sequenced customers and bounds for exact stockouts (for the case of the discrete demand distributions) and normal stockouts in the subset of unsequenced customers. In general the customers in the sequence are those near the depot, at the beginning and the end of the route. A subset of unsequenced customers is between them. A tighter lower bound was determined by considering just the customers at the end of the route.

Starting from empty sequences, a greedy strategy is used to determine the partial routes. Two customers not included yet in the partial route are evaluated for insertion into it: the customer that according to the sequence in the current solution is closer to the depot at the beginning of the route and the one closer to the depot and the end of the route. The one that increase the value of the lower bound for the partial route is selected. The selected customer will have the same position, with respect to the depot, in the partial route and in the current solution. From all partial routes contained in the current solution, one is selected for use in the optimality cut to be added. The selection takes into account the lower bound value and the number of edges, where the idea is to have a small number of edges, but greater lower bound than the actual solution. Two versions of the algorithm are compared, one using specific optimality cuts and the other using cuts obtained from partial routes. The latter version managed to solve more problems to optimality and in general faster. Instances with up to 90 customers and 105% of mean total demand computed as percentage of vehicle capacity, were solved to optimality.

The integer L-shaped method was used in Laporte et al. (2002) to solve the CVRPSD. The lower bounds are computed assuming that at most one stockout will occur per route. The problem of finding L is presented in a general way, for any probability distribution. It is then solved for demands that follow Poisson distributions and normal distributions. The LBFs are defined using the concept of *lower bounds of the expected recourse cost of partial routes*. In the routes where it exists, it is computed with a method similar to that used by Hjorring and Holt (1999), in the routes where that lower bound does not exist, it is computed as L, but using just the customers in such routes and the number of routes as the number of vehicles. The result is a constraint that establishes another lower bound for θ. A separation procedure uses a heuristic to define the subsets of customers that will be treated as consecutive (two; one on each end of the route) and unstructured subsets, for each partial route. Instances with a number of customer varying from 25 to 100 vertices and a number of vehicles between 2 and 4 were solved to optimality.

An extension of the integer L-shaped method in Hjorring and Holt (1999) and Laporte et al. (2002) was used in Jabali et al. (2014) for solving the CVRPSD exactly. In this version LBFs are used to eliminate infeasible solutions. Applied to CVRPSD, the LBFs strengthen the lower bound of the recourse cost associated with partial routes found during the process. The construction of the LBF exploits the information provided by partial routes and are developed based on the structure the partial route. An exact separation procedure identifies partial routes and generates the corresponding bounds. A basic LBF implementation is compared with an integer L-shaped algorithm with no LBF. The version with LBF was able to solve to optimality more instances, and run-times are less than half the one used by the implementation with no LBF. The exact separation procedure generates better results than the heuristic used in Laporte et al. (2002). The LBF reduces the number of cuts added to the relaxed problem. The algorithm is able to solve instances with up to 60 vertices and four vehicles, and 80 vertices and two vehicles. The recourse cost is computed analytically.

A.1.2 Integer L-shaped method with Local branching descent

Local branching for the 0-1 integer L-shaped method is introduced in Rei et al. (2007). General principles of the algorithm are presented for solving integer stochastic problems, which are applied to the single vehicle CVRPSD. The main departure from the regular integer L-shaped method is a branching process when the optimal solution to a subproblem is not integer. The result is a method that tackles stochastic optimization problems with binary first stage variables. From a CP two subproblems are obtained by adding constraints that divide the feasible space. One subproblem will have as a feasible space the solutions with no more than a given number of elements that are different from the binary elements in the optimal solution to the CP. The other subproblem corresponds to a bigger feasible space; the solutions with more than a given number of elements that are different from the binary elements in the optimal solution to the CP. This larger feasible space can later be divided by adding more constraints of the same kind. When the smaller problem is not feasible, the parameter that indicates the maximum number of elements different from the optimal solution may be increased. Lower bounds are generated for each explored subregion of the feasible space. The standard L-shaped algorithm with partial route cuts (Hjorring and Holt, 1999) is compared with two implementations of local branching, using cuts either locally or globally generated. Local branching with cuts generated locally outperforms the other implementations, solving instances with up to 90 nodes.

A.1.3 Branch-and-cut

A method that solves the deterministic equivalent of the stochastic problem was proposed to solve the VRP with stochastic travel times (Kenyon and Morton, 2003). This method is applicable when the cardinality of the sample space is not large and it is proposed to deal with the two versions of the problem: the minimization of the completion time and the maximization of the probability of being completed within a target time. However the tests were done just for the first case. A probability of occurrence is assigned to each of the realizations of the stochastic parameters. The problem is then modeled as the minimization of the expected value of the objective function. For each possible realization of the parameters, a deterministic problem is solved. The method is based on branch-and-cut as follows. The subtour elimination constraints are relaxed. The optimization problem is solved and if no subtours exist, the solution is optimal. If there are subtours not including the depot, a new solution is built joining each subtour with the main tour assigned to the same vehicle. The new solution is evaluated and if it is within a preselected percentage from the optimal solution, the algorithm stops. The percentage is computed using the objective function of the solution to the relaxed problem (lower bound) as reference. If the new solution is not within the given percentage of the optimal solution, subtours elimination constraints are added and the procedure is repeated.

The method is tested on four instances, each of which is a nine node network and had a fleet of two vehicles. Travel time follows a discrete distribution. The results are compared with the optimal solutions to the problem where mean values for the parameters are used. Solutions to the stochastic models (completion time) were better.

For the VRP with stochastic travel times, where the sample space is large, a method that integrates a branch-and-cut scheme in a Monte Carlo sampling procedure has been proposed (Kenyon and Morton, 2003). In general, this method does not find an optimal solution. However, it is possible to bound the gap between the objective function of the solution found and optimal value, with a confidence level. To do that, a lower bound and an upper bound are computed. An upper bound is computed (for the minimization of the completion time), taking a solution and computing the mean completion time, using a given number of scenarios. A lower bound is computed as the average of lower bounds estimators. Each of these lower bounds estimators are computed using a predefined number of scenarios. The total number of estimators (batch size) is also a parameter of the algorithm. Each lower bound estimator is found by using a modification of the method solves the deterministic equivalent of the stochastic problem, where subtour elimination constraints are added immediately after subtours are found. On each batch another upper bound is also used, the solutions already found in previous batches, are evaluated under the realizations of the actual batch and the smallest is selected. The method was tested on a 28 node non-completed graph, with 2 vehicles, and travel times were assumed to follow a uniform distribution. Service times were deterministic.

A similar approach as in Kenyon and Morton (2003) was used by Adulyasak and Jaillet (2014) to solve the robust and the stochastic approach of the The CVRP with deadlines under travel time uncertainty. One of the main differences is how the cuts are added, while in Kenyon and Morton (2003) constraints are added upfront, in Adulyasak and Jaillet (2014) cuts are added iteratively when a feasible vector is found. Test results were compared to an iterative algorithm developed for the robust routing problem, similar to the classical Benders algorithm and to results obtained by Kenyon and Morton (2003). Performance of the algorithms is measured on different stochastic problems. Travel times are assumed to follow different probability distributions, Triangular, Normal and Uniform. Also different number of deadlines are considered, at the last node, two nodes and all nodes. Depending on the problem, optimal solutions to instances with up to 80 nodes were found. The robust and the stochastic approach in general outperform the results by Kenyon and Morton (2003) and the iterative algorithm for the robust routing problem.

A branch-and-cut based VRP solver was used to solve the RVRP in Sungur et al. (2008). The robust formulation of the CVRP is obtained by replacing the constraint imposing capacity and connectivity in the original CVRP formulation. As a result the RVRP is more capacity constrained and may be infeasible even if the original CVRP is feasible, particularly in tight instances, where total demand is very close to the total capacity. The RVRP is solved as a deterministic problem and compared to the best solution to the original CVRP, using the cost of the total cost and the ratio of unmet demands as performance indicators. The comparison is done using randomly generated scenarios. On tests performed using standard problems, it was found that performance of the robust approach depends on the structure of the network, but still robust outperformed deterministic in several cases, particularly when the instances have clustered customers. Tests were also carried out on randomly generated clustered instances, finding that the robust approach performs better in instances with dense random zone around the depot. The robust solution was also compared to a strategy that uniformly distribute the of excess vehicle capacity among all the vehicles. Solutions found using such strategy can have a lower ratio of unmet demand, but the cost might be higher than the robust solution.

A similar approach was used to solve a different version of the RVRP (Gounaris et al., 2013). The problem is solved using CPLEX 12.1 using cuts – called robust rounded capacity inequalities (robust RCI) – , which are satisfied by all feasible solutions to the RVRP, are added to the model. Violated cuts are identified by using a variant of tabu search. In some formulations RCI are already contained. In these cases, the RCI are removed and dynamically reintroduced. It was found that the same set of routes is robust in each formulation. But some of the formulations are more efficient than others, since less computational time is needed to solve them, notably two-index vehicle flow formulation and the vehicle assignment formulation (with RCI as subtour elimination). Most instances with less than 50 customers were solved to optimality. Optimality gap of the instances that were not solve to optimality (with less than 50 customers) is below 5%. The average gap on the other instances is 6.5%.

A.1.4 Branch-and-price

A new Dantzig-Wolfe solution method for the CVRPSD was proposed in Christiansen and Lysgaard (2007), based on a partitioning formulation of the problem. The customer sequence for each route is known even when an integer solution to the problem is not. So the expected failure cost can be calculated before an integer solution is known.

Starting from a master program (P_M), where the 0-1 integrality constraints are relaxed, partitioning constraints are changed to covering constraints, allowing more than one visit to the customers. Non-elementary routes are allowed (so not all elements are different in the route). The coefficient α_{ir} (1 if customer i is visited in route r, 0 otherwise) is replaced by a_{ir} (number of times customer i is visited in route r). P_M is initialized with n single customer routes. Solving P_M, a vector of dual prices is obtained. The dual prices are used in the search for columns with negative reduced costs. If the columns are found, they are added to the LP and the problem is re-optimized. The process is iterated until no more columns with negative reduced costs exist. At this point, the current solution is optimal for P_M. If the LP solution is integer and constraints are satisfied with equality, then it is also optimal for original problem. If constraints are inequalities, then they are changed to equalities and the LP is resolved, continuing with the iterative process. If the LP solution is fractional, then branching is done in order to eventually obtain an integer solution. The branch and price algorithm is a described as a variant of branch and bound, where column generation is performed at each node in the branch and bound tree. Instances with up to 60 customers and 16 vehicles are solved to optimality.

A similar decomposition is used to reformulate the robust CVRP with deadlines and travel time/demand uncertainty (Lee et al., 2012). The problem is initially formulated as a path based set covering problem, where the decision variable is either to include or not to include a route into the solution. Integrality constraints are then relaxed, and the problem is solved with a restricted set of feasible routes, since the total number of feasible routes can be very large. In this context a feasible route is the one that meets the deadlines and capacity constraints at each customer, while most of the uncertain parameters are at their maximum deviations. The column generation subproblem is solved to find a column with negative reduced cost and when no column is found the procedure is terminated. A labeling algorithm (dynamic programming) is used to find robustly feasible routes with negative reduced cost. The existence of an optimal integer solution without cycles is not guaranteed. If the optimal solution has cycles, a branching procedure is applied. Solutions obtained by the robust model are compared to solutions obtained by a deterministic equivalent CVRP. A set of scenarios are generated and the tests estimate in which percentage of scenarios the solution feasibility is retained (robustness). In three different set of instances, the robust model improves robustness by an average of about 48%, 82% and 64% respectively, compared to the deterministic approach.

The same formulation as in Christiansen and Lysgaard (2007) was used in Gauvin et al. (2014) for solving the CVRPSD using a a branch-cut-and-price algorithm. The bounds are computed by column generation solving a restricted P_M, with only a subset of all feasible routes, initialized by single routes (one for each customer). New routes are dynamically generated by adding columns (routes) with negative reduced costs, until no more such routes exist. If the solution is integer, then it is also optimal for the original problem. If it is not, violations of valid inequalities are identified. If this is not possible, then branching is used. The generation of routes with negative reduced costs is done by solving a shortest path problem with resource constraints (SPPRC) as in Lee et al. (2012), executing a dynamic programming bidirectional labeling algorithm. The concept of ng-routes is used in combination with 2-cycle elimination. Each customer has an associated set of customers with specific cardinality. A given route is prohibited from extension to include customers that belong to some of the sets associated to customers already in the route. A new dominance rule is included and it is considered to be valid when demands follow a Poisson distribution. This new rule makes possible to eliminate more labels (partial routes) that can not lead to optimal solutions. Solving the ng-route problem must be done at various nodes in the branch-and-bound tree. A tabu search heuristic is used to generate negative reduced cost columns, using moves that respect the current imposed branching. The search is restarted after reaching a number of iterations. Two types of cuts are dynamically added to accelerate the derivation of integer solutions. Capacity cuts imposing a lower bound on the number of vehicles required to serve a subset of customers, which is identified heuristically. Also subset-row inequalities are used to prohibit the coexistence of routes that cover at least 2 customers in any triplet of customers. Violations to these inequalities are identified exactly. If still no violations in these constraints are identified, branching is used. Two options are available, branching on a sequence of two customers and branching on the number of arcs adjacent to a subset of customers. Both decisions are evaluated and one of them is selected. Results from this algorithm are compared to results in Christiansen and Lysgaard (2007). This algorithm solves 20 additional instances and manages to solve instances up to 86 times faster. Seventeen new instances are solved with up to 101 customers and 15 vehicles.

A similar approach was used in Taş et al. (2014) for solving the CVRP with soft time windows and stochastic travel times. The problem is modeled as a set partitioning problem. A restricted P_M with the integer constraints relaxed and solved using the routes of an initial feasible solution found as in Taş et al. (2013). The pricing subproblem of finding columns (routes) with negative reduced cost is modeled as the shortest path problem with resource constraints. If a new route with negative reduced cost is found, it is then added to the P_M and re-optimized. The pricing subproblem is solved using the algorithm proposed in Feillet et al. (2004) that extends a label correcting reaching algorithm (Desrochers, 1988) including node resources, so the subproblem can be solved optimally. Initially multiple visits to the customers are allowed, except for those in a given set S. If the optimal solution for the relaxed subproblem is integer, then is also optimal for the original. Otherwise, the nodes that appear in the solution more than once are added to the set S. A new dominance relation is used for selecting the routes. Non-dominated routes with non-negative reduced costs and the dominated ones are added to a set called the Intermediate Column Pool (ICP). After re-optimizing the relaxed P_M, the reduced costs of columns (route) in the ICP are recomputed and columns with negative reduced cost are added to the relaxed P_M. The pricing subproblem is solved if no columns are found in the ICP. Routes stay in the ICP for a given number of iterations and its size is kept below a threshold. Branching is used if the optimal solution to the relaxed P_M is not integer. During the tests, two stopping criteria were used, time and gap between best lower bound and best upper bound less than 5%. Test are performed using two type of branching, Depth-First (DF) and Breadth-First (BF). It was found that BF provides better results. The algorithm is able to solve instances with up to 100 customers.

In Christiansen et al. (2009) a branch-and-price algorithm was proposed to solve the CARPSD where pricing is done by dynamic programming and an estimator of the total expected cost (lower bound) is computed analytically. The problem is modeled as a set partitioning problem, where

the decision variables are the selection of routes to become part of the optimal solution. The problem is solved initially for a set of columns corresponding to the set of feasible routes, but the constraints related to the number of times that an edge must be serviced and the integrality constraints are relaxed. At each iteration the set of columns corresponding to the dual vector of constraints regarding the number of times the edge are visited, with an expected negative reduced cost, are added to the initial set of columns. The problem is iteratively re-solved (pricing) until no more columns with negative expected reduced costs are found. If the solution is integer, then it is optimal for the modified problem. If the solution is fractional, then branching and pricing is continued. An optimal integer solution to the modified problem is also optimal to the original one within some precision level, which depends on the quality of the lower bounds for the expected cost of the routes. The pricing problem is formulated as a shortest path problem and it is solved by dynamic programming. The branching rule is applied when an optimal solution to the modified problem has fractional values. The largest instance that was solved included 40 vertices and 69 service edges.

A.2 Heuristic approaches

There are many stochastic problems with no exact solution method known to work for reasonable sized problems. This creates the need for heuristics, which, even though do not guarantee to find an optimal solution to the problems, may be able to find a good solution to them. This is a popular and growing area in the literature.

A.2.1 Adaptive large neighborhood search (ALNS)

An adaptive large neighborhood search heuristic was used to solve the capacitated arc routing problem with stochastic demands (Laporte et al., 2010) starting from an initial solution constructed by an algorithm called *stochastic path scanning*, which is a modified version of the *path scanning* (PS) method. At each iteration, one heuristic is selected randomly to destroy the current solution (removing q serviced edges), and then one insertion heuristic is selected randomly to repair the damaged solution, reinserting the removed edges. The selection of q is also random. The acceptance of a new best solution is given by the *record-to-record travel* (RRT) algorithm (Dueck, 1993). The total expected cost is calculated analytically. The efficiency of the algorithm is evaluated comparing the performance of the best found solution with the optimal solution to the deterministic version of the problem. Both solutions are evaluated and compared in stochastic simulation and the expected recourse cost is considered in both cases. Solutions found by ALNS show a better performance.

An ALNS heuristic was also used to solve the CVRP with stochastic demand and time windows in Lei et al. (2011). A modified version of the *push forward insertion* (PFI) heuristic is used to generate the initial solution. At each iteration, q vertices are removed by a removal heuristic, which is randomly selected. The solution is later repaired by an insertion heuristic. As in the previous case, the selection of q is random and the acceptance of a new best solution is given by the RRT algorithm. The objective function is to minimize the total expected cost, which is calculated analytically. The efficiency of the algorithm is compared against the best-known solution to the deterministic counterpart plus the expected recourse cost associated to it.

An adaptive large neighborhood search (ALNS) was proposed to deal with the CARP with stochastic service and travel times Chen et al. (2014). A branch-and-cut algorithm for the CCP formulation of the problem was not able to find optimum for some instances with 10 vertices and 25 arcs, within one hour. The initial solution is built by constructing one route at the time. Each route is constructed by adding the arc that is not been serviced yet and is closest to the end vertex of the current route. The distance is defined by the shortest path problem. During the search, the removal and insertion heuristics are randomly selected under the control of a weight that determines the selection

probability. The weights are updated periodically during the search and depend on how successful each heuristic has been. Solutions are evaluated using the objective function for the SPR model. The algorithm terminates if there is no improvement after a certain number of iterations or the total number of iterations reaches a predefined value.

Four removal heuristics are used: deterministic worst removal, random removal, reduce-number-of-vehicle removal and probabilistic worst removal. The four insertion heuristics used in the search are deterministic greedy insertion, probabilistic insertion, probabilistic insertion with recourse and probabilistic sorting insertion. The ALNS is used both models, but in the case of SPR, the chance constraint can be violated. Experiments are conducted on instances with 5 to 25 vertices. For small instances with up to 10 vertices and 20 required arcs, test were performed using the branch-and-cut algorithm and the ALNS. The branch and cut algorithm found the optimum in 31 out of 40 instances. ALNS found it in 30. The gap between the two methods for instances where optimum was not found range between 1.45% and 3.15%. For the instance with 10 vertices and 20 required arcs, the computer runs out of memory when using branch-and-cut. The ALNS was tested using both models. It was found that CCP requires the use of more vehicles. The SPR with recourse including the penalty for rescheduling services for the next day and also the excess duration of the work, shows better results, and it is suggested for real life application.

A.2.2 Dynamic programming

In contrast to stochastic programming, dynamic programming methods are typically used to find a policy rather than a one-time solution. By “policy”, we mean a function that maps the state of the system (e.g., the capacity remaining in a vehicle and the mean demand for stops remaining on a route) to an action (e.g., return to the depot).

A dynamic programming algorithm was used to solve the stochastic CVRP with optimal restocking (Yang et al., 2000). The problem posed was to find the optimal restocking policy, i.e. the right moment for a vehicle to return to the depot and restock before a route failure occurs. At every customer there are two options, to go back to depot and restock or go on to the next customer in the route. Even though stockouts might occur, one of these two options is optimal. A solution is built using two different approaches: 1) Starting with a single route through all the customers, obtained using a combined method (insertion + Or-opt), it is later partitioned into small subroutes using dynamic programming. 2) By clustering first and then routing. Both approaches are compared with a lower bound obtained by solving the LP relaxation of a set partitioning formulation of the original problem. This was possible for instances with up to 15 vertices, for larger instances the two algorithms were compared to each other. The approach number 1 (route-first-cluster-next) shows better results when compared to the lower bounds; it found solutions within 2% of optimal. The same approach was found to perform better for bigger problems. The results obtained were also compared against the solutions to the deterministic version of the problem, found by the Clarke-Wright savings algorithm (Clarke and Wright, 1964). Average improvements up to 25% were observed.

Dynamic programming was also as part of an algorithm to solve the single vehicle CVRPSD with reoptimization (Secomandi and Margot, 2009), formulating the problem as a Markov decision process (MDP). To overcome the computational challenges involved in solving large problems, just a subset of the states in the full MDP is considered, this methodology is called *partial reoptimization*. Starting from an initial sequence of customers, the tour is divided into sets/blocks where each block is reoptimized as a MDP. Two ways of dividing the blocks lead to two different proposed heuristics. Given a parameter M and an initial sequence of customers, the blocks are formed with no more than M customers each. In the second approach, every block is formed with no more than $2M+1$ customers, in this case the same customer can be in several blocks. The performance of the heuristics is compared against the optimal reoptimization policy for instances with 10 to 15

customers. For bigger problems, it is compared with two rollout policies (Secomandi, 2001, 2003) and an estimator for the lower bound, which is found solving to optimality the deterministic problem with unsplit delivery i.e. if the demand of a particular customer is greater than the residual capacity of the vehicle, a trip to the depot for restocking is required before serving the customer. The proposed algorithms perform better than rollout policies and compared with the lower bound, the solutions obtained are on average within 10% and 13% for different type of instances.

In Secomandi (2000), after formulating the single CVRPSD as a stochastic shortest path problem, the size of the resulting dynamic programming problem is found to be computationally prohibitive. To deal with that, a neuro-dynamic methodology is employed. Instead of computing the optimal cost-to-go for every stage, an approximation is used. In principle two similar policies are used. With *approximate policy iteration* (API), the cost-to-go values and pairs of states (training set) in a particular policy are found through simulation. Then least-square fitting is used to approximate the cost-to-go of the policy for all states, which is a vector of parameters. A new policy is found in a greedy fashion, together with a training set and sample cost-to-go values. The process is repeated, either until it converges or until a predefined number of iterations is reached. Convergence is not guaranteed. The other policy is the *optimistic approximate policy iteration* (OAPI). This is a variation of the previous policy. The training set is smaller and least-square problems are solved more frequently. The vector of parameters that define the cost-to-go of the policy for all states is found as an interpolation between the one used in the previous iteration and the one that would be used in the API policy. Only the results for OAPI are reported, since it always performs better. It is compared with a rollout policy, which performs over 7% better, attributed by the author to the fact that it uses an exact evaluation, and the OAPI uses and approximation in which approximation errors may lead to poor solutions.

A bilevel Markov decision process (MDP) is used to model the coordination between vehicles in the paired cooperative reoptimization (PCR) for the two-vehicle CVRPSD (Zhu et al., 2014). In the PCR strategy multiple customers can be assigned to one vehicle. The vehicles operate independently until information is shared, this is after any of the vehicles finishes serving its assigned customers. The process starts dividing the customers into two groups, and assigns a group to each vehicle. When one vehicle has completed its assignment, the remaining customers (not served yet by the second vehicle) are divided into two groups and reassigned to each vehicle. This process is repeated until all customers are served. The sequence of the visits in each group is determined by partial reoptimization (Secomandi and Margot, 2009). At each stage of the higher level (partitioning and communication), the remaining customers are divided into two groups. At the lower level the problem is formulated as MDP as in Secomandi and Margot (2009). The bilevel MPD can be solved by dynamic-programming backward recursion, and at each stage the expected cost-to-go values can be calculated. At a higher level all possible partitions are evaluated, using an approximation approach based on an *a priori* route that traverse all unassigned customers in a particular stage. The cost-to-go value is approximated using the DTD recourse action in the a priori route. Such an *priori route* is generated by the rollout algorithm in Secomandi (2003). Two versions of PLC (Ak and Erera, 2007) were implemented to be compared with the PCR: one with two vehicles and one more that finds the number of vehicle to achieve a given service level. PCR outperforms the other two approaches, with a cost improvement ranging from 20% to 30%.

A.2.3 Rollout algorithms

The single vehicle CVRPSD with reoptimization as recourse policy was solved using a rollout algorithm in Secomandi (2001). The customer that must be visited first is selected from a sequence of customers, previously found by the cyclic heuristic (Bertsimas et al., 1995). For each customer a new sequence is obtained. The customers will keep the order given in the initial sequence, however, in every new sequence a different customer will be selected as the first. The expected total length for each arrangement is computed and the first customer in the arrangement with the lowest

cost is selected as the first customer in the route. Once the first customer is selected and visited (its demand becomes known) the next customer to visit is selected. Two options are considered, the customer j that minimizes the expected value of the total length (computed as before) and the customer k that minimizes the expected total cost when visited after going to the depot for replenishment. The option with lower expected total cost is selected and the process is repeated until all customers have been served. The algorithm is described as based on neuro-dynamic programming/reinforcement learning methodology. The total expected cost of an *a priori* solution is computed analytically. Two experiments are carried out, one with small instances where the value of an optimal reoptimization policy is known, and a second experiment with large instances. Two versions of the rollout policy are analyzed in the first experiment: In the first version, in case of a route failure, the current customer is fully served by performing a trip to the depot and restock before moving to the next customer. In the second version at a given time, there may be more than one unserved customer whose demand is known but not served yet, so in case of a route failure, the customer may not be served immediately. Both versions generate near-optimal solutions. For the second experiment with large instances, just the first version of the rollout policy is tested, using three different *a priori* solutions; a nearest neighborhood heuristic together with 2-Int improvement steps, a cyclic heuristic enhanced by dynamic programming and the static rollout heuristic initialized by the tour produced by the former one. The best results are generated when using the solutions generated by the static rollout heuristic.

A rollout approach was proposed to deal with sequencing problems with stochastic routing applications (Secomandi, 2003). As particular applications, the proposed algorithms are used with the TSP with stochastic travel times and the CVRPSD. A regular rollout algorithm is compared with iterated versions of it. Where the best found solution is later used as initial solution and the process is repeated. Tests were performed iterating the algorithm two and three times, obtaining better results in the latter version.

A.2.4 Local search

A tabu search heuristic was used to design delivery districts for the CVRPSD in Haugland et al. (2007). The districts are designed considering a long term perspective where they will stay fixed, to be used for various demand realizations. The routing every day is deterministic, since demand is known before vehicles leave the depot. Districts are constructed so the expected travel cost within each district never exceeds an upper bound. A multi-start heuristic was also implemented and compared, but the tabu search heuristics performed better in the instances tested. The expected cost of the solutions is approximated using an upper bound.

An insertion-based solution heuristic called master and daily scheduler (MADS) (Sungur et al., 2010) was proposed to create the master plan to the VRPSTW with stochastic service times and probabilistic customers. A tabu search procedure is used to try to improve the solution. The solution process has two phases. Given a number of scenarios, in the first phase, a preliminary master plan and daily routes for each scenario are created. Two routing problems are considered, one for the master plan that serves high frequency customers using worst case service time and one for each scenario of daily schedule. The master plan in built using an insertion heuristic which works in a greedy fashion, inserting the cheapest of all feasible insertions. The cost of an insertion in the master plan is given by the increment in total time and TW penalty.

The master plan is then used to construct daily schedules with a partial rescheduling recourse, for every scenario. Customers not present in the scenario are skipped and those not included in the master plan are inserted. Insertion in the daily routes incurs an additional cost, the reduction in similarity to the master plan. Daily schedules are improved with a tabu search heuristic. The second phase is iterative, every scenario reports to the master plan which customers could not be inserted. Then the master plan prioritizes the unscheduled customers by the number of scenarios

for which they could not be scheduled. Maximum priority feasible insertions are performed. Scenarios construct daily routes based on the updated master plan and the process is iterated until no improvement is achieved. A buffer capacity is used in the first phase by reducing the latest time at the depot time window. It is used in the second phase to schedule additional customers. The solution obtained by the MADS is compared to a solution obtained by a algorithm that treats each scenario independently, which is called independent daily insertion (IDI). The objective function is to maximize the number of served customers and to minimize total time and time windows penalties (weighted sum).

Two real-world instances from UPS were solved by MADS and the solution is compared with IDI and the current practice. MADS is modifies and results are also compared over consistent VRP (ConVRP) instances against the algorithm ConRTR proposed in Groër et al. (2009). Compared to IDI, MADS generates more similar routes at a cost of total time and total number of served customers, but the objective function shows a general better performance. When real-world data is used, solutions from MADS (given different levels of buffer capacity) are better than solutions from IDI and the current practice (obtained by a territory planning based routing algorithm). Compared with IDI, total time is increased, but similarity is improved. For the comparison over ConVRP instances, three criteria are used: total time, average arrival time difference and maximum arrival time difference. In average MADS performs better than ConRTR in all criteria.

The CVRP with stochastic travel times, soft time windows and service costs was solved using a tabu search heuristic in Taş et al. (2013). An initial solution is constructed by means of an insertion heuristic, then a tabu search heuristic is applied to the solution. A post-optimization procedure consisting of delaying the dispatching time of the vehicles in each route is also applied. The solutions are evaluated through an analytic approximation of the expected cost. Different versions of the algorithm are tested, where different initial feasible solutions are used: solutions obtained by the insertion heuristic, with minimum total transportation cost; solutions obtained by the insertion heuristic, with minimum total weighted cost; and solutions obtained in the literature as optimal/best-known for the deterministic problem. Better results are obtained by the algorithm when using initial solutions constructed by the insertion heuristic.

A location-routing problem with disruption risk was solved by a two-step heuristic (Ahmadi-Javid and Seddighi, 2013). During the first step, a solution is randomly built. In the second step there are two phases, location and routing, in each of those a simulated annealing heuristic is used. A 2-opt procedure is used to attempt to improve the routing. The quality of the solution is evaluated analytically. The heuristic was able to find optimal solutions found by CPLEX for small instances. It is also compared against lower bounds and to a different heuristic based on a solver for location routing problems. Optimal solutions are found, error bounds are relatively small and the location routing-based heuristic is outperformed.

Simulated annealing was also used to find solutions to the CVRPSD in Goodson et al. (2012). The main purpose is to show the potential of the cyclic-order neighborhoods by using a heuristic with a simple structure. A cycle-order solution encoding is a permutation or ordering of the set of customers. Given a cyclic-order permutation π of the customers, a set $\mathcal{R}(\pi)$ represents the feasible candidate routes consisting of contiguous elements of π. These candidate routes are generated using a sweep algorithm and their cost is computed analytically. The best solution that can be built using the routes in $\mathcal{R}(\pi)$ is found by solving a set partitioning problem. Several cyclic-order neighborhoods are used. In the k-shift neighborhood structure the k contiguous elements, starting from selected index i, are moved to the positions immediately prior to a selected index j. In the reverse neighborhood, the order of the contiguous elements from index i to index j is reversed. The exchange neighborhood consists of exchanging the position of two elements. To reduce the computational effort, an updating procedure is used to obtain the set of feasible routes associated to one cyclic-order if the routes associated with a neighbor are known. The simulated annealing is

executed in two phases. In the first phase, the classical deterministic CVRP is solved. The second phase attempts to improve the best found solution from phase I by taking into account the cost of recourse actions. Results were compared to Christiansen and Lysgaard (2007) where optimal solutions were reported for 19 out of 40 instances. The simulated annealing algorithm found 16 optimal solutions out of the 19 reported by Christiansen and Lysgaard (2007). In the 21 instances where Christiansen and Lysgaard (2007) did not find a provably optimal solution, the two phase procedure matches or improves the best found solution or the expected value of the best known deterministic solution.

The cyclic-order simulated annealing procedure was modified to solve the multi-compartment vehicle routing problem with stochastic demands (MCVRPSD) in Goodson (2015). A two-stage simulated annealing is used. In the first phase, solutions are evaluated by scenarios. While the second phase attempts to improve the solution by exactly calculating the quality of the solution. The method is compared with the results reported by Mendoza et al. (2010) and Mendoza et al. (2011). Out of 180 instances, the cyclic-order simulated annealing procedure is able to improve the best-known value in 159 instances and it matches the best known solution in 21 instances.

A tabu search algorithm was used to solve the CVRP with time windows and stochastic travel and service times in Li et al. (2010). No comparison is done with results obtained by other methods but the evaluation is done by estimating the expected values through Monte Carlo simulation.

A tabu search heuristic is proposed for the VRPSD with Pair Locally coordinated (PLC) recourse (Ak and Erera, 2007). The neighborhood structure is very similar to TABUSTOCH proposed by Gendreau et al. (1996b), but moves are evaluated exactly, not approximately. This computation is done analytically, as the expected value of all the vehicles travel cost. The initial solution is found by a sweep algorithm. A service level parameter p_α is set. p_α is used to determine the number of customers in an *a priori* type II route. The probability that the vehicle serves all the customers in its route without a recourse detour before serving any of the customers of its paired vehicle must be greater than or equal to p_α, which restricts the number of customers in the route. The number of customers in the type I route is also found using p_α. The idea is that the probability of serving all customers in the paired tours, with the combined capacity and without any detours to depot, must be greater than or equal to p_α. More vehicles are assigned to type I routes than to type II routes. The sweep algorithm assigns in a counterclockwise order, first the customers in a type I route, then those in the type II route. At the end each type I route will be operated in the counterclockwise direction and the type II routes are operated in the clockwise direction.

The neighborhood of a solution x, $N(p, r, q, x)$, contains a set of solutions generated by modifying routes as follows: i) A set A of q customers is randomly chosen. ii) $\forall a \in A$ let $B(a)$ be the set of p randomly chosen customers among the r nearest neighbors of a. iii) $\forall a \in A$, $\forall b \in B(a)$: remove a from its position and reinsert it immediately before or after b in b's route. all the solutions generated by removing a customer from its tour and re-inserting it somewhere. Here, each of q randomly selected customers is removed and reinserted, either immediately after or before one of p randomly selected customer neighbors from the set of its nearest r neighbors $(q \leq r)$. New partner tours can be created for unpaired vehicles and new tour pairs can be also generated. The PLC recourse strategy is compared with the DTD recourse strategy. Tests were conducted using instances with uniformly distributed customers and homogeneous demand distributions. In addition a test was performed with real world customer location data from grocery stores in Istanbul, Turkey, but using homogeneous demand distributions, as in the previous case. Results show better performance for the PLC recourse strategy.

Another example of the application of TABUSTOCH (Gendreau et al., 1996b) is found in Erera et al. (2010), where it is used to find solutions for the vehicle routing problem with stochastic demands and duration constraints (VRPSD-DC). The solution method relies on solving an *adversarial* op-

timization problem, which determines a customer demand realization that maximizes the actual execution duration of an a priori tour, for checking if the maximum duration of a tour is respected. The expected additional travel time due to recourse actions is computed analytically. The problem was tested against the unconstrained version CVRPSD in 54 instances. It was found that the number of vehicles required for serving the customers increased in 22 of the instances with duration constrains. An increment in the total expected travel time of more than 7% was observed in the small constrained instances. In general, however, this increment was relatively small. In some cases, as the fleet size increased, the total expected travel time decreased. This was more likely to be observed in large instances (60 or 100 customers) with large vehicle capacity.

A generalized variable neighborhood search (GVNS) was proposed to solve the CVRP with stochastic service times in Lei et al. (2012). A Clarke-Wright algorithm is used to obtain an initial solution. The local search is driven by a variable neighborhood descent scheme. Six neighborhoods are used, three inter-route and three intra-route. After sorting the neighborhoods in a non-decreasing order by their cardinality, starting from the first of them, if a new solution is better than the current one, then it replaces it. After a better solution is found, the search continues with the first neighborhood, if that is not the case, it proceeds to the next neighborhood. The search stops when no improving solution is found in the last neighborhood. The selected solution is the one with the best improvement. The search does not consider infeasible solutions. As part of the local search there is a granular search mechanism, which discards long edges and focuses on short edges that have higher probability of appearing in high-quality solutions. The process is repeated following a variable neighborhood search scheme. After the shaking, the local search is applied, and the resulting local minimum is accepted if it is within a certain threshold of the current best found solution by the previous search. Solutions are evaluated analytically by means of a closed-form expression. GVNS was compared with variable neighborhood search and variable neighborhood descent, obtaining better results at a higher computational cost. Five different metaheuristics were used in Bianchi et al. (2005) to deal with the single vehicle CVRPSD: simulated annealing (SA), Tabu search (TS), iterated local search (ILS), ant colony optimization (ACO) and evolutionary algorithm (EA). Tests are conducted to determine if using the *a priori* tour distance as an approximation of the objective function will generate better results than evaluating the solutions by means of dynamic programming recursion. It is found that EA, ILS and ACO perform better with the approximation. The TS and SA metaheuristics perform better with the evaluation by dynamic programming recursion. In general, the metaheuristics with better performance are EA, ILS and TS. An additional test is conducted where ILS and EA are hybridized with the 3-opt local search operator. While the local search is based on the *a priori* tour distance approximation, the acceptance and selection criteria are based on the dynamic programming recursion. Results are compared with the CYCLIC heuristic, and an iterated local search that uses a 3-opt exchange neighborhood, solving the problem as a TSP. Results show that hybridized versions of ILS and EA perform better. A hybrid heuristic was proposed to solve the single vehicle CVRPSD in Rei et al. (2010), using exact algorithms (local branching) and Monte Carlo sampling. This Monte Carlo local branching algorithm follows a multidescent scheme. Different solutions obtained from the TSP formulation of the problem are used as starting points, then local branching search phases are performed iteratively. The decision used for the branching criteria, and in that way limiting the feasible space in a subproblem, is fixing the maximum hamming distance relative of the solutions in the feasible space to a given reference solution. At the first branching step, the optimal solution to the TSP, where the feasible region is limited to the unexplored region in previous descents is used as a reference. In the following steps it is replaced by the optimum solution to the previous subproblem and a fixed number of local branching steps is applied.

The recourse function is approximated using scenarios. The original recourse function is replaced by the approximation in the original problem, and the approximated problem is solved to optimality or until a specified time limit is reached. Each subproblem is solved applying the L-shaped branch-and-cut algorithm from Rei et al. (2007). Three types of cuts are generated by the algorithm, one

of them is valid in all the subproblems, the others depend on the sample. A different sample is generated for each subproblem. At the end of the complete descent a number of solutions equal to the number of branching steps are produced. To identify the best one, each of them must be either evaluated using the routing cost and the actual recourse function, if possible, otherwise more sampling is required to produce an estimator of the expected total cost for each of the solutions. The procedure is repeated a certain number of times.

Comparisons were done against results from L-shaped branch-and-cut algorithm (Rei et al., 2007) and the Or-opt algorithm used to build a initial single route in a Route-First-Cluster-Next algorithm (Yang et al., 2000). Computational experiments show that the algorithm is able to obtain better results than the Or-opt algorithm and same results as L-shaped algorithm, but using less computer time.

The MPDP is solved by a local search algorithm in Beraldi et al. (2010), where an insertion algorithm is used to build the initial solution. The efficient neighborhood search takes advantage of the previous computation of the expected objective function, which is done analytically. Moves are evaluated taking a vertex from a tour and moving it a position forward or backwards, up to certain level. Another option is removing a vertex from a tour and inserting it at the first position of another tour. At every move not all the neighbors solutions have to be evaluated, just the cost should be updated (analytically), depending the move that is performed. Equity of workload among vehicles is computed as percentage deviation of the total cost. This aspect is just calculated and results presented, but not considered as a constraint or extra objective in the problem. Two ways of evaluating the neighboring solutions are tested. In one the expected cost is computing from the scratch, in the second, the efficient neighborhood search is used, which is shown to be computationally more efficient.

For the construction of the planned and operational routes in the CVRPSD with time windows, different local search heuristics are used in Erera et al. (2009). The planned primary routes are constructed by an insertion heuristic, which starts with m routes serving m seed customers, and the rest of the customers are inserted one by one. Local search is used during the process to improve the partial solution. During the construction the capacity constraints are checked, trying to keep the probability of the solution being feasible greater than or equal to a given parameter. This is done analytically. The assessment of the time window constraints is done by random sampling, the probability of satisfying the time windows constraints is estimated and a solution is considered to be time windows feasible if the probability is above certain value. During the insertion, two aspects are considered regarding the quality of a solution, the expected travel time and expected duration (may be different due to waiting times). A local search improvement procedure is applied. There is a vehicle reduction procedure that tries to eliminate the routes with the shortest average route duration. For the construction of the planned backup routes, demand scenarios are generated. Backup routes are not actual routes, since customers assigned to them are not sequenced. For each scenario customers not placing orders are skipped. If all routes are feasible, no information is obtained and the next scenario is generated. If one or more routes are infeasible, then a customer is selected randomly to be removed from the route and reinserted in the route that minimizes the change in route quality. The process stops when the solution is feasible or when a given number of moves have failed to make it feasible. If the solution is infeasible, then no information is obtained, but if feasible, a local search procedure attempts to improve the solution and then irregular customers are inserted into the solutions, starting by the ones farther from the depot. For every scenario the route serving regular customers is recorded. Vehicles with the highest count, not including the primary vehicle, are selected as backup vehicle.

The construction of the delivery (operational) routes follows the same strategy as selecting the secondary routes. But for achieving feasibility, customers can be moved from its primary route to its backup route. Service times are taken into consideration. Several aspects determine the quality

of a solution if a feasible operational set of routes can be produced: total travel time, duration of all routes, number of vehicles used, number of customers visited by primary vehicle. The results of using the methodology are compared with historical data. Two days of the week are compared: Thursdays, with the highest demand, and Mondays, with the lowest demand. For Thursdays, the proposed delivery routes are shorter and, in addition, three fourths of historical routes are infeasible. The proposed delivery schedules reduce the number of miles and the travel time. If the actual requirement of the customers being visited by just a primary or backup driver is dropped, and instead of that, any driver is allowed to visit any customer, the average total miles is reduced by 4%. For Mondays, the improvement are more evident. Fewer routes are used. There is a cost connected to serving regular customer just with a primary or backup driver since total miles will be reduced by more than 8% if more drivers are allowed to serve such customers.

The problem of package delivering with driver learning was solved in two steps by Zhong et al. (2007). In the first step the strategic decision of designing the core areas is taken. In the second step, the cell routing is done. Here a tabu search heuristic is used to solve an assignment problem whose objective is to minimize the cost of assigning the cells to core areas. This tabu search is allowed to visit the infeasible region, and in addition to simple moves, compound moves are also allowed. The probability of serving a core area within the duration of a work shift is required to be below a certain threshold. The operational routing is done by first routing the cells within the core areas and later adding the rest of the cells to the partial routes, at the lowest cost. More routes are added if needed and the learning curve model is introduced. Cells closest to the depot are not assigned to any core zone. A fixed number of core areas is used, which is taken from historical data and is equal to the minimum number of driver used over certain period. When designing the core zones, the objective is to minimize the expected total time (service and travel time) used to satisfy the demand of cells in the core areas. Cells not assigned to any particular core area are also taken into the objective as part of one additional core area with lower learning curve and higher service time. Expected total time is approximated analytically. The operational routes are built using a deterministic routing algorithm that incorporates, instead of a single customer, the cell concept and the drivers' learning curve. The performance of the tabu search heuristic is compared to a lower bound found by solving to optimality a problem where the nonlinear constraints have been replace by linear ones. On average, solutions found by the tabu search are 3% above the lower bound. The approach using core areas is tested against a policy of deciding a different routing every day. It is assumed that drivers will not reach maximum learning level on this new policy. On average the core area model uses 4% fewer drivers and its total duration time is 4% less. If in the non-core area model drivers are allowed to get the maximum learning level, the solution obtained represents a lower bound. On average the core area model uses 6% more vehicles, total duration time is 7% longer and total distance is 5% greater, compared to the lower bound, which is considered to be good performance by the algorithm.

A local search heuristic was used to solve the CVRP with soft time windows and stochastic travel times (SCVRPSTW) in Russell and Urban (2008). The solution procedure is a tabu search with three phases. First, an initial solution is obtained, which is improved in the second phase by a tabu search. In the last phase a postprocessing procedure is used to optimize the waiting times before each customer. The initial solution is built using a deterministic tabu search with a mixed neighborhood search procedure that uses node exchanges and edge exchanges. Waiting time is allowed at the depot, but not at customer locations. Once the initial solution is obtained, a tabu search heuristic that evaluate moves analytically is applied to it. This heuristic uses basically two types of moves. One move is to remove a node from its position in route r_i and inserting in route r_j (r_i may or may not be different from r_j). the second move is to swap the position of two nodes. Tabu moves can be performed if they fulfill the aspiration criteria. As a diversification strategy, the the tabu tenure is increased if solutions are found to be repeating too often. In case it happens extremely often, the search is diversified by making random moves. The post-optimization tries to find optimal waiting values before each customer.

The objective function used to guide the search is a weighted sum of the objective functions. Tests are performed using two different objective functions, $1000V + 0.5D + 0.5P$ and $1000V + D + 0.2P$, with V equal to the total number of vehicles, D the total traveled distance and P the penalties associated to servicing customers outside their time windows. Expected penalties are computed in closed form. Results are compared with the solution obtained in the first phase. When the first objective was used, the stochastic travel time approach was able to reduce the number of vehicles in 10 out of 16 instances. The total distance was reduced in 11 instances and the time windows penalty was reduced in 12 instances. If more priority was given to the total travel distance than to the expected penalty, as in the second objective, solutions with fewer vehicles and less distance traveled are obtained, however the time windows penalty increased. For this case the stochastic travel time approach was able to reduce the number of vehicles in 12 out of 16 instances, the total distance was reduced in 14 instances and the time windows penalty increased in 12 cases.

Several local search heuristics are proposed to find solutions to the robust CVRP with uncertain travel times (RVRP) (Solano-Charris et al., 2015). A Clarke and Wright heuristic for the RVRP, starting by a trivial solution where each customer is visited by a different vehicle and routes are merged in a variety of ways. A randomized version is also used where before merging two routes, a perturbation is randomly computed. A local search procedure that uses intra-route and inter-route moves: relocate (different versions), interchanges, 2-opt (two versions). Four other metaheuristics are also used: GRASP, iterated local search (ILS), Multi-Start ILS and Multi-Start ILS alternating between two search spaces, TSP tours (giant tours) and RVRP solutions. This search works on a pair (ω, τ), where ω is a RVRP solution and τ is a giant tour obtained concatenating the routes in ω (no copies of the depot are included). If τ' is obtained by perturbing τ, a *split* process is used and a solution ω' is obtained from τ'. After applying local search to it, it is compared to ω. If better, the latter is updated and a new giant tour is obtained.

The heuristics were tested on two sets of instances randomly generated. The first set consists of 18 small instances, with 10 to 20 customers and 2 to 3 vehicles. Results from these instances were compared to solutions to the MILP formulation of the problem, obtained by GLPK. The second set consists of 24 instances with 50 to 100 customers and 5 to 10 vehicles. Ten out of eighteen instances were solved to optimality by GLPK. The heuristics are very close to GLPK but faster. All 10 proven optima are found and three upper bounds are improved. For the larger instances, the tests show that the best metaheuristic is MS-ILS with giant tours, followed by MS-ILS, ILS and GRASP.

A.2.5 Constructive algorithms

The multi-compartment CVRPSD was solved by three different constructive heuristics in Mendoza et al. (2011). One of them is a stochastic Clarke-Wright heuristic, which starting from a solution with round trips to every customer, merges the routes that will have a better impact in the total expected cost. The two other heuristics are different approaches of a look-ahead heuristic with two steps. In the first, the traveling salesman problem is solved and in the second a clustering procedure is performed. The preferred iterative look-ahead technique (pilot method) (Voß et al., 2005) is used to avoid suboptimal moves of the greedy heuristics used in the routing step (nearest neighbor and nearest insertion). The difference between the two different approaches is given by the heuristic used in the routing step. For the clustering step a dynamic programming algorithm is used. The total expected cost is calculated analytically. A stochastic 2-Opt procedure is applied as post-optimization to the three heuristics, it evaluates the moves using the deterministic values, and just promising moves are evaluated in terms of the stochastic values. Results from the algorithms are compared with the results obtained by a memetic algorithm (Mendoza et al., 2010), where tests show that even though the quality of the solution is not necessarily better, the algorithms are much faster.

The algorithm is also tested for the CVRPSD and compared with results reported by Christiansen and Lysgaard (2007). The two approaches of a the look ahead heuristic algorithms were able to find some new best known solutions, and for instances with a reported optimal solution, the three algorithms were able to find a good solution in short time.

In Juan et al. (2011), the CVRPSD was solved by a multi-start search procedure, combined with the Clarke-Wright heuristic, which the authors report to have behavior similar to GRASP (M.G.C. Resende, 2010). However, the methodology used to solve the problem can be applied with any efficient algorithm for the CVRP. The strategy is to use part of the vehicle capacity as a safety stock, while the remaining capacity is used during the routing step. The CVRPSD is solved as a CVRP and the vehicle capacity is set to the actual capacity minus the safety stock. Once the a solution to the CVRP is obtained, Monte Carlo simulation is used to evaluate it. The reliability of the solution is also computed as the probability of not having failures. Although no comparison is done with other methods, the algorithm is tested on different types of instances. The only comparison is with the best found solution to the deterministic CVRP counterpart of the problem, but measuring the performance of the solution in the stochastic framework.

A multispace sampling heuristic (MSSH) is proposed by Mendoza and Villegas (2013) to find solutions to the CVRPSD. It follows a two-phase solution strategy. In the first phase, it samples multiple solutions. In the second phase it uses the sampled elements to build a solution. It combines a set of randomized heuristics for the TSP, a tour partitioning procedure and a set partitioning model. In phase one it uses a randomized TSP heuristics to build a sample of giant tours (TSP-like). From each tour, every feasible route that can be extracted without changing the order of the customers is added to a set of routes. The objective function value of the best found solution is used in phase two as an upper bound. In phase two a set partitioning formulation of the problem is solved and a solution is assembled using the routes built in phase one. The sampling heuristics are randomized nearest neighbor, randomized nearest insertion, randomized best insertion and randomized farthest insertion. Results are compared to Christiansen and Lysgaard (2007), Mendoza et al. (2011) and Goodson et al. (2012). Instance size ranges from 16 to 60 customers and the demand follows a Poisson distribution. Goodson et al. (2012) is able to find more best known solutions, but MSSH is more stable, since it finds solutions close to the best known solution more often.

A combination of GRASP and a heuristic concentration (HC) is proposed to find solutions to the CVRPSD in Mendoza et al. (2015). The GRASP uses a set of randomized route-first, cluster-second heuristics to generate starting solutions. A variable neighborhood descent procedure is used for the local search phase. A starting solution is constructed greedily using a randomized TSP heuristic that builds a giant tour; then a split procedure is used to get a feasible solution. A VND procedure is later applied to the solution. Once a local optimum is found, its routes are add to a set of routes to be used later by the HC. The route-first heuristics are similar to those used in Mendoza and Villegas (2013). The VND has two neighborhoods, re-locate and 2-opt. The evaluation process of the moves has three steps. First it checks the affected routes for feasibility. If not feasible, the the move is discarded and not evaluated. If feasible, the second step evaluates the deterministic part of the objective function. If the moves leads to a degrading solution (deterministic), then it is discarded. If the move improves the deterministic evaluation of the objective function, the third step consists of analytic evaluation of the objective function.

The heuristic (GRASP + HC) was tested on 40 instances of the classical CVRPSD Christiansen and Lysgaard (2007) where demand follows a Poisson distribution. The heuristic is able to find all the 40 best known solutions, thus outperforming state-of-art metaheuristics Goodson et al. (2012); Mendoza and Villegas (2013) that do not succeed in finding all of them. The instances for the CVRPSD with duration constraints were built by adding duration constraints to 39 of the instances in Christiansen and Lysgaard (2007). Solutions found using the GRASP + HC were compared to the solutions to the classical CVRPSD. Using 0.05 as the maximum probability of violating the

duration constraint, just 3 out of 39 solutions to the classical CVRPSD remain feasible. On the other hand the solutions that fulfill such constraint found by GRASP+HC increase the cost by 2.10% in average. The penalty formulation was computed in three different ways, linear, piecewise linear and quadratic. The performance of the best known solutions to the CVRPSD, measured as total expected overtime and the total expected overtime cost, was compared to the performance of the solutions found by the penalty formulation of GRASP+HC. The linear penalty reduces the expected overtime by 52.83%, the piecewise linear by 75.51% and quadratic by 93.52%. There is an increment on the expected duration of the routes; on average it increases 0.79%, 1.89% and 4.79% respectively.

As a way to illustrate the advantages of modeling stochastic time by Phase-type (PH) distributions, Gómez et al. (2015) uses a multispace sampling heuristic (MSSH) (Mendoza and Villegas, 2013) to deal with the distance-constrained CVRP with stochastic travel and service times (DCVRPSTT). This selection was done for convenience, since MSSH explores the same areas of the solution space independently of the route evaluator being used. A route evaluator using PH distributions and one based on normal distributions and Monte Carlo simulation are embedded into an adaptation of MSSH employing a two phase solution strategy. In the first part it samples multiple solutions. In the second phase it builds the best possible solution using parts from the sample solutions. Instances were adapted from the literature and for each instance three scenarios are assumed, having a different probability distribution for the travel time on each scenario: Erlang, lognormal and Burr distribution. The service level is the same in all cases. Two alternatives are considered for service time: deterministic and exponentially distributed. When travel time is modeled with an Erlang distribution, it was found that the normal route evaluation is the best choice, and the simulation route evaluation, the worst. When travel time is modeled with a lognormal distribution, PH route evaluator finds better routes, but is more computationally expensive. When travel time is modeled with a lognormal distribution, the PH route evaluator finds better routes, normal route evaluation is not able to find feasible solutions. In general it was found that the Normal route evaluator is a good option when the travel times do not have a large skewness. Monte Carlo simulation lead to overly optimistic solutions that do not satisfy the chance constraints. The PH route evaluator leads the algorithm to find solutions with similar quality as Monte Carlo simulation, but more reliable.

A.2.6 Progressive hedging

In Hvattum et al. (2006) a dynamic CVRP with stochastic customers and demands and time windows was solved using a sample scenario hedging heuristic, called dynamic stochastic hedging heuristic (DSHH), based on progressive hedging (Rockafellar and Wets, 1991) for the CVRP. Sample scenarios are used to guide the heuristics that build a plan for each time interval. Each scenario is solved as a deterministic CVRP using a simple insertion heuristic, common parts are used to build a solution to the dynamic and stochastic problem. The algorithm is compared against two different methods, a local search heuristic that solves the deterministic problem and a myopic dynamic heuristic that solves the problem at each stage ignoring the stochastic information. DSHH was able to find solutions with a travel distance more than 15% shorter than the myopic dynamic heuristic.

A.2.7 Evolutionary algorithms

A genetic algorithm is used in Ando and Taniguchi (2006) to solve the CVRP with soft time windows and stochastic travel times. However, a detailed description of the algorithm is not presented. The quality of the solutions is evaluated using simulation. The best found solution is compared against the usual operation, for five days. The solution found by the algorithm performs better. The average of the total cost was reduced by about 4%, and its standard deviation was reduced by about 75%. Having as result a more reliable route.

The multi-compartment CVRPSD was solved using a memetic algorithm in Mendoza et al. (2010), where the solutions obtained by a genetic algorithm are improved by two local search procedures: relocate and 2-opt. The reparation and fitness evaluation of the individuals in the population is performed in an analytic way through *stochastic split* (s-split), which is an extension of the Prins algorithm (Prins, 2004). The results obtained by the memetic algorithm were compared with the stochastic Clarke-Wright, where all possible merges are recalculated at each iteration, and with a spare capacity strategy, which consists of solving the deterministic version of the problem (using a memetic algorithm in this particular case), but preserving some free capacity on each compartment. The proposed memetic algorithm is shown to find better solutions; however, it is more time consuming. The algorithm was also tested on a set of instances for the multi-compartment CVRP, and the results were compared against a tabu search and a memetic algorithm reported by Fallahi et al. (2008). Some new best solution were obtained and on average, results show a gap of about 1% with respect to the best known solutions.

A multi-objective CVRPSD was solved using a multi-objective evolutionary algorithm in Tan et al. (2007). The quality of a solution is computed by means of a route simulation method, where several sets of demands for all customers are generated. The solution is evaluated on each set and an average value is obtained for each objective. The obtained averages are used to rank the solutions based on the Pareto dominance concept, using for this purpose the expected values of the objectives.

In Zhang et al. (2012), the stochastic travel-time CVRP with simultaneous pick-ups and deliveries is solved using a scatter search heuristic, with the initial set of solutions created using a variant of the Clarke-Wright algorithm. In this problem the chance constraints (time limit constraints) are transformed into fixed constraints and the problem can be solved as deterministic problem. The results are compared against results obtained by a genetic algorithm, the proposed algorithm produce solutions that are on average up to 13% better than the genetic algorithm.

A memetic algorithm with population management was used to deal with robustness and flexibility of the CVRP with stochastic demands and travel cost, and the VRP with stochastic customers (Sörensen and Sevaux, 2009), respectively. The algorithm consists of a genetic algorithm hybridized with two versions of tabu search that are used alternatively. The diversity of the population is controlled with a distance measure. This allows the population to be small, but keeps it diverse. The edit distance that measure the number of steps (add character, remove character, substitute character) that would be performed in one solution to become another, is used to measure the average distance between a solution and the population. The distance is required to be above a certain threshold, before the solution is added to the population. The threshold is decreased to intensify the search and it is increased again when the search is stuck in a local optimum.

The two tabu search heuristics are insert tabu search, and swap tabu search. The former attempts to insert any customer at any other tour and the latter attempts to swap any pair of customers in the solution. Robustness evaluation was used in the binary tournament of the genetic algorithm (by simulation), however it was not used in the tabu search procedures, where the deterministic objective function is used to select the next move. It was found that robust solutions to the CVRP with stochastic demands and travel cost will have a good deterministic objective function value, the reverse is not always true. For the CVRP with stochastic customers, it was found that the robust approach is not profit maximizing, since the best solution to the deterministic problem, when all customers require the service, is likely to be also the best solution to the problem with a reduced set of customers.

The SCARP was solved by a process consisting of two parts (Fleury et al., 2002, 2005b), optimization and robustness evaluation. The optimization is done by a genetic algorithm that uses a local search as a mutation operator, designed for solving the deterministic CARP (Lacomme et al.,

2001), but not all solutions are subject to local search. The algorithm stops after a specific number of iterations, or after a certain number of iterations with no improvement or when reaching a lower bound. If the lower bound is not reached in the main phase, several short restarts are performed with a higher probability for the solutions to be subjected to a local search procedure. Such procedure performs best-improvement moves and stops when no more improvements are available. Local search moves include removal of one or two consecutive edges (that require service) from a route, and reinsertion in another position, exchange of two edges, and 2-opt moves. The robustness evaluation is done to the best solution found by the optimization process. It consists of a statistic evaluation of performance indicators such as the average cost, average number of trips, percentage of solutions requiring extra trips to the depot, standard deviation of the total cost and standard deviation of the total number of trips.

For statistical purposes, the solution is evaluated in several independent scenarios. The optimization part is done using three different approaches, *tight* and *slack*. In both cases the deterministic problem is solved using the expected values of the demands as parameters. In the *slack* approach, just a percentage of the vehicle's capacity is utilized when making routing decisions. The *slack* approach shows reduction of the variability, if compared with the *tight* approach.

In addition to the *tight* and *slack* approaches, a memetic algorithm where the objective function is computed analytically was used in Fleury et al. (2004), a similar approach was called *law* in Fleury et al. (2005a). In the *law* approach, two objective functions are considered to be minimized: the average cost of the solution and the average cost plus a fixed constant multiplied by the standard deviation of the cost. Even though different instances were used in Fleury et al. (2005a) and Fleury et al. (2005a), the results are similar. The *law* approach, when the average cost is minimized, produces solutions with less variability than the *tight* approach, but not as good as the *slack* approach. However, when it uses as objective function the average cost plus a fixed constant multiplied by the standard deviation of the cost, it generates solutions with less variability.

A.2.8 Other nature-inspired heuristics

In Marinakis et al. (2013), The CVRPSD was solved by particle swarm optimization, a methodology based on simulating the social behavior of swarming organisms. The method includes some other heuristics, since in addition to the particle swarm, it also uses path relinking and local search (2-opt and 3-opt). Thus, in some sense it can be viewed as a multi-descent algorithm. In the model of the problem it was assumed that route failures could be avoided by the "optimum choice" of a threshold value to decide whether to travel to the depot for preventive restocking or not. The fitness of a solution is deterministic and evaluated analytically. The results are compared against results obtained by two other evolutionary algorithms, and the particle swarm optimization heuristic was able to outperform them.

The single vehicle CVRPSD was solved using Cross Entropy (CE) method by Rubinstein (1999) in Chepuri and Homem-de Mello (2005). This method considers that the actual optimization problem is connected to a problem of estimating rare-event probabilities. The idea behind CE is to see the selection of the optimal solution as a rare event. At every iteration a set of routes are generated according to transition probabilities, the cost of the routes is evaluated and probabilities are updated depending on that cost. For the CVRPSD the algorithm starts from an initial transition matrix so any route has the same probability to be generated, and a set of routes is generated. The routes are then evaluated using a demand sample, which is the same for each route. The best route so far is kept and compared to the one generated at each iteration. If no improvement are achieved for a given number of iterations, the algorithm stops.

Lower bounds and exact solutions are computed and several type of analysis are done for different cases: demands are IID and penalties are identical; demands are IID and penalties are not identical; demands are non-IID and penalties are identical; demands are non-IID and penalties are non-identical. If demands are IID, the expected cost can be computed analytically. If demands are non-IID, expected cost of going back to depot is ignored and a lower bound is obtained for the total cost. Demands are drawn from the same family of distributions, but the parameters depend on the node. A lower bound is also obtained for the case where demands nor non-IID and the penalties are non-uniform. For IID demands, the algorithm is compared with results of a branch-and-bound technique using the ILOG SOLVER 4.4. In most of the cases the algorithm is able to find solutions within 5% of the optimal solution. Solver is not able to find exact solutions for more than 16 nodes. For non-IID demands and uniform penalties, a lower bound of the problem is found by ILOG SOLVER 4.4. For tighter problems (the total demand is close to total capacity), values are closer to the lower bound. For non-IID demands and non-uniform penalties, no lower bound is available, but a similar conclusion is obtained regarding solution quality of tighter problems. In elite sampling, some randomly generated routes are replaced by the best routes found so far, which improves the performance of the algorithm.

Ant colony optimization is used in Woensel et al. (2007) to solve the routing problems with time-dependent travel times. A 2-opt procedure is used together with a mechanism that splits the tours by adding a depot between two customers. This is done until no improvement is achieved or the maximum number of trucks is reached. In addition, the starting time may be shifted if that represents an improvement. The quality of the solutions obtain was compared with the quality of the solutions when just the total distance is minimized. In addition, explicit enumeration is done for small instances to validate the approach. Tests are done using both objective functions: with and without variance of the travel times (except for complete enumeration which does not include the variance in the objective function.) The algorithm was tested in 28 instances from the literature with 32 to 100 customers in addition to the depot. On average there is a reduction of 22.2% in the travel time. For tests of the second objective function, the roads are randomly selected to have either a high or low coefficient of variation: 50% of roads have high and 50% low. The coefficient of variation of the travel time decreased on average 54.30% with a weight associated to the variance in the interval $[0, 0.1]$. The cost of that reduction is an increment in the average travel time of 27.87%. In the tests, the fleet size is considered unlimited.

References

Adulyasak, Y. and Jaillet, P. (2014). Models and algorithms for stochastic and robust vehicle routing with deadlines. *Transportation Science, forthcoming*, pages –.

Ahmadi-Javid, A. and Seddighi, A. H. (2013). A location-routing problem with disruption risk. *Transportation Research Part E: Logistics and Transportation Review*, 53:63 – 82.

Ak, A. and Erera, A. L. (2007). A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation Science*, 41(2):222–237.

Andersson, H., Hoff, A., Christiansen, M., Hasle, G., and Løkketangen, A. (2010). Industrial aspects and literature survey: Combined inventory management and routing. *Computers & Operations Research*, 37(9):1515 – 1536.

Ando, N. and Taniguchi, E. (2006). Travel time reliability in vehicle routing and scheduling with time windows. *Networks and Spatial Economics*, 6(3-4):293–311.

Beraldi, P., Ghiani, G., Musmanno, R., and Vocaturo, F. (2010). Efficient neighborhood search for the probabilistic multi-vehicle pickup and delivery problem. *Asia - Pacific Journal of Operational Research*, 27(3):301–314.

Bertsimas, D., Chervi, P., and Peterson, M. (1995). Computational approaches to stochastic vehicle routing problems. *Transportation Science*, 29(4):342–352.

Bianchi, L., Birattari, M., Chiarandini, M., Manfrin, M., Mastrolilli, M., Paquete, L., Rossi-Doria, O., and

Schiavinotto, T. (2005). Hybryd metaheuristics for the vehicle routing problem with stochastic demands. Technical report, Dalle Molle Institute for Artificial Intelligence, Manno, Switzerland.

Birge, J. and Louveaux, F. (1997). *Introduction to stochastic programming*. Springer, New York.

Chen, L., H, M. H., Langevin, A., and Gendreau, M. (2014). Optimizing road network daily maintenance operations with stochastic service and travel times. *Transportation Research Part E: Logistics and Transportation Review*, 64:88–102.

Chepuri, K. and Homem-de Mello, T. (2005). Solving the vehicle routing problem with stochastic demands using the cross-entropy method. *Annals of Operations Research*, 134(1):153–181.

Christiansen, C. H. and Lysgaard, J. (2007). A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, 35(6):773 – 781.

Christiansen, C. H., Lysgaard, J., and Wøhlk, S. (2009). A branch-and-price algorithm for the capacitated arc routing problem with stochastic demands. *Operations Research Letters*, 37(6):392–398.

Clarke, C. and Wright, J. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:432–441.

Coelho, L. C., Cordeau, J.-F., and Laporte, G. (2014). Thirty years of inventory routing. *Transportation Science*, 48(1):1–19.

Cordeau, J.-F., Laporte, G., Savelsbergh, M. W., and Vigo, D. (2007). Chapter 6. Vehicle routing. In Barnhart, C. and Laporte, G., editors, *Transportation*, volume 14, pages 367–428. Elsevier.

Desrochers, M. (1988). An algorithm for the shortest path problem with resource constraints. Technical report, Cahiers du GERAD G-88-27, University of Montreal.

Dueck, G. (1993). New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, 104(1):86–92.

Erera, A. L., Morales, J. C., and Savelsbergh, M. (2010). The vehicle routing problem with stochastic demand and duration constraints. *Transportation Science*, 44(4):474–492.

Erera, A. L., Savelsbergh, M., and Uyar, E. (2009). Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints. *Networks*, 54(4):270–283.

Fallahi, A. E., Prins, C., and Wolfler Calvo, R. (2008). A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research*, 35(5):1725–1741.

Feillet, D., Dejax, P., Gendreau, M., and Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229.

Fleury, G., Lacomme, P., and Prins, C. (2004). Evolutionary algorithms for stochastic arc routing problems. In *Lecture Notes in Computer Science*, volume 3005, pages 501–512. Springer Berlin Heidelberg.

Fleury, G., Lacomme, P., Prins, C., and Ramdane-Cérif, W. (2005a). Improving robustness of solutions to arc routing problems. *The Journal of the Operational Research Society*, 56(5):526–538.

Fleury, G., Lacomme, P., Prins, C., and Randame-Chérif, W. (2002). Robustness evaluation of solutions for the capacitated arc routing problem. In Barros, F. J. and Gambiasi, N., editors, *AI, Simulation and planning in high autonomy systems*, Lisbon, Portugal.

Fleury, G., Philippe, L., and Prins, C. (2005b). Stochastic capacitated arc routing problem. Technical Report LIMOS/RR-05-12, Université Blaise Pascal - Laboratorie d'informatique(LIMOS).

Gauvin, C., Desaulniers, G., and Gendreau, M. (2014). A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 50(0):141–153.

Gendreau, M., Jabali, O., and Rei, W. (2014). Stochastic vehicle routing problems. In Toth, P. and Vigo, D., editors, *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 213–239. Society for Industrial and Applied Mathematics.

Gendreau, M., Laporte, G., and Séguin, R. (1996a). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12.

Gendreau, M., Laporte, G., and Seguin, R. (1996b). A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477.

Golden, B. L., Magnanti, T. L., and Nguyen, H. Q. (1977). Implementing vehicle routing algorithms. *Networks*, 7(2):113–148.

Gómez, A., no, R. M., Akhavan-Tabatabaei, R., Medaglia, A. L., and Mendoza, J. E. (2015). On modeling stochastic travel and service times in vehicle routing. *Transportation Science, forthcoming*, pages –.

Goodson, J. C. (2015). A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 241(2):361–369.

Goodson, J. C., Ohlmann, J. W., and Thomas, B. W. (2012). Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand. *European Journal of Operational Research*, 217(2):312 – 323.

Gounaris, C. E., Wiesemann, W., and Floudas, C. A. (2013). The robust capacitated vehicle routing problem under demand uncertainty. *Operations Research*, 61(3):677–693.

Gouveia, L. (1995). A result on projection for the vehicle routing problem. *European Journal of Operational Research*, 85(3):610–624.

Groër, C., Golden, B., and Wasil, E. (2009). The consistent vehicle routing problem. *Manufacturing & Service Operations Management*, 11(4):630–643.

Haugland, D., Ho, S. C., and Laporte, G. (2007). Designing delivery districts for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 180(3):997–1010.

Hjorring, C. and Holt, J. (1999). New optimality cuts for a single-vehicle stochastic routing problem. *Annals of Operations Research*, 86:569–584.

Hoff, A., Andersson, H., Christiansen, M., Hasle, G., and Løkketangen, A. (2010). Industrial aspects and literature survey: Fleet composition and routing. *Computers & Operations Research*, 37(12):2041 – 2061.

Hvattum, L. M., Løkketangen, A., and Laporte, G. (2006). Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science*, 40(4):421–438.

Jabali, O., Rei, W., Gendreau, M., and Laporte, G. (2014). Partial-route inequalities for the multi-vehicle routing problem with stochastic demands. *Discrete Applied Mathematics*, 177:121–136.

Juan, A., Faulin, J., Grasman, S., Riera, D., Marull, J., and Mendez, C. (2011). Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands. *Transportation Research Part C: Emerging Technologies*, 19(5):751–765. Freight Transportation and Logistics (selected papers from {ODYSSEUS} 2009 - the 4th International Workshop on Freight Transportation and Logistics).

Kall, P. and Wallace, S. W. (1994). *Stochastic programming*. John Wiley & Sons, Chichester.

Kenyon, A. S. and Morton, D. P. (2003). Stochastic vehicle routing with random travel times. *Transportation Science*, 37(1):69–82.

Kulkarni, R. V. and Bhave, P. R. (1985). Integer programming formulations of vehicle routing problems. *European Journal of Operational Research*, 20(1):58–67.

Lacomme, P., Prins, C., and Ramdane-Chérif, W. (2001). A genetic algorithm for the capacitated arc routing problem and its extensions. In Boers, E., editor, *Lecture Notes in Computer Science*, volume 2037, pages 473–483. Springer Berlin Heidelberg.

Laporte, G., Louveaux, F., and Mercure, H. (1992). The vehicle routing problem with stochastic travel times. *Transportation Science*, 26(3):161–170.

Laporte, G., Louveaux, F. V., and Hamme, L. V. (2002). An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423.

Laporte, G., Musmanno, R., and Vocaturo, F. (2010). An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands. *Transportation Science*, 44(1):125–135.

Laporte, G., Nobert, Y., and Desrochers, M. (1985). Optimal routing under capacity and distance restrictions. *Operations Research*, 33(5):1050–1073.

Lee, C., Lee, K., and Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *The Journal of the Operational Research Society*, 63(9):1294–1306.

Lei, H., Laporte, G., and Guo, B. (2011). The capacitated vehicle routing problem with stochastic demands and time windows. *Computers & Operations Research*, 38(12):1775–1783.

Lei, H., Laporte, G., and Guo, B. (2012). A generalized variable neighborhood search heuristic for the capacitated vehicle routing problem with stochastic service times. *TOP*, 20(3):99 – 118.

Li, X., Tian, P., and Leung, S. C. (2010). Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics*, 125(1):137–145.

Marinakis, Y., Iordanidou, G.-R., and Marinaki, M. (2013). Particle swarm optimization for the vehicle routing problem with stochastic demands. *Applied Soft Computing*, 13(4):1693 – 1704.

Mendoza, J., Rousseau, L.-M., and Villegas, J. (2015). A hybrid metaheuristic for the vehicle routing problem with stochastic demand and duration constraints. *Journal of Heuristics*, pages 1–28.

Mendoza, J. and Villegas, J. (2013). A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. *Optimization Letters*, 7(7):1503–1516.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2011). Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45(3):346–363.

M.G.C. Resende, C. R. (2010). Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau, J.-Y. P., editor, *Handbook of Metaheuristics*, pages 283–319. Springer.

Neuts, M. (1981). *Matrix-geometric Solutions in Stochastic Models: An Algorithmic Approach*. Dover Publications.

Pillac, V., Gendreau, M., Guret, C., and Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1 – 11.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002.

Rei, W., Gendreau, M., and Soriano, P. (2007). Local branching cuts for the 0-1 integer L-shaped algorithm. *Technical report, CIRRELT-2007-23, Centre interuniversitaire de recherche sur les réseaux d'enterprise, la logistique et le transport*, 44(1):136–146.

Rei, W., Gendreau, M., and Soriano, P. (2010). A hybrid monte carlo local branching algorithm for the single vehicle routing problem with stochastic demands. *Transportation Science*, 44(1):136–146.

Rockafellar, R. and Wets, R.-B. (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147.

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1(2):127–190.

Russell, R. A. and Urban, T. L. (2008). Vehicle routing with soft time windows and erlang travel times. *The Journal of the Operational Research Society*, 59(9):1220–1228.

Secomandi, N. (2000). Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27(1112):1201–1225.

Secomandi, N. (2001). A rollout policy for the vehicle routing problem with stochastic demands. *Operations Research*, 49(5):796–802.

Secomandi, N. (2003). Analysis of a rollout approach to sequencing problems with stochastic routing applications. *Journal of Heuristics*, 9(4):321–352.

Secomandi, N. and Margot, F. (2009). Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230,.

Solano-Charris, E., Prins, C., and Santos, A. C. (2015). Local search based metaheuristics for the robust vehicle routing problem with discrete scenarios. *Applied Soft Computing*, 32:518–531.

Sörensen, K. and Sevaux, M. (2009). A practical approach for robust and flexible vehicle routing using metaheuristics and monte carlo sampling. *Journal of Mathematical Modelling and Algorithms*, 8(4):387–407.

Sungur, I., Ordóñez, F., and Dessouky, M. (2008). A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions*, 40:509–523.

Sungur, I., Ren, Y., Ordez, F., Dessouky, M., and Zhong, H. (2010). A model and algorithm for the courier delivery problem with uncertainty. *Transportation Science*, 44(2):193–205.

Taş, D., Dellaert, N., van Woensel, T., and de Kok, T. (2013). Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research*, 40(1):214 – 224.

Taş, D., Gendreau, M., Dellaert, N., van Woensel, T., and de Kok, A. (2014). 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.

Tan, K., Cheong, C., and Goh, C. (2007). Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813–839.

Toth, P. and Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1-3):487 – 512.

Voß, S., Fink, A., and Duin, C. (2005). Looking ahead with the pilot method. *Annals of Operations Research*, 136(1):285–302.

Woensel, T. V., Kerbache, L., Peremans, H., and Vandaele, N. (2007). A queueing framework for routing problems with time-dependent travel times. *Journal of Mathematical Modelling and Algorithms*, 6(1):151–173.

Yang, W.-H., Mathur, K., and Ballou, R. H. (2000). Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(1):99–112.

Zhang, T., Chaovalitwongse, W., and Zhang, Y. (2012). Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. *Computers & Operations Research*, 39(10):2277 – 2290.

Zhong, H., Hall, R. W., and Dessouky, M. (2007). Territory planning and vehicle dispatching with driver learning. *Transportation Science*, 41(1):74–89.

Zhu, L., Rousseau, L.-M., Rei, W., and Li, B. (2014). Paired cooperative reoptimization strategy for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 50(0):1–13.

Paper 4

The CVRP with route balancing and stochastic demand

The capacitated vehicle routing problem with route balancing and stochastic demand

Jorge Oyola and Halvard Arntzen

Molde University College, Molde, Norway

Abstract

This paper introduces a new problem to the VRP literature, a multi-objective stochastic optimization problem labeled the *Capacitated VRP with Route Balancing and Stochastic demand*. Extending the VRP with route balancing with random elements leads to modeling choices, that are discussed in the paper. We develop a solution methodology, the stochastic multi-objective GRASP (SMGRASP), based on a multi-objective GRASP algorithm. Computational experiments are performed, showing favorable results, compared to a method developed for a similar problem. Both in-sample and out-of-sample stability tests were conducted.

Keywords: Multi-objective VRP, Stochastic VRP (SVRP), GRASP

1 Introduction

The well known *capacitated vehicle routing problem* (CVRP) is defined over an undirected graph $G(V, E)$, where $V = v_0, \ldots, v_N$ is a set of vertices and $E = (v_i, v_j) : v_i, v_j \in V, i < j$ is a set of edges. There is a symmetric matrix $C = [c_{ij}]$ that correspond to the travel costs along edge (v_i, v_j). Vertex v_0 represents the depot where there is a homogeneous fleet of m vehicles with capacity Q. A set of customers $V \setminus v_0$ with a non-negative known demand d_i must be served. A solution to the CVRP consists of m delivery routes. These routes must be designed to ensure that each route start and end at the depot. Each customer must be visited once by exactly one vehicle. The summation of the demands of the customers in the same route, must be less than or equal to the vehicle's capacity. A different approach where the demand corresponds to items that must be collected from the customers leads to an equivalent problem. The classic objective is minimization of total route cost (Toth and Vigo, 2002).

In the real world one or more of the elements in a routing problem can be uncertain. Different formulations try to deal with such uncertainty by including stochastic parameters in the model. Vehicle routing problems (VRP) that consider uncertainty including stochastic parameters are called stochastic VRP (SVRP). In the SVRP literature it is possible to find different examples of parameters used to model the uncertainty. These include the demand, customers, travel times, service time and travel costs. The demand is currently the most studied stochastic parameter in SVRP (Gendreau et al., 1996, 2014). When the stochastic problem takes into consideration the capacity constraint of the vehicles is possible to call it stochastic CVRP (SCVRP).

In the SVRP the decision maker (DM) must choose a solution (at least partially) *before* the exact values of all parameters are completely known. This solution is called *a priori* solution. Once the *a priori* solution is implemented, some constraints may be violated when (part of) the actual parameter values are realized, e.g. the total realized demand of a planned route may actually exceed the vehicle's capacity. In this case, it is said that the solution (or the route) "fails". There are two common ways of modeling stochastic problems: as a *chance constrained program* (CCP) or

as a *stochastic program with recourse* (SPR). In the first case, the problem is solved ensuring that the probability of route failure is below a certain level and the cost of failures is typically ignored (Gendreau et al., 1996; Tan et al., 2007). In the latter case, route failures are allowed, but the DM must define a *recourse* policy, describing what actions to take in order to repair the solution after a failure. The expected transportation cost (travel cost of *a priori* solution + recourse policies cost) is optimized. SPR is more difficult to solve, but objectives are more meaningful (Gendreau et al., 1996).

Traditionally, in a SCVRP, the expected total transportation cost is minimized. This may produce substantial differences between the route lengths. In some industries the income of the driver may be affected by the traveled distance, this could be seen as unfair by the drivers. In addition, the fact of all not having the same workload can generate problems among them or between drivers and DM, affecting their welfare. In some settings, drivers are considered to be an arena of competition among the transportation companies, making their welfare a significant issue (Lee and Ueng, 1999). Generally, differences in route length are described by *route balance* measures, which may be subject to constraints or may be treated as second objective in addition to total distance.

Including the route balance in the SCVRP may describe a problem closer to real-life. This paper deals with a bi-objective vehicle routing problem with stochastic demands, where the balance of the expected route lengths is included as an objective in addition to the traditional minimization of the total expected transportation distance. The route balance is measured as the difference between the longest and the shortest expected route length. If the adopted recourse policy is to have the vehicle return to depot to reestablish capacity, one realizes that route failures may also affect route balance. As a consequence, some additional modeling choices must be made regarding the route balance, once stochastics enter the problem. We discuss this particular issue in section (2.3).

The deterministic version of this problem (where demands are not stochastic) is known as the capacitated vehicle routing problem with route balancing (CVRPRB) (Jozefowiez et al., 2007b, 2009; Oyola and Løkketangen, 2014). The extended version of the problem considered here is the capacitated vehicle problem with route balancing and stochastic demands (CVRPRBSD). In practice, the consideration of route balance is relevant since due to drivers agreements, legal restrictions or fairness, the drivers' workload might become an important aspect to be considered by the decision maker when doing the transportation planning.

Several approaches have been used to deal with the VRPRB. These approaches involve evolutionary algorithms, either combined with tabu search (Jozefowiez et al., 2002, 2007b) or with additional diversification strategies (Jozefowiez et al., 2009). In Oyola and Løkketangen (2014), an algorithm based on the GRASP metaheuristic was proposed.

For the case of the CVRPSD a wider range of approaches have been used, e.g exact methods (Hjorring and Holt, 1999; Laporte et al., 2002; Rei et al., 2007; Christiansen and Lysgaard, 2007; Gauvin et al., 2014; Jabali et al., 2014), dynamic programming (Yang et al., 2000; Secomandi and Margot, 2009; Secomandi, 2000; Zhu et al., 2014; Secomandi, 2003), rollout algorithm (Secomandi, 2001), tabu search (Haugland et al., 2007; Ak and Erera, 2007; Bianchi et al., 2005), simulated annealing (Bianchi et al., 2005), iterated local search (Bianchi et al., 2005), ant colony optimization (Bianchi et al., 2005), evolutionary algorithm (Bianchi et al., 2005; Mendoza et al., 2010; Tan et al., 2007), local hybrid heuristics (Rei et al., 2010), local search heuristic (Erera et al., 2009), constructive heuristics (Mendoza et al., 2011), cross entropy (Chepuri and Homem-de Mello, 2005).

The literature on multi-objective stochastic CVRP is on the other hand scarce. A multi-objective approach to the CVRP with stochastic demands (CVRPSD) was formulated in Tan et al. (2007). Three main objectives are minimized: the total travel time, the number of vehicles and drivers remuneration. An evolutionary algorithm was used to deal with that problem. In Juan et al. (2011) the CVRPSD is not explicitly presented as a bi-objective problem, but a tradeoff between the total expected cost and the probability of the solution suffering a route failure (reliability) is taken into consideration. This problem was solved as a single-objective problem using a multi-start search procedure, combined with the Clarke-Wright heuristic. An extension of the CVRP including location, allocation and routing under the risk of disruption is introduced in Ahmadi-Javid and Seddighi (2013). Location decisions are connected with potential producer-distributors with random capacity due to disruptions. Customers with known, non-negative demands, must be allocated to a producer-distributor. The vehicles serving the customers may suffer disruptions. The problem is not entirely treated as a multi-objective. The decision maker is presented with three different solutions: one obtained by minimizing the expected cost (moderate risk level), the second one by minimizing the conditional value-at-risk (cautious) and the third solution is obtained by considering the the worst case value for the stochastic parameters (pessimistic). A local search heuristic including simulated annealing and a 2-opt procedure was proposed for solving the problem. A multi-objective CVRP with soft time windows and stochastic travel times (SCVRPSTW) is found in Russell and Urban (2008). In the SCVRPSTW three objectives are taken into consideration, the minimization of the number of vehicles, the total traveled distance and the total expected penalties for earliness and lateness in the service. A tabu search heuristic is used to find solutions to the problem.

To the best of our knowledge two different versions of multi-objective CVRPSD have been previously studied. In Tan et al. (2007) the solutions are evaluated through simulation and the objectives are treated independently. In Juan et al. (2011) the solutions are evaluated in a deterministic way, since the problem solved is a deterministic transformation of the original, the best found solution is the one evaluated using simulation. The problem in Tan et al. (2007) is the closest to the CVRPRBSD, since the objectives in both problems can be somehow assimilated to one another. The travel time can be assimilated to the travel distance. In the CVRPSD, the number of vehicles and the travel distance (travel time) are positively correlated (Yang et al., 2000; Tan et al., 2007), so if the total distance is optimized, as in the CVRPRBSD, one could say that the number of vehicles is implicitly optimized. In Tan et al. (2007) the drivers remuneration is increased if the length of his/her tour is greater than a given value B, which is connected with the route balance, since it is expected that long routes would be avoided.

In this paper, a new algorithm is proposed to find solutions to the CVRPRBSD, it is a variant of a greedy randomized adaptive search procedure (GRASP) (Resende and Ribeiro, 2003, 2010) that includes an new insertion strategy. The GRASP metaheuristic has been previously used in solving multi-objective problems, e.g. knapsack and rule selection (Reynolds et al., 2009; Vianna and Arroyo, 2004) and CVRPRB (Oyola and Løkketangen, 2014). Obtained results will be compared with the results of an implementation of the evolutionary algorithm in Tan et al. (2007), which will be hereafter referred to as as TAN.

Following the spirit of the ruin and recreate heuristic (Schrimpf et al., 2000), the constructive phase of the GRASP procedure starts from a partially built solution. This initial partial solution is obtained using the common parts of two previously found solutions (ruin). A GRASP procedure is then applied to complete the solution (recreate). This process is called the greedy randomized adaptive search procedure with advanced starting point (GRASP-ASP) (Oyola and Løkketangen, 2014). A traditional GRASP procedure, starting from an empty solution, is also applied in an effort to diversify the set of solutions. The effectiveness of an insertion strategy based on the distance from the depot to the customers and the use of a virtual capacity during the planning process are evaluated.

The two objectives are treated independently and the quality of the solutions is evaluated by using sample average approximations. The implemented algorithm is a modification of GRASP-ASP (Oyola and Løkketangen, 2014) and will be hereafter referred to as the stochastic multi-objective GRASP (SMGRASP).

The rest of the paper is organized as follows: in Section 2 a discussion on modeling issues is presented, the algorithm used to deal with the problem studied here is described in Section 3, tests results and conclusions are presented in sections 4 and 5 respectively.

2 Model discussion

A formal mathematical model of the (deterministic) CVRPRB can be found in e.g. Oyola and Løkketangen (2014). Introducing stochastic demands means that the demanded quantity of customer i is modeled by a random variable $\boldsymbol{\xi}_i$, whose value ξ_i is only revealed after the routes are planned. In the following we use boldface symbols (e.g. $\boldsymbol{\xi}$) to denote a random variable, while a realized value for $\boldsymbol{\xi}$ is denoted by ξ.

The transition from a deterministic to a stochastic problem forces a number of modeling choices to be made. Let us first of all make sure it is understood that the CVRPRBSD is a *bi-objective* SVRP problem, so a solution to the CVRPRBSD is a set X of SVRP solutions x. Each SVRP solution on the other hand, is a set of planned routes, that may fail on execution, leading to a recourse action. The set X will normally be an approximation to the Pareto set of the bi-objective problem (or it can theoretically be the exact Pareto set). However, to make sure these concepts are meaningful, we need to make several modeling choices. Some choices are related to the individual SVRP problem: probability distributions, objective functions, recourse policy. One choice is linked to the bi-objective nature of the problem: the choice of a *domination concept* for comparing SVRP solutions. Finally there are choices regarding the computational framework, notably how to approximate objective function values when the exact computation becomes prohibitive.

2.1 Probability distribution

From a modeling perspective, any probability distribution leading to positive values can be used for the demand vector $\boldsymbol{\xi}$. For our computational experiments, we use Binomial distributions, mainly because it allows us to use integer demands without making any kind of approximation. We also suppose the random variables representing demand are independent. However, the algorithm can be used and tested on any other distribution.

2.2 Recourse

An SVRP solution x in this problem suffers a *route failure* if the accumulated demand of the customers on a route turns out to exceed or reach the vehicle capacity. The recourse policy (or action) is to reroute the vehicle to the depot to refill, and to resume service at the customer where failure occurred, in case the vehicle capacity has been exceeded. The vehicle resumes the service at the customer following the one where the failure occurred when the capacity is not exceeded but reached.

In general there is a desire to consider more sophisticated recourse policies in SVRP. However the “detour to depot” recourse policy is widely used in the literature, and secondly the topic of this research (Multi-objective stochastic VRP) is not the first place to start if one wants to experiment with more complex recourse policies.

We note however, that the proposed recourse policy does imply that randomness in the demands directly influences on both objectives (distance and route balance). Moreover, we note that a route failure will always make the affected route longer. This means on execution, the total distance of the VRP solution will increase, while the route balance may increase, decrease or remain unchanged.

2.3 Objectives

The two objectives to be defined are (i): expected total distance (cost) $C(x)$, (ii): expected route balance, $R(x)$, both including recourse effects. Here, x denotes a single solution, a set of planned routes for the vehicles. While it is obvious what the total distance means in this problem, it is not so for the expected route balance. In a deterministic CVRPRB, (see e.g. Jozefowiez et al. (2009)), the route balance is simply the difference between the longest and the shortest route. However, taking the expected value of this is not the only option. In fact we shall adopt an alternative definition of route balance following the discussion below.

There are two fairly natural choices for route balance in an SVRP, with somewhat different meanings. To explain, let $\boldsymbol{\xi}$ represent the random demand vector, and let ξ denote a particular realization (a demand scenario). Then we have two basic options.

- For a given scenario ξ, let $d_i(x, \xi)$ denote the distance traveled by vehicle i under that scenario. The *inter-scenario route balance* $R^{IS}(x)$ is simply the expected value over all scenarios of the maximal difference between $d_i(x, \xi)$ values:

$$R^{IS}(x) = \mathsf{E}\,[\,\max_{i,j}(d_i(x, \boldsymbol{\xi}) - d_j(x, \boldsymbol{\xi}))\,]\ .$$

  This is what one might intuitively think of if we say “make a stochastic CVRPRB model”.

- Alternatively, for each vehicle i, let $D_i(x)$ be the expected distance over all scenarios, i.e. $D_i(x) = \mathsf{E}\,[\,d_i(x, \boldsymbol{\xi})\,]$. Now define the *inter-vehicle route balance* $R^{IV}(x)$ as the difference between the maximal and minimal $D_i(x)$ value over all vehicles, in short

$$R^{IV}(x) = \max_{i,j}(D_i(x) - D_j(x)) = \max_{i,j}(\mathsf{E}\,[\,d_i(x, \boldsymbol{\xi}) - d_j(x, \boldsymbol{\xi})\,]$$

An interpretation: For a given solution x, the $R^{IS}(x)$ measures average maximal day-to-day difference in workload, while $R^{IV}(x)$ measures the maximal difference in average workload between vehicles (drivers). By considering the order of maximization, is possible to see that in general,

$$R^{IV}(x) \leq R^{IS}(x)\ ,$$

and that they are equal only if there are two fixed vehicles i, j that have maximum and minimum distance in *all scenarios* respectively. For driver fairness issues, one may argue that the R^{IV} is the better of these two, since it measures the average workload difference between vehicles (drivers).

Thus for the remaining part of the paper, we adopt the following two objectives to be minimized in a bi-objective sense,

expected total distance (cost) $C(x)$,

expected route balance $R(x)$,

where $R(x) = R^{IV}(x)$ is the inter-vehicle route balance as defined above.

2.4 Multi-objective stochastic optimization

In a multi-objective optimization problem (MOP) several functions are optimized (minimized or maximized) subject to a common set of constraints. Without loss of generality the MOP can be treated as a minimization problem. A general formulation of a bi-objective problem is

$$\begin{aligned} &\min_x \ (f_1(x), f_2(x)) \\ &\text{subject to} \quad g_j(x) \leq b_j, \quad j = 1, \ldots, K \ , \end{aligned}$$

where x is a vector of decision variables. The minimization problem is interpreted in the sense that the solution to the problem is the Pareto set (the non-dominated solutions).

A stochastic multi-objective problem (SMOP) arises when some parameters of the problem are stochastic. A SMOP can be formulated at different levels of generality and abstraction; denoting the random data by a vector $\boldsymbol{\xi}$, and assuming that objectives as well as constraints depend on $\boldsymbol{\xi}$, we could mimic the above, and write a bi-objective problem as follows.

$$\min_x \ (f_1(x, \boldsymbol{\xi}), f_2(x, \boldsymbol{\xi})) \tag{1}$$

$$\text{subject to} \quad g_j(x, \boldsymbol{\xi}) \leq b_j(\boldsymbol{\xi}), \quad j = 1, \ldots, K \ . \tag{2}$$

In this formulation however, we immediately run into further need for interpretation. In general it is no longer clear in (1) what "min" means, as different values of $\boldsymbol{\xi}$ may have different minimizing x. The fundamental concept of a solution x *dominating* another solution y can accordingly be given a number of different meanings. In addition, there will typically not be any x satisfying (2) for all realizations of the random data. Two common ways to deal with this is to either (i) define a *recourse* policy telling how to deal with violations of constraints, or (ii) accept solutions x with a small probability of violating the constraints. Additional technical issues regarding e.g. measurability may call for restrictions on what probability distributions one uses for $\boldsymbol{\xi}$. See Abdelaziz (2012), for further discussion of SMOP in general.

In this paper, we have a SMOP that can be formulated more neatly as

$$\min_x \ (C(x), R(x)) \tag{3}$$

$$\text{subject to} \quad g_j(x, \boldsymbol{\xi}) \leq b_j, \quad j = 1, \ldots, K \tag{4}$$

where (4) are identical to those of the deterministic variant (Oyola and Løkketangen, 2014), with the exception that capacity constraints now depend on random demand, $\boldsymbol{\xi}$. The objectives are the expected cost and the expected inter-vehicle route balance as outlined in section 2.3. The problem is further specified by the use of a recourse action as described in section 2.2, and the expected recourse cost is included in the objectives.

2.4.1 Dominance and the Pareto set

The form of the problem (3),(4) where the objectives do not explicitly depend on $\boldsymbol{\xi}$ allows a rather uncomplicated definition of dominance, i.e. we will say a solution x dominates another solution y, if

$$C(x) \leq C(y) \quad \text{and} \quad R(x) \leq R(y)\ , \quad \text{with at least one strict inequality.} \tag{5}$$

In absence of the strict inequality, we will say that the solution x weakly dominates the solution y (Knowles, 2002). A solution x is *non-dominated* if no other solution dominates x. The Pareto set PS of the problem consists of all non-dominated solutions. The set PS can be considered to represent the "optimal" solution to the SMOP problem. For realistic problem instances, determining PS is computationally infeasible, and by a solution to the SMOP, we will mean a set $\mathrm{PS}_{\mathrm{appr}}$ of VRP solutions x, which is assumed to represent an approximation to PS. When comparing algorithms A and B for solving the SMOP, we write $\mathrm{PS}_{\mathrm{appr}}$(A), $\mathrm{PS}_{\mathrm{appr}}$(B), for the best approximations produced with the two.

2.5 Computational approach

In the proposed model, customer demands are represented by independent random variables, with a particular probability distribution. The objective functions are expected values of rather complicated functions of these variables. We have not found any workable way to obtain exact expressions for the objective function values $C(x), R(x)$ based on parameters of the random data $\boldsymbol{\xi}$. Thus, to evaluate $C(x), R(x)$ *approximately*, we need to use the method of Sample Average Approximation (SAA), where objective function values are approximated by the average values computed on basis of a number n of demand scenarios. Since we then operate with *estimates* of objective values, we need to make sure our solution methods in combination with the SAA produces reliable objective values for the whole set of solutions in a Pareto set approximation $\mathrm{PS}_{\mathrm{appr}}$. The use of the SAA makes it necessary to discuss certain *stability* issues for the SMOP algorithms and the chosen computational framework, as well as performing computational tests to establish the desired stability. This is deferred to section 4.4.

3 An algorithm for solving the CVRPRB with stochastic demands

The greedy randomized adaptive search procedure (GRASP) is an iterative metaheuristic (Resende and Ribeiro, 2003, 2010) consisting of two phases: construction and local search. At every GRASP iteration a solution is built (a repair procedure can be applied in case of infeasibility), once a feasible solution is obtained, a local search is performed, until a local optimum is found. In the single objective approach, the solution is built by including, one at the time, elements to be part of it. At every stage of the construction process, all the candidate attributes that could be included in the solution at the current step are evaluated using a greedy approach. A list, known as the restricted candidate list (RCL), is created with the elements that have a higher performance according to the previous evaluation. The element to be added to the current solution is randomly selected from the RCL (Resende and Ribeiro, 2010). The process is repeated until the solution is completed, updating at every step the evaluation, list of candidates and RCL.

The proposed algorithm (SMGRASP) is an iterated process based on GRASP. The algorithm first finds two initial solutions. Each of them is found when optimizing every objective individually. Given two solutions x and $y \in \mathrm{PS}_{\mathrm{appr}}$, if there are more solutions in $\mathrm{PS}_{\mathrm{appr}}$ representing a tradeoff between x and y, such solutions may share attributes with x and/or y. The level of shared attributes with one or another solution may depend on which objective the tradeoff is favoring. Based on that expectation more solutions are generated starting from the shared elements in the initial solutions. In addition, more solutions are built using SMGRASP, but without considering the shared attributes,

but taking into account the dominance criteria. This is done as a way to introduce more diversity into the search. Once a solution is built, it is then subject to a local search procedure using a modified Or-opt. The building process (construction phase + local search) is iterated for a given number of times (G_{max}). The entire procedure is repeated, until a maximum running time is reached or a maximum number of repetitions(G_{rep}), using the β most different pairs of solutions. The repetition of the process attempts to find better solutions and improve the current $\mathrm{PS}_{\mathrm{appr}}$.

3.1 Construction phase

Solutions are constructed either using a half built solution as initial solution of the construction process or using an empty solution. Two types of solutions are built starting from an empty solution: single objective solutions and bi-objective solutions. Only two single objective solutions are built, one for each objective and at the beginning of the process, these solutions will be called *initial solutions*. The bi-objective solutions, on the other hand, are built considering both objective functions.

In case the solutions become infeasible during the construction phase a penalty (p_{inf}) is multiplied by the excess capacity and added to each of the objective function values. A repair mechanism is applied to infeasible solutions, which through insertion moves tries to make the solution become feasible.

3.1.1 The distant customer priority (DCP) insertion strategy

In a CVRPSD is expected that *route failures* not far from the depot will be less costly than failures occurring far away. A good route may be characterized by the vehicle visiting some customers near the depot in its way to visit customers located far away and then returning to the vicinity of the depot where other customers are visited in the way back to the depot. In case of having a route failure, this feature increases the probability of having it near to the depot.

The SMGRASP heuristic may tend to build the solutions assigning first the customers located near the depot and then moving gradually to those farther away. This may lead sometimes to solutions that do not qualify as good solutions according to the characteristics given previously. A strategy has been implemented to force the algorithm to first allocate in the routes the customers located far away from the depot. The rest of the customers, located in the vicinity of the depot, are expected to be allocated either at the end or at the beginning of the routes.

The strategy applies either if the building process starts from partially built solutions or from an empty solution. First, all customers to be allocated are sorted by distance to depot. If the number of such customers is c_a and $d_p \in (0, 1]$ represent a DCP parameter, the number of customers considered for insertion can be defined as M_p= c_a· d_p. Then a random number m_p between 0 and M_p is generated. The m_p most distant customers are considered by SMGRASP for insertion.

Let be S_p the set of customers considered by SMGRASP for insertion. It initially contains the m_p most distant customers to the depot. SMGRASP selects from S_p a customer i and inserts it into the solution. Customer i is removed from S_p. As long as there are non-inserted customers not in S_p, the most distant of these replaces customer i in S_p. The insertion process continues until S_p is empty.

The DCP insertion strategy is not used every time that SMGRASP builds a new solution. DCP is used by SMGRASP with probability 0.5, when building a solution. The effect of using DCP is discussed in Section 4.6.1.

3.1.2 Construction of initial solutions

The two initial solutions are constructed by SMGRASP using on each case one of the objective functions of the problem. Algorithm 3.1 describes the procedure.

Algorithm 3.1: CONSTRUCTION INITIAL SOLUTIONS($\alpha, f_1, f_2, \ldots, f_m$: $objective\ functions$)

Let α be the algorithm parameter
Let $\mathcal{S}$ be the set of solutions constructed by the algorithm
S$\leftarrow \emptyset$
for $r \leftarrow 0$ **to** m
  do S$\leftarrow$S$\cup$SINGLE OBJECTIVE SGRASP(f_r, α)
return $(\mathcal{S})$

In Algorithm 3.2 the function *initialize(Cl)* finds all possible insertions for the available customers and creates a list of pairs or attributes (i, k) consisting of a customer and a tour for possible insertion. The available customers depend on two aspects, the first is that customer must not be included in the solution yet. The second aspect depend on the DCP strategy explained in Section 4.6.1, since not all customers may be available for insertion, even though they are not part of the solution. The list of all attributes (i, k) is evaluated by function *evaluateInsertions(Cl, f_r)*, for every attribute it computes how much will the objective function f_r increase if customer i is inserted in tour k. In case f_r decreases the value will be negative.

The function *random(a)* will select randomly an attribute (i, k) from the RCL. All attributes in RCL have the same probability of being selected. Once the solution is built, the feasibility is checked and in case of being infeasible a repair mechanism is applied by using function the $repair(s)$. This mechanism uses simple insertion moves looking to reduce/eliminate the infeasibility. A maximum of 100 moves are performed, in case no feasible solution, the solution is discharged. Once all iterations are performed, the best found solution is identified by the function *findBestSol($\mathcal{U}$, f_r)*, which select from $\mathcal{U}$ the solution with the best value for the objective function f_r.

Algorithm 3.2: SINGLE OBJECTIVE SGRASP(f_r, α)

Let $c(a_i)$ be the fitness of attribute a_i
Let Cl be the list of attributes including customers not assigned yet
Let G_{max} be the number of SGRASP iterations
Let U be the set of constructed solutions
$\mathcal{U} \leftarrow \emptyset$
$j \leftarrow 0$
while $j < G_{max}$
do
- $s \leftarrow \emptyset$
- $initialize(Cl)$
- $evaluateInsertions(Cl, f_r)$
- **while** $Cl \neq \emptyset$
  - **do**
    - $c_{min} \leftarrow min\{c(a)|a \in Cl\}$
    - $c_{max} \leftarrow max\{c(a)|a \in Cl\}$
    - $RCL \leftarrow \{a \in Cl|c(a) \leq c_{min} + \alpha(c_{max} - c_{min})\}$
    - $\bar{a} \leftarrow random(a)|a \in RCL$
    - $s \leftarrow s \cup \{\bar{a}\}$
    - $update(Cl)$
    - $evaluateInsertions(Cl, f_r)$
- **if** $infeasible(s)$
  - **then** $\{repair(s)$
- **if** $feasible(s)$
  - **then**
    - $s \leftarrow OrOpt(f_r, s)$
    - $\mathcal{U} \leftarrow \mathcal{U} \cup s$
- $j \leftarrow j + 1$

$s \leftarrow findBestSol(\mathcal{U}, f_r)$
return (s)

3.1.3 Construction of bi-objective solutions

At every iteration of the process, the common elements in the β most different pairs of solutions are used as the partially built solutions. In case the current number of possible pairs is less than β, all pairs are used. A matching procedure (Ho and Gendreau, 2006) is performed to every pair and the number of common elements in every pair is recorded. Solutions with less number of common elements are considered to be more different. New solutions are built starting from partially built solutions, by adding attributes, one at the time. Every candidate A_j represents the insertion of customer i in a tour k. The selection of the attributes to be added to the partial solution requires an evaluation of all attribute candidates. This evaluation is performed using a Pareto rank (Mateo and Alberto, 2012; Zitzler et al., 2001), which takes into consideration the Pareto dominance concept.

For every candidate or possible insertion A_j, its impact on both objective functions is evaluated. Each candidate A_j is compared with all the others to compute how many of them are dominated by A_j. This corresponds to the *strength value*, $S(j)\ \forall j \in \{1, 2, \ldots, T\}$, where T is equal to the actual number of attribute candidates.

The *raw fitness* is computed for every candidate:

$$R(j) = \sum_{t:A_t \prec A_j} S(t) \quad \forall t \in \{1, 2, \ldots, T\} \tag{6}$$

Non-dominated candidates will have a *raw fitness* equal to zero. It may happen that several candidates have the same *raw fitness*. To be able to rank these candidates a *density* $D(j)$ is computed. $D(j)$ is computed as a decreasing function of the Euclidean distance of the candidate A_j to the k-nearest neighbor. The density will always lie in the interval $(0, 1)$ and it is defined as

$$D(j) = 1/(d_k + 2), \tag{7}$$

where d_k is the distance to the k-nearest neighbor.

The performance of each candidate will be measured by the *fitness* $F(j) = R(j) + D(j)$. The RCL is built with the α fraction of the candidate attributes that show a higher performance (lower fitness).

For every pair of solutions selected for SMGRASP a maximum of G_{max} solutions is built. If the initial number of customers to be inserted into the partial built solution is lower than C_{min}, then the number of built solutions will be equal to the factorial of the initial number of customers. Algorithm 3.3 describes how the construction is performed.

Algorithm 3.3: BI-OBJECTIVE SMGRASP(s_1, s_2, α)

Let s_1 and s_2, be two solutions to the problem
Let $c(a_i)$ be the fitness of attribute a_i
Let Cl be the list of attributes including customers not assigned yet
Let G_{max} be the number of SMGRASP iterations
Let U be the set of solutions built from 2 starting solutions
$A_1 \leftarrow \{(i,k)|(i,k) \in s_1\}$
$A_2 \leftarrow \{(i,k)|(i,k) \in s_2\}$
$\mathcal{U} \leftarrow \emptyset$
$j \leftarrow 0$
while $j < G_{max}$
do
- $s \leftarrow A_1 \cap A_2$
- $initialize(Cl)$
- $paretoRanking(Cl)$
- **while** $Cl \neq \emptyset$
  - **do**
    - $c_{min} \leftarrow min\{c(a)|a \in Cl\}$
    - $c_{max} \leftarrow max\{c(a)|a \in Cl\}$
    - $RCL \leftarrow \{a \in Cl|c(a) \leq c_{min} + \alpha(c_{max} - c_{min})\}$
    - $\bar{a} \leftarrow random(a)|a \in RCL$
    - $s \leftarrow s \cup \{\bar{a}\}$
    - $update(Cl)$
    - $paretoRanking(Cl)$
- **if** $infeasible(s)$
  - **then** $\{repair(s)$
- **if** $feasible(s)$
  - **then**
    - $s \leftarrow biObjectiveOrOpt(s)$
    - $\mathcal{U} \leftarrow \mathcal{U} \cup s$
- $j \leftarrow j + 1$

return $(\mathcal{U})$

Other solutions are constructed in a similar way as described in Algorithm 3.3. With the difference that the construction begins with an empty solution and no other solutions are received.

3.2 Local search

The local search phase of SMGRASP is performed by an Or-opt heuristic (Or, 1976). Two types of versions are used: a single objective (traditional version) and a bi-objective Or-opt (new move evaluation). In both cases the structure of the algorithm is the same, the difference is in the evaluation of the solutions. The single objective Or-opt uses a cost function to evaluate the quality of a move, measuring the impact in the specific objective function. The bi-objective Or-opt evaluates the solutions using the Pareto dominance criteria as described in Section 3.1.3. This approach of the Or-opt, where more than one objective function is evaluated simultaneously, to the best of our knowledge has not been previously used.

3.3 The "virtual" capacity mechanism

In CVRPSD problems it is common to employ constraints stating that the maximal allowed *expected demand* on any planned route is identical to the vehicle capacity Q. Obviously, the actual *realized demand* on a route may still exceed Q, resulting in a recourse action on the route. Let $D_{\max}$ denote the maximum allowed expected demand to be planned on any route. We call $D_{\max}$ the *virtual capacity*. Clearly, using exactly $D_{\max} = Q$ is not the only possible choice. If $D_{\max} < Q$, it means the planned routes will have some slack to absorb higher-than-expected demand, leading to less use of recourse actions. If on the other hand, $D_{\max} > Q$ a wider choice of routes are allowed to be planned, where some of these may have very high risk of needing recourse actions. Considering the SMOP of this paper, $D_{\max} < Q$ represents a restriction of the search space, whereas using $D_{\max} > Q$ represents a relaxation of the problem, with a larger search space.

Considering the fact that our problem involves two objectives that are in different ways affected by recourse costs, we wanted to see whether using $D_{\max} \neq Q$ could lead to better solutions. For instance, we see that it would typically be beneficial for the route balance objective, if the routes that are relatively short without recourse, have higher probability for failure and recourse.

One should keep in mind the following. We know that the theoretical optimum of a relaxed problem will be at least as good as that of a restricted problem. On the other hand, for the SMOP we look at here, we need to resort to approximate solutions, and given limited search time, a restriction of the search space can improve on the best approximations found.

We then define the "virtual" capacity mechanism (VCM) as taking $D_{\max} \neq Q$ into the problem formulation, allowing a tighter or wider problem to be considered. Of course, the actual capacity on vehicles Q remains the same.

Because we are comparing the SMGRASP method to another method (TAN) which do not utilize the VCM, we run most of the computational experiments without it. Then certain experiments are done comparing the versions of SMGRASP with VCM using different levels of $D_{\max}$.

4 Computational experiments and results

The proposed algorithm was implemented in C++. All tests were conducted on a computer with processor Intel (R) Xeon (R) CPU E31270 @ 3.40 GHz and 16.0 GB of RAM.

4.1 Test environment

There are no standard instances for the CVRPSD in the literature. Some authors construct their own instances, but these are not given, just a description of how were they built is provided. Some others are adapted well-known determinist instances to CVRPSD, as in Dror and Trudeau (1986); Tan et al. (2007); Juan et al. (2011).

The set of instances previously used for the deterministic version of the problem in Oyola and Løkketangen (2014) is used here. The customer demand (d_i) in the deterministic problem is used as the expected demand in the stochastic version. Two different levels of variance (50% and 75% of the expected demand) were associated to every instance. In this sense each instance turns into two, where the only difference between the two of them is the variance associated to each customer i. For each customer i the variance of the demand is equal to $v_l \cdot d_i$, where v_l corresponds to the given level of variance. This means: if $v_l = 0.5$, we take $p_i = 1/2$ and $n_i = 2 \cdot d_i$. If $v_l = 0.75$, we take $p_i = 1/4$ and $n_i = 4 \cdot d_i$, where p_i, n_i are parameters of the binomial demand distribution for customer i. Instances will be identified adding the corresponding variance level at the end of it, e.g. E016-03m-50 corresponds to the original instance E016-03m where the variance level is set to 50%.

4.2 Evaluation criteria

Measuring the quality of the set of (potentially) Pareto optimal solutions to a multi-objective problem is not straightforward. Several performance metrics have been proposed in the literature. However, no single metric is able to fairly compare two sets of solutions, since all of them present some drawbacks. Consequently more than one metric is required to make a better comparison. Some metrics are: the $\mathcal{S}$ metric , $\mathcal{C}$ metric (Jozefowiez et al., 2009, 2007b; Knowles, 2002; Mateo and Alberto, 2012), the ratio of non-dominated individuals in a set $\mathcal{X}$, $RNI(\mathcal{X})$ (Tan et al., 2006), the total number of solutions (when dealing with exact methods) (Visée et al., 1998), the generational distance (Jozefowiez et al., 2007a; Knowles, 2002; Mateo and Alberto, 2012), the D_2 metric (Ulungu et al., 1999) and hyperarea metric (Collette and Siarry, 2003).

Two of these metrics have been used to evaluate algorithms that attempt to solve the CVRPRB (Jozefowiez et al., 2009, 2007b; Oyola and Løkketangen, 2014), $\mathcal{C}$ metric and $\mathcal{S}$ metric. For that reason they seem to be a good choice to evaluate the performance of the proposed algorithm. The $\mathcal{S}$ metric measures the area in objective space dominated by a set of solutions, given a reference point. A set $\mathcal{R}$ has a better performance than a set $\mathcal{X}$, if the $\mathcal{S}$ metric of the former one is greater.

Given two sets of solutions $(\mathcal{R}, \mathcal{X})$, the $\mathcal{C}$ metric (Jozefowiez et al., 2009, 2007b; Knowles, 2002; Mateo and Alberto, 2012) measures the ratio of solutions in $\mathcal{X}$ weakly dominated by solutions in $\mathcal{R}$. The metric is not symmetric, so both $\mathcal{C}(\mathcal{R}, \mathcal{X})$ and $\mathcal{C}(\mathcal{X}, \mathcal{R})$ should be computed. This metric is not reliable if the cardinality of the sets is different. In addition it is not able to measure by how much a set outperforms another one (Knowles, 2002). A set $\mathcal{R}$ has a better performance than a set $\mathcal{X}$, if $\mathcal{C}(\mathcal{R}, \mathcal{X})$ is closer to 1 and $\mathcal{C}(\mathcal{X}, \mathcal{R})$ is closer to 0.

4.3 Example of search progress

As an illustration of how the SMGRASP solutions in objective space can evolve over the SMGRASP repetitions, some figures based on the instance E022-06m-50 were included. The figures illustrate the evolution over 20 repetitions, in a single run on the instance. Figure 1 shows how the value of the $\mathcal{S}$ metric changed during the different repetitions of SMGRASP. Figure 2, shows the Pareto set approximations in certain stages of the SMGRASP process. The actual test run here is one of the 10 runs used to obtain average results in table 8, but using a different reference point for

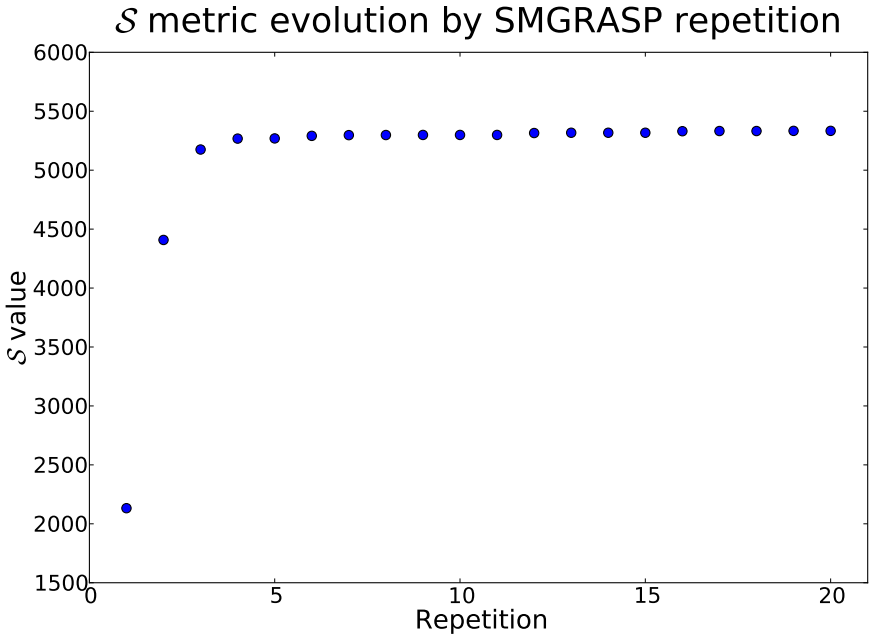

Figure 1: Value of the $\mathcal{S}$ metric at the different repetitions of SMGRASP for the instance E022-06m-50

computing the $\mathcal{S}$ metric. In producing data for figure 1, the reference point was (784.6, 42.2) corresponding to the worst value on distance and balance in any of the front approximations. In figure 1 we see what is a typical behavior, where initial poor results are rapidly improved. After approximately 5 repetitions, further improvements come in small portions at certain repetitions. All repetitions up to the 9th improve on the front, while from repetition 10 to 20 there are 6 repetitions with no improvement in the front. Figure 2 illustrates how Pareto front approximations change in the objective space. For selected repetitions we plot the front approximation, indicating which solutions are new from the previously shown front. At repetition one, only four solutions are in the front. After repetition 3, a number of improvements are found, and only one of the initial solutions remain non-dominated. By repetition 5, about ten additional solutions are introduced, dominating a fair part of the previous approximation. Finally, by repetition 20 further non-dominated solutions are found, again replacing some of the previous solutions. The complete final Pareto set approximation $\mathrm{PS}_{\mathrm{appr}}$ at repetition 20 is also shown, and we see that solutions found at various stages (even one initial solution) are present in the final approximation.

4.4 Stability

An implementation of the SMGRASP algorithm needs to use sample average approximations of the objective functions, so we make a finite set Ξ of scenarios by random sampling from the given demand distributions. The number of scenarios must be large enough to approximate the objective functions with satisfactory precision. This should be tested comparing different scenario sets Ξ_k, using some stability tests. Two types of stability tests are performed, in-sample and out-of-sample stability (Kaut and Wallace, 2007). It is said that there is in-sample stability when the objective values of the solution set obtained using one scenario set is the same, or at least similar, to the objective values of the set obtained using a different scenario set. The out-of-sample stability, on the other hand, indicates that the solution set objective function values are the same, or at least similar, when the solutions are *evaluated* in scenario sets different than the one used to find them. These tests are done using different sizes of the scenario sets: 5, 10, 20, 40 and 80.

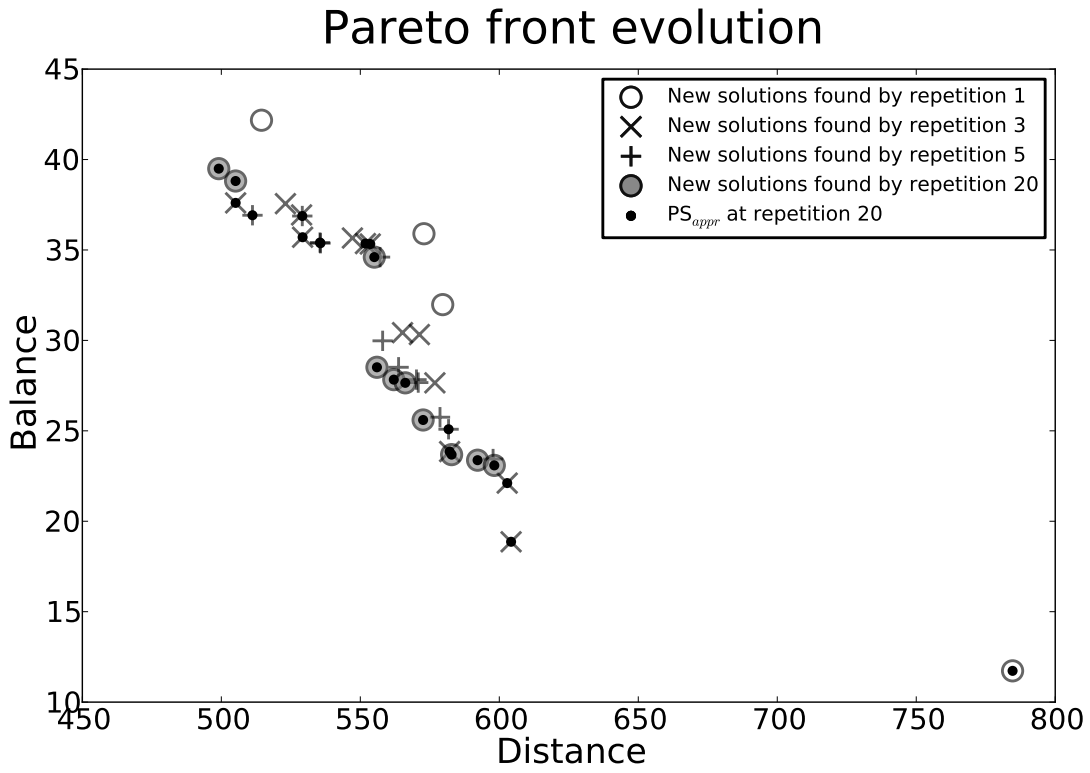

Figure 2: Changes in the $\mathrm{PS}_{\mathrm{appr}}$ *at different repetitions of SMGRASP for the instance E022-06m-50*

The tests were performed using six different instances: E016-03m, E022-06m, E030-04s, E051-05e, E076-07s and E101-10c. Each instance was associated with the two different levels of variance mentioned before (50% and 75%)

It may be difficult to directly measure the differences or similarities in the objective function values of two solutions sets. This creates a problem when testing for in-sample stability. The $\mathcal{S}$ metric is one of the criteria used to compare different algorithms, so it is a reasonable choice to measure stability in that metric. Using 20 different sets of scenarios, an approximation to the Pareto set is found, for each set. The $\mathcal{S}$ metric is computed for every set of solutions, evaluated in the respective scenario set that was used to find the solutions.

The out of sample stability test is done using the $\mathcal{S}$ metric. An additional good choice is given by the values of the objective functions, since the same solution can be evaluated using different set of scenarios, and the values can be compared. In this case, the value of the objective functions and the $\mathcal{S}$ metric are evaluated in scenario sets different than the one used to find the solutions. For every solution in the approximation of the Pareto set, the objective functions are evaluated in the scenario set used to find them and also in the 19 other scenario sets. The average and the standard deviation of each objective function is computed over these 19 scenarios and compared with the first value (in-sample value). The $\mathcal{S}$ metric is computed and compared in similar way.

4.4.1 Stability test results

Table 2 in the Appendix shows a summary of the in-sample test. For each given size, 20 different scenario sets were used. For each solution set (X_k), obtained using scenario set Ξ_k, $\mathcal{S}(X_k)$ is computed. The second column in Table 2 shows the average of $\mathcal{S}(X_k)$ over the 20 different scenario sets. The third column shows the standard deviation of the metric and the last column shows the Coefficient of Variation (CV) of the metric. In general, this value gets smaller when the size of the scenario set increases. A time limit of 3 hours was given to each experiment.

In some cases the stability may not be improved when the number of scenarios increases. This is likely due to fact that there is a time limit for the search. Increasing the number of scenarios may reduce the explored area of the search space.

Table 3 in the Appendix shows a summary of the test for out-of-sample stability. Recall here that a solution set X_k, is a set of VRP solutions,

$$X_k = \{x_k^1, x_k^2, \ldots, x_k^{m_k}\} ,$$

obtained with a particular scenario set Ξ_k. For each solution set X_k, the objective function values, $C(x_i^k)$ and $R^{IV}(x_i^k)$, are evaluated using *different* scenario sets Ξ_j. For each solution, the difference between the objective function value when evaluated in Ξ_k and the average value when evaluated in the other scenario sets Ξ_j is computed. Such difference is computed as a percentage of the objective function value when evaluated in Ξ_k. The third column in Table 3 shows the average value of such difference for $C(x)$ and the corresponding value for $R^{IV}(x)$ is found in the fourth column. It was considered important to test how different are the objective function values, when evaluated using different scenarios. The fifth and sixth column in Table 3 show the average CV for $C(x)$ and $R^{IV}(x)$ respectively when evaluated in the scenario sets Ξ_j. It is possible to see that for the instances E016-03m-50 and E016-03m-75 $R^{IV}(x)$ does not look very stable, however it is important to take into consideration that the values of $R^{IV}(x)$ found in the $\mathrm{PS}_{\mathrm{appr}}$ correspond to small values compared to the values of $C(x)$. $R^{IV}(x)$ values can range from 0.03% to 6% the best value of $C(x)$, so small variations can represent a important percentage of $R^{IV}(x)$, but not when compared to $C(x)$.

Table 4 in the Appendix shows a summary of the test for out-of-sample stability for the TAN algorithm (Tan et al., 2007), since this was not explicitly reported. No time limit is given and the parameters are the same reported by the authors.

It was decided to use 40 scenarios in the experiments. All results presented hereafter were obtained from experiments using such scenario set size, unless stated otherwise.

4.5 Parameter tuning

Parameters for SMGRASP were set after some preliminary testing. The parameter values that lead in average to better results were selected:

- Alpha parameter for the SMGRASP heuristic: 0.1
- Number of SMGRASP iterations (G_{max}): 150
- Percentage of most distant customers used by DCP insertion strategy (d_p): 25%
- Or-opt parameter: 2
- Number of SMGRASP repetitions (G_{rep}): 20
- Penalty for infeasibility (p_{inf}): 1
- Number of pair of solutions (β) used at every repetition of SMGRASP: 10

Given the running time of TAN algorithm, the running time was set initially to be 3 hours or 20 repetitions of SMGRASP, allowing the smaller instances to finish earlier.

4.6 Results

Results comparing the SMGRASP heuristic and the implementation of an evolutionary algorithm (TAN) for the CVRPSD (Tan et al., 2007) using the $\mathcal{S}$ metric and the $\mathcal{C}$ metric are presented in the Appendix in Tables 5 and 6, respectively. The average, minimum and maximum values were computed from the results obtained in the ten different runs, as well as the reference used in the computation of the $\mathcal{S}$ metric. The numbers are rounded to the nearest integer in Table 5, the same applies for the rest of the tables presenting results for the $\mathcal{S}$ metric.

According to the average values of the $\mathcal{S}$ metric, the SMGRASP heuristic is able to find a better approximation to the Pareto set in 38 out of 40 instances. If the $\mathcal{C}$ metric is used for the comparison, SMGRASP is able to find, in average, better approximations in 36 out of 40 instances. When the two metrics are used simultaneously, SMGRASP finds in average a better approximation to the Pareto set in 36 out of 40 instances, TAN is able to find a better approximation in two instances. The metrics give contradictory results in two instances (E016-03m-75 and E016-05m-50). Another possibility for comparing the two heuristics is using the maximum values of the two metrics. In the latter case, TAN is able to find a better approximation in 2 out of 40 instances, SMGRASP on the other hand, is able to do it in 34 instances. There are 4 instances where is not possible to draw a conclusion (E016-03m-75, E016-05m-75, E021-06m-75 and E022-06m-50).

It was found that given the initial parameters SMGRASP runs for longer time than TAN, as it can be seen in Table 7 in the Appendix. To make a more fair comparison, new experiments were done for SMGRASP setting the time limit close to the running time used by TAN algorithm. A summary of the results are presented in Table 1, where the second column shows the difference between the average values of the $\mathcal{S}$ metric ($\mathcal{S}$(SMGRASP) - $\mathcal{S}$(TAN)). The third column contains the the difference of the average values of the $\mathcal{S}$ metric ($\mathcal{C}$(SMGRASP, TAN) - $\mathcal{C}$(TAN, SMGRASP)). In both cases positive values shows a better performance of SMGRASP.

Detailed results of these tests are shown in Tables 8 and 9 in the Appendix. According to the $\mathcal{S}$ metric, in average SMGRASP finds better results in 38 out of 40 instances. This number goes down to 35 when the comparison is done using the $\mathcal{C}$ metric. When both metrics are used simultaneously, in average, SMGRASP finds a better approximation of the Pareto set in 34 out of 40 instances and TAN finds a better result in two instances. In four cases is not possible to draw a conclusion, since the two metrics are contradictory (E016-03m-75 and E016-05m-50, E023-05s-50 and E023-05s-75). Using the maximum values of the metrics for comparison, TAN finds better results in two instances, SMGRASP does it in 33 instances, and no conclusion is obtained in 5 out of 40 instances (E016-03m-75, E016-05m-75, E021-06m-50, E022-06m-50 and E023-05s-50).

Table 1: Summary of comparison between TAN and SMGRASP average results

Instance	Av. $\mathcal{S}$(SMGRASP) - $\mathcal{S}$(Tan)	Av. $\mathcal{C}$(SMGRASP, TAN) - $\mathcal{C}$(TAN, SMGRASP)	Av. TAN running time	Av. SMGRASP running time
E016-03m-50	-54	-0,33	1 239	1 230
E016-03m-75	38	-0,20	1 238	1 230
E016-05m-50	94	-0,02	1 172	1 160
E016-05m-75	-235	-0,29	1 143	1 124
E021-04m-50	1 977	0,52	938	930
E021-04m-75	1 730	0,59	826	820
E021-06m-50	1 293	0,36	773	758
E021-06m-75	1 718	0,63	770	760
E022-06m-50	156	0,27	840	830
E022-06m-75	1	0,26	839	830
E023-05s-50	6 343	-0,06	1 162	1 160

Table 1 Continued: Summary of comparison between TAN and SMGRASP average results

Instance	Av. $\mathcal{S}$(SMGRASP) - $\mathcal{S}$(TAN)	Av. $\mathcal{C}$(SMGRASP, TAN) - $\mathcal{C}$(TAN, SMGRASP)	Av. TAN running time	Av. SMGRASP running time
E023-05s-75	5 660	0,00	1 162	1 160
E026-08m-50	5 789	0,89	766	760
E026-08m-75	5 311	0,66	754	750
E030-04S-50	1 910	0,46	2 359	2 350
E030-04S-75	1 715	0,55	2 351	2 350
E031-09h-50	6 700	0,81	1 735	1 730
E031-09h-75	6 348	0,93	1 728	1 720
E033-05s-50	1 256	0,59	2 729	2 720
E033-05s-75	484	0,60	2 534	2 530
E036-11h-50	4 617	0,53	2 163	2 160
E036-11h-75	6 705	0,69	2 086	2 081
E041-14h-50	5 166	0,49	2 430	2 430
E041-14h-75	5 328	0,56	2 450	2 440
E048-04y-50	73 600 000	0,92	3 673	3 671
E048-04y-75	60 348 000	0,91	2 915	2 910
E051-05e-50	20 720	0,89	3 386	3 381
E051-05e-75	20 985	0,91	2 743	2 740
E076-07s-50	40 310	0,96	4 850	4 844
E076-07s-75	40 379	0,86	3 535	3 538
E076-08s-50	61 898	0,83	4 838	4 832
E076-08s-75	63 220	0,88	3 526	3 538
E076-14u-50	87 263	0,65	3 454	3 461
E076-14u-75	78 352	0,61	2 879	2 874
E101-08e-50	79 976	0,91	5 806	5 807
E101-08e-75	96 386	0,70	3 928	3 929
E101-10c-50	128 054	0,99	5 916	5 916
E101-10c-75	118 132	0,95	4 051	4 051
E101-14s-50	140 817	0,83	6 169	6 179
E101-14s-75	115 215	0,75	4 118	4 154

4.6.1 The DCP insertion strategy

Results obtained by SMGRASP were tested against a similar algorithm without the distance insertion strategy (SMGRASP-NDI). The SMGRASP finds in average better results in 31 out of 40 instances. In four instances is not possible to conclude which one finds a better approximation in average (E022-06m-75, E026-08m-75, E031-09h-50 and E076-14u-50). The algorithm without the procedure, on the other hand, finds in average better results in five instances. A comparison can be found in tables 10 and 11 in the Appendix. Using the maximum values of the metrics to compare the two versions of SMGRASP, we get different results. SMGRASP is able to perform better in 17 instances, SMGRASP-NDI find better results just in two of them and there are 21 out of 40 instances where it is not possible to draw any conclusion. Based on these comparisons, the usage of the DCP insertion strategy seems to improve the quality of the solutions obtained by SMGRASP.

4.6.2 The Virtual Capacity Mechanism

Preliminary tests were done allowing a virtual capacity $D_{\max}$ equal to a factor times the real capacity Q, e.g. $D_{\max} = c_f \cdot Q$. The following values for c_f were considered: 0.9, 0.95, 1.00, 1.05, 1.10, 1.20, 1.50, 2.0, 3.0 and 5.0 . These tests indicated that that better results can be obtained with c_f equal to 1.05. Initial results of SMGRASP as presented in tables 5 and 6 were compared against the results obtained by using the virtual capacity (VCM). The algorithm allowing the VCM finds in average better results in 27 out of 40 instances. In two of the instances is not possible to conclude which one finds in average a better approximation (E030-04s-50 and E048-04y-75). A comparison can be found in tables 12 and 13 in the Appendix. Comparing the maximum values of the metrics, is possible to see that when SMGRASP allows VCM, better results are obtained in 19 instances. If VCM is not allowed, SMGRASP finds better results in 8 instances. In 13 out of 40 instances is not possible to determine which alternative performs better.

5 Conclusions

A stochastic extension of a well known deterministic multi-objective problem was presented. A GRASP based heuristic was proposed to find an approximation to the Pareto set of the problem. Obtained results were compared against an evolutionary algorithm used to deal with a similar problem. Results indicate a high quality of the solutions obtained by the proposed algorithm (SMGRASP).

Both in-sample and out-of-sample stability tests were conducted. Results suggesting stability of the algorithm were obtained and reported.

A distance-based insertion strategy was included in SMGRASP and tests show evidence of its efficiency.

The option to plan routes with a expected cumulative demand above the vehicle capacity was tested. Test results indicate that this can be a useful mechanism that may be important for the DM to take into consideration when planning the routes.

As a possible direction for further research, parallel algorithms could be included as a way to speed up the process and/or explore more areas of the search space. Another possibility is to increase the number of objectives in the problem, this could make it even more realistic.

References

Abdelaziz, F. B. (2012). Solution approaches for the multiobjective stochastic programming. *European Journal of Operational Research*, 216(1):1–16.

Ahmadi-Javid, A. and Seddighi, A. H. (2013). A location-routing problem with disruption risk. *Transportation Research Part E: Logistics and Transportation Review*, 53:63 – 82.

Ak, A. and Erera, A. L. (2007). A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation Science*, 41(2):222–237.

Bianchi, L., Birattari, M., Chiarandini, M., Manfrin, M., Mastrolilli, M., Paquete, L., Rossi-Doria, O., and Schiavinotto, T. (2005). Hybryd metaheuristics for the vehicle routing problem with stochastic demands. Technical report, Dalle Molle Institute for Artificial Intelligence, Manno, Switzerland.

Chepuri, K. and Homem-de Mello, T. (2005). Solving the vehicle routing problem with stochastic demands using the cross-entropy method. *Annals of Operations Research*, 134(1):153–181.

Christiansen, C. H. and Lysgaard, J. (2007). A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, 35(6):773 – 781.

Collette, Y. and Siarry, P. (2003). *Multiobjective optimization: principles and case studies*. Springer, Berlin.

Dror, M. and Trudeau, P. (1986). Stochastic vehicle routing with modified savings algorithm. *European Journal of Operational Research*, 23(2):228–235.

Erera, A. L., Savelsbergh, M., and Uyar, E. (2009). Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints. *Networks*, 54(4):270–283.

Gauvin, C., Desaulniers, G., and Gendreau, M. (2014). A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 50(0):141–153.

Gendreau, M., Jabali, O., and Rei, W. (2014). Stochastic vehicle routing problems. In Toth, P. and Vigo, D., editors, *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 213–239. Society for Industrial and Applied Mathematics.

Gendreau, M., Laporte, G., and Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12.

Haugland, D., Ho, S. C., and Laporte, G. (2007). Designing delivery districts for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 180(3):997–1010.

Hjorring, C. and Holt, J. (1999). New optimality cuts for a single-vehicle stochastic routing problem. *Annals of Operations Research*, 86:569–584.

Ho, S. C. and Gendreau, M. (Mar 2006). Path relinking for the vehicle routing problem. *Journal of Heuristics*, 12(1-2):55–72.

Jabali, O., Rei, W., Gendreau, M., and Laporte, G. (2014). Partial-route inequalities for the multi-vehicle routing problem with stochastic demands. *Discrete Applied Mathematics*, 177:121–136.

Jozefowiez, N., Semet, F., and El-Ghazali, T. (2007a). The bi-objective covering tour problem. *Computers & Operations Research*, 34(7):1929–1929.

Jozefowiez, N., Semet, F., and Talbi, E. (Jun 16, 2009). An evolutionary algorithm for the vehicle routing problem with route balancing. *European Journal of Operational Research*, 195(3):761–769.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2002). *Parallel Problem Solving from Nature VII, Lecture Notes in Computer Science*, volume 2439, chapter Parallel and hybrid models for multi-objective optimization: Application to the vehicle routing problem, pages 271–280. Springer-Verlag.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (Oct 2007b). Target aiming pareto search and its application to the vehicle routing problem with route balancing. *Journal of Heuristics*, 13(5):455–469.

Juan, A., Faulin, J., Grasman, S., Riera, D., Marull, J., and Mendez, C. (2011). Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands. *Transportation Research Part C: Emerging Technologies*, 19(5):751–765. Freight Transportation and Logistics (selected papers from {ODYSSEUS} 2009 - the 4th International Workshop on Freight Transportation and Logistics).

Kaut, M. and Wallace, S. W. (2007). Evaluation of scenario-generation methods for stochastic programming. *Pacific Journal of Optimization*, 3(2):257–271.

Knowles, J. D. (2002). *Local-search and hybrid evolutionary algorithms for Pareto optimization*. PhD thesis, University of Reading.

Laporte, G., Louveaux, F. V., and Hamme, L. V. (2002). An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423.

Lee, T.-R. and Ueng, J.-H. (1999). A study of vehicle routing problems with load-balancing. *International Journal of Physical Distribution & Logistics Management*, 29(10):646 – 657.

Mateo, P. and Alberto, I. (2012). A mutation operator based on a pareto ranking for multi-objective evolutionary algorithms. *Journal of Heuristics*, 18:53–89.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2011). Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45(3):346–363.

Or, I. (1976). *Traveling Salesman-type Combinatorial Problems and Their Relation to the Logistics of Regional Blood Banking*. PhD thesis, Northwestern University.

Oyola, J. and Løkketangen, A. (2014). GRASP-ASP: An algorithm for the CVRP with route balancing. *Journal of Heuristics*, 20(4):361–382.

Rei, W., Gendreau, M., and Soriano, P. (2007). Local branching cuts for the 0-1 integer L-shaped algorithm. *Technical report, CIRRELT-2007-23, Centre interuniversitaire de recherche sur les réseaux d'enterprise, la logistique et le transport*, 44(1):136–146.

Rei, W., Gendreau, M., and Soriano, P. (2010). A hybrid monte carlo local branching algorithm for the single vehicle routing problem with stochastic demands. *Transportation Science*, 44(1):136–146.

Resende, M. and Ribeiro, C. (2003). *Handbook of Metaheuristics*, chapter Greedy randomized adaptive search procedures, pages 219–250. Kluwer Academic Publishers.

Resende, M. and Ribeiro, C. (2010). Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau, J.-Y. P., editor, *Handbook of Metaheuristics*, pages 283–319. Springer.

Reynolds, A. P., Corne, D. W., and de la Iglesia, B. (2009). A multiobjective grasp for rule selection. In *Proceedings of the 11th Annual conference on genetic and evolutionary computation (GECCO), Montreal, Canada*, pages 643–650.

Russell, R. A. and Urban, T. L. (2008). Vehicle routing with soft time windows and erlang travel times. *The Journal of the Operational Research Society*, 59(9):1220–1228.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., and Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171.

Secomandi, N. (2000). Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27(1112):1201–1225.

Secomandi, N. (2001). A rollout policy for the vehicle routing problem with stochastic demands. *Operations Research*, 49(5):796–802.

Secomandi, N. (2003). Analysis of a rollout approach to sequencing problems with stochastic routing applications. *Journal of Heuristics*, 9(4):321–352.

Secomandi, N. and Margot, F. (2009). Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230,.

Tan, K., Cheong, C., and Goh, C. (2007). Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813–839.

Tan, K., Chew, Y., and Lee, L. (2006). A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172(3):855 – 885.

Toth, P. and Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1-3):487 – 512.

Ulungu, E. L., Teghem, J., Fortemps, P. H., and Tuyttens, D. (1999). Mosa method: a tool for solving multiobjective combinatorial optimization problems. *Journal of Multicriteria Decision Analysis*, 8(4):221–221.

Vianna, D. and Arroyo, J. (2004). A grasp algorithm for the multi-objective knapsack problem. In *Proceedings of the 24th International Conference of the Chilean Computer Science Society*.

Visée, M., Teghem, J., Pirlot, M., and Ulungu, E. (Mar 1998). Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem. *Journal of Global Optimization*, 12(2):139–155.

Yang, W.-H., Mathur, K., and Ballou, R. H. (2000). Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(1):99–112.

Zhu, L., Rousseau, L.-M., Rei, W., and Li, B. (2014). Paired cooperative reoptimization strategy for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 50(0):1–13.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm. Technical report, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institue of technology (ETH).

Appendix

Table 2: In-sample test summary

Instance	Size of scenario set	Av. $\mathcal{S}$ metric	Stand. dev. of $\mathcal{S}$ metric	CV (%)
E016-03m-50	5	6538.83	418.88	6.41
	10	7475.95	323.22	4.32
	20	7719.41	236.14	3.06
	40	6755.70	132.50	1.96
	80	5800.40	101.96	1.76
E016-03m-75	5	7595.19	463.50	6.10
	10	9038.40	381.47	4.22
	20	4807.32	202.75	4.22
	40	7360.01	169.97	2.31
	80	7006.95	103.12	1.47
E022-06m-50	5	13567.70	808.31	5.96
	10	9017.68	771.95	8.56
	20	10190.5	743.29	7.29
	40	10554.8	776.32	7.36
	80	9756.95	734.02	7.52
E022-06m-75	5	7914.99	612.12	7.73
	10	10284.30	667.93	6.49
	20	9905.27	771.67	7.79
	40	11486.60	853.46	7.43
	80	10193.20	559.34	5.49
E030-04s-50	5	76657.10	561.25	0.73
	10	85064.20	549.75	0.65
	20	80688.30	570.79	0.71
	40	79503.30	596.65	0.75
	80	83753.30	730.87	0.87
E030-04s-75	5	82221.7	696.13	0.85
	10	84142.00	870.20	1.03
	20	81021.60	585.59	0.72
	40	80794.20	691.68	0.86
	80	82051.70	661.43	0.81
E051-05e-50	5	25507.50	757.13	2.97
	10	15014.30	434.53	2.89
	20	18155.40	855.28	4.71
	40	17085.60	788.77	4.62
	80	20223.50	1005.79	4.97
E051-05e-75	5	17950.2	637.74	3.55
	10	16168.00	738.88	4.57
	20	20785.20	617.11	2.97
	40	18713.90	866.30	4.63
	80	19859.40	936.46	4.72
E076-07s-50	5	76097.20	1821.63	2.39
	10	83220.20	2072.25	2.49
	20	97889.40	3045.32	3.11
	40	81164.90	2217.39	2.73
	80	104932.00	4197.75	4.00

Table 2 Continued: In-sample test summary

Instance	Size of scenario set	Av. $\mathcal{S}$ metric	Stand. dev. of $\mathcal{S}$ metric	CV (%)
E076-07s-75	5	85973.60	2092.06	2.43
	10	85174.20	3093.32	3.63
	20	107409.00	2871.58	2.67
	40	89477.80	2885.94	3.23
	80	95511.10	5139.37	5.38
E101-10c-50	5	159530.00	4469.54	2.80
	10	162766.00	5209.47	3.20
	20	222626.00	8136.41	3.65
	40	158531.00	9334.68	5.89
	80	174615	11309.9	6.48
E101-10c-75	5	171157.00	7823.23	4.57
	10	164731.00	7604.69	4.62
	20	132795.00	6480.17	4.88
	40	156901.00	12288.00	7.83
	80	173274.00	14757.30	8.52

Table 3: Out-of-sample test summary (objective function values)

Instance	Size of scenario set	Difference in $C(x)$ (%)	Difference in $R^{IV}(x)$ (%)	Average CV $C(x)$ (%)	Average CV $R^{IV}(x)$ (%)
E016-03m-50	5	5.38	67.72	6.19	26.91
	10	4.50	51.04	5.21	25.65
	20	3.74	38.53	4.76	22.91
	40	3.26	26.25	3.80	20.48
	80	2.20	17.52	2.61	20.25
E016-03m-75	5	4.95	62.67	6.07	28.96
	10	4.56	30.03	5.85	25.66
	20	3.25	26.81	4.03	25.40
	40	3.37	26.28	3.86	21.44
	80	2.11	19.20	2.71	18.96
E022-06m-50	5	2.71	6.76	4.01	10.77
	10	2.27	7.04	3.30	8.92
	20	1.51	4.29	2.22	6.54
	40	0.83	3.32	1.16	4.12
	80	0.95	2.82	1.21	3.62
E022-06m-75	5	3.44	12.23	4.82	19.61
	10	2.80	8.44	3.93	11.80
	20	2.02	7.83	2.86	9.61
	40	1.52	5.05	2.01	6.22
	80	0.95	3.77	1.33	4.70
E030-04s-50	5	0.32	1.23	0.50	1.62
	10	0.34	2.50	0.47	1.75
	20	0.15	1.65	0.28	1.28
	40	0.14	0.94	0.20	0.97
	80	0.22	1.49	0.27	0.80

Table 3 Continued: Out-of-sample test summary (objective function values)

Instance	Size of scenario set	Difference in $C(x)$ (%)	Difference in $R^{IV}(x)$ (%)	Average CV $C(x)$ (%)	Average CV $R^{IV}(x)$ (%)
E030-04s-75	5	0.60	1.59	0.72	2.14
	10	0.23	1.17	0.34	1.22
	20	0.21	3.05	0.36	1.41
	40	0.23	1.56	0.32	1.24
	80	0.13	1.10	0.19	0.98
E051-05e-50	5	6.20	32.70	9.89	23.54
	10	4.06	17.95	6.25	21.23
	20	4.87	13.29	6.69	16.52
	40	4.43	12.21	6.74	13.60
	80	2.78	14.01	3.71	12.06
E051-05e-75	5	5.59	24.77	8.12	22.59
	10	5.44	22.49	7.76	19.88
	20	5.15	15.64	7.43	16.25
	40	4.40	13.15	6.12	14.38
	80	3.55	10.16	4.90	11.47
E076-07s-50	5	7.06	12.81	11.91	12.11
	10	6.59	11.56	7.73	10.96
	20	5.34	7.89	6.54	8.70
	40	2.86	5.14	3.71	6.65
	80	2.01	4.13	2.47	5.38
E076-07s-75	5	8.68	14.65	12.47	14.47
	10	7.48	16.02	11.73	12.61
	20	5.98	8.25	7.99	9.66
	40	3.56	6.08	5.01	7.46
	80	2.53	7.78	3.60	6.03
E101-10c-50	5	9.57	12.10	15.30	12.16
	10	8.76	10.09	11.70	9.91
	20	5.86	8.46	7.75	8.35
	40	3.72	5.96	4.41	7.02
	80	1.98	3.74	2.38	4.84
E101-10c-75	5	11.88	13.07	17.50	12.19
	10	9.45	9.03	13.18	9.71
	20	6.58	7.65	8.77	8.56
	40	2.93	5.43	3.41	6.81
	80	0.94	4.13	1.76	5.56

Table 4: Out-of-sample test summary (objective function values for Tan's algorithm)

Instance	Size of scenario set	Difference in $C(x)$ (%)	Difference in $R^{IV}(x)$ (%)	Average CV $C(x)$ (%)	Average CV $R^{IV}(x)$ (%)
E016-03m-50	5	3.41	31.92	4.43	28.91
	10	3.39	26.29	3.80	32.57
	20	2.74	23.81	3.26	26.75
	40	2.63	22.99	2.92	25.27
	80	2.77	17.98	3.11	19.63

Table 4 Continued: Out-of-sample test summary (objective function values for Tan's algorithm)

Instance	Size of scenario set	Difference in $C(x)$ (%)	Difference in $R^{IV}(x)$ (%)	Average CV $C(x)$ (%)	Average CV $R^{IV}(x)$ (%)
E016-03m-75	5	3.44	35.74	4.29	29.68
	10	3.18	51.9	3.91	35.02
	20	2.63	30.84	3.02	24.37
	40	2.61	24.75	3.37	22.91
	80	3.91	25.23	4.42	22.78
E022-06m-50	5	1.15	3.19	1.71	3.98
	10	1.65	6.48	2.03	8.01
	20	1.38	5.52	1.61	6.34
	40	1.44	3.95	1.47	4.24
	80	1.19	3.39	1.27	3.13
E022-06m-75	5	1.80	6.21	2.85	9.89
	10	1.65	7.08	2.53	10.26
	20	2.14	5.38	2.41	6.43
	40	1.88	4.08	2.02	4.33
	80	1.40	3.97	1.51	4.24
E030-04s-50	5	0.01	0.06	0.01	0.09
	10	0.09	0.51	0.14	0.53
	20	0.11	0.32	0.15	0.46
	40	0.62	0.83	0.73	0.73
	80	0.55	0.57	0.44	0.61
E030-04s-75	5	0.01	0.06	0.01	0.09
	10	0.27	0.31	0.24	0.37
	20	0.36	0.42	0.50	0.63
	40	0.43	0.94	0.51	0.73
	80	0.40	1.27	0.49	1.16
E051-05e-50	5	5.87	20.07	6.96	17.75
	10	5.49	13.39	6.45	15.12
	20	4.12	9.69	4.71	12.20
	40	2.34	7.09	2.94	9.08
	80	2.65	7.03	2.57	7.80
E051-05e-75	5	6.22	17.33	7.95	17.54
	10	4.28	11.65	5.29	15.05
	20	4.10	12.05	4.79	13.26
	40	3.04	10.48	3.46	10.42
	80	2.54	7.19	2.62	8.41
E076-07s-50	5	2.02	7.22	2.80	8.77
	10	1.99	5.05	2.33	6.11
	20	1.74	3.87	2.00	4.08
	40	1.68	3.06	1.93	3.04
	80	0.71	1.91	0.93	2.30
E076-07s-75	5	2.50	9.62	3.48	9.55
	10	2.82	6.76	3.19	7.21
	20	2.15	3.97	2.58	4.73
	40	1.99	3.22	2.21	3.80
	80	1.14	2.28	1.30	2.64

Table 4 Continued: Out-of-sample test summary (objective function values for Tan's algorithm)

Instance	Size of scenario set	Difference in $C(x)$ (%)	Difference in $R^{IV}(x)$ (%)	Average CV $C(x)$ (%)	Average CV $R^{IV}(x)$ (%)
E101-10c-50	5	3.98	8.28	4.20	9.32
	10	4.52	5.89	4.67	6.85
	20	2.73	3.98	3.37	5.04
	40	2.08	3.31	2.30	3.19
	80	1.67	2.34	1.51	2.38
E101-10c-75	5	3.55	9.34	4.40	9.47
	10	3.71	5.99	5.05	7.73
	20	4.57	4.39	4.84	5.28
	40	2.27	3.21	2.26	3.60
	80	1.34	2.52	1.28	2.58

Table 5: $\mathcal{S}$ metric performance indicator

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E016-03m-50	(427.13 , 50.69)	TAN	6 273	6 106	6 432
		SMGRASP	6 205	5 919	6 356
E016-03m-75	(407.47 , 53.61)	TAN	5 510	5 089	5 693
		SMGRASP	5 524	5 377	5 845
E016-05m-50	(547.93 , 44.64)	TAN	6 669	6 175	6 991
		SMGRASP	6 733	6 339	6 985
E016-05m-75	(553.88 , 45.57)	TAN	7 121	6 684	7 444
		SMGRASP	6 880	6 523	7 561
E021-04m-50	(614.43 , 76.91)	TAN	13 963	12 941	15 471
		SMGRASP	16 013	15 517	16 556
E021-04m-75	(604.13 , 89.93)	TAN	16 011	14 486	17 037
		SMGRASP	17 859	17 395	18 499
E021-06m-50	(691.02 , 84.31)	TAN	13 587	12 282	15 783
		SMGRASP	15 043	14 363	15 736
E021-06m-75	(750.48 , 88.37)	TAN	18 400	16 940	19 166
		SMGRASP	20 356	19 252	21 460
E022-06m-50	(895.02 , 45.64)	TAN	10 699	9 653	12 060
		SMGRASP	11 323	10 034	11 810
E022-06m-75	(890.88 , 43.73)	TAN	9 817	8 784	10 699
		SMGRASP	10 063	9 031	10 952
E023-05s-50	(1 366.34 , 290.11)	TAN	178 745	175 344	182 660
		SMGRASP	187 793	185 727	190 089
E023-05s-75	(1 332.98 , 290.11)	TAN	170 369	165 526	175 035
		SMGRASP	178 177	175 684	179 868
E026-08m-50	(1 082.93 , 105.09)	TAN	25 883	23 902	28 644
		SMGRASP	33 041	31 433	34 392
E026-08m-75	(1 063.27 , 120.00)	TAN	27 950	24 835	33 018
		SMGRASP	34 209	32 263	36 109
E030-04S-50	(1 128.63 , 163.40)	TAN	87 289	84 772	90 104
		SMGRASP	90 240	89 229	91 239
E030-04S-75	(1 082.70 , 163.40)	TAN	80 295	78 479	82 798
		SMGRASP	82 898	81 589	83 896

Table 5 Continued: $\mathcal{S}$ metric performance indicator

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E031-09h-50	(1 038.75 , 83.81)	TAN	14 733	11 864	18 141
		SMGRASP	22 474	20 560	24 605
E031-09h-75	(996.66 , 82.26)	TAN	11 373	8 772	13 205
		SMGRASP	18 618	17 150	20 442
E033-05s-50	(1 369.92 , 325.45)	TAN	120 001	116 537	123 358
		SMGRASP	123 444	118 293	127 812
E033-05s-75	(1 369.92 , 339.93)	TAN	127 278	123 892	131 415
		SMGRASP	131 146	125 028	135 628
E036-11h-50	(1 149.47 , 73.85)	TAN	12 318	10 732	13 889
		SMGRASP	19 510	16 147	22 374
E036-11h-75	(1 156.97 , 81.58)	TAN	12 500	10 546	16 608
		SMGRASP	22 469	20 373	24 454
E041-14h-50	(1 485.43 , 111.92)	TAN	30 509	27 719	34 647
		SMGRASP	42 683	41 370	44 949
E041-14h-75	(1 565.33 , 83.07)	TAN	23 417	18 702	28 470
		SMGRASP	32 492	29 099	36 667
E048-04y-50	(132 372.00 , 12 620.20)	TAN	1.03×10^9	1.00×10^9	1.06×10^9
		SMGRASP	1.11×10^9	1.11×10^9	1.12×10^9
E048-04y-75	(104 464.00 , 10 912.20)	TAN	5.81×10^8	5.54×10^8	6.16×10^8
		SMGRASP	6.48×10^8	6.41×10^8	6.57×10^8
E051-05e-50	(1 068.18 , 91.95)	TAN	23 287	21 373	26 827
		SMGRASP	45 267	44 427	46 032
E051-05e-75	(1 174.32 , 89.25)	TAN	30 469	28 481	34 531
		SMGRASP	53 106	51 887	54 467
E076-07s-50	(1 534.90 , 148.19)	TAN	69 639	67 428	77 007
		SMGRASP	113 322	108 675	116 956
E076-07s-75	(1 648.56 , 132.76)	TAN	69 654	64 862	74 622
		SMGRASP	115 172	110 578	118 367
E076-08s-50	(1 674.69 , 149.79)	TAN	57 702	54 182	63 806
		SMGRASP	122 053	112 821	125 427
E076-08s-75	(1 705.66 , 154.56)	TAN	60 541	55 827	66 575
		SMGRASP	129 794	123 005	137 575
E076-14u-50	(2 319.46 , 185.43)	TAN	59 436	51 827	64 578
		SMGRASP	155 392	146 492	166 145
E076-14u-75	(2 419.92 , 162.43)	TAN	58 079	55 033	62 477
		SMGRASP	144 919	136 741	150 401
E101-08e-50	(1 886.44 , 133.87)	TAN	47 163	42 648	52 398
		SMGRASP	120 610	115 011	128 582
E101-08e-75	(2 003.35 , 216.85)	TAN	117 362	100 962	141 211
		SMGRASP	219 331	213 936	226 362
E101-10c-50	(2 568.18 , 137.67)	TAN	71 066	51 689	82 499
		SMGRASP	199 713	189 062	208 087
E101-10c-75	(2 408.86 , 149.81)	TAN	67 482	57 852	78 393
		SMGRASP	190 182	176 007	201 390
E101-14s-50	(2 575.86 , 195.07)	TAN	75 260	66 496	83 547
		SMGRASP	216 395	201 143	221 943

Table 5 Continued: $\mathcal{S}$ metric performance indicator

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E101-14s-75	(2 582.34 , 163.40)	TAN	58 033	45 954	69 995
		SMGRASP	178 681	174 986	183 409

Table 6: $\mathcal{C}$ metric performance indicator

Instance	Metric	Average	Min	Max
E016-03m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.48	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.14	0.00	0.56
E016-03m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.39	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.19	0.00	0.69
E016-05m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.25	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.22	0.00	0.67
E016-05m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.45	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.16	0.00	0.55
E021-04m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.04	0.00	0.56
	$\mathcal{C}$ (SMGRASP, TAN)	0.59	0.13	1.00
E021-04m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.04	0.00	0.33
	$\mathcal{C}$ (SMGRASP, TAN)	0.66	0.00	1.00
E021-06m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.04	0.00	0.60
	$\mathcal{C}$ (SMGRASP, TAN)	0.49	0.00	1.00
E021-06m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.02	0.00	0.29
	$\mathcal{C}$ (SMGRASP, TAN)	0.72	0.22	1.00
E022-06m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.11	0.00	0.44
	$\mathcal{C}$ (SMGRASP, TAN)	0.59	0.08	0.94
E022-06m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.16	0.00	0.48
	$\mathcal{C}$ (SMGRASP, TAN)	0.55	0.11	1.00
E023-05s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.27	0.09	0.42
	$\mathcal{C}$ (SMGRASP, TAN)	0.45	0.21	0.84
E023-05s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.27	0.07	0.51
	$\mathcal{C}$ (SMGRASP, TAN)	0.44	0.14	0.83
E026-08m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.27
	$\mathcal{C}$ (SMGRASP, TAN)	0.97	0.53	1.00
E026-08m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.02	0.00	0.38
	$\mathcal{C}$ (SMGRASP, TAN)	0.81	0.00	1.00
E030-04S-50	$\mathcal{C}$ (TAN, SMGRASP)	0.05	0.00	0.14
	$\mathcal{C}$ (SMGRASP, TAN)	0.60	0.27	0.87
E030-04S-75	$\mathcal{C}$ (TAN, SMGRASP)	0.05	0.00	0.16
	$\mathcal{C}$ (SMGRASP, TAN)	0.67	0.33	0.90
E031-09h-50	$\mathcal{C}$ (TAN, SMGRASP)	0.01	0.00	0.17
	$\mathcal{C}$ (SMGRASP, TAN)	0.93	0.40	1.00
E031-09h-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.97	0.58	1.00
E033-05s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.05	0.00	0.13
	$\mathcal{C}$ (SMGRASP, TAN)	0.73	0.43	0.94
E033-05s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.04	0.00	0.17
	$\mathcal{C}$ (SMGRASP, TAN)	0.72	0.44	0.98

Table 6 Continued: $\mathcal{C}$ metric performance indicator

Instance	Metric	Average	Min	Max
E036-11h-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.04
	$\mathcal{C}$ (SMGRASP, TAN)	0.88	0.07	1.00
E036-11h-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.94	0.29	1.00
E041-14h-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.86	0.29	1.00
E041-14h-75	$\mathcal{C}$ (TAN, SMGRASP)	0.01	0.00	0.18
	$\mathcal{C}$ (SMGRASP, TAN)	0.83	0.00	1.00
E048-04y-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.09
	$\mathcal{C}$ (SMGRASP, TAN)	0.96	0.61	1.00
E048-04y-75	$\mathcal{C}$ (TAN, SMGRASP)	0.01	0.00	0.14
	$\mathcal{C}$ (SMGRASP, TAN)	0.97	0.63	1.00
E051-05e-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.98	0.73	1.00
E051-05e-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.09
	$\mathcal{C}$ (SMGRASP, TAN)	0.97	0.58	1.00
E076-07s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.99	0.69	1.00
E076-07s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.95	0.00	1.00
E076-08s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.94	0.58	1.00
E076-08s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	1.00	0.91	1.00
E076-14u-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.81	0.36	1.00
E076-14u-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.80	0.36	1.00
E101-08e-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.95	0.45	1.00
E101-08e-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.91	0.50	1.00
E101-10c-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	1.00	1.00	1.00
E101-10c-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.99	0.91	1.00
E101-14s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.85	0.29	1.00
E101-14s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.85	0.29	1.00

Table 7: Average running time (in seconds)

Instance	TAN	SMGRASP
E016-03m-50	1 239	736
E016-03m-75	1 238	651
E016-05m-50	1 172	398
E016-05m-75	1 143	644

Table 7 Continued: Average running time (in seconds)

Instance	TAN	SMGRASP
E021-04m-50	938	1 607
E021-04m-75	826	1 397
E021-06m-50	773	1 379
E021-06m-75	770	1 294
E022-06m-50	840	2 779
E022-06m-75	839	2 105
E023-05s-50	1 162	4 630
E023-05s-75	1 162	4 413
E026-08m-50	766	3 078
E026-08m-75	754	2 539
E030-04S-50	2 359	7 984
E030-04S-75	2 351	7 308
E031-09h-50	1 735	7 614
E031-09h-75	1 728	6 691
E033-05s-50	2 729	9 623
E033-05s-75	2 534	8 396
E036-11h-50	2 163	10 231
E036-11h-75	2 086	10 672
E041-14h-50	2 430	10 801
E041-14h-75	2 450	10 518
E048-04y-50	3 673	10 801
E048-04y-75	2 915	10 801
E051-05e-50	3 386	10 800
E051-05e-75	2 743	10 800
E076-07s-50	4 850	10 805
E076-07s-75	3 535	10 803
E076-08s-50	4 838	10 801
E076-08s-75	3 526	10 803
E076-14u-50	3 454	10 809
E076-14u-75	2 879	10 812
E101-08e-50	5 806	10 821
E101-08e-75	3 928	10 811
E101-10c-50	5 916	10 811
E101-10c-75	4 051	10 808
E101-14s-50	6 169	10 820
E101-14s-75	4 118	10 823

Table 8: $\mathcal{S}$ metric performance indicator (tests with similar running time)

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E016-03m-50	(427.13 , 50.69)	TAN	6 273	6 106	6 432
		SMGRASP	6 219	5 919	6 356
E016-03m-75	(384.27 , 53.61)	TAN	4 278	3 856	4 452
		SMGRASP	4 317	4 154	4 606
E016-05m-50	(547.93 , 44.64)	TAN	6 669	6 175	6 991
		SMGRASP	6 763	6 339	6 985
E016-05m-75	(553.88 , 45.57)	TAN	7 121	6 684	7 444
		SMGRASP	6 885	6 531	7 571

Table 8 Continued: $\mathcal{S}$ metric performance indicator(tests with similar running time)

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E021-04m-50	(614.43 , 78.00)	TAN	14 190	13 151	15 718
		SMGRASP	16 167	15 705	16 614
E021-04m-75	(604.13 , 89.93)	TAN	16 011	14 486	17 037
		SMGRASP	17 741	17 342	18 245
E021-06m-50	(699.27 , 87.45)	TAN	14 852	13 511	17 112
		SMGRASP	16 145	15 615	16 992
E021-06m-75	(750.48 , 88.37)	TAN	18 400	16 940	19 166
		SMGRASP	20 118	18 495	21 460
E022-06m-50	(895.02 , 45.64)	TAN	10 699	9 653	12 060
		SMGRASP	10 855	9 606	11 646
E022-06m-75	(890.88 , 43.73)	TAN	9 817	8 784	10 699
		SMGRASP	9 819	8 713	10 797
E023-05s-50	(1 366.34 , 290.11)	TAN	178 745	175 344	182 660
		SMGRASP	185 088	183 576	188 396
E023-05s-75	(1 332.98 , 290.11)	TAN	170 369	165 526	175 035
		SMGRASP	176 029	172 157	178 705
E026-08m-50	(1 082.93 , 105.09)	TAN	25 883	23 902	28 644
		SMGRASP	31 672	30 103	34 392
E026-08m-75	(1 063.27 , 120.00)	TAN	27 950	24 835	33 018
		SMGRASP	33 261	31 856	34 680
E030-04S-50	(1 128.63 , 163.40)	TAN	87 289	84 772	90 104
		SMGRASP	89 199	86 973	91 186
E030-04S-75	(1 082.70 , 163.40)	TAN	80 295	78 479	82 798
		SMGRASP	82 011	80 739	83 606
E031-09h-50	(1 038.75 , 83.81)	TAN	14 733	11 864	18 141
		SMGRASP	21 432	19 361	23 272
E031-09h-75	(996.66 , 81.45)	TAN	11 204	8 620	13 016
		SMGRASP	17 552	15 479	19 444
E033-05s-50	(1 369.92 , 325.45)	TAN	120 001	116 537	123 358
		SMGRASP	121 257	113 193	126 808
E033-05s-75	(1 369.92 , 339.93)	TAN	127 278	123 892	131 415
		SMGRASP	127 762	120 837	133 448
E036-11h-50	(1 149.47 , 80.97)	TAN	14 189	12 493	16 012
		SMGRASP	18 805	17 068	20 828
E036-11h-75	(1 156.97 , 81.58)	TAN	12 500	10 546	16 608
		SMGRASP	19 205	14 150	22 570
E041-14h-50	(1 485.43 , 94.18)	TAN	23 916	21 480	27 319
		SMGRASP	29 082	25 397	31 550
E041-14h-75	(1 565.33 , 89.10)	TAN	25 973	20 961	31 296
		SMGRASP	31 301	29 180	33 877
E048-04y-50	(132 372.00 , 12 620.20)	TAN	1.03×10^9	1.00×10^9	1.06×10^9
		SMGRASP	1.11×10^9	1.10×10^9	1.12×10^9
E048-04y-75	(104 464.00 , 10 912.20)	TAN	5.81×10^8	5.54×10^8	6.16×10^8
		SMGRASP	6.42×10^8	6.26×10^8	6.54×10^8
E051-05e-50	(1 080.07 , 91.95)	TAN	24 273	22 381	27 809
		SMGRASP	44 993	44 071	46 343

Table 8 Continued: $\mathcal{S}$ metric performance indicator(tests with similar running time)

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E051-05e-75	(1 174.32 , 89.25)	TAN	30 469	28 481	34 531
		SMGRASP	51 455	50 288	52 920
E076-07s-50	(1 498.69 , 148.19)	TAN	65 039	62 959	72 374
		SMGRASP	105 349	101 060	109 417
E076-07s-75	(1 648.56 , 132.76)	TAN	69 654	64 862	74 622
		SMGRASP	110 032	104 966	114 153
E076-08s-50	(1 651.24 , 149.79)	TAN	54 767	51 347	60 759
		SMGRASP	116 665	110 872	121 901
E076-08s-75	(1 705.66 , 154.56)	TAN	60 541	55 827	66 575
		SMGRASP	123 761	115 358	126 979
E076-14u-50	(2 319.46 , 185.43)	TAN	59 436	51 827	64 578
		SMGRASP	146 699	135 528	155 282
E076-14u-75	(2 419.92 , 162.43)	TAN	58 079	55 033	62 477
		SMGRASP	136 431	125 615	142 993
E101-08e-50	(1 962.27 , 155.96)	TAN	68 286	63 021	73 912
		SMGRASP	148 262	140 991	153 171
E101-08e-75	(2 003.35 , 216.85)	TAN	117 362	100 962	141 211
		SMGRASP	213 748	207 019	228 417
E101-10c-50	(2 568.18 , 137.67)	TAN	71 066	51 689	82 499
		SMGRASP	199 120	187 637	204 590
E101-10c-75	(2 408.86 , 149.81)	TAN	67 482	57 852	78 393
		SMGRASP	185 614	175 805	193 256
E101-14s-50	(2 607.20 , 195.07)	TAN	79 920	71 044	88 136
		SMGRASP	220 737	202 793	232 297
E101-14s-75	(2 582.34 , 163.40)	TAN	58 033	45 954	69 995
		SMGRASP	173 248	163 603	181 775

Table 9: $\mathcal{C}$ metric performance indicator (tests with similar running time)

Instance	Metric	Average	Min	Max
E016-03m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.47	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.14	0.00	0.56
E016-03m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.39	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.20	0.00	0.69
E016-05m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.25	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.23	0.00	0.67
E016-05m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.46	0.00	1.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.17	0.00	0.55
E021-04m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.05	0.00	0.67
	$\mathcal{C}$ (SMGRASP, TAN)	0.57	0.13	1.00
E021-04m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.04	0.00	0.33
	$\mathcal{C}$ (SMGRASP, TAN)	0.63	0.00	1.00
E021-06m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.07	0.00	0.60
	$\mathcal{C}$ (SMGRASP, TAN)	0.44	0.00	1.00
E021-06m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.03	0.00	0.29
	$\mathcal{C}$ (SMGRASP, TAN)	0.67	0.11	1.00

Table 9 Continued: $\mathcal{C}$ metric performance indicator (tests with similar running time)

Instance	Metric	Average	Min	Max
E022-06m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.21	0.00	0.81
	$\mathcal{C}$ (SMGRASP, TAN)	0.49	0.00	0.94
E022-06m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.22	0.00	0.61
	$\mathcal{C}$ (SMGRASP, TAN)	0.48	0.11	1.00
E023-05s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.40	0.19	0.59
	$\mathcal{C}$ (SMGRASP, TAN)	0.34	0.10	0.59
E023-05s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.35	0.14	0.62
	$\mathcal{C}$ (SMGRASP, TAN)	0.35	0.09	0.66
E026-08m-50	$\mathcal{C}$ (TAN, SMGRASP)	0.01	0.00	0.36
	$\mathcal{C}$ (SMGRASP, TAN)	0.91	0.46	1.00
E026-08m-75	$\mathcal{C}$ (TAN, SMGRASP)	0.06	0.00	0.55
	$\mathcal{C}$ (SMGRASP, TAN)	0.72	0.00	1.00
E030-04S-50	$\mathcal{C}$ (TAN, SMGRASP)	0.09	0.00	0.22
	$\mathcal{C}$ (SMGRASP, TAN)	0.54	0.27	0.83
E030-04S-75	$\mathcal{C}$ (TAN, SMGRASP)	0.07	0.00	0.23
	$\mathcal{C}$ (SMGRASP, TAN)	0.61	0.27	0.80
E031-09h-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.08
	$\mathcal{C}$ (SMGRASP, TAN)	0.82	0.20	1.00
E031-09h-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.10
	$\mathcal{C}$ (SMGRASP, TAN)	0.94	0.57	1.00
E033-05s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.07	0.00	0.22
	$\mathcal{C}$ (SMGRASP, TAN)	0.66	0.30	0.91
E033-05s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.07	0.00	0.19
	$\mathcal{C}$ (SMGRASP, TAN)	0.66	0.37	0.97
E036-11h-50	$\mathcal{C}$ (TAN, SMGRASP)	0.01	0.00	0.29
	$\mathcal{C}$ (SMGRASP, TAN)	0.54	0.00	1.00
E036-11h-75	$\mathcal{C}$ (TAN, SMGRASP)	0.03	0.00	0.40
	$\mathcal{C}$ (SMGRASP, TAN)	0.72	0.25	1.00
E041-14h-50	$\mathcal{C}$ (TAN, SMGRASP)	0.03	0.00	0.40
	$\mathcal{C}$ (SMGRASP, TAN)	0.52	0.00	1.00
E041-14h-75	$\mathcal{C}$ (TAN, SMGRASP)	0.03	0.00	0.36
	$\mathcal{C}$ (SMGRASP, TAN)	0.59	0.00	1.00
E048-04y-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.09
	$\mathcal{C}$ (SMGRASP, TAN)	0.92	0.43	1.00
E048-04y-75	$\mathcal{C}$ (TAN, SMGRASP)	0.02	0.00	0.27
	$\mathcal{C}$ (SMGRASP, TAN)	0.92	0.38	1.00
E051-05e-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.89	0.36	1.00
E051-05e-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.14
	$\mathcal{C}$ (SMGRASP, TAN)	0.91	0.58	1.00
E076-07s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.96	0.62	1.00
E076-07s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.86	0.00	1.00
E076-08s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.83	0.42	1.00
E076-08s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.88	0.45	1.00

Table 9 Continued: $\mathcal{C}$ metric performance indicator (tests with similar running time)

Instance	Metric	Average	Min	Max
E076-14u-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.17
	$\mathcal{C}$ (SMGRASP, TAN)	0.65	0.20	1.00
E076-14u-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.61	0.07	1.00
E101-08e-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.91	0.36	1.00
E101-08e-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.70	0.25	1.00
E101-10c-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.99	0.75	1.00
E101-10c-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.95	0.55	1.00
E101-14s-50	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.83	0.35	1.00
E101-14s-75	$\mathcal{C}$ (TAN, SMGRASP)	0.00	0.00	0.00
	$\mathcal{C}$ (SMGRASP, TAN)	0.75	0.36	1.00

Table 10: $\mathcal{S}$ metric performance indicator. Comparison between SMGRASP and SMGRASP with no distance insertion strategy (SMGRASP-NDI)

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E016-03m-50	(439.37 , 19.82)	SMGRASP-NDI	2 393	2 312	2 556
		SMGRASP	2 373	2 177	2 556
E016-03m-75	(454.25 , 53.61)	SMGRASP-NDI	7 927	7 705	8 178
		SMGRASP	7 982	7 798	8 344
E016-05m-50	(486.05 , 42.95)	SMGRASP-NDI	4 132	3 869	4 319
		SMGRASP	4 108	3 818	4 315
E016-05m-75	(547.71 , 45.57)	SMGRASP-NDI	6 588	6 159	7 244
		SMGRASP	6 632	6 279	7 294
E021-04m-50	(585.94 , 74.49)	SMGRASP-NDI	13 476	12 963	14 054
		SMGRASP	13 442	12 985	13 951
E021-04m-75	(604.13 , 75.57)	SMGRASP-NDI	14 805	14 073	15 451
		SMGRASP	14 826	14 402	15 393
E021-06m-50	(712.87 , 84.31)	SMGRASP-NDI	16 536	15 863	17 461
		SMGRASP	16 701	15 936	17 461
E021-06m-75	(692.97 , 88.37)	SMGRASP-NDI	15 647	14 733	16 646
		SMGRASP	15 699	14 773	16 687
E022-06m-50	(901.87 , 41.40)	SMGRASP-NDI	9 690	8 595	10 590
		SMGRASP	9 860	8 548	10 351
E022-06m-75	(904.54 , 42.62)	SMGRASP-NDI	10 121	8 879	12 684
		SMGRASP	10 069	8 979	11 048
E023-05s-50	(1 366.34 , 290.11)	SMGRASP-NDI	186 382	183 617	187 955
		SMGRASP	187 793	185 727	190 089
E023-05s-75	(1 400.45 , 290.11)	SMGRASP-NDI	195 759	192 837	197 063
		SMGRASP	197 639	195 122	199 332
E026-08m-50	(855.45 , 100.99)	SMGRASP-NDI	10 975	9 940	12 501
		SMGRASP	11 181	9 965	12 116

Table 10 Continued: $\mathcal{S}$ metric performance indicator. Comparison between SMGRASP and SMGRASP-NDI

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E026-08m-75	(994.64 , 103.51)	SMGRASP-NDI	22 336	21 372	23 659
		SMGRASP	22 271	20 772	23 804
E030-04S-50	(1 095.65 , 163.40)	SMGRASP-NDI	83 925	81 398	85 921
		SMGRASP	84 855	83 840	85 855
E030-04S-75	(1 082.70 , 249.16)	SMGRASP-NDI	130 731	125 879	132 180
		SMGRASP	132 381	130 793	133 437
E031-09h-50	(965.85 , 62.61)	SMGRASP-NDI	11 553	9 155	13 219
		SMGRASP	11 709	10 191	13 284
E031-09h-75	(839.41 , 82.26)	SMGRASP-NDI	7 575	5 738	8 728
		SMGRASP	7 915	7 216	9 140
E033-05s-50	(1 406.07 , 325.45)	SMGRASP-NDI	135 373	131 184	137 455
		SMGRASP	135 201	130 055	139 563
E033-05s-75	(1 418.24 , 339.93)	SMGRASP-NDI	145 744	140 567	151 347
		SMGRASP	147 561	141 439	152 040
E036-11h-50	(973.65 , 73.85)	SMGRASP-NDI	9 192	6 894	11 328
		SMGRASP	9 326	6 742	11 609
E036-11h-75	(999.21 , 67.94)	SMGRASP-NDI	9 214	7 799	9 970
		SMGRASP	9 245	8 005	10 579
E041-14h-50	(1 190.97 , 111.92)	SMGRASP-NDI	14 855	12 501	19 013
		SMGRASP	15 699	14 052	17 571
E041-14h-75	(1 244.38 , 102.48)	SMGRASP-NDI	15 822	12 197	19 062
		SMGRASP	16 272	13 670	18 759
E048-04y-50	(85 834.40 , 8 947.00)	SMGRASP-NDI	3.67×10^8	3.54×10^8	3.73×10^8
		SMGRASP	3.70×10^8	3.61×10^8	3.78×10^8
E048-04y-75	(87 646.20 , 9 524.24)	SMGRASP-NDI	3.97×10^8	3.85×10^8	4.11×10^8
		SMGRASP	4.04×10^8	3.98×10^8	4.11×10^8
E051-05e-50	(1 141.26 , 35.81)	SMGRASP-NDI	18 648	17 790	19 621
		SMGRASP	19 078	18 310	19 802
E051-05e-75	(1 064.52 , 31.82)	SMGRASP-NDI	13 826	13 009	14 898
		SMGRASP	14 497	13 865	15 330
E076-07s-50	(1 584.83 , 125.78)	SMGRASP-NDI	96 903	94 033	98 957
		SMGRASP	101 741	97 257	105 153
E076-07s-75	(1 667.90 , 121.88)	SMGRASP-NDI	102 966	98 245	107 132
		SMGRASP	107 619	102 986	110 794
E076-08s-50	(1 685.95 , 124.20)	SMGRASP-NDI	92 275	87 323	98 195
		SMGRASP	101 583	93 250	104 750
E076-08s-75	(1 676.06 , 102.43)	SMGRASP-NDI	74 911	71 113	79 549
		SMGRASP	81 104	75 885	86 985
E076-14u-50	(2 166.30 , 116.68)	SMGRASP-NDI	71 960	61 477	83 842
		SMGRASP	72 815	66 895	81 299
E076-14u-75	(2 146.39 , 125.81)	SMGRASP-NDI	73 626	62 908	84 615
		SMGRASP	77 430	71 120	81 924
E101-08e-50	(1 940.77 , 141.70)	SMGRASP-NDI	127 411	118 534	132 802
		SMGRASP	135 728	130 142	143 928
E101-08e-75	(2 003.35 , 136.31)	SMGRASP-NDI	127 393	117 717	133 040
		SMGRASP	133 698	130 113	137 361

Table 10 Continued: $\mathcal{S}$ metric performance indicator. Comparison between SMGRASP and SMGRASP-NDI

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E101-10c-50	(2 553.31 , 115.81)	SMGRASP-NDI	152 763	140 687	164 994
		SMGRASP	162 827	153 943	170 434
E101-10c-75	(2 452.39 , 166.81)	SMGRASP-NDI	214 243	204 168	229 610
		SMGRASP	221 099	206 032	232 638
E101-14s-50	(2 584.37 , 145.30)	SMGRASP-NDI	144 466	134 106	157 375
		SMGRASP	154 439	140 500	159 954
E101-14s-75	(2 675.23 , 135.80)	SMGRASP-NDI	142 059	129 079	156 375
		SMGRASP	155 084	150 925	159 885

Table 11: $\mathcal{C}$ metric performance indicator. Comparison between SMGRASP and SMGRASP-NDI

Instance	Metric	Average	Min	Max
E016-03m-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.33	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.30	0.00	1.00
E016-03m-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.31	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.39	0.00	1.00
E016-05m-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.38	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.37	0.00	1.00
E016-05m-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.28	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.31	0.00	1.00
E021-04m-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.37	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.33	0.00	1.00
E021-04m-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.30	0.00	0.88
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.34	0.00	1.00
E021-06m-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.21	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.25	0.00	1.00
E021-06m-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.34	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.42	0.00	1.00
E022-06m-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.33	0.00	0.88
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.40	0.00	0.82
E022-06m-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.34	0.00	0.90
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.40	0.00	0.78
E023-05s-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.30	0.09	0.63
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.56	0.26	0.92
E023-05s-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.30	0.05	0.71
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.62	0.20	0.88
E026-08m-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.36	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.43	0.00	1.00
E026-08m-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.31	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.37	0.00	1.00
E030-04S-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.37	0.06	0.73
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.38	0.12	0.83
E030-04S-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.38	0.04	0.61
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.41	0.12	0.81
E031-09h-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.38	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.34	0.00	1.00

Table 11 Continued: $\mathcal{C}$ metric performance indicator. Comparison between SMGRASP and SMGRASP-NDI

Instance	Metric	Average	Min	Max
E031-09h-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.27	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.40	0.00	1.00
E033-05s-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.38	0.04	0.88
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.37	0.02	0.92
E033-05s-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.41	0.08	0.81
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.34	0.08	0.84
E036-11h-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.30	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.40	0.00	1.00
E036-11h-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.31	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.41	0.00	1.00
E041-14h-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.20	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.46	0.00	1.00
E041-14h-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.32	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.38	0.00	1.00
E048-04y-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.28	0.00	0.88
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.39	0.00	0.95
E048-04y-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.26	0.00	0.96
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.48	0.00	1.00
E051-05e-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.18	0.00	0.67
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.48	0.00	1.00
E051-05e-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.22	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.53	0.00	1.00
E076-07s-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.08	0.00	0.60
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.77	0.08	1.00
E076-07s-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.06	0.00	0.47
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.68	0.14	1.00
E076-08s-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.04	0.00	0.90
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.73	0.00	1.00
E076-08s-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.05	0.00	0.70
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.78	0.00	1.00
E076-14u-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.35	0.00	1.00
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.29	0.00	1.00
E076-14u-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.17	0.00	0.90
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.47	0.00	1.00
E101-08e-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.07	0.00	0.46
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.65	0.10	1.00
E101-08e-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.10	0.00	0.64
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.68	0.00	1.00
E101-10c-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.15	0.00	0.77
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.64	0.05	1.00
E101-10c-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.17	0.00	0.70
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.50	0.00	1.00
E101-14s-50	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.15	0.00	0.56
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.52	0.00	1.00
E101-14s-75	$\mathcal{C}$ (SMGRASP-NDI , SMGRASP)	0.13	0.00	0.82
	$\mathcal{C}$ (SMGRASP, SMGRASP-NDI)	0.55	0.00	1.00

Table 12: $\mathcal{S}$ metric performance indicator. Comparison between SMGRASP and SMGRASP with virtual capacity (VC105)

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E016-03m-50	(425.42 , 25.21)	VC	3 075	2 982	3 183
		SMGRASP	2 814	2 608	2 979
E016-03m-75	(449.61 , 53.61)	VC	8 317	8 170	8 568
		SMGRASP	7 739	7 560	8 096
E016-05m-50	(551.39 , 51.52)	VC	8 792	8 261	9 233
		SMGRASP	8 128	7 708	8 397
E016-05m-75	(540.76 , 51.69)	VC	8 126	7 796	8 516
		SMGRASP	7 390	7 051	8 047
E021-04m-50	(604.97 , 71.29)	VC	15 089	14 906	15 605
		SMGRASP	14 111	13 641	14 624
E021-04m-75	(604.13 , 75.57)	VC	15 569	15 084	16 166
		SMGRASP	14 826	14 402	15 393
E021-06m-50	(691.02 , 84.31)	VC	15 921	15 499	16 326
		SMGRASP	15 043	14 363	15 736
E021-06m-75	(699.14 , 88.37)	VC	17 223	16 767	17 881
		SMGRASP	16 198	15 253	17 199
E022-06m-50	(874.59 , 41.40)	VC	9 208	8 355	11 167
		SMGRASP	8 990	7 772	9 458
E022-06m-75	(919.46 , 42.62)	VC	10 805	9 784	12 059
		SMGRASP	10 548	9 396	11 625
E023-05s-50	(1 366.34 , 290.11)	VC	187 374	185 594	188 719
		SMGRASP	187 793	185 727	190 089
E023-05s-75	(1 332.98 , 290.11)	VC	177 323	175 130	179 561
		SMGRASP	178 177	175 684	179 868
E026-08m-50	(836.08 , 100.99)	VC	10 680	9 563	12 036
		SMGRASP	9 456	8 270	10 360
E026-08m-75	(987.77 , 103.51)	VC	23 595	21 915	25 546
		SMGRASP	21 658	20 192	23 168
E030-04S-50	(1 081.41 , 163.40)	VC	82 647	81 370	83 407
		SMGRASP	82 529	81 514	83 530
E030-04S-75	(1 082.70 , 163.40)	VC	82 815	81 297	83 716
		SMGRASP	82 898	81 589	83 896
E031-09h-50	(1 023.83 , 64.31)	VC	17 709	15 525	19 298
		SMGRASP	15 042	13 541	16 811
E031-09h-75	(852.49 , 82.26)	VC	11 474	10 669	12 204
		SMGRASP	8 805	8 042	10 022
E033-05s-50	(1 398.37 , 336.19)	VC	132 516	124 178	143 074
		SMGRASP	138 511	133 390	142 847
E033-05s-75	(1 459.33 , 339.93)	VC	153 836	146 561	160 394
		SMGRASP	161 516	155 390	165 993
E036-11h-50	(1 082.26 , 73.85)	VC	16 779	15 139	18 021
		SMGRASP	15 617	12 645	18 259
E036-11h-75	(1 008.42 , 61.83)	VC	9 610	8 402	10 537
		SMGRASP	8 445	7 196	9 708
E041-14h-50	(1 192.42 , 111.92)	VC	17 603	15 208	19 251
		SMGRASP	15 832	14 189	17 696

Table 12 Continued: $\mathcal{S}$ metric performance indicator. Comparison between SMGRASP and VC105

Instance	Ref. Point (Length, Balance)	Algorithm	Average	Min	Max
E041-14h-75	(1 244.38 , 83.07)	VC	14 390	11 972	16 195
		SMGRASP	12 263	10 085	14 244
E048-04y-50	(84 757.10 , 18 536.00)	VC	7.56×10^8	7.41×10^8	7.68×10^8
		SMGRASP	7.62×10^8	7.52×10^8	7.73×10^8
E048-04y-75	(87 646.20 , 19 201.00)	VC	8.26×10^8	7.97×10^8	8.46×10^8
		SMGRASP	8.30×10^8	8.13×10^8	8.42×10^8
E051-05e-50	(1 041.48 , 38.51)	VC	17 279	16 624	18 018
		SMGRASP	17 032	16 455	17 658
E051-05e-75	(1 023.34 , 32.50)	VC	13 807	13 124	14 330
		SMGRASP	13 595	12 992	14 402
E076-07s-50	(1 534.90 , 120.09)	VC	92 296	89 328	96 098
		SMGRASP	91 042	86 593	94 376
E076-07s-75	(1 648.56 , 122.09)	VC	106 193	99 103	110 553
		SMGRASP	105 492	100 874	108 665
E076-08s-50	(1 674.69 , 149.95)	VC	124 176	118 427	129 011
		SMGRASP	122 183	112 946	125 559
E076-08s-75	(1 732.22 , 116.60)	VC	98 771	95 755	103 976
		SMGRASP	99 426	93 736	105 872
E076-14u-50	(2 017.99 , 112.35)	VC	72 535	61 802	78 404
		SMGRASP	56 386	51 389	64 314
E076-14u-75	(2 316.67 , 114.96)	VC	100 013	95 678	104 499
		SMGRASP	83 736	76 865	89 536
E101-08e-50	(1 843.34 , 133.87)	VC	116 649	111 924	122 080
		SMGRASP	114 945	109 384	122 900
E101-08e-75	(2 012.48 , 161.19)	VC	163 821	154 500	173 607
		SMGRASP	161 602	157 461	166 300
E101-10c-50	(2 553.31 , 160.39)	VC	222 405	214 168	236 775
		SMGRASP	234 073	221 627	243 180
E101-10c-75	(2 406.01 , 162.33)	VC	200 064	186 423	214 181
		SMGRASP	207 353	192 485	218 693
E101-14s-50	(2 641.81 , 161.72)	VC	183 334	166 600	201 881
		SMGRASP	184 108	169 660	189 761
E101-14s-75	(2 782.13 , 225.78)	VC	292 445	273 780	308 174
		SMGRASP	301 461	297 310	306 810

Table 13: $\mathcal{C}$ metric performance indicator. Comparison between SMGRASP and VC105

Instance	Metric	Average	Min	Max
E016-03m-50	$\mathcal{C}$ (VC, SMGRASP)	0.68	0.29	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.09	0.00	0.40
E016-03m-75	$\mathcal{C}$ (VC, SMGRASP)	0.63	0.14	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.12	0.00	0.36
E016-05m-50	$\mathcal{C}$ (VC, SMGRASP)	0.47	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.02	0.00	0.20
E016-05m-75	$\mathcal{C}$ (VC, SMGRASP)	0.68	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.03	0.00	0.20

Table 13 Continued: $\mathcal{C}$ metric performance indicator. Comparison between SMGRASP and VC105

Instance	Metric	Average	Min	Max
E021-04m-50	$\mathcal{C}$ (VC, SMGRASP)	0.70	0.20	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.06	0.00	0.50
E021-04m-75	$\mathcal{C}$ (VC, SMGRASP)	0.70	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.09	0.00	0.56
E021-06m-50	$\mathcal{C}$ (VC, SMGRASP)	0.51	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.06	0.00	0.43
E021-06m-75	$\mathcal{C}$ (VC, SMGRASP)	0.68	0.22	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.09	0.00	0.50
E022-06m-50	$\mathcal{C}$ (VC, SMGRASP)	0.39	0.05	0.83
	$\mathcal{C}$ (SMGRASP, VC)	0.33	0.04	0.81
E022-06m-75	$\mathcal{C}$ (VC, SMGRASP)	0.49	0.23	0.89
	$\mathcal{C}$ (SMGRASP, VC)	0.24	0.05	0.65
E023-05s-50	$\mathcal{C}$ (VC, SMGRASP)	0.39	0.11	0.72
	$\mathcal{C}$ (SMGRASP, VC)	0.48	0.18	0.87
E023-05s-75	$\mathcal{C}$ (VC, SMGRASP)	0.37	0.04	0.73
	$\mathcal{C}$ (SMGRASP, VC)	0.50	0.11	0.83
E026-08m-50	$\mathcal{C}$ (VC, SMGRASP)	0.64	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.11	0.00	0.61
E026-08m-75	$\mathcal{C}$ (VC, SMGRASP)	0.73	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.08	0.00	0.70
E030-04S-50	$\mathcal{C}$ (VC, SMGRASP)	0.34	0.06	0.75
	$\mathcal{C}$ (SMGRASP, VC)	0.45	0.09	0.91
E030-04S-75	$\mathcal{C}$ (VC, SMGRASP)	0.34	0.05	0.63
	$\mathcal{C}$ (SMGRASP, VC)	0.45	0.13	0.80
E031-09h-50	$\mathcal{C}$ (VC, SMGRASP)	0.75	0.17	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.05	0.00	0.56
E031-09h-75	$\mathcal{C}$ (VC, SMGRASP)	0.91	0.36	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.01	0.00	0.33
E033-05s-50	$\mathcal{C}$ (VC, SMGRASP)	0.24	0.01	0.75
	$\mathcal{C}$ (SMGRASP, VC)	0.54	0.09	0.93
E033-05s-75	$\mathcal{C}$ (VC, SMGRASP)	0.24	0.01	0.66
	$\mathcal{C}$ (SMGRASP, VC)	0.51	0.13	0.86
E036-11h-50	$\mathcal{C}$ (VC, SMGRASP)	0.49	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.26	0.00	1.00
E036-11h-75	$\mathcal{C}$ (VC, SMGRASP)	0.62	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.16	0.00	1.00
E041-14h-50	$\mathcal{C}$ (VC, SMGRASP)	0.44	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.18	0.00	0.79
E041-14h-75	$\mathcal{C}$ (VC, SMGRASP)	0.57	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.14	0.00	0.92
E048-04y-50	$\mathcal{C}$ (VC, SMGRASP)	0.31	0.00	0.93
	$\mathcal{C}$ (SMGRASP, VC)	0.44	0.00	1.00
E048-04y-75	$\mathcal{C}$ (VC, SMGRASP)	0.35	0.00	0.96
	$\mathcal{C}$ (SMGRASP, VC)	0.35	0.00	1.00
E051-05e-50	$\mathcal{C}$ (VC, SMGRASP)	0.44	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.27	0.00	1.00
E051-05e-75	$\mathcal{C}$ (VC, SMGRASP)	0.50	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.26	0.00	0.85

Table 13 Continued: $\mathcal{C}$ metric performance indicator. Comparison between SMGRASP and VC105

Instance	Metric	Average	Min	Max
E076-07s-50	$\mathcal{C}$ (VC, SMGRASP)	0.44	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.35	0.00	1.00
E076-07s-75	$\mathcal{C}$ (VC, SMGRASP)	0.43	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.31	0.00	1.00
E076-08s-50	$\mathcal{C}$ (VC, SMGRASP)	0.41	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.33	0.00	1.00
E076-08s-75	$\mathcal{C}$ (VC, SMGRASP)	0.35	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.44	0.00	1.00
E076-14u-50	$\mathcal{C}$ (VC, SMGRASP)	0.88	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.00	0.00	0.10
E076-14u-75	$\mathcal{C}$ (VC, SMGRASP)	0.92	0.20	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.00	0.00	0.20
E101-08e-50	$\mathcal{C}$ (VC, SMGRASP)	0.36	0.00	0.92
	$\mathcal{C}$ (SMGRASP, VC)	0.21	0.00	0.93
E101-08e-75	$\mathcal{C}$ (VC, SMGRASP)	0.35	0.00	0.90
	$\mathcal{C}$ (SMGRASP, VC)	0.32	0.00	1.00
E101-10c-50	$\mathcal{C}$ (VC, SMGRASP)	0.15	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.64	0.00	1.00
E101-10c-75	$\mathcal{C}$ (VC, SMGRASP)	0.22	0.00	0.91
	$\mathcal{C}$ (SMGRASP, VC)	0.54	0.00	1.00
E101-14s-50	$\mathcal{C}$ (VC, SMGRASP)	0.31	0.00	1.00
	$\mathcal{C}$ (SMGRASP, VC)	0.35	0.00	0.93
E101-14s-75	$\mathcal{C}$ (VC, SMGRASP)	0.22	0.00	0.83
	$\mathcal{C}$ (SMGRASP, VC)	0.37	0.00	1.00

Paper 5

The CVRP with soft time windows and stochastic travel times

The CVRP with soft time windows and stochastic travel times

Jorge Oyola

Molde University College, Molde, Norway

Abstract

A full multi-objective approach is employed in this paper to deal with a stochastic multi-objective CVRP. In this version of the problem, the demand is considered to be deterministic, but the travel times are assumed to be stochastic. A soft time window is tied to every customer and there is a penalty for starting the service outside the time window. Two objectives are minimized, the total length and the time window penalty. Our approach includes a NSGA and a VNS heuristic. It is tested on instances from the literature, obtaining approximations of the Pareto set, in contrast to a previous solution approach where even though the problem was modeled as multi-objective, no attempt was made to approximate the Pareto set. Our method is able to find solutions that dominate some of the previously known VRP solutions.

Keywords: Multi-objective VRP, Stochastic VRP (SVRP), VNS, NSGA, Stochastic travel time

1 Introduction

The capacitated vehicle routing problem (CVRP) is a well-known problem in transportation proposed in 1959 (Dantzig and Ramser, 1959). It is defined over an undirected graph $G(V, E)$, where $V = v_0, \ldots, v_N$ is a set of vertices and $E = (v_i, v_j) : v_i, v_j \in V, i < j$ is a set of edges. There is a symmetric matrix $C = [c_{ij}]$ that correspond to the travel costs along edge (v_i, v_j). Vertex v_0 represents the depot where there is a homogeneous fleet of m vehicles with capacity Q. A set of customers $V \setminus v_0$ with a non-negative known demand d_i must be served. A solution to the CVRP consists of m delivery routes with some specific conditions. Each route must start and end at the depot. Each customer must be visited once by exactly one vehicle. The summation of the demands of the customers in the same route, must be less than or equal to Q. A different approach where the demand corresponds to items that must be collected from the customers leads to an equivalent problem. The classic objective is minimization of total travel costs (Toth and Vigo, 2002).

The previous definition holds for the deterministic CVRP, however it is expected that in the real world one or more of the elements of the CVRP will be uncertain. Such elements are included in the models in the form of stochastic parameters. This variant of the problem became known as *stochastic (capacitated) vehicle routing problem* (SVRP or SCVRP) (Gendreau et al., 1996; Oyola et al., 2015).

Several variants of SVRP are modeled using stochastic parameters. The capacitated vehicle routing problem with stochastic demand (CVRPSD) is the most studied version of SVRP (Cordeau et al., 2007). Other parameters often modeled as stochastic are the presence of customers, travel times and service times (Gendreau et al., 1996; Oyola et al., 2015).

Delivery reliability defined as the on-time delivery of products and services, is a major competitive arena for many companies (Russell and Urban, 2008). The travel time between two customers may be affected by the congestion in the road and other eventualities, such as accidents. Which makes the CVRP with stochastic travel time a relevant problem to study. This paper deals with a

stochastic multi-objective vehicle routing problem with soft time windows (TW). The penalty for servicing the customers outside the time windows is included as objective, in addition to the traditional minimization of the total length. The travel times are assumed to be stochastic. This problem is known as the VRP with soft time windows and stochastic travel times (SVRPSTW).

A model for the SVRPSTW was formulated in Russell and Urban (2008). However, their solution approach was not entirely multi-objective. A weighted sum of objective functions is used to circumvent the existence of more than one objective in the problem. By using this approach the solution to a multi-objective problem could be partially approximated (Collette and Siarry, 2003), if several combinations of weights are assigned to the objective functions. Nevertheless only two combinations of weights were used. In this paper, on the other hand, we treat the problem employing a full multi-objective approach.

The rest of the paper is organized as follows: in Section 2 the SVRPSTW is explained; the algorithm used to find solutions to the problem is presented in Section 3; the computational tests and results are described in Section 4, with all corresponding tables presenting results summarized in Apendix A and the conclusions are presented in Section 5.

2 Problem definition

A multi-objective approach to the CVRP with soft time windows and stochastic travel times (SVRPSTW) is found in Russell and Urban (2008). In SVRPSTW the demand is known in advance, and there is a deterministic service time and a time window $[e_i, l_i]$ associated with each customer i. Starting service outside the time window is allowed at a cost, either for earliness or lateness. Three objectives are taken into consideration: the minimization of the number of vehicles, the total traveled distance and the total expected penalties for earliness and lateness in the service. The problem is modeled as stochastic programming with recourse (SPR), the recourse being the cost for servicing outside the time windows. A tabu search heuristic is used to solve the SVRPSTW. Even though the problem is modeled as a multi-objective problem, just two potentially Pareto solutions are found. The three objective functions are combined into a weighted single objective which is later minimized by the tabu search algorithm. Tests were performed using two different sets of weights, one solution is found in every case.

We will reformulate SVRPSTW as a bi-objective problem, where the expected total distance and the expected penalty cost (for starting the service outside the time windows) are minimized. The Number of vehicles is given as parameter, in principle it is possible to use the value found in Russell and Urban (2008). However it is possible to set different values to the number of vehicles and solve the bi-objective problem for each value. In such a case the obtained sets of solutions will approximate the Pareto set to the original SVRPSTW.

In Russell and Urban (2008) the problem is modeled as multi-objective and solved using weighted sum of objective functions. Two different set of weights are used so just two solutions are obtained for every instance. In contrast, the objective here is to approximate the Pareto front. The obtained solution sets are compared against the solutions presented in Russell and Urban (2008). The fact that the solution to the problem is presented in a form of approximation to the Pareto front, becomes one main contribution of this work. Since no previous attempts have been made to approximate its Pareto front, despite the fact that it has been previously formulated as a multi-objective problem. Another contribution of our approach comes from the fact that we propose an algorithm that succeeds finding such approximation.

2.1 Travel time and closed-form expressions for the penalties

The travel times are considered to follow a shifted gamma (α, β, γ) distribution, where α is restricted to take only integer values. This condition on the travel time allows the exact penalty computation, at the cost of generality. The arrival time at customer i, τ_i, is equal to the cumulative travel times plus the sum of the service times of all preceding customers. The earliness penalty at customer i, Ξ_i was expressed in Russell and Urban (2008) as:

$$\Xi_i = a\beta^2 \Bigg[[(E_i/\beta) - \alpha']^2 + \alpha' - \left(\left\{ \alpha'[(\alpha'+1) + (\alpha'-1)(E_i/\beta)] + \sum_{r=2}^{\alpha'-1} \frac{r(r-1)(E_i/\beta)^{\alpha'+1-r}}{(\alpha'+1-r)!} \right\} e^{-E_i/\beta} \right) \Bigg] \quad (1)$$

Where $E_i = e_i - w_i - \delta \sum_{j \in P_{ik}} d_{jj'} - \sum_{j \in P_{ik}} (s_j + w_j)$, $\alpha' = \alpha \sum_{j \in P_{ik}} d_{jj'}$, $\alpha \in \mathbb{Z}^+$, $\beta \in \mathbb{R}^+$, $\delta \in \mathbb{R}_0^+$, a is a weight coefficient, j' is the customer immediately following j in route k, P_{ik} is the set of customers served before i in route k.

On the other hand the tardiness penalty at customer i, Λ_i can be expressed as:

$$\Lambda_i = b\beta^2 \left[\alpha'[(\alpha'+1) + (\alpha'-1)(L_i/\beta)] + \sum_{r=2}^{\alpha'-1} \frac{r(r-1)(L_i/\beta)^{\alpha'+1-r}}{(\alpha'+1-r)!} \right] e^{-L_i/\beta} \quad (2)$$

Where $L_i = l_i - w_i - \delta \sum_{j \in P_{ik}} d_{jj'} - \sum_{j \in P_{ik}} (s_j + w_j)$, b is a weight coefficient.

The weight coefficients a in earliness and b in the tardiness penalty are assumed to be equal to one. The penalty for serving the customers outside the time windows is assumed to be a quadratic loss function, so larger deviations will cause progressively larger losses. If the service at customer i starts at time τ_i and $\tau_i < e_i$, the earliness penalty will be equal to $a(e_i - \tau_i)^2$. The lateness penalty would be equal to $b(\tau_i - l_i)^2$, in case that $\tau_i > l_i$. A discussion on other alternatives for penalty functions and their mathematical expressions can be found in Russell and Urban (2008).

The waiting time before each customer, including the depot, is a decision variable. So once a vehicle arrives to a customer, it can immediately serve the customer or it can wait. In Russell and Urban (2008) this decision is done in a postprocessing procedure, since the main algorithm assumes zero waiting time. The generalized reduced gradient method was used to deal with such variables.

2.2 Multi-objective stochastic optimization

Transportation planning is considered to be inherently multi-objective in nature (Current and Min, 1986). It can be expected that including additional relevant objectives into an the optimization problem will make it more realistic.

In a multi-objective optimization problem (MOP) several functions are optimized (minimized or maximized) subject to the same set of constraints. Without loss of generality the MOP can be treated as a minimization problem, in such case, it can be stated as

$$min\,\{F(x) = (f_1(x), f_2(x), \ldots, f_n(x))\} \tag{3}$$

$$subject \quad to$$

$$x \in D \tag{4}$$

with the number of objective functions being $n \geq 2$; the decision variable vector $x = (x_1, x_2, \ldots, x_r)$; the feasible solution space D; and $F(x)$ the objective vector (Jozefowiez et al., 2007).

In a MOP no single solution is able to minimize all objectives simultaneously, since it is expected to be a conflict among them. Instead of that, a solution to a MOP is given by a set of solutions, which are called tradeoff solutions (Collette and Siarry, 2003) or Pareto optimal solutions (Jozefowiez et al., 2007). A decision maker is not expected to implement all the solutions in the set, instead of that one of them must be selected according to particular policies or preferences.

The tradeoff solutions consist of the set of non-dominated solutions. A solution y with objective function values $(f_1(y), f_2(y), \ldots, f_n(y))$, dominates a solution z, $y \prec z$, if and only if $\forall\ i\ \{1, 2, \ldots, n\}\ f_i(y) \leq f_i(z)$, and $\exists\ j\ \{1, 2, \ldots, n\}$, such that $f_j(y) < f_j(z)$. That is, the solution z does not perform better than y in any objective functions, but it performs worse in at least one. The set of non-dominated solutions is called the Pareto set or the Pareto optimal solutions (Jozefowiez et al., 2007). If the last condition is not fulfilled, solution z does not perform worse than y in any of the objectives, then it is said that y weakly dominates solution z, $y \preceq z$ (Knowles, 2002).

The method or algorithm used for solving the MOP may not guarantee a set of non-dominated solutions as result. Then the obtained solutions are not Pareto optimal solutions. If this is the case and the algorithm used to solve the MOP does not find a solution z that dominates a solution y, then the latter is considered a *potentially* Pareto optimal solution, relative to the particular algorithm or method that was used to solve the problem (Jozefowiez et al., 2007).

When dealing with stochastic MOP the dominance may be evaluated in different ways (Caballero et al., 2004), here the expected values of the objective functions will be used to compared.

2.3 Literature review

There are several variants of the SVRP. One of such variants is modeled using the times as stochastic parameters. It can either be the travel times, service times or both. In Li et al. (2010) a CVRP with soft time windows and stochastic travel and service times was solved by means of a tabu search algorithm. A variant of the previous problem, without stochastic travel times and no time windows, was presented in Lei et al. (2012), a generalized variable neighborhood search (GVNS) was proposed to solve it. In Zhang et al. (2012), the CVRP with stochastic travel time and simultaneous pick-ups and deliveries was solved using a scatter search heuristic. A variant of the last problem including soft time windows was presented in Taş et al. (2013), a tabu search heuristic was used to solve it. The same problem can be found in Taş et al. (2014), but in this case it was solved by means of a branch-cut-and-price algorithm. In Kenyon and Morton (2003) a VRP with stochastic travel and service time is described, a method that solves the deterministic equivalent

of the stochastic problem was proposed to solve the VRP with stochastic travel times. A robust CVRP with deadlines and travel time/demand uncertainty is studied in Lee et al. (2012) and solved using a branch-and-price algorithm. For a more detailed summary on different SVRP, the reader is referred to Gendreau et al. (1996, 2014) and Oyola et al. (2015).

There is scant literature on multi-objective SVRP, to the best of our knowledge examples can be found in Tan et al. (2007); Juan et al. (2011); Ahmadi-Javid and Seddighi (2013) and Russell and Urban (2008). A multi-objective approach of the CVRP with stochastic demands (CVRPSD) was formulated in Tan et al. (2007). Three main objectives are minimized: the total travel time, the number of vehicles and drivers remuneration. An evolutionary algorithm was used to deal with that problem. In Juan et al. (2011) the CVRPSD is not explicitly presented as a bi-objective problem, but a tradeoff between the total expected cost and the probability of the solution suffering a route failure (reliability) is taken into consideration. This problem was solved as a single-objective problem using local search. An extension of the CVRP including location, allocation and routing under the risk of disruption is introduced in Ahmadi-Javid and Seddighi (2013). Although the problem is not entirely treated as a multi-objective, the decision maker is presented with three different solutions: one obtained by minimizing the expected cost, the second one by minimizing the conditional value-at-risk and a third solution is obtained by considering the the worst case value for the stochastic parameters. A local search heuristic was proposed for solving the problem. A multi-objective CVRP with soft time windows and stochastic travel times (SCVRPSTW) is found in Russell and Urban (2008). A tabu search heuristic is used to find solutions to the problem.

3 Algorithm

Evolutionary algorithms (EA) have been used for dealing with different types of VRP. In Jozefowiez et al. (2009) an EA was used for solving a multi-objective CVRP. Memetic algorithms have been used to deal with different versions of the stochastic VRP (Sörensen and Sevaux, 2009; Mendoza et al., 2010). In Tan et al. (2007) an EA was used to approximate the Pareto set of a multi-objective SVRP. This may be an indication of the potential of the EA for dealing with multi-objective VRPs and SVRPs.

The target aiming Pareto search (TAPaS) algorithm (Jozefowiez et al., 2007) takes an approximation PS_{appr} to a Pareto set and improves it by using a tabu search algorithm that looks into solutions in the areas that dominate PS_{appr}. The algorithm does not search in areas that do not dominate solutions in PS_{appr}, as such it does not look for new solutions with the best found value for any of the objectives (*extremal* solutions). Including such solutions in PS_{appr} would improve the quality of the set.

We propose an algorithm that follows the same principle as TAPaS where starting from the approximation obtained from a EA, a local search procedure is applied to improve the quality of the solution. An implementation of the NSGA-II (Deb et al., 2002), using particular features for the CVRP (Prins, 2004) will be used as EA. It is described in the Algorithm 3.1.

A local search that uses different neighborhood structures, as in generalized variable neighborhood search (Lei et al., 2012) is applied after the EA. Different neighborhoods are used to attempt to optimize the different objectives and will be applied sequentially. Since the evaluation of the performance of a solution in MOP as in Zitzler et al. (2001) and Mateo and Alberto (2012) can be computationally expensive, moves leading to solutions that dominate the incumbent solution are accepted. In Lei et al. (2012) the best improvement rule is applied, in this case the application of such rule can be complex and expensive, so the first improvement rule is applied here.

3.1 Construction of initial solutions

A set of initial solutions is required, so it can be later improved by an algorithm (evolutionary or other). We propose a simple deterministic construction heuristic. Most likely high quality solutions will not be obtained, but it can be one step ahead of the randomly constructed set. Our construction heuristic is based on the one used by Chiang and Russell (2004) who used a modified version of a classical insertion heuristic proposed by Solomon (1987).

An initial diverse set of VRP solutions is constructed with priority shifting from TW penalty to total length. This is achieved by solving a sequence of deterministic *hard* time-windows problems, where the original (soft) time windows are extended by a varying slack parameter while the distance is minimized as a single objective.

The number of routes m is given as a parameter, so the construction heuristic will not build one route at a time as in Solomon (1987). Any possible insertion in every possible route is going to be considered and evaluated as in Chiang and Russell (2004). After ordering the customers, using one of several available ordering rules, at every iteration one customer will be selected and inserted in the best possible way.

In Chiang and Russell (2004) the customers are inserted in a predefined order. Three rules are used for ordering the customers:

- Smallest early TW parameter e_i
- Tightness of the TW calculated as $100(l_i - e_i) - d_{0i}$
- Largest value for d_{0i}

A very important parameter is the late TW parameter l_i, which was not considered in Chiang and Russell (2004), except in the second rule as part of the TW. The second and the third rule use the distance instead of the time. Two other rules are suggested to be used in addition to the previous three:

- Smallest late TW parameter l_i. If the TW length is the same for all customers, this will not be different from the first rule. But it will lead to a different ordering if such length is different.
- Smallest ratio of late TW parameter to distance from depot $l_i/\bar{d_{0i}}$

We use all the five rules already mentioned and they are used independently from each other. If nothing else is changed, we could obtain at most five different solutions by inserting the customers in five different orders. The best insertion for a customer was originally given by minimizing a combination of the route distance and travel time increase in Solomon (1987). Here we do not prioritize the total schedule time, so that factor is not considered. The increment in the distance when customer k is inserted in route r between customers i and j is given by the equation

$$f(i,k,j) = d_{ik} + d_{kj} - d_{ij}$$

This value must be evaluated for every feasible route and every pair of customers already in the routes, where is feasible to insert customer k. If the route is empty the increment in the distance will be equal to two times d_{0k}. The route r where $f(i,k,j)$ is minimum must be selected together with i and j. In case no insertion with TW feasibility is found, the insertion must be done in a route

with capacity constraint feasibility. If no insertion with capacity constraint feasibility is available, the customer should be inserted in the route where the capacity constraint violation is minimized. This indicates that a repair mechanism may be required. Any infeasible solution is subject to simple moves as an attempt to make it feasible, if after a certain number of iterations there is no success, the solution is discharged.

A set of initial solutions with different priorities given to two different objective functions, total distance and TW penalty violations, is desired. If the TW are considered to be hard, then a higher priority is given the second objective function. On the other hand, if no TW are taken into consideration when building the solutions, the total distance is the one being prioritized. There is a time window at the depot, however there is no penalty associated to it.

During the construction phase, the set of solutions that represents a tradeoff between the penalty cost of TW violations and the total distance, is approximated by finding solutions to different versions of the problem, where for every customer the lower and upper limits in the TW are changed. A slack s varies from 0 to s_m and will change the TW of every customer i from (e_i, l_i) to $(e_i - s, l_i + s)$. The value of s_m could be defined in several ways, here we make it equal to half the time horizon (u_0). It is important to keep the TW within the time horizon, so for every value of s, the TW for customer i must be $(\max\{e_i - s, e_0\}, \min\{l_i + s, u_0\})$. Solutions are built using this modified TW as hard TW.

For each value of s, a different solution is built using each of the five ordering rules, previously described, for deciding the order in which customers must be inserted into the routes.

As a strategy for increasing the number of built solutions and exploring more areas of the search space, a randomization is included in the construction phase. In Mendoza et al. (2015) an integer randomization factor l_f is used. We use a similar approach, generating at every iteration a random number l greater than zero and less than or equal to the minimum value between l_f and the number of customers not inserted yet. The customer located at the lth position of the ordered list must be the one selected for insertion. The process is repeated until the number of built solutions is equal to the population size.

3.2 Evolutionary algorithm (EA)

A key element in NSGA-II is the ranking of solutions in the population, used in Algorithm 3.2. The $rank$ depends on the quality of the solution. Non-dominated solutions will have $rank$ 0. Solutions dominated only by solutions with $rank$ 0 will have $rank$ 1. In general solutions dominating the solutions with rank i will have a rank from 0 to $i - 1$.

The implemented EA will use two crossover operators *Split* (Prins, 2004) and *RBX* (Potvin and Bengio, 1996). In the original implementation of TAPaS, the same operators were used. Both operators may lead to good solutions in different ways. The *Split* procedure can lead to good sequences of customers with respect to the total distance. On the other hand, the *RBX* keeps routes from the parent solutions, if such routes have a good performance regarding the objective functions, the offspring solutions can be benefited.

The mutation operator included in Algorithm 3.4 will be Or-opt as in Jozefowiez et al. (2007) and Jozefowiez et al. (2009). The move that dominates all the others is accepted. The comparison is done among the neighbors, without considering the initial solution. If no move dominates all the others, among the non-dominated moves, the selection will be done comparing the normalized values of the objective functions.

A 2-opt procedure is included as local search (ls) in Algorithm 3.4. Such procedure has been used before (Jozefowiez et al., 2007, 2009) but here, we can keep in mind that we deal with two objective functions and include the domination criteria. Given two possible moves, the first lead to solution y and the second to solution x, if they are non dominated, $\exists A \subset \{1, 2, \ldots, n\}\ \forall i \in A f_i(y) < f_i(x)$, and $\exists B \subset \{1, 2, \ldots, n\}\ \forall j \in B f_j(y) > f_j(x)$, y will be preferred if $\sum_{i \in A}(1 - f_i(y)/f_i(x)) > \sum_{j \in B}(f_j(y)/f_j(x) - 1)$. If there is no inequality, one solution is selected randomly.

An important feature of the NSGA-II is the *crowded-comparison operator* (Deb et al., 2002), which computes the crowding distance see Algorithm 3.3. A solution x is preferred to a solution y, if the rank of x is lower than the rank of y. In case that both x and y have the same rank, the solution with the greater crowding distance is preferred.

Algorithm 3.1: MAIN LOOP($f_1, f_2, \ldots, f_m$: objective functions)

Let N be the size of the population
Let P_t be the population at generation t
$t \leftarrow 0$
$P_t \leftarrow constructiveHeuristic()$
$sortPopulation(P_t)$
while $t < maxGeneration$
do $\begin{cases} P_{t+1} \leftarrow P_t \cup recombination(P_t) \\ P_{t+1} \leftarrow sortPopulation(P_{t+1}) \\ t \leftarrow t + 1 \end{cases}$
return (P_t)

Algorithm 3.2: SORTPOPULATION(P)

Let R_k be the set of solutions with rank k
Let Q be a set of solutions
$\mathcal{Q} \leftarrow \emptyset$
$rankPopulation(P)$
$i \leftarrow 0$
while $|Q| + R_i \leq N$
do $\begin{cases} crowdDistance(R_i) \\ Q \leftarrow Q \cup R_i \\ i \leftarrow i + 1 \end{cases}$
if $|Q| < N$
then $\begin{cases} sortRank(R_i) \\ Q \leftarrow Q \cup \{S \subseteq R_i / |S| = N - |Q|\} \end{cases}$
return $(\mathcal{Q})$

Algorithm 3.3: CROWDDISTANCE(R)

Let d_i be the distance measure of solution i
Let s_i be the solution in position i after sorting
$n \leftarrow |R_i|$
for $i \leftarrow 0$ **to** n
  do $\{d_i \leftarrow 0$
for $i \leftarrow 0$ **to** m
  do
$R_i \leftarrow sort(R_i, f_m)$
$d_0 \leftarrow \infty$
$d_{|R_i|-1} \leftarrow \infty$
for $i \leftarrow 1$ **to** $|R_i| - 2$
  do $\{d_i \leftarrow d_i + (f_m(s_{(i+1)}) - f_m(s_{(i-1)})/(f_m^{max} - f_m^{min})$

Algorithm 3.4: RECOMBINATION(P)

for $i \leftarrow 0$ **to** $N/2 - 1$
  do
$p_1 \leftarrow randomSolution(P)$
$p \leftarrow randomSolution(P)$
if $p \prec_c p_1$
  then $\{p_1 \leftarrow p$
$p_2 \leftarrow randomSolution(P)$
$p \leftarrow randomSolution(P)$
if $p \prec_c p_2$
  then $\{p_2 \leftarrow p$
if $rand() < prob_{co}$
  then $\{crossover \leftarrow RBX$
  else $\{crossover \leftarrow SPLIT$
$Q \leftarrow Q \cup crossover(p_1, p_2)$
$Q \leftarrow Q \cup crossover(p_2, p_1)$
if $rand() < prob_{mt}$
  then $\{mutation(Q[2i])$
if $rand() < prob_{mt}$
  then $\{mutation(Q[2i + 1])$
$ls(Q[2i])$
$ls(Q[2i + 1])$

3.3 Local search

Once the EA has been executed, every found potentially non-dominated solution is subject to a local search procedure, following a principle similar to the GVNS in Lei et al. (2012). In the local search just moves that lead to a solution that dominates the incumbent solution are accepted. In Lei et al. (2012) the record-to-record travel (RRT) (Dueck, 1993) is used as accepting criteria.

It is suggested that otherwise the GVNS yields a local optimum. In our case the search is performed as an improving mechanism, where is expected that the solutions already have a good quality and perhaps any extra improvement should be accepted. That is the reason why the RRT is not part of the local search. The process is described in Algorithm 3.5.

Algorithm 3.5: LOCALSEARCH(s_{EA} : a solution from EA)

Let Q be a set of solutions
$s^* \leftarrow s_{EA}$
$s \leftarrow s_{EA}$
while $stoppingCriteria$
do
- $s' \leftarrow shaking(s)$
- $s'' \leftarrow VNS(s')$
- **if** $s^* \not\prec s''$
  - **then** $\{Q \leftarrow Q \cup s''$
- **if** $s'' \prec s$
  - **then** $\{s \leftarrow s''$

return (Q)

The local search (VNS) performs the search using different neighborhood structures. As in Lei et al. (2012) there are inter- and intra-route operations. However the number of neighborhood structures used in the local search is lower, since the computational requirements are higher. For accepting the moves, the first improvement rule is applied. It assumed that a move will improve the incumbent solution if it leads to a neighbor solution that dominates it or to a solution that does not dominate it, but the improvement in one of the objectives is greater than the deterioration in the other one, as percentage of the current values. A random inter-route swap move is used as a *shaking* operator.

The different neighborhoods are:

- Intra-route insertion move.
- Intra-route swap moves.
- Inter-route swap moves.
- An Or-opt procedure.

3.4 Post-optimization

The waiting time at every node, including the depot, is a decision variable. Departure from the depot can be postponed as well as the beginning of the service at every customer. In Russell and Urban (2008) this decision is done in a postprocessing procedure, since the main algorithm assumes zero waiting time at customers, not at the depot. Waiting times at the depot are computed based on the expected travel times to the first customer in the routes. If the expected travel time from the depot to the next customer is shorter than the early TW parameter associated to such customer, the waiting time at the depot will be equal to the difference, zero otherwise. The postprocessing procedure may change that value. The generalized reduced gradient method is used to deal with such variables in Russell and Urban (2008). In our approach the decisions regarding waiting times are also part of a postprocessing procedure. We rely on the quasi-Newton

Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Venkataraman, 2009) in Scilab for doing such procedure, since we find very convenient to access it from C++ and it is an open source package.

3.5 Solution evaluation

The equations used by Russell and Urban (2008) to compute the exact value of the penalties and previously described in Section 2.1, require massive computation, since they deal with transcendental functions and operations with large integers. Such operations can eventually lead to an integer overflow. As an alternative to avoid such condition, natural logarithm transformations are used, which increases the computational requirements. An evaluation strategy is proposed looking to overcome such difficulty. Following the spirit of Mendoza et al. (2015), the evaluation of the solutions is done in different ways along the search process. Let us say that the EA has a maximum number of generations GEN. This total number of generations is divided into three parts from zero to gen_1, from gen_1 to gen_2 and from gen_2 to GEN. During the execution of the algorithm, while running within the first part of the generations, the solutions and the moves are evaluated using deterministic values, and the travel time from a customer i to a customer j is given by the expected value $(\alpha \cdot \beta + \delta)d_{ij}$. In the second part of the generations the solutions and moves are evaluated using sample scenarios. And only in the third part of the generations, the exact value for the expected penalties is computed using the closed-form in equations 1 and 2. Lookup tables were implemented for some of the computations, as a mechanism to reduce the processing time.

Using scenarios for evaluating the solutions could have an additional advantage in a different context. If the expected values of the objective functions are unknown, this could be approximated with scenario evaluation. In these cases an additional stability analysis must be carried out (Kaut and Wallace, 2007). The values would not be exact, but the algorithm becomes independent of the probability distributions of the stochastic parameters. This is out of the scope of this work, however, it is an aspect worth of comment.

4 Computational experiments

4.1 Test environment

The same instances as in Russell and Urban (2008) are used, these are a modified version of Solomon instances (Solomon, 1987). Four base test problems are used, R101, R102, R103 and R109. Four different sets of parameters for the travel time probability distribution (α, β and δ) are used with each instance, (1.00, 0.25, 0.75), (1.00, 0.50, 0.50), (1.00, 0.75, 0.25) and (1.00, 1.00, 1.00), identified as S1, S2, S3 and S4 respectively. The number of vehicles is also predefined and is set to be the same as in Russell and Urban (2008), which means that there may be two versions of the same instance, with a different number of vehicles available. As an illustration, we can say that the instance R101-S3-18V, corresponds to the instance R101, where the parameters of the travel time probability distribution are given by the set (1.00, 0.75, 0.25) and there are 18 vehicles available. In total there are 27 instances for computational experiments. Ten different runs per instance were performed.

The parameters of the algorithm were tuned by means of preliminary testing. Unless stated otherwise, the values given to these parameters are:

- Number of generations for NSGA, $GEN = 300$.
- Population size for NSGA, 150

- Probability of applying *RBX* crossover operator, $prob_{co} = 0.5$
- Probability of mutation, $prob_{mt} = 0.4$
- Number of iterations for VNS, 100.
- Randomization factor of construction procedure, $l_f = 5$.
- Number of generations evaluated deterministically, $GEN_1 = GEN \cdot 0.5$.
- Number of generations evaluated by scenarios, $GEN_2 - GEN_1 = GEN \cdot 0.25$
- Number of scenarios, 20.
- Maximum number of consecutive customers to move in the Or-opt procedure, 3.

All computational experiments were conducted on a computer with processor Intel (R) Xeon (R) CPU E31270 @ 3.40 GHz and 16.0 GB of RAM.

4.2 Results

Our approach is able to find solutions that dominate the solutions reported in Russell and Urban (2008). In 14 out of 27 instances such solutions are found even without applying the postprocessing procedure, as it is shown in Table 1 in Appendix A. Once the postprocessing procedure is applied, solutions dominating the ones in Russell and Urban (2008) are found in 18 out of 27 instances, however in one of the instances, R109-S3-12V, just one of the two solutions is dominated. Results are presented in Table 2. We emphasize that the NSGA&VNS solutions in tables 1 and 2 do not correspond to the approximation of the Pareto set, these solutions are only a subset of the solutions that NSGA&VNS is able to find.

It was observed that, in general, not all the vehicles are used in solutions where a higher priority is given to minimize the total length (in the Pareto set approximation). Which means that they require a lower number of vehicles than the solutions reported in Russell and Urban (2008). On the other hand, the value of the TW penalty objective is higher, therefore the reduction in the number of vehicles does not lead to Pareto dominance.

The instance R103-S1-14V is an example of the instances where solutions dominating the solution reported in Russell and Urban (2008) were not found. Figure 1 shows a section of the Pareto set approximation including both, our approximation and the solution in Russell and Urban (2008). On the other hand, the Figure 2 shows a section of the Pareto set approximation for the instance R101-S2-18V, where it is possible to see that the solution in Russell and Urban (2008) is dominated by solutions in our approximation.

The running time of our approach is much longer than in Russell and Urban (2008), as it can be seen in Table 3. This can be explained by the fact that while in Russell and Urban (2008) one solution, at most two, are found per instance, we find an approximation to the Pareto set.

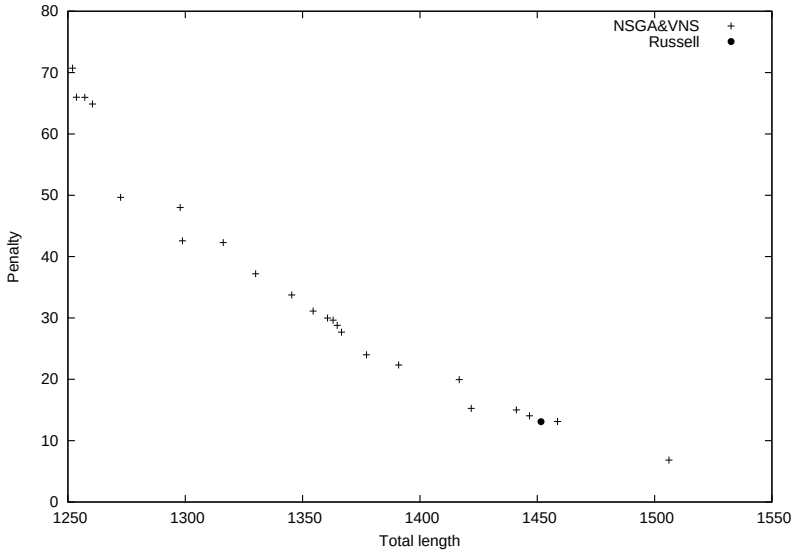

Figure 1: Our approach finds solutions that do not dominate solution found by Russell and Urban (2008) in instance R103-S1-14V

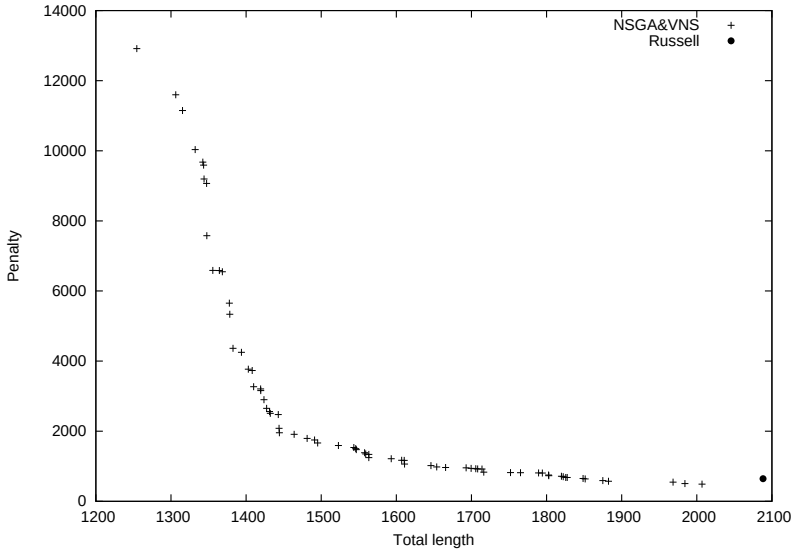

Figure 2: Our approach finds solutions that dominate solution found by Russell and Urban (2008) in instance R101-S2-18V

As expected, the quality of the Pareto set approximation is improved by the postprocessing procedure. Using the average value of the $\mathcal{S}$ metric (Jozefowiez et al., 2009, 2007; Knowles, 2002; Mateo and Alberto, 2012) to compare the two approximations to the Pareto set, before (nPost) and after (Post) the postprocessing procedure, we found that the average improvement over all instances is 0.96%. Russell and Urban (2008) reports average improvement above 20%, but such improvement is measured over the objective function value, in our case we measure the improvement of the Pareto set. This comparison is shown in Table 4. For readability purposes the values have been standardized, dividing the actual value of the $\mathcal{S}$ metric by the area of the rectangle defined by (0,0) and the reference point. The minimum, maximum and average values are calculated over the ten different runs of the algorithm.

4.3 Impact of NSGA and VNS procedure on the results

A second set of experiments was conducted looking to asses the impact of the NSGA and the VNS procedure in the quality of the obtained solutions. All instances were used in these experiments and ten runs per instance were performed. Three different configurations of the algorithm were compared: the original algorithm described in Section 3 (NSGA&VNS) and two algorithms, each consisting in one of the main components of the previous, NSGA and VNS procedure. The running time of the NSGA&VNS was reduced by setting the number of generations for the NSGA component to 30 and the iterations for the VNS to 10. The number of generations of the algorithm consisting of just the NSGA was set to 140, so its running time becomes not shorter than the one used by NSGA&VNS. For the case of the tests using the VNS procedure, the number of iterations was set to 120.

Results were compared using the $\mathcal{S}$ and the $\mathcal{C}$ metric. It is worth remarking that given two sets of solutions $(\mathcal{R}, \mathcal{X})$, the $\mathcal{C}$ metric measures the ratio of solutions in $\mathcal{X}$ weakly dominated by solutions in $\mathcal{R}$. The comparisons of the three configurations of the algorithm (NSGA&VNS, NSGA and VNS) are presented in the tables 5, 6 and 7.

NSGA&VNS and NSGA are compared using the data presented in tables 5 and 6. Using the average value of the metrics, NSGA&VNS performs better in 15 out of 27 cases. In six instances is not possible to say which configuration has a better performance since the two metrics are contradictory. NSGA&VNS and VNS, on the other hand, are compared using the data in tables 5 and 7. NSGA&VNS performs better in 17 out of 27 instances. In eight cases the two metrics lead to contradictory conclusions.

It was observed that VNS is able to find extremal solutions, that do not necessary dominate the solutions found by the other two configurations, but in some cases are good enough to dominate the solutions found in Russell and Urban (2008). For example, VNS was able to find a solution to the instance R101-S3-18V with a total length of 2077.33 and a penalty equal to 1475.20. However, the number of found solutions is limited, in general. This has a negative impact when the set of solutions is evaluated, specially when evaluating using the $\mathcal{S}$ metric. The NSGA provides a denser set of solutions, which is reflected positively in the $\mathcal{S}$ metric. In the NSGA&VNS the solutions found by NSGA are used by VNS afterwards. The latter is able to improve the solution set by either finding dominating solutions or improving the extremal solutions, outperforming the solutions sets found by NSGA and VNS when executed individually. In conclusion, these experiments indicate that the combination NSGA&VNS works better than each individual method.

5 Conclusions and further research

The Pareto solutions to a known multi-objective SVRP were approximated for a first time. Obtained results were compared to previously reported individual potentially Pareto solutions. In most of the tested instances, our approach is able to find solutions that dominate the existing solutions, in addition to a wide range of solutions where priority shifts from one objective to the other.

A relationship between the number of vehicles and the total distance was observed. For the reason that, in general, the solutions in the Pareto set approximation that minimize the total length do not use all the available vehicles. So we were able to find solutions that require less vehicles than the previously reported solutions. However, it is not possible to say that our solutions are better, since the TW penalties become higher.

Our approach provides good results in most of the tested instances, but still there is room for improvement and several directions can be applied for further research. The efficacy of our approach is decreased when dealing with problems that present a high coefficient of variation. A different construction of initial solutions or local search operators could be adapted to deal with this particular type of instances. A different aspect to consider is the running time, the closed-form expressions for computing the expected value of the TW penalties require a large number of algebraic operations. Perhaps such expressions could be approximated using a method more efficient than the scenarios, but less demanding than the closed-form. Taking waiting times into consideration during the execution of the main algorithm is likely to improve the solutions. This, however, will increase the complexity of the problem. Further work can be done dealing with such complexity and testing the impact on results.

References

Ahmadi-Javid, A. and Seddighi, A. H. (2013). A location-routing problem with disruption risk. *Transportation Research Part E: Logistics and Transportation Review*, 53:63 – 82.

Caballero, R., Cerd, E., del Mar Muoz, M., and Rey, L. (2004). Stochastic approach versus multiobjective approach for obtaining efficient solutions in stochastic multiobjective programming problems. *European Journal of Operational Research*, 158(3):633–648.

Chiang, W.-C. and Russell, R. A. (2004). A metaheuristic for the vehicle-routeing problem with soft time windows. *The Journal of the Operational Research Society*, 55(12):1298–1310.

Collette, Y. and Siarry, P. (2003). *Multiobjective optimization: principles and case studies*. Springer, Berlin.

Cordeau, J.-F., Laporte, G., Savelsbergh, M. W., and Vigo, D. (2007). Chapter 6. Vehicle routing. In Barnhart, C. and Laporte, G., editors, *Transportation*, volume 14, pages 367–428. Elsevier.

Current, J. and Min, H. (1986). Multiobjective design of transportation networks: Taxonomy and annotation. *European Journal of Operational Research*, 26(2):187 – 201.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1):80–91.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Trans. Evol. Comp*, 6(2):182–197.

Dueck, G. (1993). New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, 104(1):86–92.

Gendreau, M., Jabali, O., and Rei, W. (2014). Stochastic vehicle routing problems. In Toth, P. and Vigo, D., editors, *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 213–239. Society for Industrial and Applied Mathematics.

Gendreau, M., Laporte, G., and Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12.

Jozefowiez, N., Semet, F., and Talbi, E. (2009). An evolutionary algorithm for the vehicle routing problem with route balancing. *European Journal of Operational Research*, 195(3):761–769.

Jozefowiez, N., Semet, F., and Talbi, E.-G. (2007). Target aiming pareto search and its application to the vehicle routing problem with route balancing. *Journal of Heuristics*, 13(5):455–469.

Juan, A., Faulin, J., Grasman, S., Riera, D., Marull, J., and Mendez, C. (2011). Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands. *Transportation Research Part C: Emerging Technologies*, 19(5):751–765. Freight Transportation and Logistics (selected papers from {ODYSSEUS} 2009 - the 4th International Workshop on Freight Transportation and Logistics).

Kaut, M. and Wallace, S. W. (2007). Evaluation of scenario-generation methods for stochastic programming. *Pacific Journal of Optimization*, 3(2):257–271.

Kenyon, A. S. and Morton, D. P. (2003). Stochastic vehicle routing with random travel times. *Transportation Science*, 37(1):69–82.

Knowles, J. D. (2002). *Local-search and hybrid evolutionary algorithms for Pareto optimization*. PhD thesis, University of Reading.

Lee, C., Lee, K., and Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *The Journal of the Operational Research Society*, 63(9):1294–1306.

Lei, H., Laporte, G., and Guo, B. (2012). A generalized variable neighborhood search heuristic for the capacitated vehicle routing problem with stochastic service times. *TOP*, 20(3):99 – 118.

Li, X., Tian, P., and Leung, S. C. (2010). Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics*, 125(1):137–145.

Mateo, P. and Alberto, I. (2012). A mutation operator based on a pareto ranking for multi-objective evolutionary algorithms. *Journal of Heuristics*, 18:53–89.

Mendoza, J., Rousseau, L.-M., and Villegas, J. (2015). A hybrid metaheuristic for the vehicle routing problem with stochastic demand and duration constraints. *Journal of Heuristics*, pages 1–28.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898.

Oyola, J., Arntzen, H., and Woodruff, D. W. (2015). The stochastic vehicle routing problem, a literature review. *Optimization Online*. `http://www.optimization-online.org/DB_FILE/2016/01/5299.pdf/`.

Potvin, J.-Y. and Bengio, S. (1996). The vehicle routing problem with time windows part ii: Genetic search. *INFORMS Journal on Computing*, 8(2):165–172.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002.

Russell, R. A. and Urban, T. L. (2008). Vehicle routing with soft time windows and erlang travel times. *The Journal of the Operational Research Society*, 59(9):1220–1228.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.

Sörensen, K. and Sevaux, M. (2009). A practical approach for robust and flexible vehicle routing using metaheuristics and monte carlo sampling. *Journal of Mathematical Modelling and Algorithms*, 8(4):387–407.

Taş, D., Dellaert, N., van Woensel, T., and de Kok, T. (2013). Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research*, 40(1):214 – 224.

Taş, D., Gendreau, M., Dellaert, N., van Woensel, T., and de Kok, A. (2014). 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.

Tan, K., Cheong, C., and Goh, C. (2007). Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813–839.

Toth, P. and Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1-3):487 – 512.

Venkataraman, P. (2009). *Applied Optimization with MATLAB Programming*. Wiley.

Zhang, T., Chaovalitwongse, W., and Zhang, Y. (2012). Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. *Computers & Operations Research*, 39(10):2277 – 2290.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). SPEA2: Improving the strength pareto evolutionary algorithm. Technical report, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institue of technology (ETH).

Appendices

A Tables

Table 1: Solutions without postprocessing that dominate solutions reported in Russell and Urban (2008)

Instance	Solutions found in Russell and Urban (2008)		Solutions found by NSGA & VNS	
	Total lenght	TW penalty	Total lenght	TW penalty
R101-S1-17V	1 806.24	667.44	1 772.79	664.12
			1 788.67	633.48
			1 801.59	545.35
R101-S1-18V	2 104.32	212.90	2 057.34	201.37
R101-S3-16V	1 820.28	2 255.89	1 689.51	2 253.31
			1 720.20	2 142.24
			1 791.45	1 931.78
R101-S3-18V	2 164.59	1 910.68	1 770.14	1 906.61
			1 872.91	1 828.20
			2 059.22	1 436.09
R102-S1-16V	1 585.93	225.82	1 578.22	222.33
			1 578.93	211.08
			1 583.71	159.26
R102-S1-17V	1 792.34	161.54	1 665.60	160.26
			1 705.69	110.07
			1 788.42	66.27
R102-S2-15V	1 695.51	977.96	1 448.89	976.50
			1 614.74	477.07
			1 689.35	373.04
R102-S2-17V	1 826.80	308.07	1 733.08	301.01
			1 739.26	287.01
R102-S3-16V	1 661.68	1 138.40	1 554.79	1 136.80
			1 603.88	992.68
			1 649.25	864.60
R102-S3-17V	1 871.34	920.66	1 633.15	919.87
			1 676.07	867.07
			1 683.94	817.08
R103-S1-12V	1 465.64	375.41	1 190.59	351.56
			1 269.30	108.22
			1 432.73	63.64
R103-S2-12V	1 398.88	671.46	1 193.47	664.01
			1 222.80	576.53
			1 389.65	239.10
R103-S3-13V	1 372.24	689.17	1 327.54	544.71
	1 462.45	556.55	1 352.48	531.87
			1 354.34	493.18
R109-S1-12V	1 215.22	11.54	1 211.98	6.95
	1 216.62	6.25	1 213.38	6.02

Table 2: Solutions after postprocessing that dominate solutions reported in Russell and Urban (2008)

Instance	Solutions found in Russell and Urban (2008)		Solutions found by NSGA & VNS	
	Total lenght	TW penalty	Total lenght	TW penalty
R101-S1-17V	1 806.24	667.44	1 598.32	653.79
			1 718.13	630.79
			1 786.99	354.26
R101-S1-18V	2 104.32	212.90	1 947.11	208.07
			1 996.94	190.44
			2 046.90	143.67

Table 2 Continued: Solutions after postprocessing that dominate solutions reported in Russell and Urban (2008)

Instance	Solutions found in Russell and Urban (2008)		Solutions found by NSGA & VNS	
	Total lenght	TW penalty	Total lenght	TW penalty
R101-S2-18V	2 088.24	646.84	1 851.43	639.15
			1 874.96	592.95
			2 006.97	491.64
R101-S3-16V	1 820.28	2 255.89	1 583.34	2 246.78
			1 705.52	1 814.89
			1 782.88	1 650.55
R101-S3-18V	2 164.59	1 910.68	1 717.20	1 859.80
			1 676.35	1 615.93
			1 948.05	1 272.16
R102-S1-16V	1 585.93	225.82	1 520.03	207.48
			1 568.09	158.72
			1 583.70	131.88
R102-S1-17V	1 792.34	161.54	1 557.97	156.34
			1 647.29	108.10
			1 788.42	47.33
R102-S2-15V	1 695.51	977.96	1 387.17	955.11
			1 544.09	476.55
			1 689.35	336.09
R102-S2-17V	1 826.80	308.07	1 690.62	299.84
			1 737.33	284.80
			1 825.78	250.39
R102-S3-16V	1 661.68	1 138.40	1 484.02	1 129.57
			1 522.54	972.05
			1 648.77	764.54
R102-S3-17V	1 871.34	920.66	1 573.11	911.66
			1 596.70	863.65
			1 683.94	743.77
R103-S1-12V	1 465.64	375.41	1 190.59	323.63
			1 254.98	104.59
			1 406.24	55.15
R103-S2-12V	1 398.88	671.46	1 197.69	617.22
			1 208.97	565.55
			1 389.65	211.84
R103-S2-14V	1 466.74	142.02	1 385.06	141.56
			1 450.25	132.55
			1 459.81	122.86
R103-S3-13V	1 372.24	689.17	1 292.44	542.00
	1 462.45	556.55	1 296.76	530.46
			1 394.20	431.25
R109-S1-12V	1 215.22	11.54	1 195.56	5.51
	1 216.62	6.25	1 203.26	4.83
			1 213.38	2.29
R109-S2-12V	1 216.82	42.95	1 215.84	42.49
	1 224.31	28.36		
R109-S3-12V	1 219.31	138.23	1 230.62	93.34
	1 230.82	110.43		

Table 3: Running time (in minutes)

Instance	Running time in Russell and Urban (2008)	Time algorithm NSGA & VNS	Time postprocessing (Scilab)	Total time
R101-S1-17V	22.16	2 082.59	13.33	2 095.92
R101-S1-18V	32.03	2 151.56	13.74	2 165.31
R101-S2-17V	87.71	2 918.84	17.07	2 935.91
R101-S2-18V	98.17	3 084.58	18.39	3 102.98
R101-S3-16V	226.78	6 812.99	14.94	6 827.92
R101-S3-18V	245.24	6 762.15	14.59	6 776.74
R101-S4-17V	312.11	2 558.41	8.27	2 566.68

Table 3 Continued: Running time (in minutes)

Instance	Running time in Russell and Urban (2008)	Time algorithm NSGA & VNS	Time postprocessing (Scilab)	Total time
R101-S4-20V	234.79	2 698.06	7.90	2 705.96
R102-S1-16V	34.05	3 251.46	14.43	3 265.88
R102-S1-17V	40.19	3 363.84	13.67	3 377.51
R102-S2-15V	114.19	5 186.12	15.41	5 201.53
R102-S2-17V	127.70	5 676.19	15.18	5 691.37
R102-S3-16V	261.21	7 857.41	14.99	7 872.40
R102-S3-17V	257.76	7 588.28	12.99	7 601.28
R102-S4-16V	351.45	3 275.37	4.77	3 280.14
R102-S4-17V	318.90	3 305.82	5.04	3 310.86
R103-S1-12V	22.43	3 498.03	8.46	3 506.49
R103-S1-14V	48.43	3 913.18	9.41	3 922.59
R103-S2-12V	150.33	4 936.02	9.19	4 945.21
R103-S2-14V	173.30	5 135.72	9.48	5 145.21
R103-S3-13V	657.13[a]	6 268.29	7.42	6 275.71
R103-S4-12V	398.44	3 012.61	1.88	3 014.48
R103-S4-13V	392.86	3 076.58	2.94	3 079.51
R109-S1-12V	45.43[a]	2 396.62	6.82	2 403.43
R109-S2-12V	229.06[a]	3 722.81	7.81	3 730.62
R109-S3-12V	698.97[a]	4 723.14	8.96	4 732.10
R109-S4-12V	977.02[a]	2 272.49	1.49	2 273.78

[a] The algorithm finds two independent solutions. Time computed by adding both running times

Table 4: $\mathcal{S}$ metric performance indicator for nPost and Post

Instance	Ref. Point (Length, TW penalty)	Algorithm	Average	Min	Max
R101-S1-17V	(2 098.66 , 23 357.90)	nPost	0.448	0.445	0.453
		Post	0.456	0.452	0.461
R101-S1-18V	(2 117.43 , 25 003.90)	nPost	0.460	0.453	0.465
		Post	0.467	0.458	0.472
R101-S2-17V	(2 022.89 , 86 728.50)	nPost	0.445	0.441	0.453
		Post	0.451	0.446	0.458
R101-S2-18V	(2 116.92 , 90 445.00)	nPost	0.474	0.470	0.479
		Post	0.479	0.474	0.484
R101-S3-16V	(1 916.46 , 100 812.00)	nPost	0.354	0.332	0.373
		Post	0.360	0.336	0.381
R101-S3-18V	(2 059.22 , 78 918.00)	nPost	0.382	0.364	0.393
		Post	0.390	0.369	0.401
R101-S4-17V	(1 523.06 , 335 287.00)	nPost	0.260	0.255	0.269
		Post	0.265	0.260	0.275
R101-S4-20V	(1 676.17 , 314 186.00)	nPost	0.311	0.304	0.314
		Post	0.316	0.309	0.320
R102-S1-16V	(1 886.55 , 20 227.90)	nPost	0.413	0.404	0.423
		Post	0.416	0.406	0.426
R102-S1-17V	(1 992.93 , 23 596.60)	nPost	0.452	0.446	0.460
		Post	0.454	0.448	0.462
R102-S2-15V	(1 694.51 , 50 885.10)	nPost	0.323	0.314	0.334
		Post	0.326	0.316	0.336
R102-S2-17V	(1 885.99 , 56 088.30)	nPost	0.390	0.376	0.398
		Post	0.392	0.378	0.401

Table 4 Continued: $\mathcal{S}$ metric performance indicator for nPost and Post

Instance	Ref. Point (Length, TW penalty)	Algorithm	Average	Min	Max
R102-S3-16V	(1 789.06 , 129 380.00)	nPost	0.350	0.336	0.358
		Post	0.351	0.337	0.360
R102-S3-17V	(1 771.22 , 100 799.00)	nPost	0.335	0.324	0.356
		Post	0.337	0.325	0.360
R102-S4-16V	(1 486.58 , 323 726.00)	nPost	0.235	0.226	0.243
		Post	0.237	0.228	0.245
R102-S4-17V	(1 556.20 , 265 788.00)	nPost	0.243	0.230	0.255
		Post	0.245	0.233	0.258
R103-S1-12V	(1 466.93 , 16 508.20)	nPost	0.274	0.267	0.291
		Post	0.276	0.268	0.292
R103-S1-14V	(1 647.80 , 15 195.00)	nPost	0.344	0.332	0.359
		Post	0.345	0.333	0.361
R103-S2-12V	(1 421.61 , 66 221.60)	nPost	0.258	0.249	0.269
		Post	0.259	0.250	0.269
R103-S2-14V	(1 554.23 , 40 761.40)	nPost	0.305	0.287	0.321
		Post	0.306	0.287	0.322
R103-S3-13V	(1 419.83 , 125 422.00)	nPost	0.242	0.224	0.260
		Post	0.243	0.225	0.261
R103-S4-12V	(1 161.62 , 372 838.00)	nPost	0.126	0.120	0.135
		Post	0.127	0.121	0.137
R103-S4-13V	(1 235.24 , 269 971.00)	nPost	0.150	0.143	0.158
		Post	0.151	0.144	0.160
R109-S1-12V	(1 410.32 , 21 858.30)	nPost	0.266	0.257	0.271
		Post	0.268	0.259	0.274
R109-S2-12V	(1 342.50 , 72 944.90)	nPost	0.203	0.189	0.212
		Post	0.205	0.190	0.213
R109-S3-12V	(1 390.82 , 135 270.00)	nPost	0.233	0.221	0.248
		Post	0.236	0.223	0.251
R109-S4-12V	(1 198.65 , 353 675.00)	nPost	0.137	0.129	0.150
		Post	0.138	0.130	0.153

Table 5: $\mathcal{S}$ metric performance indicator for NSGA&VNS, NSGA and VNS

Instance	Ref. Point (Length, TW penalty)	Algorithm	Average	Min	Max
R101-S1-17V	(2 117.95 , 56 067.60)	NSGA&VNS	0.377	0.356	0.410
		NSGA	0.378	0.358	0.406
		VNS	0.305	0.270	0.327
R101-S1-18V	(2 171.52 , 56 067.60)	NSGA&VNS	0.395	0.387	0.407
		NSGA	0.390	0.366	0.404
		VNS	0.309	0.269	0.372
R101-S2-17V	(2 061.91 , 234 633.00)	NSGA&VNS	0.360	0.339	0.382
		NSGA	0.381	0.361	0.401
		VNS	0.264	0.226	0.371
R101-S2-18V	(2 150.99 , 234 633.00)	NSGA&VNS	0.391	0.365	0.409
		NSGA	0.388	0.373	0.405
		VNS	0.283	0.270	0.318

Table 5 Continued: $\mathcal{S}$ metric performance indicator for NSGA&VNS, NSGA and VNS

Instance	Ref. Point (Length, TW penalty)	Algorithm	Average	Min	Max
R101-S3-16V	(1 921.18 , 438 358.00)	NSGA&VNS	0.255	0.239	0.289
		NSGA	0.248	0.219	0.275
		VNS	0.191	0.177	0.208
R101-S3-18V	(2 157.72 , 438 358.00)	NSGA&VNS	0.351	0.335	0.371
		NSGA	0.324	0.308	0.349
		VNS	0.282	0.262	0.295
R101-S4-17V	(1 564.73 , 667 007.00)	NSGA&VNS	0.158	0.133	0.175
		NSGA	0.155	0.126	0.194
		VNS	0.077	0.060	0.103
R101-S4-20V	(1 772.16 , 783 106.00)	NSGA&VNS	0.236	0.216	0.255
		NSGA	0.230	0.213	0.263
		VNS	0.159	0.130	0.195
R102-S1-16V	(2 014.57 , 63 867.80)	NSGA&VNS	0.347	0.325	0.367
		NSGA	0.359	0.339	0.377
		VNS	0.304	0.280	0.327
R102-S1-17V	(2 135.05 , 63 867.80)	NSGA&VNS	0.394	0.363	0.423
		NSGA	0.393	0.375	0.406
		VNS	0.343	0.323	0.365
R102-S2-15V	(1 909.92 , 199 383.00)	NSGA&VNS	0.291	0.266	0.329
		NSGA	0.291	0.277	0.311
		VNS	0.273	0.244	0.305
R102-S2-17V	(2 061.51 , 199 383.00)	NSGA&VNS	0.345	0.317	0.376
		NSGA	0.350	0.326	0.369
		VNS	0.326	0.300	0.356
R102-S3-16V	(1 910.80 , 251 496.00)	NSGA&VNS	0.292	0.267	0.328
		NSGA	0.281	0.258	0.316
		VNS	0.273	0.244	0.307
R102-S3-17V	(2 034.24 , 251 496.00)	NSGA&VNS	0.330	0.308	0.372
		NSGA	0.323	0.306	0.360
		VNS	0.317	0.289	0.349
R102-S4-16V	(1 518.37 , 469 740.00)	NSGA&VNS	0.115	0.076	0.153
		NSGA	0.095	0.067	0.115
		VNS	0.081	0.066	0.105
R102-S4-17V	(1 615.26 , 345 090.00)	NSGA&VNS	0.173	0.155	0.191
		NSGA	0.135	0.094	0.158
		VNS	0.109	0.085	0.128
R103-S1-12V	(1 637.37 , 29 520.30)	NSGA&VNS	0.211	0.193	0.237
		NSGA	0.228	0.198	0.252
		VNS	0.208	0.186	0.236
R103-S1-14V	(1 832.29 , 35 477.60)	NSGA&VNS	0.298	0.279	0.320
		NSGA	0.309	0.299	0.323
		VNS	0.292	0.269	0.314
R103-S2-12V	(1 558.07 , 23 388.90)	NSGA&VNS	0.167	0.141	0.196
		NSGA	0.172	0.142	0.192
		VNS	0.164	0.141	0.193

Table 5 Continued: $\mathcal{S}$ metric performance indicator for NSGA&VNS, NSGA and VNS

Instance	Ref. Point (Length, TW penalty)	Algorithm	Average	Min	Max
R103-S2-14V	(1 733.80 , 9 535.90)	NSGA&VNS	0.241	0.217	0.260
		NSGA	0.238	0.221	0.257
		VNS	0.237	0.220	0.256
R103-S3-13V	(1 598.63 , 70 425.30)	NSGA&VNS	0.190	0.167	0.220
		NSGA	0.191	0.168	0.219
		VNS	0.190	0.168	0.219
R103-S4-12V	(1 219.23 , 78 611.60)	NSGA&VNS	0.022	0.014	0.029
		NSGA	0.021	0.013	0.039
		VNS	0.031	0.021	0.047
R103-S4-13V	(1 321.48 , 125 518.00)	NSGA&VNS	0.062	0.055	0.074
		NSGA	0.059	0.041	0.068
		VNS	0.056	0.042	0.073
R109-S1-12V	(1 561.66 , 38 849.00)	NSGA&VNS	0.159	0.148	0.178
		NSGA	0.187	0.172	0.205
		VNS	0.141	0.106	0.184
R109-S2-12V	(1 542.10 , 112 593.00)	NSGA&VNS	0.119	0.075	0.141
		NSGA	0.137	0.119	0.162
		VNS	0.115	0.090	0.143
R109-S3-12V	(1 540.96 , 30 060.10)	NSGA&VNS	0.120	0.088	0.156
		NSGA	0.129	0.106	0.145
		VNS	0.121	0.101	0.146
R109-S4-12V	(1 250.38 , 127 429.00)	NSGA&VNS	0.039	0.033	0.050
		NSGA	0.029	0.024	0.039
		VNS	0.038	0.028	0.056

Table 6: $\mathcal{C}$ metric performance indicator comparing NSGA&VNS and NSGA

Instance	Metric	Average	Min	Max
R101-S1-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.387	0.042	0.958
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.254	0.000	0.760
R101-S1-18V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.417	0.158	0.731
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.179	0.000	0.542
R101-S2-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.239	0.000	0.737
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.368	0.053	0.944
R101-S2-18V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.519	0.125	0.840
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.189	0.000	0.615
R101-S3-16V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.430	0.000	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.255	0.000	1.000
R101-S3-18V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.561	0.000	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.146	0.000	0.706
R101-S4-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.318	0.036	0.944
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.152	0.000	0.533
R101-S4-20V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.424	0.160	0.870
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.114	0.000	0.524
R102-S1-16V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.376	0.129	0.767
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.192	0.000	0.583

Table 6 Continued: $\mathcal{C}$ metric performance indicator comparing NSGA&VNS and NSGA

Instance	Metric	Average	Min	Max
R102-S1-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.435	0.069	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.164	0.000	0.741
R102-S2-15V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.385	0.000	0.950
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.177	0.000	0.733
R102-S2-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.473	0.042	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.157	0.000	0.778
R102-S3-16V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.519	0.154	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.114	0.000	0.462
R102-S3-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.591	0.000	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.133	0.000	0.875
R102-S4-16V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.501	0.059	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.100	0.000	0.800
R102-S4-17V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.593	0.235	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.096	0.000	0.571
R103-S1-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.196	0.000	0.909
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.436	0.000	1.000
R103-S1-14V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.333	0.045	0.867
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.249	0.000	0.833
R103-S2-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.203	0.000	0.875
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.489	0.000	1.000
R103-S2-14V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.368	0.000	0.875
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.269	0.000	0.900
R103-S3-13V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.229	0.000	0.875
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.341	0.000	1.000
R103-S4-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.259	0.000	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.152	0.000	1.000
R103-S4-13V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.343	0.000	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.154	0.000	0.833
R109-S1-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.131	0.000	0.500
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.444	0.000	0.846
R109-S2-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.193	0.000	0.643
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.144	0.000	0.875
R109-S3-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.259	0.000	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.279	0.000	0.857
R109-S4-12V	$\mathcal{C}$ (NSGA&VNS, NSGA)	0.423	0.077	1.000
	$\mathcal{C}$ (NSGA, NSGA&VNS)	0.028	0.000	0.667

Table 7: $\mathcal{C}$ metric performance indicator comparing NSGA&VNS and VNS

Instance	Metric	Average	Min	Max
R101-S1-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.236	0.000	0.600
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.114	0.000	0.375
R101-S1-18V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.415	0.083	0.857
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.081	0.000	0.263
R101-S2-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.357	0.000	0.700
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.084	0.000	0.350
R101-S2-18V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.318	0.000	0.625
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.038	0.000	0.125

Table 7 Continued: $\mathcal{C}$ metric performance indicator comparing NSGA&VNS and VNS

Instance	Metric	Average	Min	Max
R101-S3-16V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.436	0.125	1.000
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.070	0.000	0.400
R101-S3-18V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.283	0.111	0.667
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.130	0.000	0.500
R101-S4-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.298	0.000	0.857
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.046	0.000	0.167
R101-S4-20V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.459	0.250	1.000
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.048	0.000	0.188
R102-S1-16V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.330	0.000	0.750
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.066	0.000	0.313
R102-S1-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.250	0.000	0.625
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.097	0.000	0.368
R102-S2-15V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.259	0.000	0.700
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.148	0.000	0.400
R102-S2-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.215	0.000	0.750
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.141	0.000	0.385
R102-S3-16V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.311	0.000	0.700
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.165	0.000	0.500
R102-S3-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.271	0.000	0.714
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.140	0.000	0.467
R102-S4-16V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.108	0.000	0.500
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.270	0.077	1.000
R102-S4-17V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.241	0.000	0.625
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.143	0.000	0.667
R103-S1-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.230	0.000	0.833
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.277	0.000	0.857
R103-S1-14V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.261	0.000	0.846
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.169	0.000	0.429
R103-S2-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.187	0.000	0.833
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.251	0.000	0.750
R103-S2-14V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.239	0.000	0.875
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.195	0.000	0.600
R103-S3-13V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.175	0.000	0.667
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.250	0.000	1.000
R103-S4-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.022	0.000	0.500
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.411	0.000	1.000
R103-S4-13V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.085	0.000	1.000
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.170	0.000	1.000
R109-S1-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.000	0.000	0.000
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.165	0.000	0.500
R109-S2-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.014	0.000	0.500
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.376	0.000	1.000
R109-S3-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.010	0.000	0.250
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.375	0.000	1.000
R109-S4-12V	$\mathcal{C}$ (NSGA&VNS, VNS)	0.073	0.000	1.000
	$\mathcal{C}$ (VNS, NSGA&VNS)	0.215	0.000	1.000

Molde University College
Specialized University in Logistics
P.O. Box 2110
NO-6402 Molde
Norway
www.himolde.no

ISBN-13: 978-82-7962-209-3
ISSN: 0809-9588