

# DOCTOR OF PHILOSOPHY

A new three phase method (SDP method)  
for the multi-objective vehicle routing  
problem with simultaneous delivery and  
pickup (VRPSDP)

Iqbal Johal

2014

Aston University

**Some pages of this thesis may have been removed for copyright restrictions.**

If you have discovered material in AURA which is unlawful e.g. breaches copyright, (either yours or that of a third party) or any other law, including but not limited to those relating to patent, trademark, confidentiality, data protection, obscenity, defamation, libel, then please read our [Takedown Policy](#) and [contact the service](#) immediately

**A NEW THREE PHASE METHOD (SDPmethod)  
FOR THE MULTI-OBJECTIVE VEHICLE ROUTING PROBLEM  
WITH SIMULTANEOUS DELIVERY AND PICKUP (VRPSDP)**

**IQBAL SINGH JOHAL  
Doctor of Philosophy**

**ASTON UNIVERSITY  
May 2013**

© Iqbal Singh Johal, 2013

Iqbal Singh Johal (2013) asserts his moral right to be identified as the author of this thesis

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without appropriate permission or acknowledgement.

## SYNOPSIS

Transportation service operators are witnessing a growing demand for bi-directional movement of goods. Given this, the following thesis considers an extension to the vehicle routing problem (VRP) known as the delivery and pickup transportation problem (DPP), where delivery and pickup demands may occupy the same route. The problem is formulated here as the vehicle routing problem with simultaneous delivery and pickup (VRPSDP), which requires the concurrent service of the demands at the customer location. This formulation provides the greatest opportunity for cost savings for both the service provider and recipient.

The aims of this research are to propose a new theoretical design to solve the multi-objective VRPSDP, provide software support for the suggested design and validate the method through a set of experiments. A new real-life based multi-objective VRPSDP is studied here, which requires the minimisation of the often conflicting objectives: operated vehicle fleet size, total routing distance and the maximum variation between route distances (workload variation). The former two objectives are commonly encountered in the domain and the latter is introduced here because it is essential for real-life routing problems.

The VRPSDP is defined as a hard combinatorial optimisation problem, therefore an approximation method, Simultaneous Delivery and Pickup method (SDPmethod) is proposed to solve it. The SDPmethod consists of three phases. The first phase constructs a set of diverse partial solutions, where one is expected to form part of the near-optimal solution. The second phase determines assignment possibilities for each sub-problem. The third phase solves the sub-problems using a parallel genetic algorithm. The suggested genetic algorithm is improved by the introduction of a set of tools: genetic operator switching mechanism via diversity thresholds, accuracy analysis tool and a new fitness evaluation mechanism. This three phase method is proposed to address the shortcoming that exists in the domain, where an initial solution is built only then to be completely dismantled and redesigned in the optimisation phase. In addition, a new routing heuristic, RouteAlg, is proposed to solve the VRPSDP sub-problem, the travelling salesman problem with simultaneous delivery and pickup (TSPSDP).

The experimental studies are conducted using the well known benchmark Salhi and Nagy (1999) test problems, where the SDPmethod and RouteAlg solutions are compared with the prominent works in the VRPSDP domain. The SDPmethod has demonstrated to be an effective method for solving the multi-objective VRPSDP and the RouteAlg for the TSPSDP.

## **ACKNOWLEDGMENTS**

My deepest gratitude is provided to my parents and siblings for their continual love, support, patience as well as encouragement over the recent years when challenged with stressful episodes. They have helped me to grow as an individual, which in turn has helped this thesis to develop.

My sincere thanks go to my two supervisors Dr John Elgy and Dr Ming K Lim for their invaluable suggestions, which have set a new standard for my work for which I am extremely grateful. Their guidance has helped me in all sections of this thesis and without their help this thesis would have been a mere thought.

I am indebted to Dr Alina Patelli for guiding me through the world of evolutionary algorithms. Her deep knowledge of the subject has ignited my passion into the area, as demonstrated in this work. I am truly blessed to have spent time in her company.

Finally, but not least, I would like to thank my fellow PhD researchers, both past and present, for their kindness, friendship and support whilst sharing university offices over recent years, I have very fond memories. I hope the bonds that have been created will continue to strength over time.

## CONTENTS

<b>SYNOPSIS</b>	<b>2</b>	
<b>Acknowledgments</b>	<b>3</b>	
<b>Contents</b>	<b>4</b>	
<b>List of Figures</b>	<b>8</b>	
<b>List of Tables</b>	<b>10</b>	
<b>Acronyms</b>	<b>11</b>	
<b>1</b>	<b>Introduction</b>	<b>13</b>
<b>1.1</b>	Description of the problem	<b>13</b>
<b>1.2</b>	Research impact	<b>14</b>
<b>1.3</b>	Previous research	<b>15</b>
<b>1.4</b>	Persistent research niches	<b>16</b>
<b>1.5</b>	Research aims	<b>16</b>
<b>1.6</b>	Contributions	<b>17</b>
<b>1.6.1</b>	Design	<b>17</b>
<b>1.6.2</b>	Methods	<b>18</b>
<b>1.6.3</b>	Tools	<b>19</b>
<b>1.6.4</b>	Algorithmic contributions	<b>20</b>
<b>1.7</b>	Outline of the thesis	<b>21</b>
<b>2</b>	<b>Optimisation</b>	<b>22</b>
<b>2.1</b>	The CO problem - preliminaries	<b>22</b>
<b>2.2</b>	Computational complexity	<b>24</b>
<b>2.3</b>	Solution approaches	<b>24</b>
<b>2.3.1</b>	Exact algorithms	<b>25</b>
<b>2.3.2</b>	Approximation algorithms	<b>26</b>

<b>3</b>	<b>The Delivery and Pickup Problem</b>	<b>36</b>
<b>3.1</b>	<b>Travelling salesman problem</b>	<b>36</b>
<b>3.2</b>	<b>Vehicle routing problem</b>	<b>37</b>
<b>3.2.1</b>	<b>Capacitated vehicle routing problem (CVRP)</b>	<b>38</b>
<b>3.2.2</b>	<b>Vehicle routing problem with time windows (VRPTW)</b>	<b>39</b>
<b>3.3</b>	<b>Delivery and pickup problem (DPP)</b>	<b>39</b>
<b>3.3.1</b>	<b>Exact methods</b>	<b>42</b>
<b>3.3.2</b>	<b>Heuristics</b>	<b>44</b>
<b>3.3.3</b>	<b>Metaheuristics</b>	<b>46</b>
<b>4</b>	<b>Genetic Algorithm</b>	<b>68</b>
<b>4.1</b>	<b>Encoding</b>	<b>69</b>
<b>4.1.1</b>	<b>Binary encoding</b>	<b>69</b>
<b>4.1.2</b>	<b>Real number encoding</b>	<b>70</b>
<b>4.1.3</b>	<b>Permutation encoding</b>	<b>70</b>
<b>4.2</b>	<b>Linear evaluation</b>	<b>72</b>
<b>4.2.1</b>	<b>Pareto approaches</b>	<b>73</b>
<b>4.2.2</b>	<b>Non Pareto approaches</b>	<b>76</b>
<b>4.3</b>	<b>Selection for reinsertion</b>	<b>77</b>
<b>4.3.1</b>	<b>Proportional roulette wheel selection</b>	<b>78</b>
<b>4.3.2</b>	<b>Rank based roulette wheel selection</b>	<b>78</b>
<b>4.3.3</b>	<b>Tournament selection</b>	<b>79</b>
<b>4.3.4</b>	<b>Uniform selection</b>	<b>79</b>
<b>4.3.5</b>	<b>Select all selection</b>	<b>80</b>
<b>4.3.6</b>	<b>Elitist selection</b>	<b>80</b>
<b>4.4</b>	<b>Genetic operators</b>	<b>80</b>
<b>4.4.1</b>	<b>Crossover</b>	<b>80</b>
<b>4.4.2</b>	<b>Mutation</b>	<b>83</b>
<b>4.4.3</b>	<b>Encapsulation</b>	<b>84</b>
<b>4.4.4</b>	<b>Decimation</b>	<b>84</b>
<b>4.4.5</b>	<b>Expiry date</b>	<b>84</b>
<b>4.5</b>	<b>Selection for reinsertion</b>	<b>85</b>
<b>4.6</b>	<b>Termination criterion</b>	<b>86</b>

4.7	Multi Objective Evolutionary Algorithms	86
4.7.1	Non Elitist MOO Evolutionary Approaches	87
4.7.2	Elitist MOO Evolutionary Approaches	89
5	Method	96
	RouteAlg (Part 1)	96
5.1	Modified nearest neighbourhood (MNN) algorithm	96
5.2	Reverse procedure	98
5.3	Ejection/Reinsertion (EjRi) method	99
5.4	2-opt/Or-opt method	100
5.4.1	2-opt operator	101
5.4.2	Or-opt operator	103
	SDPmethod (Part 2)	106
5.5	Phase 1: Partial Solution Construction	107
5.5.1	Spatial clustering method (SCM)	107
5.5.2	Modified greedy selection strategy (MGSS)	110
5.5.3	Randomness inducing operator (RIO)	112
5.5.4	New modified greedy selection strategy (NEWMGSS)	114
5.6	Phase 2: Assignment possibilities	117
5.6.1	Radial expansion method (REM)	117
5.6.2	Defining segments method (DSM)	118
5.6.3	Single request adjustment method (SRAM)	119
5.7	Phase 3: Complete solution	119
5.7.1	Initial and genetic information processing	119
5.7.2	Linear evaluation	121
5.7.3	Selection for reproduction	125
5.7.4	Genetic operators	126
5.7.5	Selection for reinsertion	135
5.7.6	Termination criterion	136
5.7.7	Proposed Multi Objective Genetic Algorithm verses NSGA-II	137
6	Results and Discussion	141
6.1	Experimental setup description	141
6.2	Vehicle fleet size lower bounds	142

6.3	Published experimental case studies overview	144
6.4	RouteAlg validation results	144
6.4.1	Modified nearest neighbourhood (MNN) algorithm	145
6.4.2	Reverse procedure	147
6.4.3	Ejection/Reinsertion (EjRi) method	149
6.4.4	2-opt/Or-opt method	150
6.4.5	Results discussion	152
6.5	SDPmethod Validation Results	157
6.5.1	Comparison with best known results	157
6.5.2	Comparison with Vural (2007)	161
6.5.3	Comparison with Jun and Kim (2012)	163
6.6	Genetic algorithm analysis	166
6.6.1	Population size, maximum generations and mutation rate setup	166
6.6.2	Genetic operators	168
6.6.3	Distribution of non-dominated individuals	172
7	Conclusion	175
7.1	Problem description	175
7.2	Problem importance	175
7.3	Proposed methods	175
7.4	Introduced benefits	176
7.5	Research objectives and contributions	176
7.6	Results	177
7.6.1	Experimental comparisons	177
7.6.2	Impact of proposed design	178
7.6.3	Impact of proposed tools	178
7.7	Future research	178
	<b>List of References</b>	<b>181</b>
	<b>Appendix A. Detailed results for RouteAlg</b>	<b>192</b>
	<b>Appendix B. Detailed results for SDPmethod</b>	<b>216</b>
	<b>Appendix C. Multi Objective Genetic Algorithm Parameter Setup Results</b>	<b>249</b>

## LIST OF FIGURES

<b>Figure 2.1</b>	Ant colony optimisation outline	<b>29</b>
<b>Figure 3.1a</b>	Travelling salesman problem	<b>36</b>
<b>Figure 3.1b</b>	Hamilton cycle for the travelling salesman problem	<b>36</b>
<b>Figure 3.2</b>	Vehicle routing problem solution	<b>38</b>
<b>Figure 4.1</b>	Binary encoding	<b>70</b>
<b>Figure 4.2</b>	Real number encoding	<b>70</b>
<b>Figure 4.3</b>	Permutation encoding	<b>70</b>
<b>Figure 4.4</b>	Uniform distribution of chromosomes in the search space	<b>72</b>
<b>Figure 4.5a</b>	Dominance relationship	<b>73</b>
<b>Figure 4.5b</b>	Non-dominance relationship	<b>73</b>
<b>Figure 4.6</b>	Pareto fronts	<b>74</b>
<b>Figure 4.7</b>	Insular model	<b>77</b>
<b>Figure 4.8</b>	Rank-based roulette wheel selection	<b>79</b>
<b>Figure 4.9</b>	Gene level crossover	<b>81</b>
<b>Figure 4.10</b>	Single and two cut point crossover	<b>82</b>
<b>Figure 4.11</b>	Gene level mutation	<b>83</b>
<b>Figure 4.12</b>	DPGA fitness assignment approach	<b>93</b>
<b>Figure 5.1</b>	Modified Nearest Neighbourhood (MNN) algorithm pseudo code	<b>97</b>
<b>Figure 5.2a</b>	Reverse procedure (Incumbent cycle)	<b>98</b>
<b>Figure 5.2b</b>	Reverse procedure (Reverse cycle)	<b>98</b>
<b>Figure 5.3a</b>	Ejection/Reinsertion (EjRi) method (Incumbent cycle)	<b>100</b>
<b>Figure 5.3b</b>	Ejection/Reinsertion (EjRi) method (New cycle)	<b>100</b>
<b>Figure 5.4a</b>	2-opt operator (Incumbent cycle)	<b>101</b>
<b>Figure 5.4b</b>	2-opt operator (New cycle)	<b>101</b>
<b>Figure 5.5</b>	2-opt operator pseudo code	<b>102</b>
<b>Figure 5.6a</b>	Or-opt operator (Incumbent cycle)	<b>103</b>
<b>Figure 5.6b</b>	Or-opt operator (New cycle)	<b>103</b>
<b>Figure 5.7</b>	Or-opt pseudo code	<b>104</b>
<b>Figure 5.8</b>	Components of the SDPmethod	<b>106</b>
<b>Figure 5.9</b>	Spatial Clustering Method (SCM) pseudo code	<b>108</b>
<b>Figure 5.10a</b>	Dispersion of elements (large distance)	<b>109</b>
<b>Figure 5.10b</b>	Dispersion of elements (small distance)	<b>109</b>

<b>Figure 5.11</b>	Route distance of cluster	<b>110</b>
<b>Figure 5.12</b>	Modified Greedy Selection Strategy (MGSS) pseudo code	<b>111</b>
<b>Figure 5.13</b>	Randomness Inducing Operator (RIO) pseudo code	<b>113</b>
<b>Figure 5.14</b>	New Modified Greedy Selection Strategy (NEWMGSS) pseudo code	<b>115</b>
<b>Figure 5.15</b>	Radial Expansion Method (REM) pseudo code	<b>117</b>
<b>Figure 5.16</b>	Expanded clusters	<b>118</b>
<b>Figure 5.17</b>	Isolated requests	<b>119</b>
<b>Figure 5.18</b>	Chromosome construction	<b>120</b>
<b>Figure 5.19</b>	Pareto set separation pseudo code	<b>122</b>
<b>Figure 5.20</b>	Proportional roulette wheel selection (PRWS)	<b>125</b>
<b>Figure 5.21</b>	Homologous cut-point crossover	<b>127</b>
<b>Figure 5.22</b>	Gene level crossover	<b>128</b>
<b>Figure 5.23</b>	Mutation operator	<b>129</b>
<b>Figure 5.24</b>	Population average Hamming Distance	<b>130</b>
<b>Figure 5.25</b>	Switching parameter	<b>132</b>
<b>Figure 5.26</b>	Genetic operator switching mechanism pseudo code	<b>133</b>
<b>Figure 5.27</b>	Average distance between 1 <sup>st</sup> order Pareto front and the origin	<b>134</b>
<b>Figure 5.28</b>	Selection for reinsertion	<b>136</b>
<b>Figure 6.1</b>	Modified Nearest Neighbourhood (MNN) performance	<b>146</b>
<b>Figure 6.2</b>	Modified Nearest Neighbourhood (MNN) problem size	<b>147</b>
<b>Figure 6.3</b>	Reverse procedure performance	<b>148</b>
<b>Figure 6.4</b>	Reverse procedure problem size	<b>149</b>
<b>Figure 6.5</b>	Ejection/Reinsertion (EjRi) method performance	<b>150</b>
<b>Figure 6.6</b>	2-opt/Or-opt method performance	<b>151</b>
<b>Figure 6.7</b>	2-opt/Or-opt method problem size	<b>152</b>
<b>Figure 6.8</b>	RouteAlg comparison with Vural (2007)	<b>153</b>
<b>Figure 6.9</b>	RouteAlg comparison with Vural (2007) problem size	<b>154</b>
<b>Figure 6.10</b>	RouteAlg computational efficiency	<b>155</b>
<b>Figure 6.11</b>	RouteAlg computational efficiency based on problem size	<b>156</b>
<b>Figure 6.12</b>	Workload variation comparisons with Vural (2007)	<b>163</b>
<b>Figure 6.13</b>	Workload variation comparisons with Jun and Kim (2012)	<b>165</b>
<b>Figure 6.14</b>	Average distance between the 1 <sup>st</sup> order Pareto front and the origin	<b>169</b>
<b>Figure 6.15</b>	Average population Hamming distance	<b>170</b>
<b>Figure 6.16</b>	Non-dominated individuals found during the search	<b>173</b>

## LIST OF TABLES

<b>Table 3.1</b>	Main contributions from the VRPSDP domain	<b>64</b>
<b>Table 3.2</b>	Main test problem types adopted in the VRPSDP domain	<b>67</b>
<b>Table 4.1</b>	MOEA research works review	<b>94</b>
<b>Table 5.1</b>	Proposed Multi Objective Genetic Algorithm compared to NSGA-II	<b>140</b>
<b>Table 6.1</b>	Summary of the benchmark test problems	<b>142</b>
<b>Table 6.2</b>	Vehicle fleet size lower bounds for each test problem	<b>143</b>
<b>Table 6.3</b>	SDPmethod solution comparison with best known	<b>158</b>
<b>Table 6.4</b>	SDPmethod solution comparison with best known (Distance constraint)	<b>159</b>
<b>Table 6.5</b>	Vural (2007) comparison results	<b>161</b>
<b>Table 6.6</b>	Jun and Kim (2012) Results	<b>164</b>
<b>Table 6.7</b>	GA parameter setting	<b>167</b>
<b>Table 6.8</b>	Mutation rate tuning	<b>168</b>

## ACRONYMS

<b>ACS</b>	Ant colony system
<b>B&amp;B</b>	Branch and bound
<b>B&amp;C</b>	Branch and cut
<b>BB</b>	Building Block
<b>BPP</b>	Bin packing problem
<b>CA</b>	Cultural algorithm
<b>CO</b>	Combinatorial optimisation
<b>CUP</b>	Continuous update procedure
<b>CVRP</b>	Capacitated vehicle routing problem
<b>DE</b>	Differential evolution
<b>DPP</b>	Delivery and pickup transport problem
<b>DSM</b>	Defining segments method
<b>EA</b>	Evolutionary algorithm
<b>ED</b>	Estimated difference
<b>EjRi</b>	Ejection/Reinsertion method
<b>ES</b>	Evolutionary strategies
<b>GA</b>	Genetic algorithm
<b>GLS</b>	Guided location search
<b>GO</b>	Genetic operators
<b>GP</b>	Genetic programming
<b>GRASP</b>	Greedy randomised adaptive search procedure
<b>HC</b>	Hill Climbing
<b>MGSS</b>	Modified greedy selection strategy
<b>MNN</b>	Modified nearest neighbourhood
<b>MOEA</b>	Multi-objective evolutionary algorithm
<b>MOGA</b>	Multi-objective genetic algorithm
<b>MOO</b>	Multi-objective optimisation
<b>NEWMGSS</b>	New modified greedy selection strategy
<b>NPSO</b>	Nested particle swarm optimisation
<b>PF</b>	Pareto front
<b>PRWS</b>	Proportional roulette wheel selection
<b>PSO</b>	Particle swarm optimisation

<b>REM</b>	Radial expansion method
<b>RIO</b>	Randomness inducing operator
<b>RRT</b>	Record-to-record travel
<b>RSCIM</b>	Random seeds cheapest insertion method
<b>SA</b>	Simulated annealing
<b>SAHC</b>	Steepest ascent hill climbing
<b>SCM</b>	Spatial clustering method
<b>SOO</b>	Single objective optimisation
<b>SRAM</b>	Single request adjustment method
<b>SS</b>	Scatter search
<b>TS</b>	Tabu search
<b>VND</b>	Variable neighbourhood descent
<b>VNS</b>	Variable neighbourhood search
<b>VRP</b>	Vehicle routing problem
<b>VRPB</b>	Vehicle routing problem with backhauls
<b>VRPMB</b>	Vehicle routing problem with mixed backhauls
<b>VRPSDP</b>	Vehicle routing problem with simultaneous delivery and pickup
<b>VRPSDPTW</b>	Vehicle routing problem with simultaneous delivery and pickup amongst time windows
<b>VRPTW</b>	Vehicle routing problem with time windows
<b>TSP</b>	Travelling salesman problem
<b>TSPSDP</b>	Travelling salesman problem with simultaneous delivery and pickup

## **1. INTRODUCTION**

The logistics discipline involves managing the flow of products or information through a business, from the point of origin to the end customer (Christopher, 2011). The efficient execution of this activity ensures that a product or service is delivered in the right place and quantity, as well as at the right time, quality and cost (Chopra and Meindl, 2007). The logistics activity is of great economic importance because it involves significant amount of human and material resources and also because a substantial part of the economy relies on logistics services. For instance, it is estimated that within the UK 30% of the working population has an occupation that is associated with logistics (Rushton et al., 2010). Due to the capital intensive nature of the logistics activity it is widely acknowledged that a scientific approach is required for solving logistical problems (Rushton et al., 2010). The transport element represents the largest cost within logistics, approximately 40% of the total costs (Rushton et al., 2010). In consequence, it is imperative that efficient systems are developed, in order to reduce the cost of this element.

Transport is a crucial process that enables supply chain members, with different levels of expertise to work together, resulting in the fabrication of a product or service that is fit for consumption. The most efficient transportation mode to any final downstream customer is via road transportation (European Commission, 2012), (Transport Studies Department, 2010). Typically, the commercial road transportation market within the United Kingdom is highly competitive (European Commission, 2012), therefore an effective physical distribution system is required by the operator to schedule and route vehicles to achieve cost-effectiveness (Chopra and Meindl, 2007). The transportation problem commonly requires the minimisation of two objectives: the vehicle fleet size (number of operated routes) and the total distance travelled, given that these two represent the most important fixed and variable costs, respectively, handled by the transportation operator (Golden et al. 2008).

### **1.1 Description of the problem**

This research is focused on solving the Delivery and Pickup transport problem (DPP), wherein delivery and pickup demands may require service on the same route. The problem typically entails the delivery of goods from the depot to a set of customers and the subsequent collection of goods to be returned to the depot. The DPP is herein formulated as the vehicle routing problem with simultaneous delivery and pickup (VRPSDP), which restricts the service

of the delivery and pickup demands at a customer location to a single visit. This formulation is advantageous from both the standpoint of the commercial operator and that of the service receiver. In this context, the simultaneous service of the two demands leads to a reduction in routing distance, which in turn may reduce the number of vehicles operated by the service provider. Also, the service recipient benefits from a lower goods handling cost compared to the individual delivery of the two demands.

Nowadays, most transportation practitioners provide their customers a choice of time intervals for service, at an additional expense. Therefore, it is important for researchers to design methods able to handle time constraints, even though this additional dimension increases the complexity of the problem. The VRPSDP with time windows (VRPSDPTW) is unfortunately outside the scope of this research because a multi-objective problem is studied, in order to develop an unexplored area of research. The reader is referred to Wang and Chen (2012) and Angelelli and Mansini (2002) for a greater insight into the current state of VRPSDPTW domain.

The VRPSDP is a combinatorial optimisation problem with a complexity defined as NP-hard (Dethloff, 2001). The application of exact methods to solve the VRPSDP is limited, as the number of evaluations required to reach the optimal solution increase exponentially with the problem size. In contrast, approximation methods have a greater practical utility in solving large NP-hard problems because they improve the trade-off between solution quality and computational resource consumption.

## **1.2 Research Impact**

The initial motivation for conducting research relative to the delivery and pickup problem arose whilst the author was undertaking an internship with a multinational parcel service operator, which provided small shipment transportation via the road network. The operator determined driver assignments based on predesigned territories, which were an amalgamation of postcode sectors. Most commonly, a postcode sector defines a unique area inside a geographical district. The subsequent routes were sequenced based on milk runs, which describes a fixed routing pattern. As observed in that context, a non-holistic approach to solving the problem is likely to result in a sub-optimal solution. The desire to find an improved routing method has led to this research.

The VRPSDP is interesting both at a theoretical and commercial standpoint. The relaxed version of the problem, the vehicle routing problem is one of the most important and widely studied combinatorial optimisation problems Toth and Vigo (2002). Therefore, the theoretical interest in the VRPSDP should be equal or more significant because the formulation has an increased complexity, as fluctuating capacities have to be considered at each node. From a practical viewpoint, the specific delivery and pickup problem is of great logistical importance. For instance, the UK retail sector is experiencing a structural change caused by a significant growth in online retailing (Department for Communities and Local Government, 2013), (Royal Mail Group Limited, 2013). In 2010, the UK had the highest per capita online spend in Europe and internet sales were reported to be worth £23.4 billion (Department for Business Innovation & Skills, 2012). The growth in online retailing has caused a significant increase in UK parcel volumes (Royal Mail Group Limited, 2013), because more shoppers opt for home deliveries and collections. Parcel volumes are forecasted to grow their share of total UK inland mail volume up until 2023 (PwC Strategy & Economics, 2013). In consequence, parcel service operators are increasing their infrastructure investment to help meet the projected needs of the parcel service market (Royal Mail Group Limited, 2013), which has a current market value of £6 billion (UK Mail, 2013). The growing parcel volume will increase the demand for bi-directional transport and thus the need for practical and effective solution methods for the VRPSDP.

### **1.3 Previous research**

The VRPSDP is an important contemporary problem and the research area has grown over the past decade. Yet, the VRPSDP domain remains relatively unexplored compared to the vehicle routing problem based on a publication count comparison. The majority of solution methods that have been proposed are based on metaheuristics because they balance the trade-off between solution quality and computational resource consumption better than exact techniques. In reference to the volume of publications, evolutionary algorithms have contributed the most to this field compared to any other paradigm, as illustrated in Table 3.1 in Chapter 3. A common optimisation theme runs through most works, which relates to the minimisation of the vehicle fleet size as well as the total distance travelled. Another noteworthy point relates to the design of the solution methods. A common design procedure exists, where an initial complete solution is generated and then later optimised. On the issue of time windows, the VRPSDP domain has seen limited research relative to the publication count, possibly due to the consequence of increased problem complexity.

#### **1.4 Persistent research niches**

The solution methods published in the VRPSDP domain have yet to consider a workload balancing objective. This objective has a commercial importance because it is commonly embedded in union contracts or company regulations, in terms of minimum and maximum working hours Toth and Vigo (2002). This objective is meant to ensure there is workload equality among the routes in terms of duration. Therefore, introducing this objective to the VRPSDP domain represents a bridge over the gap between theoretical study and practical application.

A major shortcoming of the current solution design, found in the VRPSDP literature, is that the initial solution may be completely modified in the optimisation phase without safeguarding certain potentially valuable building blocks. The underlining conceptual idea is that there is likely to be a set of partial route assignments in the search space that are too spatially distant from their neighbouring routes to even contemplate their relocation. Presently, no research work has attempted to identify such assignments for the VRPSDP and then gone on to solve the remaining sub-problem. In consequence, a theoretical study in this area will represent a substantial contribution to the domain.

#### **1.5 Research Aims**

This research will address the persistent research niches that exist in the VRPSDP domain. In particular, the aims of this research are defined in terms of the following: firstly, to suggest a new method design to solve the VRPSDP; secondly, to provide software (SW) support for the suggested design, one that is proficient, and finally, conduct a new set of experimental studies to evaluate quantitatively the software implementation, which differ than the ones available in the literature, i.e., measure analysis and compare parameters that have not been previously considered.

The VRPSDP solution method introduced here aims to solve a multi objective optimisation problem with the following high level objectives:

- minimise the operated vehicle fleet size
- minimise the total routing distance travelled
- minimise the variation between the maximum and minimum vehicle routing distance

## 1.6 Contributions

This research introduces two new solution methods: SDPmethod and RouteAlg. The former is described as the main contribution to this research, which is used to solve the multi-objective VRPSDP. The RouteAlg is introduced to solve the routing problems, defined by the SDPmethod, called the travelling salesman problem with simultaneous delivery and pickup (TSPSDP).

The novelties of this work are classified in two categories: theoretical and algorithmic. A short description is provided for each contribution, along with the main advantages. The reader is referred to Chapter 5, where more detailed explanations are provided.

A classification of the proposed theoretical contributions follows here, in terms of design (architectural viewpoint), methods (general approaches for solving the problem under discussion) and tools (specific algorithm implements):

### 1.6.1 Design

- a. A new four component method, RouteAlg, gradually guides the candidate routes to a TSPSDP feasible space before investing additional computational effort on optimisation. The method applies the following approaches in sequence: Modified Nearest Neighbourhood (MNN) algorithm, Reverse procedure, Ejection Reinsertion (EjRi) method and the 2-opt/Or-opt method. It is assumed that the RouteAlg will be able to determine high quality routes for the TSPSDP because it comprises of a set of methods capable of improving search feasibility and optimisation (6.4).
- b. A three phase heuristic, SDPmethod, of a design not previously used in the VRPSDP domain, is adopted here. The proposed design will attempt to identify a set of partial assignments in the vicinity of the near-optimal solution for the first time in the VRPSDP domain and then to focus the computational effort on optimising the assignment of requests, which are deemed less intuitive. The first phase aims to identify a set of partial assignments that are likely to be in the vicinity of the near optimal solution, the second phase determines the assignment possibilities for the unassigned elements and the final phase completes the remaining assignment problem using a customised genetic algorithm. The integrated evaluation of all three phases of the algorithm is performed by measuring the quality of the considered objectives and the structural diversity of the final solutions. The rationale resulting in the development of the three phase design is as follows. Phase 1 addresses the major

shortcoming in the literature relating to the failure to protect potentially valuable building blocks in the optimisation stage. Phase 2 is an auxiliary step introduced to reduce the remaining assignment challenge to only investigate more promising areas. Finally, a genetic algorithm is employed in Phase 3 because it is widely recognised to be a powerful approach for solving NP-hard multi objective problems (Deb, 2009). It is assumed that the SDPmethod will generate a range of high quality trade-off solutions, for the conflicting objectives considered in this work, which the commercial operators can select from based on their needs.

- c. The SDPmethod is a multi-objective search heuristic that strives to promote the development of routes with a balanced workload. This objective is considered here for the first time in the VRPSDP domain. It is an essential consideration for real-life routing problems, in order to bridge the gap between theoretical study and commercial practice.
- d. A parallel genetic algorithm model is proposed in Phase 3 of the SDPmethod. The same genetic algorithm is simply run in parallel to evolve multiple populations of a diverse nature, in order to explore a wider search space.

#### 1.6.2 Methods

- a. The herein proposed Modified Nearest Neighbourhood (MNN) algorithm, the first component of the RouteAlg, is a modified version of the Nearest Neighbourhood (NN) algorithm, which determines a range of solutions for the TSP. The additional diversity provided to the search is likely to improve the TSP solution quality, when compared with the NN algorithm result.
- b. The third component of the RouteAlg is the proposed Ejection Reinsertion (EjRi) method. The method rearranges the nodes on a cycle in order to minimise TSPSDP infeasibility, which may still exist after the application of the reverse procedure. This method guides the search towards a feasible space.
- c. The final component of the RouteAlg is the newly introduced 2-opt/Or-opt method. It represents the first attempt in the VRPSDP domain to apply the 2-opt and Or-opt operators in an isolated loop, for the purpose of optimisation. These powerful operators explore each other's neighbourhood, until no further improvement is possible.
- d. The introduced Spatial Clustering Method (SCM) in Phase 1 of the SDPmethod is a new clustering technique that can determine a wide range of TSPSDP feasible routes, with

similar vehicle capacity utilisations. In the case of a non-bulk problem, the vehicle utilisation on a route is a good measure of the workload, as it illustrates the number of customers on the route. Therefore, this clustering restriction aims to ensure that routes have a balanced workload. In addition, the SCM builds clusters using circular vicinities in order to encourage compact assignment of requests. This is suggested to maximise the density of requests assigned within a given area.

- e. The proposed Modified Greedy Selection Strategy (MGSS) found in Phase 1 of the SDPmethod is a new, greedy selection scheme that can be used to determine the existence of a set of unique cluster assignments. The advantage of this method is that such a set is likely to be found quickly, if one exists. In addition, the greedy nature of the method is likely to result in a small total routing distance for the cluster set.
- f. The presented New Modified Greedy Selection Strategy (NEWMGSS) in Phase 1 of the SDPmethod is another new, greedy selection scheme, which is used to determine a diverse range of cluster sets. The advantage of this method is that it increases the robustness of the search, as the sets are spread throughout the search space.
- g. The herein proposed Radial Expansion Method (REM) defines assignment possibilities based on spatial proximity, in Phase 2 of the SDPmethod. This targeting is likely to lead to a much faster exploitation of a high quality solution space in comparison to conventional approaches, Wassan et al. (2008), Dethloff (2001) and Min (1989).

### 1.6.3 Tools

- a. New genetic operator (GO) switching logic

A new switching logic between genetic operators is introduced for the genetic algorithm implemented in Phase 3 of the SDPmethod. The switching logic is based on the population diversity. The purpose of this tool is to explore high quality areas of the search space, whilst preventing the search from being trapped indefinitely, in local optimum points.

- b. New accuracy analysis tool based on distance measuring operators

In Phase 3 of the SDPmethod, the average distance between the 1<sup>st</sup> order Pareto front and the origin is calculated in order to monitor the improvement in the accuracy of the best current genetic algorithm individuals throughout the search.

- c. Customised fitness computation formula (Patelli (2011) enhancement)  
Patelli (2011) fitness evaluation has been improved in Phase 3 of the SDPmethod by considering the estimated distance between the different Pareto sets during the computation. This will increase the accuracy of the fitness computation.

A classification of the proposed algorithmic contribution follows here:

#### 1.6.4 Algorithmic Contributions

- a. Implementation

The programming code for the RouteAlg and SDPmethod were written and implemented by the author of this research in Matlab.

- b. Adaptive vehicle utilisation setting

The SCM was implemented in such a way as to maximise the vehicle capacity utilisation, therefore reducing the size of the sub-problem. Therefore, less computational resource consumption was required at the later stages where the remaining assignment problem was deemed more challenging.

- c. Initial population modification

Certain gene values of the randomly generated initial population individuals are modified to ensure that a single segment cluster assignment found by the SDPmethod in Phase 2 is reflected in the population. The gene selection strategy will be explained later on in 5.7.1. This new software technique is meant to ensure that the spatially closest cluster assigns the gene value, thus hopefully reducing the computational time required to find an improved solution.

- d. Adaptive switch threshold setting

The genetic operator switching thresholds defined for the genetic algorithm are adaptively configured software wise, over the initial generations. The subsequent thresholds are likely to be relevant in terms of a particular dataset, when compared to the deterministically determined values. Therefore, in theory the search should progress more efficiently.

- e. Crossover restriction

A software mechanism has been set into place in order to allow the cut point and gene level crossover to exchange genetic material in a manner restricted to a sub-chromosome level. This way, the modifications are in line with the assignment possibilities identified in Phase 2. These assignment possibilities ensure that unassigned requests are prevented from being assigned to distant clusters.

f. Mutation restriction

In the genetic algorithm, the genes inside a chromosome may have their own unique alphabets, which are defined using the outcome of Phase 2. Those genes that share the same alphabet define a particular section of the chromosome called a sub-chromosome. Therefore, this software component has been added to the architecture of the suggested algorithm in order to restrict the mutation operation to a legal set of alleles. The advantage of this is the same as described previously for the crossover restriction.

g. Fresh genetic material

The population deficit created by the decimation operator is compensated for via the new Snap-shot software operator, Patelli (2011), which inserts randomly selected individuals from a previous population. This operator should generate an acceptable level of population diversity, whilst minimising the impact on the average population fitness.

h. Adaptive reinsertion

A new software operator (described in section 5.7.5) is herein introduced for the purpose of coupling the individuals' reinsertion technique with the currently used genetic operator. This will help encourage the appropriate level of accuracy or diversity required by the search, without resulting in any conflicting operations.

## 1.7 Outline of the thesis

In Chapter 2, the notion of combinatorial optimisation problem is explained, along with several methods commonly used to tackle this problem. A state of the art of the VRPSDP literature is included in Chapter 3. In Chapter 4, a basic insight is provided into the genetic algorithm and multi objective evolutionary algorithms, as these are of a key aspect of the methodology. Furthermore, the chapter reviews the most prominent multi-objective evolutionary algorithm research works, as the paradigm is applied here. Chapter 5 describes the methodology adopted in this thesis, in terms of solving the VRPSDP and the TSPSDP. The results and discussions follow in Chapter 6, where comparisons are made with the best known solutions for Salhi and Nagy (1999) test problems. Finally, in Chapter 7 the conclusion and the direction of future work are provided.

## 2. OPTIMISATION

In the context of this research, this chapter introduces the Combinatorial Optimisation (CO) problem and then relates it to a multi-objective framework. Following on, the computational complexity of a problem is examined. Finally, several methods which are unanimously considered to be efficient in exploring the search space of the CO problem (Hosny, 2010) are herein described. These methods fall under two main categories: exact and approximation procedures.

### 2.1 The CO problem - preliminaries

Combinatorial optimisation problems consist of minimising or maximising a function ( $F$ ), within a specific space in the domain ( $D$ ), as shown in Equation 2.1.

$$F : D^m \rightarrow R^n \quad F(x) = y, \quad \begin{array}{l} x \in D^m \\ y \in R^n \end{array} \quad (2.1)$$

where,  $F$  = Function,  $D$  = Domain (input),  $R$  = Range (output), and  $m$  and  $n$  are natural numbers that represent the dimension of the domain and the range.

Solving CO problems consists in finding a solution  $x^*$  in the domain, which will generate the optimum  $y^*$  value in the range.

In the case of multimodal functions [ $F$  features  $Q$  optima,  $y_i^*$ ,  $i=1..Q$ ], the CO problem solution is a set of  $K$  points  $x_i^*$ ,  $i=1..K$ ,  $F(x_i^*) = y_i^*$ .

Most real-world CO problems entail constraints; therefore constrained optimisation is an important field for researchers (Deb, 2009), (Coello Coello et al., 2007). The use of constraints limits the domain space to a certain area  $D_1$ , where feasible solutions can be found. In such a case,  $x \in D_1 \subset D$  represent the candidate solutions from the subset domain  $D_1$ , namely the ones which satisfy the feasibility restrictions.

The majority of real world CO problems involve multiple objectives (Deb, 2009). Multiple objective CO (MOO) problems consist of optimising a set of functions:

$$F_1(x) \dots F_f(x), x \in \mathbf{D}^m \quad (2.2)$$

where  $F_i, i=1 \dots f$  is defined in (2.1).

The multiple functions to be optimised are commonly conflicting (Deb, 2009), therefore a single solution that optimises all objectives may not exist, instead a range of trade-off solutions are considered, possibly an infinite number (Coello Coello and Lamont, 2004). For instance, a solution may be better with respect to a particular objective, however this may have come at the detriment of another objective. The majority of real-world MOO problems are non-linear in nature (Deb, 2009), therefore the improvement in one objective value may reduce another in a nonlinear way.

There are two essential goals for solving a MOO problem: the first is to determine a number of solutions close to the global optimal set and the second is to find a diverse spread of solutions within the same set (Coello Coello et al., 2007). The former goal is true for all optimisation problems, although the principle used to optimise a single objective function in SOO is not applicable in a MOO context because more than one objective is of importance. The later goal is unique to MOO and is introduced to increase the likelihood of finding an acceptable solution for the decision maker.

Just like single objective optimisation, MOO requires the use of a decision variable space and an objective space. However, in the case of MOO, the mapping between the two connected spaces is likely to be nonlinear, therefore the proximity of a pair of solutions in one space does not transpose directly onto the other space (Deb, 2009). In addition, the MOO goal of maintaining diversity among the global optimal set may be achieved by promoting diversity in either of the spaces. However, the coordination between the two spaces to promote diversity is not a trivial task (Deb, 2009).

The MOO problem considers all objectives to be important, therefore the decision maker is required to select a solution from a choice of optimal solutions by making compromises. The ideal solution should provide acceptable performance across all objectives (Coello Coello and Lamont, 2004). The decision maker is likely to utilise additional information about the problem, which may not have been modelled in order to select the most suitable solution.

This research is focused on solving a multi-objective CO problem called the vehicle routing problem with simultaneous delivery and pickup (VRPSDP), as defined in Chapter 3. An extension to the previously considered objectives (vehicle fleet size and total routing distance), this research work will consider the workload variation objective, in order to induce greater equality among routes.

## **2.2 Computational Complexity**

Two of the recognised complexity classification problem types are: polynomial time (P) solvable and non-deterministic polynomial time (NP) solvable. In the former problem, the number of instructions to be performed by the deterministic algorithm are  $O(n^k)$ , where  $n^k$  is the number of operations that get executed in the fragment of code. This problem type is easily solvable (Morgan, 2008). For the latter problem type, no known deterministic polynomial time solution method exists. Instead the algorithm can estimate a solution for the decision problem and verify it in polynomial time. To date, this problem type is solvable in exponential time (Hochbaum, 1997).

According to (Goldreich, 2010) a problem is classified as NP-hard if it is at least as difficult as the hardest problems in NP. NP-complete refers to a subset of problems that are both NP and NP-hard. These problems are the most complex problems within the NP domain. To date, an efficient algorithm for solving one of these problems is not known, however this does not imply that one does not exist (Hochbaum, 1997). The vehicle routing problem (VRP), see Chapter 3, a less complex variant of VRPSDP has been proven to be NP-hard (Toth and Vigo, 2002). Therefore, VRPSDP considered in this research can be classified with the same complexity (Dethloff, 2001).

## **2.3 Solution Approaches**

Much of the growth in operational research literature is attributed to the recognition by academics of the need for methods to solve real-world CO problems. Methods capable of producing effective solutions are required to reduce the cost of optimisation and/or to increase the offered service level, in terms of end-user satisfaction. Gendreau and Potvin (2010) states that the advancements in optimisation techniques for the CO problems have allowed researchers to address more complex problems in terms of dataset size and search space irregularities (discontinuity, nonlinearity, etc). This is justified by three main reasons: the

improvement in algorithmic design, the innovation in computer performance and the improved communication of ideas.

A CO problem may be solved to optimality using a brute force approach, which investigates all possible solutions. This approach is impractical for large NP-hard problems, as the number of required evaluations is significantly large. Therefore, other types of approaches are needed, which evaluate the solution space economically in order to reach optimality. These approaches are categorised as exact and approximation algorithms.

### 2.3.1 Exact Algorithms

The optimal solution for a CO problem may be located by an exact method, if one exists. This type of approach will reduce the solution space size under investigation, in order to perform fewer evaluations (Hosny, 2010). The practicality of exact methods in solving CO problems diminishes as the problem size grows, therefore, their real-world application is limited (Najera, 2010). The following prominent exact solution approaches are discussed in greater detail below: Branch and Bound (B&B) and Branch and Cut (B&C) because they have been used to solve the herein problem (Subramanian et al. 2011), (Angelelli and Mansini, 2002).

#### Branch and Bound (B&B)

The branch and bound algorithm has been widely used to solve the vehicle routing problem and many of its variants (Toth and Vigo, 2002). B&B is a tree search technique used to solve optimisation problems. This mechanism is used to divide the solution space into sub-problems, known as branches. The branching process channels the exploration process towards certain areas of the solution space. Ideally, branching should quickly converge to the optimal solution. After every branching process, the subsequent solution node is evaluated. However, this operation is computationally expensive, therefore, a branching limit is imposed. If the quality of the solution encrypted by the currently evaluated node exceeds the upper bound, no further branches will be explored from that particular point onwards. This process is referred to as pruning. Preferably, pruning should eliminate nodes situated near the tree root, in order to save computational resources. B&B continues to branch from un-pruned branches until the termination criterion is met. The approach converges within a finite number of iterations. However, the duration of these iterations may grow exponentially with the problem size.

### Branch and Cut (B&C)

This approach combines the B&B algorithm with the cutting plane technique. The underlying principle is based on reducing the search space available for exploration by integrating problem specific constraints (cuts) during the bounding stage. The type of cutting planes implemented has a bearing on the convergence speed of the algorithm (Subramanian et al., 2011).

### 2.3.2 Approximation Algorithms

The approximation methods aim to generate near-optimal solutions for a NP-hard CO problem within reasonable computing time. They are usually deployed whenever exact methods are not applicable. The extensive operational research work on approximation methods indicates that under certain circumstances researchers are willing to sacrifice solution quality in order to generate more timely solutions (Gendreau and Potvin, 2010). Approximation methods are able to generate near-optimal solutions for large problems using considerably less computational resources. However, many approximation methods employ stochastic procedures that are likely to generate different solutions with various properties when rerun, which is contrary to the previously mentioned exact counterparts.

The classical approximation methods consist of two sequential phases: solution construction and improvement. The construction phase builds an initial solution or a set of solutions for a problem, whilst trying to optimise a function. The construction phase aims to generate a feasible solution. However, computing a feasible solution to a NP-hard CO problem may be computationally expensive. Furthermore, there is no guarantee that the feasible solution is close to the optimal one. The improvement phase iteratively improves the initial solution by exploring its neighbouring solutions, until a termination criterion is satisfied. Ideally, this procedure will terminate once the optimal solution is found. For a comprehensive review on the discussed phases, the reader is referred to Braysy and Gendreau (2005a).

The following commonly studied approximation methods are discussed below: Hill Climbing (HC), Simulated Annealing (SA), Ant Colony Optimisation (ACO), Tabu Search (TS), Variable Neighbourhood Descent (VND) and Evolutionary Algorithms (Gendreau and Potvin, 2010). For a comprehensive review of the approximation heuristics, the reader is referred to Gendreau and Potvin (2010). Before proceeding to the discussion, some widely used terminology is explained. A search defines the action of identifying a feasible solution with respect to the

employed objectives in the considered problem. Whereas, a local search describes the process of exploring different neighbourhoods of an incumbent solution, in order to find an improved solution. Finally, the search space outlines a mathematical representation of the objectives in a Cartesian system of coordinates.

#### Hill Climbing Algorithm (HC)

This approach is a local search technique that explores the neighbouring search space of the incumbent solution and moves to a different one if the objective value is better. The HC analogy relates to a hill climb from the base to its peak. In the context of CO problems, the base refers to the initial solution and the peak is the best available solution. The principle of the climb is to ascend the hill by selecting a neighbouring solution, which is an improvement on the incumbent. The search space must be convex (for the comprehensive mathematical definition see Deb (2009)) for the ascent to occur. The algorithm terminates once all neighbourhood solutions fail to improve the incumbent quality. The resultant peak is likely to be a local optimum because HC is a greedy approach making a local optimum selection at each stage to replace the incumbent solution. There is no way of determining the distance between the local optimum from the global. The primary benefit of HC is its simplicity. However, the quality of the solution is dependent upon the initial solution. It is recommended to restart the procedure a number of times with a new initial solution, in order to explore various local optima, one of which will hopefully represent the global optimum.

#### Steepest Ascent Hill Climbing (SAHC)

SAHC is a well known variant of the HC, which explores all neighbours from the incumbent solution and selects the one with the greatest improvement. The purpose of this method is to advance more quickly up the hill and hopefully to a higher peak, however, this is not guaranteed.

#### Simulated Annealing (SA)

SA is a local search method that does not get easily trapped in a local optimum as it can accept a solution worse than the current one, thus improving the chances of finding the global optimum. This approach is derived from the procedure developed in Metropolis et al. (1953) and was first used in the context of CO problems by Kirkpatrick et al. (1983).

The SA approach is inspired by the physical process of annealing in Metallurgy. This process involves the heating of a metal in order to displace atoms from their current structures and then controlling the reduction in temperature for the purpose of generating a new crystalline structure of high density and with a minimum energy state.

Starting with an initial solution SA, similarly to the Hill Climbing algorithm, the search moves towards the neighbouring solution, if the objective values are improved. However, SA also considers the neighbouring solution with lower objective values with a probability of acceptance  $P_A$ , which is commonly defined as

$$P_A = \exp(-\Delta S/T) \quad (2.3).$$

The  $P_A$  is affected by the level of deterioration in the solution quality  $\Delta S$  between the incumbent and the new solution, as well as by the current permitted temperature or diversity level of the system,  $T$ . Initially,  $T$  is set to a high value and the resultant  $P_A$  values are larger, therefore encouraging exploratory steps in the search. Incrementally throughout the search,  $T$  is reduced and the resultant  $P_A$  is also diminished. Once  $T$  is reduced below a certain threshold, SA becomes more similar to hill climbing, as the relevance of  $\Delta S$  gradually diminishes.

The fine-tuning of parameters is critical for the success of SA. These include: a neighbourhood function, cost function, temperature reduction rate and the start state.

### Ant Colony Optimisation (ACO)

ACO is an optimisation technique inspired by the way a colony of ants is capable of finding the shortest path from their nest to their food sources. The principles of ACO were originally proposed by Dorigo et al. (1991) for the travelling salesman problem. In the natural world, ants lay down a chemical compound called pheromone as they travel. The resultant trail is used as a medium of communication by the ants to guide themselves to their closest food source. The intensity of the pheromone diminishes with time as the chemicals gradually evaporate, therefore the shorter routes have a more intense pheromone trail compared to longer counterparts. Subsequently, a larger volume of ants will travel on the shorter paths and gradually over time, all ants are directed onto the shortest path. This path will continue to be

used, until the food source is depleted, in which case the shortest alternative food source will be used.

The application of pheromone enables ants to adapt to changes in their surroundings. This ability has inspired the use of ACO algorithms to solve CO problems. Figure 2.1 defines the general outline of the ACO metaheuristic, as illustrated in Gendreau and Potvin (2010).

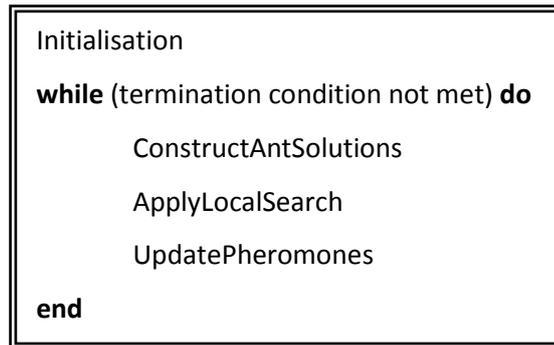


Figure 2.1 Ant colony optimisation outline

‘Initialisation’ requires the setting of parameters, i.e. the rate of pheromone distribution. ‘ConstructAntSolutions’ a population of homogenous artificial ants build various solutions taking into account the pheromone trails and other problem specific heuristic information. ‘ApplyLocalSearch’ applies a local search technique, which is recommended to improve the solution quality, (Gendreau and Potvin, 2010). ‘UpdatePheromones’ revises the pheromone trails, in order to guide the search towards greater accuracy or diversity by respectively using pheromone deposit and pheromone evaporation procedures.

### Tabu Search (TS)

TS is a memory based local search optimisation technique, originally proposed by Glover (1986). It is seen as a natural extension from local search techniques (Gendreau and Potvin, 2010). The purpose of this approach is to overcome local optima by tracking and guiding the search. The algorithm is made up of two complementary components: a local search technique and a short-term memory called tabu list. The purpose of local search is to increase the level of accuracy in the search; whereas the role of the tabu list is to promote greater diversity in the search.

The local search technique enumerates all possible transformations. The neighbouring solution that best improves the object function is selected to replace the incumbent solution.

The tabu list is a record of the recent transformations, which are prohibited from exploration for a number of iterations; this number is known as tabu tenure. The purpose of tabu list is to prevent the search from reversing the recent transformations, which have been applied to escape from the local optimum. Therefore, the tabu list aims to prevent revisiting certain areas of the search space and encourages exploration of new areas. However, the ability of the tabu list to prevent cycling depends on the list size and the way it is updated.

The size of the list can be constant or determined self adaptively. A constant tabu list maintains the same number of tabu transformations throughout the search. In order to maintain a constant list size, the older transformations have to be ejected to make space for newer transformations. A constant list size is commonly adopted because the amount of computation resources required for evaluations can be controlled. A tabu list with a self adaptive size implies the number of transformation inside the list depend on the status of the search. However, this may result in an exponential growth of the list size, which is computationally expensive.

A shortcoming of the tabu approach is that it restricts certain transformations which may not result in cycling. Furthermore, certain tabu transformations may be required to advance the search towards an area, which is more likely to feature high quality solutions. This issue is overcome with the use of an aspiration criterion, which overrides the tabu status of a transformation. A widely used aspiration criterion permits a tabu transformation if it leads to an improvement in the objective value because this solution has obviously not been visited before (Gendreau and Potvin, 2010).

For greater control of the search, intensification and diversification mechanisms may be used within TS. The intensification mechanism encourages further search in areas where good solutions are found. This requires the use of short term memory structures to record the components of the elite solutions. In contrast, the diversification mechanism prevents wasting computational time on a restrictive portion of the search space by guiding the search towards unexplored areas. This requires the use of long term memory structures, which record various

elite solutions throughout the search. The subsequent search is prohibited from further exploring similar areas.

#### Variable Neighbourhood Search (VNS)

Hansen and Mladenovic (1997) introduced VNS to tackle CO problems. The underlying principle of the approach is to change the neighbourhood structures of the local search technique (Gendreau and Potvin, 2010). The VNS algorithm repeatedly explores a set of neighbourhood structures, until a termination criterion is met. Initially, a neighbouring solution is randomly generated within a neighbourhood structure in the set. It is important that the neighbouring solution is diverse from the current neighbour, whilst maintaining the best traits of the current solution in order to generate an improved solution. A local search technique is applied to the neighbouring solution to find a local optimum. If an improved solution is found, it replaces the incumbent and the search is restarted from the first neighbourhood structure. Otherwise, the next neighbourhood structure is explored.

Gendreau and Potvin (2010) defined three key principles which give support to the VNS method design. The first relates to the fact that different neighbourhoods have their own local optimum. The second implies that a global minimum is also a local minimum relative to a given neighbourhood. The third states that it is common for several local/global minima to be closely situated in the search space, therefore the current local optimum may provide information on the global optimum.

#### Variable Neighbourhood Decent (VND)

VND is a variant of VNS. This method, unlike the previously mentioned one, applies the local search technique to the best neighbour in a set. The purpose of this approach is to invest the available computational resources in exploring the most promising neighbourhood, in an attempt to increase the likelihood of finding an improved solution. However, this approach does not guarantee the best neighbour will generate an improved lower bound compared with others in the set.

#### Evolutionary Algorithms (EA)

EAs are optimisation procedures, which mimic natural evolution. The main advantages of EAs are that they are capable of generating multiple solutions in a single run. This is a direct consequence of the fact that EAs evolve a population of potential solutions rather than one

single candidate. In the context of multi objective optimisation, this is a particularly useful feature of EAs, as they are capable of generating a set of objective tradeoffs simultaneously. A second point of appeal relative to EAs is that they do not employ any mathematical operations on the objective function (e.g. such as computing the derivative of the objective function(s) in gradient based approaches) that would entail supplementary applicability restrictions. This way, EAs may be employed to solve optimisation problems defined over irregular search spaces (characterised by nonlinearity, discontinuity, noise and/or multimodality). Thirdly, EAs employ stochastic transitions to move from one generation of potential solutions to the next. Therefore, the potential of local optima blockage is significantly diminished. The fourth advantage of EAs stems from their stochastic nature. It consists in the fact that the number of algorithm parameters which need to be configured prior to algorithm development is expectedly lower in relation to other alternative optimisation methods (Eiben and Smith, 2003), (Koza, 1998). In addition, the previously mentioned algorithm parameters may be assigned a given value as a result of a trial and error line of experiments. This significantly reduces the involvement of the human practitioner in algorithm configuration. In the case of evolutionary strategies, to be described shortly, the algorithm related parameters (max number of generations, genetic operators, applicability probability, chromosome life span, etc.) are encrypted in the actual genetic code. Thus, they are evolved along side the rest of the genetic material encrypting information relative to the solution structure. In consequence, the human practitioner involvement in algorithm configuration becomes minimal. The fifth aspect worthy of mentioning refers to EA versatility, in terms that the algorithm can be accurately configured to achieve a desired balance between exploration (discovering new areas of the search space) and exploitation (using fit genetic material from previous generation solutions and improving it in order to generate potentially better adapted offspring).

It is noteworthy that the stochastic nature of EAs is the cause of several disadvantages. Primarily, there is no guarantee that EAs will converge to the optimal solution (Coello Coello et al., 2007). However, this issue may be overcome by applying an EA to narrow down the search space to the envisaged optima and then employing a deterministic technique to further refine the results (Deb, 2009). Furthermore, it is challenging to describe the probabilistic mechanisms, at work inside EAs, within a consistent mathematical model. In consequence, no complete mathematical description of the phenomena inherent to EA behaviour is currently available in the literature. In addition to that, EAs are computationally intensive, meaning that

the stochastic transitions they imply, alongside the chromosome evaluation stage, take up considerable resources (run time and memory).

The following are the four main EA paradigms: genetic algorithms (GA), genetic programming (GP), evolutionary strategies (ES) and cultural algorithms (CA). The primary difference between these approaches is the way each sub class encrypts a solution.

The term Genetic Algorithm (GA) was introduced by Holland (1975) and refers to a global search heuristic motivated by natural genetics. This approach is a widely selected for solving combinatorial optimisation problems in a variety of problem domains (Gendreau and Potvin, 2010). The fundamental features of a GA include: encoding, evaluation, selection for reproduction, reproduction, selection for reinsertion and a termination condition. The GA encodes a population of individuals in order to investigate a wide range of points in the search space. A potential solution is defined by mapping decision variables inside a linear chromosome structure. The populated individuals are evaluated in terms of their objective values and selection probabilities for reproduction are accordingly assigned. The genetic make-up of individuals selected from the population is altered by genetic operators, in order to identify new and expectedly improved solutions. The individuals passed onto the next generation are selected by means of a reinsertion mechanism. Finally, the stochastic search continues, until a termination condition is satisfied. The GA framework is discussed extensively in Chapter 4 because this optimisation approach is grounds for a substantial part of this research methodology.

In some cases the encoding of a complex structure (e.g. function approximation) requiring adaptation is not best represented as a linear structure, but instead as a hierarchical structure. As stated by Koza (1998), Genetic Programming (GP) is able to encode hierarchical chromosomes using tree structures and graphs, unlike GA. One of the main advantages of hierarchical encoding compared to linear encoding is that unexpected dependencies between decision variables can be found easier than in the linear case (Patelli, 2011). For a comprehensive outlook on GP, see (Poli et al., 2008).

Evolutionary Strategies (ES) were originally proposed by Rechenberg and Schwefel in the early 1960s (Eiben and Smith, 2003). This approach encrypts decision variables inside the chromosome in a similar fashion to GA, but in addition encodes a set of strategy parameters

(chromosome life span, genetic operator, applicability probability, intensity rate, etc.), (Patelli, 2011), (Eiben and Smith, 2003). In the offspring generation phase, algorithm parameters are evolved alongside decision variables, therefore this approach is self adaptive. The main advantage of this approach is that the human practitioner involvement in algorithmic configuration becomes minimal.

Reynolds and Sverdlik (1994) first introduced Cultural Algorithms (CA) based on the concept that individuals could evolve and adapt much faster to changes in their environment, by integrating a cultural element into the evolutionary process. This approach consists of three essential components: population space, belief space and communication channel (Reynolds and Sverdlik, 1994). The population space contains the current discrete values of the employed objective functions. The belief space comprises of a set of individuals defining the likely vicinity of the optimal solution. This information is used in the current generation to direct the search in the desired direction by disregarding individuals situated in sub-optimal areas of the search space (Patelli, 2011). Finally, the communication channel allows information to be exchanged between the belief and the population spaces.

This chapter has introduced the combinatorial optimisation (CO) problem and existing approaches. The multi-objective CO problem of interest to this thesis is the vehicle routing problem with simultaneous delivery and pickup (VRPSDP), which is known to be NP-hard (Dethloff, 2001). Several well-known exact and approximation methods used to solve CO problems have been discussed. The approximation methods are found to be favoured over exact methods as they provide an improved trade-off between solution quality and computational effort. In particular, evolution algorithms (EAs) are considered to be a superior approximation method for multi objective optimisation, over other methods for a number of reasons. Primarily, EAs generate multiple solutions in a single run, which is advantageous for solving a multi-objective problem. The initial starting point of an optimisation method has a significant impact on the ability of the search to converge to the optimal solution (Deb, 2009), (Coello Coello et al., 2007). Since EAs simultaneously evolve a population of solutions they have a better likelihood of convergence to the optimal search space area and a lower risk of local optimum blockage compared to previously discussed approximation approaches, which explore a single point in the search space at each step (Goldberg, 1989). However, an argument could be made to run the point to point combinatorial optimisation algorithms in parallel, although this is not an efficient approach because the complete benefit of parallel

computing cannot be realised (Deb, 2009). For instance, the parallel searches are run in isolation to one another and no communication channel exists to exploit the incumbent information. This is contrary to EAs where solutions are combined and evaluated against one another for the purpose of search progression. In particular, the sexual reproduction phase is unique to EAs and is suggested to provide a potentially aggressive source of search space exploration (De Jong, 2006), which is not foreseeable with the other discussed approaches. On another important aspect is that the EA paradigm is better mapped for achieving the MOO goal of determining the global optimal set than the point to point CO methods because it generates multiple potential solutions simultaneously in a single run. The aforementioned points demonstrate why the EA paradigm is increasingly being applied to real world multi objective optimisation problems (Deb, 2009).

The EA paradigm selected for a specific problem is dependent upon the encoding type required. The EA of importance to this research is the genetic algorithm (GA) because it effectively encodes the problem considered here. A detailed background on GA is provided in Chapter 4.

A hybrid intelligent algorithm is not considered in this research because a standalone EA is capable of addressing the specific problem requirements and thereby effectively programming the search. A hybrid intelligent algorithm consists of two or more algorithms that work collectively to solve a particular problem. This type of symbiosis one method designed to compensate for the setbacks of the others. For example, Crispim and Brandao (2005) introduced a variable neighbourhood descent algorithm to address the lack of neighbourhood exploration by the tabu search algorithm. However, EAs are very adaptable because via simple configuration and tuning, the desirable level of accuracy and diversity can be introduced into the search. This way the additional computational complexity entailed by hybridising the EA with another method can be avoided.

### 3. THE DELIVERY AND PICKUP PROBLEM

This chapter begins by introducing the reader to the most commonly studied transportation problems, which relate to the service of a single demand type (delivery or pickup). This background information will help define the problem of interest, which relates to a Delivery and Pickup Problem (DPP) formulated as the Vehicle Routing Problem with Simultaneous Delivery and Pickup (VRPSDP). The primary purpose of this chapter is to provide a state of the art of the VRPSDP literature.

This chapter has the following outline. Section 3.1 defines the simplest routing problem known as the Travelling Salesman Problem (TSP). Section 3.2 introduces the Vehicle Routing Problem with some of the most well known variants. The Delivery and Pickup Problem (DPP) is discussed in Section 3.3, where a comprehensive state of the art is provided for the VRPSDP literature. The chapter is concluded with Table 3.1, which summarises the literature.

#### 3.1 Travelling Salesman Problem (TSP)

The travelling salesman problem (TSP) is the simplest type of node routing problem (Gokce, 2004) and was defined by the Irish mathematician W.R. Hamilton in the 1800s. The TSP is defined as finding the minimum cost (distance) Hamilton cycle for a set of nodes (cities). A Hamilton cycle describes a closed path that visits every node in a set, exactly once. This problem is proven to be NP-hard by Karp (1972). Figure 3.1a illustrates the TSP and Figure 3.1b defines a potential solution for the problem.

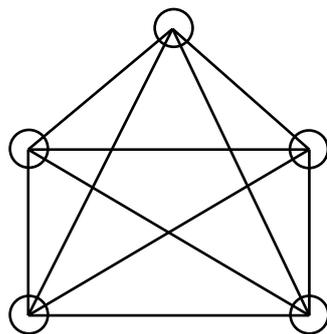


Figure 3.1a

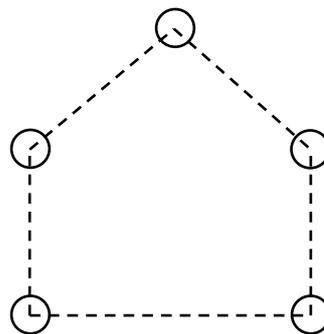


Figure 3.1b

Figure 3.1a is a complete graph, where the circles are vertices/nodes and the lines represent edges/arcs. Figure 3.1b defines the least cost Hamilton cycle for the TSP.

The TSP can be described based on its symmetry. Hence, there are symmetric (STSP) and asymmetric (ATSP) TSPs. The symmetric problem type assumes that the cost of traversing the edge is the same in both directions. The asymmetric problem type is modelled on a digraph, as the cost of traversing the edge is different in each direction.

There is an electronic library of broad test problems dedicated to the TSP, known as the TSPLIB, which was founded by Reinelt (1991). This standardisation has assisted the advancement of finding effective solution methods for the TSP, (Morgan, 2008). The reader is referred to Applegate et al. (2006) and Gutin and Punnen (2004) for detailed information on TSP and its variants like MAX TSP and TSP with multiple visits (TSPM).

### **3.2 Vehicle Routing Problem (VRP)**

The vehicle routing problem (VRP) is a combination of the bin packing problem (BPP) and the TSP (Tasan and Gen, 2012). The BPP consists of scheduling items to the minimum number of bins in a set, without exceeding the bin capacity (Toth and Vigo, 2002). The VRP is defined as finding a set of Hamilton cycles that minimise the cost of transportation. These cycles form individual driver routes and must originate from and terminate at a depot. Furthermore, the cycles must fulfil a set of demand requests, whilst abiding by certain operational constraints Toth and Vigo (2002).

In order to solve the VRP, decisions on the assignment and routing aspects of the problem are required (Alonso et al., 2008), which in turn affect the cost of transport: the number of vehicles operated and the total distance travelled. This decision process is complex, thus making VRP one of the most studied CO problems (Golden et al., 2008). A rich body of literature exists for various VRP variants, which is motivated by both its practical relevance and considerable difficulty (Toth and Vigo, 2002). It has been established in the literature that the basic VRP is a NP-hard problem (Golden et al., 2008), (Toth and Vigo, 2002).

Figure 3.2 depicts a solution for a VRP problem, where three vehicles service 10 customers from the depot.

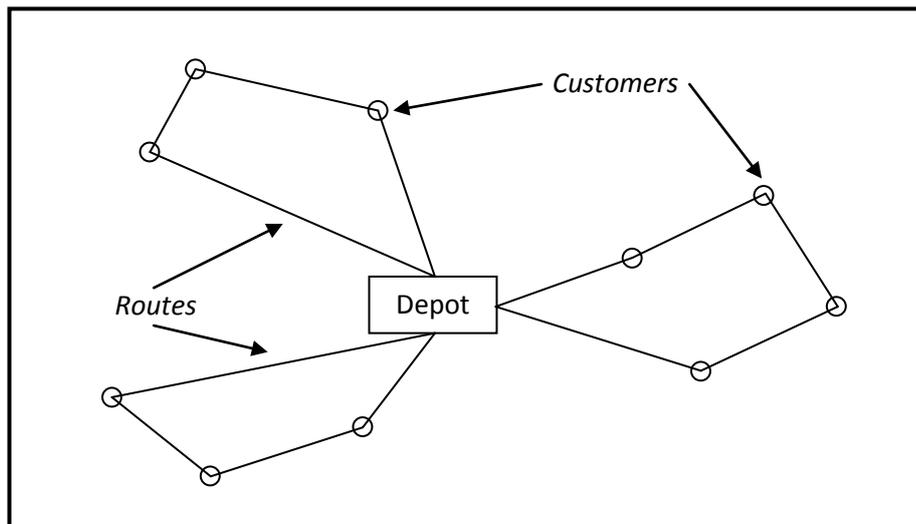


Figure 3.2 VRP Solution

### 3.2.1 Capacitated Vehicle Routing Problem (CVRP)

The VRP is commonly formulated with respect to vehicle capacity constraints and is therefore generally known as the capacitated vehicle routing problem (CVRP). The CVRP is the simplest and most studied VRP problem, (Toth and Vigo, 2002), since it was introduced by Dantzig and Ramser (1959). It is represented as the following graph theoretical problem. A complete graph  $G = (V, A)$  comprises of a set of vertices  $V = \{0, 1, \dots, n\}$  connected by a set of edges  $A$ . The vertex 0 denotes the depot and the vertices  $j = 1, \dots, n$  correspond to the customers. Each customer is associated with a non-negative demand  $d_j$ , which is to be supplied from the depot. A non-negative routing cost  $c_{ij}$  is associated with every arc  $(i, j) \in A$ , which respectively connects vertex  $i$  to vertex  $j$ . Depending on the problem type the cost matrix is either: symmetric,  $c_{ij} = c_{ji}$  or asymmetric,  $c_{ij} \neq c_{ji}$ . A symmetric cost matrix based on Euclidean distances ensures that the direct distance between two vertices  $i$  and  $j$  is either equal to or shorter than an indirect path, such that  $c_{ik} + c_{kj} \geq c_{ij}$  for all  $i, j, k \in V$ . This is referred to as triangle inequality and in several practical instances it is not convenient to deviate from such a path, (Toth and Vigo, 2002). A set of homogeneous vehicles  $K = 1, \dots, k$  are stationed at the depot with a capacity of  $Q$ , such that  $d_j \leq Q$ .

The CVRP involves finding a set of  $k$  Hamilton cycles for delivery vehicles of fixed capacities, which operate from a central distribution centre, to supply a set of customers with deterministic locations and demands for a certain commodity, such that:

- i. all Hamilton cycles originate and terminate at vertex 0,
- ii. every vertex  $j \in V \setminus \{0\}$  is visited by exactly one Hamilton cycle,
- iii. the sum of the vertex demands  $d_j$  on any Hamilton cycle cannot exceed the capacity  $Q$  of the vehicle.

There are normally two objectives associated with the VRP and its variants, which are required to be minimised:

- operated vehicle fleet size,
- total distance travelled.

### 3.2.2 Vehicle Routing Problem with Time Windows (VRPTW)

The vehicle routing problem with time windows (VRPTW) is derived from the inclusion of time window constraints in the CVRP. This is a well studied problem (Hosny, 2010), because most real world transport problems imply time window constraints, for example, supermarket delivery within a given time slot. In this problem, a prearranged service time window interval is associated with every request. Therefore, the service of a request is only permitted within the time interval. The driver may arrive at the customer premise before the beginning of the time interval, however, must wait until the start of the interval before the service can commence. If the driver arrives after the end of the interval, this results in the violation of operational constraints therefore, causing solution infeasibility. Furthermore, the depot vertex has a time window interval, where every vehicle must leave after a certain time and return before a certain time. The objectives of the VRPTW are the same as for CVRP.

VRPTW is modelled as an asymmetric problem (Toth and Vigo, 2002) because the reverse of the route may lead to infeasibility due to the time window constraints being violated. An outline of the exact approaches for the VRPTW can be found in Cordeau et al. (2001) and a two part survey on the heuristic approaches can be found in Braysy and Gendreau (2005a,b).

### 3.3 Delivery and Pickup Problem (DPP)

The delivery and pickup problem (DPP) is an extension of the CVRP, where the pickup demands share the same route as delivery demands, or vice versa. This problem is commercially motivated by the efficiencies, which can be realised from servicing delivery and pickup demands on the same route. The theoretical point of interest is derived from the fact that bi-directional transportation is the most challenging task inside a closed loop supply chain

(Wang and Chen, 2012). The DPP may be formulated in different ways: vehicle routing problem with backhauls (VRPB), vehicle routing problem with mixed backhauls (VRPMB) and vehicle routing problem with simultaneous delivery and pickup (VRPSDP). The aforementioned formulations are a special case of CVRP, when either the delivery or pickup demands equal zero, therefore any DPP formulation is considered NP-hard. A detailed survey on the DPP can be found in Berbeglia et al. (2007).

The vehicle routing problem with backhauls (VRPB) was first addressed by Deif and Bodin (1984). This formulation provides a servicing restriction that prioritises the service of delivery requests before the fulfilment of pickup requests. This restriction reduces the complexity of the DPP. In the past, this was a practical formulation because vehicles only provided rear door access to the storage area, which made pickups prior to the service of deliveries extremely difficult (Toth and Vigo, 2002). Another reason for the adoption of this formulation is that in certain situations, delivery requests have a higher priority compared to pickup requests (Toth and Vigo, 2002). For instance, an engineering firm may require the delivery of a product for repair before a collection can be arranged for the restored product.

The reader is referred to the following studies relating to VRPB for further information: Liu and Chung (2009), Parragh et al. (2008), Brandao (2006), Ropke and Pisinger (2006), Toth and Vigo (1997), Halse (1992) and Goetschalckx and Jacobs-Blecha (1989).

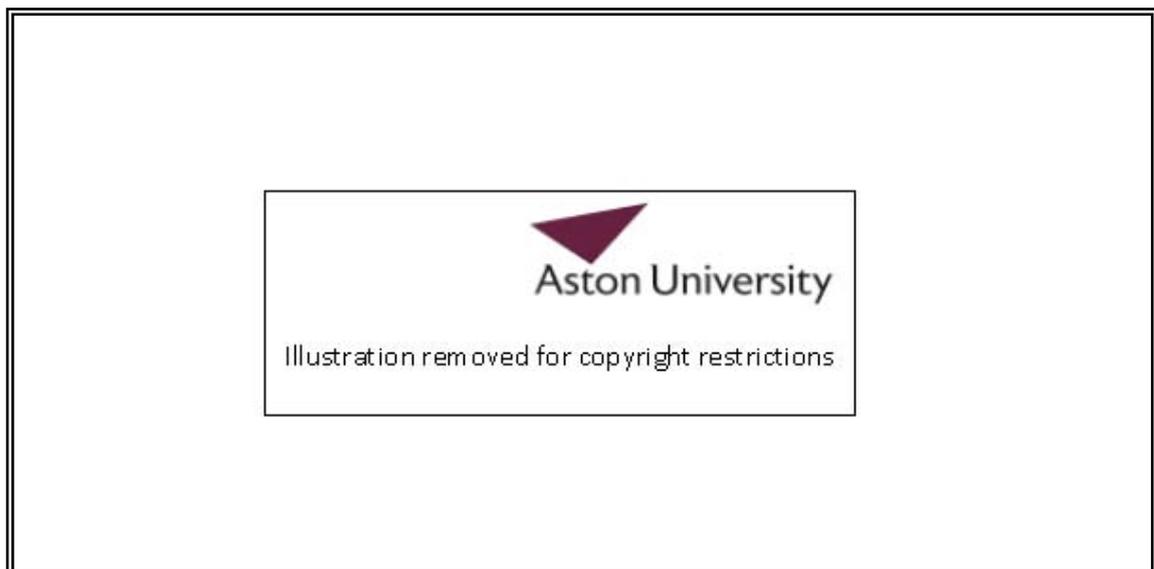
The introduction of vehicles with access through a slide panel has reduced the complexity of organising onboard loads, which has led to the formulation of the vehicle routing problem with mixed backhauls (VRPMB). Therefore, VRPMB does not consider the restriction in VRPB that prohibits the service of deliveries prior to pickups. This greater flexibility in terms of sequencing enables improved solutions to be found, in terms of transportation cost. However, this formulation has increased complexity to the VRPB as the vehicle load is not monotonically increasing or decreasing along the route. Therefore, maintaining vehicle capacity feasibility becomes an issue.

The reader is referred to the following studies for further information on the VRPMB domain: Parragh et al. (2008), Crispim and Brandao (2005), Sural and Bookbinder (2003) and Dethloff (2002).

The vehicle routing problem with simultaneous delivery and pickup (VRPSDP) is an extension of the VRPMB, which requires the delivery and pickup demand at a customer location to be serviced in a single visit. This is in contrast to the VRPB and VRPMB, which insist on an independent visit for the delivery and pickup service. In consequence, the complexity of this formulation is greater than VRPMB because fluctuating capacities have to be considered at each node. However, the VRPSDP formulation generates greater benefits than the previously mentioned formulations. In relation to the service operator, a reduction in the cost of the operation may be achieved, in terms of either a smaller vehicle fleet size being operated and/or a reduction in the total distance travelled. In addition, operational capacity is likely to be increased, therefore, more requests can be included into the routes. In terms of the environment, less CO<sub>2</sub> is being emitted per unit of economic output. For the service recipient, fewer resources need to be allocated when in receipt of the service, therefore reducing their handling effort.

There are many commercial applications for the VRPSDP (Zachariadis et al., 2009), which would benefit from an efficient solution procedure. These include: parcel service industry plan routes for vehicles that deliver and collect parcels, soft drink distribution of drinks and collection of empty bottles and the laundry service for restaurants and hotels.

Subramanian et al. (2010b) have defined the VRPSDP problem as follows:



The VRPSDP problem was first introduced in 1989, yet, it received limited attention in the 1990s (Subramanian et al., 2010a). However, in the recent decade the problem has seen much attention (Wang and Chen, 2012), as the importance of reverse logistics is being

acknowledged. The solution approaches can be broadly classified as exact and approximation methods. The literature is dominated by approximation approaches, (Subramanian et al., 2010a), which better manage the tradeoffs between solution quality and computational resource consumption, when compared to exact methods. The early algorithms for the VRPSDP are based on the simple construction and improvement heuristics; whereas the present approaches can be described as metaheuristics (Parragh et al., 2008). The latter type refers to solution methods that combine local improvement procedures with strategies to escape from the local optimum. The remaining part of this chapter is dedicated to provide a state of the art for the research work on solving VRPSDP, summarised in Table 3.1.

### 3.3.1 Exact Methods

#### Branch and Price

Angelelli and Mansini (2002) proposed the first exact method for the vehicle routing problem with simultaneous delivery and pickup among time windows (VRPSDPTW). A branch and price algorithm based on the set covering formulation was developed. The set covering formulation is used for the master problem because it is numerically more stable than linear programming relaxation. The authors tested different branching and pricing strategies for their effectiveness.

From a critical point of view, the approach is only able to solve to optimality instances comprising of up to 20 customers, (Subramanian et al., 2010b). In addition, Angelelli and Mansini (2002) choose not to modify previously published VRPSDP test problems like Salhi and Nagy (1999) and Dethloff (2001) in order to consider VRPSDPTW, therefore causing fragmentation in the field.

Dell'Amico et al. (2006) also applied a branch and price algorithm to solve the VRPSDP problem. The authors tested two different strategies to solve the pricing subproblem: exact dynamic programming and state space relaxation. The branch and price algorithm based on state space relaxation proved to be more competitive because it consumed substantially less computational time than the exact dynamic programming approach. Furthermore, the same approach solved the majority of test problems to optimality.

From a critical stand point, a tabu search algorithm was used in the initialisation stage, therefore the quality of the resultant solution will have a bearing on the performance of the branch and bound algorithm. Another aspect of importance relates to the size of the test problems used for evaluation. The authors believe their approach is viable for small to medium size instances. However, the size of their largest test problem (40 customers) is smaller than the smallest considered in Salhi and Nagy (1999) benchmark instances. Furthermore, a large number of vehicles are operated to service a relatively small number of requests, therefore the routes operated are small in terms of customers, which results in a simpler routing problem. This is in contrast to Salhi and Nagy (1999) test problems, where the number of requests on the route is larger, thus making them more difficult.

### Branch and Cut

Subramanian et al. (2010b) proposed an undirected and directed two commodity flow formulation for the VRPSDP. These approaches were developed to provide stronger inequalities by tightening the bounds of the flow variables. These formulations were tested for their superiority using a branch and cut algorithm. On average, the undirected two commodity flow formulation better performed in terms of solution quality for all test problem sets.

It may be noteworthy that the approach is very computationally expensive when compared with other publicised heuristic approaches. A contributing factor to the slow speed is the use of auxiliary flows like additional variables and constraints in the formulation to control vehicle capacity (Subramanian et al., 2011).

Subramanian et al. (2011) introduce a branch and cut algorithm to solve the VRPSDP. Unlike Dell'Amico et al. (2006) and Subramanian et al. (2010b), which used auxiliary flows to maintain vehicle capacity feasibility along the route, the authors considered vehicle capacity constraints in the middle of the route in a relaxed manner inside the branch and cut algorithm. This approach is introduced to reduce the computational resource consumption.

From a critical view point, the authors publish their lower bound solutions, but not their upper bound ones. Therefore, it is difficult to evaluate the performance of their method with respect to producing consistent quality solutions. On an alternative issue, the number of vehicles operated for Salhi and Nagy (1999) test problems: CMT3Y, CMT4Y and CMT5Y are higher than

what is physically required to service the demand. Therefore, a discrepancy in the way the test problem is generated by the authors might exist.

### 3.3.2 Heuristics

#### Cluster-first and route-second approach

Min (1989) first introduced the VRPSDP concept to handle the real world problem involving the collection and distribution of books between 22 local libraries and a central library using two vehicles. A cluster first and route second approach was adopted. The first phase constructed a capacity feasible cluster for each route and the second phase solved the respective routing problem using a branch and bound algorithm. Route capacity feasibility was obtained by iteratively solving the routing problem with the use of arc penalties, which penalised unacceptable delivery/pickup sequences.

The approach was evaluated using actual routes, which were determined manually by the driver. The proposed method substantially improved the solution quality of the actual routes. This improvement was suggested to be a consequence of the cluster technique, which aimed to minimise the overlapping of routes.

Analysing the contribution of the paper, one might assume that the use of penalties is a good technique to enable alternative routes to be found. However, it may be said that the routing method applied a large amount of effort in obtaining route capacity feasibility and little effort in optimising the routes. From a different stand point, a clustering phase that initially ensures capacity feasibility is important because otherwise, computational resource may be wasted in trying to obtain feasibility, when it is not possible. Another aspect of importance is that the presented results were limited and a small problem is solved using only two vehicles.

Dethloff (2001) highlighted the importance of VRPSDP to reverse logistics. The author introduced a “cluster first and route second” algorithm. The clustering phase incrementally inserts requests into a growing route, until no further insertions are feasible. This process is repeated with an additional route and so forth, until all requests have been assigned. The aforementioned method is repeated with each request serving as an initial seed and the solution with the lowest total route time is selected as the final solution.

The solution quality of the algorithm depends primarily on the insertion method. The proposed insertion method is an extension of the cheapest insertion method, which considers several metrics: travel distance (TD), residual capacities (RC) and residual surcharges (RS). The RC and RS were considered in order to prevent greedy insertions based only on TD. RC considered the additional amount which could be delivered or picked up following the insertion of a customer, therefore the effect of the insertion on the vehicle capacity is considered at each stage. RS introduced a penalty for the late insertion of a distant customer, to circumvent against unfavourable travel distances, which may have otherwise been prevented.

Dethloff (2001) was the first to propose a mathematical model for the VRPSDP, which has subsequently enabled other authors to develop several solutions to the problem. In addition, the concept of RC and RS seems to be an important contribution because future consequences of the current insertion decisions are taken into account, which is experimentally proven to yield better results. On a different issue of importance, by running the method separately for all nodes, a number of solutions can be found, however this may be computationally expensive if a large number of nodes exists. With respect to the computational results, the number of vehicles operated exceeds the required size for Salhi and Nagy (1999) datasets: CMT2X, CMT2Y, CMT5X and CMT5Y. In addition, the results indicate that the algorithm is not effective in solving instances with a route length constraint.

Montane and Galvao (2002) proposed eight heuristic procedures for the VRPSDP, which all cluster first and route second. These heuristics differed in terms of the clustering approach, initial routing procedure and the TSPSDP heuristic used. The clusters were constructed using either a tour partitioning heuristic or the sweep algorithm. The initial route was constructed using either the 2-optimal or 3-optimal procedure. The TSPSDP problem was solved using one of the four TSPSDP heuristics introduced in their work.

From a critical point of view, relative to workload balancing, the sweep algorithm implies that the last route to be designed is likely to have less vehicle capacity utilisation than the rest of the fleet. On a different issue of importance, there is no information available in the paper with respect to the independent evaluation of the four TSPSDP heuristics and the selection of the best one to be combined with the assignment procedures. From a different stand point,

there is no information available relating to the performance of the proposed heuristics on benchmark test problems.

### 3.3.3 Metaheuristics

#### Tabu Search

Montane and Galvao (2006) proposed a tabu search algorithm to solve the VRPSDP. A new neighbourhood was generated using inter-route operators. The resultant neighbourhood was iteratively searched by an intra-route operator 2-opt, until no further route improvement was possible.

Related to the aspect that every neighbourhood was completely searched and the most feasible neighbourhood move was selected for further improvement using the 2-opt operator. From a critical point of view, the advantage of this strategy is that computational resources are being targeted towards the most promising areas. However, there is a possibility that the less feasible neighbourhoods could have generated an even better solution, if the 2-opt operator had been applied to them. However, this approach would most likely require more computational resource consumption. On a different issue of importance, a tour partitioning method was proposed to construct the initial solution, and was repeated at different starting nodes with the aim of improving the solution quality. The paper has not described the method used for selecting the different starting nodes or the number of times a new starting node is selected, therefore the contribution provided in terms of the tour partitioning method cannot be quantified.

Wassan et al. (2008) introduced a Reactive Tabu Search (RTS-VRPSDP) algorithm. The algorithm differed from similar approaches because the tabu list size was dynamically determined, where the size was dependent on the number of solution repetitions. The paper argued that a dynamic list better guided the search compared to a fixed list size.

The initial solution was constructed using a modified version of the sweep algorithm introduced by Gillet and Miller (1974). The construction approach prevented the requests immediately surrounding the depot from the sweep process, which were individually serviced on their own route. This modification provided greater flexibility in the optimisation phase, which was anticipated to generate improved solutions. The forward and backward sweep was

performed on each node and the best solution was selected for improvement using local search operators. In particular, the reverse operator had no bearing on the solution quality, but was used to reduce the maximum capacity on the route, therefore creating additional slack for further insertions.

Analysing the contribution of the paper one might assume that the exclusion of requests closest to the depot is a sensible approach because this provides greater assignment possibilities, since all drivers will depart and converge at the depot. The percentage of requests excluded from the sweep process was experimentally determined. However, Wassan et al. (2008) failed to provide details on how this parameter affected the search. On a different issue of importance, the sweep process is performed from each node, therefore this approach provides a range of solutions to possibly optimise.

It is noteworthy that the reverse operator may reduce the maximum node capacity on the route, however, it may not reduce the capacity at the point where the insertion is needed. Furthermore, additional sections of the route may become greater constrained, therefore increasing the difficulty of insertion. On a different issue of importance, the approach used by Wassan et al. (2008) to derive Salhi and Nagy (1999) CMT1Y dataset might vary from other authors because their best known solution is lower than the optimum (Subramanian et al., 2010a).

#### Ant Colony Optimisation

Gokce (2004) proposed an ant colony system based algorithm to solve the VRPSDP problem. The author chose to update the candidate list during the search in order to make better routing decisions, which would lead to shorter route distances. The visibility function, apart from considering the travel distance between two customers, also took into account the distance from the depot and the associated time window of the next customer to be visited. These characteristics led to the construction of better quality routes. Subsequently, a 2-opt procedure was applied to the best routes in order to increase accuracy in the search.

The initial routes were constructed in a sequential manner, in order to ensure vehicle capacity feasibility. However, it is noteworthy that this approach does not guarantee the smallest vehicle fleet size will be operated. In addition, the application of the 2-opt procedure on the best routes does not guarantee a more substantial increase in search accuracy. It is plausible

that lesser routes combined with the 2-opt procedure may lead to more improved routes. On a different issue of importance, the consideration of the customer time window in the visibility function is an important contribution, as it emphasises on the service fulfilment conditions during the search. However, the weighting of the characteristic should not be too dominant, for the fear that it will prohibit a lower cost solution from being found.

Gajpal and Abad (2009) introduced an Ant Colony System (ACS) algorithm. The nearest neighbour algorithm was used to determine the initial solution, which initialised the trail intensities. The ACS generated ant solutions that were improved using two inter-route operators and an intra-route operator. The trail intensities of the ants were updated based on their current solution quality, therefore providing extra weight to the best solution. This increased the accuracy in the search.

The amount of computational resource consumed was affected by the number of ants used. The one used by Gajpal and Abad (2009) was twice the number of vehicles operated. However, from a critical prospective, the author had not shown any experimental justification for the decision. On a different standpoint, the manner trail intensities were updated was at a loss to diversity, which may result in the search being trapped at the local optimum. In addition, Gajpal and Abad (2009) did not mention the number of vehicles in their best solutions (Subramanian et al., 2011), therefore a direct comparison between their algorithm and previously published methods is not possible.

Catay (2010) proposed an Ant Colony Optimisation (ACO) algorithm. This algorithm introduced a new savings based visibility function and a rank-based pheromone update procedure, which were considered during the selection of the next point in the search space to be visited by the ant. The purpose of the saving-based visibility function was to service neighbouring nodes before proceeding to another area of the search space, therefore the search space can be explored more efficiently because the previously explored space is not revisited. In relation to the rank-based pheromone update procedure, the level of pheromone deposited depends on the rank of the solution, with higher order solutions depositing greater pheromone on solution components. The solutions with the greatest rank were the best in the current generation. This pheromone depositing approach encouraged the exploration of the search space areas hosting the best known solutions.

The initial routes were constructed using the nearest neighbourhood method. From a critical point of view, this method could ensure route capacity feasibility, but at the cost of having no control over the vehicle fleet size. If a vehicle fleet size cap is enforced, then feasibility is not guaranteed. On a different issue of importance, the pheromone deposit approach was elitist, as it encouraged a shift of the search process towards the areas of the previously defined good solutions. Another advantage of the technique is that it learns from previous generations. However, it is important that a strong diversity mechanism is available to prevent the search being trapped in the local optimum.

#### Particle Swarm Optimisation

Ai and Kachitvichyanukul (2009) presented a Particle Swarm Optimisation (PSO) algorithm. The routes were constructed from seed locations using a cheap insertion heuristic and optimised after each insertion with 2-opt procedure. The approach considered the spatial proximity of requests to the seeds during the route construction phase because a minimisation of the total routing distance was believed to be a consequence.

Considering the paper in a general context, it might be concluded that clustering requests based on the spatial proximity to a set of seeds is a logical concept and is believed to have contributed to the performance of the approach. However, for this method to be applicable, there should be no breaks in the network. In addition, the right spatial spread of the seeds is required, otherwise poor route assignments will be generated. From a different stand point, Ai and Kachitvichyanukul (2009) stated that the PSO algorithm is scalable because computational efficiency is maintained. However, the authors have not evaluated the performance of the algorithm when scaled using the benchmark test problems of Montane and Galvao (2006), which consider up to 400 customers.

Kanthavel et al. (2012) introduced a nested Particle Swarm Optimisation (NPSO) algorithm, which consisted of two phases: the first clustered assignments using the sweep algorithm and the second attempted to find the least cost routes for the assignments.

From a critical point of view, a consequence of the sweep algorithm is that the last route will contain fewer assignments because the previous clusters have maximised their own assignments. In circumstances where workload balancing is an issue, such an approach may prove to be sub-optimal. Furthermore, if the search space is large, this will result in elongated

routes, which are operationally not desirable because customers are likely to be located too far apart. On a different issue of importance, Kanthavel et al. (2012) has made no reference to the operated fleet size for any of the test problems studied, therefore a direct comparison cannot be made with previously published computational results. In addition, the average solution quality over the 5 runs, an indicator of the efficiency of the approach, is not provided in the article.

### Simulated Annealing

Attempting to solve the transportation problem for blood banks, Ganesh and Narendran (2008) introduced a route construction and improvement approach called TASTE to solve the single vehicle VRPSDP problem. The initial route was constructed to ensure that a set of negative net load nodes were serviced before a set of positive net load nodes. This route sequencing approach increased the likelihood that a capacity feasible route would be determined, because the fulfilment of deliveries took precedent over pickups. An enhanced simulated annealing (SA) algorithm (ESA) was proposed to improve the initial solution. The ESA determined a unique cooling schedule (T) for each route depending upon the solution quality compared to its counterparts. Therefore, the better routes were assigned a higher value of T, thus having a greater probability of being selected.

It is notable that the node sequencing restriction in the first phase may prevent the construction of a least cost route from being determined. Another aspect of importance relates to the fact that ESA is an elitist approach, which increases the probability of local optima blockage.

### Evolutionary Algorithms

Vural (2003) proposed a Dual GA to solve the VRPSDP. The initial population was constructed using the random keys method, which encoded a solution with random numbers. This encoding method provided a greater level of diversity, which improved the robustness of the algorithm. The author applied an intra-route procedure called Or-opt in order to reduce the routing distance, which increased the level of accuracy in the search. An adaptive mutation operator was proposed, which increased the probability of mutation with every generation an improved solution was not found.

From a critical perspective, an argument can be made that the Or-opt procedure is not required because the crossover operator tends to improve accuracy anyway. A possible reason for the use of this method is that the crossover operator is followed by the mutation operator, which diminishes the accuracy in the search. In addition, the fitness function is based on the total route distance travelled by a set of vehicles and the number of vehicles operated is not considered. In consequence, the selection process may favour chromosomes with a larger vehicle fleet size than necessary. Moreover, Vural (2003) omitted the computational times of the Dual GA for Dethloff (2001) test problems, therefore the efficiency of the algorithm cannot be analysed. However, the author has mentioned that the algorithm is not computationally competitive with other heuristic methods.

Vural (2007) proposed a GA approach that consisted of three phases: construction, genetic operators and route improvement. The author used integer encoding to reduce the effort required in generating meaningful phenotypes from binary representation.

A multi start greedy randomised adaptive search procedure (GRASP) heuristic was used to construct the initial population. This method built routes in a sequential manner in order to form a solution. A sequential version of GRASP heuristic was used instead of a parallel version for two reasons: in order to minimise the vehicle fleet size and to prevent the need to reduce the vehicle fleet size due to excessive vehicle capacity slack.

A relaxed version of GRASP heuristic was used to introduce new individuals into the population, if there was no improvement in solution quality after a number of generations. This heuristic added diversity to the search in order to prevent it from being trapped at the local optimum. The number of new individuals inserted into the population was limited to a quarter of the population. This limit ensured that the population maintained dominated by high quality genetic material.

The new chromosomes were added into the population after a number of generations no improvement to the phenotype had been recorded for. From a critical point of view, the concept of adding new individuals to the population in order to increase diversity is potentially beneficial as long as new individuals are introduced into the population only once diversity falls below an acceptable level. On a related subject, replacing up to half of the population with new individuals, would not amount to a significant effect, if the elite individuals remain in

the population to dominate the selection reproduction and reinsertion stages of the algorithm. Therefore, a better method might be to replace the entire population. Furthermore, the relaxed GRASP may not provide diverse enough individuals into the population. In addition, an alternative to the author's approach, the Relaxed GRASP may be based on population genotype (representation of a chromosome in the decision variable space) diversity instead of phenotype (representation of a chromosome in the objective space) diversity, with potentially promising results.

The sequential GRASP does not seem to guarantee that the size of the vehicle fleet will be minimum. The inherent advantage of this is that the algorithm would decide the appropriate number of vehicles. It may also be noteworthy that the author's parameter tuning approach may have benefited from employing an adaptive technique.

Erbao and Mingyong (2010) proposed a differential evolution algorithm (DE) to solve the VRPSDP with time windows. This approach was implemented because it demonstrated previous success in solving combinatorial optimisation problems. The algorithm combined arithmetic operators with those operators used to evolve the search. The DE in comparison to other EAs is known to be relatively greedy and less stochastic (Erbao and Mingyong, 2010). Therefore, a self adaptive crossover probability threshold was adopted to ensure sufficient diversity in the search. The probability threshold varied with the number of generations. The selection process guaranteed the survival of the fittest individuals, therefore ensured a level of accuracy in the search, as the offspring only replaced their parents in the population, if the fitness values were improved.

Analysing the contribution of the paper one might assume that the randomly generated initial population manages, in most likelihood, to cover the search space uniformly. In addition, the same can be said about a constant population size, which is used through the generations in order to ensure consistent computational resource consumption. Contrastingly, applying the crossover and mutation operator in the same generation has a destructive effect on the accuracy of the search and may lead to slow convergence. In addition, the crossover probability threshold was set to a low value in the early stages of the evolution to promote diversity in the search. At a later evolutionary stage, the probability threshold was increased to promote accuracy in the search. However, the approach used to set the threshold is not efficient because crossover should be extensively applied early on in the evolution, as the

initial population is randomly generated and sufficient diversity already exists. Once diversity begins to fall, the probability threshold should be lowered to allow for an increase in diversity to escape local optimum (Eiben and Smith, 2003). It may also be noteworthy that Erbao and Mingyong (2010) used relatively small size test problems to evaluate DE.

Tasan and Gen (2012) introduced a GA to solve the VRPSDP problem. Permutation encoding was used to represent a solution, which directly defined a set of routes with their sequence of service. This representation helped to construct a vehicle capacity feasible solution. The population was randomly generated in order to uniformly cover the search space, therefore increasing the probability of finding the most fruitful area for exploration. A penalty cost was added to the fitness value in proportion to the number of unfeasible routes encoded inside the chromosome. Therefore, the adoption of a penalty encouraged the search towards feasible areas of the search space. The GA parameters: population size and number of generations are determined using a pilot test problem. This reduced the bias involved in deterministic GA parameter setting.

The selection pressure favoured higher quality solutions, therefore, from a critical standpoint, the best known areas of the search are going to be explored extensively, resulting in the quick convergence to the local optimum. However, from an alternative viewpoint, the level of diversity in the search is likely to rapidly depreciate. Therefore, the application of a mutation operator with a rate of 0.03 might have a negligible effect on the search, in terms of escaping local optima. A possible justification for not applying a larger mutation rate is that it may decrease the search accuracy, as the crossover and mutation operators are applied in sequence.

Analysing the contribution of the paper one might assume that the method used to assign penalties to guide the search towards feasibility is logical, as each route is considered in the decision. However, the GA may be restricted from finding improved solutions, if the total distance travelled is made to be a redundant objective. From a different standpoint, a single dataset size of 34 customers is considered for GA parameters setting. However, this configuration might not have been the optimum for the other dataset sizes.

Tasan and Gen (2012) did not evaluate their GA using benchmarked test problems. It was noted that the largest evaluated instance contained 5 fewer nodes than the smallest 50 node

test problem provided by Salhi and Nagy (1999). On a related subject, the experimental results of Tasan and Gen (2012) reinforced the general view that GA is an efficient method for solving VRPSDP instances with a small number of nodes. Unfortunately, the approach does not explore larger size instances. It may also be noteworthy that no mention is made of the vehicle fleet size in the experimental results.

Wang and Chen (2012) employed a co-evolution GA to solve the VRPSDP with time windows. The co-evolution GA simultaneously evolved two populations: 1 and 2, where the role of each respectively was to encourage diversity and accuracy in the search. The fittest individuals in Population 1 were mated with individuals in Population 2 in order to help the search progress towards a new space populated by potentially fitter solutions. A co-evolution GA was used to overcome the shortcomings of the traditional GA, where the search either converged quickly to a poor solution or excessive computational resource was consumed in order to find an acceptable solution.

The authors used a heuristic technique called Random Seeds Cheapest Insertion Method (RSCIM) to derive an initial population for both populations, which consisted of high quality individuals and those of diverse structures. Wang and Chen (2012) implemented a GA to guide the initial population to an improved space. The RSCIM introduced diversity by randomly selecting seeds to initialise the construction of routes. The requests were selected to be inserted into a route in a random order. The insertion took place in the location where the maximum saving was achieved. The purpose of this technique was to reduce the time taken to reach a reasonable local optimum and to encourage the search towards the global optimum.

From a critical perspective, correctly tuned genetic operators can generate the required amounts of search diversity using a single population. Adopting a separate second population would increase computational cost and still not be as effective as evolving sub-populations, as the latter are allowed to communicate, thus increasing selection pressure and encouraging the production of fitter individuals. From a related standpoint, there is no real reason for selecting individuals with the greatest fitness from Population 1 for mating, as fitter individuals do not guarantee greater diversity in Population 2.

The introduction of the RSCIM heuristic outlines the importance of accuracy and diversity in the initial population in the context of an effective solution space search. However, it may be

argued that diversity should be the only consideration when constructing the initial population, in order to cover the search space uniformly. From a different standpoint, RSCIM heuristic must build an initial population with optimal solution properties in order to reach the global optimum, otherwise, the search will converge at the local optimum. Furthermore, it may be argued that the time taken to construct the initial population using RSCIM heuristic may be better spent in the evolutionary process.

From a critical point of view, the application of an accuracy inducing crossover operator followed by the eleven mutation inducing operators inside Population 2 is likely to diminish the accuracy generated. Furthermore, the application of eleven mutation operators is highly probable to cancel one another's effects. In addition, in the presence of eleven diversity promoting operators, the question arises whether Population 1 adds any benefit.

Wang and Chen (2012) evaluated their algorithm with respect to the vehicle fleet size and total travel distance results. However, it is noteworthy to mention that the fitness function only considered the latter, therefore selection does not imply both objectives, which may hinder the performance of the evolution. Equally important, the efficiency of the co-evolution GA remains unverified because the authors have provided limited information on how the basic GA is configured.

Zhang et al. (2012) proposed an evolutionary algorithm called scatter search (SS) because it is claimed to be efficient in exploring a wider search space. The solution space is explored by evolving a set of elite and diverse solutions using unifying principles, while making minimum use of randomisation. The network was modelled using stochastic travel times to better reflect real life transportation problems. Zhang et al. (2012) modified Clarke and Wright (1964) saving function to better construct an initial solution. Their approach influenced the insertion decision with respect to the vehicle load utilisation levels. The approach encouraged the insertion of net delivery demand requests earlier in the route and net pickup demand requests towards the end of the route. The initial solution generated by the modified saving procedure was altered using a neighbourhood operator to generate a set of solutions to form the initial population. The purpose of this approach was to have a diverse set of solutions, with a similar structure to the initial solution.

Generally, the modelling of stochastic travel times narrowed the gap between theoretical study and commercial relevance. This is an important contribution because all previous VRPSDP research work only considered deterministic travel times. Although, modelling their approach with deterministic travel times might have allowed Zhang et al. (2012) to perform a computational result comparison with other approaches. Moreover, the modified saving function is a greedy approach, which may restrict the construction of an improved solution. The procedure does encourage the construction of capacity feasible solutions making the approach a valuable contribution to the literature.

From a critical point of view, the initial population is unlikely to comprise of a diverse set of solutions because the individuals were variations of a single solution generated using the modified saving procedure. Therefore, it is improbable that the initial population will cover the search space uniformly. In addition, the SS algorithm randomly paired solutions for mating from a set of solutions that were either of high quality with respect to the objectives or of a diverse structure. On the contrary, a random pairing of solutions is unlikely to guide the search towards a zone characterised by greater accuracy or diversity.

With respect to the GA adopted for comparison purposes it may be concluded that the application of crossover followed potentially by mutation, may have affected the accuracy increasing effects of the former. Although, the probability that the mutation operator was applied was 0.1. The level of diversity introduced by the operator when applied was randomly determined. In addition, a limit was placed on the number of GA generations, potentially causing the evolution to converge prematurely.

The SS algorithm parameters were set based on the computational performance using a particular test problem type. Analysing the contribution of the paper one might assume that the concept of adaptive parameter setting is logical. In addition, the GA parameter were deterministically set, which puts the GA at a natural disadvantage against the SS algorithm during the comparison stage. It is also noteworthy that the computational time is related to the running time of the algorithm and not to the time taken to find the best solution, therefore, it is not conclusive that the SS algorithm is more scalable than GA. Another reason to support this is that the evaluated test problem sizes are limited in their range. On a different issue of importance, the authors simulated the same test problem 10 times in order

to measure the performance of the SS algorithm. This is important in order to determine the computational efficiency of the algorithm.

### Hybrid

Crispim and Brandao (2005) proposed a hybrid algorithm comprising of two metaheuristics: tabu search (TS) and variable neighbourhood descent (VND) to solve the VRPSDP. The hybrid was suggested to offer greater levels of diversity than either of the standalone metaheuristics because the TS prohibited recently applied local search moves through the use of a tabu list and the VND changed the neighbourhood structure.

A modified VND was used to explore a new neighbourhood structure after a certain number of iterations without a solution improvement. This was in contrast to the original VND introduced by Hansen and Mladenovic (1997), which changed the neighbourhood structure once no further improvement was possible. The modified VND was adopted in order to reduce computational resource consumption. On a different issue of importance, a single tabu list is used for all neighbourhoods to avoid additional computational resource expense for updating multiple tabu lists and to prevent recent search moves from recurring.

Analysing the contribution of the paper, the increase in computational efficiency provided by the modified VND is an important contribution, even at the expense of some loss of accuracy in the search. The issue is to maintain a reasonable level of neighbourhood exploration, therefore determining the right number of iterations before changing the neighbourhood structure is important. This increases the complexity of the method, which is further extended when an adaptive iteration number is sought.

The hybrid algorithm is applied to an initial solution constructed using the principles of the sweep algorithm. However, if the number of routes exceeds the minimum number of vehicles required to service the region based on capacity requirements, a bin-packing problem is solved with a tabu search algorithm with the aim of finding a set of minimum routes. Therefore, the application of the sweep heuristic does not guarantee an initial solution with the minimum vehicle fleet size. The Crispim and Brandao (2005) requirement, that the initial solution operates the smallest vehicle fleet size, is important because computational resources may be wasted in optimising a poor initial solution with many more vehicles than actually required.

Crispim and Brandao (2005) did not compare their computational results with Dethloff (2001). However, the hybrid algorithm managed to generate improved solutions for Salhi and Nagy (1999) test problems, except for CMT12Y. From a different stand point, Crispim and Brandao (2005) calculate the minimum vehicle fleet size based on capacity requirements:

$$\max \left\{ \frac{\text{Total demand of delivery customers}}{\text{Capacity of vehicle}}, \frac{\text{Total demand of pickup customers}}{\text{Capacity of vehicle}} \right\} \quad (3.1).$$

Yet, the authors have operated one more vehicle than is capacity feasibly required for test problems CMT3Y, CMT4Y and CMT5Y.

Zachariadis et al. (2009) combined two well-known metaheuristics: Tabu search and Guided location search (GLS) to solve the VRPSDP. The latter was used to guide Tabu search to a more diversified search, therefore preventing the search from being trapped in a local optima. The key to GLS success related to its ability in identifying features of a low quality solution causing local optimality and then quantifying appropriate penalties to prevent this from occurring. The hybrid was applied to an initial solution constructed using the heuristic by Paessens (1988). The approach built routes in a sequential manner based on cost savings. The hybrid was compared with several benchmark instances reported in the literature. It was proved that the approach was capable of generating high quality solutions and in some cases, yielded improved results than previously reported, with respect to the vehicle fleet size and total routing distance objectives.

Zachariadis et al. (2009) introduced a hybrid approach in order to combine the merits of two metaheuristics so as to efficiently search the solution space where a single metaheuristic is deemed to be restrictive. However, this belief is not strongly reflected in the body of literature, as the majority of research work is based on a single solution method. On a different issue of importance, the hybrid was applied to an initial solution that had been constructed using a heuristic technique. This solution will have a bearing on the ability of the hybrid approach to converge towards the optimal solution. It is unknown whether the initial solution will share traits with the optimal solution, therefore its relevance to the proposed approach is not quantifiable.

Subramanian et al. (2010a) proposed a parallel heuristic operating over multiple processors to solve the VRPSDP. The parallel nature enabled large size instances to be solved. The approach consists of an iterative local search (ILS) framework with an integrated variable neighbourhood descent procedure that used a random neighbourhood ordering (RVND). The accuracy in the search increased using this greedy approach, which at every stage attempted to improve the solution, using inter and intra route improvement techniques. A diversity mechanism was employed in order to circumvent against being trapped at the local optimum, which allowed a set of permutations without any solution improvement.

Diversity is introduced into the search by randomly selecting one of three available mechanisms. From a critical point of view, the selection is not explicitly related to the current state of the search, in terms that, should diversity be insufficient at some point, an appropriate diversity mechanism is not guaranteed to be selected.

The authors have applied 6 inter-route operators to generate a newly improved solution, followed by 4 intra-route operators to improve recently modified routes. However, the benefit of the combination of operators has not been demonstrated. Therefore, it is possible that similar outcomes may have been achieved by using fewer operators. Furthermore, these operators are exhaustively applied therefore are computationally resource expensive.

The computational results illustrate that the solution approach is very competitive compared to other previously published methods in terms of the test cases evaluated. A contributing factor may be that the main parameters were experimentally determined. However, with respect to Salhi and Nagy (1999) test problems: CMT3Y, CMT4Y and CMT5Y, one more vehicle is operated than what is physically required.

#### Further Heuristic Algorithms

Nagy (1996) proposed an integrated heuristic (IH) and three modified versions to solve the single and multi depot VRPSDP problem. The IH applied inter and intra route operators to the initial solution in different combinations in order to produce an optimised solution. The first IH variant, PEN, constructed the initial solution using penalties to encourage the development of a strong capacity feasible solution. Therefore, a greater amount of computational resources may be used for optimisation. The second IH variant, ALT, allows for a certain level of solution infeasibility into the search in order to explore new areas of the search space. The level of

infeasibility is set by a parameter. This method introduced a certain level of diversity to the search in order to better explore the search space. The third IH variant, SO, is similar to ALT, with the exception that the level of infeasibility is allowed to oscillates according to a schedule. This ensured that after a number of accuracy inducing steps, a number of diversity inducing steps are applied, therefore the search space can be thoroughly explored.

It is noteworthy that ALT and SO heuristics have attempted to introduce diversity into the search in order to explore the search space. However, the issue with the ALT heuristic is that the parameter is not guaranteed to provide the required levels of diversity when required. The SO heuristic does not relate diversity to the need of the search, instead diversity is automatically applied after a number of generations, which may affect search accuracy and prevent a good solution from being located.

The initial solution was required to be weak capacity feasibility. This did not have to be the case as inter route operators could have been applied at a later stage to gain weak capacity feasibility. However, this approach is deemed sensible, as it ensures that a strong capacity feasible solution can be determined using intra route operators, except in circumstances where route length restriction are present. This approach is likely to reduce the computational time required to find an appropriate solution. In contrast, the initial solution was not required to be strong capacity feasible because it was thought this may lead to poor solution quality, which may not be improvable by inter and intra route operators. However, this should not be the case as the same operators are applied to strong capacity feasible solution at a later stage for solution improvement.

Nagy (1996) introduced two local search techniques for the VRPSDP called NECK and UNNECK. The former routine aimed to reduce the net vehicle load throughout the route in order to service customer close to the depot with either large delivery or pickup demands. NECK separated the service of the delivery and pickup demand into two entities, which were serviced independently. Later stages of the search UNNECK were used to recombine the two separated entities. However, this is not guaranteed to occur, therefore these routines should not be applied to VRPSDP, as the simultaneous service constraint may not be satisfied. Another routine of interest is the 2-opt operator, this was iteratively applied to the route segment, which minimised the vehicle net load to the lowest level. However, this greedy approach does not guarantee strong capacity feasibility.

Chen and Wu (2006) introduced a two phase method to solve the VRPSDP. The first phase constructs routes in a sequential manner using an insertion-based algorithm called PROC\_INS and the latter phase improves the initial solution using a hybrid heuristic titled HeuSDP. The PROC\_INS insertion criterion managed the vehicle capacities at the nodes, therefore a greater number of requests could be inserted into the route. This was possible because the route position of a request with a large net delivery demand will be serviced at the start of the route and the request with a large net pickup demand will be serviced towards the end of the route. The HeuSDP consists of local search techniques; tabu list and record-to-record travel (RRT). Firstly, inter and intra local search techniques were used to search for an improved solution. A tabu list was employed to prevent the local search techniques from recycling visited paths unless such moves results in a better neighbourhood solution. Finally, the Record-to-record travel (RRT) was a scheme similar to Simulated Annealing (SA) where a neighbouring solution was accepted if it is not worse than the best-known solution plus a gradual lowered deviation.

The PROC\_INS insertion criterion represents an important contribution in terms of effectively managing the vehicle capacity along the cycle. However, the heuristic sequentially builds routes by maximising their vehicle utilisation, which may had resulted in the final route having less workload than the others, therefore resulting in an inequitable workload. In addition, the PROC\_INS inserts requests in a greedy manner, therefore this may not be the best sequence of insertion. Another aspect of importance relates to the results based comparisons. The authors found very improved results against Salhi and Nagy (1999), therefore illustrating the effectiveness of their approach. However, the authors did not compare their results against Dethloff (2001) and if they had their improvements in solution quality would have been less significant.

Bianchessi and Righini (2007) presented and compared the computational output of construction, local search and tabu search based algorithms for the VRPSDP. The proposed construction method was based on a modified version of the tour partitioning algorithm, which ensured capacity feasibility of the sub-routes. This algorithm was compared to a tour partitioning method without the proposed amendment. The comparison indicated that the modified tour partitioning algorithm exploited the capacity of the vehicles more efficiently than the method that did not. Bianchessi and Righini (2007) compared several neighbourhood structures for exploration by local search algorithms. The variable neighbourhood structure

was found to be the most efficient because results similar to the best obtained were found, with significantly less computational resource consumption. This observation was reinforced by the results obtained by exploring these several neighbourhoods with the tabu search algorithm. In summary, the local search algorithm with a variable neighbourhood was arguably substantially more computationally efficient compared with the tabu search algorithm, but with slightly poorer solution qualities.

The modified tour partitioning algorithm removes and then reinserts all capacity unfeasible requests. However from a critical perspective, the sequence of reinsertion may not be optimal, which in turn may lead to a higher solution cost. Another aspect of importance relates to the pitfall of using a variable neighbourhood structure because the first best neighbourhood is selected at each local search step to replace the incumbent, which may result in a sub-optimal selection. However, this approach does effectively balance the needs of exploitation and computational time. From a different standpoint, Bianchessi and Righini (2007) implement a dynamical tabu list size in order to guide the search between accuracy and diversity focused search. As opposed to having a deterministic list size, this concept does not reflect the state of the search.

Jun and Kim (2012) proposed a three stage heuristic approach to solve VRPSDP, which consisted of a: route construction procedure, route improvement procedure and diversity inducing procedure. The latter two were iteratively applied in order to guide the search to an improved space. The initial routes were constructed using a modified version of Gillett and Miller (1974) classical sweep algorithm, which initialised a cluster using polar angles and the subsequent requests were inserted relative to their distance. This modification overcame the issue with the incumbent method, which resulted in elongated cone shaped routes, which were undesirable from a practical viewpoint. The initial solution was then modified using inter and intra local search mechanisms, in the search for an improved solution, until a stopping criterion was met. Thereafter, a diversity inducing mechanism was introduced to prevent the search from becoming stuck in the vicinity of the local optimum. This was achieved by ejecting requests and then reinserting them into existing or new routes.

The modification to the sweep algorithm was an important contribution by the authors, which built more practical solutions because clusters could be built above one another. This is in contrast to other research works: Kanthavel et al. (2012), Wassan et al. (2008) and Montane

and Galvao (2002) that has not considered this practical importance. Despite the improvement to the sweep algorithm, a fundamental flaw still persists with the method, which relates to the equitable vehicle utilisation between routes, since, the last route to be constructed was likely to be utilised less compared with the previously clustered vehicles. On a different issue of importance, the level of diversity introduced by the method is randomly determined and has no bearing to the state of the evolution. A consequence of the approach is that sufficient levels of diversity required to progress the search may not be induced during the appropriate times during the search.

The main contributions from each research work is summarised below in Table 3.1. It is evident that the literature has predominately focused on approximation methods to solve the VRPSDP. To date, only a limited number of papers are known to have used exact methods to solve this problem, as they are computational resource expensive. Over the past decade, a greater proportion of the research works have focused on metaheuristics, which evolutionary algorithms are the biggest contributors. It is noteworthy that a prominent solution method design exists in the literature, where an initial complete solution is constructed and then later optimised. A consistent shortcoming of this approach is that high quality initial solution traits are not protected from dismantlement in the optimisation phase. Goldberg (1989) introduced the building block (BB) hypothesis, which suggests that the recombination of high quality solution traits will effectively guide the search towards the optimal solution. The failure of the current methods to protect certain traits may decelerate the progress of the search towards the higher quality areas, therefore the methods in the domain are likely to be computationally expensive. Moreover, an often considered objective for the vehicle routing problem is the workload variation objective Toth and Vigo (2002), which aims to minimise the maximum variation between route distances. However, no research work has addressed this important objective in the VRPSDP domain. The shortcomings provided here in relation to the studied problem are addressed by a new three phase approach, SDPmethod, which is validated in Chapter 6 by comparing its results against the best known solutions in the domain.

Table 3.1 main contributions from the VRPSDP domain

<b>Method Type</b>	<b>Method</b>	<b>Contributions</b>	<b>Publication</b>
Exact	Branch and Price	<ul style="list-style-type: none"> <li>• 1st exact method for VRPSDP/TW</li> <li>• Set covering formulation</li> </ul>	Angelelli and Mansini (2002)
		<ul style="list-style-type: none"> <li>• Exact dynamic programming</li> <li>• State space relaxation</li> </ul>	Dell'Amico et al. (2006)
	Branch and Cut	<ul style="list-style-type: none"> <li>• Undirected &amp; directed two commodity flow formulation</li> </ul>	Subramanian et al. (2010b)
		<ul style="list-style-type: none"> <li>• Relaxed vehicle capacity constraints</li> </ul>	Subramanian et al (2011)
Heuristic	Cluster-first and route-second	<ul style="list-style-type: none"> <li>• Introduced VRPSDP concept</li> </ul>	Min (1989)
		<ul style="list-style-type: none"> <li>• Reverse logistics</li> <li>• Residual capacities &amp; Residual surcharges</li> </ul>	Dethloff (2001)
		<ul style="list-style-type: none"> <li>• 8 heuristic procedures</li> </ul>	Montane and Galvao (2002)
	Tabu Search	<ul style="list-style-type: none"> <li>• 8 heuristics procedures for the initial solution</li> </ul>	Montane and Galvao (2006)
		<ul style="list-style-type: none"> <li>• Dynamic tabu list size</li> </ul>	Wassan et al. (2008)
	Ant Colony Optimisation	<ul style="list-style-type: none"> <li>• Savings based visibility function</li> <li>• Rank-based pheromone update procedure</li> </ul>	Catay (2010)
	Ant Colony System	<ul style="list-style-type: none"> <li>• Candidate list</li> <li>• Visibility function</li> </ul>	Gokce (2004)
		<ul style="list-style-type: none"> <li>• Inter &amp; intra route operators applied to ant solutions</li> </ul>	Gajpal and Abad (2009)
	Particle Swarm Optimisation	<ul style="list-style-type: none"> <li>• Cluster-first &amp; route-second</li> </ul>	Kanthavel et al. (2012)
		<ul style="list-style-type: none"> <li>• Spatial proximity from seeds</li> </ul>	Ai and Kachitvichyanukul (2009)
	Simulated Annealing	<ul style="list-style-type: none"> <li>• Unique cooling schedule (T)</li> </ul>	Ganesh and Narendran (2008)
	Differential Evolution Algorithm	<ul style="list-style-type: none"> <li>• Arithmetic operators</li> <li>• Self adaptive crossover probability threshold</li> </ul>	Erbao and Mingyong (2010)

<u>Method Type</u>	<u>Method</u>	<u>Contributions</u>	<u>Publication</u>
	Genetic Algorithm	<ul style="list-style-type: none"> <li>• Random keys method</li> <li>• Adaptive mutation probability threshold</li> </ul>	Vural (2003)
		<ul style="list-style-type: none"> <li>• Integer encoding</li> <li>• GRASP heuristic</li> </ul>	Vural (2007)
		<ul style="list-style-type: none"> <li>• Permutation encoding</li> </ul>	Tasan and Gen (2012)
	Co-evolution Genetic Algorithm	<ul style="list-style-type: none"> <li>• Evolves two populations</li> <li>• Random Seeds Cheapest Insertion Method</li> </ul>	Wang and Chen (2012a)
	Scatter Search	<ul style="list-style-type: none"> <li>• Stochastic travel times</li> </ul>	Zhang et al. (2012)
	Hybrid	<ul style="list-style-type: none"> <li>• Tabu Search &amp; Variable Neighbourhood Descent</li> <li>• Greater diversity</li> </ul>	Crispim and Brandao (2005)
		<ul style="list-style-type: none"> <li>• Tabu Search &amp; Guided Location Search</li> <li>• Greater diversity</li> </ul>	Zachariadis et al. (2009)
		<ul style="list-style-type: none"> <li>• Iterative Local Search &amp; Variable Neighbourhood Descent</li> <li>• Parallel heuristic</li> </ul>	Subramanian et al. (2010a)
	<u>Further Heuristic Algorithms</u>	<ul style="list-style-type: none"> <li>• Methods for the single &amp; multi depot VRPSDP</li> </ul>	Nagy (1996)
		<ul style="list-style-type: none"> <li>• Neighbourhood selection based on a deviation level</li> </ul>	Chen and Wu (2006)
		<ul style="list-style-type: none"> <li>• Construction, local search &amp; tabu search methods</li> </ul>	Bianchessi and Righini (2007)
		<ul style="list-style-type: none"> <li>• Modified sweep algorithm</li> </ul>	Jun and Kim (2012)

Further Heuristic Algorithms - an approach that cannot be classified into a single type.

Table 3.2 summarizes the test problem types used in the aforementioned twenty-eight VRPSDP research works. Overall, Salhi and Nagy (1999) test problems were the most often implemented in the domain, as 54% (15) research works choose to adopt them. A contributing factor to their popularity may be derived from them being the earliest significant size test problems in the domain, mounting to 200 nodes. The next most commonly applied problems were by Dethloff (2001), as 46% (13) of papers considered their datasets containing up to 50 customers. One of the reasons why Salhi and Nagy (1999) test problems were more frequently applied than Dethloff (2001) was that their smallest test problem was the same size of the largest in Dethloff (2001). It is also noteworthy to mention that a substantial proportion of research papers designed their own test problems, 39% (11), which has caused fragmentation in the field with respect to evaluating different solution approaches. Also, Montane and Galvao (2006) introduced the largest size test problems with up to 400 customers. However, the adoption of these test problems has been limited to 14% (4) of the research works reviewed here. The low adoption rate can be contributed to the simplified nature of the routing problem, caused by the limited vehicle capacity availability in relation to the demand associated with each customer. In addition, Min (1989) was the only research work to use a real world dataset. Moreover, Ai and Kachitvichyanukul (2009) was the only research work that used Dell'Amico et al. (2006) test problems. In this particular instance Ai and Kachitvichyanukul (2009) wanted to compare the performance of their approach against an exact method introduced by Dell'Amico et al. (2006). After reviewing the test problems types in the literature, the SDPmethod will be applied to Salhi and Nagy (1999) benchmark instances, as they have been widely studied in the domain. This will improve the rigor of the experimental evaluation in chapter 6 because a greater number of experiential results are available for comparison.

Table 3.2 summarises the test problem types adopted in the literature

Paper	Test problem						
	Actual	Create	Min (1989)	Salhi and Nagy (1999)	Dethloff (2001)	Dell'Amico et al. (2006)	Montane and Galvao (2006)
Kanthavel et al. (2012)				X			
Jun and Kim (2012)				X			
Tasan and Gen (2012)		X					
Wang and Chen (2012)		X					
Zhang et al. (2012)					X		
Subramanian et al. (2011)				X	X		X
Catay (2010)			X	X	X		
Erbao and Mingyong (2010)		X					
Subramanian et al. (2010a)				X	X		X
Subramanian et al. (2010b)				X	X		X
Ai and Kachitvichyanukul (2009)				X	X	X	
Gajpal and Abad (2009)			X	X	X		
Zachariadis et al. (2009)				X	X		X
Ganesh and Narendran (2008)		X					
Wassan et al. (2008)				X			
Bianchessi and Righini (2007)					X		
Vural (2007)				X	X		
Chen and Wu (2006)		X		X			
Dell'Amico et al. (2006)		X					
Montane and Galvao (2006)		X		X	X		
Crispim and Brandao (2005)				X			
Gokce (2004)			X		X		
Vural (2003)			X		X		
Angelelli and Mansini (2002)		X					
Montane and Galvao (2002)		X					
Dethloff (2001)		X	X	X			
Nagy (1996)		X					
Min (1989)	X						
Total	1	11	5	15	13	1	4

Actual - used a real data set; Create - converted their own VRPSDP dataset, either by randomly generating their own or converting non VRPSDP datasets; remaining columns - used data sets proposed in specified papers. Self referencing is not included.

#### 4. GENETIC ALGORITHM

This chapter provides a detailed description of genetic algorithm design because the method strongly contributes to the proposed methodology. The most prominent multi objective evolutionary algorithms research works are also reviewed, in order to establish the advancements that have been made in the field because a multi objective optimisation problem is considered here.

The term Genetic Algorithm (GA) was first introduced by Holland (1975) and refers to a global search heuristic used to solve hard combinatorial optimisation problems. The approach is derived from biology analogy and it relates to natural genetics: the purpose of selective breeding of plants and animals is to produce offspring with desirable characteristics (phenotype), which are defined at a genetic level (genotype), (Golden et al., 2008). Similarly, a genetic algorithm encodes the genotype of each parent inside a chromosome. The chromosomes with a high probability of generating offspring with an improved phenotype are selected as parents to mate. The recombination process uses genetic operators to generate offspring. The purpose of these operators is to encourage search accuracy and diversity, which facilitates the process of finding the optimal solution. The following outlines the basic steps of a Genetic Algorithm, which are described in greater detail later on.

GA Steps:

1. **Initialisation** consists in constructing a population of potential solutions (chromosomes), which map potential solutions for a particular problem.
2. **Evaluation** determines the quality of an individual based on their objective values. Some approaches Tansen and Gen (2012) and Vural (2003) consider the objective values met by an individual to be the fitness of that certain individual. Others Deb (2009), Coello and Coello et al. (2007) apply additional processing (scaling, normalisation) to objective values in order to obtain fitness values. This research employs the second variant.
3. **Selection for Reproduction** selects chromosomes as parents for mating.
4. **Genetic Operators** generate offspring by modifying the genetic makeup of their parents.
5. **Selection for Reinsertion** individuals are selected for the population in the next generation.

6. **Termination Criterion** is a stopping condition and steps 2-5 are repeated until this criterion is met.

#### 4.1 Encoding

A chromosome stores decision variables, which encode a potential solution. The encryption of a chromosome is problem specific and a standardised approach does not exist. Furthermore, a particular problem may allow for multiple encryptions and the type chosen will influence the choice of genetic operators (Patelli, 2011), (Deb, 2009). Therefore, the most appropriate encryption type should be selected, so compatible genetic operators can be defined to explore the search space efficiently (Patelli, 2011). The following are the main encoding methods available: Binary string, Real number string and Permutation string.

##### 4.1.1 Binary Encoding

The chromosome consists of a set of so-called genes represented in binary form [ 0 1 ]. Binary encoding can be used in all problems, but it is not very effective all the time (De Jong, 2006). The 0-1 Knapsack problem is an example where binary encoding is effective. The 0-1 Knapsack problem involves the optimisation of a function subject to a constraint and is formally defined as,

$$\begin{aligned}
 & \text{maximise } \sum_{j=1}^n p_j x_j \\
 & \text{subject to } \sum_{j=1}^n w_j x_j \leq c \\
 & x_j \in \{0, 1\}, \quad j = 1, \dots, n.
 \end{aligned} \tag{4.1}$$

An example of a 0-1 Knapsack problem is where a retailer needs to maximise the value  $p_j$  of items  $x_j$  inside a customer's basket, whilst abiding to the weight limit  $c$ . Figure 4.1 illustrates the binary encoding of a 0-1 Knapsack problem, where a binary code represents whether an item is in the basket and a chromosome encodes a solution. In the assumed encoding the value of 1 defines the item is in the basket, whereas the value of 0 refers to the opposite case.

Item	Value (£)	Weight (kg)
1	3.5	3
2	6.6	3
3	8.2	6

Basket weight constraint = 10kg

	Item 1	Item 2	Item 3	
Chromosome	0	1	1	Basket Value = £14.8 Weight = 9kg

Figure 4.1 Binary encoding

#### 4.1.2 Real Number Encoding

Instead of binary values, the chromosome consists of real number gene values, an encoding example is shown in Figure 4.2. It is widely accepted that real number encoding is a better approach than binary encoding for function optimisation problems because the topological configuration of the genotype and phenotype space are equal for real number encoding, therefore it is easier to utilise previously proven genetic operators for the search (Gen and Cheng, 2000).

	Gene 1	Gene 2	Gene 3	Gene 4
Chromosome	0.1356	0.4341	0.1608	0.5667

Figure 4.2 Real number encoding

#### 4.1.3 Permutation Encoding

The chromosome is made up of integers, which encode a sequence for an ordering problem like the Travelling Salesman Problem (TSP). This type of encoding places a restriction on the genetic operators to ensure a set of gene values are always present inside the chromosome. Figure 4.3 illustrates the permutation encoding for a TSP, as each gene value symbolises a city to be visited and the chromosome corresponds to a Hamilton cycle without the source/end node.

	Gene 1	Gene 2	Gene 3	Gene 4
Chromosome	1	7	3	5

Figure 4.3 Permutation encoding

A GA population consists of a set of chromosomes that populate the search space under investigation. The population size is an important consideration because a trade-off exists between the degree of parallel search required and the amount of computation run time available. The former depends on the size of the search space. The search space dimensions increase with the number of objectives in the Multiple Objective Optimisation (MOO) problem, which the population size should reflect by being larger in order to encourage the generation of high quality solutions (De Jong, 2006). However, a larger population requires greater computational effort at the expense of more run time, which may not be available for real time problems. In addition, in the earlier stages of the evolution more computational effort is wasted on improving chromosomes with poor phenotypes. Therefore, from a computational efficiency standpoint it is advisable to minimise the population size. However, from a biological perspective, a small population with insufficient genetic diversity is unlikely to produce quality offspring as the individuals therein present the risk of sharing the same genetic defect (Ashlock, 2006). Therefore, in the context of GA, a small population is likely to be trapped in a given area of the search space (e.g. a local optimum).

There are two approaches that can be used to generate the initial population: random and solution based. The former approach generates the initial population randomly based on a probability distribution. The uniform distribution of chromosomes throughout the search space, as shown in Figure 4.4 is required when the genotype encoding the optimum phenotype is unknown. The random generation of individuals is more likely to provide a uniform coverage of the search space, if the considered population size is sufficiently large. Furthermore, the individuals of the randomly generated population may not be feasible, therefore obtaining feasibility may consume extensive computational resources. Alternatively, the latter approach constructs an initial population of feasible solutions in the vicinity of a known solution of acceptable quality with a heuristic technique. This may enable GA to find an improved solution in shorter computational time. However, the heuristic may not be capable of generating a feasible population with diverse genotypes, in which case GA may prematurely converge. Furthermore, the extra computation required to generate a feasible population could instead be used to exploit and explore the random population, which is constructed in negligible time.

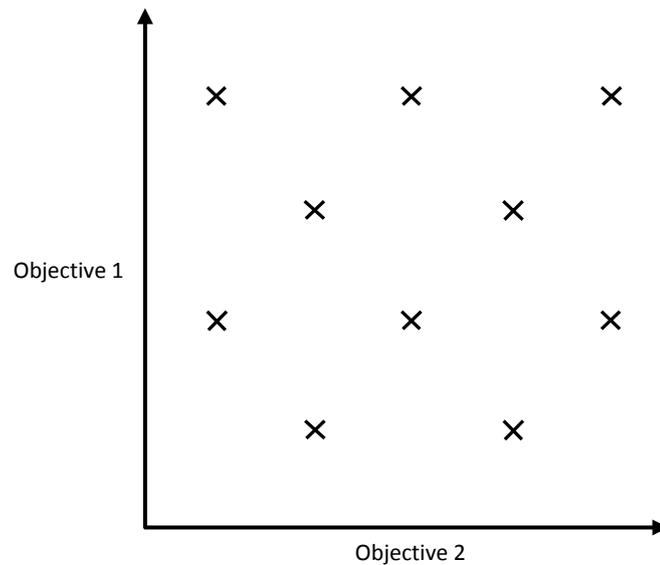


Figure 4.4 Uniform distribution of chromosomes in a two-dimensional search space

#### 4.2 Linear Evaluation

The reproductive potential assessment of individuals inside a population is performed in relation to their fitness. Hartl and Clark (1989) as mentioned in Fogel (2006) provide a biological notion of fitness, as the probability that an individual will survive and reproduce in a specific environment. In the context of evolutionary algorithms employing roulette selection techniques, the fitness values are regarded as selection probabilities that sum up to one (Patelli, 2011). These values are utilised when selecting individuals for reproduction and reinsertion. To encourage continuous improvement through the evolutionary process, the fitness value relates to the objective value(s). The fitness function determines how well an individual satisfies the objective(s) and assigns a value to it (Coello Coello et al., 2007). Therefore, this performance information can be used to effectively explore the current search space (Koza, 1998). Where a minimisation problem is considered the fitness value is an inverse of the objective value(s). An individual with a higher fitness than its counterparts will have a greater probability of selection for reproduction or reinsertion compared to the others (Coello Coello et al., 2007). Hence, fitness of an individual is not measured in direct isolation, but in relation to the entire population.

In a Single Objective Optimisation (SOO) problem, the fitness computation stage is straight forward as only one objective value is involved. One of the approaches to solving Multi Objective Optimisation (MOO) problem is to compute a single fitness value, which illustrates all objectives. In mathematics the problem is known as transitioning from a multi dimension

(hyper) space to a single dimension space. One approach is to convert the MOO problem into a SOO problem via aggregation. However, this requires information on the weightings of the objectives, which may not be known. If the weightings are known the following well-known techniques can compute the fitness values: Raw Fitness Assignment, Adjusted Fitness, Proportional Fitness and Ranking Fitness (Patelli, 2011).

As classified in Patelli (2011), in the absence of weighting information for MOO problems fitness values can be determined using either Pareto or non-Pareto approaches. The former approaches compare individuals in terms of dominance. Dominance is used to determine fitness values for individuals. An individual is considered non-dominated compared with another, if at least one objective value is better and none are worse (Deb, 2009). In Figure 4.5a, individual A dominates B, therefore is fitter because all objective values are lower (a minimisation problem involving two objectives is assumed). At the same time, Figure 4.5b illustrates non-dominance relationship, as neither individual A nor B dominates the other because neither is better than the other over both objectives.

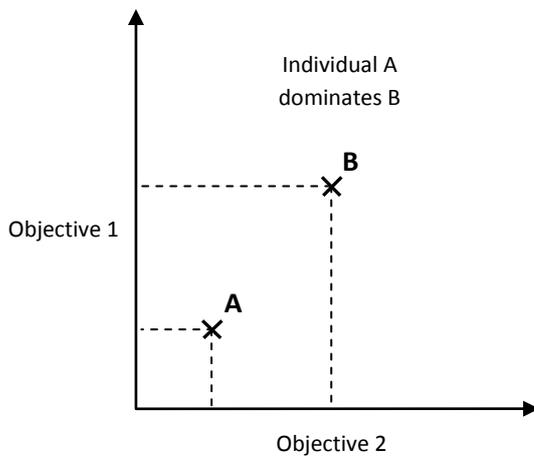


Figure 4.5a Dominance relationship

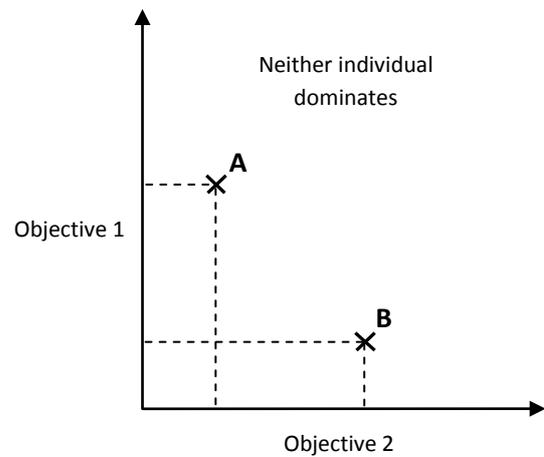


Figure 4.5b Non-dominance relationship

#### 4.2.1 Pareto Approaches

The concept of dominance is used in most MOO problems methods, (Deb, 2009) because it provides intuitive numerical support for fitness computation. A MOO problem with conflicting objectives has a set of solutions with differing objective values that can be considered optimal, (Deb, 2009). The non-dominated solutions in the entire search space are referred to as Global Pareto Optimal/Pareto Optimal (Deb, 2009), (Goldberg, 1989), see Figure 4.6 which illustrates the Pareto optimal front as '1<sup>st</sup> PF'. Deb (2009) defined the MOO problem as having two aims:

to find a set of solutions closest to the Pareto optimal front and to find as many diverse solutions as possible within that particular set in terms of the considered objective values. A Pareto front is a curve that graphically best fits the Pareto set.

When computing fitness values for a population of solutions, it is useful to rank individuals in terms of dominance. Pareto methods can be used to divide individuals in several sets, which are referred to as n-order Pareto fronts, where n is an integer  $n \in [1, n]$ . The level of individual fitness on a given Pareto front decreases with the order because their objective values are further away from the optimal ones. The individual(s) inside a Pareto set do not dominate one another, therefore are in a non-dominance relationship. However, the individuals dominate the ones in higher order Pareto sets. Therefore, apart from the 1<sup>st</sup>-order Pareto set, all other sets are dominated by at least one individual in a lower order set. Figure 4.6 illustrates 3 Pareto sets with their corresponding Pareto fronts for a minimisation problem.

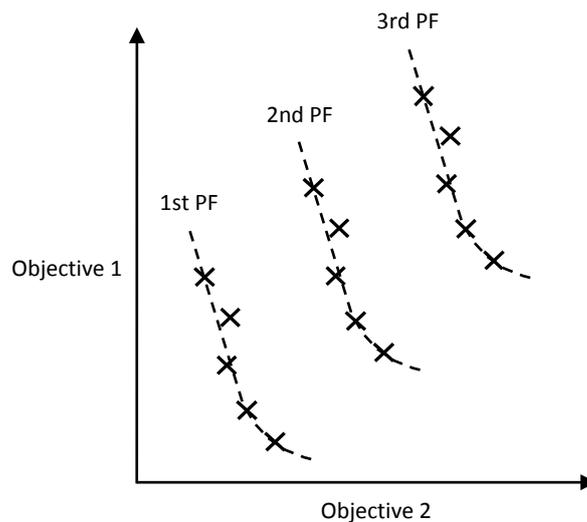


Figure 4.6 Pareto fronts

Deb (2009) described and investigated the average complexity of methods to find a non-dominated set. The Continuously Update procedure is faster than the Naive and Slow procedure because it has halved the number of computations. Naive and Slow procedure compares the dominance for every individual against the population. The Continuously Update procedure is similar to the aforementioned method, but instead only compares the individuals in the population with non-dominated individuals.

Once the Pareto sets have been determined the following published methods can compute individual fitness values in relation to their order.

As stated in De Jong (2006), Goldberg (1989) introduced a dominance ranking approach that ordered a population of individuals based on the number of individuals they each dominate. However, a shortcoming of the approach is the fitness value of the same individual may be highly different in the next generation, as fitness is determined in relation to other individuals.

Deb (2001) introduced a fitness sharing formula to assign fitness values to individuals on nth order Pareto fronts (4.2). The formula ensures fitness values reflect the relationship between different ordered Pareto fronts and prohibits an individual in the nth order Pareto set from having a higher fitness than any individual in the nth-1 order. In addition, the formula encourages the search towards weakly explored areas by assigning higher fitness values to isolated individuals on the fronts compared to those in a density populated areas. The diversity promoting aspect of the formula is based on Goldberg's (1989) fitness sharing concept, which is a widely used approach to increase diversity in MOO (Coello Coello et al., 2007). The sharing concept supports the construction of a distributed spread of nondominated individuals by encouraging the selection of solutions located in less crowded regions of the search space. This is achieved through penalising individuals situated in densely populated neighbourhoods by sharing their fitness values between neighbours. In contrast, isolated individuals retain their fitness values thereby their selection is encouraged and convergence to a single point in the search space is prevented.

$$Fit(c) = \frac{F_R}{\sum_{\substack{z \in PF_i \\ d(c,z) \leq \sigma}} \left[1 - \frac{d(c,z)}{\sigma}\right]} \quad (4.2)$$

In (4.2), every Pareto front ( $PF_i$ ), where  $i^{\text{th}}$  marks the order, has a unique reference fitness value ( $F_R$ ) that defines the maximum fitness value of an individual on the front.  $F_R$  is the minimum fitness value of an individual in  $PF_{i-1}$ . However, in the case of the  $PF_1$ ,  $F_R$  equals one. The Euclidean distance ( $d$ ) is measured between the chromosome ( $c$ ) and every individual inside the cluster ( $z$ ). The cluster defines a neighbourhood of individuals on  $PF_i$  grouped within a radius denoted as  $\sigma$ .

A pitfall of Deb's (2001) method is that he employs a fixed user defined  $\sigma$ , which may not adequately reflect the actual search space coverage, resulting in the assignment of poor fitness values. To overcome this shortcoming, Patelli (2011) introduced dynamic computation of  $\sigma$  for each front, which calculates the average distance between individuals. Equation 4.3 is used to calculate  $\sigma$ , which Patelli (2011) has shown the modification to be fruitful.

$$\sigma_i = \frac{\sum_{c,z \in PF_i} d(c,z)}{N} \quad (4.3)$$

where  $N$  defines the number of individuals in  $z$ .

However, both Deb (2001) and Patelli (2011) fail to actually reflect the fitness relationship between different ordered Pareto fronts because the geometric distance is not considered. The problem arises in how  $F_R$  is determined for the next Pareto Front. Currently,  $F_R$  is set as the lowest fitness value of an individual in  $PF_{i-1}$ . Imagine a situation where two Pareto fronts are closely positioned in the search space, therefore the objective values between the individuals in the Pareto fronts are only slightly dissimilar. However, the chromosomes in the 2<sup>nd</sup> order Pareto front will inheritably have a much lower fitness than their neighbouring Pareto front individuals. Therefore, the author of this work recommends the use of a coefficient, which more accurately reflects the distance between the Pareto fronts, so the fitness levels are more differentiated.

#### 4.2.2 Non Pareto Approaches

Poli et al. (2008) introduces a dynamic fitness function in the context of MOO problems, which is not dominance related. This approach requires the multiple objectives to be ranked in accordance of importance to the search. Initially, the population is evolved based on the fitness function defined in terms of the most important objective, until all chromosomes occupy a satisfactory space. The fitness function is then redefined to also consider the second most important objective and the population is again allowed to evolve. This process is continued until the population is situated in the required area of the objective space. This approach is computationally expensive because the population is constantly shifted (Patelli, 2011). In addition, the method is applicable only when the relative importance of the considered objectives is available beforehand.

Coello Coello et al. (2007) introduce the Insular Model to solve a MOO problem. The population is divided into subsets and each constitutes an island. Individuals belonging to the same island are evaluated relative to only one of the considered objectives. Individuals in different islands are encouraged to mate, therefore promoting offspring adapted with respect to multiple objectives. Figure 4.7 illustrates the insular model with two objectives and two islands, where each island evaluates and optimises a different considered objective.

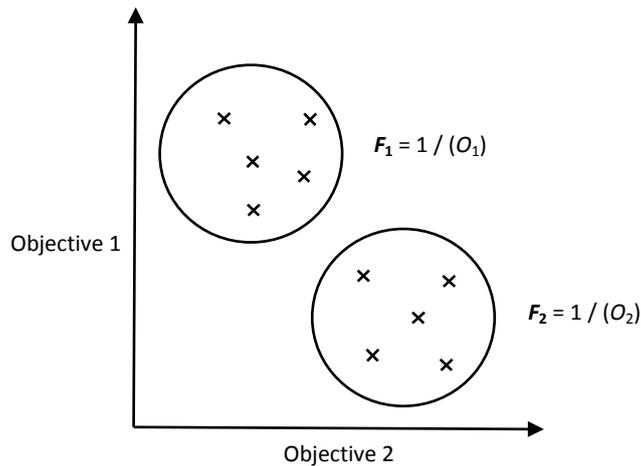


Figure 4.7 Insular model

### 4.3 Selection for Reproduction

This section solely relates to the selection of individuals for reproduction and does not discuss selection for the next generation, which is discussed shortly in the reinsertion section. The role of selection is to select individuals from the current population for the reproductive pool that are likely to generate well adapted offspring. To encourage the search towards the optimal region of the search space, selection should employ fitness values assigned to the individuals (Gendreau and Potvin, 2010). The adopted selection strategy is important as it influences the degree of accuracy and diversity in the evolution. The selection of individuals for reproduction can be configured deterministically or stochastically (De Jong, 2006). The former type does not base selection on probability, unlike the latter. A deterministic approach may restrict the number of times an individual is eligible to reproduce in a generation, whereas, the stochastic approach does not and individuals are randomly selected based on their fitness values.

A stochastic approach assigns an individual with a fixed probability for selection and parents are chosen randomly using a fitness-proportional probability distribution (De Jong, 2006).

Therefore, individuals with a greater fitness are assigned a greater probability mass to encourage accuracy in the search (De Jong, 2006). In addition, stochastic approaches provide diversity in the search, therefore increasing the robustness of the algorithm by reducing the probability of being trapped in a sub-optimal area of the search space (De Jong, 2006). The proportional roulette wheel selection, rank-based roulette wheel selection, tournament selection and uniform selection are the main stochastic selection approaches available. The fundamental difference between these strategies is the manner in which selection probabilities are distributed.

#### 4.3.1 Proportional Roulette Wheel Selection

Individuals are randomly selected for reproduction with a probability directly proportional to their fitness. The fitness of each individual represents a proportion of the roulette wheel, and the circumference equals the sum of all fitness values. After spinning the roulette wheel, the individual associated to the segment of disk that the pointer indicates is selected for reproduction. This approach encourages fitter individuals to participate in reproduction, therefore performing in the spirit of the survival of the fittest principle. At the same time, the roulette wheel preserves diversity by providing every individual with an opportunity for selection. The shortcomings of the roulette wheel selection are as follows. Selection based on relative fitness does not ensure convergence to the global optimum (Fogel, 2006), (Rudolph, 1994). The fitter individuals may dominate the reproductive pool thus preventing the reproduction of other individuals, resulting in a loss of genetic diversity in the offspring. Furthermore, in case the individuals in a population share a similar level of fitness, it may be difficult to explore new areas of the search space. Moreover, to use this selection approach on minimisation problems the fitness function for minimisation must be converted to a maximisation function. For example, the fittest individual for the TSP has the lowest route length. To overcome the latter problem the following rank based approach is introduced.

#### 4.3.2 Rank-based Roulette Wheel Selection

A ranking based selection approach is considered simpler and more efficient than the roulette wheel approach (Gendreau and Potvin, 2010). The individuals in a population are ranked in accordance to their fitness values. The roulette wheel disk portioning mechanism is similar to the previous case, with the difference that in the case of a ranking approach, the distance between two consecutive individuals is not reflected by their fitness values. Instead, the portion of disk awarded to individuals is in relation to their rank, with higher ranked

individuals being assigned greater disk space, as illustrated in Figure 4.8. Therefore, the selection is not dominated by a few individuals and diversity is maintained in the population. However, the selection method is likely to result in slow convergence compared to the fitness proportional roulette wheel approach because less disk space may be awarded to elite individuals and thus their ability to influence the search is diminished. The approach focuses on accuracy in the early stages of the evolution as the population contains a wide range of fitness values.

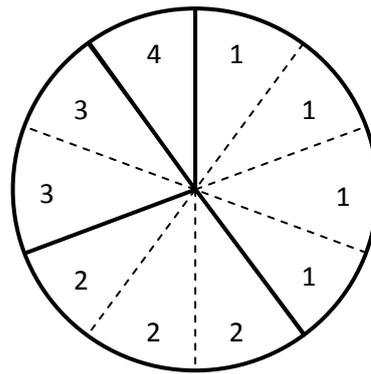


Figure 4.8 Four individuals represented on a rank-based roulette wheel

#### 4.3.3 Tournament Selection

The most common selection approach for the genetic algorithm is the Tournament Selection, due to its efficiency and simple implementation (Goldberg and Deb, 1991). A number of individuals are randomly selected from a population to compete in a tournament and the quantity selected is referred to as the tournament size. The individual with the greatest fitness is selected to become a parent. A large tournament size results in dominant individuals taking over the population leading to diversity loss. As a consequence, the selection process becomes increasingly greedy and leads to a decline in genetic diversity in the population with time. To overcome this shortcoming the tournament size is commonly set at two (Razali and Geraghty, 2011).

#### 4.3.4 Uniform Selection

The approach randomly selects individuals to become parents without considering their fitness values. Every individual has a uniform probability for selection. The approach is diversity friendly and does nothing to encourage accuracy in the search.

#### 4.3.5 Select All Selection

Unlike the previous methods, this approach is deterministic. In a “select all” approach fitness is not a consideration as the entire population is copied to the reproductive pool. This is the most diversity prone strategy, as all individuals will generate offspring. The strategy is applicable, if the area of the search space with the optimal solution is known, then diversity will assist the exploration of the space. However, when the location is unknown, this strategy is unlikely to lead to a convergence towards the optimal search space. In addition, the computational time may be wasted because parents with poor fitness are considered for reproduction.

#### 4.3.6 Elitist Selection

The elitist selection concept encourages the reproduction of the so-called elite individuals by always selecting individuals with the greatest fitness. The benefit of such an approach is that it accelerates the search process, therefore reducing the time necessary for the algorithm to produce a satisfactory solution (Patelli, 2011), (Koza, 1998). However, the excessive selection of elite individuals will reduce the level of genetic diversity in the population increasing the chance of local optima blockage. The elitist concept is applicable to both deterministic and stochastic approaches.

### 4.4 Genetic Operators

Genetic operators are applied to individuals in the reproductive pool with the aim of generating new offspring of a better fitness in relation to their parents. Reproductive variation is the primary source of exploration (De Jong, 2006). The most commonly applied operators are: crossover and mutation, which complement one another and neither is considered more useful than the other (De Jong, 2006). Other operators are encapsulation, decimation and expiry. Encapsulation like crossover is accuracy focused and decimation like mutation is diversity focused. In contrast, the expiry operator promotes accuracy and diversity in the search. The following describes the aforementioned operators in relation to the reproduction process in further detail.

#### 4.4.1 Crossover

Crossover refers to the exchange of genetic material between two parents in the reproductive pool to generate new offspring of improved fitness from their creators. A common approach is to pair parents to mate in order of insertion into the reproductive pool. The type of crossover

procedure adopted affects the level of accuracy and diversity in the search. The role of the crossover in the early stages of the evolution is to promote accuracy by exploiting current individuals in the population. However, as the diversity of the genotypes in the population diminishes, the mating of very similar parents will generate offspring with very similar genotypes to their creators, resulting in increased chances of a local optima blockage. In such a situation, the role of crossover is to promote greater diversity in the population.

There are two types of crossover extremes: null crossover and gene level crossover. Null crossover does not alter the genetic material of the parents, so the resultant offspring are identical copies of their creators. Therefore, no progress is made in terms of exploration. Gene level crossover produces offspring with the greatest level of diversity compared to their parents because each two genes occupying similar positions on their parents structure are swapped. The gene level crossover approach, also known as uniform crossover leads to excessive diversity; consequently the accuracy is unlikely to improve through the generations (Gendreau and Potvin, 2010). The gene level crossover procedure involves the assignment of probability values on all genes in the parent set. A crossover threshold probability controls the intensity of the operator, and those genes within the threshold are selected for crossover. This type of crossover is computationally expensive (Ashlock, 2006), because the number of exchanges between the parents are likely to be large. Figure 4.9 illustrates gene level crossover.

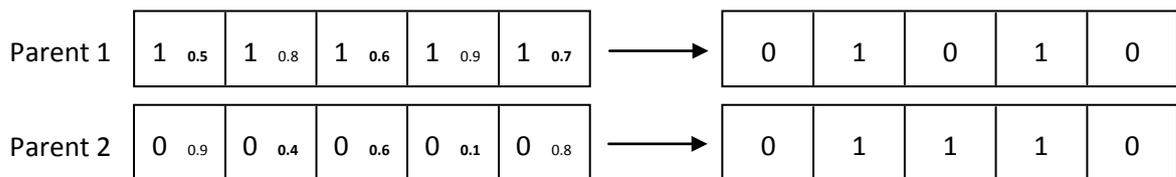


Figure 4.9 Gene level crossover, where crossover threshold equals 0.7

Unlike the aforementioned crossover procedures, n cut point crossover increases accuracy through the generations. The level of diversity provided by this crossover procedure increases with the number of n cut points. However, the level of accuracy diminishes once the number of cut points become too large because the valuable Building Blocks (BBs) are not maintained. The n cut point crossover selects n index points in each parent. The cut points in each parent are labelled in sequential order from one end of the parent to the other. The crossover will

occur on the same index, exchanging the latter part of each parent. The Figure 4.10 illustrates the single and two cut point crossover.

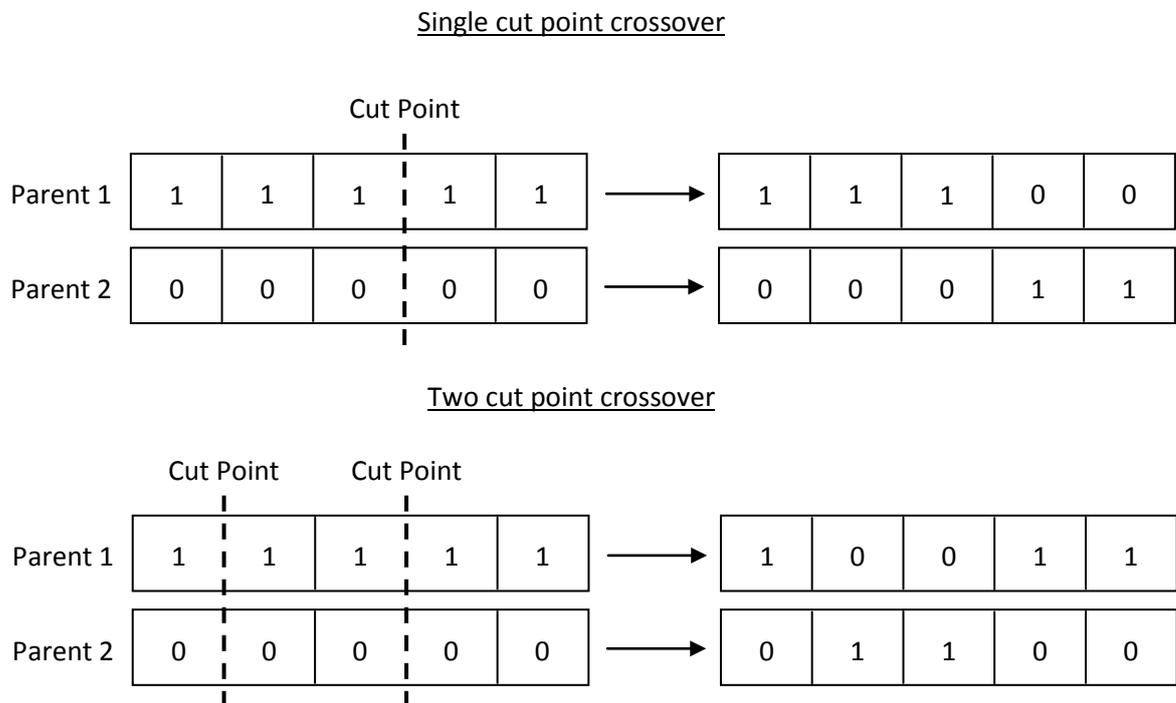


Figure 4.10 Single and two cut point crossover

The cut point selected for crossover can either be homologous or non-homologous. The former selects cut points with the same indices in parents. The resultant offspring will be of equal size to their parents. The crossover of genetic material between identical parents using homologous cut point will lead to zero diversity in the offspring from their parents. Consequently, the crossover operator is rendered useless because the genotype is not of new search space. This issue should be considered when pairing parents in the reproductive pool. The latter approach refers to the selection of different cut points indices of the parents. Subsequently, the issue of bloat arises in GA and GP where the size of an offspring increases from their parents, without an improvement in fitness (Langdon and Poli, 1998).

Crossover operators with a single or multi cut points are widely adopted (Patelli, 2011), (Spears, 1995). In contrast, Fogel (2006) experimental work for solving the travelling salesman problem showed that the uniform crossover on average outperformed the two point crossover, which in turn surpassed the single point crossover. The uniform crossover can advance the search, unlike the cut point crossover, hence its better performance. However,

the rate of increase in the fitness is greater for both cut point crossovers, when compared with the uniform crossover. In addition, the uniform crossover only surpasses the cut point crossovers once diversity has fallen in the population.

#### 4.4.2 Mutation

Mutation is an operator which alters one parent only, in terms that it replaces a gene value with another from the available alleles. Alleles are defined as a set of legal values that can be assigned to a gene. The role of mutation is to preserve diversity in the population, (Deb, 2009). It does so, by gradually introducing new genes into the population (Fogel, 2006), (Ashlock, 2006). A successful mutation assigns gene values that are useful to the evolution. The level of mutation can have a positive or negative effect on the evolution (Deb, 2009). The introduction of mutation can prevent the search from being trapped in a particular space and can guide it to new areas. In contrast, high levels of mutation have a destructive effect on the accuracy of the search because in each generation offspring are too diverse from their parents, therefore search spaces are not effectively explored.

Apart from the number of genes selected to be mutated, the manner of modification has a bearing on the level of diversity (De Jong, 2006). The gene level mutation procedure provides the greatest level of diversity and is comparable to the gene level crossover procedure (Patelli, 2011). A gene value is only mutated if the random probability assigned exceeds the mutation probability threshold, which controls the level of mutation in the search. To prevent unacceptable accuracy loss, the mutation rate is usually set low (Deb, 2009). Figure 4.11 illustrates the gene level mutation procedure.

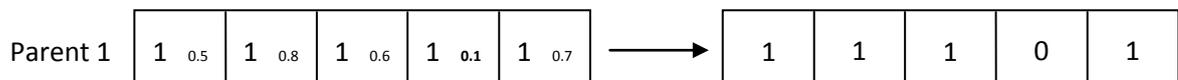


Figure 4.11 Gene level mutation, where mutation threshold equals 0.1

At any instance in the evolution the mutation operator can replace a gene value with another from the set of alleles. Therefore, mutation can reintroduce into the population gene values that were discarded during the evolution. Whereas, gene level crossover is limited to gene values inside the current population. Hence, the reason why mutation is required even in the presence of the diversity promoting gene level crossover. However, the ability of the mutation operator to diversify a population that has converged through crossover is deemed slow

because of the low mutation rate (Fogel, 2006). Therefore, mutation by itself is not sufficient to maintain population diversity (Deb, 2009) and subsequently the promotion of diversity through crossover is important.

#### 4.4.3 Encapsulation

Koza (1998) the encapsulation operation protects the genetic material inside Building Blocks (BBs) from being modified by genetic operators. These blocks have a good influence on the overall fitness of the host individual. However, the main challenge relative to encapsulation is identifying the well adapted BBs and estimating their quality in terms of the amount of positive influence they have on the fitness of the host individuals. This may be done by means of various heuristics (Patelli, 2011), however this is outside the scope of this work. The operator is accuracy focused. Like mutation, the operator is asexual.

#### 4.4.4 Decimation

The decimation operation probabilistically deletes individuals in the offspring pool based on their fitness (Koza, 1998) and maintains a proportion of the population for the next generation. New individuals have to be constructed for the next generation to compensate for the population deficit caused by decimation. The purpose of the decimation operation is to reintroduce diversity back into the population on a scale unprecedented to that of crossover (Koza, 1998). The operator is extremely diversity friendly and is only appropriate where the current population has a very large percentage of individuals of poor fitness or the genotype diversity has fallen to an unacceptable level. The operator is applied to prevent the persistence of the aforementioned situations in the evolution, which wastes valuable computational time.

#### 4.4.5 Expiry Date

Expiry Date operator protects certain individuals in the reproductive pool from being deleted for  $x$  generations. The operator promotes both accuracy and diversity. Accuracy is maintained in the population because fit parents are not replaced for  $x$  generations. However, genetic operators are applied to the parents with an expiry date, although their offspring cannot replace them. Diversity is maintained in the population by deleting individuals with an expiry date after a number of generations. The expiry date can relate to a fixed number of generations or to the objective value. In the latter case, after a number of generations without improvement in the fitness value the individual is deleted from the population.

#### 4.5 Selection for Reinsertion

This section deals with the selection of individuals for the following generation and has no bearing on the selection of individuals for reproduction, discussed previously. The reinsertion method impacts the level of accuracy and diversity in the search. The reinsertion of champion individuals into the population will improve the accuracy of the search, at the risk of premature convergence. In contrast, the reinsertion of individuals without relation to fitness values provides excessive diversity and provides no promotion in terms of accuracy.

De Jong (2006) classified two models for reinsertion: non-overlapping and overlapping. The former approach restricts the survival of individuals to one generation because only the offspring are eligible for reinsertion. The pitfall of such a model is that offspring with diminished fitness relative to their parents are guaranteed to survive in spite of the incurred loss in overall population fitness. Therefore, the model does not allow the encouragement of accuracy in the search. In contrast, the latter approach provides a much stronger selection pressure because individuals can be reinserted from the current population as well as from the offspring pool.

The following describes some of the reinsertion methods discussed in Asklock (2006). The Random Replacement procedure randomly selects a parent to be replaced irrespective of the fitness. The Proportional Roulette Wheel Replacement is identical to proportional roulette wheel selection discussed in selection for reproduction, but with one difference. Individuals are selected to be replaced with a probability inversely proportional to their fitness. The Rank-based Roulette Wheel Replacement is identical to the rank-based roulette wheel selection discussed in selection for reproduction, but rank individuals in the opposite order. The Absolute Fitness Replacement procedure replaces the least fit individuals in the population with the offspring, therefore the approach is considered accuracy friendly. Local Elite Replacement the fitness of the two parents and two offsprings are compared and the two fittest individuals replace the parents in the current population. This approach encourages accuracy. Random Elite Replacement, each offspring is compared against a randomly selected individual in the population and only replaces it, if at least as good. There is a possibility that the disregarded offspring are better quality than some of the individuals in the population, therefore this approach does not encourage accuracy in the search.

#### **4.6 Termination Criterion**

In principle the GA is a stochastic approach which could run forever (Gendreau and Potvin, 2010), therefore a stopping criterion is required. Koza (1998) as stated in Patelli (2011) referred to three termination situations: generational, evaluation based and exceptional. The generational situation is the most popular and entails algorithm termination after a given number of iterations (Koza, 1998). Another termination condition not mentioned by Koza (1998), but closely related to the generational situation is “clock wise”, where the evolution terminates once the maximum computational time limit has expired. Ideally, GA should terminate once the solution is found, however there is no way to guarantee that the global solution will be reached (De Jong, 2006). Therefore, the evaluation criterion should enforce the evolution to terminate once a solution minimises the objectives to an acceptable extent. The exception condition stops the evolution when a certain situation occurs in the search, such as diversity loss. The evolution should terminate once diversity falls below a certain level and genetic operators are incapable of reintroducing diversity (Patelli, 2011), (Gendreau and Potvin, 2010). Diversity may relate to the phenotype or genotype. The individuals in the current population may have similar fitness values, yet that does not necessarily imply a diversity loss at genotype level. Therefore, to avoid premature termination, convergence should relate to the level of genotype diversity in the population (De Jong, 2006). Hence, that is the reason why genotype diversity criterion for termination is more commonly employed than the phenotype (Gendreau and Potvin, 2010).

#### **4.7 Multi-objective evolutionary algorithms**

This section reviews the most prominent multi-objective evolutionary algorithms (MOEAs) research works, as the paradigm is applied here. Non-elitist and elitist multi objective optimisation (MOO) approaches are discussed. A non-elitist approach cannot guarantee the survival of the best individual into the next generation, irrespective of the high selection pressure that may exist. Therefore, there is a risk that the next population is situated in an inferior area of the objective space than the previous one (Patelli, 2011). However, these approaches are adopted to promote greater search diversity, which is known to prevent premature convergence to local optima. In contrast, an elitist approach ensures the preservation of the best individual for the next generation, therefore always improving the population fitness level. The primary principle governing this approach is that the use of elitists will improve the probability of global optimum convergence (Deb, 2009). The reader is

referred to Coello Coello et al. (2007) and Deb (2009) for an extensive review of the subject area.

#### 4.7.1 Non Elitist MOO Evolutionary Approaches

One of the early attempts to use GAs for solving MOO problems is the Vector Evaluated Genetic Algorithm (VEGA) by Schaffer (1984). The algorithm considers one sub-population for each of the considered objectives, thus promoting the survival of specialised individuals (i.e., highly efficient at optimising a given objective function). The working principle of VEGA is that by allowing the best individuals from all sub-populations to exchange genetic material, the resulting offspring would represent feasible trade-offs with respect to the entire set of objectives.

The main challenge of VEGA is that it fosters two opposing phenomena: the mating of champions from different sub-populations encourages the creation of individuals with an acceptable performance across all objective functions; whereas the speciation process in each sub-population develops offspring with a significant bias towards one particular objective (Deb, 2009). Since the two phases are run sequentially during the algorithm, the diversity of the first order Pareto front is diminished over generations and ultimately causes the algorithm to converge towards a near optimal region of the search space (usually with respect to a subset of objective functions).

The first GA to adopt a rank based fitness assignment method to solve a MOO problem was Fonseca and Fleming's (1993) multi-objective genetic algorithm (MOGA). The principle of the algorithm was to increase the level of search accuracy, by awarding a greater selection pressure to individuals with more dominance. In addition, a niching operator was adopted to redirect the selection pressure towards individuals located in less populated areas of the first order Pareto front, which should allow for the construction of a diverse nondominated front.

From a critical viewpoint, the main shortcoming of the niching operator is that the user must define the niching parameter  $\sigma_{share}$ , and when incorrectly set, this may cause the algorithm to run inefficiently (Van Veldhuizen and Lamont, 2000). Another issue of concern relates to the potential for slow algorithm convergence to the Pareto optimal set, caused by a reduction in selection pressure. This situation may occur through fitness sharing, where an individual on a lower order Pareto front may have their fitness value diluted because of a high niche count.

Therefore, an individual on a higher order Pareto front may have a greater selection pressure than one on a lower order front Deb (2009).

Srinivas and Deb (1994) proposed the Nondominated Sorting Genetic Algorithm (NSGA) to address the selection bias in VEGA caused by speciation. To address this issue the nondominating ranking procedure in Goldberg (1989) and a fitness sharing mechanism were implemented in the fitness computation stage. The fitness sharing mechanism reduced the selection pressure of individuals located in densely populated areas on the 1<sup>st</sup> order Pareto front, thereby encouraging the selection towards less crowded areas. Srinivas and Deb's (1994) experimental studies compared the NSGA to VEGA, and both approaches demonstrated the ability to converge to the Pareto optimal set. However, NSGA demonstrated the ability to maintain a steady distribution over the 1<sup>st</sup> order Pareto front to the end of the simulation, which was not replicated by VEGA.

From a critical point of view, the nondominating ranking procedure used to determine dominance is computationally expensive (Coello Coello et al., 2007). In addition, the NSGA performance is sensitive to the user defined preset niching parameter  $\sigma_{share}$ , which may not be very effective for the entire search.

The first MOGA to employ a tournament selection method based on Pareto dominance was Niche Pareto Genetic Algorithm (NPGA) in Horn et al. (1994). The tournament selection method randomly selected two individuals from the current population to compete based on their dominance relationship. A nondominated individual would win the tournament to encourage convergence to the Pareto optimal set. In the case of a nondominated relationship with respect to each other, the individual located in the least dense region of the partially filled next generation population was selected as the winner, in order to achieve the objective of finding a diverse spread of nondominated solutions (Deb, 2009).

From a critical viewpoint, the NPGA does not perform fitness computation, unlike some MOGAs. Therefore, this method saves on the computational resources required for this additional step. In addition, the complexity of the NPGA is not linearly dependent on the number of objective functions under evaluation. Thus, the algorithm is likely to be computationally efficient for solving problems with a large number of objectives (Deb, 2009). In contrast, a shortcoming of the approach is that two parameters require defining:  $\sigma_{share}$  and

$t_{dom}$ . The former parameter is used to calculate the nich count required for fitness sharing and the second relates to the subset population size. The accurate setting  $\sigma_{share}$  parameter is more important in NPGA than NSGA, as each individual situated within that distance will provide an equal contribution to the nich count. Whereas, in the latter method the contribution made by each individual is relative to their position from the seed (Deb, 2009). In addition, Horn et al. (1994) provided no guidelines to define  $t_{dom}$ . However, their experimental study suggested the size of the subset population to be approximately 10% of the current population. The setting of  $t_{dom}$  is important because it is used to control the selection pressure and influence the convergence speed of the algorithm. Their experimental results demonstrated  $t_{dom}$  sensitivity on the performance of NPGA. It was realised that a too small  $t_{dom}$  would create a noisy Pareto front with a risk of slow convergence. In contrast, if  $t_{dom}$  was set too large the computational complexity of the algorithm would increase and there would be a risk of premature convergence.

#### 4.7.2 Elitist MOO Evolutionary Approaches

Deb et al. (2002) proposed the Nondominated Sorting Genetic Algorithm II (NSGA-II) to address the shortcomings that existed with Srinivas and Deb's (1994) NSGA: high computational complexity of the nondominated sorting algorithm; non-elitist approach and the need for a preset  $\sigma_{share}$  parameter for fitness sharing. Deb et al. (2002) overcame the first issue by introducing a new nondominating sorting algorithm with a lower computational complexity, although the storage requirements for this method were higher. To address the second issue, a crowded tournament selection procedure was applied to guarantee elitist survival. However, where individuals shared the same Pareto front, the solution located in a less crowded region would survive to promote greater solution diversity. To address the third issue a crowded-comparison procedure was proposed to compute neighbourhood solution density, without the need for the user defined  $\sigma_{share}$  parameter. This procedure proved to be more computationally efficient.

Deb et al. (2002) experimentally validated NSGA-II's superiority over Knowles and Corne's (1999) Pareto Achieved Evolutionary Strategy (PAES) and Zitzler's (1999) Strength Pareto Evolutionary Algorithm (SPEA), by showing the algorithm's convergence to a diverse Pareto optimal front. In addition, NSGA-II was simulated on a constrained MOO problem and compared against the Ray-Tai Seow algorithm. Again, NSGA-II had a better convergence to the Pareto optimal set and an improved distributed spread along the 1<sup>st</sup> order Pareto front.

NSGA-II was proposed to address the shortcomings with NSGA; however in the study no experiments were conducted to compare both methods with respect to convergence to a diverse Pareto optimal front, therefore NSGA-II's superiority cannot be verified. In addition, Deb et al. (2002) arbitrarily set NSGA-II parameters, which may have an adverse effect on the algorithm performance. Instead an improved approach would have been to experimentally tune parameters to the test problem. Moreover, NSGA-II repeated refinement of the population over the generations to ensure diversity on the 1<sup>st</sup> order Pareto front was computational resource intensive (Deb, 2009). Furthermore, the refinement process may cause solutions in a region desired by the decision maker to be deleted. However, this process is important to maximise the probability of finding a solution required by the decision maker, considering that their requirements may not be known prior to the experimental study.

Knowles and Corne (2000) proposed the Pareto Achieved Evolutionary Strategy (PAES). Their main contribution was a selection for reinsertion method that evaluated the dominance relationship between a parent and their offspring. The nondominated individual was selected for reinsertion. In contrast, a nondominance relationship gave rise to a set of possible outcomes, which were assessed using an external population of nondominated individuals, similar to the one used in Horn et al. (1994). In the event an offspring shares the same dominance relationship with the individuals in the external population, the selection decision is made with respect to maximising diversity in the search space. A crowding procedure superimposed a grid over the objective space containing the nondominated solutions and the parent or offspring located in the least dense cell was selected for reinsertion.

The external population plays a positive role in finding a diverse nondominated set. On a different issue of importance, the crowding procedure used a standard grid cell size, which contributed to the uniform spread of nondominated individuals. However, the cell depth must be defined by the user and this may lead to an inefficient setting. For instance, population diversity may not be maintained with a large cell depth and it may prove computationally expensive to evaluate a greater number of smaller cells. In addition, the cell density may not provide an accurate representation of the search space coverage, as the individuals could be located anywhere within the cell. Instead, the average distance between the individuals in a cell is a more accurate diversity measurement (Maceachren, 1985). Moreover, an increase in the number of objective functions evaluated would increase the number of grid cells

exponentially, therefore increasing the difficulty of generating a diverse spread of nondominated solutions.

Zitzler and Thiele (1999) introduced the Strength Pareto Evolutionary Algorithm (SPEA). The method applied a fitness assignment procedure similar to the one introduced by Fonseca and Fleming (1993), where an external population of nondominated individuals was used to compute fitness values (Deb, 2009). The fitness assignment procedure assigned a strength value to each external population member relative to the number of individuals it dominated in the current population. The fitness value of a current population member was then calculated by adding the strength value of each nondominated individual in the external population that dominated it (Deb, 2009), (Coello Coello et al., 2007). Thereafter, a tournament selection procedure was applied to the combined current and external population. A clustering algorithm was used to update the external population with a fix number of diverse nondominated solutions. The clustering algorithm iteratively grouped neighbouring individuals until the number of sets amounted to the population size. Then a single individual with the minimum average distance among others was selected to form the diverse nondominated set (Deb, 2009).

From a critical point of view, the fitness assignment phase is likely to be computationally resource expensive compared to MOEAs with a single population. However, the SPEA clustering algorithm is parameter-less and is considered to be more robust than NSGA, which requires the tuning of the niching parameter  $\sigma_{share}$ . Another aspect of importance relates to the impact the external population size has on the selection process. For instance, a large population size will consume greater computational resources because of the increase in the number of evaluations required (Van Veldhuizen and Lamont, 2000). In contrast, a smaller population will make it difficult to achieve a diverse spread of nondominated solutions. On a different issue of importance, the individuals on the same nondominated set were not assigned a uniform fitness value, but instead their values were relative to the number of individuals they each dominate. Therefore, a bias exists that favours certain individuals on the 1<sup>st</sup> order Pareto front more than others. However, in the absence of objective weighting information such a bias is unjustified. Also, the tournament selection procedure is likely to select the nondominated individuals from the external population rather than those individuals in the current population. This may help achieve greater convergence to the Pareto

optimal region, however does little to promote the construction of a diverse nondominated set.

Van Veldhuizen (1999) proposed the Multiobjective Messy GA (MOMGA). The goal was to investigate the link between building blocks and MOEA search. The method consisted of a primordial and juxtapositional phase. The primordial phase identified and maintained important building blocks for the search. For instance, an initial population was constructed containing all building block combinations of a specified size. Thereafter, a set of template chromosomes were used to evaluate the building blocks and to ensure the survival of blocks that optimise each objective function. The juxtapositional phase recombined those building blocks to generate optimal solutions using the cut and splice genetic operator. The populated solutions were compared to one another using the Horn et al. (1994) tournament selection procedure, where a fitness sharing scheme was applied.

From a critical point of view the algorithm is computationally resource expensive in the early generations because the primordial phase requires the evaluation of a large population, where every possible building block is represented. Instead, a more computationally efficient approach would have been to realise a diverse set of building blocks for the initial population. Using a different template for each objective function prevents a bias in the search towards any one particular region, as would have been the case with a single template. Moreover, an external population archives the nondominated individuals found in the search. This population is important to prevent the stochastic nature of the algorithm from eliminating elite solutions.

Osyczka and Kundu (1995) proposed a distance based Pareto genetic algorithm (DPGA), which computed fitness values for nondominated individuals relative to their distance from an external population of elites. The fitness value of an individual was the sum of the spatially closest elite fitness and the minimum Euclidean distance between the pair. Therefore, the method assigned a large fitness value to a distant nondominated individual from the external population of elites, which was critical to achieving a diverse Pareto optimal set. For instance, a distant nondominated solution that dominates any external population elite must be located closer to the Pareto optimum set. Furthermore, a distance nondominated solution in the same dominance relationship with the external population would support the generation of a diverse Pareto optimal set.

From a critical standpoint, the benefit of storing an unlimited number of nondominated solutions inside an external population was to provide a decision maker with a wider choice. However, this choice is likely to increase the algorithmic complexity, especially as the archive is utilised in the fitness computational stage. In recognition of this issue, Zitzler and Thiele (1999) restricted the external population size. Osyczka and Kundu (1995) did not employ a fitness sharing method and therefore the shortcomings of using a niching operator were not experienced as with many other MOEAs. However, the fitness assignment method suffered from a major weakness, as the elite fitness was sensitive to the order of insertion into the external population. Therefore, it was possible for an elite residing in a densely populated area to have a higher fitness than one located in a less crowded area, as shown in Figure 4.12. As a result, the method may not generate a diverse set of nondominated solutions.

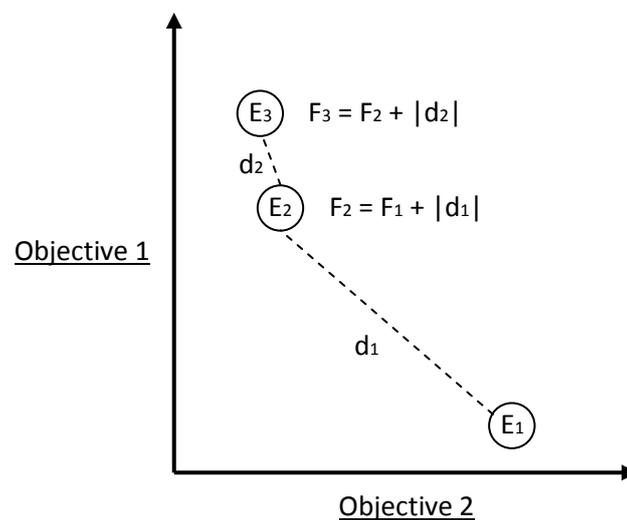


Figure 4.12 Elite individual E1 has a lower fitness than E2 and E3

Table 4.1 summarise the contributions of the MOEA research works reviewed above.

Table 4.1 MOEA research works review

Method Type	Method	Advantage(s)	Disadvantage(s)	Publication
Elitist	Nondominated Sorting Genetic Algorithm II (NSGA-II)	Most prominent MOGA	Algorithm convergence property loss, as the nondominated set exceeds the population size	Deb et al. (2002)
		No niching operator required	Cycle between converging towards the Pareto optimal set and finding a distributed set	
		Theoretical proven to lead to the Pareto optimal set		
	Pareto Achieved Evolutionary Strategy (PAES)	Grid based crowding procedure capable of finding an evenly distributed Pareto optimal set	Algorithm complexity increases with the objective dimension size	Knowles and Corne (2000)
			Preset cell depth parameter	
	Strength Pareto Evolutionary Algorithm (SPEA)	Clustering method does not require parameter setting	Bias towards certain nondominated individuals	Zitzler and Thiele (1999)
		Exhibits a convergence proof to the Pareto optimal set	Clustering algorithm computational complexity is greater than the crowding strategy of NSGA-II	
	Multiobjective Messy GA (MOMGA)	Construction and maintenance of high quality building blocks	Algorithm is computationally expensive at the early generations of the search	Van Veldhuizen (1999)
			Preset $\sigma$ share parameter	
	Distance based Pareto genetic algorithm (DPGA)	Single fitness metric used to achieve the MOO goals	Algorithm computational complexity increases with generations, as elite archive size is not restricted	Osyczka and Kundu (1995)
		Fitness assignment method is sensitive to the order of insertion inside the elite archive		
Non Elitist	Niched Pareto Genetic Algorithm (NPGA)	Algorithmic complexity not overly dependent on the number of objective functions	Method performance is more sensitive to $\sigma$ share parameter than NSGA	Horn et al. (1994)
		No fitness computation performed		
	Nondominated Sorting Genetic Algorithm (NSGA)	Pareto optimal set convergence, as fitness is based on the non-dominated sets	Nondominated sorting algorithm has a high computational complexity	Srinivas and Deb (1994)
			Preset $\sigma$ share parameter	
			Method performance is sensitive to $\sigma$ share parameter	
	Multi objective Genetic Algorithm (MOGA)	Simple fitness assignment scheme	High selection pressure potential for upper order Pareto set individuals	Fonseca and Fleming (1993)
	Vector Evaluated Genetic Algorithm (VEGA)	Easy to implement	Bias towards champion solution of each objective	Schaffer (1984)
		Find solution near to the individual best of each objective	MOO goal for diversity is not met	

This chapter provides critical insight into the structure and practical use of the most significant EAs documented in the literature, with special focus on MOEAs. EAs represent powerful computational tools for solving computational challenging optimisation problems. Moreover, EAs exhibit a significant potential with respect to solving a wide range of practical problems, given their inherent extensibility. Specifically, the generic EA framework is easily adjustable to fit the requirements of various implementation scenarios, by seamlessly integrating additional tools and custom algorithms (Coello Coello et al., 2007). A framework for such extensions is proposed in the following chapter.

## 5. METHOD

The methodology chapter is divided into two parts: the first introduces the routing heuristic, RouteAlg, for the travelling salesman problem with simultaneous delivery and pickup (TSPSDP) and the second presents the main contribution of this research, SDPmethod, to solve the multi-objective vehicle routing problem with simultaneous delivery and pickup (VRPSDP).

### PART 1 - RouteAlg

A routing algorithm called RouteAlg is herein proposed to solve the Travelling Salesman Problem with Simultaneous Delivery and Pickup (TSPSDP). This algorithm is of a new design and comprises of four heuristics: Modified Nearest Neighbourhood (MNN) algorithm (Section 5.1), Reverse procedure (Section 5.2), Ejection/Reinsertion (EjRi) method (Section 5.3) and 2-opt/Or-opt method (Section 5.4) respectively, in order to determine a high quality solution for the NP-hard problem, as defined in Chapter 3. The Modified Nearest Neighbourhood (MNN) algorithm generates a reasonably good solution to the relaxed version of the TSPSDP, known as the Travelling Salesman Problem (TSP). The Reverse procedure reverses the orientation of the TSP route, in order to induce TSPSDP feasibility. The Ejection/Reinsertion (EjRi) method also guides an infeasible solution towards a TSPSDP feasible space. Finally, the 2-opt/Or-opt method uses intra-route operators in order to find an optimised solution. A heuristic based solution procedure is deliberately chosen to better manage the tradeoffs between accuracy and computational resource consumption.

#### **5.1 Modified Nearest Neighbourhood (MNN) algorithm**

The MNN algorithm is a routing method used to solve the Travelling Salesman Problem (TSP), which is a relaxed version of the TSPSDP problem. A simplified version of the problem is solved given that the two problems are identical in the absence of vehicle capacity violations, therefore it may be hypothesised that a high quality solution for the TSP will share traits with a similar quality TSPSDP solution. This claim is supported by the experimental results in Chapter 6, where the solutions are thoroughly evaluated.

The MNN algorithm is an adaptation of the Nearest Neighbourhood (NN) algorithm because it is known to produce a reasonably good Hamilton cycle for the Euclidean TSP at a low

computational cost (Gutin et al., 2001). However, the NN algorithm has a greedy nature, therefore, the MNN algorithm is used in this work in order to hedge against the greedy nature of NN algorithm, whilst not hindering the aforementioned advantages. MNN algorithm will generate a set of diverse routes. Figure 5.1 outlines the pseudo code for MNN algorithm.

```

1  MNN() returns G
2  Depot → 0
3  Initialise G: [0, 1, 2, 3, ..., N, 0]
4  Total Length TL(G) = ∞
5  for i=1:N
6      Initialise G': [0, i]
7      Initialise unvisited_list: [1, 2, 3, ..., N] \ [i]
8      while (unvisited_list is not empty)
9          find j from unvisited_list, where aij is min
10         add j → G'
11         unvisited_list = [j≠i]
12         current node i=j
13     end
14     add 0 → G'
15     if TL(G') < TL(G)
16         G=G'
17     end
18 end

```

Figure 5.1 MNN algorithm pseudo code. Notations: G - graph; G' - new graph; TL - total length; i,j - number of nodes; a<sub>ij</sub> - edge between nodes i and j; unvisited\_list – list of unvisited nodes.

The graph (G) defines a Hamilton cycle, which starts at the depot (0) and connects a set of nodes (i=1, N) before terminating at 0. The initial G is arbitrarily constructed to include all nodes with a total length (TL) preset to infinity (∞). For each node i, the creation of a graph (G') of a shorter length than the recorded best solution G is attempted (lines 4 to 17). The list of unvisited nodes unvisited\_list is initialised to contain all nodes except i. At each step, node j from unvisited\_list with the shortest distance a<sub>ij</sub> is found and added to G', until all nodes from unvisited\_list have been added (lines 7 to 9). If G' is shorter than the best so far G, it is updated to G' (lines 14 to 15).

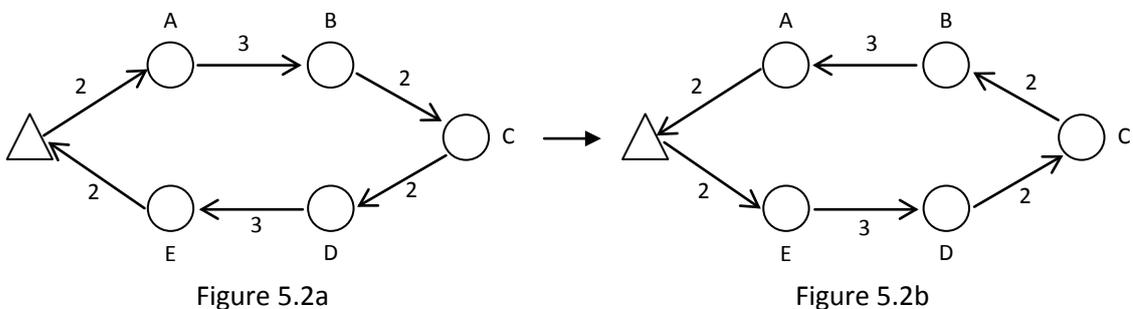
A simple extension to the NN algorithm is to select a different starting node every time the problem is rerun, in order to generate a number of solutions, where the best known is selected. However, this approach is not applicable here because the TSPSPD formulation defines the depot as the source node for the Hamilton cycle. To overcome this issue, the MNN algorithm selects a different first node to be serviced after the depot in every run, in order to

generate a range of cycles. The proceeding node insertion strategy of the MNN algorithm is identical to the NN algorithm, where a greedy selection is made at every stage. The Hamilton cycle with the minimum weight is selected as the initial solution, since a well constructed initial route will normally generate a good final solution (Jun and Kim, 2012).

The level of diversity introduced by the MNN algorithm is limited as additional computational resource consumption at this stage of RouteAlg is not recommendable because the resultant solution would not be built in accordance to TSPSDP feasibility.

## 5.2 Reverse procedure

The second component of the RouteAlg is the Reverse procedure, which is used to guide the MNN algorithm solution towards a TSPSDP feasible space. In the case that a violation is encountered, the cycle is reversed in an attempt to reduce vehicle capacity infeasibility, as in Wassan et al. (2008), Nagy (1996). However, the Reverse procedure does not guarantee a reduction in the level of TSPSDP infeasibility and in certain instances may have an adverse effect. In this work, the reverse route will replace the incumbent, if TSPSDP feasibility is improved. The subsequent weight of the cycle remains unchanged because the same edges are presented, as shown in Figure 5.2(a,b).



Wassan et al. (2008) experientially tested their method inclusive and exclusive of the Reverse procedure. The results indicated that the former scenario led the search towards a higher quality area of the solution space. In addition, the procedure had an insignificant computational cost  $O(N)$ , where  $N$  is the number of nodes, which further justified its implementation. The aforementioned reasons underpin the implementation of the Reverse procedure inside the RouteAlg.

### 5.3 Ejection/Reinsertion (EjRi) Method

The third component of the RouteAlg is the EjRi method, which also modifies an infeasible TSPSDP Hamilton cycle in order to guide the search towards a feasible space. The proposed method is designed to induce time window feasibility at the nodes for the TSPSDPTW. It is important to mention here that the consideration of time windows is outside the scope of this research. In spite of this, the EjRi method will be discussed here inclusive of time windows to demonstrate its setup to a wider audience. The requests that have infringed the capacity and time window constraints are removed from the route. Although, the route feasibility is not recalculated each time a request is ejected because the optimal sequence of ejection is unknown. A consequence of the employed ejection method is that a greater number of requests may be removed from the cycle than is actually necessary. The ejected requests are then reinserted inside the cycle using Equation 5.1, which defines a unique position for reinsertion, in order to increase the likelihood of TSPSDP feasibility.

$$\text{Tour}=[\text{Depot}, \text{Ejected Time Window}, \text{Cycle excluding Ejection}, \text{Ejected Pickup}, \text{Depot}] \quad (5.1)$$

The 'Ejected Time Window' nodes are added between the depot and the first node of the 'Cycle', in ascending order of their time window restriction, with the tightest time window nodes being serviced first. This ordering is typical for routing problems with time windows (Potvin and Rousseau, 1995) because the probability that the nodes are serviced within their respective time windows is increased. In contrast, the ejected pickup nodes are added between the last node of the 'Cycle' and the depot, in ascending order of net load increase on the vehicle capacity. The ordering of net pickup nodes was suggested by Chen and Wu (2006). These nodes are reinserted after the 'cycle' because the vehicle capacity slack is greatest at this point. In the situation, where multiple nodes share the same time window or net load capacity, the order of reinsertion is random. Figure 5.3(a,b) illustrates a solution generated to a problem using the EjRi method.

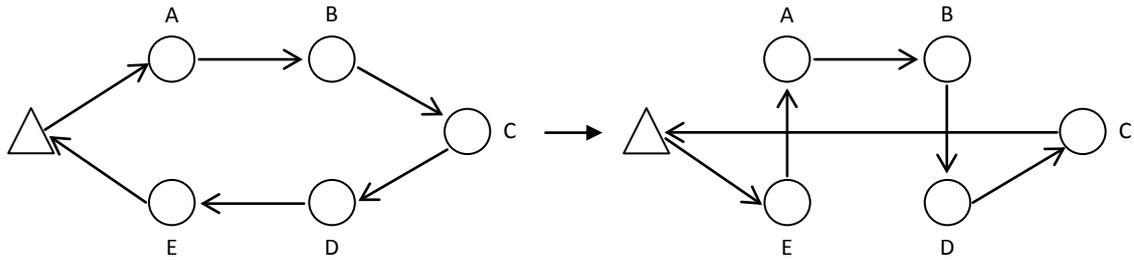


Figure 5.3a

Figure 5.3b

Figure 5.3a illustrates a route with unfeasible nodes: vehicle capacity infeasible 'C' and time window infeasible 'E'. These nodes are ejected and reinserted into the route using the above approach, as shown in Figure 5.3b.

EjRi method is inspired by the approach adopted in Bianchessi and Righini (2007), where the nodes that violated the capacity feasibility constraints were ejected from the cycle and were reinserted in reverse order of ejection. Unfortunately, their sequence of reinsertion does not guarantee capacity feasibility along the route. In addition, their approach did not consider time window constraints, which are considered by EjRi method.

A limitation of the EjRi method is that it does not guarantee a TSDSDP feasible solution and may in certain circumstances lead to further infeasibility. In addition, the reinsertion of nodes inside the cycle in order to gain a feasible solution may deteriorate the solution quality of the cycle. Another consequence relates to the fact that the EjRi method does not eject nodes that violate the maximum route capacity constraint, because Formula 5.1 does not consider such a restriction.

#### 5.4 2-opt/Or-opt method

A 2-opt/Or-opt method is a local optimisation technique. In the context of this research, 2-opt/Or-opt method is applied to the Hamilton cycle determined by the EjRi method, in order to overcome the limitations of the method and to further minimise the weight of the cycle. There are many different ways to modify a cycle in order to find an improved solution. In the context of this research, two intra-route operators are applied: 2-opt and Or-opt. An intra route operator rearranges a proportion of the cycle in order to define a neighbouring solution to the incumbent. The 2-opt/Or-opt method applies the 2-opt and Or-opt operators in a loop, which have also been used in combination in the following VRPSDP literature: Jun and Kim

(2012), Subramanian et al. (2010a) and Chen and Wu (2006). The loop will terminate when an improved neighbourhood solution can be found by an operator. In addition, the operators will only replace the incumbent cycle with a feasible TSPSDP solution. The 2-opt and Or-opt operators are described in greater detail below.

#### 5.4.1 2-opt operator

The 2-opt operator was introduced by Lin (1965) and it has been used extensively to improve vehicle routing solutions (Taillard et al., 1997), as it quickly converges to the local optimum (Potvin and Rousseau, 1995). The 2-opt operator removes two non-adjacent edges and sequentially inserts two new edges in order to identify a different Hamilton cycle. The 2-opt searches the neighbourhood of the current solution, where there are  $O(N^k)$  possibilities of selecting two links to be replaced in order to construct a new cycle, where  $k$  is the number of edges being replaced and  $N$  is the number of edges in the cycle (Potvin and Rousseau, 1995). The cycle with the greatest improvement replaces the incumbent. This procedure is repeated, until an improved cycle can no longer be found, in which case a 2-optimal is said to be found. The 2-opt operator is applied in the identical manner in this research. Figures 5.4(a,b) illustrate the application of 2-opt operator and Figure 5.5 describes the pseudo code for the operator.

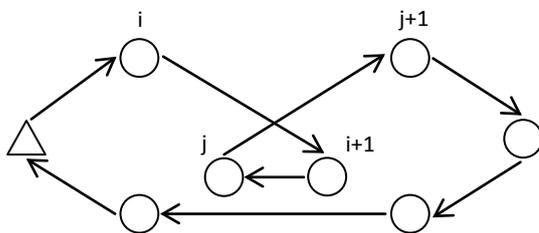


Figure 5.4a

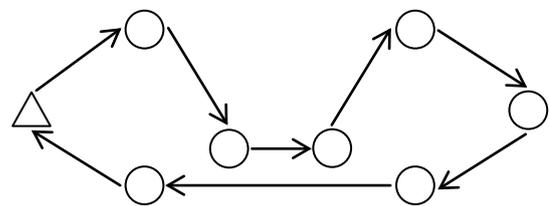


Figure 5.4b

Figure 5.4a illustrates the current cycle, where the edges  $(i, i+1)$  and  $(j, j+1)$  were deleted and the edges  $(i, j)$  and  $(i+1, j+1)$  were inserted in order to construct the neighbouring solution in Figure 5.4b.

```

1  2-opt( $G=[e_1, \dots, e_{N-1}]$ ) returns  $G$ 
2   $G_{current}, G'$ 
3
4  do
5      Total Length (TL) = Length ( $G$ )
6       $G_{current}=G$ 
7      for  $i=1:N-1$ 
8           $G' = G$ 
9           $G' = G' \setminus \{ (i, i+1), (j, j+1) \}$ , where  $j \neq i+1$ 
10          $G' = G' \cup \{ (i, j), (i+1, j+1) \}$ 
11         if  $TL(G') < TL(G_{current})$ 
12              $G_{current}=G'$ 
13         end
14     end
15
16     if  $TL(G_{current}) < TL(G)$ 
17          $G=G_{current}$ 
18     else
19         break
20     end
21 end

```

Figure 5.5 Pseudo Code for 2-opt operator. Notations:  $G$  - graph,  $G_{current}$  - incumbent graph;  $G'$  - new graph; TL - total length.

The graph ( $G$ ) comprises of edges  $e_1, \dots, e_{N-1}$ , which define a Hamilton cycle of a total length  $TL(G)$ . The 2-opt operator searches the neighbourhood of the  $G$  by removing edges  $(i, i+1)$  and  $(j, j+1)$  and replacing them with edges  $(i, j)$  and  $(i+1, j+1)$  (lines 9 and 10). The new graph  $G'$  will replace the best graph found so far ( $G_{current}$ ), if  $TL(G')$  is lower than  $TL(G_{current})$  (lines 11 and 12). Once the neighbourhoods of  $G$  have been explored,  $G_{current}$  will replace  $G$ , if  $TL(G_{current})$  is lower than  $TL(G)$  (lines 16 and 17). The aforementioned process is repeated with the new graph  $G$ , until no further improvement can be made.

The 2-opt operator is not well adapted to problems with time windows because the orientation of the route cannot be preserved. However, the reversal of a proportion of the cycle may be advantageous in introducing route capacity feasibility, as shown in Figure 5.4(a,b), where the cycle orientation between  $i+1$  and  $j$  was reversed. Therefore, it is useful for TSPSDP problems, and may be beneficial to TSPSDPTW problems with limited number of time window restrictions.

#### 5.4.2 Or-opt operator

The Or-opt operator was proposed by Or (1976), which has proven to produce high quality solutions (Potvin and Rousseau, 1995), whilst being computational resource efficient (Toth and Vigo, 2002). This operator removes a maximum of three adjacent nodes from the cycle and reinserts them in the same sequence between two adjacent nodes in the cycle. This operator explores the removal of all three, two and one adjacent nodes and their insertion in all possible points in the cycle. The cycle with the greatest improvement replaces the incumbent and the procedure is repeated, until no further improvement is possible. The Or-opt operator maintains the orientation of the relocated nodes and the cycle, therefore it is a powerful approach for a problem containing time windows (Potvin et al., 1996). The Figures 5.6(a,b) illustrates the application of Or-opt operator and Figure 5.7 describes the respective Pseudo code.

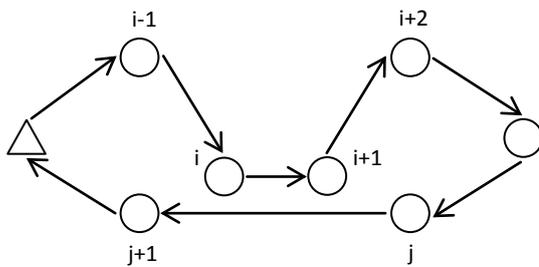


Figure 5.6a

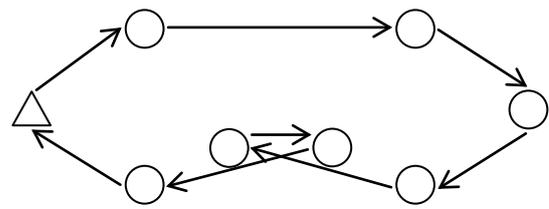


Figure 5.6b

Figure 5.6a the adjacent clients  $i$  and  $i+1$  were re-inserted in another position in Figure 5.6b.

```

1 Or-opt(G=[e1, ..., eN-1]) returns GFinal
2 G1, G2, G3, GFinal, GCurrent, G'
3
4 GFinal=G
5 TL(GFinal)=∞
6
7 for k=1:∞
8     opt 1
9     G3=G
10    for i=1:N-5
11        for j=i+3:N-2
12            G'=G
13            G' = G' \ { (i-1, i), (i+2, i+3), (j, j+1) }
14            G' = G' ∪ { (i+2, j), (i, j+1), (i-1, i+3) }
15            if TL(G')<TL(G3)
16                G3=G'
17            end
18        end
19    end
20    opt 2
21    G2=G
22    for i=1:N-4
23        for j=i+1:N-2
24            G'=G
25            G' = G' \ { (i-1, i), (i+1, i+2), (j, j+1) }
26            G' = G' ∪ { (i+1, j), (i, j+1), (i-1, i+2) }
27            if TL(G')<TL(G2)
28                G2=G'
29            end
30        end
31    end
32    opt 3
33    G1=G
34    for i=1:N-3
35        for j=i+1:N-2
36            G'=G
37            G' = G' \ { (i-1, i), (i, i+1), (j, j+1) }
38            G' = G' ∪ { (i, j), (i, j+1), (i-1, i+1) }
39            if TL(G')<TL(G1)
40                G1=G'
41            end
42        end
43    end
44
45    Gcurrent=min(TL(G1), TL(G2), TL(G3))
46
47    if TL(Gcurrent) < TL(GFinal)
48        GFinal=Gcurrent
49        G=Gcurrent
50    else
51        break
52    end
53
54 end

```

Figure 5.7 Or-opt pseudo code. Notations:  $G$  - graph including edges  $e_1, \dots, e_{N-1}$ ;  $G_{Final}$  - best stored graph in simulation;  $G_{Current}$  - best graph determined in generation;  $G'$  - new graph;  $G_1$  -

best graph with one node sequence,  $G_2$  - best graph with two consecutive node sequence,  $G_3$  - best graph with three consecutive node sequence; TL - total length.

The initial graph ( $G$ ) consists of a set of edges  $e_1, \dots, e_{N-1}$ , which arbitrarily connect  $N$  nodes. In Opt 1, three consecutive nodes are relocated to a new position in  $G$ , thus creating a new graph  $G'$  (lines 13 and 14). The  $G'$  with the lowest total length (TL) is saved as the best known solution  $G_3$  (lines 15 and 16). All insertion positions are explored for a set of consecutive nodes (line 11) and the process is repeated for all consecutive combinations (line 10). An identical process follows for Opt 2 and Opt 3, with the exception that two and one consecutive nodes are respectively considered for relocation to a new position in  $G$  (lines 25 and 26), (lines 37 and 38). The minimum length graph  $G_3$ ,  $G_2$  or  $G_1$  will replace the smallest length graph found so far  $G_{Final}$ , if it has been minimised (lines 47 and 48). The aforementioned process will continue until no further improvement is possible.

## PART 2 - SDPmethod

In this section a three phase procedure called SDPmethod is introduced to solve the VRPSDP. Phase 1 determines a diverse range of partial solutions, which are hypothesised to build near optimal complete solutions. Phase 2 defines the assignment possibilities of the remaining unassigned requests to the established routes based on spatial proximity. Finally, Phase 3 attempts to find the near-optimal solutions by exploring the assignment possibilities defined in Phase 2 using a Parallel GA Model. Figure 5.8 illustrates the components of the SDPmethod.

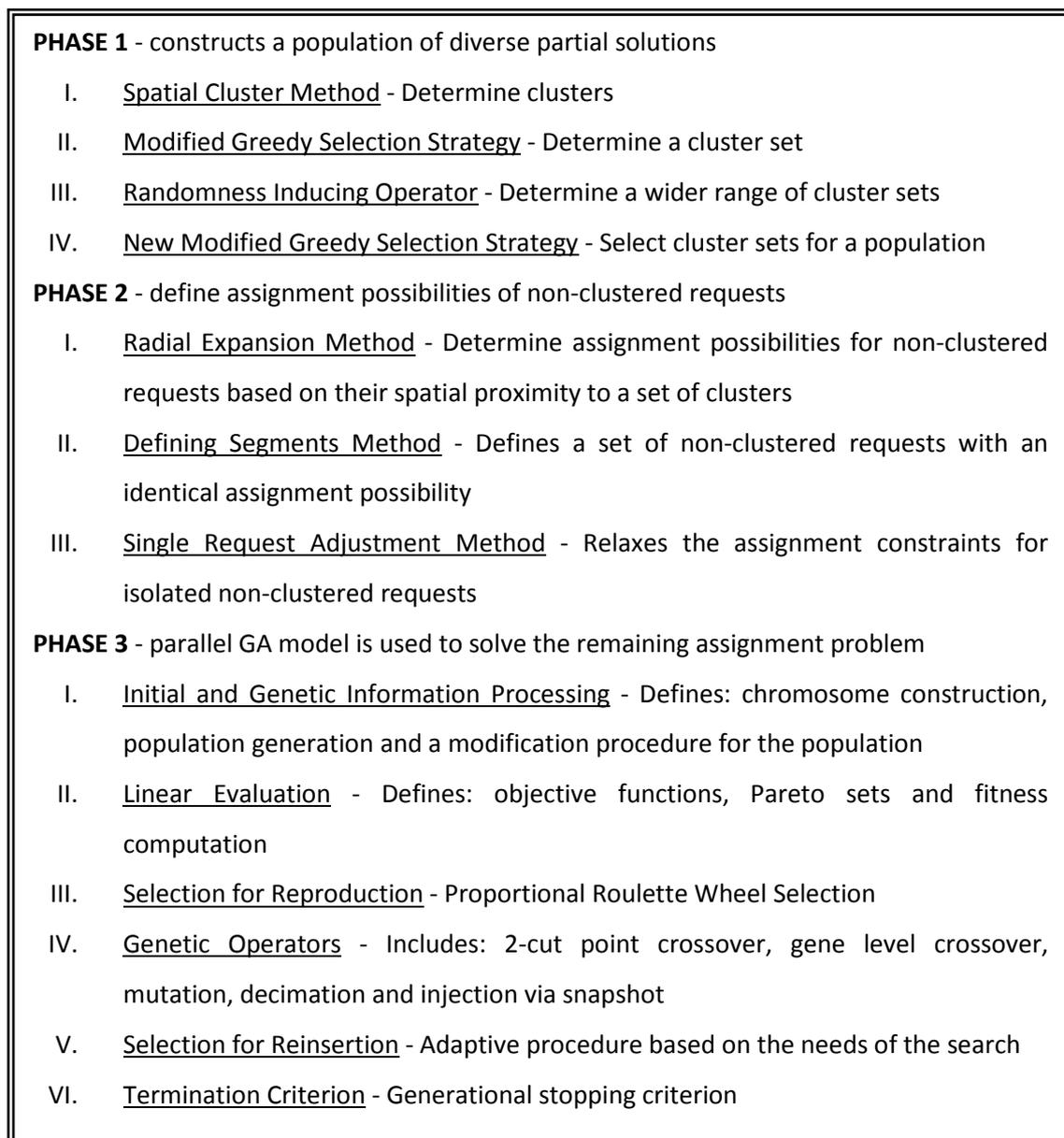


Figure 5.8 Components of the SDPmethod

## 5.5 Phase 1 (Partial solution construction)

The purpose of Phase 1 is to construct a population of diverse partial solutions for the VRPSDP. It is hypothesised that the partial solutions will cover areas of the search space likely to generate near-optimal solutions. This phase consists of a family of procedures: Spatial Clustering Method (SCM), Modified Greedy Selection Strategy (MGSS), Random Inducing Operator (RIO) and New Modified Greedy Selection Strategy (NEWMGSS), which are sequentially applied. The role of SCM is to define various feasible vehicle capacity and route time assignments, termed clusters. MGSS is employed to determine a potential combination of clusters for the VRPSDP. Following the success of the aforementioned method, further set combinations are explored using RIO. Finally, NEWMGSS selects a range of diverse cluster sets for exploration. The aforementioned are now discussed in greater detail.

### 5.5.1 Spatial Clustering Method (SCM)

SCM is a clustering method that can be applied over any data set in order to group certain data samples together, provided that they share given common features. The reader is referred to Xu and Wunsch (2009) for extensive information on clustering techniques. The purpose of SCM is to define a set of clusters, where each one represents a feasible route assignment in terms of VRPSDP problem context. The method consists of two sequential steps: determine capacity and then route time feasible clusters. The former step maximises the capacity utilisation of each cluster, whereas, the latter step aims to determine the minimum route service time of each cluster assignment.

The SCM clusters requests based on the spatial proximity to a reference point, termed seed. A seed refers to the geographical location of a request represented as a set of latitude and longitude coordinates. Every request is represented as a point in a 2-dimensional coordinate space. A circular vicinity of a given radius is considered for each seed, thus forming a set of clusters. The purpose of the procedure is to maximise the requests situated inside each cluster, whilst complying with a set of uniform volume and weight constraints. These constraints equate to the capacity defined in terms of volume and weight of a given vehicle from a homogenous fleet used to service the region. This way all clusters are guaranteed to represent a feasible vehicle load. However, the capacity constraints may be reduced at later stages of the SDPmethod, as explained in MGSS section. Figure 5.9 illustrates the cluster approach adopted in pseudo code.

```

1  SCM(P=[p1, ..., pN], V=[v1, ..., vN], W=[w1, ..., wN], Volmax, Wgtmax, R) returns pi_FeasibleList
2  Vol=0, Wgt=0
3  pi_list=[]
4
5  for i=1,N
6      r=R, Vol=0, Wgt=0
7      pi_list=[]
8      for j=1,N
9          if dist(pj,pi)≤r
10             Vol = Vol + vj
11             Wgt = Wgt + wj
12             pi_list = [pi_list, pj]
13             if Vol > Volmax OR Wgt > Wgtmax
14                 break
15             else
16                 r= r + R
17                 pi_FeasibleList = pi_list
18             end
19         end
20     end
21 end

```

Figure 5.9 SCM pseudo code. Notations: P - set of nodes; V - node volume vector; W - node weight vector; Vol<sub>max</sub> - Maximum volume of cluster; Wgt<sub>max</sub> - Maximum weight of cluster; p<sub>i</sub>\_FeasibleList - feasible node list within certain distance of p<sub>i</sub>; R - preset distance; p<sub>i</sub>\_list – node list within certain distance of p<sub>i</sub>.

The vector (P) defines a set of nodes situated on a plane, where vectors (V) and (W) respectively state their volume and weight. A cluster (p<sub>i</sub>\_List) is initialised by node p<sub>i</sub> and nodes n<sub>j</sub> are included in the cluster in increasing order of distance to p<sub>i</sub> (lines 9 to 12). The expansion of the cluster is controlled by radius (r), which is iteratively increased by a preset distance (R). The R parameter value has been selected after a series of trial and error experiments. R is set to 5% of the distance between two of the furthest nodes in the plane, which provides a convenient tradeoff between exploration and computational resource efficiency. The cluster expansion phase will cease once no further insertions are possible without either violating the cluster total volume (Vol<sub>i</sub>) or total weight (Wgt<sub>i</sub>) (lines 13 to 14), thus forming a capacity feasible cluster (p<sub>i</sub>\_FeasibleList). The aforementioned process is repeated to construct a capacity feasible cluster for every node p<sub>i</sub>.

A circular vicinity is used to cluster in order to encourage compact assignments. The most recognised measure of compactness for a shape is the 'perimeter to area ratio' (Maceachren, 1985), and is based on the compactness of the outer boundary. A circular geographical

coverage has the shortest perimeter for a given area (Angel, 2010). It is hypothesised that clustering based on a circular vicinity will maximise the density of requests assigned in a given area. Although, clustering requests based on spatial distances from the seed does not guarantee the requests are closely clustered. The measure of compactness based on dispersion of elements within an area considers the unit as a whole and therefore provides the most accurate measure of compactness (Maceachren, 1985). For example, Figures 5.10(a,b) illustrate two circular areas with an identical perimeter to area ratio, but the dispersion of elements between the three nodes in Figure 5.10b is lower than that of Figure 5.10a.

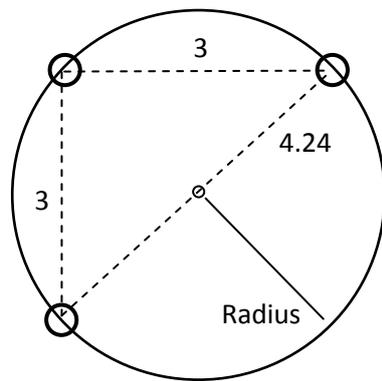


Figure 5.10a

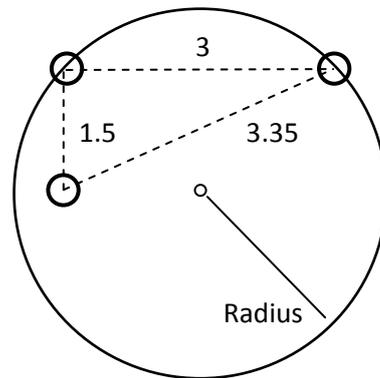


Figure 5.10b

Since, the dispersion of elements within a given area is a more precise measure of compactness the route distance for each capacity feasible cluster is calculated using the RouteAlg. This measurement determines the distance taken by a driver to service all the requests inside a cluster, including deadheading to and from the depot, as illustrated in Figure 5.11 for a four node problem. In the presence of a route distance constraint, the cluster assignments with routing distances within the constraint bounds are deemed service feasible. The capacity and route distance feasible clusters are referred to as clusters and are used to construct cluster sets in MGSS.

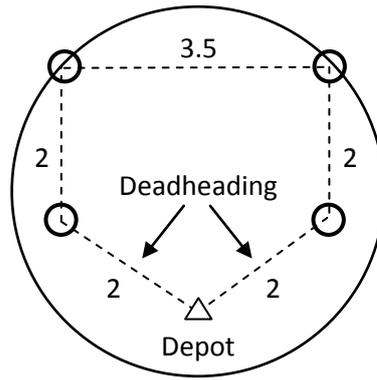


Figure 5.11

### 5.5.2 Modified Greedy Selection Strategy (MGSS)

MGSS is a selection mechanism, which is applied after the termination of SCM. MGSS makes an order dependant choice at each step with the purpose of discovering the global optimum. The purpose of MGSS is to combine a set of clusters, which respectively define a minimum number of routes required to service the demand. A greedy method is used to generate at least one cluster set, even if it is sub-optimal.

The MGSS aims to maximise a compactness function in Equation 5.2 and therefore selects clusters to form part of the set in descending order of compactness. The compactness metric is defined as the average routing distance between requests inside a cluster, including deadheading. The cluster with the highest compactness measure on average has the smallest distance between servicing requests compared to all other routes. The maximisation of compactness allows for an increase in the level of slack available for non-cluster requests assignment. However, this does not guarantee a feasible or high quality final solution, although, it does increase the probability of finding such a solution.

$$Compactness (cluster) = \frac{D_c}{N_c} \quad (5.2)$$

where,  $D_c$  defines the minimum routing distance for the cluster nodes and  $N_c$  is the total number of cluster nodes.

The consideration of deadheading during compactness computation may distort the compactness level of the cluster. Those clusters spatially closest to the depot will inherently have a greater compactness than those situated further away. However, the exclusion of

deadheading may result in route unfeasibility at a later stage. Furthermore, the sequence of service for a cluster may change as deadheading is considered, therefore the compactness measurement may no longer hold true.

Initially, a random cluster is selected to form part of the set. The remaining clusters are sequentially selected in descending order of their level of compactness. The duplicate assignment of requests between clusters is not permitted because a request is only assignable to a single route to meet the requirements of the VRPSDP. Therefore, the clusters that share any part of their assignment with the set are prohibited from the selection process. Where two or more clusters share the same level of compactness, a cluster is randomly selected into the set. The sequential selection process terminates once the number of clusters becomes equal to the number of drivers required to service the demand. To increase the robustness of the greedy approach, the above procedure is repeated until every cluster has been selected as the first in the set. This is likely to increase the probability of finding feasible cluster set(s). Figure 5.12 outlines the MGSS pseudo code.

```

1  MGSS(C=[c1, ..., cN]) returns Gi
2
3  for i=1:N
4      Gi=[ci]
5      D=[c1, ..., cN] \ [ci]
6      arrange D in descending order so CL1 ≤ CL2 ≤, ..., CLN
7      count=1
8
9      for j=1:M
10         if (dj∩Gi)=∅
11             add cluster Gi=[Gi∪dj]
12             count=count+1
13         end
14
15         if count=countMAX, where countMAX
16             break
17         end
18     end
19
20 end

```

Figure 5.12 MGSS pseudo code. Notations: C - vector containing all possible cluster assignments; G - cluster set; D - vector contains all possible cluster assignments, excluding the first cluster already assigned to the set; CL - cluster total compactness measure; count<sub>MAX</sub> - maximum size of the cluster set.

The vector (C) defines a set of vehicle load capacity feasible clusters, where cluster  $c_i$  is used to initialise a cluster set ( $G_i$ ) (line 4). A cluster  $d_j$  from vector (D) is considered for assignment to  $G_i$  in descending order of the compactness measure, where D is a copy of C excluding  $c_i$ . However, the assignment will only take place if the nodes defining  $d_j$  have not previously been assigned to another cluster in  $G_i$  (lines 10 and 11). The assignment of clusters from D to  $G_i$  will continue until the maximum cluster set size ( $\text{count}_{\text{MAX}}$ ) is reached (lines 15 and 16). The above process is repeated N times by initialising a cluster set with a different cluster in C.

It is highly probable that a cluster set is unlikely to be determined using MGSS, if the clusters capacity constraints are set too high in SCM. This issue can be overcome by reducing the uniform cluster capacity constraints and restarting from SCM. This allows for the construction of smaller cluster assignments, therefore reducing the probability of assignment duplication between clusters. An inverse relationship exists between the cluster capacity constraints and the number of cluster sets that can be determined using MGSS. The incremental reduction of cluster capacity in SCM will increase the number of possible unique sets, therefore improving the probability of MGSS in determining feasible cluster set(s). However, a major consequence of reducing the cluster capacity is it results in a greater number of non-cluster requests, those that have not been allocated to any cluster, which will need to be assigned in the genetic algorithm. This would be contrary to the purpose of SCM, which aims to maximise assignment to the clusters, therefore greater effort can be employed on assigning the remaining requests. Therefore, to avoid repeated unfeasibility, the starting cluster capacity constraint can be set lower than the actual vehicle capacity. To ensure the progress towards a cluster set in MGSS, the cluster capacity constraints are incrementally reduced by 5%, until a set is defined. The 5% reduction ratio is a pre-set algorithm parameter, which has been determined via a trial and error line of experiments.

### 5.5.3 Randomness Inducing Operator (RIO)

RIO is identical to the gene level mutation operator used by evolutionary algorithms for search space exploration. The term RIO is used instead to avoid confusion with the mutation operator, which is implemented in Phase 3. The purpose of this operator is to discover additional cluster sets, to those found by MGSS. The adoption of RIO is necessary because unlike MGSS, it is capable of exploring a wider range cluster set combinations.

The RIO is applied to a population of cluster sets, where each set defines a number of driver routes required to form a solution. The size of the population is equivalent to the number of populations in the Parallel GA model. The population comprises of a number of randomly selected cluster sets found in MGSS. In case of a population deficit caused by an insufficient number of cluster sets in MGSS, the remaining cluster sets are randomly generated, which will contribute to greater diversity in the population and will assist with the exploration.

RIO is asexual and is applied to every cluster set inside the population. The intensity of this operator is controlled using a probability threshold  $P_T$ , which is set to 50% to provide a degree of random search. A random probability is assigned to every cluster and those that fall under the  $P_T$  threshold are replaced by a randomly selected cluster. The newly formed sets are evaluated in terms of uniqueness and are stored. The population is added to a tabu list comprising of previously determined populations, therefore avoiding the repeated evaluation of sets. This procedure is reiterated for 10000 generations. Figure 5.13 illustrates the pseudo code.

```

1  RIO(CS=[csi,j], where i=1, ..., N, j=1, ..., M, CList=[c1, ..., cQ]) returns CS
2  DummyCS
3  PT
4
5  for i=1,P
6      for j=1,N
7          DummyCS=CSj
8          for k=1,M
9              if rand(1)<PT
10                 replace DummyCSk with randomly picked cq ∉ DummyCS
11             end
12         end
13         if count(DummyCS)==count(unique(DummyCS))
14             CS=[CS;DummyCS]
15         end
16     end
17 end

```

Figure 5.13 RIO pseudo code. Notations: CS - population of feasible cluster sets; CList - list of feasible clusters; DummyCS - copy of a cluster set in the population at generation  $j$ ;  $P_T$  - probability threshold.

A cluster set (DummyCS) is iteratively selected for modification from vector (CS). A random selection probability is assigned to a cluster (DummyCS<sub>k</sub>) and if below the probability threshold  $P_T$ , then it is replaced with an arbitrary cluster  $c_q$  from vector CList, where  $c_q$  cannot be

identical to any cluster in DummyCS (lines 9 and 10). Once all clusters have been considered for modification and there is no duplication of nodes between them, DummyCS is added to CS (lines 13 and 14). The aforementioned is repeated P times to increase the number of cluster sets in CS.

This operator may have been used in place of MGSS to find a cluster set. Instead, it is selected as subsequent procedure to prevent wasting computational resources, if no feasible cluster set exists with those clusters defined in SCM.

#### 5.5.4 New Modified Greedy Selection Strategy (NEWMGSS)

NEWMGSS is a selection scheme similar to MGSS. The purpose of this procedure is to determine a population of cluster sets for further assignment in the Parallel GA Model. It is desirable for such a population to converge to the near-optimal solution. The selection scheme selects cluster sets with the highest compactness, whilst ensuring sufficient diversity remains inside the population. The population diversity is important because cluster sets may share identical or very similar assignments to one another. The selection of similar cluster sets for the population is undesirable because it results in the repeated exploration of the same search space area. It is argued that a compact and diverse population type will increase the robustness of the search. Figure 5.14 illustrates the pseudo code.

```

1  NEWMGSS(S=[s1, ..., sY], L=[l1, ..., lY], IdenMax, PopSizeMax) returns BestPop
2  Pop, DumS, DumL, IdenElem, ElemMax, IdenRatio
3  PopSize, Pass, PopL, BestPopL
4
5  BestPop=[]
6  BestPopL=∞
7
8  for g=1:Y
9      Pop=Sg, PopL=Lg
10     DumS = S \ Sg, DumL = L \ Lg
11     DumL,DumS=sort(DumL,DumS)
12     PopSize=0
13
14     for h=1:Y-1
15         Pass=0
16
17         for i=1:M
18             IdenRatio=[]
19
20             for j=1:N
21                 IdenElem=[], ElemMax=[]
22
23                 for k=1:N
24                     IdenElem=[IdenElem, count(Popij∩DumSh,k)]
25                     ElemMax=[ElemMax, max(count(Popij) OR count(DumSh,k))]
26                 end
27
28                 IdenRatio=[IdenRatio, max(IdenElem)/ElemMax(max(IdenElem))]
29
30             end
31
32             if sum(IdenRatio)/N<=IdenMax
33                 Pass=Pass+1
34             else
35                 break
36             end
37         end
38
39         if Pass=M
40             Pop=[Pop∪DumSh,k]
41             PopL=PopL+DumLh
42             PopSize=PopSize+1
43         end
44
45         if PopSize=PopSizeMax
46             if PopL<BestPopL
47                 BestPop=Pop
48             end
49             break
50         end
51     end
52 end
53
54 end

```

Figure 5.14 NEWMGSS pseudo code. Notations: \*S - vector with previously established cluster sets by MGSS and RIO; L - compactness measure of each cluster set in S; BestPop - a list of

diverse cluster sets selected for the population; Pop - a population of cluster sets determined in a generation; PopL – compactness measure of Pop in a generation; IdenMax - maximum permitted similarity between two cluster sets inside a Pop; DumS - vector S excluding  $S_g$ ; DumL - vector L excluding  $L_g$ ; IdenElem - vector that records the number of identical elements between two clusters; ElemMax - vector that records the largest count of identical nodes between two clusters from different sets; IdenRatio - average similarity of a cluster set DumS[h] and the Pop; PopSizeMax - maximum size of Pop; PopSize –population size count; Pass - diversity validation metric.

Vector (S) defines the cluster sets constructed in MGSS and RIO. The population (Pop) is initialised with the inclusion of cluster set  $S_g$  (line 9). Iteratively, a cluster set DumS<sub>h</sub> from vector DumS is considered for insertion into Pop in decreasing order of their compactness measure (Equation 5.3), where DumS is a copy of S excluding  $S_g$ .

$$Compactness = \frac{\sum_{i=1}^m D_{ci}}{\sum_{i=1}^n N_{ci}} \quad (5.3)$$

where  $D_{ci}$  defines the minimum routing distance in a cluster,  $N_{ci}$  is the number of cluster nodes and  $m$  is the number of clusters in a set.

DumS<sub>h</sub> must introduce diversity into Pop in order to be included. A similarity metric (IdenRatio) is used to measure the proportion of requests that are identical between a pair of cluster sets (lines 23 to 28). DumS<sub>h</sub> is only eligible for insertion, if IdenRatio is within the maximum similarity threshold (IdenMax) for all comparisons between DumS<sub>h</sub> and the cluster sets in Pop (lines 32 to 40). IdenMax is set to 50% because it was demonstrated over a number of runs to provide a satisfaction tradeoff between exploration and computational resource consumption. The assignment of cluster sets from DumS to Pop will continue until the maximum Pop size (PopSizeMax) is reached (lines 45 and 49). The above process is repeated Y times by initialising a new Pop with a different cluster set in S (lines 8 and 9). The Pop with the greatest compactness measure is stored and selected for further investigation in Phase 2 and 3 of the SDPmethod (lines 46 and 47). However, if a diverse population cannot be found, Phase 1 is repeated with lower cluster capacity constraints.

## 5.6 Phase 2 (Assignment possibilities)

Following the construction of a partial solution in Phase 1, the purpose of this phase is to reduce the complexity of the remaining assignment challenge. This problem is computational complex because a significantly large number of assignment possibilities exist. Phase 2 is used to define assignment possibilities of non-clustered requests to clusters determined in Phase 1. This phase consists of three sequential procedures: Radial Expansion Method (REM), Defining Segment Method (DSM) and Single Request Adjustment Method (SRAM). The REM method is used to determine assignment possibilities based on radial proximity of non-cluster requests to the clusters. This method derives from an assumption that the assignment probability increases inversely with the Euclidean distance between the non-cluster request and clusters. The requests with the same assignment possibilities are combined to form a segment in DSM. Finally, the segments with a single assignment possibility restriction are relaxed using SRAM. The aforementioned methods are adopted because they are simple and computationally inexpensive.

### 5.6.1 Radial Expansion Method (REM)

The purpose of REM is to determine assignment possibilities for non-clustered requests based on their relative distance to the set of clusters. It is alleged that such assignment possibilities will define fruitful search spaces for exploration, as shown in the pseudo code in Figure 5.15.

```
1  REM(G=[P=[p1, ..., pN], S=[s1, ..., sM], R=[r1, ..., rM], c) returns Q
2  Q=[], T=[]
3
4  while (count(P) ~=count(T))
5
6      for i=1:M
7
8          Ri=Ri+c
9          Qi=[Qi,find(P(dist(P,Si)≤Ri))]
10         Qi=unique(Qi)
11         T=[T,Qi]
12         T=unique(T)
13
14     end
15
16 end
```

Figure 5.15 REM pseudo code. Notations: P - unassigned nodes; S - cluster seeds; R - vector of cluster radiuses; c - fixed distance, Q<sub>i</sub> - vector that defines the number of nodes within a particular distance of the cluster seed, T - termination validation metric.

The cluster radius  $R_i$  is increased by a fixed length ( $c$ ) from cluster seed  $S_i$  and the unassigned nodes ( $P$ ) that are within this distance are added to vector  $Q_i$  (lines 8 and 9), which defines a group of nodes in the spatial proximity of the cluster represented by  $S_i$ . The above process is repeated for all cluster seeds in  $S$  (lines 6 to 14), until the number of nodes in vector ( $T$ ) is equal to the number of nodes in  $P$  (line 4).

The circular vicinities used to determine the set of clusters in SCM are expanded from their seeds. The process of incremental expansion is carried out in parallel, until all requests in the region are overlapped, as shown in Figure 5.16. A circular vicinity represents a cluster in an expanded form. The assignment of non-clustered requests is limited to those cluster(s) where the circular vicinity(s) overlap. REM restricts the assignment of unassigned requests to cluster(s) based on spatial proximity to the cluster seeds. It is assumed that fruitful search spaces will be identified using the approach described. REM assumes a fully connected graph, however where an asymmetric network exists the performance of the approach is hindered. A possible example, which illustrates the previously stated problem, consists in accessibility constraints like geographic obstacles or one-way streets.

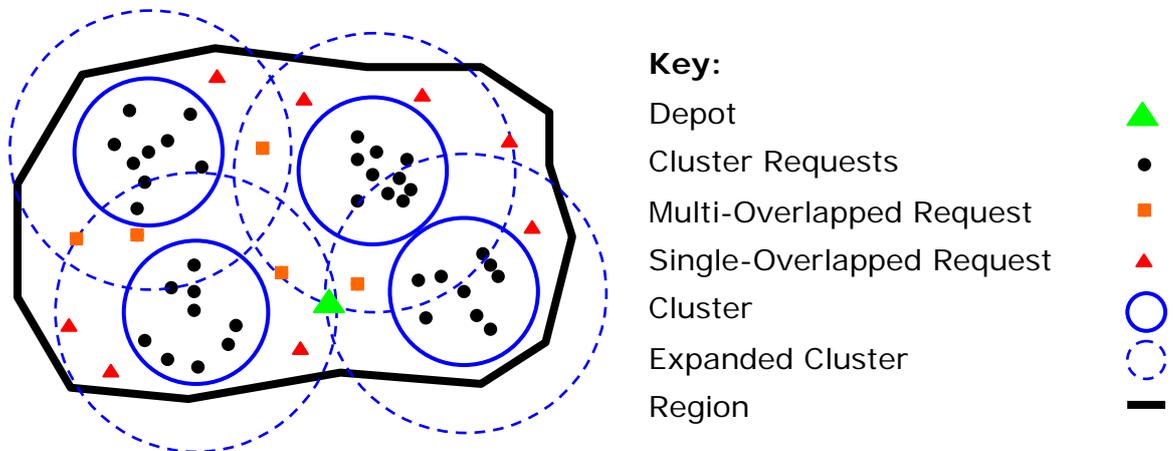


Figure 5.16 Expanded Clusters

### 5.6.2 Defining Segments Method (DSM)

The purpose of DSM is to organise the assignment possibilities determined in REM. The goal is to create segments that can be later used as building blocks (a sub-chromosome that potentially increases the quality of the host individual). A segment is defined as a set of requests situated in the overlapping region of one or more expanded cluster vicinities, as

shown in Figure 5.16. The assignment decisions for the combined segments represent a complete assignment of unassigned requests.

### 5.6.3 Single Request Adjustment Method (SRAM)

The segment(s) containing a single cluster do not require evaluation in terms of assignment. Therefore, these requests are the most isolated from their neighbouring cluster(s) because after the expansion phase they are located inside one vicinity only, as shown in Figure 5.17. When a single request is situated in the expansion area of one cluster, it may only be assigned to that driver. This situation may later impede the generation of a feasible solution given its reduced assignment flexibility. To reduce this risk, the assignment flexibility is increased by considering an additional cluster, namely the one which is spatially closest to the isolated request. The aforementioned cluster is added to the segment.

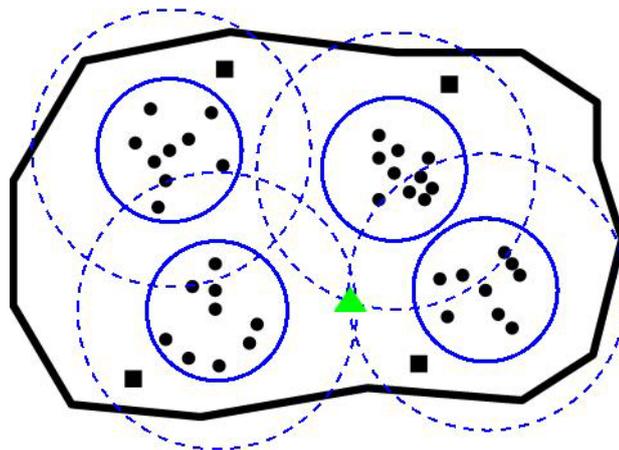


Figure 5.17 Isolated Requests, which are represented by the square nodes

## 5.7 Phase 3 (Complete solution)

The purpose of Phase 3 is to assign the non-clustered requests using a Parallel GA Model, where the same genetic algorithm is run in parallel for each cluster set. These requests are assigned to the clusters constructed in Phase 1 using the assignment possibilities determined in Phase 2. The approach consists of the following steps: encoding, linear evaluation, selection for reproduction, application of genetic operators, selection of reinsertion and termination. These are now discussed in greater detail.

### 5.7.1 Initial and genetic information processing

The employed encoding approach consists of three procedures: chromosome construction, population generation and a modification procedure for the population. The first procedure

defines a chromosome, which encodes a solution. The second procedure builds a population of chromosomes for exploration. The final procedure amends the population to better represent the assignment possibilities defined in Phase 2.

A chromosome is encrypted inside a linear structure using a finite alphabet. In the context of this work, such a linear structure represents an allocation of non-clustered requests. A chromosome comprises of sub-chromosome(s), each defining a potential assignment to a segment constructed in Phase 2. The sub-chromosomes are combined in a fixed sequence to form a chromosome. Each gene value encrypts a cluster and its position within the sub-chromosome represents the request assigned to it. The sub-chromosome genes are attributed values from the same alleles set. Alleles define the legal set of clusters that can be assigned to a gene. The length of a chromosome is equal to the number of unassigned requests in the region and does not vary through the generations. Figure 5.18, depicts how sub-chromosomes are combined to form a chromosome.

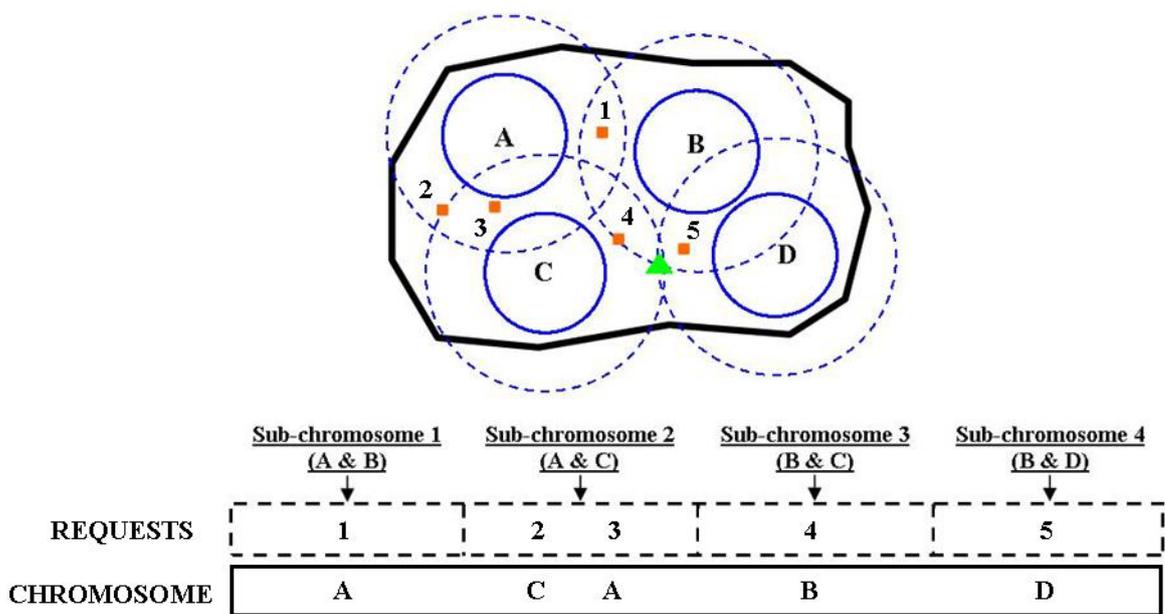


Figure 5.18 Chromosome Construction

The population is built in such a way so that it covers the current search space under investigation. The population consists of 500 individuals because this size provides the best compromise between solution quality and algorithm speed. The population size has been determined experimentally in Chapter 6 by analysing the performance of the SDPmethod for different population sizes over a number of datasets. The population size is fixed to prevent an

incontrollable increase in the number of individuals during the evolution process. The initial population is randomly generated to effectively cover the solution space. All alleles are expected to be present inside the population.

The gene values initially assigned to certain sub-chromosomes are replaced to more accurately reflect the outcome of Phase 2. More specifically, within Phase 2 segments containing only one cluster C1 are assigned an additional cluster C2, in order to increase assignment flexibility (section 5.6.3). The original cluster C1 is considered to be the most likely candidate for optimal assignment, since during the expansion phase no other cluster was spatially closer. This represents the reason for only considering cluster C1 for the initial population. However, this gene value restriction is not enforced during the evolution.

### 5.7.2 Linear Evaluation

This section deals with a threefold procedure consisting of defining: objective functions, Pareto sets and fitness computation. Here follows a more comprehensive description of these procedures.

#### Objective Functions

Objective functions are mathematic instruments used to assess the solution quality. In the context of this research, a chromosome is evaluated with respect to the minimisation of three objective functions: (1) number of unfeasible routes constructed for a predetermined vehicle fleet size, (2) total routing distance, (3) maximum variation between route distances (workload variation). The cost associated with servicing customer demand increases with the distance travelled. Most transportation operators aim to achieve similar workloads for all their routes, thus the variation between route workloads needs to be minimised. The number of unfeasible routes determines the number of unserviceable routes in each solution. In order to identify unfeasible routes, it is necessary to evaluate each route in terms of capacity and route distance feasibility. If a potential solution encrypts only feasible routes, the optimisation problem comprises of two dimensions: total routing distance and workload variation. The workload variation objective is meant to guide the solution towards equitable areas of the search space.

### Pareto Set Separation

The Pareto approach is an intermediary procedure required for fitness computation. The individuals are ranked inside the population in terms of dominance. Deb (2009) Continuously Update Procedure (CUP) is the Pareto approach implemented in this research because the method is computationally inexpensive compared to 'Naive and Slow'. The entire population is ranked because all individuals are considered for reproduction. This strategy will result in greater population diversity, which is required to progress the search. Figure 5.19 outlines the pseudo code for CUP.

```
1  ParetoSetSeparation (S=[s1,s2, ..., sN]) returns ND(count)
2  REM
3  add, count
4  count=1
5
6  while S≠∅
7
8      ND(count)=S1
9
10     for i=2:N
11         add=0
12         REM=[]
13         for j=1:ND(count)
14             if Si<ND(count)j
15                 add=1
16                 REM=[REM, ND(count)j]
17             end
18             if Si=ND(count)j
19                 add=1
20             end
21         end
22         if add=1
23             ND(count)=ND(count)∪Si
24             ND(count)=ND(count)\Sj
25         end
26     end
27
28     count=count+1
29     S=S\ND(count)
30
31 end
```

Figure 5.19 Pareto Set Separation pseudo code. Notations: S - population: ND(count) - nondominated set, which decreases in importance in ascending order of rank, REM - vector with dominated individuals.

The 1<sup>st</sup> order Pareto set (ND1) is initialised with individual  $S_1$  from the population (S) (line 8). For each individual in S a dominance test is performed with incumbent individuals in ND1 (lines 10 to 26). In consequence, the dominated solutions are removed and the nondominated solutions are inserted into ND1 (lines 23 and 24), which results in the discovery of a nondominated set. This is repeated with the exclusion of ND1 individuals, in order to discover the next order Pareto set ND(count). The process terminates once S is empty (line 6).

A Pareto approach is adopted because the problem under consideration consists of multiple objectives that are conflicting, and the weighting information is unknown. To prevent the construction of an undesired solution, objective weights assumptions are avoided, therefore aggregation methods are non-applicable. A major benefit of a Pareto approach is that it can generate a range of solutions, each representing a different objectives trade-off, thus reflecting various user needs. A non-Pareto approach is not adopted because multiple objectives are not simultaneously considered, therefore the method is deemed slower than the one selected, see Chapter 4.

#### Fitness Computation

The dominance information is used to compute fitness values. These values are a relative measure computed by taking into account information with respect to all individuals inside the same population. The individuals with the higher fitness values have the greatest probability of survival. Fitness values are calculated using the following threefold procedure: Deb (2001) with Patelli (2011) amendment fitness formula, Estimated Difference (ED) and New Fitness Formula (NFF). The former method is used to calculate preliminary fitness values. The second method calculates the average fitness values between different fronts. This information is used in the latter method to recalculate fitness values that are truly representative for selection. The following describes the fitness computational method.

- *Deb (2001) Fitness Formula with Patelli (2011) Amendment*

Deb (2001) fitness formula with Patelli (2011) amendment (Chapter 4) is used to calculate preliminary fitness values for the individuals in the population. The fitness values are computed with respect to the order of the fronts, therefore individuals in  $PF_i$  are assigned a higher fitness than any individual on  $PF_{i+1}$ . In addition, the formula distinguishes between individuals on the same Pareto front in the following manner. An individual on a front situated in a weakly explored space is assigned a greater fitness value compared to an individual in an

extensively explored space. Therefore, this method promotes the exploration of new search spaces and discourages that of the same search space.

- *Estimated Difference (ED)*

The Estimated Difference (ED) is a coefficient that estimates the average fitness difference between a Pareto front(i) and the origin of the search space axes. The purpose of this metric is to gain an understanding of the relative quality of solutions situated on different fronts. Equations 5.4, 5.5 and 5.6 are used to compute ED.

$$ND = [S_j, j = 1, M] \quad (5.4)$$

where  $ND$  defines the Pareto set with individuals  $S_j$  and  $M$  refers to the number of individuals in the set.

$$d_j = \sqrt{(S_j^1)^2 + (S_j^2)^2 + (S_j^3)^2} \quad (5.5)$$

where  $d_j$  outlines the distance in the objective space between individual  $S_j$  and the origin of the axes. The upper indices represent the objective being evaluated.

$$d_{avg} = \frac{1}{M} \sum_{j=1}^M d_j \quad (5.6)$$

where  $d_{avg}$  denotes the average distance in the objective space between  $ND$  and the origin of the axes.

- *Newly Modified Deb (2001) Fitness Formula (NFF)*

This method considers the relative distance between different Pareto fronts when computing fitness values. The ED is incorporated into Deb (2001) fitness formula and the fitness values are recalculated. Equation 5.7 defines the fitness formula with the new amendment. As opposed to Deb (2001) and Patelli (2011), this approach additionally considers ED in a request to increase the fitness values accuracy.

$$Fit(c) = \frac{F_R}{\sum_{\substack{z \in PF_i \\ d(c,z) \leq \sigma}} \left[1 - \frac{d(c,z)}{\sigma}\right] \cdot ED} \quad (5.7)$$

The proposed fitness method requires fitness values to be calculated in two stages. Initially, fitness values are calculated for the purpose of determining the average distance between Pareto fronts and their origin. This information is used to refine the fitness values, i.e. calculate more accurate fitness values. This research experimentally supports in Chapter 6 the fact that the increase in accuracy will outweigh the extra computation expense.

### 5.7.3 Selection for Reproduction

The selection method takes into account a factor called selection pressure, which tends to increase in favour of fitter individuals, in order to encourage accuracy in the search. The Proportional Roulette Wheel Selection (PRWS) is adopted because it directly tunes the selection pressure whilst computing fitness values. Therefore, the area of the roulette wheel portion awarded to individuals is proportional to their fitness, with fitter individuals being assigned greater disk space, as illustrated in Figure 5.20.

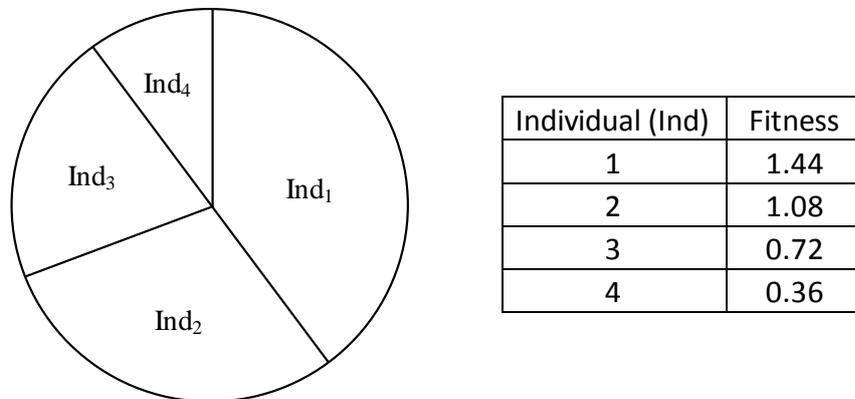


Figure 5.20 PRWS

A shortcoming of PRWS is that, in case certain individuals are significantly fitter than their peers, they tend to get over selected, which may lead to diversity loss. This is due to the fact that the distance among two individuals is proportional to the difference between their fitness values. The Ranked Roulette Wheel Selection is less prone to diversity loss compared to PRWS because the proportion of disk awarded to an individual is in relation to their rank. However, this selection approach has not been adopted because it provides less selection pressure than

PRWS resulting in worse accuracy. Instead, genetic operators are used to reintroduce diversity into the population, as described in the next section.

#### 5.7.4 Genetic Operators

The role of genetic operators is to produce offspring, which will desirably guide the evolution towards optimality. The following genetic operators are applied: 2-cut point crossover, gene level crossover, mutation, decimation and snapshot. The genetic operators are described in greater detail and their implementation via a switching mechanism is discussed in relation to the evolutionary process. Finally, a metric used to monitor the level of accuracy in the search is described.

##### 2-cut point crossover

The 2-cut point crossover randomly selects two unique homologous indices in the layout of the two parents and in turn exchanges the genetic material of the latter considering the previously selected cut-off points, as illustrated in Figure 5.21. A homologous cut point selection scheme is used in order to maintain the chromosome structure, which encompasses a solution. Therefore, a heterogeneous cut point selection scheme is not considered because the resultant offspring will represent incomplete solutions. As each sub-chromosome (SC) may only assume gene values from a specific alleles set, selecting cut points at different locations within the parents may lead to illegal assignments.

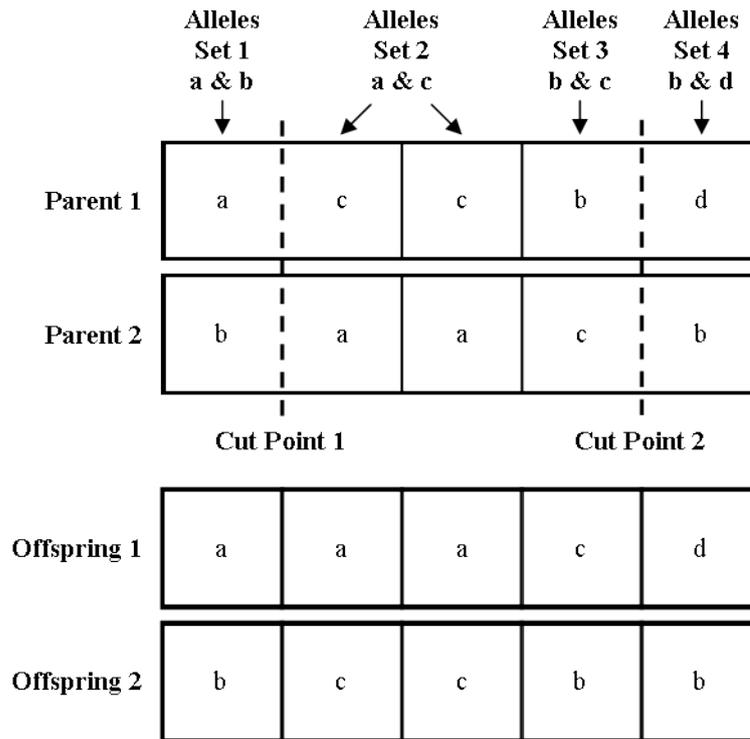


Figure 5.21 Homologous cut-point crossover

The experimental results in Chapter 6 illustrate the success of 2-cut point crossover to improve the level of accuracy in the search compared to other genetic operators. However, at a certain point in the evolutionary process, the operator may no longer generate offspring with improved fitness values because a local optimum has been reached. For the search to continue, population diversity is required.

#### Gene level crossover

The gene level crossover operator can provide significant search diversity and is controlled by a crossover threshold. This is set to 50%, which is standard probability used to promote the generation of diverse offspring (Deb, 2009). Probability values are assigned to the genes inside a chromosome using a random number generator. The genes assigned with a probability under the defined threshold are selected as cut points. The crossover of gene values is limited to the same sub-chromosome type. The sub-chromosomes and their subsequent genes are sequentially selected for crossover. A gene may only be selected once as a cut point. Where a pairing of genes inside a sub-chromosome is not possible, the genes are not swapped. Figure 5.22 illustrates the gene level crossover procedure with a probability of 0.7.

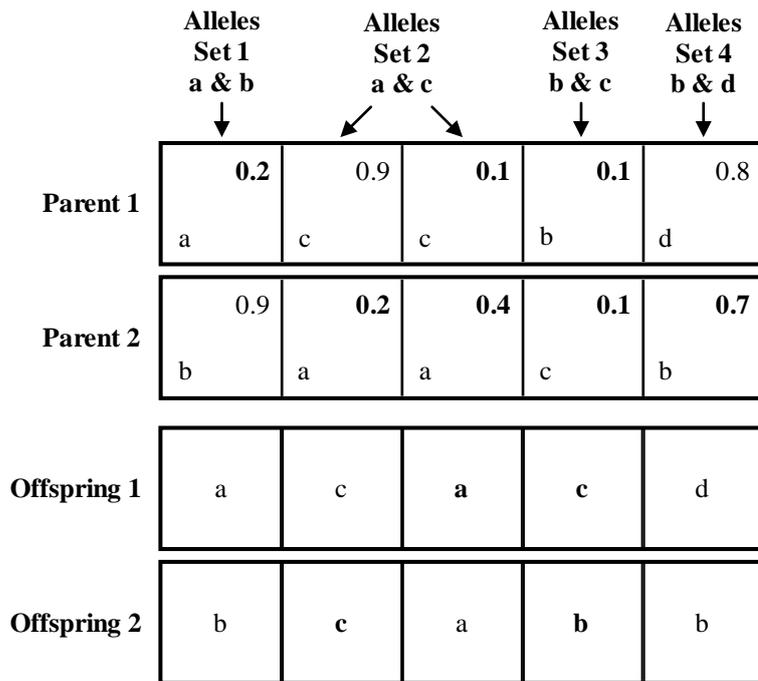


Figure 5.22 Gene level crossover

Like all crossover operators, the pitfall of the gene level one is that the mechanism is restricted to gene values existing within the population. In other words, this operator has no means of introducing new alleles to the population, which would encourage the exploration of new search space areas.

### Mutation

The mutation operator provides an additional level of diversity to the search because it can introduce new alleles into the population. The level of mutation is controlled by the mutation probability threshold. This is set to 0.2 because it provides the highest level of search accuracy over a number of problems, as shown in section (6.6.1). Mutation is an asexual operator, which assigns probabilities values to genes in the same way as described in gene level crossover. The genes which have been assigned a probability under the threshold are mutated to a legal sub-chromosome allele. Figure 5.23 illustrates the operator with a mutation rate of 0.2.

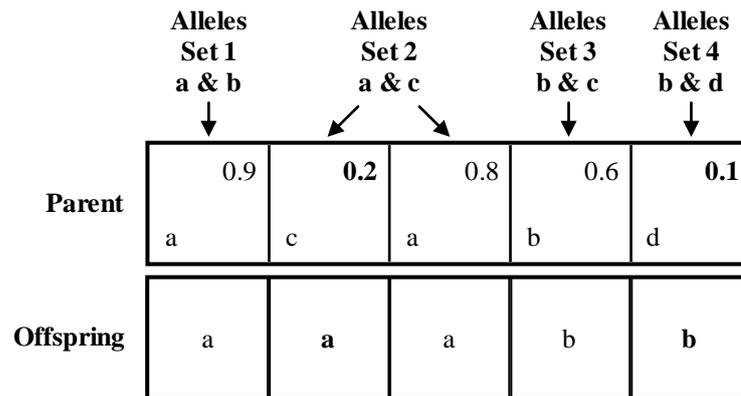


Figure 5.23 Mutation operator

### Decimation

The decimation operator deletes a proportion of the individuals inside the reproductive pool. The purpose of this operator is to prevent offspring with a genotype similar to that of their parents from surviving into the next generation. This operator deletes 10% of such individuals in the reproductive pool with the lowest solution quality, which is defined based on fitness values. The percentage is rounded to the nearest integer.

### Snap-shot (Injection)

The resultant offspring deficit is filled using Patelli (2011) snap-shot operator. This operator reintroduces individuals from the population recorded 10 generations ago into the current offspring population. The purpose of selecting individuals from a previous generation is to increase the population diversity, but at the same time minimise the negative impact on the average population fitness. In contrast, the deficit could have been filled using newly generated individuals; however, this might compromise the search, as randomly built chromosomes are unlikely to be fit. Since population diversity is not directly related to chromosome encoding, the promising results of using the Snap-shot operator reported in Patelli (2011) might also prove beneficial in the context of this approach.

### Hamming Distance

The aforementioned genetic operators are each applied at different stages of the evolution, depending on the level of population diversity, which is measured using Hamming Distance. The Hamming Distance measures the number of requests that are unique between a pair of individuals (Coello Coello et al., 2007). The greater the Hamming Distance the higher the diversity level. Figure 5.24 illustrates how the Hamming Distance is calculated in a population

of four individuals consisting of 5 genes. The average Hamming Distance will be used as the diversity measurement for the entire population.

<u>Population</u>	<u>Alleles Sets</u>				
	a and b	a and c	b and c	b and d	
Individual 1	a	c	c	b	d
Individual 2	b	a	a	c	b
Individual 3	a	c	a	c	b
Individual 4	b	c	a	b	b

<u>Pairings</u>	<u>Identical Gene Values</u>					<u>Hamming Distance</u>
Individual 1, Individual 2	1	1	1	1	1	5
Individual 1, Individual 3	0	0	1	1	1	3
Individual 1, Individual 4	1	0	1	0	1	3
Individual 2, Individual 3	1	1	0	0	0	2
Individual 2, Individual 4	0	1	0	1	0	2
Individual 3, Individual 4	1	0	0	1	0	2
Average Population Hamming Distance						2.83

Figure 5.24 Population average Hamming Distance

Diversity is not measured in every generation because it is unlikely to change drastically at such a fast pace and it also has a time complexity of  $O(P^2)$ , where P denotes the size of the population. Hamming Distance is computed every 10 generations to allow the genetic operators sufficient time to affect the search. The number of generations is deterministically determined.

DIVmin and DIVmax are thresholds used to define the type of genetic operator to be employed in the search. DIVmin represents the minimum acceptable level of population diversity required in the search. Whereas, DIVmax defines a sufficient level of population diversity. The DIVmax is adaptively determined using the initial population because diversity is thought to be maximum at this stage. The DIVmin is set 30% lower than DIVmax. It is plausible that DIVmax will not be achievable throughout the search because it is expected that the population will converge to a certain area of the search space, over generations. Therefore, this adaptive parameter may need to be lowered.

The following novel self adaptive approach is adopted because it has a positive effect on the search, in addition to the fitness assignment contribution brought by (5.2), as illustrated in Chapter 6. At every diversity measurement interval, the Hamming Distance is computed and depending upon the population diversity, a different genetic operator is employed. At the start of the evolutionary process the improvement in accuracy is of prime importance because the population is diverse. Therefore, the accuracy improving 2-cut point crossover is the first genetic operator employed. However, if the Hamming Distance falls below DIVmin at the measurement interval, the focus of the search changes to promote greater diversity, in an attempt to prevent the search from becoming trapped in a particular area. The gene level crossover, mutation or snapshot and decimation operators are available to stimulate greater search diversity. The self adaptive approach applies these operators in ascending order of their expected effect on population diversity, in order to minimise the impact on the search accuracy. Gene level crossover provides the least amount of diversity because it is restricted to a set of alleles inside the population. In contrast, mutation is likely to provide a greater level of diversity because it may have access to alleles outside the population. Finally, the decimation and snapshot operators provide the greatest impact of population diversity because they delete and replace entire individuals within the population. Therefore, when the Hamming Distance is below DIVmin the self adaptive approach operates as follows. First the 2-cut point crossover operator is switched to gene level crossover to promote greater diversity. During the next interval, if DIVmax is met, the 2 cut point crossover is reintroduced to encourage accuracy. However, if the gene level crossover operator is unsuccessful in regaining DIVmax, it is substituted with the mutation operator in order to insert new alleles into the population. Once again, if DIVmax is reached, the 2 cut point crossover operator is reapplied instead of the mutation operator. Otherwise, 10% of the offspring population is deleted and new genetic material is added into the population using decimation and snapshot operators, respectively. Subsequently, cut point crossover is reintroduced in the same run. Figure 5.25 illustrates the switching parameter and Figure 5.26 outlines the respective pseudo code.

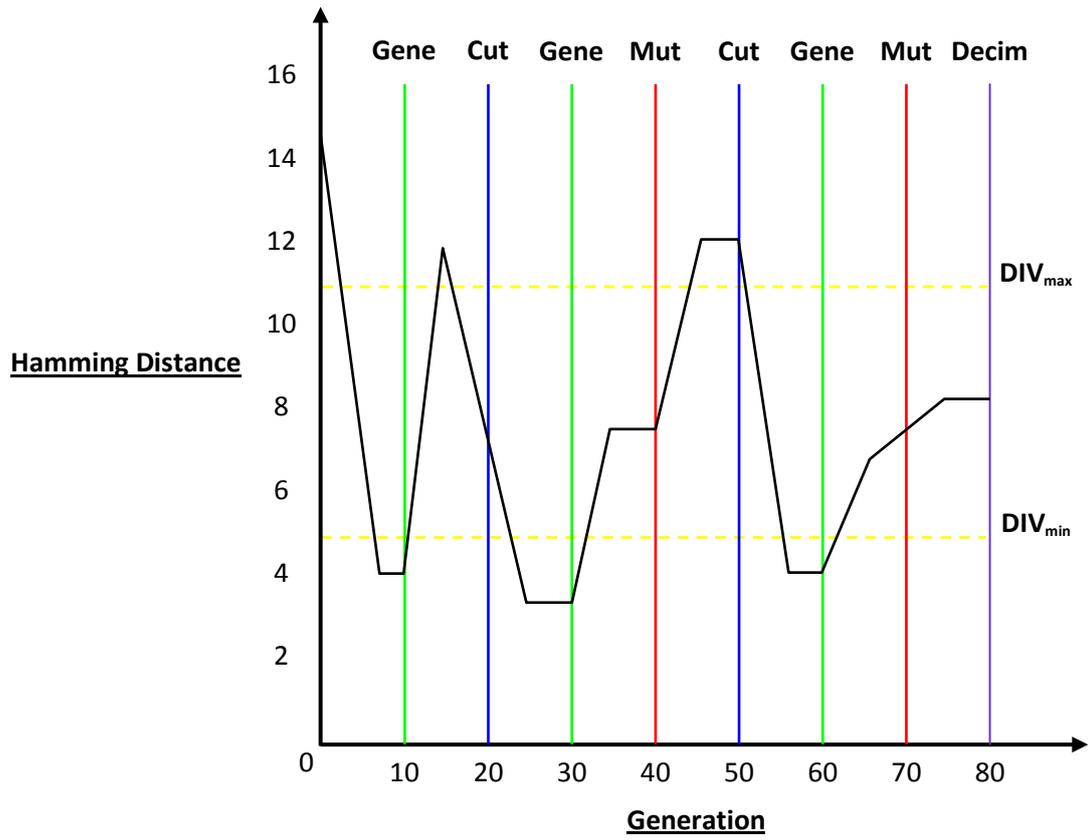


Figure 5.25 Switching Parameter

```

1 GeneticOperatorSwitchingMechanism(div, lb, ub) returns O
2 initialise O=CPC
3
4 do
5
6     if ub<=div
7         O=CPC
8     else
9         if lb<=div & O==CPC
10            O=CPC
11        else
12            if O==CPC
13                O=GLC
14                break
15            end
16            if O==GLC
17                O=Mut
18                break
19            end
20            if O==Mut
21                apply DEC & SS
22                O=CPC
23                break
24            end
25        end
26    end
27
28 end

```

Figure 5.26 Genetic operator switching mechanism pseudo code. Notations: div - average population diversity, lb - minimum diversity threshold, ub - maximum diversity threshold, O - Operator, CPC - 2-cut point crossover operator, GLC - Gene level crossover operator, Mut - Mutation operator, DEC - Decimation operator, SS – Snapshot operator Gene level crossover operator.

The 2-cut point crossover (CPC) is set as the default genetic operator, which continues to be applied as long as the population diversity remains above the lower threshold level (lines 6 to 10). In the case population diversity falls below this level, a set of diversity promoting operators: gene level crossover (GLC), mutation (Mut), decimation (DEC) and snapshot (SS) are applied in turn, until the upper threshold point is reached (lines 11 to 21), at which point CPC is applied again.

The purpose of the aforementioned self adaptive mechanism is to prevent the search from becoming trapped in a particular search space area. However, a fitness metric that measures the improvement in the phenotype is required. Therefore, the average distance between the

1<sup>st</sup> order PF and the origin of the search space axes is computed. Figure 5.27 illustrates this process. If the PF has not moved closer towards the origin, after 5 intervals, the Decimation and Snapshot operators are applied. The Snapshot operator is meant to replenish the population whilst having a reduced negative influence on the overall population quality, in comparison to randomly generating new individuals. The subsequent operator that follows is the 2 cut point crossover.

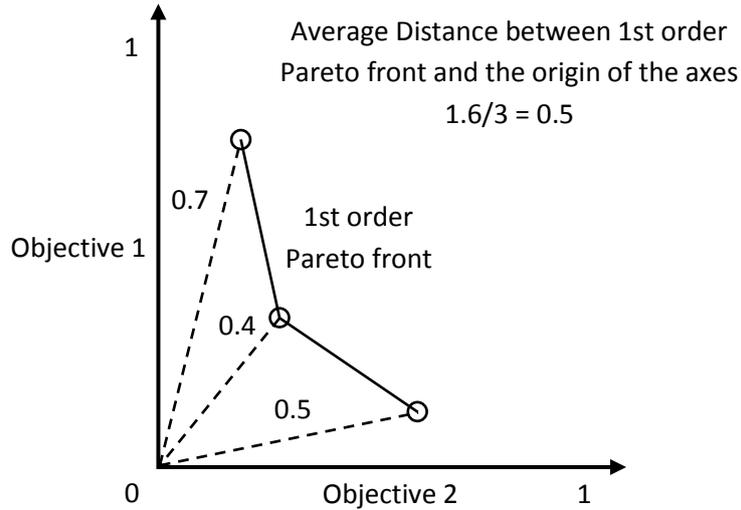


Figure 5.27 Average distance between 1<sup>st</sup> order Pareto front and the origin of the axes

The MOO problem considered here provided every objective with an equal importance, therefore to calculate the distance between the 1<sup>st</sup> order Pareto front and the origin of the axes, the objective values were normalised by scaling between 0 and 1. Normalisation was introduced to remove the bias resulting from different objective scale dimensions. Equation 5.8 defines the normalisation approach, where  $xn_{ij}$  is the normalised value of  $x_{ij}$ , the objective function value achieved by individual  $j$  for objective function  $i$ . Notations  $x_{max}$  and  $x_{min}$  represent, respectively, the highest and lowest objective values achieved by an individual in the first order Pareto front.

$$xn_{ij} = (x_{ij} - x_{min}) / (x_{max} - x_{min}) \tag{5.8}$$

The normalised values are afterwards averaged over the entire first order Pareto front, as shown in equation (5.9), where  $M$  represents the number of individuals in the first order Pareto front.

$$xn_i = \frac{\sum_{j=1}^M xn_{ij}}{M} \quad (5.9)$$

Finally, the distance between the 1<sup>st</sup> order Pareto front (PF1) and the origin  $O$  of the search space axes is computed using Equation 5.10, where  $Q$  denotes the number of objectives.

$$D(O, PF1) = \sqrt{\sum_{i=1}^Q xn_i^2} \quad (5.10)$$

where  $D(O, PF1)$  represents the distance between the 1<sup>st</sup> order Pareto front and the origin.

#### 5.7.5 Selection for Reinsertion

Selection for reinsertion chooses the individuals from the current and offspring population for the next generation population. The stochastic Proportional Roulette Wheel Selection adopted in the selection for reproduction stage of the algorithm is not implemented here, as the high selection pressure is unlikely to encourage population diversity required for search space exploration. Instead, an adaptive deterministic selection approach is used to make a selection decision based on the current genetic operator in use, which is previously selected in accordance to the level of population diversity.

The reinsertion decision is made using a population consisting of all current and offspring individuals. The individuals in the population are sorted according to their fitness values. The adopted reinsertion approach depends on the genetic operator currently employed. If the accuracy promoting cut point crossover is employed, the fittest individuals in the population will be selected for the next generation, therefore encouraging the increase accuracy of the search. Where more than two individuals share the same level of fitness, two individuals are randomly selected for the next generation. Whereas, if either the gene level crossover or mutation operator is applied, the selection process selects individuals occupying the centre of the population in terms of fitness. This is expected to promote diversity. The other option to select the worse performing individuals is dismissed even in the case when this is diversity

friendly because this does not encourage the search in terms of accuracy. Figure 5.28 illustrates the selection process for four individuals.

<b><u>Selection for Reinsertion Accuracy Inducing Strategy</u></b>		
<u>Population</u>	<u>Fitness Values</u>	<u>Next Generation</u>
Offspring 2	0.9	Accept
Individual 2	0.85	Accept
Offspring 4	0.8	Accept
Individual 4	0.75	Accept
Individual 3	0.7	Reject
Offspring 3	0.65	Reject
Offspring 1	0.6	Reject
Individual 1	0.55	Reject

<b><u>Selection for Reinsertion Diversity Inducing Strategy</u></b>		
<u>Population</u>	<u>Fitness Values</u>	<u>Next Generation</u>
Offspring 2	0.9	Reject
Individual 2	0.85	Reject
Offspring 4	0.8	Accept
Individual 4	0.75	Accept
Individual 3	0.7	Accept
Offspring 3	0.65	Accept
Offspring 1	0.6	Reject
Individual 1	0.55	Reject

Figure 5.28 Selection for Reinsertion

#### 5.7.6. Termination Criterion

The evolutionary process is stochastic, therefore a stopping criterion is required. The generational stopping criterion is adopted because it is most commonly applied within evolutionary algorithms (Koza, 1998). The maximum number of search generations is experimentally tuned for each test problem evaluated in Chapter 6. The following evaluation (convergence) and exception stopping criterions were considered, however none are applicable in this research context. The evaluation criterion is not used because acceptable objective values have not been defined. The exception criterion is not applicable because genetic operators are used to reintroduce diversity into the population, therefore this criterion is unlikely to be met.

### 5.7.7 Proposed GA compared to NSGA-II

The genetic algorithm proposed in SDPmethod is compared against the most prominent Pareto MOGA method, NSGA-II. The analysis will compare the design of each genetic algorithm aspect, in order to gain a better understanding of performance. Table 5.1 defines the commonalities and dissimilarities between the two methods.

With respect to initialisation, both NSGA-II and SDPmethod aim to maximise exploration of the search space by randomly generating the initial population individuals. In particular, the SDPmethod experimentally tunes the population size for an improved search space exploration, although no such reference is made for the original NSGA-II.

In the fitness computation stage, both methods perform nondominated sorting on the combined parent and offspring population, in an attempt to maintain the best individuals through to the next generation. The nondominated sorting methods have a  $O(MN^2)$  computational complexity, where  $M$  is the number of objectives and  $N$  is the population size. Thus, the resources required to solve the problem are likely to be comparable. Moreover, each method assigns a uniform fitness value to individuals in the same nondominated set and a lower fitness importance to a set in ascending order of rank. Consequently, this fitness assignment approach promotes the selection of individuals on lower order Pareto sets, which increases the accuracy inducing capabilities of the two methods. However, a noticeable difference remains between how the solutions on the same front are differentiated by the two approaches. The SDPmethod distinguishes between solutions on the same Pareto set through fitness sharing, which is not adopted by NSGA-II due to the challenges related to defining a sharing parameter. Instead NSGA-II adopts an alternative crowding procedure in the selection for reinsertion phase, to set apart solutions on the same nondominated set. However, the SDPmethod introduces dynamic sizing of the sharing parameter, which overcomes the issues of defining the parameter. Therefore, the primary difference between the two methods is the way in which solutions on the same Pareto set are differentiated.

The SDPmethod and NSGA-II differ in the selection for reinsertion, as the proportional roulette wheel selection and the binary tournament selection are respectively applied. The proportional roulette wheel selection provides greater search accuracy than the binary tournament because the selection for reproduction pressure is higher (De Jong, 2006).

Furthermore, the binary tournament may suffer from slow convergence to the Pareto optimal set as the tournament set may comprise of randomly selected low fitness individuals.

The two methods differ with respect to the genetic operators used for offspring generation. The NSGA-II is limited to crossover and mutation operators for generating search accuracy and diversity respectively. Whereas, SDPmethod utilises the aforementioned, as well as a gene level crossover operator, in order to induce greater search diversity. Moreover, SDPmethod unlike NSGA-II, may utilise the population based operators decimation and snapshot to stimulate additional search diversity. In consequence, SDPmethod has a wider range of diversity inducing operators compared to NSGA-II, which reduces the risk of premature convergence. In addition, the SDPmethod has a more advanced approach towards search progression because it is able to differentiate between the need for greater search accuracy or diversity. This is achieved with the use of an adaptive genetic operator control mechanism, which applies a particular operator based on the needs of the search. Since the adaptive mechanism can apply varying degrees of diversity in the search, the SDPmethod can be considered superior in navigating through the search space.

With respect to selection for reinsertion, both methods consider the parent and offspring individuals for reinsertion, which increases the likelihood for fitter individuals to remain in the search. NSGA-II applies an elitist approach by ensuring the fittest individuals are selected for the next generation population. In contrast, the SDPmethod selection strategy depends upon the needs of the search. For instance, NSGA-II elitist approach is adopted if the focus of the search is to induce greater accuracy, whereas mid range ranked fitness individuals are inserted for the next generation, if the focus of the search is to increase diversity. This adaptive selection strategy enables the exploration of improved areas of the search space. The two methods differ in the situation where the nondominated set to be added exceeds the next generation population size. NSGA-II applies the Crowding-sort procedure to select a diverse set of solutions from that set in the selection stage, whereas the SDPmethod used the previously determined fitness values to make the decision. There is no selection difference when using either fitness sharing or the crowding sort procedure, if the sharing parameter is calculated correctly, which in the case of SDPmethod is dynamically determined. Finally, both methods may suffer a loss of convergence ability when the 1<sup>st</sup> order Pareto set is larger than the next generation population because some non-dominated individuals are deleted to make space for others (Deb, 2009).

To summarise, the SDPmethod is an improvement to the NSGA-II because it provides a higher selection pressure at the selection for reproduction stage, which is likely to result in a mating pool of elite individuals. In addition, the SDPmethod has a larger depository of reproduction operators to generate a diverse offspring population. Finally, the recombination and selection for reinsertion stages are adaptively aligned with the needs of the search, unlike the NSGA-II.

Table 5.1 Proposed Multi Objective Genetic Algorithm compared to NSGA-II

Genetic algorithm aspect	Common Characteristics	Unique Characteristics		Expected advantage(s) of proposed MOGA over NSGA-II
		NSGA-II	SDPmethod	
<i>Initialisation</i>	Initial population randomly generated			
			Experimentally tuned population size	Better search space coverage
<i>Linear Evaluation</i>	Nondominating sorting with combined parent and offspring population - $O(MN^2)$		Fitness sharing	
	Uniform fitness for same nondominated set individuals			
<i>Selection for Reproduction</i>		Binary tournament selection	Proportional roulette wheel selection	Greater search accuracy
<i>Genetic and Population Operators</i>		Crossover and mutation	2-cut point crossover, gene level crossover, mutation, decimation and snapshot	Larger range of diversity inducing genetic operators
			Adaptive parameter tuning to control the evolution	Superior navigation through the search space
<i>Selection for Reinsertion</i>	Combined parent and offspring population	Elitist	Elitist and non-Elitist	Superior navigation through the search space
		Crowding procedure		

## 6. RESULTS AND DISCUSSION

This chapter will present and discuss the experimental contributions of this research in relation to the introduced routing heuristic, RouteAlg and the novel VRPSDP method called SDPmethod. The chapter is organised as follows: the experimental setup description is provided in section 6.1, the vehicle fleet size lower bounds are defined for the studied test problems in section 6.2, section 6.3 contains the published experimental case-studies overview, the RouteAlg results are presented and discussed in section 6.4, whereas the SDPmethod results are validated in section 6.5 and the genetic algorithm, a crucial component of the SDPmethod, is analysed in section 6.6.

### 6.1 Experimental Setup Description

The author implemented the RouteAlg and SDPmethod in Matlab and evaluated them on the test problems from Salhi and Nagy (1999), the most widely used benchmarked instances in the VRPSDP domain (Jun and Kim, 2012).

The problem size ranges between 50 and 199 customers. There are two test problem types CMT $k$ X and CMT $k$ Y, where CMT is the name of the test problem,  $k$  is the problem identifier ( $k=1, \dots, 14$ ) and X or Y refers to the demand type. A summary of the test problems is provided in Table 6.1. The Salhi and Nagy (1999) test problems were originally derived from the VRP instances given in Christofides et al. (1979). For every customer ( $j$ ), the original demand was split to form a new delivery and pickup demand. The ratio  $r_j$  of the split was calculated as  $\min((x_j/y_j), (y_j/x_j))$ , where  $x_j$  and  $y_j$  are the coordinates for customer  $j$ . The delivery demand  $d_j$  is set to  $r_j \times t_j$  and the pickup demand  $p_j$  to  $(1-r_j) \times t_j$ , where  $t_j$  is the original customer demand. Another set of problems classed as Y are generated by switching the demands for every other customer. In addition, the Euclidean distance between two customers can be calculated using their coordinates.

Table 6.1 Summary of the benchmark test problems

<u>Problem</u>	<u>Number of Customers</u>	<u>Vehicle Capacity (Tons)</u>	<u>Maximum Route Length</u>
CMT1X/Y	50	8	N/A
CMT2X/Y	75	7	N/A
CMT3X/Y	100	10	N/A
CMT4X/Y	150	10	N/A
CMT5X/Y	199	10	N/A
CMT6X/Y	50	8	200
CMT7X/Y	75	7	160
CMT8X/Y	100	10	230
CMT9X/Y	150	10	200
CMT10X/Y	199	10	200
CMT11X/Y	120	10	N/A
CMT12X/Y	100	10	N/A
CMT13X/Y	120	10	720
CMT14X/Y	100	10	1040

Column 1 - Name of the test problem, Column 2 - Number of nodes considered in the problem, Column 3 - Homogenous vehicle capacity in Tons, Column 4 - Maximum length of a route.

The vehicle capacity units for CMT1X/Y, ..., CMT10X/Y test problems were originally defined in terms of tons, whereas CMT11X/Y, ..., CMT14X/Y were defined in terms of Centum weight (cwt). For the purpose of consistency, the latter test problems are converted to tons, as presented in Table 6.1. The Salhi and Nagy (1999) test problems CMT1X/Y, ..., CMT10X/Y were generated using a random uniform distribution, whereas CMT11X/Y, ..., CMT14X/Y were generated using a clustering distribution, which is more reflective of the actual routing problems than the former, Christofides et al. (1979). In addition, CMT6X/Y, ..., CMT10X/Y, CMT13X/Y and CMT14X/Y test problems have a maximum route length restriction imposed.

## 6.2 Vehicle Fleet Size Lower Bounds

The vehicle fleet size lower bounds ( $n_v$ ) for the Salhi and Nagy (1999) test problems are provided in Table 6.2. Equation 6.1 is used to calculate the bounds based on the capacity requirements of each problem.

$$n_v = \max \left( \sum \text{Delivery Weight} / \text{Vehicle Capacity}, \sum \text{Pickup Weight} / \text{Vehicle Capacity} \right) \quad (6.1)$$

The  $n_v$  described in Table 6.2 for CMT6X/Y, ..., CMT10X/Y and CMT13X/Y, ..., CMT14X/Y test problems may need to be increased because they include a maximum route distance restriction. However, with respect to the test problems with no route limit, the lower and upper bounds are equal. The  $n_v$  in Table 6.2 have been determined without considering service times for each node.

Table 6.2 Vehicle fleet size lower bounds based on the capacity demands for each test problem

<b><u>Test Problem</u></b>	<b><u>Total Delivery Weight (Tons)</u></b>	<b><u>Total Pickup Weight (Tons)</u></b>	<b><u>Vehicle Capacity (Tons)</u></b>	<b><u>Vehicle Fleet Size Lower Bounds</u></b>
CMT1X	23.02	15.83	8	3
CMT1Y	17.14	21.71	8	3
CMT2X	40.84	27.36	7	6
CMT2Y	35.23	32.97	7	6
CMT3X	41.9	31	10	5
CMT3Y	36.42	36.48	10	4
CMT4X	64.92	46.83	10	7
CMT4Y	53.56	58.19	10	6
CMT5X	93.55	65.75	10	10
CMT5Y	76.6	82.7	10	9
CMT6X	23.02	15.83	8	3
CMT6Y	17.14	21.71	8	3
CMT7X	40.84	27.36	7	6
CMT7Y	35.23	32.97	7	6
CMT8X	41.9	31	10	5
CMT8Y	36.42	36.48	10	4
CMT9X	64.92	46.83	10	7
CMT9Y	53.56	58.19	10	6
CMT10X	93.55	65.75	10	10
CMT10Y	76.6	82.7	10	9
CMT11X	30.55	38.2	10	4
CMT11Y	32.83	35.92	10	4
CMT12X	47.1	43.4	10	5
CMT12Y	46.16	44.34	10	5
CMT13X	30.55	38.2	10	4
CMT13Y	32.83	35.92	10	4
CMT14X	47.1	43.4	10	5
CMT14Y	46.16	44.34	10	5

Column 1 - Name of the test problem, Column 2 - The sum of delivery weights for all nodes, Column 3 - The sum of pickup weights for all nodes, Column 4 - Homogenous vehicle capacity in Tons, Column 5 - Vehicle fleet size lower bounds based on the largest total weight requirement.

### **6.3 Published experimental case-studies overview**

The RouteAlg is compared against Vural (2007), the only paper featuring route information within the scope of this research. Vural (2007) has provided eighty six routing solutions for the following test problems: CMT1X/Y, CMT2X/Y, CMT3X/Y, CMT4X/Y, CMT5X, CMT11X/Y and CMT12X/Y. The published results have been validated here and all are correct with the exception of CMT11Y, where the total routing distance is understated. The herein experimental study will evaluate the RouteAlg with respect to only seventy routing problems studied in Vural (2007), whereas the other sixteen trivial problems with one or two route nodes are excluded because they provide no grounds for comparison. The SDPmethod is tested on all Salhi and Nagy (1999) test problems because several solutions have been published, which define the number of vehicles operated and the total routing distance. A comparison is made with the following research works in regards to the test problems with no maximum route distance restriction: Jun and Kim (2012), Subramanian et al. (2010a), Wassan et al (2008), Vural (2007) and Chen and Wu (2006). For those test problems where a limit is applied, the SDPmethod is compared against Wassan et al (2008) and Montane and Galvao (2006) because the authors have not imposed a service time at each node, which is consistent with this work. However, all the aforementioned research works, with the exception of Vural (2007) have not published their individual routing solutions, therefore their results cannot be verified and the solution quality in terms of the workload objective variation considered here cannot be universally analysed.

The comparison conducted here has excluded many research works such as: Goksal et al (2013), Zachariadis et al (2010), Gajpal and Abad (2009), Crispim and Brandao (2005) and Dethloff (2001) because the test problem type CMT $k$ Y is constructed in a different manner than is defined in Salhi and Nagy (1999). The excluded works have swapped the delivery and pickup demands for every customer in the test problem type CMT $k$ X, in order to generate the test problem type CMT $k$ Y. However, Salhi and Nagy (1999) recommend the swap should have been carried out for every other customer.

### **6.4 RouteAlg Validation Results**

The components of the RouteAlg: Modified Nearest Neighbourhood (MNN) algorithm, Reverse procedure, Ejection/Reinsertion (EjRi) and 2-opt/Or-opt method are analysed in terms of their contribution to the performance of the algorithm. Thereafter, the complete algorithm is

compared against the solutions published by Vural (2007). Detailed routing information is provided in Appendix A, which has been determined in a single run.

#### 6.4.1 Modified Nearest Neighbourhood (MNN) algorithm

The MNN algorithm is used to generate a reasonably good solution to the Travelling Salesman Problem (TSP), which is a relaxed version of the routing problem solved here. The proposed MNN algorithm is compared against the Nearest Neighbourhood (NN) algorithm and the findings are summarised in Figure 6.1. It is important to mention that the MMN algorithm will generate results of at least equal quality to the NN algorithm, as the former is derived from the latter, but offers additional diversity. The MNN algorithm is proven to be a more competitive approach in comparison to the NN algorithm because improved solutions have been found for the TSP for 80% (56) of the test problems studied, with a mean and median improvement of 5.81% and 4.38%, respectively, as shown in Figure 6.1. The most frequent amount of improvement was found in the 0% → 1% band with 19 routes. Although, there are other bands with a fair number of solutions that have reduced the routing distance, noticeably 1% → 2%, 4% → 5%, 8% → 9% and 10% → 11% bands. The largest improvement was found in 27% → 28%, which improved the solution quality by 27.36% for CMT2Y – Route: 3. In contrast, the MMN algorithm was unable to improve the solution quality for 20% (14) test problems. This result was not confined to a particular problem size, but was spread across routes with between 4 to 30 nodes. However, Figure 6.2 illustrates that the level of improvement introduced by MNN algorithm did on average diminish with an increase in the problem size, after eliminating statistical singularities 6% → 7% and 11% → 12% samples.

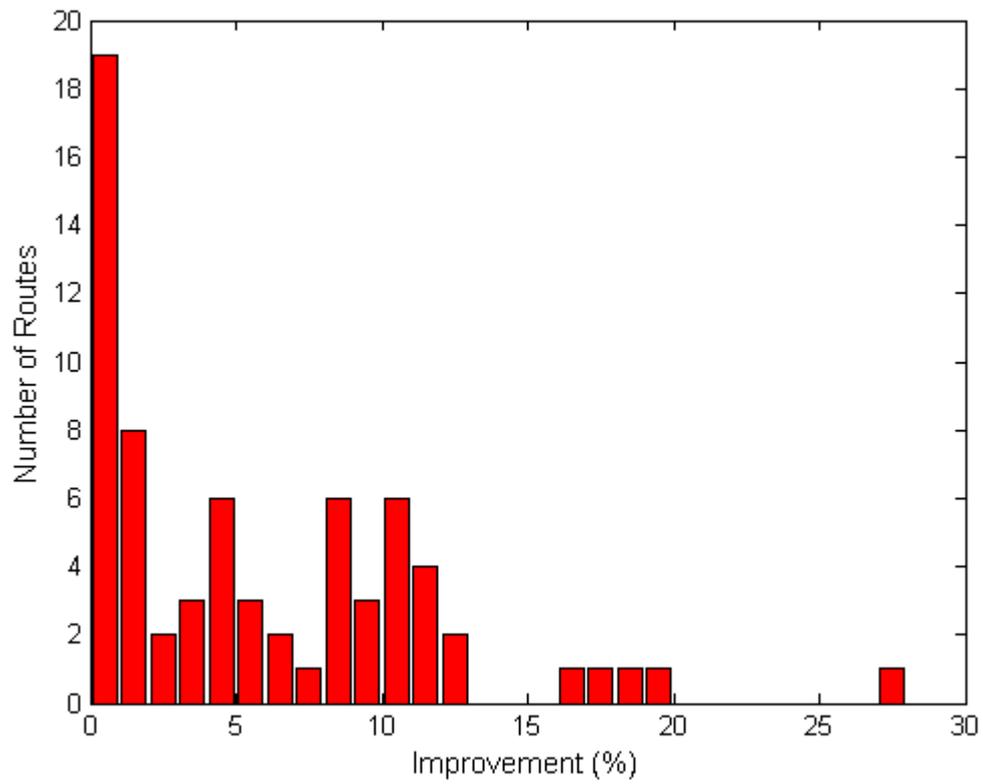


Figure 6.1: A bar graph showing the frequency the MNN algorithm found an improved route compared against the NN algorithm. Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

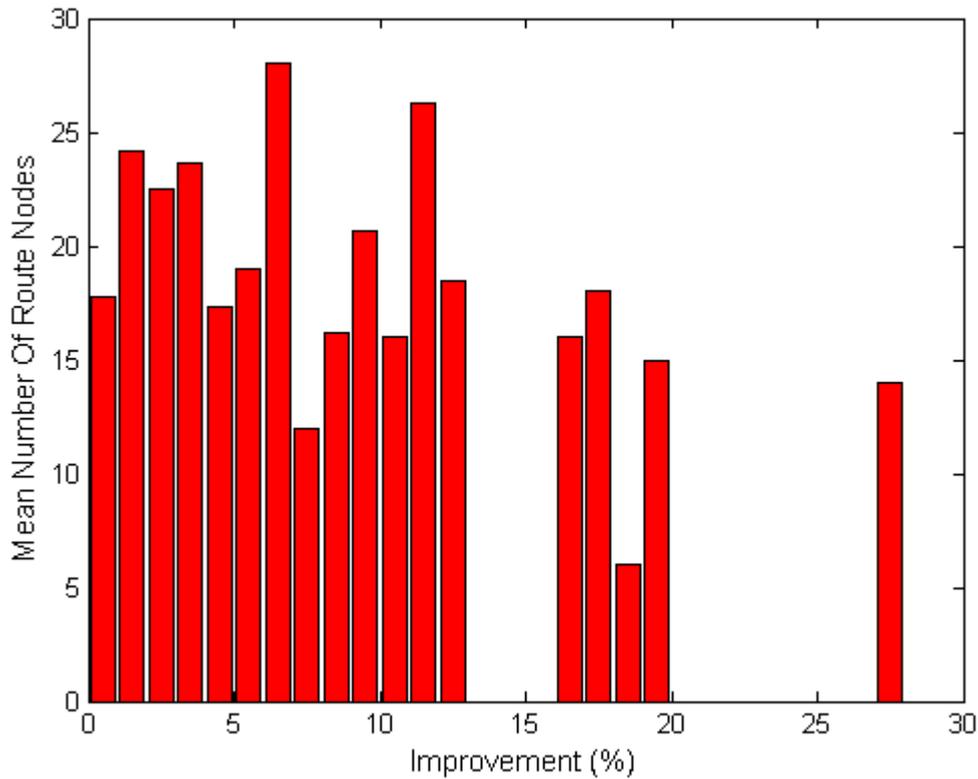


Figure 6.2: Mean number of route nodes. Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

#### 6.4.2 Reverse procedure

For a vast majority, 81.43% (57) routes, the MNN algorithm was able to generate capacity feasible solutions. In consequence, the Reverse procedure was applied to address the capacity infeasibility issue relating to the remaining 13 routes. A capacity infeasible solution defines a route where a vehicle capacity violation has occurred on at least one node and is measured as

$$\frac{R_E}{R} * 100 \tag{6.2}$$

where  $R_E$  defines the number of route nodes where the vehicle capacity has been exceeded and  $R$  is the number of route nodes.

Figure 6.3 illustrates the effect of reversing the orientation of the 13 capacity infeasible routes has on their feasibility. The reverse procedure reduced the level of capacity infeasibility for 53.85% (7) of the routes. In particular, the reverse procedure led to complete capacity feasibility in 38.46% (5) of instances: CMT2X – Route: 2; CMT2Y – Route: 3 & 5; CMT3Y –

Route: 3 and CMT5X – Route: 3, where the latter had the largest capacity infeasibility at 77.27%. In contrast, the reverse procedure had a negative impact on feasibility for 46.15% (6) routes. However, this outcome mainly occurred for solutions with an insignificant initial infeasibility level. For instance: the incumbent infeasibility level for CMT2Y – Route: 2, CMT3Y – Route: 1 and CMT4Y – Route: 1 & 3 was 6.25%, 7.69%, 3.7% and 3.57%, respectively. Therefore, the experimental results suggest that the Reverse procedure is more efficient at introducing capacity feasibility on routes with an initially large infeasibility compared to ones with a relatively low base. On a different aspect of importance, no correlation exists between the Reverse procedure ability to induce route capacity feasibility with the route node size, as capacity infeasibility was increased and decreased over a range of routes sizes, as shown in Figure 6.4.

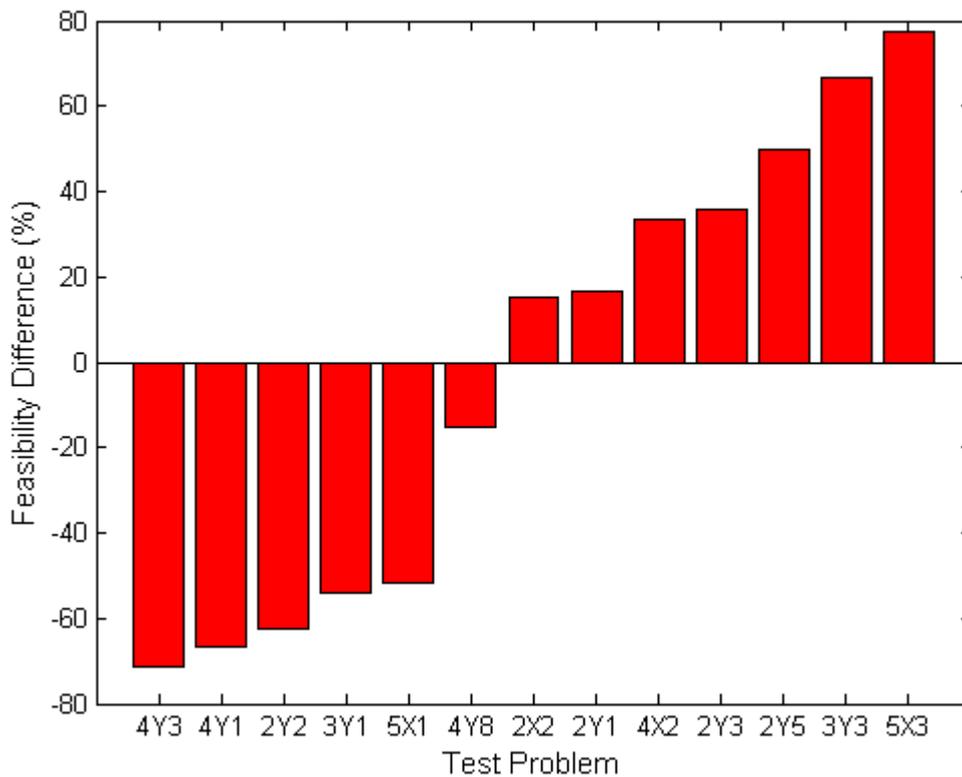


Figure 6.3: A bar graph showing the percentage difference in route TSPSDP infeasibility following the application of the Reverse procedure. X-axis label - Last digit of the bar identifies the route in the particular test problem.

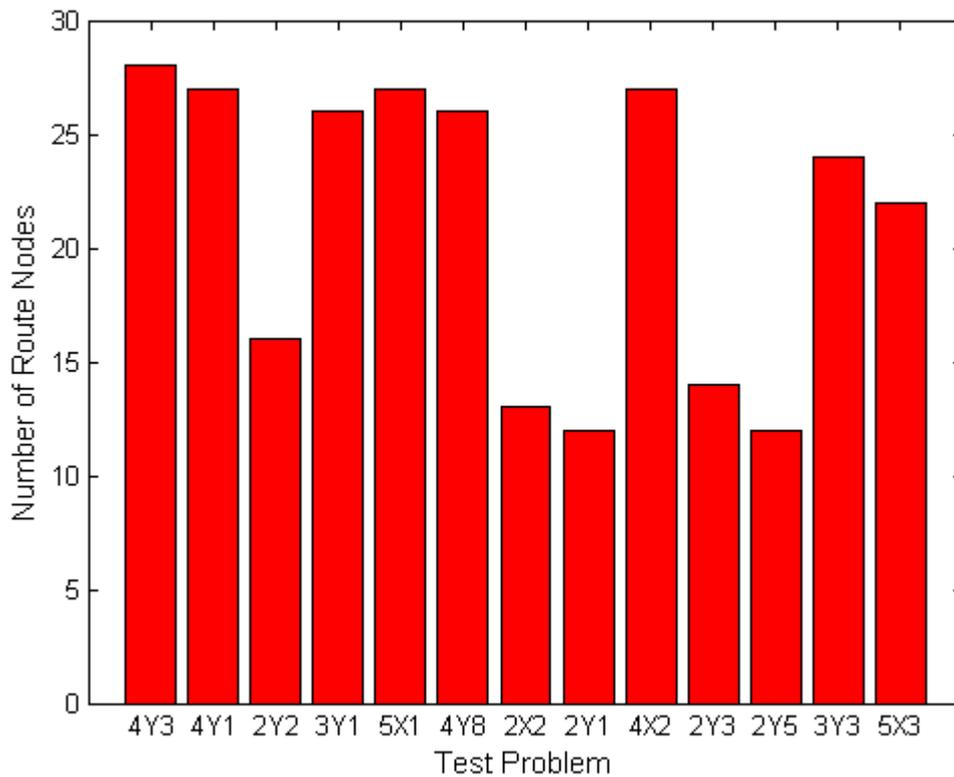


Figure 6.4 Number of route nodes. X-axis label - Last digit of the bar identifies the route in the particular test problem.

### 6.4.3 Ejection/Reinsertion (EjRi) method

The EjRi method guides the TSP solution towards a TSPSDP feasible space by resequencing nodes on a route in accordance to the criteria defined in Equation 5.1. The EjRi method was only applied to 9.30% (8) of the routes because the aforementioned MNN algorithm and the Reverse procedure were able to generate TSPSDP feasibility for the other routes. The TSPSDP infeasibilities levels of these routes, prior to the application of the EjRi method, are illustrated in Figure 6.5. The largest level of route infeasibility is 25% for CMT2Y – Route: 1 and followed closely by CMT4Y – Route: 8 at 23.08%. Whereas, the other instances have far lower route infeasibility levels, for example, CMT4Y – Route: 1 and 3. The EjRi method managed to introduce TSPSDP feasibility in all cases (ranging from 12 to 28 nodes), which demonstrates the effectiveness of the method. The subsequent route distances following the application of EjRi method can be found in the Appendix A in Table 14, which will be compared to the routing results from the forthcoming 2-opt/Or-opt method.

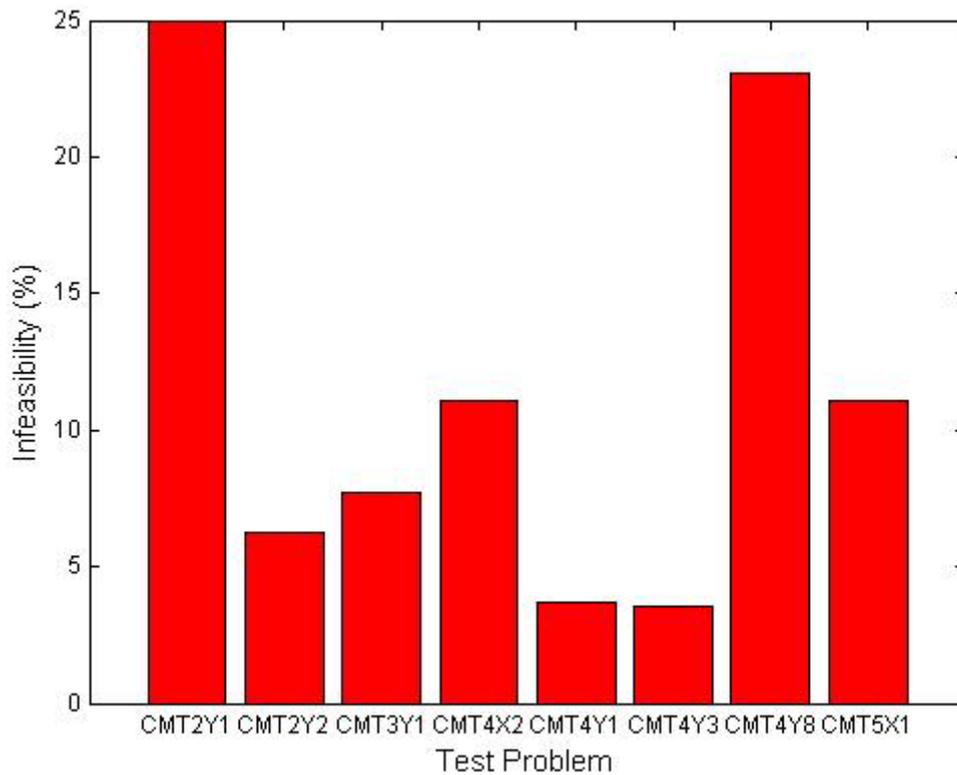


Figure 6.5: A bar graph showing the TSPSDP infeasibility percentage of the routes before EjRi method is applied. X-axis label - Last digit of the bar identifies the route in the particular test problem.

#### 6.4.4 2-opt/Or-opt method

The 2-opt/Or-opt method is an optimisation method that uses intra-route operators in order to improve the quality of the existing route. The 2-opt/Or-opt method is evaluated by comparing its results with pre-optimisation routing distances, which have been determined by the MNN algorithm or following the application of the EjRi method. The results have been summarised in Figure 6.6. The 2opt/Or-opt method improved the solution quality for 91.43% (64) of routes, with a mean improvement of 9.13%. However, for a great number of the routes the level of improvement is modest, see 0% → 1% band, which has 6 routes with no improvement. This is mainly a consequence for routes with a small number of nodes. Figure 6.7 shows a general positive correlation at various rates between the solution quality improvements introduced by the 2-opt/Or-opt method against an increase in the problem size. Initially, a sharp increase in the mean problem size (up to ~22 nodes) leads to a mild increase in solution quality up until 7% → 8% band. Thereafter, a small increase in the problem size results in a substantial increase in solution quality between 16% → 17% to 27% → 28% bands, after excluding the statistical anomaly 18% → 19% band. Another aspect of

importance is that the 2-opt/Or-opt method improved the solution quality of all routes that were amended by the EjRi method. In particular, the largest improvement was achieved for CMT2Y – Route: 2, which was previously modified by the EjRi method. This improvement amounted to 32.74%, as shown in the 32% → 33% band. Therefore, the 2-opt/Or-opt method is powerful enough to improve pre-optimisation solutions, which are of a poorer quality. Moreover, the 2-opt/Or-opt method’s contribution in terms of overcoming route TSPSDP infeasibility is difficult to evaluate given the success of the reverse procedure and the EjRi method on this issue. However, the 2-opt/Or-opt method was able to improve the solution quality, whilst maintaining capacity feasibility, therefore is proven to be a powerful local search method for the TSPSDP.

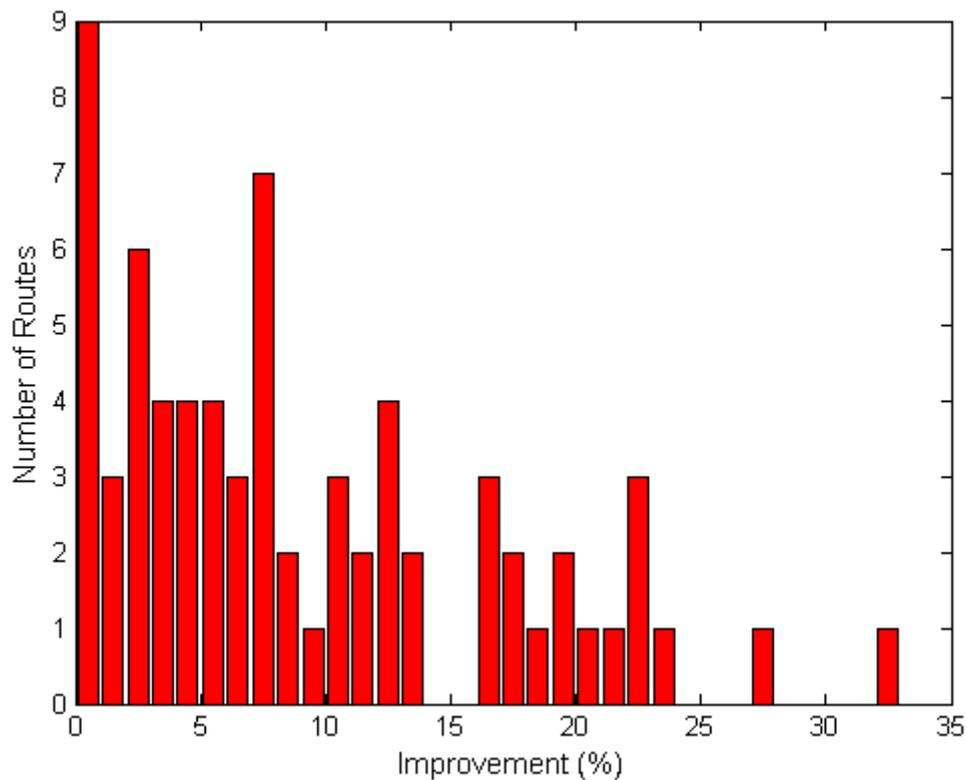


Figure 6.6: A bar graph showing the frequency the 2-opt/Or-opt method found an improved route compared against the MNN algorithm and EjRi method. Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

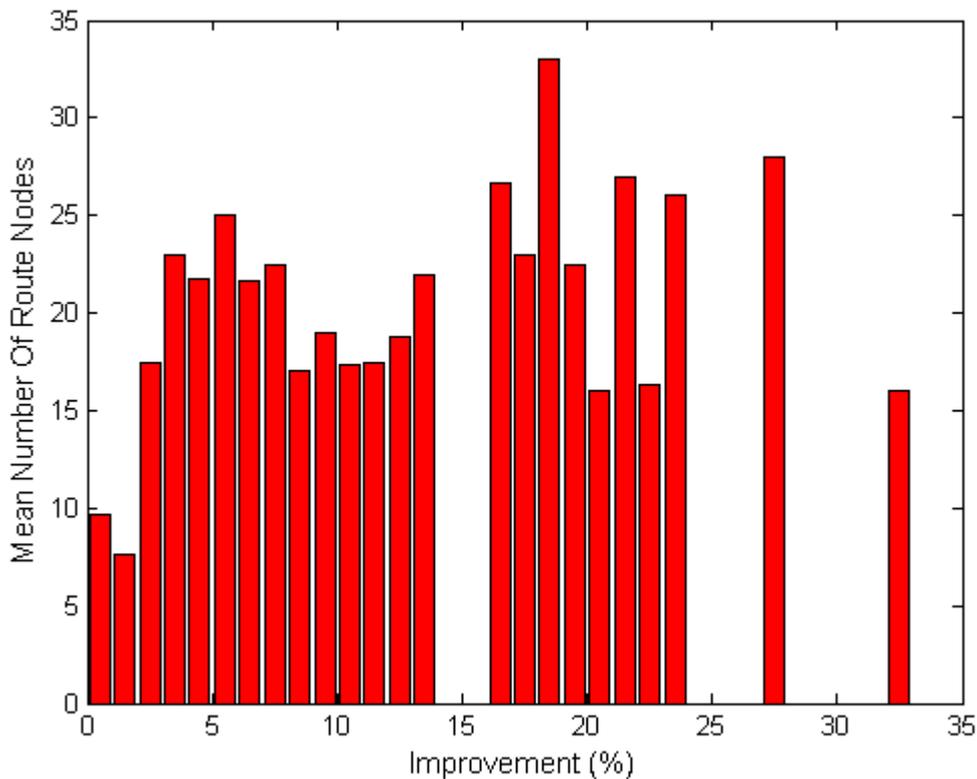


Figure 6.7: Mean number of route nodes. Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

#### 6.4.5 Results discussion

##### Solution quality

The computational output from the herein introduced RouteAlg is compared against that of Vural (2007) and is summarised in Figure 6.8. The RouteAlg found new best known solutions for 50% (35 routes) of the test problems evaluated by Vural (2007). The mean improvement was 2.12% with the largest single improvement of 16.58%, as shown in 16%  $\rightarrow$  17% band for test problem CMT5X – Route: 4. Although, the most amount of improvement lies within the 0%  $\rightarrow$  1% band, the RouteAlg has generated a fair number of solutions, which feature a decrease in routing distance, especially in the 1%  $\rightarrow$  2% and 5%  $\rightarrow$  6%. In contrast, the RouteAlg found worse solutions than Vural (2007) for only 6.98% (6 routes) of the test problems: CMT1Y – Route 2, CMT2X – Route 2, CMT2Y – Route 2, CMT4Y – Route 4, CMT5X – Route 5 and CMT12Y – Route 5 and the number of nodes on each route were 16, 13, 16, 28, 25 and 20, respectively. However, 4.65% (4) routes were only less than 1% worse than Vural (2007). The worst solution found by RouteAlg was only 2.26% higher than the one found by Vural (2007), in terms of routing distance, for the instance CMT5X – Route: 5. Given that the number of testcases where RouteAlg generated improved routes with respect to Vural (2007)

results far exceeds the number of worse solutions, the RouteAlg is recommended as a noteworthy tool in the field of TSPSDP problem solving. In addition, Figure 6.9 illustrates that the performance of the RouteAlg relatively improves with an increase in the problem size, as there are two regions of increase delimited by: 0% → 1% to 3% → 4% bands and 5% → 6% to 13% → 14% bands.

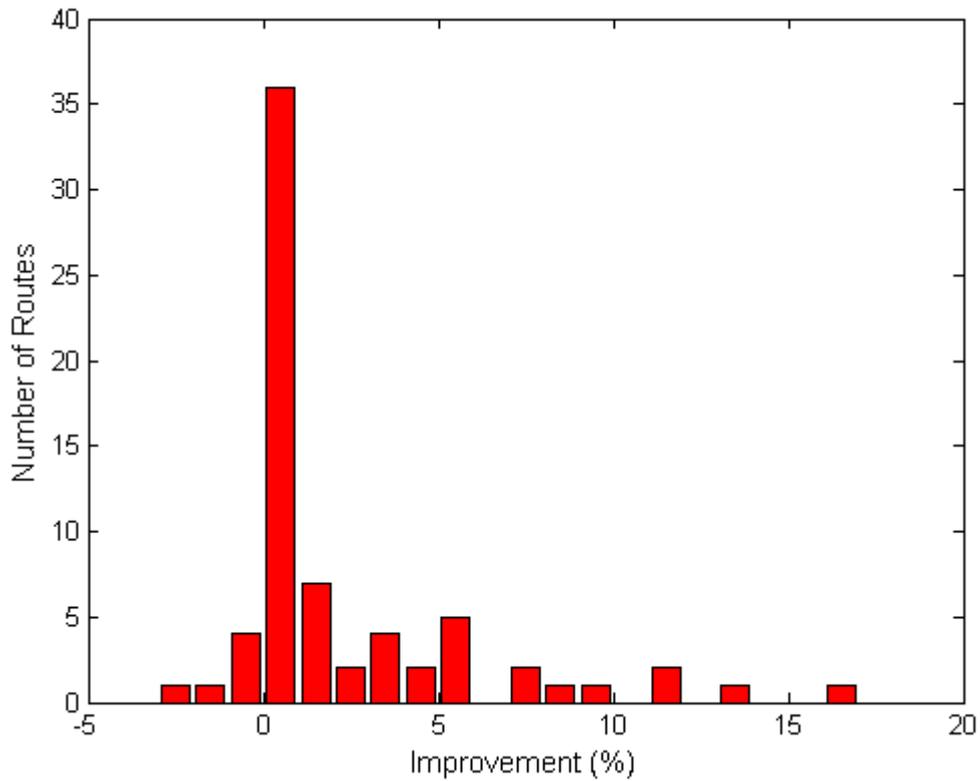


Figure 6.8: A bar graph showing the frequency the RouteAlg found an improved route compared against those found by Vural (2007). Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

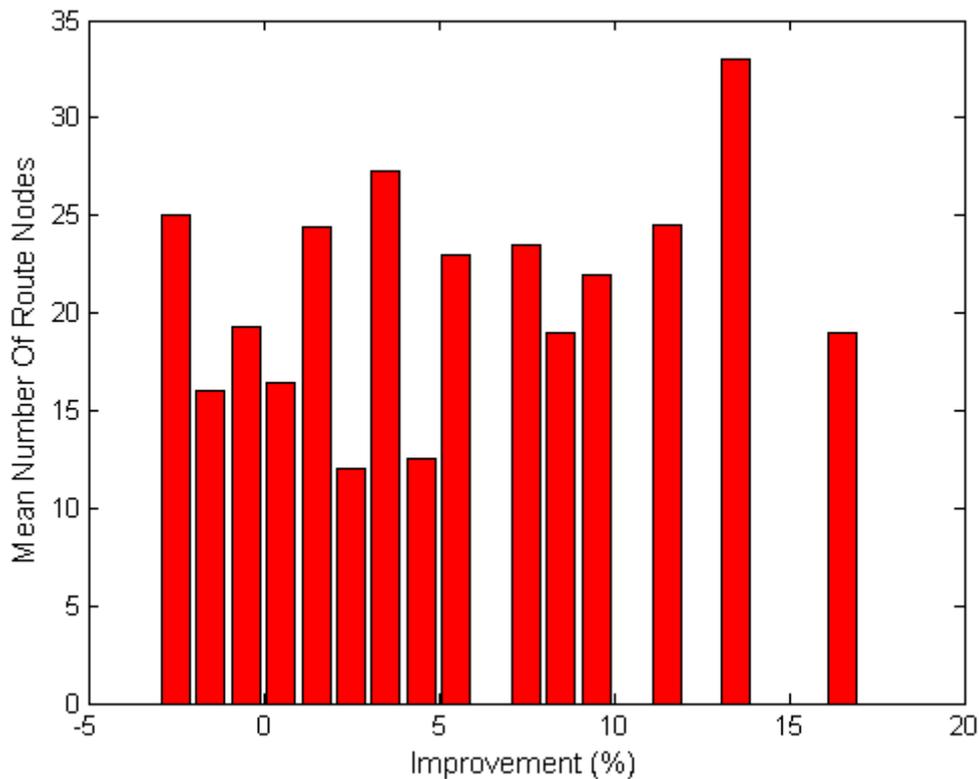


Figure 6.9: Mean number of route nodes. Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

### Computational time

The RouteAlg computational time consumption in regards to the test problems evaluated is summarised in Figure 6.10. Unfortunately, the computational time of the RouteAlg cannot be compared against Vural (2007) because this measure has not been published in their work. The RouteAlg is proven to be computationally resource efficient for all test problems varying between 4 to 36 nodes, as the mean computational time consumption was 1.49s. Furthermore, 55.71% (39) of the routes were solved within the 0s  $\rightarrow$  1s band, which increased to 77.14% (54) with the 0s  $\rightarrow$  2s band. The RouteAlg consumed the most computational time 8.09s, for CMT11Y – Route: 3, which contained the second greatest number of nodes (33) compared to the other studied problems. Subsequently, the link between the number of nodes on a route and the computational time consumed by the RouteAlg is analysed, see Figure 6.7. There is a positive correlation between the computational time consumed by the RouteAlg and the number of route nodes. The computational time consumption of the RouteAlg remains constant in the 1  $\rightarrow$  10 nodes band, but subsequently the rate of growth begins to exponentially increase. Another noteworthy point relates to the widening of absolute values between routes with the same number of nodes, as the route size is

increased. This variation becomes stronger for routes in band  $20 \leq \rightarrow < 31$ , as illustrated in Figure 6.11. A potential cause may relate to the number of loops applied in the 2-opt/Or-opt method.

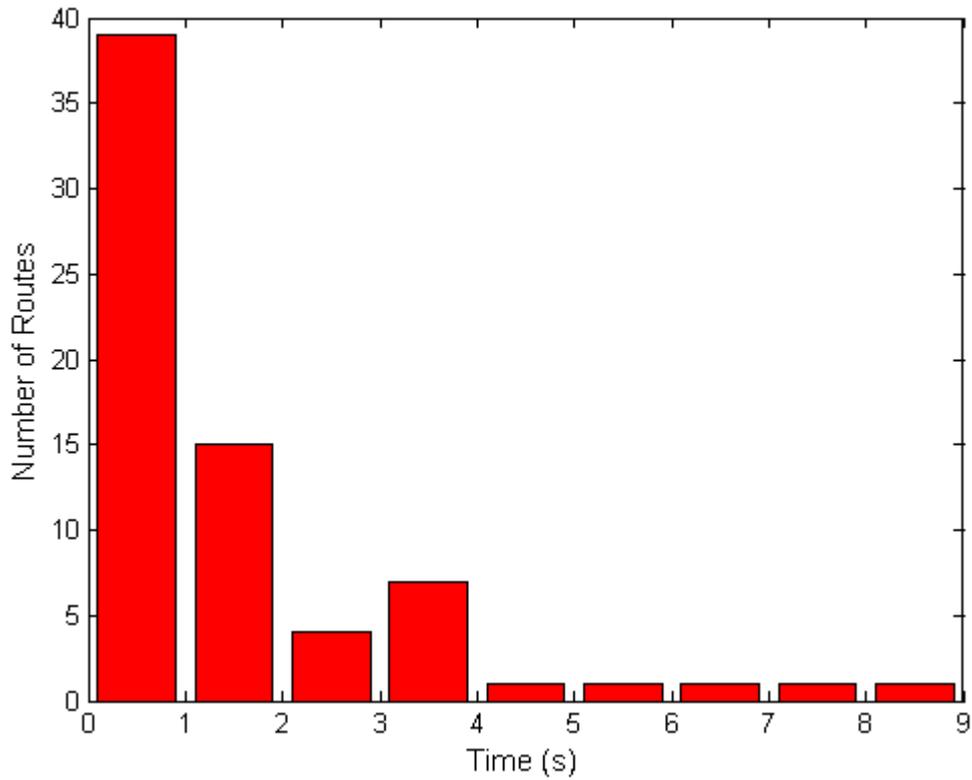


Figure 6.10: A bar graph showing the computational time consumption frequency of the RouteAlg determined routes. Band limit convention: band  $x \rightarrow y$  contains all values in interval  $x \leq i < y$ .

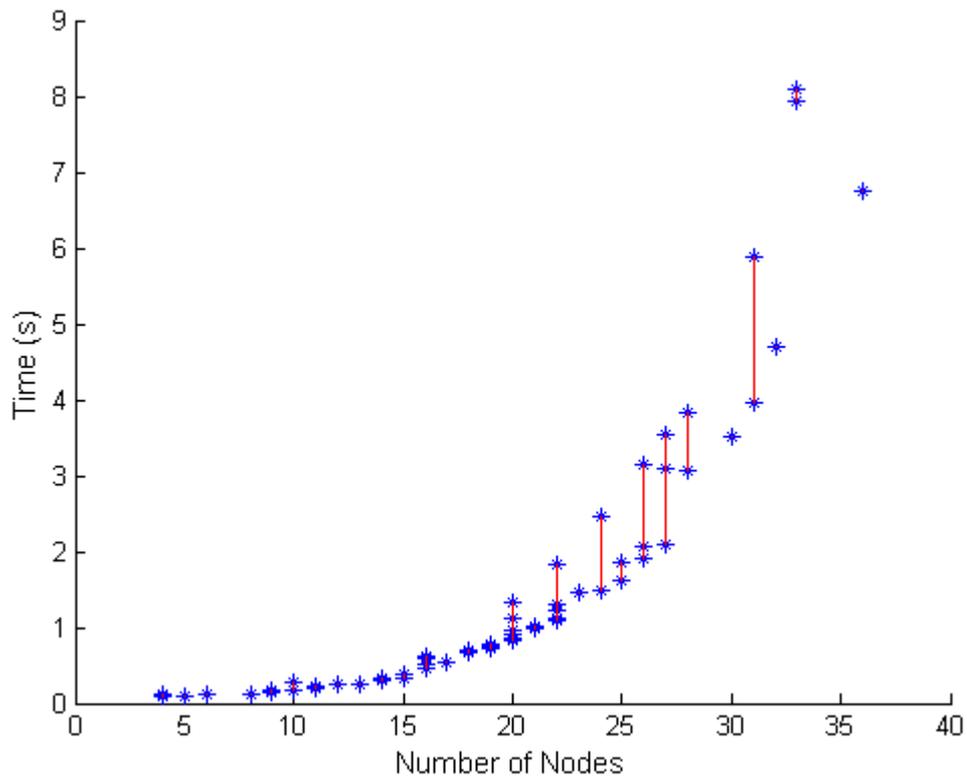


Figure 6.11: A absolute value chart showing the computation time consumption of the RouteAlg against the route size. Vertical Red Lines - Computational time consumption offset between different test problems of equal size.

The RouteAlg in summary has proven to be an effective method for solving the TSPSDP, as equal or improved quality solutions were found for 91.43% (64 routes), when compared against the results provided by Vural (2007). Another noteworthy contribution is its computational efficiency, as the RouteAlg solved 55.71% (39) of the instances within less than one second. In addition, the RouteAlg has proven to be powerful in determining feasible TSPSDP solutions. This is contributed to the fact that it features three procedures, which are able to induce such feasibility: Reverse procedure, EjRi method and 2-opt/Or-opt method. The EJRI method in particular has proven to be successful in this regard because all TSPSDP infeasible solutions became feasible following its application.

## 6.5 SDPmethod Validation Results

The proposed SDPmethod is evaluated in this section, in terms of minimising the following objectives: vehicle fleet size, total routing distance and workload variation. The latter objective refers to the difference between the longest and shortest routes encrypted by a solution and is considered here for the first time in the VRPSDP domain. A comparison is made between the best known solutions found in the literature for the Salhi and Nagy (1999) test problems with those generated by the SDPmethod. However, a limitation of the comparison is that the workload variation objective cannot be evaluated in the case of the best known solutions since their corresponding routes have not been published. In order to evaluate this objective, the workload variation has been manually computed for the solutions of Vural (2007) and Jun and Kim (2012), courtesy of the data personally provided by the authors.

### 6.5.1 Comparison with Best Known Results

The best known solutions for Salhi and Nagy (1999) test problems found in the VRPSDP domain are provided in Tables 6.3 and 6.4, along with those obtained by the introduced SDPmethod. The tables are identical, with the exception that the test problems provided in Table 6.4 have imposed a maximum route distance constraint. A detailed analysis of the results is provided here, in terms of the following objectives: the operated vehicle fleet size, the total routing distance and the workload variation, in the respective order. The latter objective refers to the maximum deviation among route distances. The solutions utilised for this experimental analysis are selected from the central area of the 1<sup>st</sup> order PF, which comprises a set of trade-offs between the employed objectives, the total routing distance and the workload variation, with practical utility, as opposed to solutions situated towards the PF extremes. The actual routes for the solutions published in Tables 6.3 and 6.4 are provided in the Appendix B, in Figures B.1 → B.28. The complete set of solutions on the 1<sup>st</sup> order PF is provided in Appendix B, in Tables B.1 → B.28.

Table 6.3 Result comparison against best solutions found for Salhi and Nagy (1999) test problems

<u>Test Problem</u>	<u>Best Known Results</u>			<u>SDPmethod Results</u>		
	<u>Total Routing Distance</u>	<u>Vehicle Fleet Size</u>	<u>Ref</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>	<u>Vehicle Fleet Size</u>
CMT1X	466.77	3	S	495.12	1.97	3
CMT1Y	466.77	3	S	484.18	5.45	3
CMT2X	668.77	6	W	766.78	22.58	6
CMT2Y	663.25	6	W	767	11.86	6
CMT3X	715.32	5	J	825.76	15.11	5
CMT3Y	745.46	4	W	800.12	9.8	4
CMT4X	852.46	7	S	1248.65	18.62	7
<b>CMT4Y</b>	<b>852.35</b>	<b>7</b>	<b>C</b>	<b>1088.4</b>	<b>33.51</b>	<b>6</b>
CMT5X	1029.25	10	S	1650.1	31.83	10
CMT5Y	1054.46	9	W	1289.38	34.29	9
CMT11X	833.92	4	S	1208.93	32.1	4
CMT11Y	830.39	4	W	1023.76	19.55	4
CMT12X	644.7	5	W	850.75	29.98	5
CMT12Y	662.99	5	J	777.21	15.23	5

Column 4 - References: J - Jun and Kim (2012), S - Subramanian et al (2010, 37), W - Wassan et al (2008) and C - Chen and Wu (2006). Red highlight - SDPmethod solution has a smaller vehicle fleet size in comparison to the best known solution.

Table 6.4 Result comparison against best solutions found for Salhi and Nagy (1999) test problems, with distance constraint

<u>Test Problem</u>	<u>Best Known Results</u>			<u>SDPmethod Results</u>		
	<u>Total Routing Distance</u>	<u>Vehicle Fleet Size</u>	<u>Ref</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>	<u>Vehicle Fleet Size</u>
CMT6X	471.89	3	W	480.57	2.15	3
CMT6Y	467.7	3	W	484.18	5.45	3
CMT7X	663.95	6	W	766.78	22.58	6
CMT7Y	662.5	6	W	845.82	13.04	6
CMT8X	720	5	M	854.65	2.21	5
<b>CMT8Y</b>	<b>721</b>	<b>5</b>	<b>M</b>	<b>820.5</b>	<b>3.79</b>	<b>4</b>
CMT9X	880.61	7	W	1200.82	12.74	7
<b>CMT9Y</b>	<b>886.84</b>	<b>7</b>	<b>W</b>	<b>1385.3</b>	<b>29.09</b>	<b>6</b>
CMT10X	1079.99	10	W	1650.1	31.83	10
<b>CMT10Y</b>	<b>1058.09</b>	<b>10</b>	<b>W</b>	<b>1289.38</b>	<b>34.29</b>	<b>9</b>
<b>CMT13X</b>	<b>858.48</b>	<b>5</b>	<b>W</b>	<b>1208.93</b>	<b>32.1</b>	<b>4</b>
CMT13Y	880.56	4	W	1027.69	13.12	4
CMT14X	644.7	5	W	850.75	29.98	5
<b>CMT14Y</b>	<b>659.52</b>	<b>6</b>	<b>W</b>	<b>847.93</b>	<b>20.39</b>	<b>5</b>

Column 4 - References: W - Wassan et al (2008) and M - Montane and Galvao (2006). Red highlight - SDPmethod solution has a smaller vehicle fleet size in comparison to the best known solution.

#### Vehicle Fleet Size

The SDPmethod solutions were identical to the lower bound vehicle fleet sizes defined in Table 6.2, which demonstrates the efficiency of the SDPmethod in determining solutions with the minimum fleet size. It is also noteworthy that the maximum route distance restriction imposed on the test problems in Table 6.4 had no bearing on the minimum vehicle fleet size because the identical fleet size was determined in Table 6.3.

The SDPmethod improved the best known solutions in terms of a reduction in the vehicle fleet size for 21.43% (6) of the test problems: CMT4Y, CMT8Y, CMT9Y, CMT10Y, CMT13X and CMT14Y, where one less vehicle was utilised. The majority of the improvements with respect to the vehicle fleet size were found in Table 6.4, where a maximum route distance limit was imposed on the test problems, with the exception of the CMT4Y test problem in Table 6.3. This suggests that the SDPmethod is more robust than its counterparts in finding solutions to problems where a maximum route distance constraint is imposed. This is contributed to the multi-objective nature of the SDPmethod, which aims to balance the workload variation, an item to be discussed shortly. In consequence, the SDPmethod on average was able to increase

the number of nodes assigned to each vehicle, which reduced the need to expand the vehicle fleet size.

#### Total Routing Distance

Initially, the routing solutions provided by the SDPmethod in Table 6.3 and 6.4 may appear uncompetitive compared to the best known. However, the SDPmethod solutions were selected from the central vicinity of the 1<sup>st</sup> order Pareto front (PF), in order to balance the trade-off between the two objectives. It is noteworthy that the SDPmethod did find solutions on the 1<sup>st</sup> order PF that were improved in terms of the total routing distance objective, than the solutions provided by the SDPmethod in Tables 6.3 and 6.4, as shown in the Appendix B, in Tables B.1 → B.28, but, these were worse in terms of the workload variation objective.

The SDPmethod found new best solutions, in terms of total routing distance for 10.71% (3) of the test problems: CMT1Y, CMT6X and CMT6Y, which were 462.22, 471.53 and 462.22 respectively. The solution routes are provided in the Appendix B, in Figures B29 → B31. Furthermore, the SDPmethod found the optimal solution for the CMT1X test problem defined by Subramanian et al (2010, 37) as 466.77, see Appendix B, in Figure 32.

The SDPmethod solutions were expected to have a higher total routing distance than the best known for the test problems where the vehicle fleet size had been reduced because the remaining routes have more nodes to service, resulting in longer routing distances. This type of solution may be welcomed by an operator, if the increase in the total routing distance is moderate because the vehicle fleet size represents the largest proportion of total costs.

#### Workload Variation

The workload variation objective was considered here for the first time in the VRPSDP domain. Unfortunately, a direct workload variation comparison between the SDPmethod solutions and the best known was not possible, as the routes for the latter have not been published. Subsequently, a direct comparison is made against the research works of Vural (2007) and Jun and Kim (2012). This comparison was made possible due to the support of the authors, which is profoundly appreciated. However, a limitation of the comparison is that the authors have not provided the workload difference for all test problems studied in this research.

### 6.5.2 Comparison with Vural (2007)

Table 6.5 presents the solutions obtained by Vural (2007) and the SDPmethod for a set of Salhi and Nagy (1999) test problems. It is used to evaluate the solution quality, in terms of the objectives considered here.

Table 6.5 Vural (2007) results comparison

<u>Test Problem</u>	<u>Vural (2007) Results</u>			<u>SDPmethod Results</u>		
	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>	<u>Vehicle Fleet Size</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>	<u>Vehicle Fleet Size</u>
CMT1X	484.22	91.47	4	495.12	1.97	3
CMT1Y	607	97.73	7	484.18	5.45	3
CMT2X	757.9	66.88	7	766.78	22.58	6
CMT2Y	725.78	39.29	6	767	11.86	6
CMT3X	774.74	66.65	5	825.76	15.11	5
CMT3Y	823.97	10.57	4	800.12	9.8	4
CMT4X	1103.59	33.93	7	1248.65	18.62	7
CMT4Y	1247.09	95.98	11	1088.4	33.51	6
CMT5X	1237.58	61	8	1650.1	31.83	10
CMT11X	1152.05	95.53	9	1208.93	32.1	4
CMT11Y	925.87	95.51	7	1023.76	19.55	4
CMT12X	802.56	51.98	6	850.75	29.98	5
CMT12Y	664.42	26.09	5	777.21	15.23	5

Column 3 - The maximum deviation between route distances found by Vural (2007), Column 6 - The maximum deviation between route distances found by the SDPmethod. Red highlight - SDPmethod solution has a smaller vehicle fleet size and/or total routing distance in comparison to the Vural (2007) solution.

#### Vehicle Fleet Size

The SDPmethod determined solutions with a reduced vehicle fleet size for 53.85% (7) of test problems, as shown in Table 6.5. The largest improvement was found for CMT4Y and CMT11X test problems, where a reduction of 5 vehicles was achieved. In contrast, Vural (2007) found an improved vehicle fleet size solution than SDPmethod for 7.69% (1) test problems, CMT5X. This solution operated two fewer vehicles than is capacity-wise feasible, which is defined as 10, in Table 6.2. Therefore, a discrepancy may exist in terms of how CMT5X test problem was generated by Vural (2007).

### Total Routing Distance

The SDPmethod solutions have not improved the total routing distances found by Vural (2007) for a substantial proportion of the test problems, although they are in the near vicinity (within a mean of 5.31%) for test problems: CMT1X, CMT2X, CMT2Y, CMT3X, CMT11X and CMT12X, as shown in Table 6.5. However, this was expected for the solutions with a reduced vehicle fleet size because the fewer routes may have to travel further distances to service the additional nodes that were incorporated. Nevertheless, the SDPmethod has obtained solutions with a reduced total routing distance for 23.08% (3) of test problems: CMT1Y, CMT3Y and CMT4Y. Although, contrary to the above, the solutions determined for CMT1Y and CMT4Y test problems had also operated a smaller vehicle fleet size.

### Workload Variation

The solutions obtained by the SDPmethod were considerably better in terms of workload variation for all test problems, as illustrated in Figure 6.12. Noticeably, the maximum workload deviation between route distances is significantly higher for Vural (2007) test problems, when compared against the offset achieved by the SDPmethod. For 38.46% (5) of test problems: CMT1X, CMT1Y, CMT4Y, CMT11X and CMT11Y, the workload deviation exceeded 90%, as show in Figure 6.8. This large variation implies that at least one vehicle in the respective test problems was severely underutilised. Whereas, the largest workload variation found by the SDPmethod was 33.51% for CMT4Y test problem, which in comparison is sizeably less than Vural (2007) maximum. The best workload variation result determined by Vural (2007) was 10.57% for CMT3Y test problem, which is in the near vicinity of the offset determined by the SDPmethod. However, the SDPmethod solution was also shorter in terms of total routing distance.

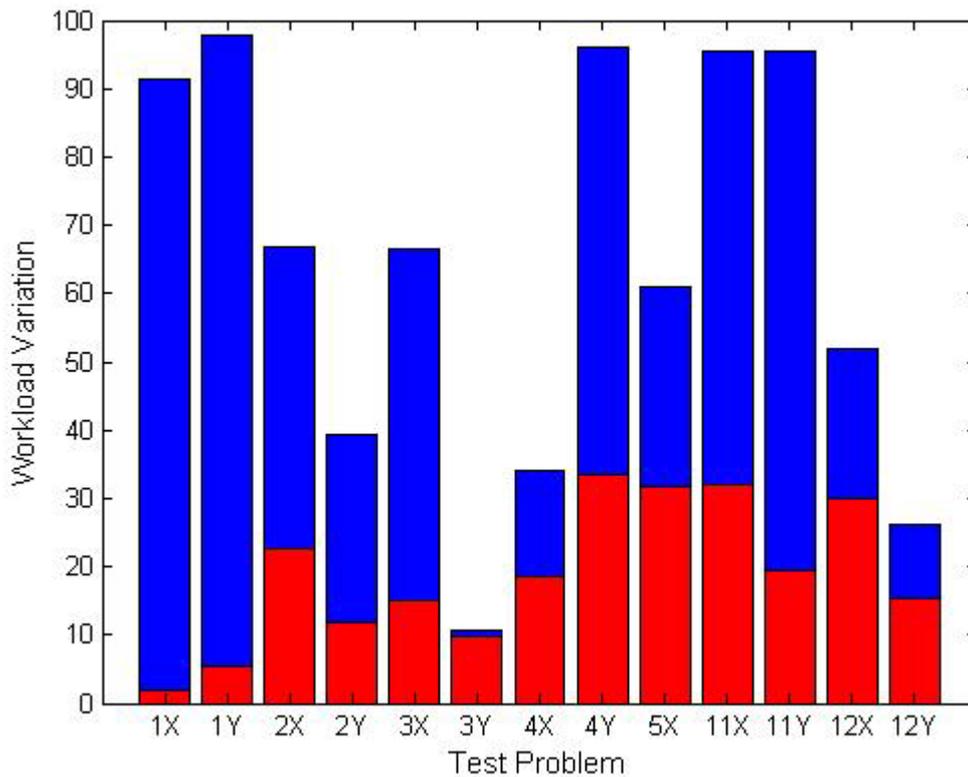


Figure 6.12 Workload variation comparisons with Vural (2007). Bars: Red - workload variation found by the SDPmethod and Blue - workload variation found by Vural (2007). Both coloured bars start on the horizontal axis.

### 6.5.3 Comparison with Jun and Kim (2012)

A direct result based comparison is not possible for all the test problems in Table 6.6 because only the instances: CMT3X, CMT3Y, CMT4Y and CMT5X have been constructed in an identical manner. The remaining test problems have applied a service time at the customer nodes, which was not considered here. Therefore, a direct comparison was made only in terms of the vehicle fleet size and total routing distance minimisation, whereas the entire test problem set was considered for the workload variation objective.

Table 6.6 Jun and Kim (2012) results comparison

<u>Test Problem</u>	<u>Jun and Kim (2012) Results</u>			<u>SDPmethod Results</u>		
	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>	<u>Vehicle Fleet Size</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>	<u>Vehicle Fleet Size</u>
CMT3X	715.32	74.03	5	825.76	15.11	5
CMT3Y	719.33	79.91	5	800.12	9.8	4
CMT4Y	847.58	65.09	7	1088.4	33.51	6
CMT5X	1036.36	78.65	10	1650.1	31.83	10
CMT6X	555.43	61.5	6	480.57	2.15	3
CMT6Y	555.43	61.47	6	484.18	5.45	3
CMT7X	901.22	35.48	11	766.78	22.58	6
CMT9X	1161.37	64.71	14	1200.82	12.74	7
CMT9Y	1161.37	64.71	14	1385.3	29.09	6
CMT10X	1392.36	86.13	18	1650.1	31.83	10
CMT13X	1549.79	80.86	11	1208.93	32.1	4
CMT14X	821.75	61.72	10	850.75	29.98	5
CMT14Y	821.75	61.69	10	847.93	20.39	5

Column 3 - The maximum deviation between route distances found by Jun and Kim (2012), Column 6 - The maximum deviation between route distances found by the SDPmethod. Red highlight - SDPmethod solution has a smaller vehicle fleet size and/or total routing distance in comparison to the Jun and Kim (2012) solution.

#### Vehicle Fleet Size

The SDPmethod was competitive in terms of the vehicle fleet size objective because the number of vehicles operated in each solution was either smaller or equal to that found by Jun and Kim (2012) for CMT3X, CMT3Y, CMT4Y and CMT5X test problems, as shown in Table 6.6. In particular, the vehicle fleet size was reduced by one for 50% (2) of the test problems: CMT3Y and CMT4Y.

#### Total Route Distance

Jun and Kim (2012) solutions found a lower total routing distance for the CMT3X, CMT3Y, CMT4Y and CMT5X test problems, when compared to the herein presented SDPmethod. However, longer route distances were implied for 50% (2) of the test problems: CMT3Y and CMT4Y because a smaller vehicle fleet size solution was found by the SDPmethod, as shown in Table 6.6.

### Workload Variation

The SDPmethod found solutions with substantially lower workload variation for all test problems, when comparing against Jun and Kim (2012), as shown in Figure 6.13, therefore indicating its competitiveness in minimising this objective. The success is further demonstrated by the fact that the minimum workload variation found by the SDPmethod was 2.15% for the CMT6X test problem, which was considerably less than the lowest 35.48% determined by Jun and Kim (2012) for the CMT7X test problem. Moreover, the largest workload variation found by the SDPmethod was 33.51% for the CMT4Y test problem, which was significantly less than the 86.13% found by Jun and Kim (2012) for the CMT10X test problem. Therefore, the solutions provided by Jun and Kim (2012) indicate that there is considerable inequality in terms of route distances, especially as for 92.31% (12) of the test problems studied, the workload offset exceeded 60%.

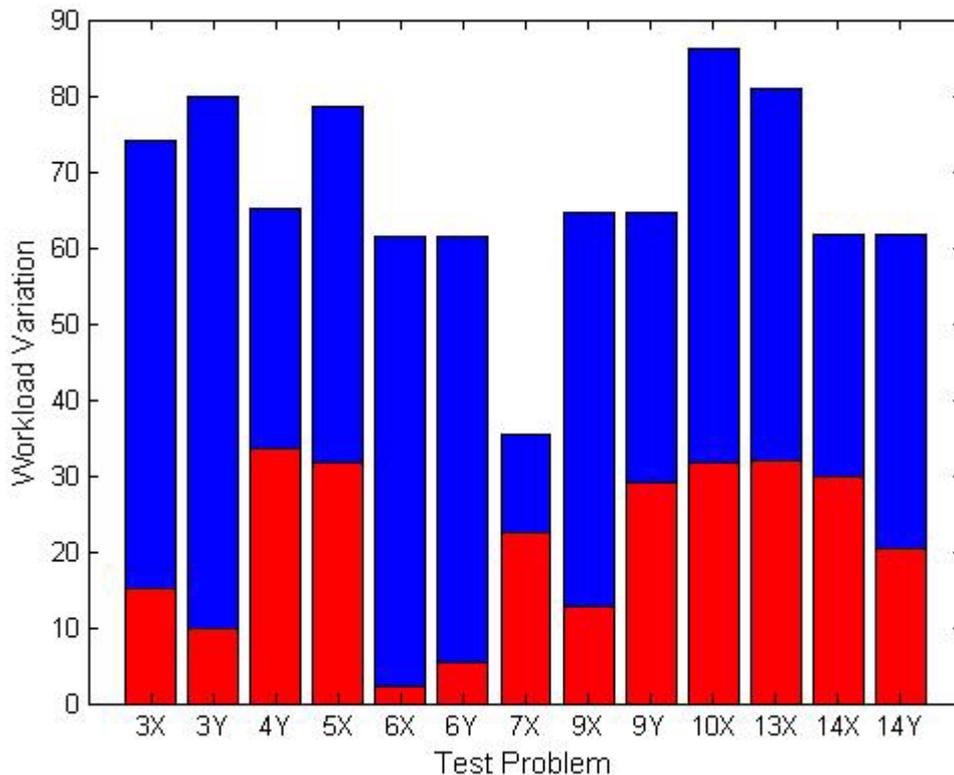


Figure 6.13 Workload variation comparisons with Jun and Kim (2012). Bars: Red - workload variation found by the SDPmethod and Blue - workload variation found by Jun and Kim (2012). Both coloured bars start on the horizontal axis.

The suggested SDPmethod has been demonstrated to be a competitive method in solving the VRPSDP, when compared to other methods from the domain. The SDPmethod has found solutions to all test problems in alignment with the minimum vehicle fleet sizes determined in

Table 6.2. In particular, the SDPmethod improved the vehicle fleet size solution quality for 21.43% (6), 53.85% (7) and 50% (2) comparable test problems, in comparison with the best known solutions, Vural (2007) and Jun and Kim (2012), respectively. Therefore, the SDPmethod has proven to be extremely effective in terms of minimising this objective. Another issue of importance relates to the minimisation of the total routing distance objective, as the SDPmethod has found improved solutions in relation to the best known ones for 10.71% (3) of the test problems and one solution equal to the best known one. The SDPmethod solutions are also better than those found by Vural (2007), in terms of the total routing distance objective for 23.08% (3) test problems. In general, the SDPmethod has been able to generate solutions in the vicinity of the best known total routing distances, while providing competitive results in terms of workload difference in all test instances. Indeed, the SDPmethod solutions are better in terms of minimising the workload variation for all comparisons made with Vural (2007) and Jun and Kim (2012) results. The level of improvement in the majority of cases is substantial, therefore illustrating the effectiveness of the method in generating routes with similar workload variation.

## **6.6 Genetic Algorithm Analysis**

The genetic algorithm introduced in Phase 3 of the SDPmethod will be analysed here as it presents a key contribution to this work. The following is an outline of the discussion. The experimental outcome used to dynamically determine the population size and mutation rate is analysed first. Secondly, a control mechanism enforced via switching thresholds is used to transition between different genetic operators. The effect these operators had on the search will be discussed here in greater detail. Thirdly, a cross analysis is performed using different sets of measurements: the Hamming distance and the average distance between the 1<sup>st</sup> order PF and the origin, as illustrated by Figures 6.10 and 6.11, respectively. The Hamming distance measures the level of population diversity at set intervals, while the distance between the 1<sup>st</sup> order PF and the origin provides insight on the accuracy improvement at every generation. Finally, the genetic algorithm is analysed in terms of the distribution of non-dominated individuals across the 1<sup>st</sup> order PF, as depicted in Figure 6.12. The aforementioned figures all relate to the CMT3X test problem, which will be the focus of this analysis.

### **6.6.1 Population size, maximum number of generation's and mutation rate setup**

A convenient value for the genetic algorithm population size is determined via a set of experiments conducted relative to various individual test problems. The results are presented

in the Appendix C, in Table C.1. The purpose of the statistical output analysis is to determine the population size that provides the highest level of search accuracy within 100 generations. Table 6.7 describes the best results obtained over 10 runs for CMT1X test problem, which is taken from Appendix C, Table C.1. It is evident that a relationship exists between the search accuracy and the implemented population size, given that the distance between the 1<sup>st</sup> order PF and the origin of the search space axes continues to decrease as the population size increases, until the saturation point is reached at population size 500. Therefore, a population size of 500 is used for the genetic algorithm. The other algorithm parameter configured via this line of experiments is the maximum number of search generations. The latter is set to 100 because the best results (in terms of minimising the distance between the first order PF and the search space axes origin) found for each considered population size are between generations 87 and 93, as illustrated in Table 6.7. With respect to the remaining test problems in Appendix C, in Table C.1, a sufficiently similar setup is found to that of CMT1X. The saturation point of the population size (i.e., the value where the distance between the first order PF and the search space axes origin stops decreasing) is less than a hundred individuals, therefore it is reasonable to consider a population size of 500 for all test problems.

Table 6.7 GA Parameter Setting

<b>Pop Size</b>	<b>Avg Dis</b>	<b>Generation</b>
100	0.31	89
200	0.29	91
300	0.27	93
400	0.26	90
500	0.25	87
600	0.25	88

Column 1 - Number of individuals in the population, Column 2 - the average distance between all individuals on the 1<sup>st</sup> order Pareto front and the origin and Column 3 - the generation at which the minimum average distance was found.

A sensitivity analysis for mutation probability determination was carried out, as shown in Table 6.8, for three arbitrarily chosen test problems: CMT1X, CMT3X and CMT5X. The best accuracy in each case is highlighted in bold. The genetic algorithm population size is set to the previously configured value of 500 individuals for the purpose of the mutation sensitivity analysis. Since there were no noteworthy differences in the recorded values for the monitored accuracy output, the chosen value for mutation probability is 0.2, as the average accuracy of the individuals reached peak values for this particular entry.

Table 6.8 Mutation Rate Tuning

<u>Test Problem</u>	<u>Mutation Probability (%)</u>	<u>Accuracy</u>	<u>Generation</u>
CMT1X	0.05	0.44	93
	0.10	0.36	91
	0.15	0.34	88
	<b>0.20</b>	<b>0.31</b>	<b>87</b>
	0.25	0.35	84
CMT3X	0.05	0.75	97
	0.10	0.65	94
	0.15	0.59	93
	<b>0.20</b>	<b>0.5</b>	<b>93</b>
	0.25	0.55	96
CMT5X	0.05	0.98	98
	0.10	0.96	92
	0.15	0.94	94
	<b>0.20</b>	<b>0.92</b>	<b>90</b>
	0.25	0.94	96

Column 1 - Test problem, Column 2 - Mutation rate, Column 3 - Average distance between the 1<sup>st</sup> order Pareto front and the origin of the axes and Column 4 - Generation when the minimum average distance between the 1<sup>st</sup> order Pareto front and the origin of the axes was found.

### 6.6.2 Review of the proposed Multi Objective Genetic Algorithm

The search impact of the genetic operator control mechanism enforced via switching thresholds will be discussed here by performing a cross analysis using different sets of measurements: the Hamming distance and the average distance between the 1<sup>st</sup> order PF and the origin, as illustrated by Figures 6.14 and 6.15, respectively.

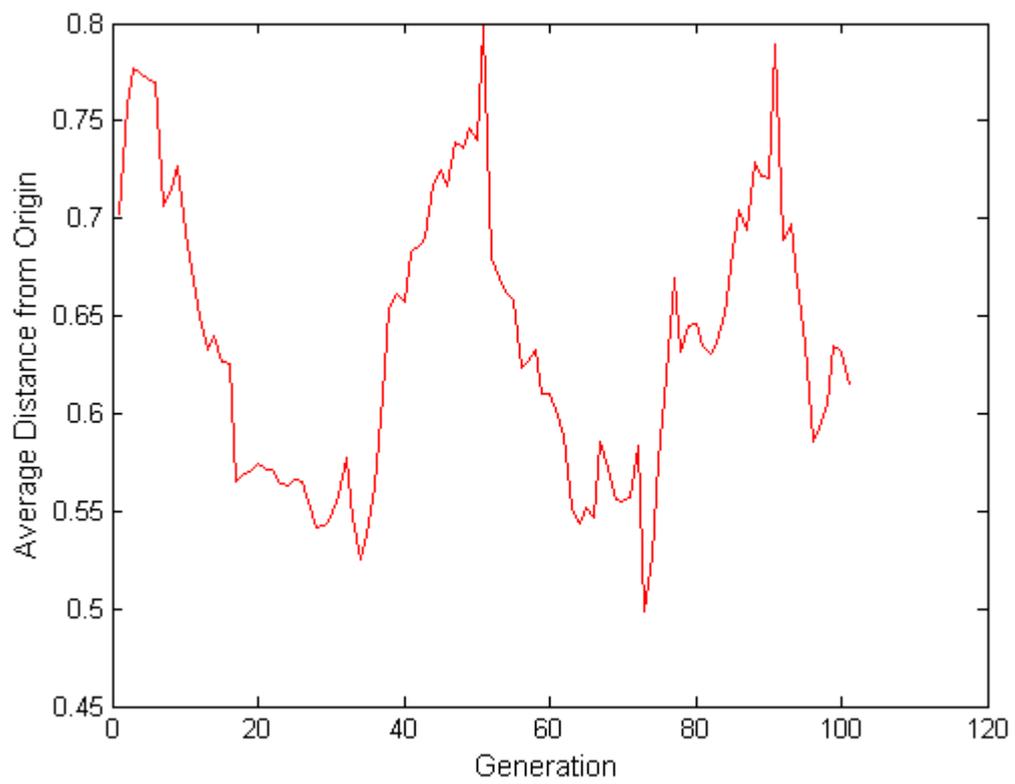


Figure 6.14 The average distance between the 1<sup>st</sup> order Pareto Front and the origin. Line: Red -  
The average distance between the 1<sup>st</sup> order PF and the origin.

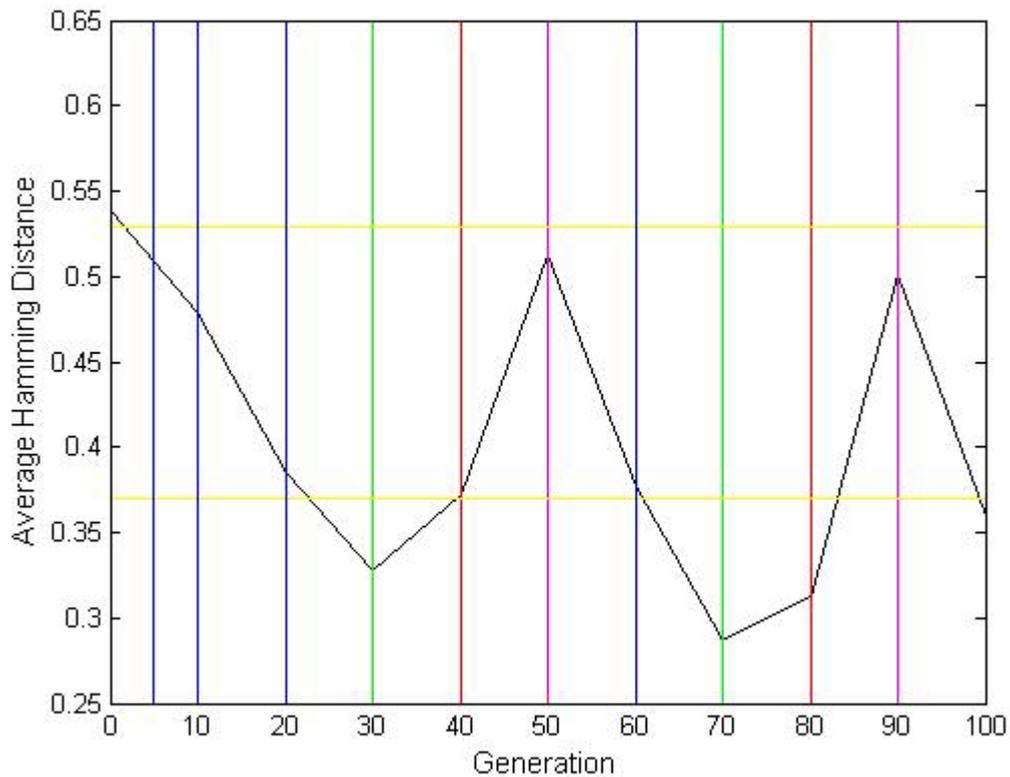


Figure 6.15 The average population Hamming Distance. Vertical Lines: Blue - 2-cut point crossover operator is introduced, Green – gene level crossover operator is introduced, Red - mutation operator is introduced and Purple - decimation, snap-shot and 2-cut point crossover operators are introduced in the respective order. Horizontal Lines: Yellow lower - the minimum acceptable level of population diversity required in the search and Yellow upper - defines a sufficient level of population diversity.

### Cut Point Crossover

The 2-cut point crossover operator was applied in conjunction with an elite selection strategy for reproduction and reinsertion, in order to increase the level of accuracy in the search. This operator was dynamically applied, depending on the average Hamming distance of the population. As the population diversity was above the considered threshold level, the 2-cut point crossover operator was applied between generations 6 → 30, 51 → 70 and 91 → 100, as shown in the Figure 6.14. The implementation of the operator corresponded to the sizeable increase in the search accuracy, as shown in Figure 6.15. Over the generations, a gradual decline in the distance between the 1<sup>st</sup> order PF and the origin is noticeable, as shown in Figure 6.11 at generations 34 and 73.

A shortcoming of the 2-cut point crossover operator is that the population becomes dominated by elite individuals, which eventually resulted in the search becoming trapped in a particular area of the space. The population diversity had fallen below the considered threshold level at generations 23, 61 and 99, Figure 6.14, which allowed for little or no additional improvement in terms of accuracy, Figure 6.15. Therefore, to prevent this situation from persisting, a control mechanism enforced via switching thresholds was used to change the genetic operator.

The link between accuracy and the population diversity is experimentally illustrated by the fact that it was necessary to apply the 2-cut point crossover for a greater number of generations in the initial stages of the search compared to the later. This is a likely consequence of the initial population being randomly generated and therefore the level of diversity was found to be the greatest at this point.

#### Gene Level Crossover

The gene level crossover operator was the first genetic operator to be applied, in order to regain the population diversity to the required threshold level. This operator was applied in conjunction with a diversity friendly reproduction and reinsertion selection scheme, in order to effectively induce diversity. As the measured average population diversity dropped below the considered threshold at generations 31 → 40 and 71 → 80, gene level crossover operator was applied to compensate for that effect, as shown in Figure 6.14. However, this was followed by a limited improvement in population diversity, which may be attributed to the exchange of genetic material between similar individuals. This issue was probably further exaggerated by the crossover process being confined at a sub-chromosome level, probably implying a limited number of genes were available. On a different issue of importance, the distance between the 1<sup>st</sup> order PF and the origin increased during the application of the gene level crossover operator, as illustrated in Figure 6.15. This tendency was expected as a greater number of exchanges between individuals were likely, which was possibly exaggerated by the diversity friendly selection scheme.

#### Mutation

The mutation operator was applied to induce population diversity to the considered threshold level, should the gene level crossover operator fail in this respect. In case of the population being dominated by elite individuals, this operator was capable of introducing new gene

values into the population from the alleles set, which the gene level crossover operator may have not had access to. The mutation operator was dynamically applied between generations 41 → 50 and 81 → 90, as shown in Figure 6.14, due to the fact that the considered threshold level for population diversity had not been achieved by the gene level crossover operator. The mutation operator led to an increase in the level of diversity at a higher rate than the one obtained by applying the gene level crossover operator. The introduction of gene values was likely to have led to the re-insurgence of diversity. However, despite the sizeable increase in the population diversity, results show that it remained slightly below the considered threshold level. It is also noteworthy that the level of accuracy in the search deteriorated in the generations when the mutation operator was applied, as shown in Figure 6.15. This effect was most likely amplified by the fact that the implemented selection scheme was diversity friendly.

#### Decimation and Snapshot operators

The most powerful diversity friendly genetic operators at the use of the search were the decimation and snap-shot. The decimation operator deleted 10% of the individuals in the population and the snap-shot operator filled the deficit with individuals from a past generation, in order to generate additional population diversity and to maintain a link with the gained accuracy. The operators were applied following the failure of the mutation operator in increasing the population diversity to the considered threshold level. As the measured average population diversity was below the considered threshold at generations 50 and 90, the decimation and snap-shot operators were applied, as shown in Figure 6.14. Following the application of the decimation and snapshot operators, the 2-cut point crossover was employed, given the fact that the measured level of population diversity was the highest at those points, using the applied portfolio of genetic operators.

#### 6.6.3 Distribution of non-dominated individuals

Figure 6.16 defines the non-dominated solutions that have been determined during the search for the CMT3X test problem. The set contains solutions that are evenly distributed along the entire spread of the 1<sup>st</sup> order PF, representing feasible trade-offs between the two considered objectives namely: the total routing distances and the workload variation. This indicates that the proposed SDPmethod has been successful in solving the multi-objective problem studied in this work, given that most of the generated solutions were located in the central area of the PF. Indeed, there are few extreme points on the 1<sup>st</sup> order PF, suggesting that the employed operators have been successful in directing the search towards the central area of the PF. It is

also noteworthy that the even distribution of individuals on the 1<sup>st</sup> order PF is a consequence of the fitness formula 5.2, illustrated in chapter 5, which promotes the search space exploration around the non-dominated individuals and especially those where the search had previously been limited.

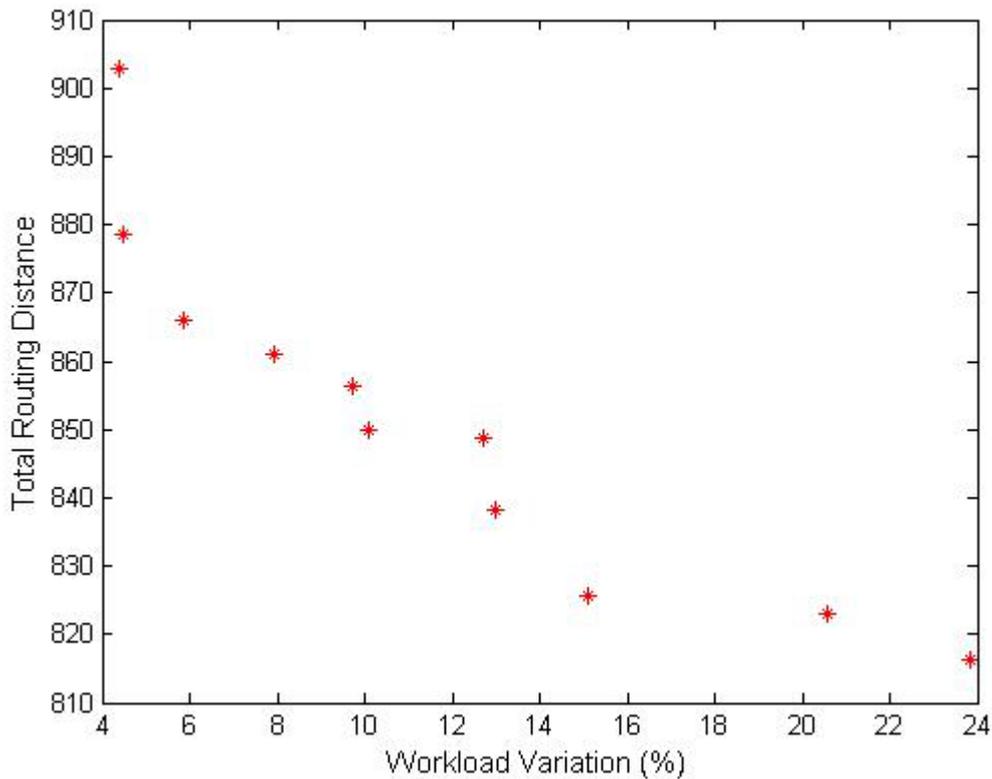


Figure 6.16 Non-dominated individuals found during the search. Red Asterisk - the objective qualities of a non-dominated solution that has been determined during the search.

This section has demonstrated the efficiency of the genetic algorithm in guiding the search towards the vicinity containing the global solutions, whilst preventing it from becoming trapped in a local optimum. For the majority of the search, the level of population diversity is within the defined thresholds. Furthermore, the genetic operators combined with the selection mechanism have proven successful in regaining the population diversity to the considered threshold level in the situations where greater diversity is required, generations 23 → 40 and 61 → 83. The even distribution of individuals on the 1<sup>st</sup> order PF reveals the success of the genetic algorithm in generating a variety of tradeoffs between the considered objectives, which is similar to the findings of Deb et al. (2002) for a different optimisation problem. As this analysis shows, the experimental results obtained and recorder herein support the intuitive expectations described and conceptually justified in Chapter 5.

This chapter has evaluated the proposed RouteAlg and SDPmethod using the widely benchmarked Salhi and Nagy (1999) test problems. The RouteAlg has proven to be a competitive routing heuristic for the TSPSDP because high quality solutions are obtained using low amounts of computational effort. The solution quality for 93.02% (80) routes studied matched or improved the results provided by Vural (2007). In addition, 64% (55) of the instances are solved in less than one second. The main contribution of this research work is the SDPmethod, a solution method for the multi-objective VRPSDP, which aims to minimise the following objectives: operated vehicle fleet size, total routing distance and the workload variation. The SDPmethod found solutions to all test problems in alignment with the lower bound vehicle fleet sizes determined in Table 6.2. Furthermore, the SDPmethod improved the vehicle fleet size solution quality for 21.43% (6) of test problems, in comparison with the best known solutions in the domain. In addition, the SDPmethod has found improved total routing distance solutions in relation to the best known ones for 10.71% (3) of the test problems and one solution equal to the best known one. The SDPmethod solutions are better in terms of minimising the workload variation for all comparisons made with Vural (2007) and Jun and Kim (2012) results. The level of improvement in the majority of cases is substantial, therefore illustrating the effectiveness of the method in generating routes with similar workload variation.

## 7. CONCLUSIONS

### 7.1 Problem description

The thesis addresses the delivery and pickup transportation problem (DPP), an extension of the vehicle routing problem (VRP), where the delivery and pickup demands may occupy the same route. This problem has been formulated here as the Vehicle Routing Problem with Simultaneous Delivery and Pickup (VRPSDP), which requires the concurrent service of delivery and pickup demands at a customer location, in the respective order of service. A new multi-objective VRPSDP is tackled here, which requires the minimisation of the following objectives: operated vehicle fleet size, total routing distance and the workload variation (maximum variation between route distances).

### 7.2 Problem importance

The theoretical interest in designing effective methods for the DPP has been fuelled partially by the growing importance of reverse logistics (Dethloff, 2001) and by the complex nature of the problem, arising from the difficulty of managing the fluctuating capacities at the route nodes (Parragh et al., 2008). The significance is further supported by the fact that the VRPSDP formulation provides commercial benefits in terms of lower routing costs for the service provider and less the handling effort for the recipient, when compared against other formulation types (Parragh et al., 2008). The current interest in the VRPSDP is high as demonstrated by the amount of literature works that have been published within the past decade (Wang and Chen, 2012).

### 7.3 Proposed methods

This thesis presents a new methodology to solve the VRPSDP encompassing two components: SDPmethod and RouteAlg. The main contribution of this research work is the solution to solve the multi-objective VRPSDP, herein termed SDPmethod. For the first time in the VRPSDP domain, the SDPmethod has attempted to identify a partial solution in the vicinity of the near-optimal solution and then focused the computational effort on solving the sub-problem, which is deemed more difficult because the assignment decisions are less intuitive. Another noteworthy contribution is the routing heuristic, RouteAlg, proposed to solve the travelling salesman problem simultaneous delivery and pickup (TSPSDP), which is a sub-problem of the VRPSDP. Initially, the RouteAlg invests computational effort in guiding the search towards a TSPSDP feasible space, before spending additional resources on optimising the solution.

#### **7.4 Introduced benefits**

The implications of this research work are substantial. The SDPmethod can generate a range of non-dominated solutions for the multi-objective VRPSDP, representing various trade-offs between the considered objectives. The non-dominated solutions allow users to make a selection based on their specific needs, which narrows the gap between theoretical study and practical implementation. Moreover, the SDPmethod employs a workload variation objective, which is considered here for the first time in the VRPSDP domain. The purpose of this objective is to encourage the generation of routes of similar lengths, which is an important consideration in the practical domain. For instance, a balanced workload between drivers is likely to reduce the employee turnover rate because workers are no longer disgruntled about receiving dissimilar therefore unfair amounts of work. Also, the compliance visibility with respect to the host country's driving hours regulations becomes easier to manage with a workload balanced solution. Additionally, the SDPmethod is of a new design, which protects potentially valuable building blocks from modification during the solutions' evolution. The benefit of the proposed design is that computational effort can be targeted towards less intuitive assignments, which should hopefully increase the computational efficiency of the SDPmethod. This design represents a complete transformation of the existing approach, found extensively in the VRPSDP domain, where the initialisation and modification of a solution are separate steps within the optimisation phase. Moreover, the solution design has a practical importance on problems where a client may want to achieve some efficiency gains (e.g. operate a reduced vehicle fleet size), without jeopardising their entire solution design, which is possible with the use of building blocks.

The RouteAlg is a powerful heuristic for generating high quality feasible solutions for the TSPSPD. The RouteAlg design focuses on introducing TSPSPD feasible solution, therefore increasing the probability that a near-optimal solution can be found because it is located in the feasible area of the search space.

#### **7.5 Research objectives and contributions**

To address the research objectives, this thesis has introduced a new theoretical design, the SDPmethod, to solve the multi-objective VRPSDP. The method consists of three phases: the first phase constructs a set of diverse partial solutions, where at least one is expected to be in the vicinity of the near-optimal solution, the second phase determines assignment possibilities for each sub-problem and finally the third phase uses a parallel genetic algorithm to solve the

sub-problems. This is the first time in the VRPSDP domain that a parallel genetic algorithm has been presented, which has independently evolved a set of diverse populations. To enhance the performance of the genetic algorithm, the following tools have been introduced: a new genetic operator switching mechanism enforced via diversity thresholds and a customised fitness computational formula. In addition, a previously neglected or inaccessible tool to measure the accuracy in the search is proposed, which calculates the distance between the 1<sup>st</sup> order Pareto front and the origin. The SDPmethod is proficient as it can generate a wide range of good solutions for the multi-objective problem, for which a workload variation objective is new to the domain. The aforementioned contributions have all been demonstrated to be successful at addressing the objectives, as illustrated by the results.

## **7.6 Results**

### **7.6.1 Experimental Comparisons**

The SDPmethod solutions are compared against the best known in the VRPSDP domain and the results are provided in Chapter 6, Tables 6.3 and 6.4. The SDPmethod has obtained an improvement over the best known solutions for a number of test problems, in terms of minimising the following objectives: operated vehicle fleet size and the total routing distance travelled. In particular, the SDPmethod found solutions for all test problems in alignment with the lower bound vehicle fleet sizes determined in Chapter 6, Table 6.2. In addition, the routing solutions found by the SDPmethod are mainly in the vicinity of the best known.

The SDPmethod has considered a new workload variation objective. Unfortunately, a direct comparison with the best known solutions is not possible in relation to this objective because routing solutions have not been published. Instead, a direct comparison is made with Vural (2007) and Jun and Kim (2012) research works, which have provided this thesis with their routing information. The SDPmethod solutions obtained an improvement in the workload variation for all test problems and in most cases it is shown to be substantial, as illustrated in Chapter 6, Figures 6.8 and 6.9, respectively.

The RouteAlg solutions for the TSPSPD were benchmarked against the results from Vural (2007), see Appendix A, Tables A1-A13. The comparison was limited to Vural (2007) because this was the only available research work with published routing solutions. The RouteAlg obtained TSPSPD feasible routing solutions for all studied instances. In particular, the RouteAlg

results were equal or improved for the majority of studied instances, therefore underpinning the competitive nature of the routing heuristic.

#### 7.6.2 Impact of proposed design

The genetic algorithm helped generate a set of non-dominated solutions on the 1<sup>st</sup> order PF for the multi-objective VRPSDP tackled here, as illustrated in Table 6, Figure 6.12. The majority of non-dominated individuals were evenly distributed along the centre vicinity of the 1<sup>st</sup> order PF, indicating the effectiveness of the genetic algorithm in determining solutions with balanced tradeoffs between objective values.

#### 7.6.3 Impact of proposed tools

A new genetic operator switching logic controlled via diversity thresholds is introduced in the VRPSDP domain for the first time to effectively guide the search towards higher quality search space. This mechanism has demonstrated to be successful in inducing accuracy and diversity into the search at the required generations, as illustrated in Chapter 6, Figure 6.11. The mechanism has also prevented the search from becoming trapped at the local optimum, which is a major concern for all approximation methods.

This research for the first time in the VRPSDP domain has evaluated the performance of the genetic algorithm during the evolution by measuring the distance between the 1<sup>st</sup> order Pareto front and the origin. The introduced genetic algorithm has demonstrated to be an effective method in improving the level of accuracy in the search, as illustrated in Chapter 6, Figure 6.10.

Patelli (2011) fitness formula is enhanced to consider the estimated distance between the Pareto sets, therefore increasing the accuracy of the fitness computation. This extension led to an improvement in the pace of search accuracy because a rapid decrease in the distance between the 1<sup>st</sup> order Pareto front and the origin was demonstrated in Chapter 6, Figure 6.10 for a particular test problem.

### 7.7 Future Research

This research work has not considered time window restrictions at the nodes, which are encountered in the practical domain. Therefore, to bridge the gap between theoretical study and practical relevance, the SDP method should be modified to consider the VRPSDP with time

windows (VRPSDPTW). It is important to mention that this research work has introduced the EJRi method, a component of the RouteAlg, which rearranges the cycle nodes, in order to induce capacity and time window feasibility. The success of this method in terms of introducing capacity feasibility has been demonstrated here. However, a computational study is required in relation to evaluate the performance of the method on time windows.

Future research may investigate the SDPmethod solution quality for a relaxed version of the VRPSDP, where the simultaneous service restriction is loosely applied during the search. This formulation is likely to increase the flexibility of the SDPmethod, as vehicle loads can be better managed.

The underlining framework of the proposed genetic algorithm is universally applied by practitioners, (Gendreau and Potvin, 2010), therefore the genetic algorithm enhancements introduced here, for example, the genetic operator switching mechanism via population diversity thresholds, may be applied to other research works over a wide range of disciplines. On a different standpoint, future research may substitute the genetic algorithm introduced here with an evolutionary strategy (ES), in order to evaluate the effect a self adaptive algorithm configuration may have on the search. It is noteworthy that the proposed genetic operator switching mechanism enforced via population diversity thresholds has effectively managed the application of genetic operators during the search. It is therefore suggested that the search improvement provided by ES is likely to be limited. Although, the ES will substantially reduce the level of input required from the human practitioner, in order to configure parameter settings.

Future research work may want to apply the SDPmethod to other vehicle routing problem (VRP) variants. The SDPmethod without modification can be applied directly to solve the VRP by setting either the customer delivery or pickup demands to zero. In addition, the SDPmethod can be applied to the DPP with the vehicle routing problem with mixed backhauls (VRPMB) formulation, where the customer requires the service of a single demand type, Nagy and Salhi (2005). In particular, the SDPmethod should be applied to a multi-objective problem, as it is proven to be successful in this regard. On a different issue of importance, the 1<sup>st</sup> order PF solutions provided for Salhi and Nagy (1999) test problems in the Appendix B are benchmark results, which should be used by researchers to evaluate their own methods, in order to progress the quality of solution methods in the domain.

Additional research attention is required in the VRPSDP domain with regards to the identification of high quality solution traits in the initial solution construction phase, as their determination here has successfully led to the identification of high quality solutions.

The research was evaluated using the widely benchmark test problems of Salhi and Nagy (1999), which contained between 50 and 199 customers. Future research may want to evaluate the performance of the SDPmethod using a larger test problem size, for instance, on Montane and Galvao (2006) test problems comprising of up to 400 nodes.

## REFERENCES

- AI, T.H. and KACHITVICHYANUKUL, V. (2009) A Particle Swarm Optimisation for Vehicle Routing Problem with simultaneous pickup and delivery. *Computers and Operations Research*. Volume 36. Issue 5. p. 1693-1702.
- ANGELELLI, E. and MANSINI, R. (2002) *The Vehicle Routing Problem with Time Windows and Simultaneous Pick-up and Delivery, Quantitative Approaches to Distribution Logistics and Supply Chain Management*. Chapter 5, p. 249-267. Springer, Berlin-Heidelberg.
- APPLEGATE, D.L., BIXBY, R.E., CHVATAL, V. and COOK, W.J. (2006) *The Traveling Salesman Problem A Computational Study*. Princeton and Oxford: Princeton University Press.
- ASHLOCK D. (2005) *Evolutionary Computation for Modelling and Optimization*. Springer Science+Business Media, Inc., New York.
- BEASLEY, J.E. (1983) Route first-cluster second methods for vehicle routing. *Omega*. Volume 11. Issue 4. p. 403-408.
- BELLMAN, R. (1957) *Dynamic programming*. Princeton N.J.: Princeton University Press.
- BERBEGLIA, G., CORDEAU, J-F., GRIBKOVSKAIA, I. and LAPORTE, G. (2007) Static pickup and delivery problems: a classification scheme and survey. *TOP*. Volume. 15, Issue. 1. p. 1-31.
- BIANCHESSI, N. and RIGHINI, G. (2007) Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Computers & Operations Research*. Volume 34. Issue 2. p. 578–594.
- BRANDAO, J. (2006) A new tabu search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*. Volume 173. Issue 2. p. 540–555.
- BRAYSY, O. and GENDREAU, M. (2005a) Vehicle Routing Problem with Time Windows, Part 1: Route Construction and Local Search Algorithms. *Transportation Science*. Volume 39. Issue 1. p. 104-118.
- BRAYSY, O. and GENDREAU, M. (2005b) Vehicle Routing Problem with Time Windows, Part 2: Metaheuristics. *Transportation Science*. Volume 39. Issue 1, p. 119-139.

CATAY, B. (2010) A new saving-based ant algorithm for the vehicle routing problem with simultaneous pickup and delivery. *Expert systems with applications*. Volume 37. Issue 10. p. 6809-6817.

CHEN, J-F. and WU, T-H. (2006) Vehicle routing problem with simultaneous deliveries and pickups. *Journal of the Operational Research Society*. Volume 57. Issue 5. p. 579-587.

CHOPRA, S. and MEINDL, P. (2007) *Supply Chain Management – Strategy, Planning & Operations*. 3<sup>rd</sup> ed. Prentice Hall.

CLARKE, G. and WRIGHT, J.W. (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*. Volume 12. Issue. 4. p. 568-581.

CHRISTOFIDES, N., MINGOZZI, A. and TOTH, P. (1979) The vehicle routing problem. In: Christofides, N., Mingozi, P., Toth, P., and Sandi, C., (eds), *Combinatorial Optimisation*, Wiley, Chichester, pp. 315-338.

CHRISTOPHER, M. (2011) *Logistics & Supply Chain Management*. 4<sup>th</sup> ed. Financial Times Prentice Hall.

COELLO COELLO, C.A. and LAMONT, G.B. (2004) *Applications of Multi-Objective Evolutionary Algorithms*. *Advances In Natural Computation – Vol. 1*. World Scientific Publishing Co. Pte. Ltd.

COELLO COELLO, C.A., LAMONT G.B. and VAN VELDHUIZEN D.A. (2007) *Evolutionary Algorithms for Solving Multi-Objective Problems*. 2nd ed. Springer.

CRISPIM, J. and BRANDAO, J. (2005) Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. *Journal of the Operational Research Society*. Volume 56. Issue 11. p. 1296 - 1302.

DEB, K. (2001) *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Ltd, Wiltshire, GB.

DEB, K. (2009) *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Ltd, Wiltshire, GB.

DEB, K., PRATAP, A., AGARWAL, S. and MEYARIVAN, T. (2002) A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, Volume 6. Issue 2. p. 182-197.

DEIF, I and BODIN L. (1984) Extension of the Clark and Wright algorithm for solving the vehicle routing problem with backhauling. In: *Kidder A (ed). Proceedings of the Conference on Computer Software Uses in Transportation and Logistics Management*. Babson Park, MA, USA, p. 75–96.

DEJONG, K.A. (2006) *Evolutionary Computation – A Unified Approach*, The MIT Press, USA.

DELL'AMICO, M., RIGHINI, G. and SALANI, M. (2006) A Branch-and-Price Approach to the Vehicle Routing Problem with Simultaneous Distribution and Collection. *Transportation Science*. Vol. 40. Issue 2. p. 235-247.

DEPARTMENT FOR BUSINESS INNOVATION & SKILLS. (2012) *BIS Retail Strategy*. London. (URN 12/1197).

DEPARTMENT FOR COMMUNITIES AND LOCAL GOVERNMENT. (2013) *The Future of High Streets, Progress since the Portas Review*, London. (ISBN: 978-1-4098-3946-0).

DETHLOFF, J. (2001) Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up. *OR Spektrum*. Volume 23. Issue 1. p. 79-96.

DETHLOFF, J. (2002) Relation between vehicle routing problems: an insertion heuristic for the vehicle routing problem with simultaneous delivery and pick-up applied to the vehicle routing problem with backhauls. *Journal of the Operational Research Society*. Volume 53. Issue 1. p. 115 - 118.

DORIGO, M. and GAMBARDELLA, L.M. (1991) Ant colony system: A cooperative learning approach to the travelling salesman problem. *IEEE Transactions on Evolutionary Computation*. Volume 1. Issue 1. p. 53-66.

EIBEN A.E. and SMITH J.E. (2003) *Introduction to Evolutionary Computing*. Springer-Verlag Berlin Heidelberg, Germany.

ERBAO, C. and MINGYONG, L. (2010) An improved differential evolution algorithm for vehicle routing problem with simultaneous pickups and deliveries and time windows. *Engineering Applications of Artificial Intelligence*. Volume 23. No. 2. p. 188-195.

European Commission. (2012) *Road Transportation - A change of gear*. Luxembourg: Publications Office of the European Union.

FOGEL D.B. (2006) *Evolutionary Computation – Toward a New Philosophy of Machine Intelligence*. John Wiley & Sons, Inc., Hoboken, New Jersey.

FONSECA, C.M. and FLEMING, P.J. (1993) Genetic algorithms for multiobjective optimization: Formulation, discussion, and generalization. *In proceedings of the Fifth International Conference on Genetic Algorithms*. San Mateo, California, p. 416-423.

GAJPAL, Y. and ABAD, P. (2009) An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*. Vol. 36 No. 12. p. 3215-3223.

GANESH, K. and NARENDRAN, T. (2008) TASTE: a two-phase heuristic to solve a routing problem with simultaneous delivery and pick-up. *The International Journal of Advanced Manufacturing Technology*. Volume 37. Issue 11-12. p. 1221-1231.

GEN, M. and CHENG, R. (2002) *Genetic Algorithms & Engineering Optimization*. John Wiley & Sons, Inc., United States of America.

GENDREAU, M. and POTVIN, J.Y. (2010) *Handbook of Metaheuristics*. 2<sup>nd</sup> ed. Springer New York Dordrecht Heidelberg London.

GLOVER, F. (1986) Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*. Volume 13. Issue 5. p. 533-549.

GOETSCHALCKX, M. and JACOBS-BLECHA, C. (1989) The vehicle routing problem with backhauls. *European Journal of Operational Research*. Volume 42. Issue 1. p. 39–51.

GOLDBERG, D.E. (1989) *Genetic Algorithms in search Optimisation and Machine Learning*. Addison Wesley Longman, Inc., USA.

GOLDBERG, D.E. and DEB, K. (1991) A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. *Foundations of Genetic Algorithms*. Morgan Kaufmann Publishers, Inc, USA, p. 69-93.

GOLDEN, B., RAGHAVEN, S. and WASIL, E. (2008) *The Vehicle Routing Problem*. Springer Science+Business Media, LLC, New York.

GOKCE, E.I. (2004) *A revised ant colony system approach to vehicle routing problems*. Master's Thesis for Graduate School of Engineering and Natural Sciences. Turkey: Sabanci University.

GOKSAL, F.P., KARAOGLAN, I. and ALTIPARMAK, F. (2013) A hybrid discrete particle swarm optimisation for vehicle routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*. Volume 65. Issue 1. p. 39-53.

GOLDREICH, O. (2010) *P, NP, and NP-Completeness, The basics of computational complexity*. Cambridge University Press, New York, USA.

GUTIN, G., YEO, A. and ZVEROVICH, A. (2001) *Traveling Salesman Should not be Greedy: Domination Analysis of Greedy-Type Heuristics for the TSP*. BRICS Report Series, Department of Computer Science, University of Aarhus, Denmark.

GUTIN, G. and PUNNEN, A.P. (2004) *The Traveling Salesman Problem and Its Variations*. New York, Boston, Dordrecht, London, Moscow: Kluwer Academic Publishers.

HALSE, K. (1992) *Modeling and solving complex vehicle routing problems*. PhD thesis for Institute of Mathematical Statistics and Operations Research. Denmark: Technical University of Denmark.

HANSEN, P. and MLADENOVIC, N. (1997) Variable neighbourhood search. *Computers & Operations Research*. Volume 24. Issue 11. p. 1097-1100.

HOCHBAUM, D. (1997) *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, Boston, MA.

HOLLAND, J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, Michigan.

HORN, J., NAFPLOITIS, N. and GOLDBERG, D. (1994) A niched Pareto genetic algorithm for multi-objective optimization. *In Proceedings of the First IEEE Conference on Evolutionary Computation*. Piscataway, NJ, p. 82-87.

HOSNY, M.I. (2010) *Investigating Heuristic and Meta-Heuristic Algorithms for Solving Pickup and Delivery Problems*. PhD Thesis for School of Computer Science and Informatics. Cardiff: Cardiff University.

JUN, Y. and KIM, B-I. (2012) New best solutions to VRPSDP benchmark problems by a perturbation based algorithm. *Expert Systems with Applications*. Volume 39. Issue 5. p. 5641-5648.

KANTHAVEL, K., PRASAD, P.S.S. and VIGNESH, K.P. (2012) Optimisation of Vehicle Routing Problem with Simultaneous Delivery and Pickup using Nested Particle Swarm Optimisation. *European Journal of Scientific Research*. Volume 73. Issue 3. p. 331-337.

KARP, R.M. (1972) Reducibility among combinatorial problems. In Miller and Thatcher, *Complexity of Computer Computations*, Plenum Press, New York. p. 85-103.

KIRKPATRICK, S., GELATT, C.D. and VECCHI, M.P. (1983) Optimisation by Simulated Annealing. *Science*. Volume 20. Issue 4598. p. 671-680.

KNOWLES, J. and CORNE, D. (1999) The Pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. *Proceedings of the 1999 Congress on Evolutionary Computation*. Piscataway, NJ: IEEE Press, p. 98-105.

KNOWLES, J.D. and CORNE, D.W. (2000) Approximating the non-dominated front using the Pareto archived evolution strategy. *Evolutionary Computation Journal*, Volume 8. Issue 2. p. 149-172.

KOZA J.R. (1998) *Genetic Programming – On the Programming of Computers By Means of Natural Selection*. 6<sup>th</sup> ed. The MIT Press, Cambridge, Massachusetts, London, England.

LANGDON, W.B. and POLI, R. (1998) Fitness causes bloat: Mutation. *Genetic Programming Lecture Notes in Computer Science*. Volume 1391. p. 37-48.

LIN, S. (1965) Computer solutions of the travelling salesman problem. *Bell System Technical Journal* Volume 44. p. 2245-2269.

LIU, S-C. and CHUNG, C-H. (2009) A heuristic method for the vehicle routing problem with backhauls and inventory. *Journal of Intelligent Manufacturing*. Volume 20. Issue 1. p. 29-42.

MACEACHREN, A.M. (1985) Compactness of Geographic Shape: Comparison and Evaluation of Measures. *Swedish Society for Anthropology and Geography*, Volume 67. Issue 1. p. 53–67.

MESTER, D., BRAYSY, O. AND DULLAERT, W. (2007) A multi-parametric evolution strategies algorithm for vehicle routing problems. *Expert Systems with Applications*. Volume 32. Issue 2. p. 508-517.

- METROPOLIS, N., ROSENBLUTH, A., ROSENBLUTH, M., TELLER, E. and TELLER, J. (1953) Equation of state calculations by fast computing machines. *Journal of Chemical Physics*. Volume 21. Issue 6. p. 1087-1092.
- MIN, H. (1989) The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Research Part A*. Volume 23. Issue 5. p. 377-386.
- MONTANE, F.A.T. and GALVAO, R.D. (2002) Vehicle Routing Problems with Simultaneous Pick-up and Delivery Service. *OPSEARCH*. Volume 39. Issue 1. p. 19-33.
- MONTANE, F.A.T. and GALVAO, R.D. (2006) A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers & Operations Research*. Volume 33 No. 3. p. 595-619.
- MORGAN, M.J.W. (2008) *GAPS: a hybridised framework applied to vehicle routing problems*. PhD thesis for School of Computer Science. Cardiff: Cardiff University.
- NAGY, G. (1996) *Heuristic methods for the many-to-many location-routing problem*. Doctor of Philosophy, Management Mathematics Group, School of Mathematics and Statistics. Birmingham: University of Birmingham.
- NAJERA, A.G. (2010) *Multi-Objective Evolutionary Algorithms for Vehicle Routing Problems*. PhD thesis for School of Computer Science. Birmingham: University of Birmingham.
- NELLES, O. (2001) *Nonlinear Systems Identification*. Springer-Verlag Berlin Heidelberg Press.
- OR, I. (1976) *Travelling salesman-type combinatorial problems and their relation to the logistics of regional blood banking*. PhD thesis for Department of Industrial Engineering and Management Sciences. Evanston, IL: Northwestern University.
- OSYCZKA, A. and KUNDU, S. (1995) A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural and Multidisciplinary Optimisation*, Volume 10. Issue 2. p. 94-99.
- PARRAGH, S.N., DOERNER, K.F., and HARTL, R.F., (2008) A survey on pickup and delivery problems, *Journal für Betriebswirtschaft*. Volume 58. Issue 1. p 21-51.
- PATELLI, A. (2011) *Genetic Programming Techniques for Nonlinear Systems Identification*, PhD thesis for Systems Engineering, Gheorghe Asachi Technical University, Iasi, Romania.

POLI, R., LANGDON, W.B. and MCPHEE, N.F. (2008) *A field guide to genetic programming*.  
<http://www.gp-field-guide.org.uk>.

POTVIN, J-Y. and ROUSSEAU, J-M. (1995) An Exchange Heuristic for Routeing Problems with Time Windows. *Journal of the Operational Research Society*. Volume 46. Issue 12. p. 1433-1446.

POTVIN, J-Y., KERVAHUT, T., GARCIA, B-L. and ROUSSEAU, J-M. (1996) The vehicle routing problem with time windows Part 1: Tabu Search. *Inform Journal on Computing*. Volume 8. Issue 2. p. 158-164.

PWC STRATEGY & ECONOMICS. (2013) *The outlook for UK mail volumes to 2023*. London.

RAZALI, N.M. and GERAGHTY, J. (2011) Genetic Algorithm Performance with Different Selection Strategies in Solving TSP. In: *The 2011 International Conference of Computational Intelligence and Intelligent Systems*, 6-8 July, Imperial College, London.

REINELT, G. (1991) A travelling salesman problem library. *ORSA Journal on Computing*. Volume 3. Issue 4. p. 376-384.

RENNER, G. and EKART, A. (2003) Genetic algorithms in computer aided design. *Computer-Aided Design*, Volume 35. Issue 8. p. 709-726.

REYNOLDS, R.G. and SVERDLIK, W. (1994) Problem Solving Using Cultural Algorithms. *IEEE World Congress on Computational Intelligence*. Volume 2. p. 645-650.

ROPKE, S. and PISINGER, D. (2006) A unified heuristic for a large class of Vehicle Routing Problems with Backhauls. *European Journal of Operational Research*. Volume. 171. Issue. 3. p. 750-775.

ROYAL MAIL GROUP LIMITED. (2013) *Annual Report and Financial Statements 2012-13*. London.

RUSHTON, A., CROUCHER, P. and BAKER, P. (2010) *The Handbook Of Logistics & Distribution Management*. 4<sup>th</sup> ed. Kogan Page Limited.

SALHI, S. and NAGY, G. (1999) A Cluster Insertion Heuristic for Single and Multiple Depot Vehicle Routing Problems with Backhauling. *Journal of the Operational Research Society*. Volume 50. Issue 10. p. 1034-1042.

- SAVELSBERGH, M.W.P. (1985) Local search in routing problems with time windows. *Annals of Operations Research*. Volume 4. Issue 1. p. 285-305.
- SCHAFFER, J.D. (1984) *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms*. PhD thesis, Nashville, TN: Vanderbilt University.
- SPEARS, W.M. (1995) Adapting crossover in evolutionary algorithms. In: *4<sup>th</sup> Annual Conference on Evolutionary Programming*. Cambridge, MA: MIT Press, p. 367-384.
- SRINIVAS, N. and DEB, K. (1994) Multi-objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary Computation Journal*, Volume 2. Issue 3. p. 221-248.
- SUBRAMANIAN, A., DRUMMOND, L.M.A., BENTES, C., OCHI, L.S. and FARIAS, R. (2010a) A parallel heuristic for the Vehicle Routing Problem with Simultaneous Pickup and Delivery. *Computers and Operations Research*. Volume 37. Issue 11. p. 1899-1911.
- SUBRAMANIAN, A., OCHI, L.S. and UCHOA, E. (2010b) *New Lower Bounds for the Vehicle Routing Problem with Simultaneous Pickup and Delivery*. Technical Report 01/10, Universidade Federal Fluminense, Niteroi, Brazil.
- SUBRAMANIAN, A., UCHOA, E., PESSOA, A.A. and OCHI, L.S. (2011) Branch-and-cut with lazy separation for the vehicle routing problem with simultaneous pickup and delivery. *Operations Research Letters*. Volume 39. Issue 5. p. 338-341.
- SURAL, H. and BOOKBINDER, J.H. (2003) The Single-Vehicle Routing Problem with Unrestricted Backhauls. *Networks*. Volume 43. Issue 3. p. 127 - 136.
- 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*. Volume 31. Issue 2 p. 170 - 186.
- TASAN, A.S. and GEN, M. (2012) A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Computers and Industrial Engineering*. Volume 62. Issue 3. p. 755-761.
- TOTH, P. and VIGO, D. (1997) An exact algorithm for the vehicle routing problems with backhauls. *Transport Science*. Volume. 31, Issue. 4. p. 372–385.

TOTH, P. and VIGO, D. (2002) The vehicle routing problem, Cordeau, J-F., Desaulniers, J., Desrosiers, M., Solomon, M.M., and Soumis, F., (2001), *VRP with Time Windows*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, p. 157-193.

TOTH, P. and VIGO, D. (2002) *The vehicle routing problem*. SIAM, United States of America.

TRANSPORT STUDIES DEPARTMENT. (2010) *Freight Modal Choice Study: Addressable Markets, Department for Transport*. University of Westminster, London.

VAN VELDHUIZEN, D.A. and LAMONT, G.B. (2000) Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation*, Volume 8. Issue 2. p. 125-147.

VELDHUIZEN, D.V. (1999) *Multiobjective Evolutionary Algorithms: Classifications Analyses, and New Innovations*. PhD thesis, Dayton, OH: Air Force Institute of Technology. Technical Report No, AFIT/DS/ENG/99-01.

VURAL, A.V. (2003) *A GA based meta-heuristic for capacitated vehicle routing problem with simultaneous pick-up and deliveries*. Master's thesis for Graduate School of Engineering and Natural Sciences. Turkey: Sabanci University.

VURAL, A.V. (2007) *The vehicle routing problem with simultaneous pick-up and deliveries and a GRASP-GA based solution heuristic*. Degree of Doctor of Philosophy in Engineering for Department of Industrial and Systems Engineering. USA: Mississippi State University.

WANG, H-F. and CHEN, Y-Y. (2012) A genetic algorithm for the simultaneous delivery and pickup problems with time windows. *Computers and Industrial Engineering*. Volume 62. Issue 1. p. 84-95.

WASSAN, N.A., WASSAN, A.H. and NAGY, G. (2008) A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. *Journal of Combinatorial Optimisation*. Volume 15. Issue 4. p. 368-386.

XU, R. AND WUNSCH, D.C. (2009) *Clustering*. John Wiley & Sons, Inc., Hoboken, New Jersey.

ZACHARIADIS, E.E., TARANTILIS, C.D. and KIRANOUDIS, C.T. (2009) A hybrid metaheuristic algorithm for the vehicle routing problem with simultaneous delivery and pick-up service. *Expert Systems with Applications*. Volume 36. Issue 2. p. 1070.

ZHANG, T., CHAOVALITWONGSE, W.A. and ZHANG, Y. (2012) Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. *Computers and Operations Research*. Volume 39. Issue 10. p. 2277-2290.

ZITZLER, E. and THIELE, L. (1999) Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, Volume 3. Issue 4. p. 257-271.

## **APPENDIX A**

### **RouteAlg**

## Routing Results

Table A.1

<b>CMT1X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	149.87	149.17	0.47	0	0	0	129.96	129.96	0	0.31
2	197.97	189.97	4.04	0	50	0	150.69	150.69	0	0.52
3	246.39	216.29	12.22	0	57.89	0	187.57	187.57	0	0.74
4	16	16	0	0	0	0	16	16	0	0.08
Mean	152.56	142.86	4.18	0	26.97	0	121.06	121.06	0	0.41

Table A.1: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.2

<b>CMT1Y</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	204.26	204.26	0	0	38.1	0	193.41	193.41	0	1
2	247.42	238.37	3.66	0	0	0	197.19	196.14	-0.54	0.62
3	16.12	16.12	0	0	0	0	16.12	16.12	0	0.09
4	42.05	42.05	0	0	0	0	42.05	42.05	0	0.09
5	46.17	46.17	0	0	0	0	46.17	46.17	0	0.09
6	120.22	109.63	8.81	0	0	0	107.18	109.63	2.23	0.16
7	4.47	4.47	0	0	0	0	4.47	4.47	0	0.09
Mean	97.24	94.44	1.78	0	5.44	0	86.66	86.86	0.24	0.31

Table A.2: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.3

<b>CMT2X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	124.34	101.56	18.32	0	0	0	101.54	101.54	0	0.11
2	153.18	152.62	0.37	15.38	0	0	136.32	136	-0.24	0.26
3	156.04	147.67	5.36	0	53.33	0	144.49	147.05	1.74	0.34
4	87.57	87.57	0	0	0	0	67.49	67.49	0	0.17
5	111.24	101.26	8.97	0	0	0	99.55	103.74	4.04	0.16
6	164.05	164.05	0	0	76.47	0	151.81	151.81	0	0.53
7	50.28	50.28	0	0	0	0	50.28	50.28	0	0.1
Mean	120.96	115	4.72	2.2	18.54	0	107.35	108.27	0.79	0.24

Table A.3: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.4

<b>CMT2Y</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	151.71	135.52	10.67	41.67	25	0	127.83	127.83	0	0.24
2	203.23	170.59	16.06	6.25	68.75	0	151.44	148.47	-2	0.58
3	197.73	143.63	27.36	35.71	0	0	127.9	127.9	0	0.32
4	134.46	134.46	0	0	41.18	0	124.97	124.97	0	0.54
5	145.41	134.67	7.39	50	0	0	104.67	104.67	0	0.24
6	102.22	93.67	8.36	0	0	0	91.94	91.94	0	0.11
Mean	155.79	135.42	11.64	22.27	22.49	0	121.46	120.96	-0.33	0.34

Table A.4: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSDP infeasible, Column 6 - The percentage of TSPSDP infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSDP infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.5

<b>CMT3X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	305.09	279.88	8.26	0	60.87	0	226.03	226.03	0	1.45
2	227.82	201.5	11.55	0	80	0	179.58	180.62	0.58	1.62
3	142.09	139.76	1.64	0	0	0	138.61	138.61	0	1.12
4	201.98	190.63	5.62	0	40.91	0	154.1	154.1	0	1.84
5	75.37	75.37	0	0	0	0	75.37	75.37	0	0.11
Mean	190.47	177.43	5.41	0	36.36	0	154.74	154.95	0.12	1.23

Table A.5: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.6

<b>CMT3Y</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	270.32	241.47	10.67	7.69	61.54	0	208.91	235.16	11.16	3.15
2	226.48	226.48	0	0	80	0	186.83	193.96	3.68	3.51
3	267.96	235.85	11.98	66.67	0	0	207.19	207.19	0	1.5
4	199.15	199.15	0	0	0	0	187.09	187.66	0.3	0.95
Mean	240.98	225.74	5.66	18.59	35.39	0	197.51	205.99	3.79	2.28

Table A.6: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSDP infeasible, Column 6 - The percentage of TSPSDP infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSDP infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.7

<b>CMT4X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	162.53	151.73	6.64	0	0	0	146.25	151.27	3.32	0.84
2	252.15	239.89	4.86	44.44	11.11	0	201.19	203.92	1.34	3.08
3	138.93	136.75	1.57	0	86.36	0	132.92	147.49	9.88	1.08
4	145.23	137.62	5.24	0	0	0	134.6	134.6	0	0.86
5	174.71	174.71	0	0	74.07	0	163.23	169.61	3.76	2.1
6	147.5	140.94	4.45	0	0	0	133.02	137	2.91	0.37
7	178.12	173.9	2.37	0	0	0	157.45	159.5	1.29	0.78
Mean	171.31	165.08	3.59	6.35	24.51	0	152.67	157.63	3.21	1.3

Table A.7: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.8

<b>CMT4Y</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	210.98	204.36	3.14	3.7	70.37	0	186.01	196.84	5.5	2.09
2	10	10	0	0	0	0	10	10	0	0.09
3	249.04	249.04	0	3.57	75	0	224.87	224.87	0	3.83
4	220.99	213.71	3.29	0	75	0	197.94	196.74	-0.61	3.07
5	18.11	18.11	0	0	0	0	18.11	18.11	0	0.09
6	24.08	24.08	0	0	0	0	24.08	24.08	0	0.09
7	273.21	267.77	1.99	0	92.31	0	249.02	268.99	7.42	1.91
8	185.01	180.3	2.55	23.08	38.46	0	164.87	167.25	1.42	2.06
9	122.28	122.28	0	0	0	0	107.14	107.55	0.38	0.27
10	12.65	12.65	0	0	0	0	12.65	12.65	0	0.09
11	20	20	0	0	0	0	20	20	0	0.09
Mean	122.4	120.21	1	2.76	31.92	0	110.43	113.37	1.28	1.24

Table A.8: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.9

<b>CMT5X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	232.25	208.86	10.07	11.11	62.96	0	192.66	204.38	5.73	3.54
2	167.29	134.5	19.6	0	0	0	123.45	129.66	4.79	0.39
3	181.46	164.25	9.48	77.27	0	0	136.62	136.62	0	1.3
4	130.29	129.03	0.97	0	0	0	126.08	151.14	16.58	0.78
5	243.34	243.34	0	0	48	0	211.04	206.38	-2.26	1.85
6	165.01	150.11	9.03	0	0	0	136.99	145.31	5.73	0.75
7	205.5	181.85	11.51	0	30.43	0	161.52	181.79	11.15	1.47
8	93.61	83.35	10.96	0	0	0	82.3	82.3	0	0.15
Mean	177.34	161.91	8.95	11.05	17.67	0	146.33	154.7	5.22	1.28

Table A.9: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.10

<b>CMT11X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	363.91	323.49	11.11	0	0	0	306.5	309.82	1.07	7.93
2	350.23	343.37	1.96	0	0	0	316.09	321.15	1.58	5.89
3	357.83	353.76	1.14	0	0	0	296.07	305.51	3.09	4.7
4	14.14	14.14	0	0	0	0	14.14	14.14	0	0.09
5	92.97	92.97	0	0	0	0	92.97	92.97	0	0.09
6	58.51	52.16	10.85	0	0	0	52.16	52.16	0	0.22
7	16.56	16.56	0	0	0	0	16.56	16.56	0	0.1
8	17.9	17.07	4.64	0	0	0	17.07	17.07	0	0.11
9	22.66	22.66	0	0	0	0	22.66	22.66	0	0.1
Mean	144.29	137.35	5.42	0	0	0	126.02	128	0.64	2.14

Table A.10: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSDP infeasible, Column 6 - The percentage of TSPSDP infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSDP infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.11

<b>CMT11Y</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	350.74	326.91	6.79	0	0	0	315.22	316.43	0.38	6.76
2	341.58	313.52	8.21	0	0	0	294.79	296.03	0.42	3.96
3	227.63	225.42	0.97	0	0	0	183.63	211.82	13.31	8.09
4	76.2	74.96	1.63	0	0	0	72.46	72.46	0	0.54
5	14.99	14.99	0	0	0	0	14.99	14.99	0	0.1
6	58.51	52.16	10.85	0	0	0	52.16	52.16	0	0.2
7	14.14	14.14	0	0	0	0	14.14	14.14	0	0.09
Mean	154.83	146.01	4.06	0	0	0	135.34	139.72	2.02	2.82

Table A.11: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.12

<b>CMT12X</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	140.17	138.34	1.31	0	0	0	132.09	132.29	0.15	2.47
2	148.75	142.75	4.03	0	0	0	135.82	144	5.68	1.23
3	198.99	164.22	17.47	0	0	0	151.47	151.47	0	0.66
4	183.12	161.01	12.07	0	0	0	150.16	151.43	0.84	0.69
5	152.02	152.02	0	0	0	0	150.62	150.62	0	0.46
6	72.74	72.74	0	0	0	0	72.74	72.74	0	0.1
Mean	149.3	138.51	5.81	0	0	0	132.15	133.76	1.11	0.94

Table A.12: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSDP infeasible, Column 6 - The percentage of TSPSDP infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSDP infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.13

<b>CMT12Y</b>										
Route	NN Results	MNN Results	Difference (NN & MNN) [%]	MNN Infeasibility [%]	Reverse MNN Infeasibility [%]	EjRi Infeasibility [%]	RouteAlg Results	Vural (2007) Results	Difference (Vural (2007) & RouteAlg) [%]	Time [s]
1	117.8	117.6	0.17	0	0	0	108.78	115.54	5.85	1.32
2	140.6	138.86	1.24	0	0	0	133.98	145.91	8.18	0.72
3	146.21	131.64	9.97	0	0	0	125.6	135.81	7.52	1.01
4	157.01	150.25	4.31	0	0	0	147.18	149.34	1.45	1.11
5	135.16	124.13	8.16	0	0	0	118.01	117.82	-0.16	0.92
Mean	139.36	132.5	4.77	0	0	0	126.71	132.88	4.57	1.02

Table A.13: Column 1 - Route name, Column 2 - Routing distance determined using the Nearest Neighbourhood (NN) algorithm, Column 3 - Routing distance determined using the Modified Nearest Neighbourhood (MNN) algorithm, Column 4 – Percentage variation between the routing distance determined by the NN algorithm and the MNN algorithm, Column 5 - The percentage of nodes on the route determined by the MNN algorithm that are TSPSPD infeasible, Column 6 - The percentage of TSPSPD infeasible nodes on the reverse MNN algorithm route, Column 7 - The percentage of nodes on the route determined by the EjRi method that are TSPSPD infeasible, Column 8 - Routing distance determined using the 2-opt/Or-opt method, Column 9 - Routing distance determined by Vural (2007), Column 10 - Percentage difference between the routing distance determined by the 2-opt/Or-opt method and Vural (2007) and Column 11 - Computational time consumption of the RouteAlg.

Table A.14

<u>Dataset</u>	<u>Route</u>	<u>Distance</u>
CMT2Y	1	143.64
	2	225.15
CMT3Y	1	271.35
CMT4X	2	259.99
CMT4Y	1	212.96
	3	308.45
	8	198.1
CMT5X	1	246.18

Table A.14: Column 1 - Name of dataset, Column 2 - Routing name, Column 3 - The routing distance following the application of the EjRi method.

**Routing Information**

Figure A.1

<b><u>Route</u></b>	<b><u>CMT1X</u></b>																				
1	0	11	38	9	50	16	2	29	21	34	30	39	10	49	5	0					
2	0	47	4	18	13	41	40	19	42	44	45	33	15	37	17	12	46	0			
3	0	32	1	22	20	35	36	3	28	31	26	8	48	23	7	43	24	25	14	6	0
4	0	27	0																		

Figure A.1: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.2

<b><u>Route</u></b>	<b><u>CMT1Y</u></b>																						
1	0	11	38	5	49	9	50	16	29	21	34	30	10	39	33	45	15	44	42	37	17	4	0
2	0	47	18	41	19	40	13	14	24	43	7	23	6	48	26	8	32	0					
3	0	12	0																				
4	0	2	0																				
5	0	25	0																				
6	0	1	22	20	35	36	3	28	31	27	0												
7	0	46	0																				

Figure A.2: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.3

<u>Route</u>	<u>CMT2X</u>																		
1	0	38	65	66	11	59	8	0											
2	0	17	40	50	18	55	25	31	10	58	72	39	9	12	0				
3	0	51	16	63	1	43	42	64	41	56	23	49	24	3	44	32	0		
4	0	75	30	48	5	29	45	27	52	46	34	0							
5	0	68	2	28	22	61	21	74	4	67	26	0							
6	0	7	35	53	14	19	54	13	57	15	37	20	70	60	71	69	36	47	0
7	0	6	33	73	62	0													

Figure A.3: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.4

<u>Route</u>	<u>CMT2Y</u>																		
1	0	7	53	11	66	65	38	31	10	58	35	8	46	0					
2	0	17	12	72	39	9	32	44	50	25	55	18	24	49	51	6	26	0	
3	0	75	68	2	22	64	42	43	41	56	23	63	16	3	40	0			
4	0	67	34	52	27	13	57	15	37	20	70	60	71	69	36	47	74	30	0
5	0	33	1	73	62	28	61	21	48	5	29	45	4	0					
6	0	54	19	59	14	0													

Figure A.4: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.5

<u>Route</u>	<u>CMT3X</u>																							
1	0	50	33	81	51	9	35	71	65	66	20	32	90	63	64	49	36	46	47	19	11	62	10	52
	0																							
2	0	26	12	80	68	76	77	3	79	78	34	29	24	54	4	55	25	39	67	23	56	75	72	73
	21	40	0																					
3	0	89	6	96	59	99	5	84	17	45	8	82	48	7	88	31	70	30	1	69	27	28	53	0
4	0	94	95	97	92	98	37	100	91	16	86	38	44	14	42	43	15	57	41	22	74	2	58	0
5	0	13	87	93	85	61	60	83	18	0														

Figure A.5: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.6

<u>Route</u>	<u>CMT3Y</u>																							
1	0	69	1	50	33	81	51	9	71	35	34	78	79	3	77	76	68	80	29	24	55	25	67	23
	39	4	28	0																				
2	0	89	6	96	99	59	98	85	91	100	37	92	97	2	57	42	14	43	15	41	22	74	75	56
	72	73	21	54	12	26	40	0																
3	0	53	58	13	87	95	93	44	38	86	16	61	84	17	45	8	46	47	36	49	64	11	62	88
	52	0																						
4	0	94	5	60	83	18	7	82	48	19	63	90	32	66	65	20	30	70	10	31	27	0		

Figure A.6: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.7

<u>Route</u>	<u>CMT4X</u>																							
1	0	112	53	40	149	139	39	67	25	55	130	24	121	79	3	77	116	68	80	109	12	0		
2	0	132	51	120	9	103	66	20	128	131	32	90	63	126	108	10	62	107	11	64	49	143	36	47
	124	46	52	146	0																			
3	0	89	147	6	94	117	95	92	37	93	5	118	84	113	17	45	125	8	114	48	19	123	106	0
4	0	13	59	60	83	82	7	148	88	127	31	101	70	30	122	1	69	28	138	26	105	0		
5	0	58	137	87	97	96	104	99	98	85	91	100	42	57	145	41	22	133	23	56	75	74	73	72
	110	4	54	150	0																			
6	0	134	29	129	78	34	135	35	136	65	71	81	33	102	50	76	0							
7	0	27	111	21	115	2	144	15	43	142	14	119	44	38	140	86	141	16	61	18	0			

Figure A.7: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.8

<b>Route</b>	<b>CMT4Y</b>																							
1	0	6	59	96	93	98	37	100	91	85	61	60	18	83	46	124	47	48	82	106	7	88	123	19
	107	11	10	70	0																			
2	0	112	0																					
3	0	1	33	78	9	120	135	35	136	65	71	103	20	30	131	32	90	126	63	64	49	143	36	114
	125	45	17	84	5	0																		
4	0	13	95	117	87	137	2	115	73	21	72	74	23	41	145	15	43	42	142	14	44	86	140	38
	141	16	92	99	104	0																		
5	0	58	0																					
6	0	94	0																					
7	0	146	27	69	101	122	51	66	128	108	62	148	52	89	147	118	8	113	119	97	144	57	75	39
	139	110	40	0																				
8	0	127	31	132	111	76	116	77	3	68	80	150	109	54	130	55	25	134	24	29	121	79	129	34
	81	102	50	0																				
9	0	105	26	12	149	4	67	56	133	22	53	0												
10	0	28	0																					
11	0	138	0																					

Figure A.8: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.9

<b>Route</b>	<b>CMT5X</b>																							
1	0	147	6	94	95	92	59	99	96	118	84	113	17	45	8	114	18	153	106	7	82	46	124	47
	36	143	11	90	0																			
2	0	77	3	129	78	34	135	35	136	65	71	9	120	81	33	122	0							
3	0	27	111	28	138	154	12	109	150	134	54	130	55	25	139	39	67	23	56	74	72	21	149	0
4	0	105	152	137	2	115	57	15	43	38	140	86	141	91	100	85	93	104	97	117	0			
5	0	146	88	148	62	159	10	70	101	69	116	80	68	20	66	128	131	32	126	63	64	49	107	19
	123	52	0																					
6	0	13	151	98	37	87	144	42	142	14	119	44	16	61	5	60	83	125	48	89	0			
7	0	26	76	50	157	51	103	79	158	121	29	24	155	4	110	75	133	22	41	145	73	40	58	53
	0																							
8	0	132	1	102	30	108	31	127	112	156	0													

Figure A.9: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.10

<u>Route</u>	<u>CMT11X</u>																							
1	0	67	69	70	71	74	72	75	78	80	79	77	76	73	68	40	43	45	51	50	48	42	39	38
	41	32	35	36	34	33	30	28	21	94	0													
2	0	99	54	57	55	60	63	66	64	62	61	65	59	47	49	46	44	37	29	31	27	24	22	25
	19	23	26	20	17	16	12	91	0															
3	0	81	84	117	113	83	2	3	4	5	6	7	9	10	11	15	14	13	8	52	53	58	56	98
	110	115	97	109	108	118	18	114	90	0														
4	0	120	0																					
5	0	1	0																					
6	0	107	104	103	116	100	96	93	92	89	85	112	0											
7	0	119	88	0																				
8	0	87	86	111	82	0																		
9	0	95	102	101	106	105	0																	

Figure A.10: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.11

<b>Route</b>	<b>CMT11Y</b>																							
1	0	21	26	23	20	17	16	19	25	22	24	27	33	30	31	34	36	35	28	32	29	44	46	47
	50	51	45	39	79	77	78	75	72	74	73	71	70	0										
2	0	110	37	38	42	41	49	48	43	40	59	57	54	52	53	55	58	61	65	62	64	66	63	60
	56	80	68	76	67	69	104	106	0															
3	0	82	81	112	84	117	113	83	2	1	3	4	5	6	7	9	10	11	15	14	13	12	8	108
	118	109	115	98	116	97	96	92	86	88	0													
4	0	111	85	89	91	90	18	114	94	93	102	101	99	100	103	107	105	120	0					
5	0	87	95	0																				
6	0	107	104	103	116	100	96	93	92	89	85	112	0											
7	0	119	0																					

Figure A.11: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.12

<u>Route</u>	<u>CMT12X</u>																							
1	0	43	42	41	40	44	45	48	46	47	49	52	50	51	31	32	33	35	37	38	39	36	30	28
	26	0																						
2	0	20	21	22	23	25	24	27	29	34	17	11	10	8	9	6	7	5	3	4	2	1	75	0
3	0	90	91	89	88	85	84	83	82	81	78	76	71	70	73	77	79	80	72	0				
4	0	87	86	74	62	61	64	55	54	53	56	58	60	59	68	69	63	65	67	0				
5	0	98	96	95	94	92	93	97	100	99	12	14	16	15	19	18	13	0						
6	0	66	57	0																				

Figure A.12: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure A.13

<u>Route</u>	<u>CMT12Y</u>																						
1	0	20	24	25	27	29	32	33	31	35	37	38	39	36	34	30	28	26	23	22	21	0	
2	0	99	100	97	93	92	94	95	96	98	91	89	88	85	84	82	83	86	87	90	0		
3	0	43	42	41	40	57	55	54	53	56	58	60	59	44	45	46	48	51	50	52	49	47	0
4	0	69	66	68	64	61	72	80	79	77	73	70	71	76	78	81	74	62	63	65	67	0	
5	0	7	3	5	75	1	2	4	6	8	9	12	14	16	15	19	18	17	13	11	10	0	

Figure A.13: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

## **APPENDIX B**

### **SDPmethod**

### 1<sup>st</sup> order Pareto front solutions

Table B.1

<b>CMT1X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	466.77	30.71
2	483.45	18.71
3	484.87	11.5
4	487.63	9.82
5	488.01	8.28
6	490.8	7.42
7	492.59	6.86
8	493.47	6.57
9	493.66	6.14
10	495.12	1.97
11	504.75	1.46
12	563.7	1.12
13	579.88	0.7
14	585.24	0.33
15	587.17	0.31
16	623.46	0.29

Table B.1: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.2

<b>CMT1Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	462.22	18.19
2	471.78	8.01
3	473.4	7.64
4	484.18	5.45
5	506.33	3.13
6	508	3.03
7	509.65	2.38
8	554.05	1.45
9	566.99	0.06

Table B.2: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.3

<b><u>CMT2X</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	766.78	22.58

Table B.3: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.4

<b><u>CMT2Y</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	731.78	17.26
2	766.11	14.17
3	767	11.86
4	779.37	8.84
5	782.26	6.96

Table B.4: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.5

<b><u>CMT3X</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	816.38	23.85
2	822.89	20.56
3	825.76	15.11
4	838.29	13
5	848.79	12.69
6	849.99	10.11
7	856.49	9.71
8	860.98	7.94
9	866.06	5.88
10	878.7	4.47
11	902.7	4.4

Table B.5: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.6

<b><u>CMT3Y</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	769.59	38.74
2	774	32.77
3	776.05	22.66
4	780.75	20.68
5	791.09	20.13
6	793.76	19.85
7	795.38	18.65
8	797.73	14.39
9	800.12	9.8
10	805.9	9.56
11	809.09	9.43
12	815.93	4.81
13	816.6	4.74
14	819.15	4.14
15	825.59	2.7
16	841.12	1.78

Table B.6: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.7

<b><u>CMT4X</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1191.19	44.66
2	1197.2	36.53
3	1248.65	18.62
4	1262.9	15.5
5	1263.57	14.19
6	1355.47	10.41

Table B.7: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.8

<b><u>CMT4Y</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1088.4	33.51

Table B.8: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.9

<b><u>CMT5X</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1601.44	42.75
2	1612.68	34.14
3	1650.01	31.83
4	1668.27	28.09

Table B.9: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.10

<b><u>CMT5Y</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1234.01	70.07
2	1283.26	39.73
3	1289.38	34.29
4	1360.67	32.72
5	1364.67	21.08

Table B.10: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.11

<b><u>CMT11X</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1127.6	126.65
2	1127.81	81.78
3	1128.44	78.03
4	1147.12	64.49
5	1197.8	41.46
6	1198.44	39.94
7	1203.07	36.69
8	1208.93	32.1
9	1220.21	30.21
10	1263.19	28.71

Table B.11: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.12

<b><u>CMT11Y</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	867.58	88.37
2	951.43	87.1
3	954.7	80.79
4	954.73	62.82
5	960.85	48.9
6	979.87	45.21
7	982.48	40.01
8	987.48	38.71
9	1021.11	23.48
10	1023.76	19.55
11	1032.11	17.74
12	1057.04	17.31
13	1082.4	15.02
14	1091.22	13.3
15	1099.03	12.97
16	1106.82	11.63

Table B.12: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.13

<b>CMT12X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	831.79	62.15
2	850.75	29.98
3	885.24	28.92

Table B.13: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.14

<b>CMT12Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	749.29	33.35
2	749.46	32.88
3	777.21	15.23
4	815.08	14.64
5	831.25	12.95
6	841.64	12.34
7	848.69	11.69
8	865.19	9.79
9	936.93	8.73
10	952.29	7.29

Table B.14: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.15

<b>CMT6X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	471.53	13.96
2	474.93	13.5
3	480.57	2.15
4	503.02	2.04
5	514.54	0.39

Table B.15: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.16

<b>CMT6Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	462.22	18.19
2	471.78	8.01
3	473.4	7.64
4	484.18	5.45
5	506.33	3.13
6	508	3.03
7	509.65	2.38
8	554.05	1.45
9	566.99	0.06

Table B.16: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.17

<b>CMT7X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	766.78	22.58

Table B.17: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.18

<b>CMT7Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	735.31	55.64
2	737.96	54.14
3	739.36	49.8
4	742.56	45.69
5	753.7	41.56
6	754.71	39.66
7	762.09	39.31
8	763.51	36.75
9	774.79	29.59
10	807.38	25.89
11	821.7	22.37
12	835.42	21.23
13	836.95	21.01
14	845.82	13.04
15	875.15	12.34
16	882.44	11.18
17	892.21	8.19
18	897.96	8.13

Table B.18: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.19

<b>CMT8X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	836.81	22.25
2	849	17.82
3	849.17	14.74
4	849.34	8.83
5	854.65	2.21
6	1012.92	1.42

Table B.19: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.20

<b>CMT8Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	769.59	38.74
2	774	32.77
3	776.05	22.66
4	780.75	20.68
5	791.09	20.13
6	793.76	19.85
7	795.38	18.65
8	797.73	14.39
9	800.12	9.8
10	805.9	9.56
11	809.09	9.43
12	815.93	4.81
13	816.6	4.74
14	819.15	4.14
15	825.59	2.7
16	841.12	1.78

Table B.20: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.21

<b>CMT9X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1139.78	51.08
2	1166.01	39.49
3	1191.39	25.11
4	1198.09	25.03
5	1200.82	12.74

Table B.21: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.22

<b>CMT9Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1385.3	29.09

Table B.22: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.23

<b>CMT10X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1601.44	42.75
2	1612.68	34.14
3	1650.01	31.83
4	1668.27	28.09

Table B.23: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.24

<b>CMT10Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1234.01	70.07
2	1283.26	39.73
3	1289.38	34.29
4	1360.67	32.72
5	1364.67	21.08

Table B.24: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.25

<b>CMT13X</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	1127.6	126.65
2	1127.81	81.78
3	1128.44	78.03
4	1147.12	64.49
5	1197.8	41.46
6	1198.44	39.94
7	1203.07	36.69
8	1208.93	32.1
9	1220.21	30.21
10	1263.19	28.71

Table B.25: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.26

<b><u>CMT13Y</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	902.87	62.25
2	911.04	60.26
3	912.43	51.49
4	980.19	49.47
5	984.75	42.65
6	996.1	31.61
7	1018.93	19.03
8	1020.75	18.29
9	1027.69	13.12
10	1042.8	11.78
11	1048.23	9.69
12	1049.62	9.57
13	1050.46	8.14
14	1057.32	5.77
15	1106.97	5.67

Table B.26: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.27

<b><u>CMT14X</u></b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	831.79	62.15
2	850.75	29.98
3	885.24	28.92

Table B.27: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

Table B.28

<b>CMT14Y</b>		
<u>Solution</u>	<u>Total Routing Distance</u>	<u>Workload Variation (%)</u>
1	773.62	37.14
2	814.41	35.28
3	815.18	31.83
4	839.16	29.4
5	841.23	27.77
6	847.93	20.39
7	860.94	16.97
8	870.01	14.02
9	872.3	12.9

Table B.28: Column 1 - Name of solution, Column 2 - Total routing distance, Column 3 - Maximum variation between route distances.

**Routing Information for the SDPmethod Solutions**

Figure B.1

<b><u>Route</u></b>	<b><u>CMT1X</u></b>																					
1	0	27	48	8	26	31	28	3	36	35	20	2	22	1	32	0						
2	0	46	11	38	5	49	9	50	16	29	21	34	30	10	39	33	45	15	44	37	17	0
3	0	12	47	18	4	42	19	40	41	13	25	14	24	43	7	23	6	0				

Figure B.1: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.2

<b><u>Route</u></b>	<b><u>CMT1Y</u></b>																			
1	0	46	11	38	5	49	9	34	30	10	39	33	45	15	44	37	17	12	0	
2	0	47	18	4	42	19	40	41	13	25	14	24	43	7	23	6	0			
3	0	27	48	8	26	31	28	3	36	35	20	29	21	50	16	2	22	1	32	0

Figure B.2: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.3

<u>Route</u>	<u>CMT2X</u>																		
1	0	75	68	2	33	63	23	56	49	24	3	17	51	6	0				
2	0	73	1	43	41	42	64	22	62	28	61	21	47	48	0				
3	0	46	8	35	7	53	59	19	54	13	57	15	0						
4	0	67	40	12	58	72	39	9	25	55	18	50	32	44	16	0			
5	0	26	4	34	52	27	45	30	74	69	36	71	60	70	20	37	5	29	0
6	0	14	11	66	65	38	10	31	0										

Figure B.3: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.4

<u>Route</u>	<u>CMT2Y</u>																		
1	0	51	12	39	31	10	65	53	7	35	8	46	26	0					
2	0	11	66	59	14	19	54	13	27	45	52	34	0						
3	0	67	38	58	72	9	25	55	50	32	44	3	40	0					
4	0	75	30	48	47	36	69	71	60	70	20	37	57	15	5	29	4	0	
5	0	33	63	43	42	41	56	23	49	24	18	17	0						
6	0	6	16	73	1	64	22	62	28	61	21	74	2	68	0				

Figure B.4: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.5

<u>Route</u>	<u>CMT3X</u>																								
1	0	13	94	6	96	59	93	91	16	44	14	38	86	17	45	46	8	83	84	5	60	89	0		
2	0	69	1	51	9	78	79	3	77	29	24	54	55	25	39	67	23	56	75	74	72	21	40	53	0
3	0	58	2	57	87	97	95	92	99	61	85	98	37	100	42	43	15	41	22	73	4	26	0		
4	0	28	12	80	68	34	35	71	65	66	30	70	31	88	62	11	82	52	0						
5	0	27	76	50	33	81	20	32	90	10	63	64	49	36	47	19	48	7	18	0					

Figure B.5: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.6

<u>Route</u>	<u>CMT3Y</u>																												
1	0	53	58	13	6	94	95	97	92	37	86	38	43	15	42	87	2	57	41	22	74	56	39	72	73	21	40	0	
2	0	27	50	76	77	12	80	68	3	79	33	81	35	34	78	29	24	54	55	25	67	23	75	4	26	0			
3	0	1	70	30	51	9	71	65	66	20	32	90	63	64	11	19	47	48	62	10	88	31	69	28	0				
4	0	89	18	83	60	5	99	96	59	93	85	98	100	91	14	44	16	61	84	17	45	8	46	36	49	82	7	52	0

Figure B.6: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.7

<u>Route</u>	<u>CMT4X</u>																							
1	0	96	104	93	53	138	26	149	40	21	73	72	74	75	56	39	139	4	110	54	134	121	3	132
	69	0																						
2	0	58	112	6	99	92	98	100	119	14	142	42	144	57	2	115	133	22	15	43	38	140	86	61
	17	84	5	146	0																			
3	0	27	111	28	12	109	80	150	116	77	50	102	81	78	29	24	130	55	25	67	23	41	145	0
4	0	147	123	10	31	101	1	70	122	30	128	20	103	136	34	129	79	68	0					
5	0	105	87	117	95	59	91	44	16	118	60	83	114	8	45	46	124	48	19	11	89	0		
6	0	76	33	51	9	120	135	35	71	65	66	131	32	90	108	126	63	64	49	47	0			
7	0	94	13	137	97	37	85	141	113	125	36	143	107	62	148	88	127	106	7	82	18	52	0	

Figure B.7: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.8

<u>Route</u>	<u>CMT4Y</u>																							
1	0	106	7	11	64	123	48	124	47	36	46	82	114	8	125	45	17	86	113	84	5	118	60	83
	18	0																						
2	0	147	96	95	97	92	59	99	104	93	85	98	37	100	91	16	141	44	140	38	119	42	57	145
	115	117	94	6	52	0																		
3	0	87	144	41	22	133	75	74	72	110	4	139	39	56	23	67	25	55	130	24	134	0		
4	0	132	69	1	101	70	30	122	51	9	71	65	20	128	131	90	63	126	108	10	31	127	88	148
	62	107	49	143	19	0																		
5	0	54	68	121	29	129	79	78	34	135	35	136	66	32	103	120	81	33	102	50	3	77	116	76
	0																							
6	0	53	105	26	149	109	80	150	12	138	28	111	27	146	89	61	14	142	43	15	2	73	21	40
	58	137	13	112	0																			

Figure B.8: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.9

<b>Route</b>	<b>CMT5X</b>																								
1	0	105	144	87	117	92	151	99	16	61	60	166	18	83	125	45	46	36	143	175	182	7	194	153	
	52	31	101	0																					
2	0	79	120	103	161	71	136	135	34	169	121	29	24	163	187	0									
3	0	146	112	6	95	93	37	15	2	22	75	186	21	198	110	80	68	12	26	180	40	53	0		
4	0	156	58	178	115	171	72	179	130	54	177	150	196	157	185	78	134	55	25	170	23	56	0		
5	0	152	137	97	142	14	38	44	192	91	193	100	98	104	5	173	84	199	8	124	48	82	89	0	
6	0	111	116	122	1	132	69	162	190	127	106	148	62	107	123	47	174	118	96	183	0				
7	0	3	158	129	33	164	35	65	66	188	20	128	30	160	181	64	63	88	0						
8	0	28	176	50	81	9	51	70	159	189	10	108	131	32	90	126	11	19	49	168	114	167	0		
9	0	13	57	172	42	43	140	119	191	141	86	113	17	85	59	94	147	0							
10	0	27	102	77	76	184	138	154	109	195	149	155	4	165	67	39	139	197	74	133	41	145	73	0	

Figure B.9: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.10

<b>Route</b>	<b>CMT5Y</b>																							
1	0	18	114	46	36	143	175	11	126	10	189	159	62	148	88	127	190	31	70	101	162	69	132	167
	27	0																						
2	0	146	106	194	182	7	82	48	123	19	107	64	49	168	47	124	174	8	199	118	0			
3	0	90	63	181	32	128	188	103	161	9	120	164	135	35	136	65	71	66	20	30	122	0		
4	0	92	37	98	93	85	100	193	91	191	141	44	119	192	14	142	42	43	38	140	86	113	16	61
	173	84	83	60	0																			
5	0	108	131	160	51	81	33	185	79	129	78	34	169	29	121	158	3	77	116	196	76	157	102	50
	1	176	0																					
6	0	52	166	151	97	58	152	53	105	26	180	40	197	56	68	184	138	154	28	0				
7	0	137	149	195	179	198	110	4	155	139	187	67	170	25	55	165	130	54	134	24	163	150	80	177
	109	12	0																					
8	0	21	73	171	74	72	39	23	186	75	133	22	41	145	15	172	144	57	178	115	2	0		
9	0	111	13	87	117	95	94	96	59	104	99	5	17	45	125	153	89	147	6	183	112	156	0	

Figure B.10: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.11

<u>Route</u>	<u>CMT11X</u>																							
1	0	56	60	63	66	64	62	61	65	57	59	43	45	48	51	50	49	47	46	44	42	41	37	29
	32	35	36	33	30	27	24	22	19	20	26	21	0											
2	0	119	81	82	111	86	85	112	84	117	83	108	2	4	6	7	9	10	15	13	8	109	97	110
	98	67	70	120	0																			
3	0	87	92	89	91	90	18	114	115	100	71	72	74	76	68	40	39	38	34	31	28	25	23	17
	16	14	3	113	0																			
4	0	95	102	105	106	107	104	116	73	77	79	52	54	58	55	53	80	78	75	69	103	99	101	96
	93	94	118	12	11	5	1	88	0															

Figure B.11: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.12

<u>Route</u>	<u>CMT11Y</u>																							
1	0	120	106	116	110	115	109	21	20	23	26	28	31	36	34	33	30	27	24	22	16	15	11	10
	6	5	4	3	1	2	108	114	93	96	0													
2	0	17	19	25	32	35	29	37	38	39	42	41	44	46	47	49	50	51	48	45	43	40	54	0
3	0	100	57	59	65	61	62	64	66	63	60	56	58	55	53	52	79	80	78	77	68	76	73	74
	72	75	71	70	67	107	99	95	0															
4	0	105	102	101	104	103	69	98	97	94	92	89	91	90	18	118	8	12	13	14	9	7	83	113
	117	84	85	112	81	119	82	111	86	87	88	0												

Figure B.12: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.13

<b>Route</b>	<b>CMT12X</b>																							
1	0	7	9	96	95	94	92	93	97	100	12	14	16	15	19	18	17	13	0					
2	0	90	98	99	2	1	75	5	3	4	6	8	23	26	28	30	34	29	27	25	24	22	21	20
	0																							
3	0	49	47	43	42	41	40	44	45	46	48	51	52	32	33	35	37	38	39	36	10	11	85	0
4	0	67	65	63	69	66	62	74	72	61	64	68	55	54	56	57	59	50	31	0				
5	0	91	88	89	87	86	84	83	82	81	78	76	71	70	73	77	79	80	53	58	60	0		

Figure B.13: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.14

<b>Route</b>	<b>CMT12Y</b>																							
1	0	5	1	98	4	6	8	11	12	14	16	15	19	18	17	13	28	25	24	23	0			
2	0	59	58	56	57	51	31	32	33	35	37	38	39	36	34	29	30	27	26	0				
3	0	65	63	72	61	55	54	53	60	40	41	42	44	46	45	48	50	52	49	47	43	20	22	21
	0																							
4	0	87	86	85	84	81	78	76	71	70	73	77	79	80	74	62	64	68	69	66	67	0		
5	0	75	3	7	10	9	2	99	100	97	93	92	94	95	96	91	89	88	82	83	90	0		

Figure B.14: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.15

<u>Route</u>	<u>CMT6X</u>																		
1	0	12	47	18	4	17	37	15	45	44	42	19	40	41	13	25	14	6	0
2	0	11	38	9	50	16	2	29	21	34	30	39	33	10	49	5	46	0	
3	0	32	1	22	20	35	36	3	28	31	8	26	7	43	24	23	48	27	0

Figure B.15: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.16

<u>Route</u>	<u>CMT6Y</u>																			
1	0	46	11	38	5	49	9	34	30	10	39	33	45	15	44	37	17	12	0	
2	0	47	18	4	42	19	40	41	13	25	14	24	43	7	23	6	0			
3	0	27	48	8	26	31	28	3	36	35	20	29	21	50	16	2	22	1	32	0

Figure B.16: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.17

<u>Route</u>	<u>CMT7X</u>																		
1	0	75	68	2	33	63	23	56	49	24	3	17	51	6	0				
2	0	73	1	43	41	42	64	22	62	28	61	21	47	48	0				
3	0	46	8	35	7	53	59	19	54	13	57	15	0						
4	0	67	40	12	58	72	39	9	25	55	18	50	32	44	16	0			
5	0	26	4	34	52	27	45	30	74	69	36	71	60	70	20	37	5	29	0
5	0	14	11	66	65	38	10	31	0										

Figure B.17: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.18

<u>Route</u>	<u>CMT7Y</u>															
1	0	4	45	27	13	54	67	26	40	49	73	2	75	0		
2	0	7	35	11	66	59	14	19	8	46	34	52	15	0		
3	0	29	5	57	37	20	70	60	71	36	47	48	30	62	0	
4	0	17	51	16	3	44	25	55	18	24	23	63	33	6	68	0
5	0	53	38	65	31	10	58	72	39	9	50	32	12	0		
5	0	1	43	56	41	42	64	22	28	61	69	21	74	0		

Figure B.18: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.19

<u>Route</u>	<u>CMT8X</u>																						
1	0	89	6	95	59	93	60	83	8	82	7	47	36	49	64	63	32	90	62	88	31	27	0
2	0	96	99	85	98	37	87	42	100	91	44	38	86	16	61	5	84	17	45	46	18	52	0
3	0	28	76	77	79	34	35	71	65	66	20	51	30	70	10	11	19	48	0				
4	0	69	1	50	3	33	81	9	78	29	24	68	80	12	54	55	4	56	75	2	40	58	0
5	0	13	94	97	92	14	43	15	57	41	22	23	67	25	39	74	72	73	21	26	53	0	

Figure B.19: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.20

<u>Route</u>	<u>CMT8Y</u>																						
1	0	53	58	13	6	94	95	97	92	37	86	38	43	15	42	87	2	57	41	22	74	56	39
	73	21	40	0																			
2	0	27	50	76	77	12	80	68	3	79	33	81	35	34	78	29	24	54	55	25	67	23	75
	26	0																					
3	0	1	70	30	51	9	71	65	66	20	32	90	63	64	11	19	47	48	62	10	88	31	69
	0																						
4	0	89	18	83	60	5	99	96	59	93	85	98	100	91	14	44	16	61	84	17	45	8	46
	49	82	7	52	0																		

Figure B.20: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.21

<u>Route</u>	<u>CMT9X</u>																						
1	0	147	95	97	117	26	149	109	150	79	24	134	54	110	4	139	39	56	75	133	41	74	72
	21	40	0																				
2	0	6	94	96	104	99	100	61	84	125	113	86	141	44	119	14	142	42	43	15	57	144	53
	0																						
3	0	27	111	102	76	116	3	129	34	29	121	68	80	130	55	25	67	23	22	145	115	2	105
4	0	101	31	148	62	11	70	122	30	128	103	65	136	35	135	78	33	77	12	0			
5	0	69	146	89	60	118	5	83	114	8	47	46	17	140	38	16	85	98	137	58	0		
6	0	28	50	51	81	120	9	71	66	20	131	32	90	108	126	63	64	49	143	36	124	127	0
7	0	112	13	87	92	59	37	91	93	45	82	48	123	19	107	10	88	7	106	18	52	1	132

Figure B.21: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.22

<u>Route</u>	<u>CMT9Y</u>																						
1	0	69	106	82	114	99	104	95	97	87	144	42	142	119	14	43	15	57	2	115	145	41	22
	23	39	138	0																			
2	0	132	70	10	90	128	20	103	33	81	34	129	79	3	77	50	76	116	68	80	150	121	29
	134	55	67	109	12	0																	
3	0	1	137	59	37	85	61	44	38	113	17	84	5	118	60	83	125	45	8	36	143	49	0
4	0	105	93	52	148	62	7	48	124	46	47	123	19	107	11	64	63	126	108	66	65	136	71
	120	0																					
5	0	111	26	149	54	130	110	4	139	56	75	74	72	73	21	40	100	18	127	31	30	9	35
	102	0																					
6	0	53	28	27	146	88	101	32	131	122	78	25	58	13	117	92	98	91	141	140	86	16	96
	6	147	89	112	0																		

Figure B.22: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.23

<b>Route</b>	<b>CMT10X</b>																							
1	0	105	144	87	117	92	151	99	16	61	60	166	18	83	125	45	46	36	143	175	182	7	194	153
	52	31	101	0																				
2	0	79	120	103	161	71	136	135	34	169	121	29	24	163	187	0								
3	0	146	112	6	95	93	37	15	2	22	75	186	21	198	110	80	68	12	26	180	40	53	0	
4	0	156	58	178	115	171	72	179	130	54	177	150	196	157	185	78	134	55	25	170	23	56	0	
5	0	152	137	97	142	14	38	44	192	91	193	100	98	104	5	173	84	199	8	124	48	82	89	0
6	0	111	116	122	1	132	69	162	190	127	106	148	62	107	123	47	174	118	96	183	0			
7	0	3	158	129	33	164	35	65	66	188	20	128	30	160	181	64	63	88	0					
8	0	28	176	50	81	9	51	70	159	189	10	108	131	32	90	126	11	19	49	168	114	167	0	
9	0	13	57	172	42	43	140	119	191	141	86	113	17	85	59	94	147	0						
10	0	27	102	77	76	184	138	154	109	195	149	155	4	165	67	39	139	197	74	133	41	145	73	0

Figure B.23: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.24

<b>Route</b>	<b>CMT10Y</b>																							
1	0	18	114	46	36	143	175	11	126	10	189	159	62	148	88	127	190	31	70	101	162	69	132	167
	27	0																						
2	0	146	106	194	182	7	82	48	123	19	107	64	49	168	47	124	174	8	199	118	0			
3	0	90	63	181	32	128	188	103	161	9	120	164	135	35	136	65	71	66	20	30	122	0		
4	0	92	37	98	93	85	100	193	91	191	141	44	119	192	14	142	42	43	38	140	86	113	16	61
	173	84	83	60	0																			
5	0	108	131	160	51	81	33	185	79	129	78	34	169	29	121	158	3	77	116	196	76	157	102	50
	1	176	0																					
6	0	52	166	151	97	58	152	53	105	26	180	40	197	56	68	184	138	154	28	0				
7	0	137	149	195	179	198	110	4	155	139	187	67	170	25	55	165	130	54	134	24	163	150	80	177
	109	12	0																					
8	0	21	73	171	74	72	39	23	186	75	133	22	41	145	15	172	144	57	178	115	2	0		
9	0	111	13	87	117	95	94	96	59	104	99	5	17	45	125	153	89	147	6	183	112	156	0	

Figure B.24: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.25

<u>Route</u>	<u>CMT13X</u>																							
1	0	56	60	63	66	64	62	61	65	57	59	43	45	48	51	50	49	47	46	44	42	41	37	29
	32	35	36	33	30	27	24	22	19	20	26	21	0											
2	0	119	81	82	111	86	85	112	84	117	83	108	2	4	6	7	9	10	15	13	8	109	97	110
	98	67	70	120	0																			
3	0	87	92	89	91	90	18	114	115	100	71	72	74	76	68	40	39	38	34	31	28	25	23	17
	16	14	3	113	0																			
4	0	95	102	105	106	107	104	116	73	77	79	52	54	58	55	53	80	78	75	69	103	99	101	96
	93	94	118	12	11	5	1	88	0															

Figure B.25: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.26

<u>Route</u>	<u>CMT13Y</u>																								
1	0	3	4	10	11	15	14	13	12	17	16	19	20	21	26	23	28	25	22	24	27	33	30	31	
	34	36	35	32	29	49	50	48	40	0															
2	0	37	38	39	42	41	44	46	47	51	45	43	59	65	61	62	64	66	63	60	56	58	55	53	
	54	57	52	0																					
3	0	120	70	69	90	114	18	118	108	8	9	1	2	83	113	117	84	112	81	119	82	111	86	85	
	89	92	87	88	0																				
4	0	95	102	101	99	100	116	97	94	96	93	91	5	6	7	109	115	110	98	68	73	76	77	79	
	80	78	75	72	74	71	67	103	104	107	106	105	0												

Figure B.26: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.27

<b>Route</b>	<b>CMT14X</b>																							
1	0	7	9	96	95	94	92	93	97	100	12	14	16	15	19	18	17	13	0					
2	0	90	98	99	2	1	75	5	3	4	6	8	23	26	28	30	34	29	27	25	24	22	21	20
	0																							
3	0	49	47	43	42	41	40	44	45	46	48	51	52	32	33	35	37	38	39	36	10	11	85	0
4	0	67	65	63	69	66	62	74	72	61	64	68	55	54	56	57	59	50	31	0				
5	0	91	88	89	87	86	84	83	82	81	78	76	71	70	73	77	79	80	53	58	60	0		

Figure B.27: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.28

<b>Route</b>	<b>CMT14Y</b>																							
1	0	7	3	1	98	95	92	93	100	12	14	16	15	19	18	17	13	11	9	0				
2	0	63	5	4	6	8	10	23	26	28	30	39	38	35	33	29	27	25	24	22	21	20	0	
3	0	47	49	52	32	34	36	37	31	51	50	48	45	44	60	58	56	59	40	41	42	46	43	0
4	0	67	65	74	79	77	73	70	80	53	54	57	55	68	64	61	72	62	66	69	0			
5	0	90	91	89	87	86	81	78	76	71	82	83	84	85	88	94	97	96	99	2	75	0		

Figure B.28: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

**New Best Known Solutions**

Figure B.29

<b><u>Route</u></b>	<b><u>CMT1Y</u></b>																			
1	0	46	11	38	5	49	9	34	30	10	39	33	45	15	44	37	17	12	0	
2	0	47	18	4	42	19	40	41	13	25	14	24	43	7	23	6	0			
3	0	27	48	8	26	31	28	3	36	35	20	29	21	50	16	2	22	1	32	0

Figure B.29: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.30

<b><u>Route</u></b>	<b><u>CMT6X</u></b>																		
1	0	12	47	18	4	17	37	15	45	44	42	19	40	41	13	25	14	6	0
2	0	11	38	9	50	16	2	29	21	34	30	39	33	10	49	5	46	0	
3	0	32	1	22	20	35	36	3	28	31	8	26	7	43	24	23	48	27	0

Figure B.30: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

Figure B.31

<u>Route</u>	<u>CMT6Y</u>																			
1	0	46	11	38	5	49	9	34	30	10	39	33	45	15	44	37	17	12	0	
2	0	47	18	4	42	19	40	41	13	25	14	24	43	7	23	6	0			
3	0	27	48	8	26	31	28	3	36	35	20	29	21	50	16	2	22	1	32	0

Figure B.31: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

**Best Known Solutions**

Figure B.32

<u>Route</u>	<u>CMT1X</u>																					
1	0	27	48	8	26	31	28	3	36	35	20	2	22	1	32	0						
2	0	46	11	38	5	49	9	50	16	29	21	34	30	10	39	33	45	15	44	37	17	0
3	0	12	47	18	4	42	19	40	41	13	25	14	24	43	7	23	6	0				

Figure B.32: Column 1 - Routing name, Column 2-onwards - The sequence nodes are serviced on the route, where '0' denotes the depot.

## APPENDIX C

### Multi Objective Genetic Algorithm Parameter Setup

**Population size and maximum number of generation's setup results**

Table C.1

<u>Test Problem</u>	<u>Population Size</u>	<u>Accuracy</u>	<u>Generation</u>
CMT1X	100	0.45	89
	200	0.41	91
	300	0.36	93
	400	0.35	90
	<b>500</b>	<b>0.29</b>	<b>87</b>
	600	0.3	88
CMT1Y	100	0.41	91
	200	0.37	99
	300	0.32	96
	400	0.3	91
	<b>500</b>	<b>0.27</b>	<b>95</b>
	600	0.28	97
CMT2X	100	0.68	95
	200	0.61	95
	300	0.57	94
	400	0.49	89
	<b>500</b>	<b>0.45</b>	<b>100</b>
	600	0.45	96
CMT2Y	100	0.62	91
	200	0.57	99
	300	0.45	96
	400	0.42	91
	<b>500</b>	<b>0.4</b>	<b>95</b>
	600	0.42	97
CMT3X	100	0.79	95
	200	0.71	95
	300	0.56	100
	400	0.53	89
	<b>500</b>	<b>0.48</b>	<b>97</b>
	600	0.5	96
CMT3Y	100	0.77	91
	200	0.69	99
	300	0.64	96
	400	0.58	91
	<b>500</b>	<b>0.44</b>	<b>95</b>
	600	0.47	97
CMT4X	100	0.85	95
	200	0.79	95
	300	0.76	92
	400	0.73	89
	<b>500</b>	<b>0.7</b>	<b>100</b>
	600	0.71	96

<u>Test Problem</u>	<u>Population Size</u>	<u>Accuracy</u>	<u>Generation</u>
CMT4Y	100	0.83	91
	200	0.8	99
	300	0.72	96
	400	0.7	91
	<b>500</b>	<b>0.66</b>	<b>95</b>
	600	0.68	97
CMT5X	100	0.96	95
	200	0.95	95
	300	0.94	100
	400	0.92	89
	<b>500</b>	<b>0.9</b>	<b>96</b>
	600	0.9	100
CMT5Y	100	0.93	91
	200	0.91	99
	300	0.86	96
	400	0.83	91
	<b>500</b>	<b>0.79</b>	<b>95</b>
	600	0.8	97
CMT6X	100	0.51	95
	200	0.37	100
	300	0.35	95
	400	0.29	89
	<b>500</b>	<b>0.26</b>	<b>100</b>
	600	0.29	96
CMT6Y	100	0.51	91
	200	0.46	99
	300	0.35	96
	400	0.31	91
	<b>500</b>	<b>0.27</b>	<b>95</b>
	600	0.28	97
CMT7X	100	0.7	95
	200	0.66	95
	300	0.56	93
	400	0.51	89
	<b>500</b>	<b>0.49</b>	<b>99</b>
	600	0.49	96
CMT7Y	100	0.66	91
	200	0.6	99
	300	0.57	96
	400	0.53	91
	<b>500</b>	<b>0.5</b>	<b>95</b>
	600	0.52	97

<u>Test Problem</u>	<u>Population Size</u>	<u>Accuracy</u>	<u>Generation</u>
CMT8X	100	0.85	95
	200	0.7	95
	300	0.64	89
	400	0.54	100
	<b>500</b>	<b>0.5</b>	<b>100</b>
	600	0.52	96
CMT8Y	100	0.8	91
	200	0.62	99
	300	0.55	96
	400	0.5	91
	<b>500</b>	<b>0.45</b>	<b>95</b>
	600	0.48	100
CMT9X	100	0.88	95
	200	0.86	95
	300	0.79	100
	400	0.75	89
	<b>500</b>	<b>0.7</b>	<b>97</b>
	600	0.71	96
CMT9Y	100	0.89	91
	200	0.87	99
	300	0.84	96
	400	0.82	91
	<b>500</b>	<b>0.78</b>	<b>95</b>
	600	0.78	97
CMT10X	100	0.97	95
	200	0.96	95
	300	0.93	99
	400	0.91	89
	<b>500</b>	<b>0.89</b>	<b>99</b>
	600	0.9	96
CMT10Y	100	0.9	91
	200	0.86	99
	300	0.83	96
	400	0.8	91
	<b>500</b>	<b>0.78</b>	<b>95</b>
	600	0.78	97
CMT11X	100	0.93	95
	200	0.9	95
	300	0.88	100
	400	0.85	89
	<b>500</b>	<b>0.83</b>	<b>96</b>
	600	0.85	100

<u>Test Problem</u>	<u>Population Size</u>	<u>Accuracy</u>	<u>Generation</u>
CMT11Y	100	0.83	91
	200	0.81	99
	300	0.75	96
	400	0.73	91
	<b>500</b>	<b>0.66</b>	<b>95</b>
	600	0.7	97
CMT12X	100	0.71	95
	200	0.7	95
	300	0.68	96
	400	0.64	89
	<b>500</b>	<b>0.61</b>	<b>100</b>
	600	0.62	100
CMT12Y	100	0.74	91
	200	0.71	99
	300	0.63	96
	400	0.56	91
	<b>500</b>	<b>0.49</b>	<b>95</b>
	600	0.53	97
CMT13X	100	0.94	95
	200	0.91	95
	300	0.89	100
	400	0.86	89
	<b>500</b>	<b>0.84</b>	<b>93</b>
	600	0.84	96
CMT13Y	100	0.73	91
	200	0.71	99
	300	0.69	96
	400	0.63	91
	<b>500</b>	<b>0.59</b>	<b>95</b>
	600	0.6	97
CMT14X	100	0.79	97
	200	0.74	95
	300	0.69	95
	400	0.66	89
	<b>500</b>	<b>0.6</b>	<b>100</b>
	600	0.63	96
CMT14Y	100	0.69	91
	200	0.66	99
	300	0.59	96
	400	0.56	91
	<b>500</b>	<b>0.53</b>	<b>95</b>
	600	0.54	97

Column 1 - Test problem, Column 2 - Number of individuals in the population, Column 3 - Average distance between the 1<sup>st</sup> order Pareto front and the origin of the axes and Column 4 - Generation the minimum average distance between the 1<sup>st</sup> order Pareto front and the origin of the axes was found.