Hybrid Metaheuristics for Solving Multi-Depot Pickup and Delivery Problems

by

Pairoj Chaichiratikul

Imperial College London
Imperial College Business School

September 2013

Submitted to Imperial College London
In partial fulfilment of the requirement for the degree of
Doctor of Philosophy

Declaration of Originality

This thesis consists of the research work conducted in Imperial College Business School at Imperial College London. I declare that the work presented in this thesis is my own, except where acknowledged in the thesis.

Pairoj Chaichiratikul

London, September 30, 2013

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution-Non Commercial-No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or distribution, researchers must make clear to others the license terms of this work.

Acknowledgements

First of all, I would like to thank my supervisor, Prof. Eleni Hadjiconstantinou for her support, help and guidance throughout this research.

On a more personal level, I would like to give special thanks to my parents Nimitr and Rungrat and my two sisters, Praparat and Preamsuda for their kindness, patience, and financial support throughout my studies.

I would also like to thank the members of "Operations Research" group at Imperial College Business School for the fruitful academic discussions we had together: thank you Dr. Evelina Klerides, Dr. Wolfram Wiesemann, Dr. Ma Nang Laik, Dr. Efstratios Rappos , Alexandra Spachis and, William Wu for your help, suggestions, and opinions. I would also like to thank my colleagues at Imperial College School: Dr. Suttipong Thajchayapong, Dr. Panavy Pookaiyaudom, Dr. Komkrit Ovararin, Dr. Worrawat Sritrakul, Panita Surachaikulwattana, Sirirat Rattanapituk, Gunyawee Teekathananont, and Hae-Kyung Shin.

To my family

Abstract

In today's logistics businesses, increasing petrol prices, fierce competition, dynamic business environments and volume volatility put pressure on logistics service providers (LSPs) or third party logistics providers (3PLs) to be efficient, differentiated, adaptive, and horizontally collaborative in order to survive and remain competitive. In this climate, efficient computerised-decision support tools play an essential role. Especially, for freight transportation, efficiently solving a Pickup and Delivery Problem (PDP) and its variants by an optimisation engine is the core capability required in making operational planning and decisions. For PDPs, it is required to determine minimum-cost routes to serve a number of requests, each associated with paired pickup and delivery points. A robust solution method for solving PDPs is crucial to the success of implementing decision support tools, which are integrated with Geographic Information System (GIS) and Fleet Telematics so that the flexibility, agility, visibility and transparency are fulfilled. If these tools are effectively implemented, competitive advantage can be gained in the area of cost leadership and service differentiation.

In this research, variants of PDPs, which multiple depots or providers are considered, are investigated. These are so called Multi-depot Pickup and Delivery Problems (MDPDPs). To increase geographical coverage, continue growth and encourage horizontal collaboration, efficiently solving the MDPDPs is vital to operational planning and its total costs.

This research deals with designing optimisation algorithms for solving a variety of real-world applications. Mixed Integer Linear Programming (MILP) formulations of the MDPDPs are presented. Due to being NP-hard, the computational time for solving by

exact methods becomes prohibitive. Several metaheuristics and hybrid metaheuristics are investigated in this thesis. The extensive computational experiments are carried out to demonstrate their speed, preciseness and robustness.

Contents

1 Introduction **27**

1.1 Motivation 28

1.2 Modelling and solution methods 30

1.3 Goals 33

1.4 Thesis Overview 33

2 Literature Review **37**

2.1 Routing and Scheduling Problems 37

2.1.1 Travelling Salesman Problem (TSP) 37

2.1.2 Vehicle Routing Problems (VRPs) 38

2.1.3 Pickup and Delivery Problems (PDPs) 40

2.1.4 Rich Vehicle Routing Problems 41

2.2 Logistics Outsourcing Models 44

2.3 Solution Methodology 46

2.3.1 Exact Methods 47

2.3.2 Heuristics 51

2.3.3 Meta-heuristics 54

2.3.4 Hybrid Meta-heuristics 61

2.3.5 Summary 64

3 A Memetic Algorithm for the Multi-depot Pickup and Delivery Problem **65**

3.1 Introduction 65

3.2 Literature Review . . . 67
3.2.1 Multi-Depot Vehicle Routing Problems (MDVRP) . . . 67
3.2.2 Pickup and Delivery Problems (PDPs) . . . 69
3.2.3 Memetic Algorithms . . . 71
3.3 Problem Description and Formulation . . . 73
3.4 An Illustrative Example . . . 76
3.5 Test Problem Generation . . . 78
3.5.1 Study of problem parameters . . . 79
3.5.2 Tuning Performance Features in CPLEX . . . 79
3.5.3 Investigation of Borderline Customers . . . 82
3.6 Design of a MA for the MDPDP . . . 84
3.6.1 Framework . . . 85
3.6.2 Population Structure and Initialisation . . . 86
3.6.3 Fitness Function . . . 88
3.6.4 Tournament Selection . . . 89
3.6.5 Recombination Operator . . . 89
3.6.6 Local Search . . . 91
3.6.7 Replacement Strategy . . . 95
3.6.8 Fix and Forward Insertion Method and Reduction Rule . . . 95
3.7 Computational Experiments . . . 99
3.7.1 Implementation and Parameter Setting . . . 99
3.7.2 Computational Results . . . 100
3.8 Summary . . . 101

4 An Adaptive Memetic Large Neighbourhood Search for the Multi-Depot Pickup and Delivery Problem with Time Windows and Special Requests 106

4.1 Introduction . . . 106
4.2 State-of-the-art Reviews of Related Problems . . . 107
4.2.1 Gaps in the Literature . . . 110

4.3 Problem Formulation . . . 112
4.3.1 Notations . . . 112
4.3.2 Mathematical Model . . . 114
4.4 Design of Hybrid Metaheuristics . . . 116
4.4.1 State-of-the-Art Review of Related Methodologies . . . 116
4.4.2 Conceptual Design . . . 117
4.5 An Adaptive Memetic Large Neighbourhood Search (AMLNS) . . . 124
4.5.1 Removal Operators . . . 126
4.5.2 Identical Vehicle Crossover (IVX) . . . 130
4.5.3 Insertion Operators . . . 133
4.5.4 Adaptive Mechanism . . . 134
4.5.5 Applying Noise to Objective Function . . . 136
4.5.6 Initialisation . . . 136
4.5.7 Master Local Search Framework . . . 137
4.5.8 Reduction Rules for Improving Computational time . . . 142
4.6 Computational Experiments . . . 146
4.6.1 Small-sized Test Instances . . . 146
4.6.2 Medium-sized and Large-sized Test Instances . . . 147
4.6.3 Analysis of Typical Search . . . 153
4.6.4 Computational Results . . . 160
4.7 Discussion . . . 168
4.8 Summary . . . 171

5 Development of the Adaptive Memetic Large Neighbourhood Search: Implementational Aspects 173

5.1 Single-solution Approach . . . 177
5.1.1 ALNS using Threshold Accepting . . . 177
5.1.2 ALNS using Modified Threshold Accepting . . . 179
5.2 Population-based Approach . . . 181

5.2.1 Multiple ALNS . . . 182
5.2.2 Memetic Algorithm and ALNS . . . 184
5.3 Hybridisation between Population- and Single- Solution Approaches . . . 187
5.3.1 Adaptive Memetic Large Neighbourhood Search (AMLNS) . . . 187

6 An Integrated Truck and Semi-trailer Routing Problem: A Practical Multi-Depot Pickup and Delivery Problem for Road Freight Transport 190

6.1 Introduction . . . 190
6.2 Problem Overview . . . 191
6.3 State-of-the-art Reviews of Related Problems . . . 195
6.3.1 Multi-Depot Pickup and Delivery Problem with Time Windows and Special Request . . . 197
6.3.2 Multi-dimensional Capacity Constraints . . . 197
6.3.3 Truck and trailer routing problem . . . 198
6.3.4 sub-contraction . . . 199
6.3.5 Capacity-Driven Activity Based Costing (CDABC) . . . 201
6.3.6 Pricing . . . 204
6.3.7 Gaps in the Literature . . . 205
6.3.8 Problem Complexity . . . 207
6.4 Problem Description . . . 208
6.4.1 Requests . . . 208
6.4.2 Trucks . . . 208
6.4.3 Semi-trailers . . . 209
6.4.4 Costing . . . 209
6.4.5 Pricing . . . 210
6.4.6 Subcontractors . . . 211

7 Solution Methods for the Integrated Truck and Semi-trailer Routing Problem 215

7.1 Problem Formulation . . . 215
7.1.1 Assumptions . . . 216
7.1.2 Notations . . . 217
7.1.3 Objective Function . . . 220
7.1.4 Constraints . . . 222
7.1.5 Illustrative Example . . . 226
7.2 Adaptive Memetic Large Neighbourhood Search (AMLNS) . . . 230
7.2.1 Removal Operators . . . 230
7.2.2 Insertion Operators . . . 233
7.2.3 Adaptive Mechanism . . . 234
7.2.4 Applying Noise to Objective Function . . . 235
7.2.5 Initialisation . . . 235
7.2.6 Master Local Search Framework . . . 235
7.3 Computational Experiments . . . 235
7.3.1 Development of Sets of Problem Instances . . . 236
7.3.2 Tuning instances . . . 238
7.3.3 Parameter tuning . . . 238
7.3.4 Analysis of Typical Search . . . 241
7.3.5 Computational results . . . 242
7.4 Discussion . . . 249
7.4.1 Algorithmic Perspectives . . . 249
7.4.2 Managerial Perspectives . . . 250
7.5 Summary . . . 251

8 Conclusions **253**

8.1 Contributions . . . 253
8.2 Concluding remarks . . . 256
8.3 Future work . . . 259

A Lemma and Proof for Reduction Rule of Time Windows **275**

B Comparison of Run Times for the AMLNS **278**

C New Best Known Solutions on Ropke and Pisinger's (2006) Instances **279**

D Capacity-Driven ABC for Road Freight Transport **293**

E Schedule of the Best Known Solution of Problem 50F (Chapter 7) **296**

List of Tables

3.1 Experimental parameters 100

3.2 Comparison of GA with the optimal solutions, solved by CPLEX, in increasing vehicles' capacity 102

3.3 Comparison of GA with the optimal solutions, solved by CPLEX, in varying the distribution of vehicles in each depot 103

4.1 The features of the benchmark test instances used in Ropke and Pisinger (2006) 148

4.2 Tuning Instances for Problem Type in Each Problem Size 149

4.3 Experiments between Cooling Rate and Start Threshold 152

4.4 Experiments between First and Second Tournament Selection Probability . . . 152

4.5 Experiments between Probability and Randomness of Crossover 153

4.6 Parameters and settings used throughout the development 154

4.7 Hamming distance between accepted solutions and the currently best known solution, using the AMLNS for Problem 500A 160

4.8 Computational results for Multi-Depot PDPTW with Special Requests, 50 and 100 requests 162

4.9 Computational results for Multi-Depot PDPTW with Special Requests, 250 and 500 requests 163

4.10 Detailed Schedule of the New Best Known Solution for Problem 50 A 165

4.11 Detailed Schedule of the New Best Known Solution for Problem 50 A (cont.) . 166

4.12 Detailed Schedule of New Best Known Solution for Problem 50 A (cont.) 166

5.1 Experiments on design and parameters for TA-ALNS 178
5.2 Experiments on designs and parameters for TA-ALNS 180
5.3 Experiments on design and parameters with computational results for MALNS 184
5.4 Crossover rules to measure route quality 185
5.5 Experiments on design and parameters for MA-ALNS 186
5.6 Computational results for MA-ALNS 7 and AMLNS 1 designs 189

6.1 The relationship between direct, indirect, variable and fixed costs of a particular job 202

7.1 Routes and costs of own and subcontractor's vehicles 229
7.2 The features of the test instances for the ITSRP 237
7.3 Parameter N_{SR} vs. $Obj\ fn$ 239
7.4 First measure of IVX in different designs vs. $Obj\ fn$ 240
7.5 Second measure of IVX in different designs vs. $Obj\ fn$ 241
7.6 Comparison between the ALNS and AMLNS for 50 requests 243
7.7 Comparison between the ALNS and AMLNS for 100 requests 244
7.8 Semi-trailer assignment of the best known solution: problem 50F (100 locations) 246
7.9 Schedules of truck 17, 21, and 25 of best known solution: problem 50F 247

B.1 Scaling factors for computational time 278

C.1 Best Known Solutions of Problem 100B 281
C.2 Best Known Solutions of Problem 250C: Route 0-29 282
C.3 Best Known Solutions of Problem 250C: Route 30-59 283
C.4 Best Known Solutions of Problem 250C: Route 60-74 284
C.5 Best Known Solutions of Problem 500D: Route 0-29 285
C.6 Best Known Solutions of Problem 500D: Route 30-59 286
C.7 Best Known Solutions of Problem 500D: Route 60-89 287
C.8 Best Known Solutions of Problem 500D: Route 90-119 288
C.9 Best Known Solutions of Problem 500D: Route 120-149 289

D.1 Cost elements of Routing and Truck Operation 294

D.2 Cost rates corresponding to Table D.1 for order costing 294

D.3 Cost rates corresponding to Table D.1 for daily operational planning 294

E.7 Cost elements of the best known solution of problem 50E 296

E.1 Detailed schedule of the best known solution of problem 50F: truck 0,1,2,3 . . . 297

E.2 Detailed schedule of the best known solution of problem 50F: truck 4,5,6,7 . . . 298

E.3 Detailed schedule of the best known solution of problem 50F: truck 8,9,10,11 . . 299

E.4 Detailed schedule of the best known solution of problem 50F: truck 12, 13,14,15 300

E.5 Detailed schedule of the best known solution of problem 50F: truck 16,17,18,19,20,21301

E.6 Detailed schedule of the best known solution of problem 50F: truck 22, 23, 25 . 302

List of Figures

2.1 Dimensions of richness in VRPs (Drexl,2012:p.49) 43
2.2 Logistics Outsourcing Models(Langley et al. 2004, p.23) 44
2.3 3PLs Are Evolving Into Supply Chain Orchestrators(Langley et al. 2009, p.34) 45
2.4 Standard Mixed-Integer Linear Programming (MILP) Formulation 48
2.5 Local optimum and global optimum in a search space (Talbi 2009, p.91) 51
2.6 Local search (steepest descent) behavior in a given landscape (Talbi 2009, p.122) 52
2.7 Exchange of links (i,k),(j,l) for links (i,j), (k,l) 53
2.8 Simulated annealing escaping from local optima (Talbi 2009, p.127) 55
2.9 Iterated Local Search escaping local optimum (Talbi 2009, p.147) 57
2.10 Inspiration from an ant colony searching an optimal path between the food and the nest (Talbi 2009, p.241) 59
2.11 Particle swarm with their associated positions and velocities 60

3.1 Illustrative Example: Network typology of 2 depots and 10 requests (20 locations) 77
3.2 Illustrative Example: Optimal solution obtained from CPLEX 78
3.3 Network typlogy for testing the depot clustering heuristic with borderline paired requests 83
3.4 The optimal solution of test instance 25 84
3.5 Flowchart of Memetic Algorithm (MA) for MDPDP 85
3.6 Example of chromosome representation for MDPDP 88
3.7 Illustrative example of PDRC 90

3.8 Illustrative example of available insertion position 96
3.9 Illustrative example of fix-forward insertion: 1^{st} fix (pickup) and 1^{st} forward (delivery) position 96
3.10 Illustrative example of fix-forward insertion: 1^{st} fix (pickup) and 7^{th} forward (delivery) position 96
3.11 Illustrative example of fix-forward insertion: 2^{nd} fix (pickup) and 2^{nd} forward (delivery) position 96
3.12 Illustrative example of fix-forward insertion: 2^{nd} fix (pickup) and 7^{th} forward (delivery) position 97
3.13 Illustrative example of reduction rule in fix-forward insertion: vehicle loading . 98
3.14 Illustrative example of reduction rule in fix-forward insertion: updated vehicle loading 98
3.15 Illustrative example of reduction rule in fix-forward insertion: marked inserting location 98
3.16 The optimal solutions of test instances containing three geographical distributions:uniform (a), clustered (b), semi-clustered (c) and 2 (d), 3 (e), 4 (f) depots in the uniform distribution 104

4.1 Flowchart of the AMLNS 125
4.2 The Identical Vehicle Crossover (IVX) 133
4.4 Route structures for MD-PDPTW-SR 155
4.3 Zhu (2003)'s encoding for VRPTW (Zhu 2003,p.3) 155
4.5 Search Trajectories (left) and Population Diversity (right) of Solutions by the AMLNS for Problem 50A 156
4.6 Roulette Wheel Probability (left) and Smoothened Score (right) of the Selected Solution using AMLNS for Problem 50A 157
4.7 Search Trajectories (left) and Population Diversity (right) of Solutions by AMLNS for Problem 500A 158

4.8 New Best Known Solution of Problem 50A (100 locations) obtained by the AMLNS . . . 165
4.9 New Best Known Solution of Problem 500E (1000 locations) . . . 167

5.1 Design Matrix for Hybridising Metaheuristics . . . 175
5.2 (Left) Thresholds between a linear and exponential cooling rate for TA-ALNS 3 and MTA-ALNS 3, respectively. (Right) Search trajectories between TA-ALNS 3 and MTA-ALNS 3 . . . 181
5.3 Search Trajectories of Solutions by MA-ALNS 7 (left) and AMLNS (right) for Problem 500L . . . 188

6.1 (Left) Truck with 3 axles (Right) Semi-trailers with 3 axles . . . 191
6.2 (Left) 20ft Reefer container (special request for truck) (Right) 20ft Dry container . . . 192
6.3 (Left) Normal request for semi-trailer (Right) Special request for semi-trailer . 193
6.4 (Left) The combination of 2-axles truck and 2-axles trailer (Right) the combination between 3-axles truck and 2-axles trailer . . . 194
6.5 Iterative sub-contraction Process . . . 213

7.1 Pickup and corresponding delivery locations of 6 requests . . . 227
7.2 Optimal solution of the illustrative example . . . 228
7.3 (Left) ALNS and (Right) AMLNS used for solving problem 100L . . . 241
7.4 Network topology of best known solution: problem 50F (100 locations) . . . 245
7.5 Network topology of best known solution: problem 100Q (200 locations) . . . 248

8.1 Characteristics of VRPs considered in this thesis . . . 257

A.1 Fix-forward Insertion using Reduction Rule of Time Windows . . . 276
A.2 Figure illustrating for 1^{st} and 2^{nd} decisions . . . 276
A.3 Figure illustrating for 3^{rd} and 4^{th} decisions . . . 277

C.1 Two Best Known Solutions of Problem 50K . . . 280

C.2 The New Best Known Solution of Problem 50E (49923.61) 290
C.3 The New Best Known Solution of Problem 50L (64936.76) 291
C.4 The New Best Known Solution of Problem 50H (56761.36) 292

List of Algorithms

3.1 Outline of the genetic routing system 72

3.2 Modified Worst Removal 92

3.3 Modified Shaw's removal heurisitc 93

4.1 Pool Template for Hybrid Metaheuristics 118

7.1 Semi-trailer Repair Operator 232

7.2 Heuristic selection of semi-trailer 233

7.3 Procedure of Semi-trailer Insertion Heuristic (SIH) 234

Acronyms

3PL	Third Party Logistics Provider
4PL	Fourth Party Logistics Provider
ACO	Ant Colony Optimisation
AGES	Active Guided Evolution Strategy
ALNS	Adaptive Large Neighbourhood Search
AMLNS	Adaptive Large Neighbourhood Search
B&B	Branch-and-Bound
B&C	Branch-and-Cut
B&P	Branch-and-Price
BB	Building Block
BCRC	Best Cost Route Crossover
CCVRP	Cumulative Capacitated Vehicle Routing Problem
CDABC	Capacity-Driven Activity-Based Costing
CVRP	Capacitated Vehicle Routing Problem
DARP	Dial-A-Ride Problem
DP	Dynamic Programming
DVRP	Distance Constrained Vehicle Routing Problem
EA	Evolutionary Algorithm

FSMVRPTW	Fleet Size and Mix Vehicle Routing Problem with Time Windows
FLGA	Fuzzy Logic guided Genetic Algorith
FCGA	Family Competition Genetic Algorithm
GA	Genetic Algorithm
GAMSA	GA-based Multiple SA
GDA	Great Deluge Algorithm
GenClust	Genetic Clustering
Genset.	Power generator set
GGA	Grouping Genetic Algorithm
GLS	Genetic Local Search
GRASP	Greedy Randomised Adaptive Search Procedure
GVRP	General Vehicle Routing Problem
HGA	Hybrid Genetic Algorithm
HMOEA	Hybrid Multi-objective Evolutionary Algorithm
ILS	Iterated Local Search
ITSRP	Integrated Truck Semi-Trailer Routing Problem
IVX	Identical Vehicle Crossover
LLP	Lead Logistics Provider
LNS	Large Neighbourhood Search

LP	Linear Programming
LRP	Location-Routing Problem
LSP	Logistics Service Provider
MA	Memetic Algorithm
MALNS	Multiple Adaptive Large Neighbourhood Search
MA-ALNS	Memetic Algorithm and Adaptive Large Neighbourhood Search
MD-PDPTW-SR	Multi-depot Pickup and Delivery Problem with Time Windows and Special Requests
MDVRP	Multi-depot Vehicle Routing Problem
MILP	Mixed Integer Linear Programming
MIP	Mixed Integer Programming
MLS	Multi-start Local Search
MTA	Modified Threshold Accepting
MTA-ALNS	Modified Threshold Accepting and Adaptive Large Neighbourhood Search
MV-PDP	Multi-vehicle Pickup and Delivery Problem
NP	Non Deterministic Polynomial Time Problems
OR	Operations Research
P	Polynomial Time Problems
PALNS	Parallel Adaptive Large Neighbourhood Search

PDP	Pickup and Delivery Problem
PDPTW	Pickup and Delivery Problem with Time Windows
PDRC	Pickup and Delivery Route Crossover
PSO	Particle Swarm Optimisation
SA	Simulated Annealing
SDVRP	Site-dependent Vehicle Routing Problem
SIH	Semi-trailer Insertion Heuristic
SREX	Selective Route Exchange Crossover
St.Thres	Start Threshold
TA	Threshold Accepting
TA-ALNS	Threshold Accepting and Adaptive Large Neighbourhood Search
TDABC	Time-Driven Activity-Based Costing
Thres	Threshold
TMS	Transportation Management System
TS	Tabu Search
TSP	Travelling Salesman Problem
TTRP	Truck and Trailer Routing Problem
TTVRP	Trucks and Semi-trailer Vehicle Routing Problem
VCPS	Vehicle Capacity Planning System
VNS	Variable Neighbourhood Search

VRP	Vehicle Routing Problem
VRPB	Vehicle Routing Problem with Backhauling
VRPPD	Vehicle Routing Problem with Pickups and Deliveries
VRPTW	Vehicle Routing Problem with Time Windows
UHGS	Unified Hybrid Genetic Search

Chapter 1

Introduction

The efficient transportation of goods plays an important role in the economy of nations. Logistics management require considerable attention to reduce the transportation costs. Advancements in logistics planning systems encourage industrial players to reduce the amount of money spent on distribution and transportation. These industrial players attempt to create competitive advantage over competitors in terms of cost leadership. Among IT tools, optimisation tools are among one of the most powerful equipment for logistics planning.

Optimisation tools have been applied most often to logistics and supply chain decision problems. One of the most important operational decisions related to transportation along a supply chain relates to the Vehicle Routing Problem (VRP). Transportation costs account for a significant amount of the total cost of a product. In some industries, such as food and drink, distribution costs at the transportation level can amount for up to 70% of the value added costs of goods according to Chopra and Meindl (2004). Indeed, the transportation process represents a relevant component (typically from 10% to 20%) of the final cost of the goods in general. Toth and Vigo (2002) pointed out that, in the last few decades, the use of optimisation packages, for managing the provision of goods and services in distribution system (based on Operations Research and Mathematical Programming techniques) has received considerable attention. The large number of real-world applications, both in North America and in Europe, has

shown that the use of computerized distribution systems produces substantial savings (generally from 5% to 20%) in the global transportation costs.

The trend of outsourcing logistics activities from manufacturers to logistics service providers (LSPs) and Third-party Logistics Providers (3PLs) has recently emerged. Transportation is one of the activities most frequently outsourced. A large number of companies have outsourced all of the functions that fall outside their core competencies to LSPs and 3PLs. These providers play a significant role in driving supply chains forward. For logistics service industries, the movement of goods is one of their core activities. A fleet of vehicles, which transports goods from origin(s) to destination(s), requires a large amount of resources: fuel, tyres and other consumable parts. For some companies, these resources account for up to 50% of the final cost of the service. It is therefore interesting to investigate how optimisation techniques can help reduce costs through effective transportation planning.

For freight transportation, door-to-door delivery and local courier services consider both pickup and delivery of shipments. Finding the minimum-cost routes of vehicles for these services falls into the category of Pickup and Delivery Problems (PDPs) in Operations Research (OR). Moreover, the PDPs are also embedded in the planning of handicapped-person transportation, automated guided vehicles and helicopter routing.

1.1 Motivation

Currently, the 3PLs and Logistics Service Providers (LSPs) market are competitive at the regional, national and international levels. Moreover, in the last few decades, fuel and operating costs have increased dramatically. To gain a competitive advantage, a large number of 3PLs and LSPs have sought to lower the cost of their operations, which has led to rising interest in the applications of Operations Research(OR). Owing to advances in computer technology, OR techniques have become more powerful, capable

of efficiently solving optimisation problems in reasonable time frame. Together with the development of the Geographic Information System (GIS) and the Global Positioning System (GPS), OR techniques turn transportation planning into reality.

Several companies and research centres have developed algorithms and software for transportation optimisation that solve vehicle routing problems and their variants. These includes Paragon[1], OPTRAK[2], and PROCOM[3]. The business needs, environment and characteristics of each logistics company are different, and routing and scheduling software must be customised to reflect these different requirements. Moreover, logistics outsourcing models evolve from time to time: from LSPs to 3PLs, from 3PLs to Fourth Party Logistics (4PLs). However, the optimisation problems embedded in these models nevertheless still require optimisation tools for streamlining their operations. Therefore, an in-depth understanding of PDPs and VRPs, the core optimisation problems, is necessary for developing a model and solution algorithm that responds to specific requirements and is flexible.

The Pickup and Delivery Problem (PDP) is a variant of the Vehicle Routing Problem (VRP). In VRP, the problem is to find a set of minimum cost routes for a fleet of vehicles to serve a number of customers. Each customer is visited once. Each vehicle starts and ends at the same depot. For PDP, every transportation request is associated with a pickup and corresponding delivery location. A vehicle must also depart from and return to the same depot.

Research on PDPs is relatively scarce, compared to the body of research that exists on VRPs. Possible reasons for this discrepancy include the fact that VRPs are embedded in the distribution planning problem of raw materials or products by own fleets. Although currently VRPs are more widely researched, the outsourcing of transportation task to LSPs and 3PLs, which must efficiently solve PDPs, has received more attention in the last few decades. Another reason PDPs may be neglected is that, compared

[1] www.paragonrouting.com
[2] optrak.com
[3] www.procomp.fi

to the VRP, PDPs are more difficult to handle due to their underlying problem constraints. There are many variants of PDPs, including heterogeneous fleet, maximum duration time, multiple vehicles, multiple depots, time windows, special requests and multi-dimensional capacity constraints. It is clear that gaps between theory and practice exist. Several metaheuristics have been applied to solve variants of PDPs. However, some authors argue that hybridised metaheuristics can improve the performance of pure approaches.

As a consequence, there remains ample room for the development of OR techniques to model and solve real-world optimisation problems especially PDPs. Moreover, PDPs are NP-hard problems for which no optimal algorithm running in polynomial time is expected to be found. The effort required for solving these problems increases dramatically with the problem size or the number of requests. Ropke (2005) emphasised that solution methods for rapidly changing business environments have to be fast, robust and precise. Moreover, these methods must be easy to apply to the specific problem and its variants. Gendreau and Tarantilis (2010) also confirmed that the performance of different algorithms on vehicle routing problems depends on efficiency, effectiveness, simplicity and flexibility.

1.2 Modelling and solution methods

Real-life routing problems incorporate practical complexities in addition to the classical VRPs and PDPs. Efficient modelling and solution methods are two major elements required to solve these problems. Classical models of VRPs and PDPs often oversimplify the problems occurring in real world. Modelling can be considered both an art and a science: real-life problems require a model that can tackle complexities arising in operational planning, the art of modelling comes into play when modellers face the challenging task of representing real-life constraints. The model should not be unnecessarily sophisticated. However, the problems tend to be complex in nature. The

model should be manageable for computer programming. The problems considered in this thesis are built up from the basic model and incorporates real-life constraints for modelling variants of PDPs. Efficient methods are then developed to implement these models with the view to obtaining good solutions in reasonable time.

The Travelling Salesman Problem (TSP), VRP and PDP are categorised into combinational optimisation problems where a large number of possible solutions are obtained depending on problem size. The basic VRP without capacity constraint is the TSP. In TSP, a salesman must visit n cities exactly once, and start from and return to the same city. The problem is to find a minimum distance route. The TSP is easy to understand but difficult to solve. To illustrate the growth rate due to problem size, with a symmetric TSP problem of size n, the number of possible solutions is $n!/2$. Then, if $n = 30$, the number of possible solutions is approximately 10^{32}. With a computer evaluating the cost of trillion (10^{12}) solutions per second, it approximately requires 10^{12} years to obtain all possible solutions. With $n = 31$, it then require over 10^{13} years or an increase of 31 times the possible solutions if just one city is added. It is evidenced that simple enumeration is prohibitive. The use of OR techniques is inevitable. There are two major solution methods, exact and heuristic approaches, that are capable of coping with these problems, but the issues of problem sizes toward computational time and precision still exist.

One issue of solving the TSP and VRPs is that the computational complexity clearly increases when the problem sizes become larger. This issue can be explained by means of a theoretical schema that involves the notion of "polynomially-bounded" algorithms. The problems which can be solved by the polynomially-bounded algorithms are denoted by P. In general, the problems in class P can be efficiently solved to optimality. Unlike class P, the class NP-hard is a large class of combinatorial problems for which no optimal algorithm running in polynomial time is expected to be found. Most vehicle routing and scheduling problems fall into the NP-hard class where efforts required for solving the problems increase exponentially with the problem sizes.

Mathematical models are idealised representations expressed in terms of a mathematical symbol and expression. Similarly, the mathematical model of a business problem is the system of equations and expressions that describe the essence of the problem. Decision variables, objective function, constraints and, parameters are used in the mathematical model. The problem is to choose the values of the decision variables so as to maximise or minimise the objective function, subject to the specified constraints. Exact methods can guarantee to obtain the optimal solution to a decision problem up to a certain size if sufficient time and space is given. Exact methods are clever in reducing the search space in order to find the optimal solution. However, for NP-hard problems embedded in transportation planning, obtaining solutions for large instances by exact methods are time-consuming. The resulting computational time may be unacceptably long and prohibitive for a rapidly changing business environment. Ropke (2005) solved PDP with time windows (PDPTW) by using the Branch-and-Cut-and-Price for up to 500 requests with varying success and time.

Heuristics cannot guarantee that the optimal solution is found. However, this solution method is much faster. Robust heuristics must be designed in order to find good feasible solutions. Researchers design heuristics to solve test problem instances and obtain competitively good solution quality within a reasonable amount of time. The limitation of heuristics is their capability of escaping from local optimum. In the last few decades, a special class of heuristics, called metaheuristics, has received considerable attention. Metaheuristics are equipped with mechanisms to jump from local optima to new points of search space. A metaheuristic can embed problem-domain specific heuristics within its general framework in order to solve many types of problems. Ropke and Pisinger (2006) solved for PDPTW using Adaptive Large Neighbourhood Search (ALNS) and obtained good solution quality within reasonable computational time for large-scale problems. Practitioners prefer using heurisitcs and metaheuristics to solve real-life problems of realistic sizes arising in industry.

1.3 Goals

The core problems studied in this thesis are Multi-depot Pickup and Delivery Problems (MDPDPs), which arise when several depots are considered as covering a large geographical area of customer locations. Many heuristics and metaheuristics have been applied to solve different variants of PDPs. This research further investigates the real-world application of PDPs' variants and is motivated by the fact that current practitioners in logistics businesses employ optimisation tools for managing their fleet with the view to minimising the total costs. Moreover, in order to expand their businesses to other regions, an efficiently computerised approach to transportation planning is required to support large-scale computations which occur repeatedly as well as to reduce human error and the reliance of businesses on highly-skilled employees to obtain solutions. The efficiency of an algorithm underpins key success factors of logistics businesses, such as cost, service level and flexibility. Aided by the optimisation technology, geographical coverage is also one of the key success factors that the logistics businesses are concerned. The real-life large geographical coverage problems have MDPDP at their core.

This thesis focuses on investigating several variants of MDPDPs and developing efficient methods for solving them. The models and algorithms are developed for different problem formulations by incorporating a number of constraints that vary from Chapter to Chapter. The goal is to tackle real-life routing problems arising in logistics businesses by efficiently implementing the proposed new algorithms developed in the thesis.

1.4 Thesis Overview

This section provides an overview of the thesis.

- Chapter 2 provides the literature review of Routing and Scheduling Problems, namely, Travelling Salesman Problems, Vehicle Routing Problems, Multi-depot Vehicle Routing Problems, Pickup and Delivery Problems, and Rich Vehicle Routing Problems. The solution methods for tackling the routing and scheduling problems are discussed. These include exact methods, heuristics, metaheuristics and hybrid meta-heuristics. For each method, the advantages and disadvantages are discussed, as well as its application in solving variants of routing and scheduling problems.

- Chapter 3 examines the Multi-depot Pickup and Delivery Problem (MDPDP), which involves serving a number of pickup and delivery locations using a heterogeneous fleet of vehicles located at several depots. This problem is formulated as a mixed-integer linear programming problem. The objective is to find minimum distance routes subject to precedence, capacity and maximum-route length constraints. This is an NP-hard problem and we use ILOG CPLEX for optimally solving instances of small size only. A Memetic Algorithm is proposed, implemented and computationally tested on various generated test instances. Competitive near-optimal solutions are reported. The work described in Chapter 3 was presented at the 24^{th} European Conference on Operational Research, 11-14 July 2010 in Lisbon, Portugal.

- Chapter 4 investigates a Multi-depot Pickup and Delivery Problem with Time Windows and Special Requests (MD-PDPTW-SR). The objective of this NP-hard problem is to minimise the routing cost of the fleet of vehicles serving transportation requests over a large geographical coverage area and subject to customer requirements. A new Adaptive Memetic Large Neighbourhood Search (AMLNS) is proposed by hybridising the Adaptive Large Neighbourhood Search (ALNS) and Memetic Algorithm (MA). The proposed meta-heuristic is computationally tested on standard benchmark instances from the literature. The computational results

are promising; the proposed heuristic is capable of obtaining improved feasible solutions for several instances. The work described in Chapter 4 was presented at the 9^{th} International Conference on Computational Management Science, 18-20 April 2012, Imperial College London, England, and a more enhanced version of the AMLNS was also presented at the 25^{th} European Conference on Operational Research, 8-11 July 2012, Vilnius, Lithuania.

- Chapter 5 demonstrates the development of the Adaptive Memetic Large Neighbourhood Search (AMLNS). The recombination process of key components in the selected meta-heuristics is used to evolve hybrid meta-heuristics. These hybrids include single-solution approaches, population-based approaches, and hybridisation between population- and single- solution approaches. The empirical investigation of hybrid meta-heuristics are statistically conducted. The emerging hybrid metaheuristic, the AMLNS, shows the promising results, as used in Chapter 4.

- Chapter 6 examine a routing problem arising in freight transportation. This problem is an extension of the MD-PDPTW-SR which takes into account the following additional constraints: semi-trailer assignment, sub-contraction, special requests for trucks and trailers, and multi-dimensional capacity constraints. We will refer to this as the Integrated Truck and Semi-trailer Routing Problem (ITSRP), which is widely used to model the essence of real-life routing problems in freight transportation. Schedulers have to plan the fulfilment of their requests not merely by routing and scheduling their own fleet, but also by selecting them to be outsourced to external carriers in some cases. The entire problem considers three fulfilment modes, namely, self-fulfilment, sub-contraction on request basis, and sub-contraction on tour basis. The ITSRP consists of finding a feasible routing and scheduling plan at minimal execution cost.

- Chapter 7 formulate the Integrated Truck and Semi-trailer Routing Problem (ITSRP) as a Mixed Integer Linear Program (MILP). The MILP is solved by CPLEX.

A set of test instances are generated to simulate the real-life ITSRP and are solved using the AMLNS and ALNS. New semi-trailer assignment operators for the AMLNS and ALNS are proposed. In addition, the route measures of the IVX are modified according to the changed objective function. The experiments of results demonstrate that the AMLNS is competitive compared to the ALNS.

- Chapter 8 summarises the research carried out in this thesis and highlights the thesis's contribution to the field. Suggestions for future work are put forward and aspects of computational implementation in practice are discussed.

Chapter 2

Literature Review

2.1 Routing and Scheduling Problems

Most routing and scheduling problems are combinatorial optimisation problems which involve the selection of a combination of customers to be visited by feasible routes. The objective of this section is to introduce several underlying routing and scheduling variants of the Multi-Depot Pickup and Delivery Problems (MDPDPs). These problems have been extensively studied in the literature for several decades. Therefore, a large number of solution methods have been maturely developed. This section will review the best performing methods available, some of which are used to solve the routing problems investigated in this thesis.

2.1.1 Travelling Salesman Problem (TSP)

The Travelling Salesman Problem (TSP) is the fundamental problem of all routing problems. Moreover, it is one of the most intensively studied problems in computational mathematics. The objective of solving TSP is to find a shortest tour through a given set of cities by visiting each city once and returning to the starting city. The TSP is proven to be a NP-hard problem. Despite this, a recent state-of-the-art exact al-

gorithm, the advanced Branch-and-Cut-and-Price algorithms of Applegate (2007), can optimally solve TSP instances of up to 85,900 cities. The research on TSP is somewhat saturated and regularly successful. The TSP is a generic core model that captures the combinatorial essence of most vehicle routing problems.

2.1.2 Vehicle Routing Problems (VRPs)

More than 50 years have elapsed since Dantzig (1959) introduced the VRP through its application within gasoline delivery. Laporte (2007) claimed that the TSP and VRP are two of the most popular problems in the study of combinatorial optimisation. VRP is a generalisation of the TSP, though cities and salesmen in the TSP can be seen as customers and vehicles in the VRP. Toth and Vigo (2002) discussed the basic version of VRP, Capacitated VRP (CVRP). In CVRP, all customers correspond to deliveries and the demand is deterministic, known in advance. The vehicles are identical and based at a single central depot, and only the capacity restrictions for the vehicles are imposed. The objective is to minimise the total cost of serving all of the customers. As an extension of TSP, the CVRP is known to be a NP-hard problem. However, Laporte (2007) emphasised that the VRP is practically more difficult to solve than a TSP of the same size. Since, VRPs consider practical constraints which add difficulties and complexities to solve the problems.

Toth and Vigo (2002) considered the main variants of the VRPs as Vehicle Routing Problem with Time Windows (VRPTW), Vehicle Routing Problem with Pickup and Delivery (VRPPD) and Vehicle Routing Problem with Backhauling (VRPB). In addition, Golden and Assad (1988) pointed out that the VRPs may be interlinked with other levels of planning decisions such as the Location-Routing Problem (LRP). The LRP simultaneously seeks an optimal facility location and route design, interrelating the routing problem and location-allocation problem. Vehicle routing problems can be represented by mathematical models such as Mixed Integer Linear Programming

(MILP). If time window constraints are imposed on the vehicle routing problem, it is called a Vehicle Routing Problem with Time Windows (VRPTW). A more complicated version of a time interval occurs when a task requires a specified number of service times over a certain duration of time, such as a week, and constraints can incorporate the pattern of days for serving those tasks. This problem is called a Period Vehicle Routing Problem (PVRP). Pickup and delivery tasks for the same vehicle explicitly determine the task precedence relationship. The vehicle routing problem with a precedence relationship is called a Vehicle Routing Problem with Pickup and Delivery (VRPPD). The Vehicle Routing Problem with Backhauling (VRPB) consists of line-haul customers, who require a given quantity of products to be delivered, and backhaul customers, from whom a given quantity of inbound product must be picked up. Other possible variants are a heterogeneous fleet, multiple depots and a precedence relationship between paired customer locations.

Mester and Bräysy (2005) proposed an effective metaheuristic algorithm for the VRPTW called Active Guided Evolution Strategies (AGES). The algorithm combines the strengths of the well-performing guided local search and evolution strategies. Computational experiments were carried out on 302 benchmark problems. The authors obtained improved feasible solutions in 86% of all test instances within a reasonable time.

Pisinger and Ropke (2007) presented a unified heuristic which is able to solve five different variants of VRPs: VRPTW, CVRP, MDVRP, SDVRP, and OVRP. All problem variants can be transformed into a Rich Pickup and Delivery Problem with Time Windows. The Adaptive Large Neighbourhood Search (ALNS) applied was able to improve 183 best known solutions out of 486 benchmark tests. The heuristic also shows promising results for a large class of vehicle routing with backhauls.

Vidal et al. (2013) proposed a Unified Hybrid Genetic Search (UHGS) metaheuristic for solving 29 vehicle routing variants, 42 benchmark instances sets, with 1099 instances overall. The UHGS combines four main optimisation methodologies: 1) hybridisation

of genetic algorithm with local search procedures; 2) the use of penalised infeasible solutions, managed through two distinct sub-populations during the search; 3) a solution representation without trip delimiters; 4) an advanced population management method with diversity-and-cost objective for solution evaluation. The UHGS matches or outperforms the current state-of-the-art problem-tailored algorithms. Overall, 1046 of the 1099 best known solutions have been either retrieved or improved.

2.1.3 Pickup and Delivery Problems (PDPs)

Lokin (1978) introduced the precedence constraints, which are required to formulate Pickup and Delivery Problems (PDPs) into the traditional TSP. Berbeglia et al. (2006) stated that PDPs are class of VRPs in which goods or passengers are transported between an origin and a destination. Lenstra and Kan (1981) have confirmed that the PDP is a NP-hard problem. Savelsbergh and Sol (1995) provided unified notation of most PDPs and a brief overview of existing solution methods until 1995.

Parragh et al. (2008) conducted a comprehensive survey on Pickup and Delivery Problems (PDPs). They classified the PDPs into two categories. The first category, Vehicle Routing Problems with Backhauls (VRPB), deals with the transportation of products from the depot to line-haul customers and from backhaul customers to the depot. The second category refers to transportation between customers where goods are moved from pickup to corresponding delivery locations, denoted as Vehicle Routing Problem with Pickups and Deliveries (VRPPD). Parragh et al. (2008) classified the VRPPD into two subclasses. The first subclass refers to situations where pickup and delivery locations are unpaired. In other words, identical products are considered. Each unit picked up could be used to fulfil the demand of any delivery customer. Both multi- and single-vehicle cases were studied in the literature. The second VRPPD subclass comprises the classical Pickup and Delivery Problem (PDP) and the Dial-A-

Ride Problem (DARP). Both types consider transportation associated with an origin and a destination, resulting in pairs of pickup and delivery points. The PDP deals with the transportation of goods while the DARP is concerned with passenger transportation. This difference is usually expressed in terms of additional constraints or objectives. However, the mathematical formulation of DARP and PDP shares some characteristics and can be used interchangeably. In this thesis, the PDP is focused because of our interest in its various applications for transportation companies.

Li and Lim (2001) proposed a metaheuristic with an annealing-like restart strategy to guide the local search in three neighbourhoods and solve the general m-PDPTW. A K-restarts annealing procedure with tabu-list is applied to avoid cycling in the search process. The authors generated 56 problem instances of 100 customers from Solomon's benchmark instances for VRPTW. The computational experiments on six different data sets show that the algorithm is efficient for solving practical-sized multiple PDPTW problem instances with various distribution properties.

Several authors proposed efficient metaheuristics to solve the variants of PDPTW, such as ALNS of Ropke and Pisinger (2006) and Memetic Algorithm of Nagata and Kobayashi (2010). These authors were able to improve the best known solutions of Li and Lim's (2001) benchmark instances.

2.1.4 Rich Vehicle Routing Problems

Hasle and Kloster (2007), the founders of a generic VRP solver, discussed the variants of rich VRPs for industrial applications. The strengths of the generic VRP solver, SPIDER, are its modelling flexibility and solution quality. The variants were categorised into: (1) fleet, vehicle, driver; (2) depots, tours, start, and stop locations; (3) order types, type of operation; (4) distances, travel times, and service times; (5) waiting time, time windows and capacity constraints; (6) idiosyncratic constraints and objec-

tives; (7) stochastic and dynamics and (8) response time. The authors confirmed that, for industrial problem sizes, the heuristic approach is the only viable approach. The SPIDER heuristic approach is based on Local Search using construction, tour depletion, and iterative improvement. Several variants of VRPs such as CVRP, DVRP, VRPTW, PDPTW, FSMVRPTW were tested. The SPIDER found new best known solutions and obtained competitive results in comparison to other state-of-the-art heuristics.

Goel (2008) introduced a general model capable of handling the complexities evolving from various characteristics arising from real-life vehicle routing problems. The model is termed the *General Vehicle Routing Problem* (*GVRP*). The real-life requirements include the employment of external carriers, route restriction, pickup and delivery requests, drivers' working hours. Fleet-telematics, dynamic VRP and Large Neighbourhood Search algorithms were also discussed. The number of case studies and computational studies was investigated.

Drexl (2012) categorised the dimensions of richness in VRPs according to requests, fleet, route structure, objectives and scope of planning. The author presented an overview of these dimensions of richness in real-world VRPs, as shown in 2.1. Trends in VRP research move toward the richer problems and more robust solution methods that work well for a broad range of problems both in terms of running time and solution quality. The author claimed that the most successful heuristics are so-called hybrid procedures that combine several classical ones. Self-adaptation and hyperheuristics, metaheuristics and parallel algorithms are among the active areas of VRPs' research.

Schmid et al. (2013) provided basic models for the related variants of VRPs in the context of supply chain management. These include lot-sizing, scheduling, packing, batching inventory and inter-modality. The mathematical models and solution methods were also discussed.

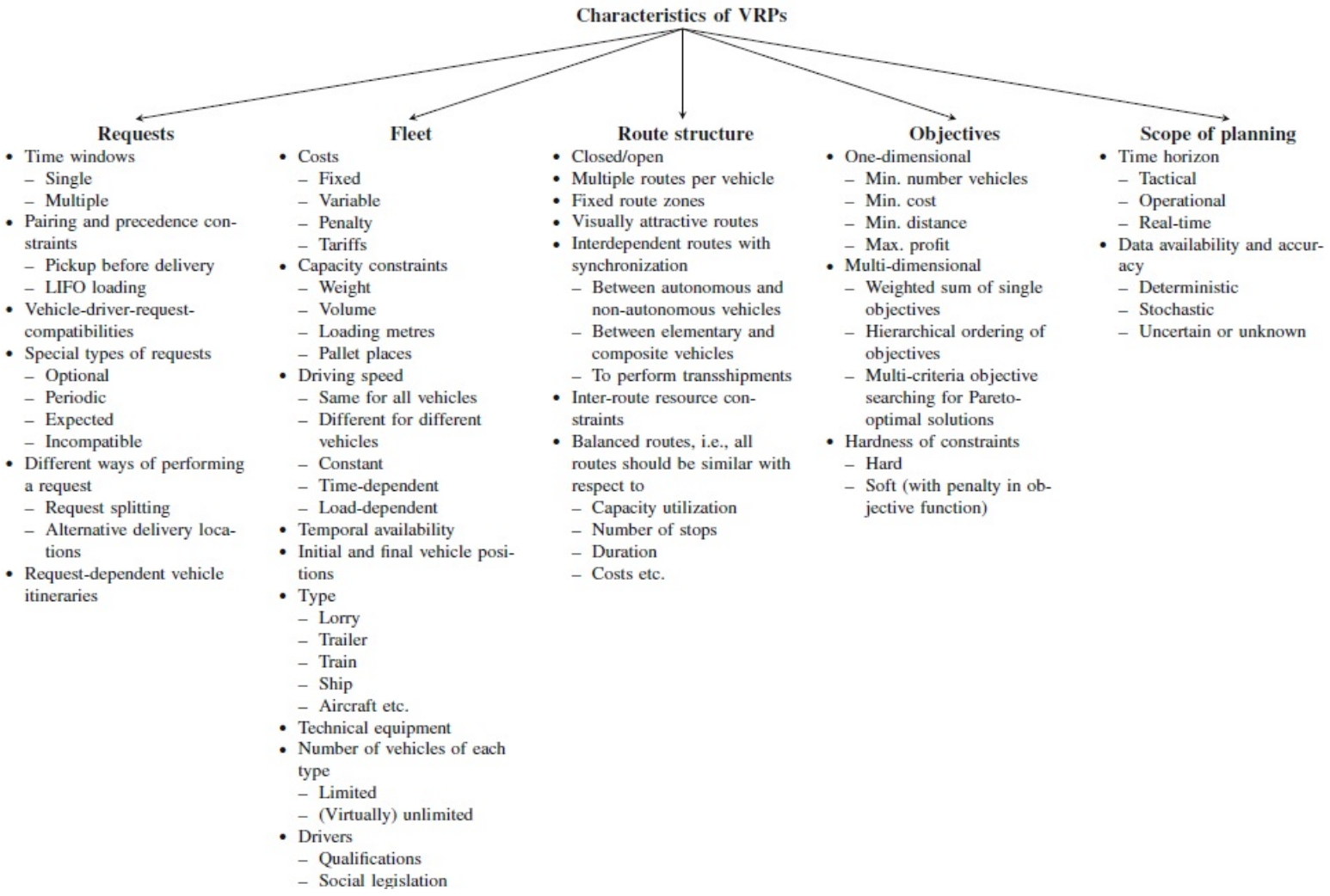


Figure 2.1: Dimensions of richness in VRPs (Drexl,2012:p.49)

2.2 Logistics Outsourcing Models

Optimisation techniques are developed alongside logistics outsourcing models. The evolution of logistics outsourcing plays an important role in advancing models and algorithms by motivating researchers to respond to industrial needs. Routing and scheduling problems are embedded in the planning process of these models. In this section, the terminology and routing and scheduling problems of logistics outsourcing models are described.

Langley et al. (2000) conducted a comprehensive study of the use of third-party logistics (3PL) services in the United States. This comprehensive study had been annually reported until 2013. Regarding logistics outsourcing models, the study revealed at one end of the spectrum, clients keep their logistics in-house, or so-called insourcing. However, once a client made the decision to outsource logistics, the outsourced services, geographic coverage and expected benefits were all over the map. In Figure 2.2, Langley et al. (2004) illustrated the changes in key attributes as the 3PL relationship models evolve in Figure 2.2.

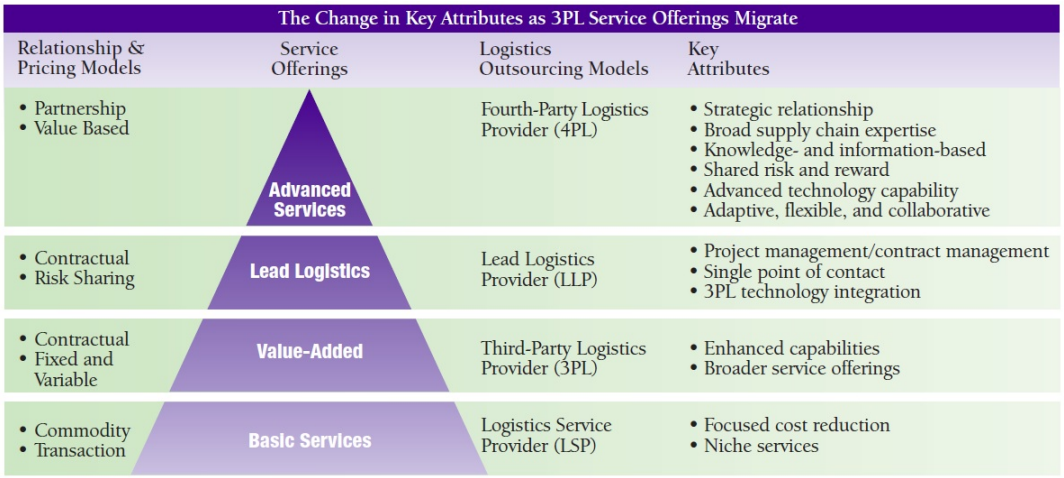


Figure 2.2: Logistics Outsourcing Models(Langley et al. 2004, p.23)

Gattorna, Selen and Ogulin (2004) defined the terms LSP, 3PL, LLP, 4PL as follows:

LSP Logistics Service Provider: any organisation that provides a range of logistics service capabilities to participating members of industry supply chains.

3PL Third Party Logistics: an external party that performs all or part of the corporate logistics activities on behalf of the shipper, such as transportation, warehousing and inventory management

LLP Lead Logistics Provider: a service provider that combines and utilises advanced capabilities to optimise logistics and supply chain activities across multiple (subordinate) 3PLs/LSPs

4PL Fourth Party Logistics: a new business model, integrating resources, capabilities and the technology of the lead enterprise(s) and other organisations with complementary capabilities, to design, build, and run comprehensive supply chain solutions.

In Figure 2.3, Langley et al. (2009) showed the evolution of the logistics service provider from a traditional model offering individual, mostly execution-based, services, through the 3PL and 4PL/lead logistic provider model to a full orchestrator of supply chain services.

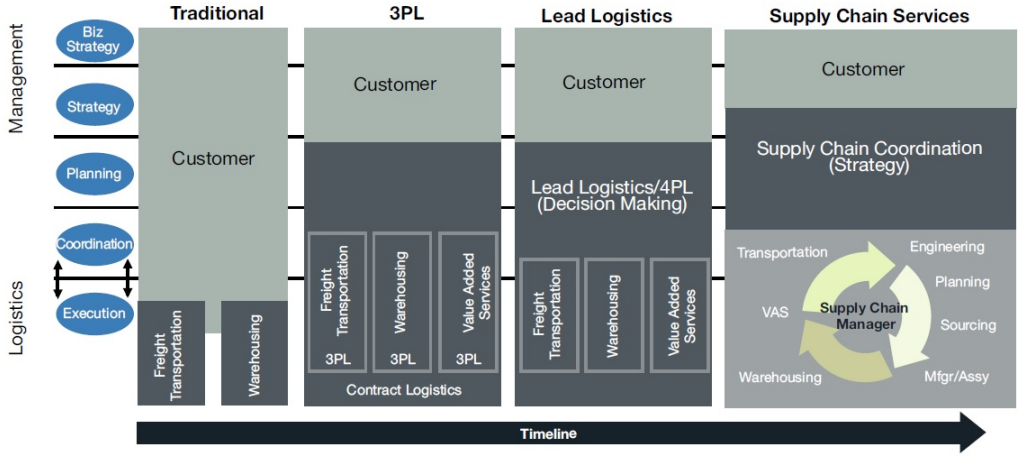


Figure 2.3: 3PLs Are Evolving Into Supply Chain Orchestrators(Langley et al. 2009, p.34)

In Figure 2.3, a full orchestrator co-develops a supply chain coordination strategy in concert with the shipper and then manages the complete cycle of supply chain activities. When logistics providers evolve from one stage to the next, they assume more control and responsibility for the customer's supply chain.

Bhatti et al. (2010) mentioned that an increasing number of 3PLs led to chaos of another kind. The LLP is a 3PL with advantages of scale and other abilities which allow it to act as the lead 3PL. It serves as a single point of contact in regards to the organisation and all the 3PLs it has hired. The LLP may or may not have assets e.g. trucks, but they are capable of integrating and co-ordinating the activities of the other 3PLs. Gattorna, Ogulin and Selen (2004) argued that LLPs are primarily the same as 3PLs providers but equipped with extra visibility tool, optimisation modelling for decision support purposes, and payment rewarded by a fee and tariff, linked to some mathematical modelling of costs, and corresponding benefits. For example, i2 technologies of JDA[1] are used for LLPs' Transportation Management System (TMS). The Load Planning and Optimisation feature of the TMS enables users to plan multi-site, supports for private and third-party carrier environments and provides advanced carrier selection. These functions can improve asset utilisation, service levels, and planner productivity. Moreover, they can reduce freight expenses and incremental costs.

Vehicle routing and scheduling are the core transportation planning problems for insourcing. When considering large LSPs, 3PLs, and LLPs, the Multi-Depot Pickup and Delivery Problem or its variants are embedded.

2.3 Solution Methodology

In the 1950s, Dantzig and Ramser (1959) proposed the first mathematical programming formulation and algorithmic approach for the solution of the gasoline delivery problem. A few years later, several exact algorithms and heuristic algorithms were proposed for

[1] www.jda.com

the optimal and approximate solution of different version of the VRP. Exact algorithms and heuristics are two major classes of solution methods for solving variants of TSP, VRPs and PDPs. A special class of heuristics that has been successful in the last two decades is meta-heuristics. Moreover, over the last few years, hybrid meta-heuristics have received special attention for solving practical problems. These solution methods are described in the following order: exact methods, heuristics, metaheuristics and hybrids.

2.3.1 Exact Methods

The Vehicle Routing Problem and its variants can be formulated as Integer Programming (IP), Mixed Integer Programming (MIP) and Mixed Integer Linear Programming (MILP). Due to the fact that these problems are all NP-hard, the number of finite solutions in the enumeration procedure for finding an optimal solution can be very large. Hillier and Lieberman (2010) suggested that any enumeration procedure should be cleverly structured so that only a fraction of feasible solutions needs to be examined. Toth and Vigo (2002) stated that many exact approaches for the capacitated VRP (CVRP) are inherited from the extensive and successful work done for the exact solutions of TSP. The most effective exact approaches for CVRP are mainly Branch-and-Bound algorithms. The Branch-and-Cut algorithm has been extremely successful in finding optimal solutions for large instances of TSP. It mainly uses a combination of three kinds of techniques: automatic problem pre-processing, the generation of cutting plane and clever branch-and-bound. The PDPTW and variants studied in this thesis involve solving MILPs. A MILP formulation is shown in Figure 2.4.

$$\begin{aligned} \min_{\mathbf{x},\mathbf{y}} \quad & z \triangleq \mathbf{c}^{\mathsf{T}}\mathbf{x} + \mathbf{d}^{\mathsf{T}}\mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{E}\mathbf{y} \left\{ \begin{array}{c} \leq \\ = \\ \geq \end{array} \right\} \mathbf{b} \\ & \mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}, \quad \mathbf{y} \in \{0,1\}^{n_y} \end{aligned}$$

Figure 2.4: Standard Mixed-Integer Linear Programming (MILP) Formulation

Simplex Method is a general and algebraic procedure for solving linear programming problems. The MILP partially consists of continuous decision variables in linear programming. Hillier and Lieberman (2010) summarised the Simplex Method as a method comprised of initialisation, optimality test and iteration. The Simplex Method is extremely efficient in practice. However, one key limitation that prevents many more applications is the method's assumption of divisibility, which requires non-integer values to be used for decision variables. In many practical problems, decision variables have some integer values. The Simplex Method is applied to MILPs in solving the LP relaxation problem, obtained by deleting the integer restriction. Several exact algorithms such Branch-and-Bound, Branch-and-Cut, Branch-and-Price methods use a sequence of LP relaxations to solve the overall MILP problem efficiently. In order to solve optimisation problems by exact methods, IBM ILOG CPLEX optimiser [2], an elite state-of-the-art software package, is widely used. The CPLEX provides flexible, high-performance mathematical programming solvers for linear programming, and mixed integer programming.

Branch-and-Bound (B&B) approach applies the concept of divide and conquers to solve integer programming and mixed integer programming. Hillier and Lieberman (2010) suggested that the original "large" problem is too difficult to be solved directly; it is, therefore, divided into smaller and smaller sub-problems until these sub-problems can be conquered. Dividing (branching) is done by partitioning the entire set of feasible solutions into smaller subsets. After branching, new sub-problems are generated. Then,

[2] http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

the LP relaxation of these problems is solved by the Simplex Method to obtain an optimal solution. Conquering (fathoming) is carried out partially by bounding how good the best solution in the subset can be and then discarding the subset if its bounds indicate that it cannot possibly contain an optimal solution for the original problem. The algorithm stops when all nodes of the search tree are either pruned or solved. Toth and Vigo (2002) presented B&B algorithms for the capacitated VRP.

Branch-and-Cut (B&C) is equipped with the capability of solving very large problems. Hillier and Lieberman (2010) discussed early reports of solving as many as a couple thousand variables. B&C mainly uses a combination of three kinds of techniques: automatic problem pre-processing, cutting plane generation, and clever Branch-and-Bound techniques. The automatic problem pre-processing involves a "computer inspection" of the user-supplied formulation to spot reformulations which fall into three categories: fixing variables, eliminating redundant constraints and tightening constraints. The generation of cutting planes can reduce the feasible region for the LP relaxation without eliminating feasible solutions. The cutting plane is a new functional constraint that reduces the feasible region for the LP relaxation without eliminating feasible solutions for the MILP. The B&C approach generates many cutting planes before applying clever branch-and-bound techniques. As a result, LP relaxation is tightened. In other words, Ropke (2005) described that the B&C method is to simply to generate valid, violated inequalities throughout the branch and bound tree and not only in the root node. The valid inequalities are typically selected from some preselected families of valid inequalities. Hillier and Lieberman (2010) pointed out that the combination of cutting planes and branch-and-bound techniques provides a powerful algorithmic approach for solving large-scale problems. Ropke (2005) proposed Branch-and-Cut Algorithms for solving the PDPTW efficiently.

Branch-and-Price (B&P) focuses on column generation to solve large scale Mixed Integer Programming (MIP) problems. Barnhart et al. (1998) discussed that the philosophy of Branch-and-Price is similar to that of Branch-and-Cut except that the pro-

cedure focuses on column generation rather than row generation. Cutting and pricing are complementary procedures for tightening an LP relaxation. A sub-problem, called the pricing problem, is a separation problem for the dual LP that is solved in order to identify columns to enter the basis. If such columns are found, the LP is reoptimised. Branching occurs when no column price out to enter the basis and the LP solution does not satisfy the integral conditions. Branch-and-Price allows column generation to be applied throughout the Branch-and-Bound tree. It is also possible to combine the Branch-and-Cut and Branch-and-Price paradigms, the so-called Branch-and-Cut-and-Price algorithm, to obtain even tighter lower bounds. Ropke and Cordeau (2006) introduced a Branch-and-Cut-and-Price algorithm with additional valid inequalities for PDPTW.

Dynamic Programming (DP) provides a systematic procedure for determining the optimal combination of decisions. Hillier and Lieberman (2010) described the DP as an approach designed to find the optimal policy for the overall problem, i.e., a prescription of the optimal policy decision at each stage for each of the possible states. A problem can be divided into stages, with a policy decision required at each stage. DP requires formulating an appropriate recursive relationship for each individual problem. The solution procedure starts at the end and moves backward stage by stage- each time finding the optimal policy for that stage- until it discovers the optimal policy starting at the initial stage and yielding an optimal solution for the entire problem. Applying the DP produce significant computational savings of time and space over using exhaustive enumeration. Hang et al. (2003) solved linear relaxation to optimality by using an exact dynamic programming algorithm to solve the sub-problem exactly in the practical pickup and delivery problem.

2.3.2 Heuristics

Laporte (2000) and Toth and Vigo (2002) classified the heuristic methodology of VRP into either classical heuristic or meta-heuristic approaches. Toth and Vigo (2002) expressed that even though classical heuristics perform a relatively limited exploration of the search space, they typically produce good quality solutions within a reasonable time. Moreover, most classical heuristics can be easily extended to account for the diversity of constraints encountered in real-life contexts. The solutions obtained by classical heuristics, however, are sometimes trapped in local optima, as shown in Figure 2.5.

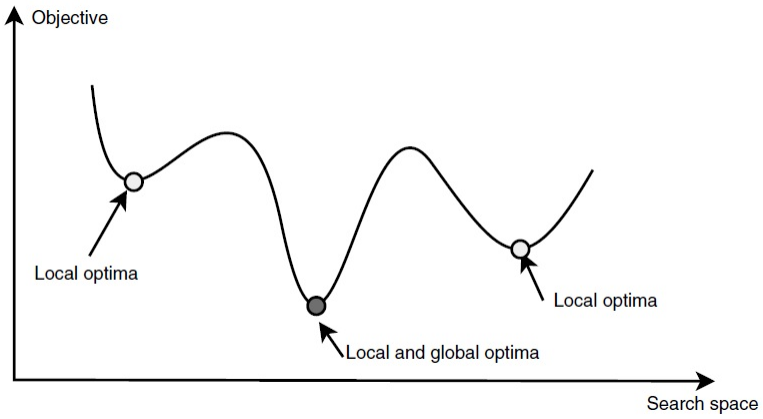


Figure 2.5: Local optimum and global optimum in a search space (Talbi 2009, p.91)

Figure 2.5 represents the local optimum and global optimum in a search space according to Talbi (2009). For Figure 2.5, Talbi (2009) stated that a solution $s \in S$ is a local optimum if it has a better quality than all its neighbourhood; that is, $f(s) \leq f(s^{'})^2$ for all $s^{'} \in N(s)$. However, classical heuristics have no mechanism to jump from local optima to a new point of search.

Hosny (2010) stated that heuristic algorithms refer to the experience-based, common-sense approach to problem solving. The search for a good problem solution is usually divided into two phases: construction and improvement. The construction refers to the

process of creating one or more initial feasible solutions that will act as a starting point. The improvement attempts to modify the starting solution(s) to better solutions.

Talbi (2009) described that, Local Search starts with a given initial solution. At each iteration, the heuristic replaces the current solution by a neighbour that improves the objective function as shown in Figure 2.6. In Figure 2.6, the search stops when all candidate neighbours are worse than the current solution i.e. local optimum is reached.

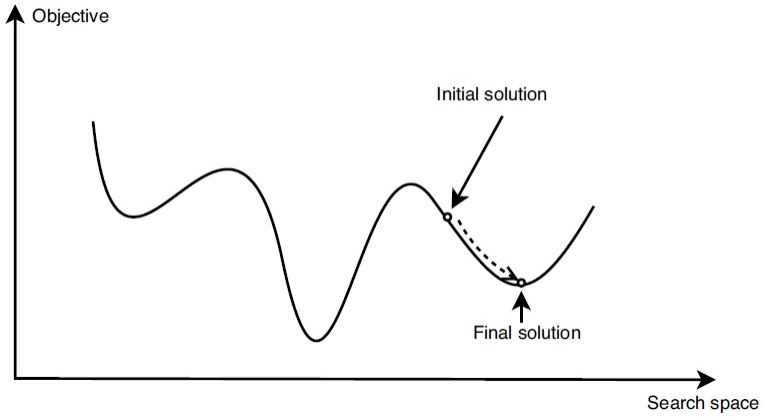


Figure 2.6: Local search (steepest descent) behavior in a given landscape (Talbi 2009, p.122)

Solomon (1987) presented several construction heuristics for solving vehicle routing and scheduling problems with time windows. The tour-building heuristics can be categorised into sequential and parallel methods. Sequential procedures construct one route at a time until all customers are scheduled, while parallel procedures construct a number of routes simultaneously. The number of construction heuristics was discussed, namely, Saving Heuristics, Time-oriented-Nearest-Neighbour Heuristic, Insertion and Time-Oriented Sweep Heuristic. In addition, the time feasibility condition as a reduction rule of improving computational time was shown. The computational results indicate that the insertion I1 heuristic proved to be very successful. Potvin and Rousseau (1993) discussed a parallel version of heuristic I1 proposed in Solomon (1987). First, each route is initialised with a different "seed" customer. Then, the remaining unscheduled customers are sequentially inserted into any route until all customers are routed. Liu and Shen (1999) confirmed that several parallel heuristics are better per-

formed than sequential insertion heuristics. Regret heuristic is one of the well-known parallel insertion heuristics, developed by Potvin and Rousseau (1993), for solving the VRPTW. The regret-heuristic is equipped with a kind of "look-ahead information" so that the placement of hard requests is not postponed to the late iterations. Ropke and Pisinger (2006) and Pisinger and Ropke (2007) applied different regret heuristics for effectively solving variants of VRPs.

For improvement procedures, Potvin (1996) mentioned that among the local improvement procedures, the $k - opt$ exchange heuristics are the most widely used, in particular, the 2-opt, 3-opt, Or-opt, and Lin-Kernighan heuristics. Potvin (1996) illustrated an example of a 2-opt exchange in Figure 2.7.

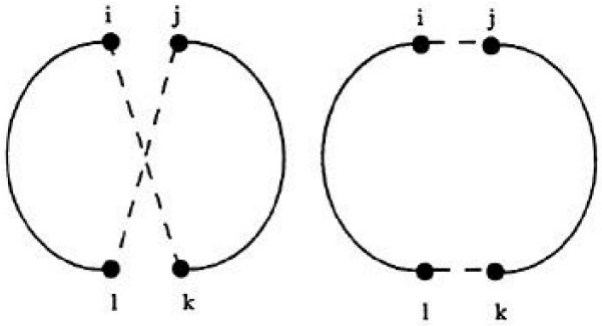


Figure 2.7: Exchange of links (i,k),(j,l) for links (i,j), (k,l)

In Figure 2.7, the paths (i,k) and (j,l) are selected for 2-opt operator. The paths are then exchanged and become $(i,j),(k,l)$. It is clear that the total distance may improve from applying this operator. Typically, exchange heuristics are applied iteratively until a local optimum is found. In other words, a tour cannot be improved further via the exchange heuristic under consideration. The 3-opt, Or-opt, and Lin-Kernighan heuristics are extensions of the 2-opt operator.

However, heuristics are often trapped in local optimum. Therefore, several mechanisms that are capable of escaping from local optimum have been developed in the last decades. These mechanisms are embedded in meta-heuristic approaches equipped with ways to jump from local optima effectively and seen as a class of heuristics.

2.3.3 Meta-heuristics

Hillier and Lieberman (2010) stated that the meta-heuristic approach is a general kind of solution method that organises the interaction between classical heuristic procedures and higher level strategies to create a process that is capable of escaping from local optima and performing a robust search of a feasible region. Different meta-heuristic methods execute the escape in different ways. Talbi (2009) discussed that two contradictory criteria namely diversification and intensification must be taken into account. The diversification refers to the exploration of the search space while the intensification refers to the exploitation of the good solutions found. Promising regions are the search space where good solutions are located. In intensification, the promising regions are explored more thoroughly in order to find better solution. In diversification, non-explored regions should be visited to ensure all regions of the search space are explored. Blum and Roli (2003), Talbi (2009), and Ombuki and Hanshar (2009) confirmed that finding a good balance between diversification and intensification is essential for meta-heuristics. The metaheuristic must both quickly identify regions in the search space with high quality solutions and not to waste too much time in regions of the search space which are either already explored or which do not provide high quality solutions.

Blum and Roli (2003) discussed different ways to classify and describe metaheuristics such as nature-inspired vs. non-nature inspired and population-based vs. single point search. Blum and Roli (2003) and Talbi (2009) described how the population-based and single point search are classified by the number of solutions used at the same time. Metaheurisitcs based on a single solution are called trajectory methods and encompass local search based algorithms, such as Simulated Annealing, Tabu Search, Iterated Local Search, and Variable Neighbourhood Search. Single-solution approaches work on a single solution at each time-step, describing a curve in the search space. In contrast, population-based metaheuristics perform a search process which describes the evolution of a set of points in the search spaces. The population-based meta-heuristics that are

widely applied are Genetic Algorithms (GA), Ant Colony Optimisation (ACO), and Particle Swarm Optimisation (PSO).

Single-solution based Metaheuristics

Simulated Annealing (SA), introduced by Kirkpatrick et al. (1983), is a widely used meta-heuristic approach that has been successfully in tackling many combinatorial optimisation problems. Talbi (2009) explained that SA applies the concepts of statistical mechanics whereby the annealing process requires heating and slowly cooling a substance to reach a strong crystalline structure. The SA is a stochastic algorithm enabling under some conditions the degradation of a solution. The objective is to escape from local optima and delay the convergence.

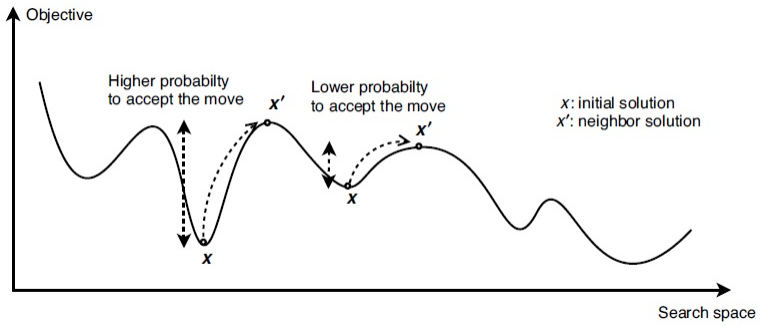


Figure 2.8: Simulated annealing escaping from local optima (Talbi 2009, p.127)

Figure 2.8 shows the way SA escapes from local optima according to Talbi (2009). The author explained that "the higher the temperature, the more significant the probability of accepting a worst move". At a given temperature, the lower the change of the objective function, the more significant the probability of accepting the move. A better move is always accepted. The temperature is steadily decreased according to a cooling schedule. At the end of the search, few non-improving solutions are accepted. Other

similar methods include Threshold Accepting, Record-to-Record Travel, the Great Deluge Algorithm and Demon Algorithms. Ropke and Pisinger (2006) and Pisinger and Ropke (2007) applied SA and obtained a good solution quality for many variants of VRPs and PDPTW.

Tabu Search (TS) is a well-known meta-heuristic method introduced by Glover (1986). Glover and Kochenberger (2003) explained that TS accepts non-improving moves to escape from local optima when all neighbours are non-improving solutions. When a better neighbour is found, it replaces the current solution. However, when the local optimum is reached, the search carries on by choosing a candidate that is worse than the current solution. The best solution in the neighbourhood is selected as the current solution even if non-improving solutions are found. The TS introduces the concept of tabu list to avoid cycles. Tabu list constitutes the short-term memory that manages a memory of the solutions or moves recently applied. It can avoid cycles by discarding the neighbours that have been previously visited. It memorises the recent search trajectory. In addition, medium-term memory can be applied to intensify the search. Moreover, the long-term memory can be applied for diversification. Glover and Kochenberger (2003) confirmed that the TS practically provides solutions very close to optimality and is among the most effective ways to tackle NP-hard problems. Cordeau et al. (1997) also applied the Tabu Search to efficiently solve the MDVRP.

Iterated Local Search (ILS) is a simple but effective meta-heuristic, introduced by Martin et al. (1991). Stutzle (1999) described how the ILS applies local search techniques to an initial solution until it reaches a local optimum. Then, it perturbs the solution before restarts the local search. Perturbation is a mechanism used to escape from the basis of the attraction of the local optimum. The acceptance criterion is the conditions that the new local optimum must satisfy to replace the current solution. Talbi (2009) illustrated the principle of ILS in Figure 2.9.

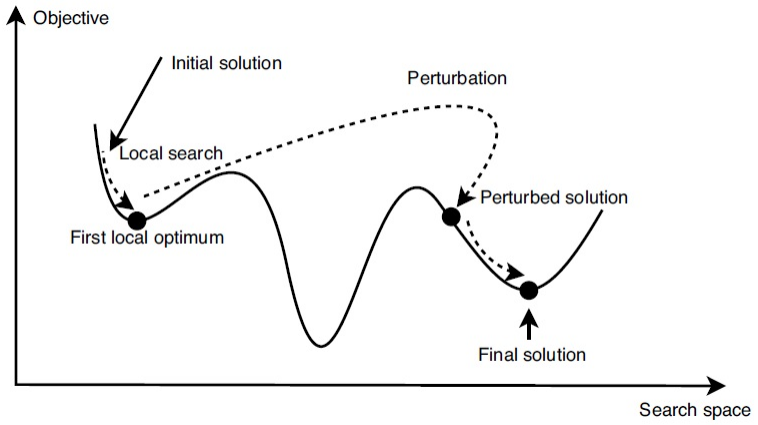


Figure 2.9: Iterated Local Search escaping local optimum (Talbi 2009, p.147)

Talbi (2009) suggested, "the perturbation operator may be seen as a large random move of the current solution". The operator should keep some part of the solution and perturb strongly another part of the solution in order to move, hopefully, to another basin of attraction. Blum and Roli (2003) argued that too small a perturbation might not enable the system to escape from local optimum. However, too strong a perturbation would make the algorithm similar to a random restart local search. Ibaraki et al. (2005), Ibaraki et al. (2008), Hashimoto et al. (2006), Hashimoto et al. (2008) and Subramanian (2012) applied the ILS to effectively solve several variants of VRPs.

Variable Neighbourhood Search (VNS) was introduced by Mladenovic and Hansen (1997). The authors described how the VNS framework provides a systematic change of neighbourhood in a local search algorithm. It increasingly explores distant neighbourhoods of the current incumbent solution and escapes from the solution to a new one if and only if an improvement has been made. In this way, favourable characteristics of the incumbent solutions are already at their optimal value and may be kept and used to obtain promising neighbourhoods. In addition, a local search routine is applied repeatedly to move from these neighbouring solutions to local optima. Blum and Roli (2003) explained that VNS's main cycle is composed of three phases: shaking, local search and move. The objective of the shaking phase is to perturb the solution to

obtain a new good starting point for the local search. The choice of neighbourhoods of systematically increasing cardinality yields a progressive diversification. Polacek et al. (2004) and Kytojoki et al. (2007) applied the VNS for solving variants of VRPs.

Population-Based Metaheuristics

Genetic Algorithm (GA) was first developed by John Holland in the 1970s to investigate the adaptive processes of natural systems. The concept of GA is to simulate the biological evolution through natural selection, crossover, mutation, and survival of the fittest in living organisms. Goldberg (1989) described how a *population* of strings or *chromosomes* is used to represent the solutions. The chromosomes are evaluated according to their objective function of $fitness$ and are selectively mated to reproduce $offspring$ through the use of genetic operators: crossover, mutation. The recombination of strings is operated by *crossover* allowing a rapid exploration of the search space by producing large jumps while attempting to improve the fitness of offspring. $Mutation$ allows a small amount of random search to a single chromosome to maintain the diversity in the population. The survival of the fittest or *replacement* ensures that the overall solution quality increases from one generation to the next generation. Blum and Roli (2003) discussed the use of a population and mutation ensures an exploration of the search space. Although, the selection, crossover, and replacement constitutes the exploitation of good solutions, the intensive use of these operators can cause *premature convergence* or the lack of population diversity. Goldberg (1989), Rocha and Neves (1999), and Lozano et al. (2008) confirmed that the population is crucial to a GA's ability to explore the search space. The variants and extensions of GAs include Evolution Strategies, Evolutionary Programming, and Genetic Programming. Thangiah and Salhi (2001) and Baker and Ayechew (2003) applied GAs to solve variants of VRPs effectively.

Ant Colony Optimization (ACO) is a recent meta-heuristic introduced by Marco Dorigo and his colleagues in the 1990s. Dorigo and Stutzle (2004) stated that the concept of ACO is to imitate the cooperative behaviour of real ants performing complex tasks such as transporting food and finding the shortest paths to food sources. A chemical trail or pheromone is left on the ground to guide the other ants toward the target points. The communication mechanism is that the larger the amount of pheromone on a particular path, the larger the probability that other ants select the path. For a set of ants, paths are chosen according to the smelt quantity of pheromone. Goss et al. (1989) illustrated the process of an ant colony searching for an optimal path between their food and their nest in Figure 2.10.

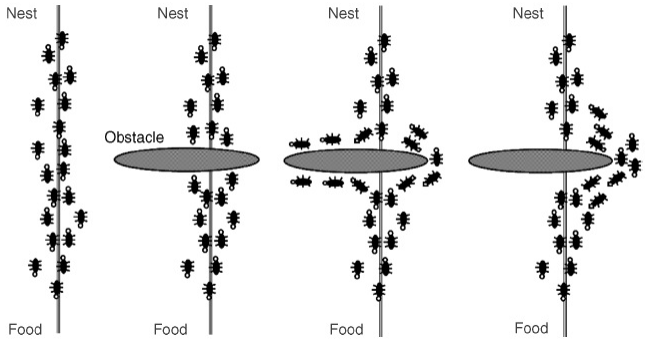


Figure 2.10: Inspiration from an ant colony searching an optimal path between the food and the nest (Talbi 2009, p.241)

Figure 2.10 illustrates an experiment carried out by Goss et al. (1989), as shown in Talbi (2009): when ants face an obstacle on the paths, with less travel time, the ants will end up leaving a higher level of pheromone. The higher the pheromone trail that is left, the more other ants follow. Eventually, the shortest path is selected. Blum and Roli (2003) suggested that the component of ACO managing the update of pheromone values has the effect of changing the probability distribution for sampling the search. The component of ACO is guided by the objective function and also influenced by a function applying the pheromone evaporation. This component is to intensify the search while there is a diversifying component that depends on the greediness of the

pheromone update. Pellegrini et al. (2007) and Yu et al. (2009) applied the ACO for solving variants of VRPs.

Particle Swarm Optimisation (PSO) is a population-based search method proposed by Kennedy and Eberhart (1995). The authors described that the PSO simulates the social behaviour and movement of natural organisms such as bird flocking and fish schooling to find a place with enough food. Talbi (2009) illustrated the decision space of particle swarm in Figure 2.11.

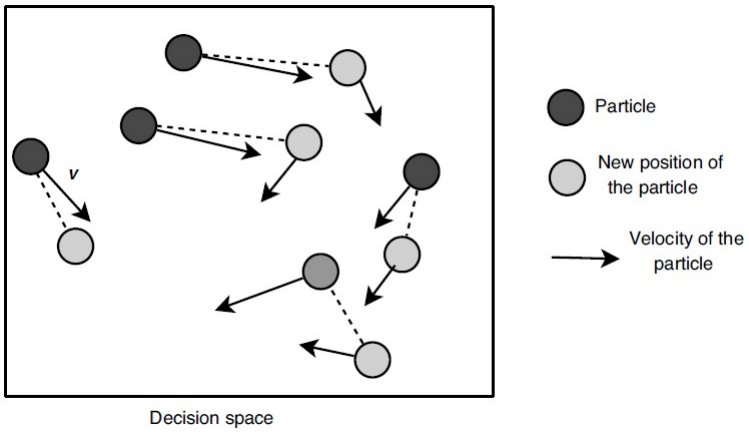


Figure 2.11: Particle swarm with their associated positions and velocities

In Figure 2.11, Talbi (2009) explained that, at each iteration, a particle moves from one position to another in the decision space. Optimisation takes advantage of the cooperation between the particles. The success of some particles will influence the behaviour of their peers. Each particle successively adjusts its position toward the global optimum according to the two conditions: the best position visited by it and the best position visited by the whole swarm. Particle neighbourhood define the social influence or the degree of communication between the particles. Using large neighbourhoods, more individuals are attracted to the best global solution: large neighbourhoods encourage the intensification of the search toward the best global solution. When using small neighbourhoods, more diversification of the search space is carried out. Ai and Kachitvichyanukul (2009) applied the PSO to solve the VRP with simultaneous Pickup

and Delivery Problem.

2.3.4 Hybrid Meta-heuristics

Over the last few years, hybrid meta-heuristics has received more attention, due to its success in solving combinatorial optimisation problems, and is now seen as a class of heuristics. Blum et al. (2011) explained that the main motivation behind the hybridization of different algorithms is to exploit the complementary strengths of different optimisation strategies and gain the advantage from synergy effect. The authors emphasised that developing an effective hybrid approach is in general a difficult task which requires expertise from different areas of optimisation. Talbi (2009) discussed that the design of hybrid metaheuristics involves issues such as functionality and architecture of the algorithm. Blum and Roli (2003) suggested that one of the most popular means of hybridisation concerns the use of trajectory methods in population-based methods, such as GAs and ACOs using local search procedures. The population-based methods are better in identifying promising areas in the search space, while trajectory methods are better in exploring promising areas in the search. Blum et al. (2011) discussed that the current state of research does not provide conclusive answers about appropriate hybrid metaheuristics working well for a particular type of problem. For the development of well-performing algorithms, Raidl (2006), Blum et al. (2011) and Sorensen (2012) suggest the following: 1) a careful literature search with the aim of identifying the most successful optimisation approaches for the problem at hand or for similar problems; 2) the study of different ways of combining the most promising features of the identified approaches.

Talbi (2009) and Blum et al. (2011) identified one successful way of using metaheuristic hybrids, combining metaheuristics with (complementary) metaheuristics. The authors encouraged the use of population-based approaches hybridised with single-solution approaches. Talbi (2009) refers to this hybrid as a Low-level Teamwork Hybrid (LTH). GAs make use of local search methods, called Memetic Algorithms (MAs), which

are usually successful in solving combinatorial optimisation problems. For variants of VRPs, MAs are also well-known for achieving a high performance level, for example that of Vidal et al. (2013) Nagata et al. (2010), Nagata and Kobayashi (2010). The Adaptive Large Neighbourhood Search (ALNS) using SA and Large Neighbourhood Search (LNS) are widely adopted due to their solution quality and speed for solving a wide range of VRPs. Therefore, this section is devoted to the design issues of this type of metaheuristic hybrid from different perspectives.

Concerning the philosophy of engineering design principles, Goldberg (2002) discussed the systematic approaches for invention or design for genetic algorithms and other innovating machines such as airplanes. Conceptual engineering are comprised of three tools: design decomposition, modelling middle and integration principles. In brief, decomposition refers to breaking large problems into smaller ones. It is a commonplace technique in design to attempt to build subsystems that correspond to the sub-function or so-called functional requirements. For example, the lift, control, and propulsion subsystems of an aircraft correspond to those sub-functions or functional requirements of an aircraft. The modelling middle investigates models that are little (less complex), applicable (small-size) and facet-wise (small number of facets). The integration principle is used to unify little models into dimensional scales such as time.

Goldberg (2002) outlined the GA design theory as follows: 1) Know what GAs process-building blocks (BBs) are; 2) Know the BB challenger-BB wise difficult problems; 3) Ensure an adequate supply of raw BBs; (4) Ensure an increased market share for superior BBs; 5) Know BB takeover and convergence times; 6) Make decisions well among competing BBs; 7) Mix BBs well. It should be noted that building blocks (BBs) refer to well-adapted sets of features that form subcomponents of effective solutions. The basic idea is that GAs (1) identify building blocks or subassemblies of good solutions and (2) recombine different subassemblies to form high performance solutions. The author stated that selecting and combining the good features of two or more approaches can promote intelligent jumping while the combination might be better than

either individuals. In addition, when selection and mutation are applied, they become a form of a hill-climbing mechanism, where a mutation creates variants in the neighbourhood of the current solution, called continual improvement. Continuing to experiment in a local neighbourhood is a powerful means of potential improvement. A GA will be called competent if it can solve hard problems, quickly, accurately, and reliably.

Similar to Goldberg (2002), Sorensen's (2012) outline of the design principles of metaheuristic hybrids is two-fold: 1) focus on the problem; 2) analyse your method. To focus on the problem, the following suggestions are made: 1) Do not develop a method without a problem; 2) Study the problem in detail; 3) Know the literature on the problem and on related problems; 4) Study the relationship between methods from the literature and the problem; 5) Use the best parts from existing methods for the problems (e.g. metaheuristics). To analyse the method, the following issues are suggested: 1) Deconstruct the methods; 2) Make sure each component matters; 3) Find the best parameter settings; 4) Use statistics; 5) Try to find out why and how the methods work; Use the best parts from different sub-areas of optimisation (e.g. metaheuristics).

Blum et al. (2011) emphasised that it is important that the contribution of different components to the algorithms' performance must be identified by considering theoretical models for describing properties of hybrid metaheuristics and using an experimental methodology. Blum and Roli (2003) introduced a framework, called the $I\&D$ (Intensification and Diversification) frame to put different intensification and diversification components into relation with each other. Although the metaheuristics are different, the mechanisms efficiently explore search spaces which are all based on intensification and diversification. Chapter 5 will show how a meta-heuristic hybrid is constructed.

Goldberg (2002) stated that the design of hybrid metaheuristics should combine strengths, and eliminate weaknesses among several methods. Mahfoud and Goldberg (1995) proposed Parallel Recombination Simulated Annealing by incorporating strengths and eliminating weaknesses between SA and GA. Nagata and Kobayashi

(2010), Nagata et al. (2010), Vidal et al. (2013), Mester and Bräysy (2005) hybridised the variants of GAs with local search or other metaheuristics for solving VRPs.

2.3.5 Summary

Vehicle routing problem and its variants are extensively investigated in the last few decades. However, there are still gaps between theory and practice in terms of problem characteristics while the solution methods are somewhat well-developed. The complexity in logistics business models and customer requirements motivates the research community to further bridge the gaps. While, the requirement of maintaining a high level of service quality, cost reduction, and timely optimisation for logistics planners in the rapidly changing business environments of the large size problems drives the needs for developing efficient and effective optimisation techniques.

The preceding section provided a survey of literature related to the problem targeted i.e. the optimisation problems of LSPs, 3PLs, LLPs for freight transport. Since, these problems have gained little attention but they are important to the economy of logistics sector. Several optimisation techniques have been successful in solving the related problems. However, it is interesting to see the strengths and weaknesses of these solution methods which challenge the researchers to explore the areas of improvement. One way to go from strength to strength is to hybridise the state-of-the-art optimisation techniques of related problems. The effort in this thesis is devoted to both tackle problem complexities and develop well-performing optimisation techniques.

Chapter 3

A Memetic Algorithm for the Multi-depot Pickup and Delivery Problem

3.1 Introduction

The Pickup and Delivery Problem is the core basis of optimisation problems arising in, for example, local courier operations and freight transportation. Parragh et al. (2008) explained that the single vehicle variant of the PDP, which a capacity constraint is not imposed, can be referred to as the Pickup-Delivery Travelling Salesman Problem. For multiple vehicle cases, which the capacity constraints are imposed, the variant is referred to as the Multi-vehicle Pickup and Delivery problem (MV-PDP). This Chapter investigates the Multi-depot Pickup and Delivery Problem (MDPDP), an extension of CVRP and TSP, in which multiple-depots, precedence constraints, heterogeneous fleets of vehicles, and maximum route length constraints are considered. The multiple-depots characteristic arises in practice when large LSPs or 3PLs seek to gain geographical coverage.

Parragh et al. (2008) presented the basic model for MV-PDP using Mixed Integer Linear Programming (MILP). Cordeau (2006) showed the linearisation of some constraints, demonstrated in Parragh et al. (2008), using a Big M formulation. Ropke and Pisinger (2006) presented the formulation of PDPTW which can be applied to multiple depot cases. However, the MDPDP is NP-hard as it is an extension of CVRP. Unfortunately, exact methods are not practical for solving large-size problems: computational time considerably increases as the problem size becomes larger. Therefore, heuristics are preferred due to the prohibitive computational time in rapidly changing business environments. However, the solutions of heuristics are sometimes trapped in local optima. Therefore, several researchers resort to apply meta-heuristics. Metaheuristics provide a general framework for embedding heuristics to escape from local optima. Several metaheuristics are widely used for variants of VRPs because of their robustness. To summarise, the investigation of MDPDP is motivated by both theoretical challenges and practical significance.

Blum and Roli (2003) described that one way to classify metaheuristic algorithms is population-based vs. single point search. It is Goldberg's (1989) belief that single solution approaches can locate false peaks in multi-modal (many peaks) search spaces for problems, such as, vehicle routing and scheduling problems. The population-based approaches such as GA can reduce the probability of locating false peaks. The GA climbs many peaks simultaneously and, as a result, provides robustness and parallelism. GAs are easy to implement and do not depend as much on the quality of the initial solutions as in the case of other heuristics and meta-heuristics (Hosny and Mumford 2007). Blum and Roli (2003) confirmed that population-based algorithms provide a natural, intrinsic way to explore the search space. Yet, the final performance strongly depends on the way the population is manipulated. Creput et al. (2004) and Jih et al. (2002) confirmed that GAs have been successfully applied to solve combinatorial optimisation problems such as PDPTW. Jayalakshmi et al. (2001), Talbi (2002), Blum and Roli (2003), Talbi (2009) and Blum et al. (2011) confirmed that the hybridization of

GA with other local search heuristics is powerful in the exploration of the search space and the exploitation of solutions found. Several variants of hybrid Genetic Algorithms, or Memetic Algorithms, prove to be successful in solving Combinatorial Optimisation Problems, Vehicle Routing Problems, Pickup and Delivery problems. Therefore, this Chapter investigates a Memetic Algorithm (MA) to solve the MDPDP. As there is no benchmark instance for MDPDP, randomly generated instances were used to investigate the computational performance of CPLEX and Memetic Algorithm. Due to being NP-hard, the CPLEX cannot solve a large-sized problem. However, the CPLEX's results can be used to validate the MA.

3.2 Literature Review

This section provides a survey of related work for solving the MDPDP. Up to present, several sub-problems of the MDPDP are extensively investigated and solved by exact, heuristics, metaheuristics, and hybrid-metaheuristics. These problems, together with their GA-related solution methods, include Vehicle Routing Problems, Multi-Depot Vehicle Routing Problems, Pickup and Delivery Problem with Time Windows.

3.2.1 Multi-Depot Vehicle Routing Problems (MDVRP)

The housing of vehicles can be classified as either a single depot or multiple depots. Lawrence (1983) claimed that the Multi-Depot Vehicle Routing Problem (MDVRP) is a generalisation of the VRP problem in that fleets of vehicles serve their customers from a number of depots rather than one. All other constraints placed on the single-depot VRP still apply. Moreover, some additional constraints and assumptions may appear in Multi-Depot Problems. Renaud et al. (2002) claimed that the MDVRP is an NP-hard problem and very difficult to solve to optimality even for relatively small-size instances. Many methodologies have been proposed for the single-depot VRP. Nevertheless, these methodologies cannot be properly extended to deal with the presence of several depots

because of storage and computational requirements.

Salhi and Sari (1997) claimed that little attention has been paid on MDVRP even though, in practice, it is likely that a distribution system operates from several depots. The authors proposed a multi-level composite heuristic for solving the multi-depot vehicle fleet mix problem. The route perturbation procedure (RPERT), modified for multiple-depots, is referred to as MULTI-RPERT. Two reduction tests were devised; one for single depot routing and the other for multi-depots routing problems. The computational experiments were carried out on the standard benchmark problems varying in size from 50 to 360 customers, using 2 to 9 depots, and 5 different vehicle capacities. The algorithm was capable of finding 7 new best known solutions out of 26 test problems. Also, it yielded solutions which are on average just over 1% above the best solutions and required only 10% of the computational time compared to the benchmark instances.

Skok et al. (2000) used the "steady-state" genetic algorithm to solve a Multiple Depot Capacitated Vehicle Routing Problem. In this study, the initial population was created randomly, and six crossover operators were compared. The experiment showed that Cycle Crossover and Fragment Reordering Crossover performed well. Three mutation operators were tested, and Order Based Mutation was a clear winner. Several test instances were used. The authors claimed that the proposed GA was effective in producing high quality solutions in a reasonable amount of time.

Thangiah and Salhi (2001) applied an Adaptive Genetic Clustering (GenClust) method to solve the MDVRP. The GenClust method is based on using a route primitive. The GA is used to adaptively search for the attributes of a set of circles that cluster customers using the routing cost as the fitness value for individual chromosomes. GenClust uses the local search method and customer interchange method to improve the solution. Moreover, the post-optimisation phase is applied. Two reduction tests are embedded to speed up computation. The GenClust was tested on benchmark problems varying in size from 50 to 360 customers and two to nine depots. The GenClust

obtained 10 new best known solutions and matched one best known solution.

Ombuki and Hanshar (2009) presented a Genetic Algorithm (GA) for solving the MDVRP with capacity and route-length constraints. The algorithm was tested on 23 classic MDVRP standard benchmark problems with 50 to 360 customers. The proposed GA is compared with the state-of-the-art GA, *GenClust*, which was developed by Thangiah and Salhi (2001). The author claimed that the proposed GA improves the solution quality and obtains 17 out of 23 new GA solutions compared to the best published GA. The computational results show that the proposed GA is equally good compared to other existing non-GA based meta-heuristics.

Lau et al. (2010) proposed a Fuzzy Logic guided Genetic Algorithm (FLGA) to solve the MDVRP. The role of FuzzyLogic is to dynamically adjust the crossover rate and mutation rate after ten consecutive generations. Partial Uniform and Partial Order (PUPO) crossover and Partial Uniform and Partial Swap (PUPS) were developed. A number of benchmark problems are utilised to investigate its search ability by comparing with various search techniques, Branch-and-Bound, standard GA, SA, TS. The results show that the FLGA method outperforms other search methods.

Several authors proposed effective single-solution metaheuristics for solving the MDVRP such as ALNS of Pisinger and Ropke (2007), Iterated Local Search of Subramanian (2012) and UHGS of Vidal et al. (2013). These authors are able to improve the best known solutions of standard benchmark instances of MDVRP.

3.2.2 Pickup and Delivery Problems (PDPs)

Lokin (1978) introduced the precedence constraints, which are required to formulate the PDP, into the traditional TSP. Savelsbergh and Sol (1995) first provided a unified notation of most PDP and a brief overview of existing solution methods. Li and Lim (2001) proposed a tabu-embedded simulated annealing algorithm to solve the PDPTW. The test instances of Li and Lim (2001) have been widely tested by several researchers. Parragh et al. (2008) carried out a survey of variants in PDPs. In this section, hybrid

Genetic Algorithms for solving PDPs are surveyed.

Jih et al. (2002) proposed a Family Competition Genetic Algorithm (FCGA) for solving the single vehicle PDPTW: "The FCGA is based on GA with the added concept of families. The concept is that, for every population, each individual owns its family. To maintain, the constant size of a population, only the champion at a family survived". A set of randomly generated instances was created. From their experiments, the FCGA outperform the traditional GA in most cases.

Pankratz (2005) proposed a Grouping Genetic Algorithm (GGA) for solving the multiple-vehicle PDPTW that features a group-oriented genetic encoding in which each gene represents a group of requests instead of a single request. The GGA, which adopts the concept of a grouping problem in Falkenauer (1998), applies a steady-state approach without duplicates. In order to detect duplicates, a simple comparison of objective values is used. The group-oriented crossover operator, the group-oriented mutation operators and embedded insertion heuristics are used. The GGA was tested with benchmark instances, provided by Nanry and Barnes (2000) and Li and Lim (2001), for the PDPTW. The experimental results of the GGA are competitive to previous metaheuristic methods for solving the PDPTW.

Rekiek et al. (2006) applied a Grouping Genetic Algorithm to the Handicapped Person Transportation (HPT) problem, which is a real-life application based on the concepts of the PDP. This GGA also modifies the GA's Falkenauer (1998) for the grouping problem. The crossover consists of four steps: select crossing sections; inject group(s); eliminate empty group(s) and group(s) with doubles; and reinsert missing objects. The mutation and inversion operators are also applied to each chromosome with a small probability. The local improvement procedures, namely Fareast heuristic and Go-and-Return heuristics are also used. The test problem, generated for the Brussels region, consists of a trip, with service requirements for 164 clients and 18 vehicles.

Hosny and Mumford (2007) presented a duplicate gene encoding that guarantees the satisfaction of the precedence constraints for solving single vehicle PDPTW. The author

discussed that the genetic encoding developed can avoid the precedence issue by simply assigning the same code to both the pickup and its designated delivery locations, relying on a simple decoder to identify its first occurrence as the pickup and the second as the delivery. The use of merge crossover (MX1) and directed mutation show competitive computational results from the data set of Jih et al. (2002).

3.2.3 Memetic Algorithms

Goldberg (1989) defined that a Simple Genetic Algorithm (GA) is composed of three operators: reproduction, crossover, and mutation. Moscato (1989) introduced the concept of Memetic Algorithm (MA) by illustrating the martial arts that are considered memes. Dawkins (1976) introduced the word meme to denote the idea of a unit of imitation in cultural transmission which in some aspects is analogous to the gene. Moscato (1989) discussed that while GAs are inspired in trying to emulate biological evolution, MAs would try to mimic cultural evolution. In the context of OR, the MA is a marriage between a population-based global search and the heuristic local search made by each of the individuals. Goldberg (1989) refers to MAs as Hybrid Genetic Algorithms. Moscato and Cotta (2003) discussed that MAs are intrinsically concerned with exploiting all available knowledge about the problem. The incorporation of problem domain knowledge is a fundamental feature that characterises MAs. The success of MAs can probably be explained as being a direct consequence of the synergy of the different search approaches they incorporate. These approaches include heuristics, approximation algorithms, and local search techniques. The MA exploits the global perspective of GA as the population-based approach, the rapid convergence by tournament selection and the problem specific knowledge for genetic operators. Krasnogor and Smith (2005) and Nguyen et al. (2007) discussed the design issues of MA being applied to well-known combinatorial optimisation problems. Krasnogor (2005) presented a review of Genetic Local Search, GLS_Based_Memetic_Algorithm and Genetic hybrid Algorithm.

To the best of our knowledge, there is no GAs or MAs applied to the MDPDP which

is an extension of MDVRP. Therefore, the framework of hybrid GA or MA, as used in Ombuki and Hanshar (2009) for MDVRP, is investigated in this Chapter, due to its simplicity and computational performance, and shown in Algorithm 3.1.

Algorithm 3.1 Outline of the genetic routing system

1. *Generate an* **initial population**, *POP*;
2. **Evaluate** *the fitness F(x) of each chromosome x of the population, and calculate the average fitness;*
3. *Create a new population by repeating the following steps unitl the new population is complete;*

- **Selection** *Select two parent chromosomes from the population, POP, by tournament selection;*
- **Recombination** *Apply crossover with a probability to the parents to form new offspring. If no crossover is performed, offspring is an exact copy parents;*
- **Mutation** *With a mutation probability, apply intra-depot mutation to mutate new offspring. If a certain number of generation is reached and mutation was applicable use, inter-depot mutation instead;*
- **Acceptance** *Place a new offspring in the population, replacing the parents;*
- **Elitism** *Randomly replace 1% of the population with the best 1% parents' population;*

4. *Update the old population with the newly generated population;*
5. *If the preset number of generation is reached, stop, return the average fitness, and the fitness of the best (chromosome) solution in the population;*
6. *Else go to step 2;*

Ombuki and Hanshar (2009) described how the evolutionary part is carried out with ordinary GAs, using crossover and selection operations on chromosomes. In Algorithm 3.1, the tournament selection with elite retention is used to perform a fitness-based selection of individuals. The GA applies an adaptive inter-depot mutation to re-assign some of the boundary customers to different depots from the initial static clustering which takes place before evolution. Intra-depot mutation involves bringing diversity within the routes of each depot. Three types of intra-depot mutations were used, namely, reversal, single customer rerouting, and customer swapping. The removed customers

are reinserted into the best feasible insertion location within the entire chromosome. The single customer rerouting can be considered a local search operator.

3.3 Problem Description and Formulation

In MDPDP, a heterogeneous fleet of vehicles located at several depots transports goods to satisfy customer requests from pickup to corresponding delivery points or paired requests. The pickup locations must precede their delivery destinations and both must be served by the same vehicle. The number of vehicles stationed at each depot is known. The pickup and delivery requests are also known in advance. The vehicles must start from and return to the same depot. Each vehicle has limited capacity. In addition, a route-length restriction for each vehicle is imposed. The objective function is to find minimum-distance routes served by those vehicles to satisfy customer requirements. All vehicles have to serve each request once, and all transportation requests must be met. Parragh et al. (2008) presented the multi-vehicle Pickup and Delivery Problem formulations, which can be extended to time window constraint and maximum user ride time constraint for multi-vehicle Dial-a-ride Problem (DARP). Ropke and Pisinger (2006) also presented a formulation for PDPTW, which can be extended to the multi-depot characteristic.

Exact methods cannot solve the MDPDP to optimality within a reasonable time frame because MDPDP is NP-hard. Therefore, the MILP formulation is only solved by default exact algorithms using CPLEX 11.0 for validation purpose. The PDPTW formulation provided by Ropke and Pisinger (2006) is adapted for the MDPDP. According to Cordeau (2006), in order to solve as large a problem as possible, the formulation must exclude infeasible networks, due to precedence relationships, and use Big-M to formulate a linear form. Moreover, several controlling parameters in CPLEX, such as MIP search and Variable Selection, are tuned to obtain optimal solutions in a reasonable amount of time. First, the notation used throughout the formulation is given.

Then, the formulation and an illustrative example are shown.

The MDPDP is comprised of n requests, and m vehicles. The problem is defined on a graph, $P = \{1, \ldots, n\}$ is the set of pickup nodes, $D = \{n+1, \ldots, 2n\}$ is the set of delivery nodes. Request i is represented by nodes i and $i+n$. K is the set of all vehicles, where $\mid K \mid = m$. Some vehicles can only service some requests. Let $N = P \cup D$. Denote $\tau_k = 2n + k$, $k \in K$, and $\tau_k^{'} = 2n + m + k$, $k \in K$ be the nodes that represent the start and end depots of vehicle k, respectively. The graph $G = (V, A)$ contains the nodes $V = N \cup \{\tau_1, \ldots, \tau_k\} \cup \{\tau_1^{'}, \ldots, \tau_m^{'}\}$ and the arcs $A = V \times V$. For each arc $(i, j) \in A$, the distance $d_{ij} > 0$ and a travel time $t_{ij} > 0$. The triangle inequality for time is satisfied: $t_{ij} < t_{il} + t_{lj}$ for all $i, j, l \in V$. For each node $i \in N$, l_i is the amount of goods that must be loaded onto the vehicle at the particular node $l_i > 0$ for $i \in P$, and $l_i = -l_{i-n}$ for $i \in D$. The capacity of vehicle $k \in K$ is denoted C_k. R^k is the maximum distance allowance for vehicle k.

Four types of decision variables are used in the mathematical model. x_{ijk}, $i, j \in V, k \in K$ is a binary variable that is one if the arc between node i and j is used by vehicle k and zero otherwise. $S_{ik}, i \in V, k \in K$ is a non-negative integer that indicates when vehicle k starts the service at location i. L_{ik},$i \in V, k \in K$ is a nonnegative integer corresponding to the total load of vehicle k at vertex i. z_i, $i \in P$, is a binary variable that indicates whether or not request i is placed in the request bank. The variable is one if the request is placed in the request bank and zero otherwise. For practical reasons, the arc set, A_k is reduced to the feasible network: $A^{'} = \{(i, j) : i, j \in V, i \neq \tau_k^{'}, j \neq \tau_k, i \neq j, i \in P \Rightarrow j \neq \tau_k^{'}, i = \tau_k \Rightarrow j \notin D, i \in D \Rightarrow j \notin P \ \ where\, i = j + n\}$. Parragh et al. (2008) pointed out that non-linear constraints can be linearised using a big M formulation. Therefore, we formulated the model in linear form by applying Big-M formulation to speed up the search.

A mathematical model of the problem is

$$Min \quad \alpha \sum_{k \in K} \sum_{(i,j) \in A'} d_{ij} x_{ijk} + \gamma \sum_{i \in P} z_i \tag{3.1}$$

Subject to:

$$\sum_{k \in K_i} \sum_{j:(i,j) \in A'} x_{ijk} + z_i = 1 \qquad \forall i \in P \tag{3.2}$$

$$\sum_{j:(i,j) \in A'} x_{ijk} - \sum_{j:(n+i,j) \in A'} x_{n+i,j,k} = 0 \qquad \forall k \in K, i \in P \tag{3.3}$$

$$\sum_{j \in P \cup \{\tau_k'\}} x_{\tau_k,j,k} = 1 \qquad \forall k \in K \tag{3.4}$$

$$\sum_{i \in D \cup \{\tau_k\}} x_{i,\tau',k} = 1 \qquad \forall k \in K \tag{3.5}$$

$$\sum_{i:(i,j) \in A'} x_{ijk} - \sum_{i:(i,j) \in A'} x_{jik} = 0 \qquad \forall k \in K, \forall j \in N \tag{3.6}$$

$$S_{ik} + s_i + t_{ij} - M(1 - x_{ijk}) \leq S_{jk} \qquad \forall k \in K, \forall (i,j) \in A' \tag{3.7}$$

$$S_{ik} \leq S_{n+i,k} \qquad \forall k \in K, \forall i \in P \tag{3.8}$$

$$L_{ik} + l_j - M(1 - x_{ijk}) \leq L_{jk} \qquad \forall k \in K, \forall (i,j) \in A' \tag{3.9}$$

$$Max\{0, l_j\} \leq L_{ik} \leq Min\{C_k, C_k + l_j\} \qquad \forall k \in K, \forall i : (i,j) \in A' \tag{3.10}$$

$$\sum_{(i,j) \in A'} d_{ij} x_{ijk} \leq R^k \qquad \forall k \in K \tag{3.11}$$

$$x_{ijk} \in \{0,1\} \qquad \forall k \in K, \forall (i,j) \in A'$$

$$z_i \in \{0,1\} \qquad \forall i \in P$$

$$S_{ik} \geq 0 \qquad \forall k \in K, \forall i \in V$$

$$L_{ik} \geq 0 \qquad \forall k \in K, \forall i \in V$$

The objective function is to minimise the weighted sum of the travelled distance,

and the number of requests not scheduled. Constraint (3.2) ensures that each pickup location is visited or placed in the request bank. Constraint (3.3) ensures that both pickup and corresponding delivery requests are visited by the same vehicle. Constraints (3.4) and (3.5) ensure that every vehicle departs from a start terminal and returns to a designated end terminal. Together with constraint (3.6), this ensures that consecutive paths between τ_k and $\tau_k^{'}$ are established for each $k \in K$. Constraints (3.7) ensure that S_{ik} is set correctly along the paths. These constraints also prevent sub-tours. Constraint (3.8) ensures that each pickup precedes its delivery location. Constraints (3.9) and (3.10) ensure that the load variables satisfy the vehicle capacity. Constraint (3.11) ensures that every vehicle do not travel exceed the pre-defined distance.

3.4 An Illustrative Example

This section presents an illustrative example consisting of 2 depots and 10 paired pickup and delivery locations (20 locations). The network topology is produced by yEd graph editor. Figure 3.1 shows the test instance no. 1, as described in Table 3.2.

In Figure 3.1, the squares represent depots. The triangles and circles labelled by the same numbers denote the pickup and corresponding delivery locations respectively. The blue dashed arrows represent the precedence relationship between the pickup and associated delivery locations. To illustrate the transportation demand, in Figure 3.1, these paired pickup and delivery locations are not yet scheduled. The geographical distribution is uniform. Using the notation defined for the MDPDP mathematical formulation earlier, the following input data is given for this illustrative example:

- $n = 10$
- $P = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\},\ D = \{11, 12, 13, 14, 15, 16, 17, 18, 19, 20\}$
- $(l_i, l_{i+n}) = \{(l_1, l_{11}), (l_2, l_{12}), (l_3, l_{13}), (l_4, l_{14}), (l_5, l_{15}), (l_6, l_{16}), (l_7, l_{17}), (l_8, l_{18}), (l_9, l_{19}), (l_{10}, l_{20})$
  $= \{(15, -15), (12, -12), (12, -12), (12, -12), (7, -7), (16, -16), (7, -7), (19, -19), (7, -7), (9, -9)\}$

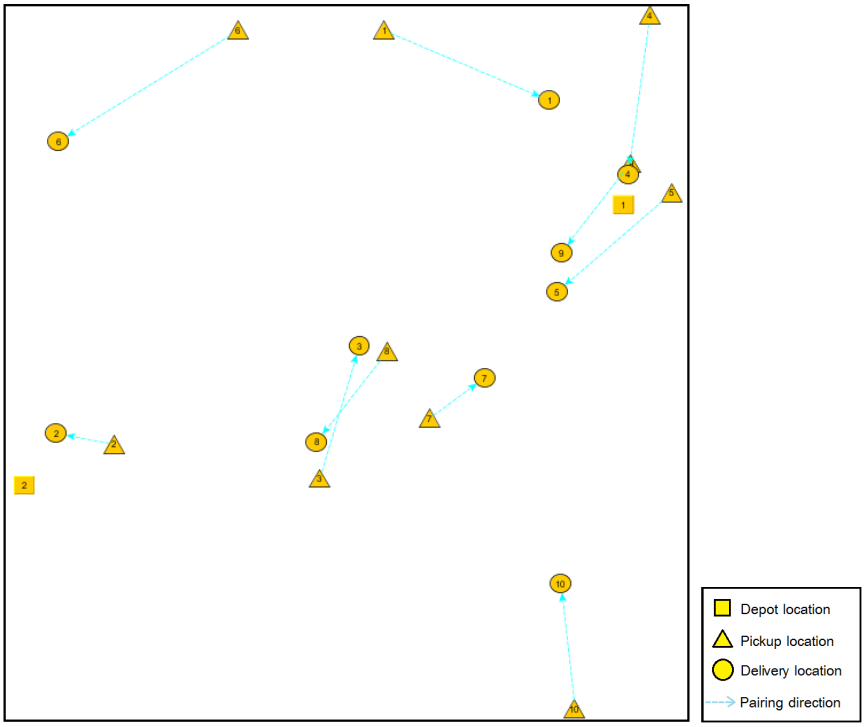


Figure 3.1: Illustrative Example: Network typology of 2 depots and 10 requests (20 locations)

- $C^k = 20 \quad \forall k \in K$
- $R^k = 1450 \quad \forall k \in K$

The program for implementing the mathematical formulation is written in Microsoft Visual Studio 2008: C# and run on Intel Core 2, Processor 2.49 GHz, 3.48 GB of RAM. This example of MDPDP is optimally solved by ILOG CPLEX 11.0. The objective function is computed from the total distance travelled by the vehicles. The optimal solution of this illustrative example is demonstrated in Figure 3.2.

In Figure 3.2, the optimal routes are demonstrated by coloured arrows. Notice that the different colours from the same depots represent different vehicles. The dashed lines of the transportation requests are superimposed by the vehicles' routes when the vehicles pick up a certain amount of goods and, immediately transport them to the designated delivery location. Otherwise, the vehicles can detour to pick up or deliver at other locations before completing the transportation requests e.g. paired requests 3

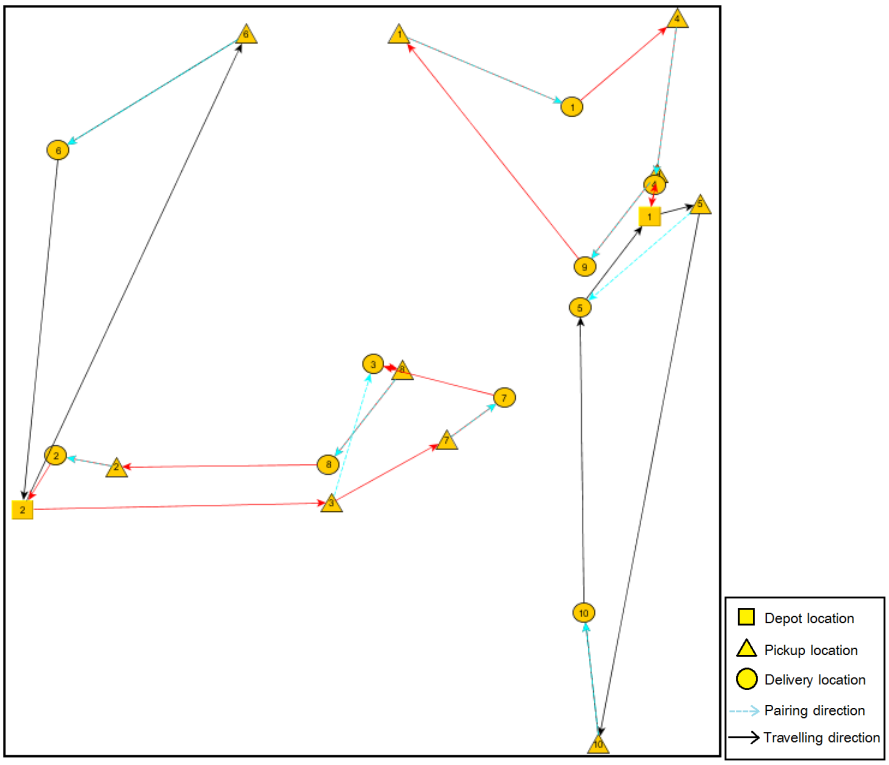


Figure 3.2: Illustrative Example: Optimal solution obtained from CPLEX

and 5. This usually generates savings toward the total travelling distance.

3.5 Test Problem Generation

To computationally evaluate the performance of CPLEX and a Memetic Algorithm, different test instances with varied characteristics are solved. To the best of our knowledge, there is no standard test instance for the MDPDP but there is a set of test instances for its variants, for example that from Ropke and Pisinger (2006). Therefore, 32 small test instances are generated by adapting the test instances of Ropke and Pisinger (2006). Experiments were conducted on the problem with up to 10 requests (20 locations), 4 vehicles and 4 depots. Several authors claimed that problems consisting of borderline customers between depots are more difficult to be solved. Therefore,

the last 8 instances are designed for testing the difficulty of solving borderline requests.

The computational complexity of the MDPDP is evaluated in terms of computational time corresponding to each problem parameter and the number of decision variables generated. Several performance features in CPLEX are experimented to obtain the appropriate setting, which reduces the computational time. Then, the MA is validated, and its performance is compared with optimal solutions obtained by CPLEX for these test instances. The detailed characteristics of test instances are shown in Table (3.2) and Table 3.3.

3.5.1 Study of problem parameters

Based on the experiments carried out, the computational results confirmed that the larger the number of paired requests and vehicles, the larger the number of decision variables and constraints, and, as a result, the longer the computational time. The S_{ik} and L_{ik}, as continuous variables, have less influence compared to x_{ijk}.The maximum route length and vehicle capacity parameters only constrain a vehicle from servicing customers according to its restrictions. The computational time varies for these two parameters, since, for Mixed-Integer Linear Programming (MILP), the computational time is mostly influenced by the number of discrete decision variables: integer and/or binary variables. The larger the problem size to be solved, the more the dynamic search or Branch-and-Cut algorithm in CPLEX requires branching decision variables.

Based on the solutions obtained for the test instances, we observe that the use of reduced feasible network A' and Big M linearisation can reduce the CPLEX's computational time from 10 % to 20%.

3.5.2 Tuning Performance Features in CPLEX

There are a number of parameters in performance features that are used for controlling search strategies. Applying suitable parameters for a specific problem can lead

to a better performance in terms of computational time. The controlling parameters tested are, namely, MIP search, Variable Selection, Node Selection, Backtracking tolerance, Branching direction, MIP Emphasis, Probe, Repeat presolve, CutPass, HeurFreq, Flowcover, Mircut and Rinsheur. According to the reference manual[1], these controlling parameters are described as having the following characteristics:

- **MIP Search** sets the search strategy for a mixed integer program whether applying a dynamic search or conventional Branch-and-Cut based on characteristics of the model.
- **Variable Selection** sets the rules for selecting the branching variable at the node which has been selected for branching, namely, minimum feasibility, maximum infeasibility, pseudo cost and strong branching.
- **Node Selection** sets the rule for selecting the next node to process when backtracking and includes Depth-first search, Best-bound search, Best-estimate search and alternative best-estimate search.
- **Backtracking Tolerance** controls how often backtracking is done during the branching process: the objective function value of the best integer feasible solution, the best remaining objective function value of any unexplored node and the objective function value of the most recently solved node are all used to control backtracking.
- **Branching Direction** decides which branch, the up or the down branch, should be taken first at each node, uses down branch selected first, let CPLEX choose and up branch selected first to determine direction.
- **MIP Emphasis** controls the trade-off between speed, feasibility, optimality, and moving bounds in MIP through the application of balance optimality and feasi-

[1] http://www-eio.upc.es/lceio/manuals/cplex-11/pdf/refparameterscplex.pdf

bility, the emphasis of feasibility over optimality, the emphasis of optimality over feasibility, the emphasis of moving best bound and the emphasis of finding hidden feasible solutions.

- **Probe** sets the amount of probing on variables to be performed before MIP branching, namely, no probing, let CPLEX choose, moderate probing level, aggressive probing level, very aggressive probing level.
- **Repeat Presolve** decides whether to re-apply presolve, with or without cuts, to a MIP model after processing at the root is otherwise complete, by letting CPLEX choose, turning off represolve, using represolve without cuts, using represolve with cuts or using represolve with cuts and allowing new root cuts.
- **CutPass** sets the upper limit on the number of cutting plane passes CPLEX performs when solving the root node of a MIP model, by choosing either none, by letting CPLEX choose or the number of passes to perform.
- **HeurFreq** decides how often to apply the periodic heuristic. The choices are: none, let CPLEX choose or apply the periodic heuristic at this frequency.
- **Flowcover** decides whether or not to generate flow cover cuts for the problem. Flowcover choices are: do not generate flow cover cuts, let CPLEX choose, generate flow cover cuts moderately or generate flow cover cuts aggressively.
- **Mircut** decides whether or not to generate MIR cuts. The choices are: do not generate MIR cuts, let CPLEX choose, generate MIR cuts moderately or generate MIR cuts aggressively.
- **Rinsheur** decides how often to apply the relaxation induced neighbourhood search heuristic. The choices are none, let CPLEX choose or frequency to apply RINS heuristic.

In addition, the automatic tuning tool is also experimented to find an initial setting. To compare the performance of each performance feature, 13 representative instances were used. First, automatic tuning obtained the initial set of parameters. Then, beginning with the MIP, each setting was experimented on, and the setting producing the best time was selected and fixed. This continues untill all of the parameters are tested. The average computational time was reduced by 58.15% for all 13 test instances. The results showed that the pseudo reduced costs used in the **Variable Selection**, using represolve with cuts and allowing new root cuts in **Repeat Presolve**, and the default parameters for the rest are the appropriate setting for solving these test instances.

3.5.3 Investigation of Borderline Customers

Gillett and Johnson (1974) and Golden et al. (1977) considered how MDVRP, which contains borderline customers, becomes more difficult to be solved by heuristics. The authors defined the borderline customer as those located approximately halfway between two depots. Gillett and Johnson (1974), Golden et al. (1977), Salhi and Sari (1997), Salhi and Nagy (1999) and Nagy and Salhi (2005) proposed heuristics to tackle borderline customers by using a depot clustering algorithm, one of the most widely used approaches for Multi-Depot Vehicles Routing and Scheduling Problems (MDVRSP).

Intuitively, when applying the depot clustering fashion to multi-depot problems, it is rather difficult to predictably assign borderline customers to a particular depot. However, the problem considered in this study is a variant of Pickup and Delivery Problem. Therefore, this difficulty for the MDPDP should be re-investigated for the MA.

Unlike MDVRSP, the MDPDP requires a completed pickup and delivery for each customer, served by the same vehicle. Therefore, borderline customers of the MDPDP are defined as "the pickup and corresponding delivery requests that are approximately

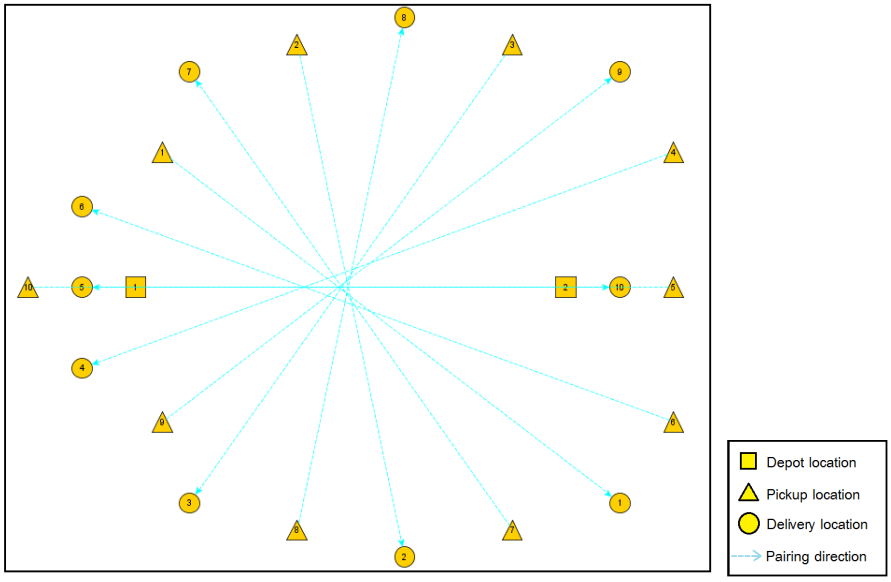


Figure 3.3: Network typology for testing the depot clustering heuristic with borderline paired requests

equidistant from several depots". Hereafter, borderline customers of MDPDP are referred to as the borderline paired requests. According to Salhi and Sari (1997), the ratio of measuring the borderline status of a customer (ε) is determined as "the distance of the customer locations to nearest depot divided by that of second nearest depot". If ε is greater than 0.7, the customers are considered borderline customers. Similarly, the borderline of a paired request (δ) is defined as "the sum between the distance of the pickup and delivery locations to nearest depot divided by that of second nearest depot" with δ= 0.7. From observations, the average value of δ for all customers in each distribution: uniform, clustered, semi-clustered from 24 instances, as shown in Table (3.2) and (3.3), are 0.49, 0.55 and 0.57, respectively. These values is below δ= 0.7. Therefore, 8 new test instances, for which the average value of δ for all customers is equal to 0.98 are designed. Their network typology of requests and its optimal solution

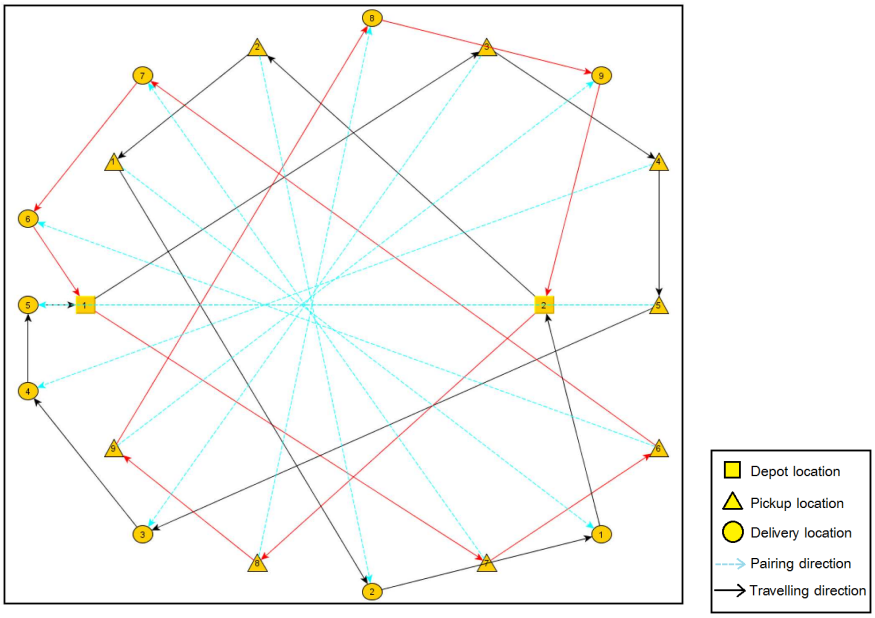


Figure 3.4: The optimal solution of test instance 25

of test instance 25 is displayed in Figure 3.3 and Figure 3.4, respectively.

3.6 Design of a MA for the MDPDP

As expected, based on the reported experimental results, CPLEX cannot solve large problems within a reasonable amount of time. Hence, a heuristic approach to efficiently solving practical-sized MDPDPs must be developed. Among meta-heuristics, Memetic Algorithms are well-known for their capability to perform reasonably well on highly constrained problems. Moreover, Ombuki and Hanshar (2009) applied MA to MDVRP and emphasised that the Hybrid GA or MA is equally good, compared to other existing non-GA based meta-heuristics, and it yields competitive results. Therefore, the use of MAs seems to be a justified option for MDPDP, which is highly constrained and classified as NP-hard.

In this Chapter, we applied the framework of MA proposed by Ombuki and Hanshar (2009), with modifications. The proposed Memetic Algorithm consists of three phases: evaluation of the fitness of each chromosome, selection of the parent chromosomes and

applications of the genetic operators to the parent chromosomes. Recombination or crossover operators replace some of the genes in one parent, with some other genes of the other parent, consequently, introducing changes to produce an offspring. The better solution is accepted. Otherwise, the offspring is copied exactly. If identical genes, evaluated by fitness function, are recombined, a random offspring is generated, as in the procedure of Random Offspring Generation (ROG) of Rocha and Neves (1999). The random offspring generation is embedded in the recombination operator to prevent premature convergence from the ineffectiveness in cross-fertilising the identical individuals. Mutation operators are applied to a single chromosome, where some of the individuals are selected with probability. The local search operators such as depot-clustering and worst removal heuristics are secondary operators that aid the MA for further exploitation of the solution space and, as a result, provide improvement of the solution quality. The evolution process is repeated until the termination condition is met.

3.6.1 Framework

The flowchart of our proposed MA is shown in Figure (3.5).

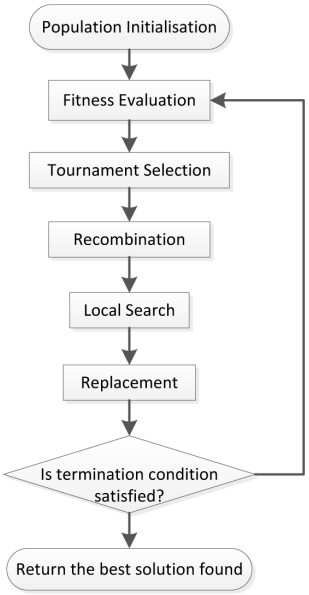


Figure 3.5: Flowchart of Memetic Algorithm (MA) for MDPDP

3.6.2 Population Structure and Initialisation

Initial Depot Clustering

A depot-clustering algorithm is slightly modified from that of Ombuki and Hanshar's (2009) so that it may be applied in population initialisation and further used within inter-depot operators. Initially, each paired request, i and $i + n$, is assigned to the nearest depot in terms of the Euclidean distance of each pickup and associated delivery location to the depot. Some paired requests are identified in a similar way to that of assigning borderline paired requests in Section 3.5.3 with $\delta = 0.7$. By using the inter-depot operator, borderline paired requests can be reassigned to other potential depots.

Population structure

In order to apply a GA or MA to a particular problem, it is required to select a chromosome representation that is suitable for and which will be efficient in the implementation of MDPDP. Moscato (1989) mentioned that a genetic, or a zero-one representation, would be useful under certain circumstances. However, for some problems, they are not the best representations, and one must use those that naturally belong to the problem. Pankratz (2005) encoded the PDPTW into cluster level (phenotype) and routing level (genotype). The author discussed that, for the PDPTW, it is not obvious how cluster and routing sub-problems can be simultaneously represented by a homogeneous encoding such as the standard binary representation of the classic Genetic Algorithm. The author also emphasised that permutation encoding such as in TSP can be problematic when being applied to highly constrained multi-vehicle routing problem like PDPTW. Moreover, most order-based crossover operators are context-insensitive. In other words, they do not take into account contextual information, such as precedence relationship, during recombination. As a result, even small modifications by genetic operators such

as a crossover operator will cause the offspring to have almost no phenotypical similarities to its parents. Therefore, it is hard for the GA to sample meaningful building blocks. Pankratz (2005), Rekiek et al. (2006) and Matthew and Gary (2005) apply a similar structure of chromosome representation for solving vehicle routing and scheduling problems. Ombuki and Hanshar (2009) applied the chromosome representation for MDVRP which specifies the number of routes (i.e., vehicles) and also the delivery order of each of these routes. The authors adopted a chromosome representation of the MDVRP that consists of several integer vectors, say n, where n corresponds to the number of depots. Each vector consists of a cluster of routes; each route is composed of an ordered subset of customers (genes). The structure includes the permutation of sequences in routes. Hosny and Mumford (2007) employed a duplicate gene encoding to guarantee the satisfaction of the precedence constraint in PDPTW. The same codes in terms of paired numbers are assigned to both pickup and its associated delivery locations. The pickup or delivery nodes of the same code are identified by the supply (+) and demand (-). This kind of encoding is simple and will eliminate the problem of backtracking to repair an infeasible solution violating the precedence relationship. In this way, a local search operator is capable of applying to the chromosome without decoding back to the solution structure. Based on their success, a chromosome representation for a MDPDP, considering PDPTW and MDVRP as sub-problems, adapts these approaches.

A chromosome representation for MDPDP consists of several lists, say n, where n corresponds to the number of depots, as shown in Figure 3.6. Each depot comprises a number of routes, and each route is comprised of a subset of paired requests. An example of a randomly generated chromosome which represents two depots and eight paired pickup-delivery locations served by four vehicles is illustrated in Figure 3.6. Pickups are shown in boldface while deliveries are presented in italics. The vehicle id is listed and corresponds to the depots where it starts and ends its journey.

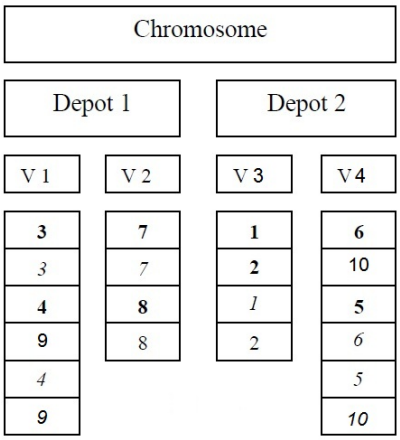


Figure 3.6: Example of chromosome representation for MDPDP

3.6.3 Fitness Function

The fitness value of each chromosome is determined by calculating the total distance travelled by vehicles and the weighted penalty for the number of unscheduled request(s) and maximum-route length violation. The fitness function of a chromosome is:

$$F = \sum_{k \in K} (D(k) + w_1 \times MV(k)) + w_2 \times UR$$

Where $D(k)$ is the total distance of route k, $MV(k)$ is the length exceeding the route length allowance of route k, and UR is the number of unscheduled requests. For each route length violation, the weight $w_1 = 1000$ is multiplied. For each unscheduled request, the weight $w_2 = 10000$ is multipled.It is worth noting that the penalty function is incorporated to consider maximum-route length restrictions as a soft constraint so that the MA can further explore the search space in tightly constrained maximum-route length. The evaluation of fitness function is carried out only when particular routes change in order to reduce redundancy and computational time.

3.6.4 Tournament Selection

Pankratz (2005) also used the binary tournament selection for PDPTW because of its low time complexity, compared to the classical "roulette wheel selection scheme". Ombuki and Hanshar (2009) applied binary tournament selection in MA for the MD-VPR. Moreover, the authors implement the techniques of using tournament selection probability which provides adjustable sensitivity to the tournament selection. As a result of its flexibility, we therefore apply these accordingly.

In every generation, parents must be selected for mating and reproduction. Two individuals called "a tournament set" are randomly selected from the population. A random number, r, between 0 and 1 is generated. If r is less than a certain parameter, say Φ (tournament selection probability), the fittest chromosome in the tournament set is then chosen as the one to be used for reproduction. Otherwise, any individual is selected for reproduction from the tournament set randomly. This procedure is repeated to choose another chromosome for mating and reproduction.

3.6.5 Recombination Operator

The recombination operator is one of the most important components in improving the solution quality. The crossover of the proposed MA adapts the Best Cost Route Crossover (BCRC) developed by Ombuki and Hanshar (2009). The BCRC is the problem specific operator for MDVRP that ensures the feasibility of solutions generated through genetic evolution.

In order to clarify the BCRC for the MDPDP, it is hereafter referred to as the Pickup and Delivery Route Crossover (PDRC). An illustrative example of PDRC is shown in Figure 3.7.

In Figure 3.7, parent 1, ($P1$), and parent 2, ($P2$), contain 2 depots and 8 requests served by 4 vehicles. Assume that one paired request is selected in this example. This crossover deals with one mutual depot in both parents, i.e. say depot D_1 is randomly

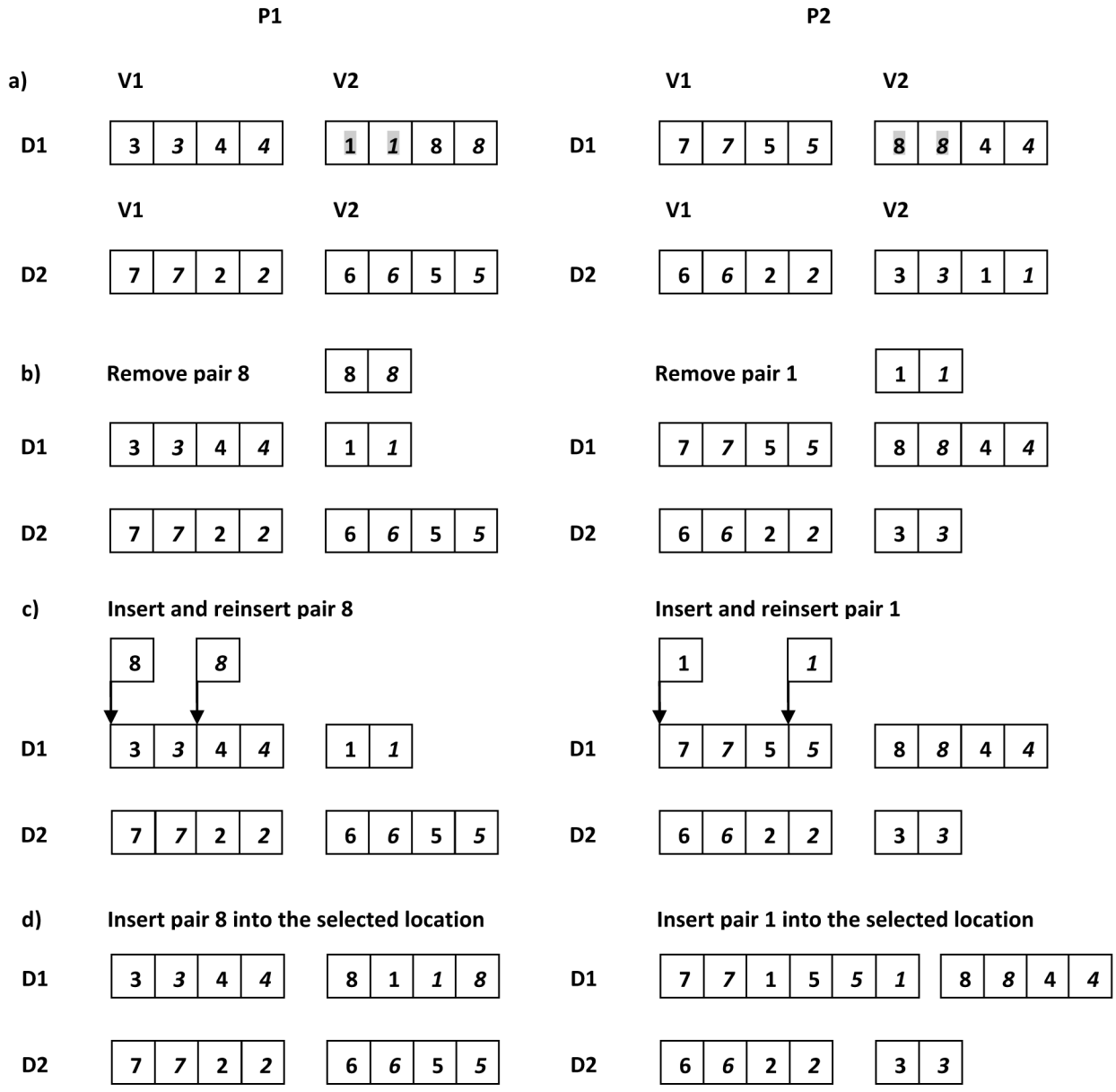


Figure 3.7: Illustrative example of PDRC

selected.

- In step a), D_1 of $P_1$and P_2 both have two vehicles and each vehicle in each parent is randomly selected. For example, vehicle 2 with the pair 1 in D_1 of P_1 is randomly chosen. Similarly, vehicle 2 with the pair 8 in D_1 of P_2 is randomly chosen.
- In step b), the chosen pairs from the opposite parents are removed. For example, pair 8 is removed from P_1(random selection from P_2 in step a). Likewise, pair 1 is removed from P_2.
- In step c), the pairs removed from step b) find all feasible locations and are inserted at the best insertion cost location.
- In step d), fitness function is calculated and the offspring are accepted only when the fitness is improved.

It is noted that, in step c), we develop the insertion technique for the Pickup and Delivery problem for this MA. The insertion prevents violating precedence relationships between paired pickup and delivery locations, which is called the Fixed-Forward method as described in Section 3.6.8.

During the experiment, premature convergence to local optima occurred and resulted in making crossing the same chromosomes ineffectual. Once identical chromosomes are mated and detected by fitness comparison, a new random offspring is generated and recombined. The Random Offspring Generation increases the effectiveness of the recombination operator while promoting population diversity.

3.6.6 Local Search

Local search operators are designed to intensify the search in a particular search

Algorithm 3.2 Modified Worst Removal

Function Modified Worst Removal (chromosome j, route k, depot d)
set of pickup requests: $P = \{1, \ldots, n\}$,$i \in P$;
for(int $i = 1$; $i \leq n$; i++);
Calculate $c(i, k, d)$;
Sort $c(i, k, d)$ in descending order;
return first entry of the cost (i, k, d) list

space. The GA is hybridised with local search operators using domain-specific knowledge, such as greedy removal, Shaw's removal and depot clustering heuristics.

Similar to the recombination operator, the local search procedures are comprised of removal, insertion, fitness evaluation and acceptance. In the proposed MA, several heuristics that remove paired request(s) at one route and insertion is then performed. The operators are devised to search in several boundaries for the MDPDP, namely, at intra- and inter-depot levels. At the intra-depot level or within a depot, two neighbourhood operators, rerouting and trip exchange, are applied by using problem-specific knowledge in the paired request removal. In the inter-depot level or between depots, alternative depot search is introduced in order for paired requests to find the feasible insertion locations in other potential depots. The removal heuristic used in the rerouting and alternative depot search is the "worst removal heuristic", originally presented by Ropke and Pisinger (2006). The concept is that it seems reasonable to remove requests with high costs and insert them at another place in the solution to obtain a better solution value. The cost without request i in route k, depot d is defined as $c(i, k, d)$. The slightly modified worst removal for MDPDP is shown in pseudo code in Algorithm 3.2.

In the trip exchange intra-depot operator, Shaw's removal heuristic, as used in Shaw (1997) and Ropke and Pisinger (2006), is applied with slight modifications by specifying the selected routes. The concept is to remove requests that are similar in terms of distance between paired requests, since it seems reasonably easy to swap or

Algorithm 3.3 Modified Shaw's removal heurisitc

Function Modified Shaw's Removal (route k_1, route k_2, depot d)
set of pickup requests in k_1: $P_1 = \{1, \ldots, n\}$, $i \in P_1$;
ramdomly select a request from P_1;
set of pickup requests in k_2: $P_2 = \{1, \ldots, m\}$;
for (int $j = 1$; j $\leq m$; $j++$)
  Calculate $R(i, j)$;
Sort $R(i, j)$ in ascending order;
return first entry of the cost $R(i, j)$ list

shuffle similar requests between two routes in order to seek potentially better solutions. The relatedness measure is given by

$$R(i, j) = d_{A(i),A(j)} + d_{B(i),B(j)}$$

Where $A(i)$, $B(i)$ represent the pickup and delivery locations of request i, respectively. $d_{A(i),A(j)}$, $d_{B(i),B(j)}$ denote the distance from $A(i)$ to $A(j)$, and $B(i)$ to $B(j)$, respectively. The pseudo code for removing requests is shown in Algorithm (3.3).

Rerouting intra-depot operator

This operator randomly selects the route that contains at least one request. Then, the worst removal as shown in 3.2, is applied. Sequentially, we use the fix-forward insertion method to find neighbourhoods of the solutions. Only the better solution is accepted. Otherwise, there is no change made to the chromosome. The rerouting operations inside routes and among routes are carried out by this rerouting intra-depot operator.

Trip exchange intra-depot operator

The operator randomly selects the depot and two routes. The modified Shaw's removal heuristic in Algorithm (3.3) is implemented and the selected requests in both routes are removed. The fix-forward insertion is used for one route at a time with a

view to improving solution quality. However, in the first route, the first solution and other solutions are always accepted if their fitness values are improved, since there is no reference fitness value to compare to justify whether or not the first route's fitness has been improved. This intuition is also applied to the second route. After the trip exchange intra-depot operator is completed, the non-improving solution may be accepted. This can be seen as the mutation operator. The experiments, however, showed that the operator often produced a higher quality chromosome.

Alternative depot operator (inter-depot operator)

The concept of Depot Clustering Algorithm in Ombuki and Hanshar (2009) is adapted to the inter-depot operator used in this study. This operator removes a paired request(s) on a randomly selected route using a modified worst removal heuristic and allows the swapping of the paired request(s) from one depot to another to help explore the search space while seeking improvements. The candidate depots are considered according to Ombuki and Hanshar (2009) with slight modifications due to investigating the MDPDP. The potential depots can be derived from the following inequality:

$$\frac{distance(p, d_i) - min}{min} \leq BOUND$$

where $distance\,(p, d_i)$ is the Euclidean distance from the pickup and delivery locations of p request to depot d_i, min is the distance from p to the nearest depot, and $BOUND$ is a constant value. In this study, $BOUND = 2$ according to Ombuki and Hanshar (2009). The fix-forward insertion is then applied through all routes to find the lowest insertion cost location. The alternative depot operator, as a unary operator, complements the recombination operator by considering moving paired requests to other potential depots in a sensible way.

3.6.7 Replacement Strategy

In this study, the MA applies generational replacement strategy according to Ombuki and Hanshar (2009). The generational replacement strategy replaces the parents so that the size of the population remains constant. It is possible that applying a genetic operator produces non-improving solutions. The non-improving offspring is also accepted in order to keep exploring the search space, while the best solution found is kept during the search.

3.6.8 Fix and Forward Insertion Method and Reduction Rule

To insert a paired request into another route, a systematic insertion method of Hang et al. (2003) called GENERATE is refined. Owing to the nature of the pickup and delivery problem, load is reduced after unloading, and a reduction rule for preventing capacity violations is introduced so as to avoid checking capacity violations at every insertion location. This method is hereafter refers to as the "fix-forward" insertion method.

Fix-forward insertion method

For n existing customer locations served on a route, there are $n+1$ insertion positions available. For example, in Figure 3.8, a route in a depot consists of 3 paired requests or 6 customer locations, which are visited in a particular order. There are 7 insertion positions available in which the pickup node of the paired request $(4+, 4-)$ can be inserted.

The principle is that once the pickup node, $4+$, is fixed at one location, the delivery node, $4-$, tries to insert and reinsert forward to the end of the route. Then, $4+$ is moved forward and fixed at the next position. These steps are repeated through to the end of the route for both $4+$ and $4-$. The procedure is illustrated in Figure 3.9, 3.10, 3.11, and 3.12.

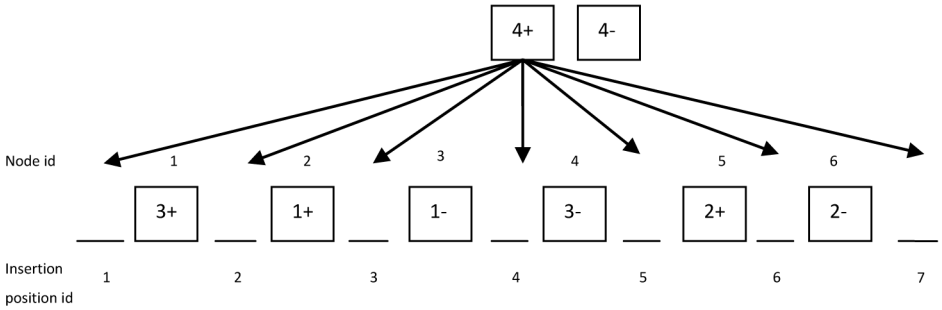


Figure 3.8: Illustrative example of available insertion position

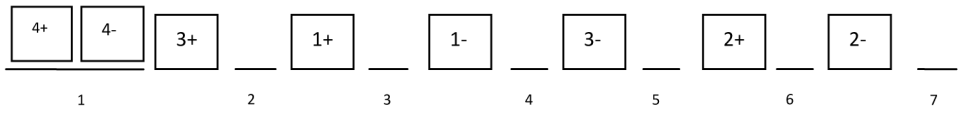


Figure 3.9: Illustrative example of fix-forward insertion: 1^{st} fix (pickup) and 1^{st} forward (delivery) position

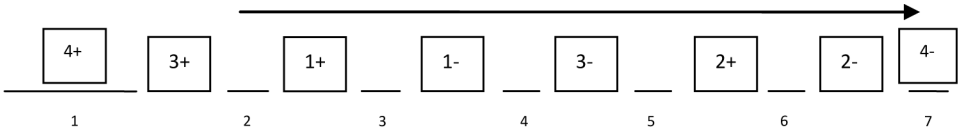


Figure 3.10: Illustrative example of fix-forward insertion: 1^{st} fix (pickup) and 7^{th} forward (delivery) position

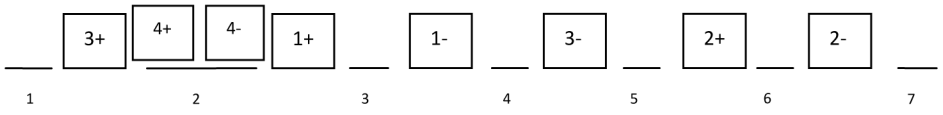


Figure 3.11: Illustrative example of fix-forward insertion: 2^{nd} fix (pickup) and 2^{nd} forward (delivery) position

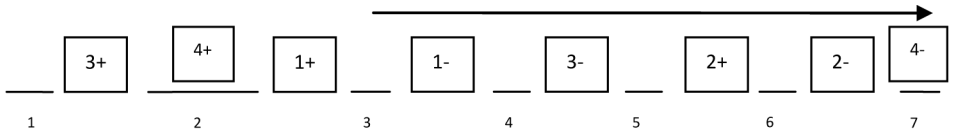


Figure 3.12: Illustrative example of fix-forward insertion: 2^{nd} fix (pickup) and 7^{th} forward (delivery) position

The fix-forward method without reduction rule or GENERATE has a known time complexity or neighbourhood exploration of

$$O(\sum_{i=0}^{n}(n+1-i))$$

Reduction Rule due to Vehicle Capacity

Generally, the normal procedure in each neighbourhood exploration is to insert each request and check to see if the sum of the load violates the vehicle capacity. To reduce computational time, a novel reduction rule is proposed by considering vehicle capacity as the hard constraint. Therefore, the insertion continues without calculating the sum of load status again and again. The procedure is comprised of three steps. For example, in Figure 3.13, there are three requests: (1+,1-), (2+,2-), (3+,3-). These have the supply and demand units of goods as follows;

$(1+, 1-) = (+1.3, -1.3)$

$(2+, 2-) = (+1.5, -1.5)$

$(3+, 3-) = (+1.2, -1.2)$

First step: As shown in Figure 3.13, the current load status in the route is calculated. For $n+1$ insertion locations, the vehicle load at the 1^{st} location is always set to zero. The vehicle loads at 2^{nd} toward $(n+1)^{th}$ positions are the accumulated sum of load status once visited.

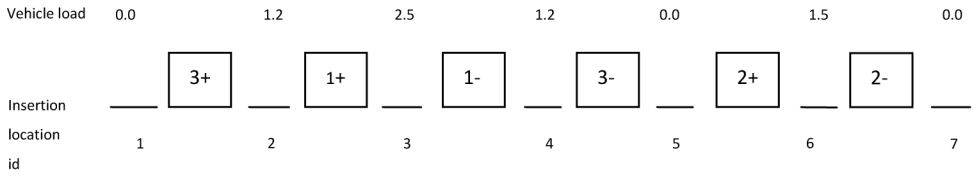


Figure 3.13: Illustrative example of reduction rule in fix-forward insertion: vehicle loading

Second step: In Figure 3.14, the new request to insert is considered and the vehicle load status from 1^{st} to $(n+1)^{th}$ positions is updated. For example, the pair request $(4+, 4-)$, which has the supply and demand units of $(+0.8, -0.8)$, is inserted in this route. The vehicle load in each insertion location is then updated by summing the supply unit of the customer's pickup request with the vehicle load.

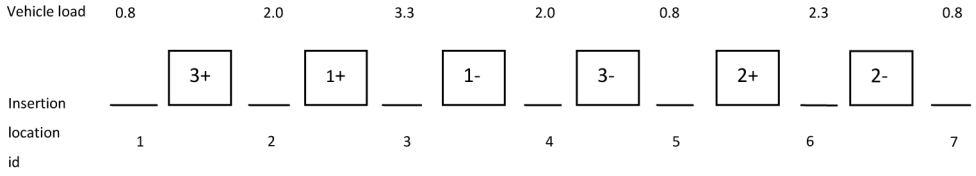


Figure 3.14: Illustrative example of reduction rule in fix-forward insertion: updated vehicle loading

Third step: In Figure 3.15, the vehicle load is checked against the vehicle capacity. In any insertion location, if the vehicle loads exceeds the vehicle capacity, then that location is marked. For example, the vehicle capacity is 2.9, then the 3^{rd} pickup location where the load exceeds the vehicle capacity is marked.

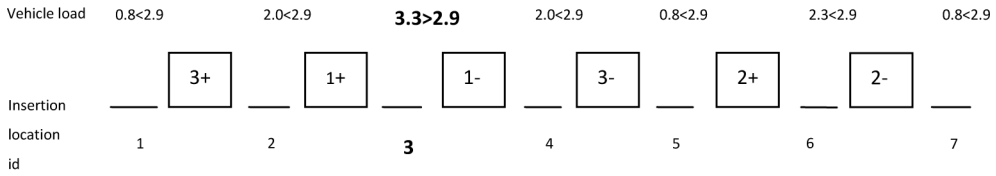


Figure 3.15: Illustrative example of reduction rule in fix-forward insertion: marked inserting location

The pickup node cannot be inserted in the marked insertion location and, for any fixed pickup position, the delivery request can only be moved up to the position before the marked insertion location.

There are several advantages of using the fix-forward insertion method embedding with this reduction rule. First, it ensures that the paired request is served by the same vehicle. Second, unlike the method used in Moon et al. (2002) which transformed the route into a network graph for validating precedence relationships, this proposed rule is simple and avoids checking precedence relationships between the pickup and corresponding delivery requests. Third, it avoids checking vehicle capacity violation. In addition, by preventing insertion at the expectedly violated inserting location, it eliminates redundancy in insertion and recalculating total load at all insertion locations again and again. Moreover, the reduction rule does not abandon any feasible location. The experiments validate the reduction rule against the normal procedure. The reduction rule significantly reduces the computational time, compared to the normal procedure.

3.7 Computational Experiments

3.7.1 Implementation and Parameter Setting

The MA was implemented in Microsoft Visual Studio 2008: C#, on Intel Core 2, Processor 2.49 GHz, 3.48 GB of RAM and evaluated on the 32 test instances, as described in Section 3.7.2, solved by CPLEX.

It is widely known that obtaining good GA parameter setting that works for a particular problem is a non-trivial task. There are a number of critical factors to determine a robust parameter setting. These include population size, number of generations, genetic representation, type of selection and genetic operator probabilities.

The experiments were conducted for tuning parameters using 4 representative test

Parameters	Setting
Population size	400
Chromosome initialisation	random
Min-Max generation span	300-3000
Termination ratio if not improved	0.4
Tournament prob.	0.6
Recombination prob.	0.75
Intra-depot operator prob.	0.3
Inter-depot operator prob.	0.3

Table 3.1: Experimental parameters

instances. The initial set of parameters was set, based on experience while developing the heuristic, then, these parameters were improved one-by-one. For each parameter, a number of values in a specified range are allowed while the rest of the parameters is kept fixed. For each parameter setting, we apply the heuristic on our set of test problems five times. The best setting that produces the best average gap is selected. Then, the next parameter is experimented on. These parameters include the population size, % of heuristic initialisation, tournament probability, recombination probability, intra-depot probability and inter-depot probability. The experimental parameters found are shown in Table 3.1.

It should be noted that the algorithm stops when the best solution has not been improved for the last $0.4 \times n$ generations.

3.7.2 Computational Results

Each test instance is run 10 times using the proposed MA. The results are then compared to the optimal solutions obtained from CPLEX. The average and best values of 10 solutions, and its average computational time are demonstrated in Table 3.2 and Table 3.3. In these tables, each geographical distribution (Geo dist.) uses a varied number of problem parameters, including vehicles' capacity (Veh Cap), maximum route

length (Max Legt), the distribution of vehicles in each depot (Veh/Dep) and the number of depots. The geographical distributions are uniform (U.), clustered (C.) or semi-clustered (SC.). Veh Const. refers to the vehicle constraints. In terms of computational results, Sol. represents the objective value and Gap(%) shows the percentage deviation of GA's solutions from the optimal solutions obtained by CPLEX. hr:mm:ss refers to the computational time.

The network typology of the optimal solutions for test instances 2, 4, 6, 13, 19, and 20 are shown as a, b, c, d, e, f, respectively in Figure 3.16. The results are obtained from CPLEX so that the proposed Memetic Algorithm can be comparatively validated. In order to obtain the optimal solution of large solvable problems by CPLEX, the computational time is limited to 4 days.

Over 32 test instances, the average value of MA's avg. sol. in 10 runs for 32 test problems is 0.015%. The average values of percentage deviation in MA's avg. sol. of clustered problems and those with borderline customers are 0.095% and 0.15% respectively. The MA can find the optimal solutions for all test instances within 10 runs. The average computational time of the MA is approximately 6 seconds while that of CPLEX is more than 16 hours. Therefore, the experiment shows that the clustered problems and those with borderline customers are rather difficult to be solved than the uniform and semi-clustered problems. Overall, the MA can produce competitive results, compared to those obtained by CPLEX, in reasonable time for this set of small-sized test problems.

3.8 Summary

The Multi-depot Pickup and Delivery Problem (MDPDP) is one of the NP-hard problems which arise in real-life logistics problems. In this Chapter, a mathematical formulation for MDPDP was presented. Various test instances were generated by adapting test instances from the literature. The problem parameters that critically in-

No.	Dist.	Problem size			Veh Const.		Veh/Dep		CPLEX's		MA's avg. sol.		MA's best sol.	
		Dep	Req	Veh	Legt	Cap	1	2	Sol.	hr:mm:ss	Gap (%)	hr:mm:ss	Gap (%)	hr:mm:ss
1	U.	2	10	4	1400	30	2	2	5498.66	00:05:44	0	00:00:05	0	00:00:05
2		2	10	4	1400	30	2	2	5485.84	01:47:48	0	00:00:07	0	00:00:07
3	C.	2	10	4	1900	30	2	2	4522.23	04:02:13	0	00:00:05	0	00:00:05
4		2	10	4	1900	30	2	2	4030	33:11:17	0	00:00:07	0	00:00:07
5	SC.	2	10	4	2000	30	2	2	4623.02	00:15:32	0	00:00:04	0	00:00:04
6		2	10	4	2000	30	2	2	4339.13	04:08:09	0	00:00:06	0	00:00:06
7	U.	2	10	4	1550	30	2	2	5345.88	03:36:46	0	00:00:07	0	00:00:07
8		2	10	4	1700	30	2	2	5235.09	06:11:00	0	00:00:07	0	00:00:07
9	C.	2	10	4	2000	30	2	2	3922.74	07:18:56	0.11	00:00:07	0	00:00:10
10		2	10	4	2250	30	2	2	3774.34	04:20:13	0.08	00:00:07	0	00:00:07
11	SC.	2	10	4	2300	30	2	2	4292.64	07:25:07	0	00:00:06	0	00:00:06
12		2	10	4	2500	30	2	2	4184.56	26:13:13	0	00:00:07	0	00:00:07

Table 3.2: Comparison of GA with the optimal solutions, solved by CPLEX, in increasing vehicles' capacity

No.	Dist.	Problem size			Veh Const.		Veh/Dep				CPLEX's		MA's avg. sol.		MA's best sol.	
		Dep	Req	Veh	Legt	Cap	1	2	3	4	Sol.	hr:mm:ss	Gap (%)	hr:mm:ss	Gap (%)	hr:mm:ss
13	U.	2	10	4	1500	30	1	3	-	-	5879.79	29:08:51	0	00:00:08	0	00:00:08
14		2	10	4	1500	30	3	1	-	-	5017.46	00:46:07	0	00:00:07	0	00:00:07
15	C.	2	10	4	1900	30	1	3	-	-	4401.59	75:29:54	0	00:00:08	0	00:00:08
16		2	10	4	1900	30	3	1	-	-	4030	05:53:33	0	00:00:07	0	00:00:07
17	SC.	2	10	4	2000	20	2	2	-	-	4339.13	06:42:54	0	00:00:07	0	00:00:07
18		2	10	4	2000	30	2	2	-	-	4455.35	24:41:31	0	00:00:07	0	00:00:06
19	U.	3	10	3	2000	30	1	1	1	-	4436.64	00:12:15	0	00:00:05	0	00:00:05
20		4	10	4	2000	30	1	1	1	1	4004.53	00:15:05	0	00:00:04	0	00:00:04
21	C.	3	10	3	2000	30	1	1	1	-	3320.42	00:55:25	0	00:00:04	0	00:00:04
22		4	10	4	2000	30	1	1	1	1	2845.25	00:25:58	0	00:00:04	0	00:00:03
23	SC.	3	10	3	2000	30	1	1	1	-	3984.85	01:55:37	0	00:00:04	0	00:00:04
24		4	10	4	2000	30	1	1	1	1	3437.79	01:14:23	0	00:00:04	0	00:00:04
25		2	9	4	2400	30	2	2	-	-	8948.19	10:18:20	0.18	00:00:08	0	00:00:08
26		2	9	4	2400	50	2	2	-	-	7367.41	102:07:45	0	00:00:08	0	00:00:08
27		2	9	4	2600	20	2	2	-	-	9015.89	18:31:46	0	00:00:06	0	00:00:06
28		2	9	4	3000	20	2	2	-	-	7748.41	67:41:18	0	00:00:07	0	00:00:07
29		2	9	4	2600	20	1	3	-	-	9015.5	05:52:16	0.12	00:00:07	0	00:00:07
30		2	9	4	2600	20	3	1	-	-	9016.84	06:05:19	0	00:00:07	0	00:00:07
31		3	9	3	3000	20	1	1	1	-	7926.84	37:13:40	0	00:00:05	0	00:00:05
32		4	9	4	3000	20	1	1	1	1	7907.19	35:12:30	0	00:00:04	0	00:00:04

Table 3.3: Comparison of GA with the optimal solutions, solved by CPLEX, in varying the distribution of vehicles in each depot

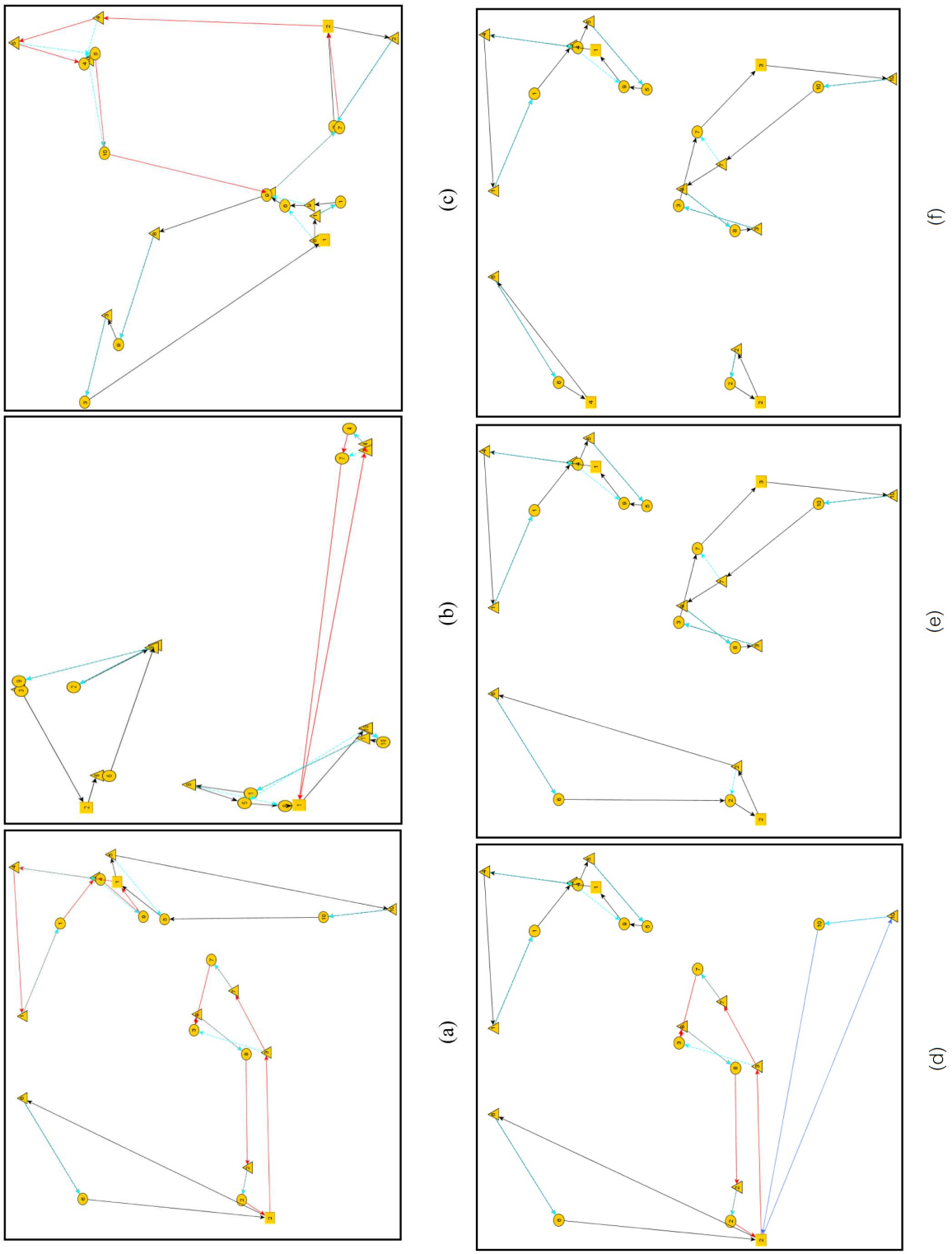


Figure 3.16: The optimal solutions of test instances containing three geographical distributions:uniform (a), clustered (b), semi-clustered (c) and 2 (d), 3 (e), 4 (f) depots in the uniform distribution

fluence the computational time of CPLEX were investigated. The performance features available in CPLEX 11.0 were tuned to solve the 32 test instances. Due to being an NP-hard problem, exact methods cannot handle larger problem sizes within a reasonable time frame. As a result, we resort to using a meta-heuristic approach to solve the problem to near optimality in a timely manner. A Memetic Algorithm (MA) for the MDPDP was presented. Chromosome representation, genetic operators and framework were designed. Several removal and insertion heuristics are used to search the neighbouring solutions. The fix-forward insertion method is presented with a reduction rule for improving the computational time. The implementation and evaluation of the MA were conducted on a set of 32 test instances that were solved to optimality by CPLEX. The computational study demonstrates the capability of the proposed MA to find near optimal solution within seconds. The literature review shows that the research relating to this problem and its variants still receive limited attention in the literature. This calls for further investigation of GAs, MAs and other meta-heuristics to solve the related MDPDPs.

Chapter 4

An Adaptive Memetic Large Neighbourhood Search for the Multi-Depot Pickup and Delivery Problem with Time Windows and Special Requests

4.1 Introduction

Vehicle Routing and Scheduling Problems are one of the most important problems in managing logistics and supply chain networks. Efficient routes not only reduce cost significantly, but also maintain the service level. The vehicle routing problem with time window (VRPTW) is one of the most studied NP-hard combinatorial optimisation problems. It consists of designing minimum cost routes for a fleet of vehicles to satisfy a set of requests within specified time windows. The pickup and delivery

problem with time windows (PDPTW) is a variant of VRPTW where each request is served from a pickup location to its delivery location by the same vehicles within specified time windows. The Multi-Depot Pickup and Delivery with Time Windows (MDPDPTW) is an extension of PDPTW in which the fleet of vehicles are located in several depots. The problem concerns a core basis of managing a fleet of vehicles in Logistics Service Providers (LSPs), Third-Party Logistics Providers (3PLs), horizontal cooperation among freight carriers. These providers can apply the MDPDPTW to solve Full-Truck-Load and Less-Than-Truck-Load transportation requests in a large geographical coverage area.

As it is a special case of VRPTW, the MDPDPTW is an NP-hard problem. The optimal solution cannot be obtained in a reasonable computational time using exact approaches, especially when the problem size is large. In real-life scenarios, both a reasonable computational time and good solution quality must be achieved. As a result, we resort to meta-heuristics with the view to obtaining near optimal solutions in a timely manner.

In this Chapter, we designed an Adaptive Memetic Large Neighbourhood Search (AMLNS), which incorporates several local search operators, in order to solve a variant of MDPDPTW. This variant extends the MDPDPTW in such a way that the heterogeneous fleet of vehicles and special requests are considered. Moreover, routes can depart and return to different depots. We refer this problem as a Multi-depot Pickup and Delivery Problem with Time Windows and Special Requests (MD-PDPTW-SR).

4.2 State-of-the-art Reviews of Related Problems

Ropke and Pisinger (2006) recently studied the PDPTW and multi-depot PDPTW with special requests (MD-PDPTW-SR). The authors applied an Adaptive Large Neighbourhood Search (ALNS) to efficiently solve the large test instances of Li and Lim (2001), which employed up to 500 requests or 1000 locations. The ALNS is composed

of a number of competing sub-heuristics that are used with a frequency corresponding to their recorded performance. The heuristic was tested on more than 350 benchmark instances of PDPTW, and it was able to obtain the new best known solutions for more than 50% of the problems upon Bent and Van Hentenryck's (2004) computational results. Ropke and Pisinger (2006) also generated new test instances for MD-PDPTW-SR and reported the computational results. They confirmed that the use of several competing sub-heuristics, instead of just one, results in robustness. The ALNS demonstrated the capability of handling such large instances in a reasonable time period.

Pisinger and Ropke (2007) further applied the ALNS to solve variants of the vehicle routing problems. The authors slightly modified the ALNS from that of Ropke and Pisinger (2006) by incorporating more removal heuristics. The heuristic demonstrated robustness and efficiency by improving a large number of best known solutions for the 486 benchmark instances of VRPTW variants. ALNS is an extension of the large neighbourhood search framework of Shaw (1998) with an adaptive layer that chooses a number of removal and insertion heuristics to intensify and diversify the search. In addition to Shaw (1998), in which the algorithm accepts only better solutions, Ropke and Pisinger (2006) and Pisinger and Ropke (2007) applied Simulated Annealing (SA), which, on occasion, accepts solutions being worse than the current solution leading to a high-quality solution.

The computational result of ALNS for solving the multi-depot PDPTW with special requests (MD-PDPTW-SR) in Ropke and Pisinger (2006) is reported at www.diku.dk/~sropke. The authors further clarified that the ALNS that produced the computational result in the website is the updated ALNS from Pisinger and Ropke (2007). [1].

Dondo et al. (2007) presented a new Mixed Integer Linear Programming (MILP) for

[1] On the website, the author mentioned "that these are not the results from the paper, but a table constructed later on, with a somewhat updated heuristic. The original results appear to be lost."

the multiple vehicle pickup and delivery problem with time windows (MVPDPTW). The formulation is capable of dealing with heterogeneous vehicles, multiple depots, many-to-many requests and pure pickup/delivery nodes. The optimal solutions of the problems with 36 locations including one test instance of multi-depot PDPTW were solved by ILOG OPL.

Therefore, the only available heuristics and large size standard benchmark test instances for solving the MD-PDPTW-SR are those of Ropke and Pisinger (2006). For other related problems, Bent and Van Hentenryck (2004) proposed a two-stage hybrid algorithm to solve the PDPTW. The first stage uses a simple simulated annealing algorithm to reduce the number of routes, while the second stage applies a Large Neighbourhood Search (LNS) to decrease the total travel cost. The heuristic was also used to experiment on the test instances of Li and Lim (2001). It demonstrated the improvement of 47% and 76% of the best solutions on the 200 and 600 customer benchmarks, respectively.

Variants of Genetic Algorithms (GAs) were also used to solve the PDPTW. Pankratz (2005) applied the Grouping Genetic Algorithms to solve Nanry and Barnes (2000)'s and Li and Lim (2001)'s benchmarks, up to 100 requests. The computational results are competitive with the results of Nanry and Barnes (2000) and Lau and Liang (2001). Nagata and Kobayashi (2010) applied a Memetic Algorithm (MA) to solve the PDPTW. The authors developed a particular crossover operator to tackle this tightly constrained problem. A simple hill climbing algorithm with the first improvement strategy is then used as the local search algorithm. The MA was tested on Li and Lim's (2001) benchmarks and improves almost 50% of the best-known solutions upon those of Ropke and Pisinger (2006) and Bent and Van Hentenryck (2004).

4.2.1 Gaps in the Literature

Reduction in the number of removed requests

For the ALNS, Pisinger and Ropke (2007) observed that it may be beneficial to reduce the number of requests (q) that are removed in each iteration as the simulated annealing framework generally accepts only minor changes toward the end of the search. As a result, this could speed up the algorithm or allow it to perform more iteration within the same amount of time. We believe that, in order to schedule this reduction, a master local search framework should equip this mechanism. However, the ability to accept solutions in SA is controlled by the exponential probability function and temperature. Both of which are difficult to be manipulated in different problem sizes.

In this Chapter, we aim to redesign the algorithm so that the master local search framework allows the continuous exploration of good solutions, with a schedule of small requests to remove, while maintaining the same search behaviour as the SA.

Parallelising the ALNS

Ropke (2009b) attempted to design a parallel ALNS (PALNS) for solving variants of VRPs with the view to speeding up the search. In brief, one current solution and one global best solution was shared among the worker threads and each thread obtained a copy of the current solution and performed "destroy and repair" operations on its local copy. The shared current and global best solutions were updated as necessary. The weights of destroy and repair methods, the temperature of the SA and iteration counter were also shared to the worker threads. Other operators were similar to those in Pisinger and Ropke (2007). The computational results showed linear speedup. However, the author believed that the PALNS seemed to be working against the SA principle. For example, the authors considered a current solution x at some point during the search. In a sequential LNS, the search might move away from this solution during,

for example, 8 iterations. However, a parallel LNS may move away from this solution for 7 iterations, but when the destroy and repair operation that was initiated with x is finished, it may move back to a solution close to x and thereby cancel the work done in the intermediate iterations. They suggested that further experiments are necessary to fully understand this effect.

In our view, instead of searching one single solution in parallel using different operators, it may be beneficial if a number of diverse solutions were searched using a population-based approach in parallel, due to the following reasons:

Firstly, Jones (1995) mentioned that one particular operator has one search landscape. In ALNS or PALNS, we believe that the randomised parameters, the number of operators and the random number of requests to remove within a limited range, increase the degree of freedom of the search. Therefore, it is difficult to control the search direction that attempts to move away from the current solution without cycling. With the use of shared weights in the roulette wheel in PALNS, when some operators have high probability of selection but the same operators may be selected with some differences due to randomness. Even though the PALNS applies parallel computing to search for several solutions at a time, at every iteration, it then starts from a single solution previously accepted. Therefore, it should still be categorised as a single-solution approach or trajectory method, while, the operators of population-based approaches work on diverse populations or several reference points. As a consequence, it is possible that the accepted solution may not move far away from the previously accepted solution and the search spreading from a single solution may concentrate on one basin of attraction. As a result, it is rather difficult to prevent cycling.

Secondly, in Pisinger and Ropke (2007), several moves have a distance of zero, meaning that no changes were made to the solution vectors. Thus, such moves should be avoided. In addition, we believe that searching one current solution to several near solutions in parallel may not be able to move some solutions trapped in deep local optima. Moreover, some basins of attraction containing local optima may be far away

from one another. We expect that searching several diverse solutions in parallel, like in typical population-based approaches, can potentially explore more search space.

One way to handle these suggestions is to hybridise the ALNS with a population-based approach in order to combine strengths and counteract limitations. Thus, we sought a rather simple metaheuristic as the local search framework at the master level. However, each metaheuristic framework has its own philosophy, characteristic and behaviour. All key components and their contribution toward intensification and diversification must be investigated. In hybrid metaheuristics, the synergy effect of diversification and intensification is essential. Therefore, we investigated the design principle of hybrid metaheuristics accordingly.

4.3 Problem Formulation

The formulation is based on Desaulniers et al. (2002) and Ropke and Pisinger (2006). The authors presented the formulation for the PDPTW that can be adapted to the (MD-PDPTW-SR).

4.3.1 Notations

i location

n number of pickup and delivery request

K set of all vehicles

m number of vehicles, $m =\mid K \mid$

P set of pickup nodes, $P = \{1, \ldots, n\}$

D set of delivery nodes, $D = \{n+1, \ldots, 2n\}$

l_i demand/supply at vertex i

a_i earliest time to begin service at vertex i

b_i latest time to begin service at vertex i

s_i service duration at vertex i

K_i set of vehicles that are able to serve request i,

P_k set of pickups that can be served by vehicle k;$P_k \subseteq P$

D_k set of deliveries that can be served by vehicle k;$D_k \subseteq D$

τ_k set of start terminal of vehicle k;

$\tau_k^{'}$ set of end terminal of vehicle k;

d_{ij} distance from vertex i to j

t_{ij} travel time from vertex i to j

C_k Capacity of vehicle$k$

x_{ijk} $= \begin{cases} 1 & , if\ arc\,(i,j)\ is\ traversed\ by\ vehicle\ k \\ 0 & , else \end{cases}$

z_i $= \begin{cases} 1 & , if\ request\ i\ is\ placed\ in\ the\ request\ bank \\ 0 & , else \end{cases}$

S_{ik} a non-negative number that indicates when truck k with semi-trailer t starts the service at location i

L_{ik} a non-negative number that indicates space of truck k with semi-trailer t when leaving vertex i

4.3.2 Mathematical Model

A mathematical model of the problem is

$$Min \quad \alpha \sum_{k \in K} \sum_{(i,j) \in A_k} d_{ij} x_{ijk} + \beta \sum_{k \in K} (S_{\tau'_k,k} - a_{\tau_k}) + \gamma \sum_{i \in P} z_i \tag{4.1}$$

Subject to:

$$\sum_{k \in K_i} \sum_{j:(i,j) \in A_k} x_{ijk} + z_i = 1 \qquad \forall i \in P \tag{4.2}$$

$$\sum_{j:(i,j) \in A_k} x_{ijk} - \sum_{j:(n+i,j) \in A_k} x_{n+i,j,k} = 0 \qquad \forall k \in K, \forall i \in P_k \tag{4.3}$$

$$\sum_{j \in P_k \cup \{\tau'_k\}} x_{\tau_k,j,k} = 1 \qquad \forall k \in K \tag{4.4}$$

$$\sum_{i \in D_k \cup \{\tau_k\}} x_{i,\tau',k} = 1 \qquad \forall k \in K \tag{4.5}$$

$$\sum_{i:(i,j) \in A_k} x_{ijk} - \sum_{i:(i,j) \in A_k} x_{jik} = 0 \qquad \forall k \in K, \forall j \in N_k \tag{4.6}$$

$$x_{ijkt} = 1 \Rightarrow S_{ikt} + s_i + t_{ij} \leq S_{jkt} \qquad \forall k \in K, \forall (i,j) \in A_k \tag{4.7}$$

$$a_i \leq S_{ik} \leq b_i \qquad \forall k \in K \forall, i \in V_k \tag{4.8}$$

$$S_{ik} \leq S_{n+i,k} \qquad \forall k \in K, \forall i \in P_k \tag{4.9}$$

$$x_{ijkt} = 1 \Rightarrow L_{ikt} + l_j \leq L_{jkt} \qquad \forall k \in K, \forall (i,j) \in A_k \tag{4.10}$$

$$L_{ik} \leq C_k \qquad \forall k \in K, \forall i \in V_k \tag{4.11}$$

$$L_{\tau_k k} = L_{\tau'_k k} = 0 \qquad \forall k \in K \tag{4.12}$$

$$
\begin{aligned}
x_{ijk} &\in \{0,1\} && \forall k \in K, \forall (i,j) \in A_k \\
z_i &\in \{0,1\} && \forall i \in P \\
S_{ik} &\geq 0 && \forall k \in K, \forall i \in V_k \\
L_{ik} &\geq 0 && \forall k \in K, \forall i \in V_k
\end{aligned}
$$

It is important to note that some vehicles can only service some requests. For example, a request might require the vehicles with freezing compartment or other compatible equipment. Therefore, for all i and k: $k \in K_i \Leftrightarrow i \in P_k \wedge i \in D_k$. Special requests are those where $K_i \neq K$. Let $N = P \cup D$ and N_k=$P_k \cup D_k$. Denote $\tau_k = 2n + k$, $k \in K$, and $\tau_k^{'} = 2n + m + k$, $k \in K$ be the nodes that represent the start and end depots of vehicle k, respectively. The graph $G = (V, A)$ contains the nodes$V = N \cup \{\tau_1, \ldots, \tau_k\} \cup \{\tau_1^{'}, \ldots, \tau_m^{'}\}$ and the arcs $A = V \times V$. For each vehicle, we have a subgraph $G_k = (V_k, A_k)$, where $V_k = N_k \cup \{\tau_k\} \cup \{\tau_k^{'}\}$ and $A_k = V_k \times V_k$. For each arc $(i, j) \in A$, the distance $d_{ij} > 0$ and a travel time $t_{ij} > 0$. The triangle inequality for time is satisfied: $t_{ij} < t_{il} + t_{lj}$ for all $i, j, l \in V$. For each node $i \in N$, $l_i > 0$ for $i \in P$, and $l_i = -l_{i-n}$ for $i \in D$.

The objective function in Equation 4.1 is to minimise the weighted sum of the travelled distance, the sum of the time spent by all vehicles and the number of requests not scheduled. Constraint 4.2 ensures that each pickup location is visited or placed in the request bank. Constraint 4.3 ensures that both pickup and corresponding delivery requests are visited by the same vehicle. Constraints 4.4 and 4.5 ensure that every vehicle departs from a start terminal and return to a designated end terminal. Together with constraint 4.6, this ensures that consecutive paths between τ_k and $\tau_k^{'}$ are established for each $k \in K$. Constraints 4.7 and 4.8 ensure that S_{ik} is set correctly along the paths and that satisfy time windows of i. These constraints also prevent sub-tours. Constraint 4.9 ensures that each pickup precedes its delivery location. Constraints 4.10, 4.11 and

4.12 ensure that the load variables is formed correctly along the paths and satisfy the vehicle capacity.

4.4 Design of Hybrid Metaheuristics

As discussed in Section 2.3.4, we first survey the related methodologies that provide competitive results for variants of vehicle routing problems. The conceptual design is described in order to understand the rationales underpinning our hybrid metaheuristic.

4.4.1 State-of-the-Art Review of Related Methodologies

Rapid changing environments in business need solution methods that are fast, easy to understand, flexible, accurate, and robust in terms of consistent performance across different problems. Gendreau and Tarantilis (2010) surveyed the state-of-the-art metaheuristics for solving large-scale problem instances of Vehicle Routing Problems. The authors suggest that many approaches have failed to provide a good compromise between quality and computational time while a few approaches scored well on other dimensions, such as simplicity and flexibility. Parallel and cooperative search methods should be considered to take advantage of available multiple CPUs. The authors concluded that Nagata et al. (2010), using a penalty-based edge assembly Memetic Algorithm, is one of the most effective and efficient approaches. In terms of the simplicity of its structure, Pisinger and Ropke (2007), using ALNS, demonstrated a good compromise between speed and accuracy. The ALNS has also been applied to a wide variety of different VRP variants due to its simplicity and flexibility. Braysy (2004b) also scored well due to its simple structure, using multi-start local search with Threshold Accepting (TA). We also noticed that the concept of Memetic Algorithm proposed by Nagata et al. (2010) for solving VRPTW is similar to that of Nagata and Kobayashi (2010) for solving PDPTW.

In terms of methodological comparison, Yagiura and Ibaraki (2001) investigated

several metaheuristics and compared their performance. In their experiments, TA was shown to be the best metaheuristic when compared to several other metaheuristics namely, SA, GLS, ILS, GDA, GRASP, GA, MLS, TA. Braysy (2012) confirmed that the record-to-record travel algorithm and TA are among the most efficient metaheuristics in the literature.

Ulder et al. (1991) argued that Genetic Local Search should not be viewed as being opposed to SA and TA because elements of these strategies can be implemented in Genetic Local Search at the improvement or selection step. Genetic Algorithms and Memetic Algorithms, as population-based approaches, proved successful in efficiently solving variants of VRPs and PDPTW. We then focused on the Memetic Algorithm by Nagata and Kobayashi (2010) to hybridise with the ALNS, using TA, in order to obtain a fast and reliable metaheuristic.

4.4.2 Conceptual Design

Blum and Roli (2003) report that a current trend is the hybridisation of methods in the direction of the integration of a single point search algorithm into population-based ones. Grefenstette (1987), Goldberg (1989), Merz and Freisleben (1999), Hart et al. (2005) and Blum and Roli (2003) showed that GAs are useful for identifying good areas of the search space, i.e. exploration but they are often less good at refining near-optimal solutions i.e. exploitation. GAs use diverse population to search in different regions of the search space, which then restores the search of promising solutions rather than replacing the single solution. Goldberg (1989) stated that, when problem-specific information exists, it may be advantageous to consider a GA hybrid. GAs may be crossed with various problem-specific search techniques to form a hybrid that exploits the global perspective of the GA and the convergence of the problem specific technique. Ulder et al. (1991), Davis (1991) and Blum et al. (2011) confirmed that metaheuristic hybrids, in some way, are often successful at managing to combine the advantages of population-based methods to ensure an exploration of the search space with the strength

Algorithm 4.1 Pool Template for Hybrid Metaheuristics

Initialise pool P by an external procedure;
while termination = FALSE **do**
  $S \leftarrow OF(P)$;
  if $\mid S \mid > 1$ **then**
    $S' \leftarrow SCM(S)$
  else
    $S' \leftarrow S$;
  $S'' \leftarrow IM(S')$;
  $P \leftarrow IF(S'')$
Apply a post-optimising procedure to P.

of trajectory methods and to help identify quickly good areas in the search space.

Raidl (2006), Greistorfer and VoB (2005) discussed a pool template by which they cover different classes of metaheuristics and hybrids. The authors pointed out that most existing metaheuristics share some ideas but differ in certain characteristics and key components. "Making these key components explicit and collecting them yields a toolbox of components from which one can choose in the design of an optimisation algorithm, as it seems to be most appropriate for the target problem at hand." Greistorfer and Voß (2005) introduced a pool template, as shown in Figure 4.1.

In Figure 4.1, P represents $Pool$, IF/OF stand for $Input/Output\ Function$. IM represents $Improvement\ Method$ and SCM stands for $Solution\ Combination\ Method$. Interpreting metaheuristics as instances of such a common template results in a decomposition of the algorithms.

We perceived that this pool template can provide a unified view of metaheuristics and their hybrids. It comprehensively covers the single-solution and population-based approaches in terms of their key components. We attempted to combine strengths and eliminate weaknesses from the selected metaheuristics by using the ALNS as a point of departure i.e. possible improvements from Pisinger and Ropke (2007), as discussed in Section 4.2.1. There are several possible hybrids of key components from the metaheuristics of Nagata and Kobayashi (2010), Pisinger and Ropke (2007) and

others. Therefore, we begin with the suggestions of Pisinger and Ropke (2007): (1) use a small number of requests when necessary and (2) make the ALNS parallel. We consider that Threshold Accepting (TA), a deterministic version of SA, can easily schedule the use of a small number of requests to remove. Also, it is widely known that GAs and MAs are among the most successful metaheuristics used for combinatorial optimisation problems. Therefore, the conceptual design of our proposed hybrid metaheuristic is discussed in regards to the key elements of GAs, ALNS and TA perspectives by framing into the pool template. Moreover, the intensification and diversification effects of each operator and their interactions are discussed.

Pool **and** S

Greistorfer and Voß (2005) defined that, for example, simulated annealing in terms of this template has $\mid S \mid = 1$ and when the overall so-far best solution is collected $\mid P \mid = 2$.

Goldberg (1989) pointed out that a single-solution approach may locate a false peak while population-based approaches such as GAs climb many peaks in parallel. Blum and Roli (2003) confirmed that population-based algorithms provide a natural, intrinsic method for the exploration of the search space.

In our opinion, the concept of GAs should avoid the PALNS concern of Ropke (2009b), specifically that the search may return to the near locations solutions visited earlier. Moreover, it should help in the exploration of the search space where local optima may be far away from each other in tightly constrained problems. The use of population constitutes the diversification effect.

$OF(P)$

Raidl (2006) identified OF, output function, as selection technique for GAs while the SA simply returns the current solution. When using a population-based approach such as GAs, one important operator is the selection operator, e.g. binary tournament

selection. Goldberg (2002) described that the selection pressure ensures the propagation of good building blocks or subassemblies of good solutions. Therefore, the tournament selection introduces more intensification than diversification to the search process.

$SCM(S)$ and $IM(S)$

For GAs, Raidl (2006) identified SCM- crossover operators; IM-mutation operators, repair schemes, decoding functions. The population-based approaches such as GAs typically consist of SCM and IM, whereas the single solution approach only uses IM.

Goldberg (2002) explained that the basic concept is that GAs identify and recombine different subassemblies of good solutions to form high performance solutions. The recombination of these subassemblies is usually carried out by crossover or SCM. The crossover operator with repair, as proposed by Nagata and Kobayashi (2010) and Pankratz (2005), exchanges the routes between two selected parents. In this way, a good sequence of locations is still maintained. Nagata and Kobayashi (2010) suggested that meaningful building blocks from combining both parents must be inherited. The authors designed the Selective Route Exchange Crossover (SREX) to tackle this problem. Grefenstette (1987) also confirmed the success of using heuristic crossover operators. Since the selection pressure based on the overall tour length is insufficient to distinguish among small competing sub-tours, the probabilistic choices of selected-sub tours are however preferable to deterministic ones. Other well-performing hybrid GAs include the use of local search operators. The crossover, we suggest, constitutes to both diversification and intensification. In terms of diversification, the crossover changes or replaces some subassemblies of one solution from the other. This can be considered as one way to jump from one point to another. In terms of intensification, the high quality solutions are mated in order to apply the crossover. The crossover even recombines the good subassemblies from both solutions and expectedly forms a higher solution. We believe that heuristic crossover posses more diversification than intensification due to route relocation and implicit mutation.

Pisinger and Ropke (2007) demonstrated that the removal and insertion operators performed well on the variants of VRPs. These operators are referred to as Large Neighbourhood Search (LNS) operators, as originally introduced by Shaw (1998). By using these operators, Pisinger and Ropke (2007) specified the range of the number of requests to remove i.e. for small problems, the interval is $[0.1n, 0.4n]$ where n is the number of customers or requests, while for larger instances, it is [30,60]. Berger and Barkaoui (2004) considered large neighbourhood search as the mutation operators in a parallel hybrid genetic algorithm for the vehicle routing problem with time windows. The interval range is [12,17] customers in the problem size of 100 customers. Therefore, the LNS in Pisinger and Ropke (2007) can be viewed as mutation operators in variants of GAs. Moreover, we believe that when the number of requests to remove is small, the LNS acts as a local search. The LNS in Pisinger and Ropke (2007) is equipped with several removal and insertion operators, randomised parameters and random selection of the number of requests to remove. These operators help diversify the search. Pisinger and Ropke (2007) stated that the adaptive mechanism to choose among a number of insertion and removal heuristics is to intensify and diversify the search. Pisinger and Ropke (2007) confirmed that a search of the ALNS can quickly move away from the currently best known solution, compared to ideas of VNS where one tries to remain close to the currently best known solution. Therefore, we suggest that in the ALNS, the diversification force is stronger than intensification.

If the GA and the ALNS are hybridised, a number of population or points of search should help the ALNS to explore very far away from the currently best solution. This idea is preferable for the ALNS as mentioned in Pisinger and Ropke (2007). The population can be viewed as the memory of complete good solutions. The crossover operator can take advantage of the information from these single solutions by recombining good subassemblies. Moreover, we can perceive that crossovers, which consist of removing subassemblies before replacing them from another solution, causing some requests to move to the request bank, have the same mechanism as removal operators that try to

remove some requests before using insertion.

LNS operators apply on one solution. Therefore, crossovers should help LNS tackle the search from the population of solutions. The ALNS operators, as heavy mutation operators, should help the crossover in terms of avoiding premature convergence and maintaining the population diversity, since when the population become too similar or identical, crossing over two identical parents may generate the same offspring. For adaptive MAs, Krasnogor and Smith (2005) confirmed the significant improvement by using multiple local search operators and an adaptive mechanism. We expect that the crossover and ALNS are complementary key components that improve the search in hybrid metaheuristics.

The next question is where these operators should apply. Krasnogor and Smith (2005) pointed out that, by applying a local search after each of the genetic operators, the population of individuals consists solely of local optima. Aguirre and Tanaka (2002) found that applying mutation parallel to a crossover is more effective than mutation serial to crossover. The best performance was achieved by a parallel varying mutation self-adaptive GA. From this concept, it is interesting to incorporate the use of ALNS operators parallel to the crossover operator. From our experiments, the concept of parallel varying mutation self-adaptive GA provides promising results.

$IF(S)$

IF is the input function of solutions obtained, back to the *Pool*. Raidl (2006) mentioned that the input function, IF, of GAs refers to the replacement strategy. The IF of SA applies the Metropolis criterion in order to either accept or reject the new solution. The temperature update can also be considered part of the input function.

The replacement strategy in GAs has an impact on intensification and diversification. The replacement strategy used in Vidal et al. (2013) is the steady-state replacement strategy, similar to the steady-state replacement used in Syswerda (1991) and Pankratz (2005). It allows the deterioration of some good solutions. For example, a very good

solution is made worse but is still accepted after a pre-defined number of solutions,λ, is removed. However, without the additional mechanism developed by Vidal et al. (2013), we observed that the steady-state replacement which replaces worst solutions generally converges quickly and even worse for small-sized population. We also believe that the modified objective function and the population management of Vidal et al. (2013) can make the convergence slower. However, this mechanism is complex and requires considerable computational efforts. The MA of Nagata and Kobayashi (2010) only replaces one parent rather than both parents in the population. The author confirmed that this selection model is superior to conventional ones in maintaining population diversity because it prevents two-parent solutions from being replaced by two similar offspring solutions. The crossover of Nagata and Kobayashi (2010) ensured that the better offspring was more similar to the first parent than to the second parent. However, we view that this replacement strategy is highly intensified. Since only better offspring replaces the first parent; therefore, the solution may not traverse much of the search space. As a result, we attempt to simplify this mechanism using a replacement strategy similar to that of Nagata and Kobayashi (2010), with slight modifications by accepting some non-improving solutions according to Threshold Accepting (TA). TA diversifies the search in the beginning and intensifies the search toward the end.

Moreover, in order to schedule the use of a smaller number of removed requests, TA is quite flexible. Braysy et al. (2003) hybridised the Hybrid Genetic Algorithm (HGA) with TA post-processor and demonstrated a good performance over a set of 356 benchmark instances for the VRPTW. It is interesting to note that the recombination of the HGA is similar to that of Nagata and Kobayashi's (2010) in terms of removing the whole route at a time. Braysy et al. (2009) modified the TA of Dueck and Scheuer (1990) and improved upon the framework of Mester and Bräysy (2005) and Braysy et al. (2003). For other related applications, Liu (2011) improved the Genetic Local Search (GLS) with the TA by only applying a local search when the new solution is accepted by the TA. For our design, when using a smaller number of requests to remove, the

LNS becomes a local search. Then, toward the end of the search, the small number of requests to remove should be applied, so as to correspond to the accepting threshold and the intensification effect required for searching large-sized problems.

One may hybridise metaheuristics with or without special mechanisms. In this hybrid metaheuristic, some mechanisms are designed to integrate these key components together. Our algorithm stops after a number of iterations as in Ropke and Pisinger (2006) and Pisinger and Ropke (2007).

We reintegrate the key components with the view to constructing the mechanism that allows the balance between diversification and intensification. The optimal balance of intensification and diversification or exploitation and exploration is required.

4.5 An Adaptive Memetic Large Neighbourhood Search (AMLNS)

We proposed the Adaptive Memetic Large Neighbourhood Search (AMLNS) with the view to searching efficiently, improving computational time, and simplifying implementation. Moreover, considering the ALNS as the PDPTW solver as stated in Pisinger and Ropke (2007), the AMLNS should be able to substitute the ALNS and solve several variants of VRPs, once these are transformed to the Rich PDPTW. The AMLNS is a hybrid metaheuristic based on the ALNS, MA and TA. Following the discussion of each key component, hybrid metaheuristics should be recombined from the most promising key components. There are many possibilities for hybrid metaheurisitcs. We demonstrated the development of the AMLNS according to the discussion of conceptual design in Chapter 5. The AMLNS differs from typical ALNS in that the number of solutions, tournament selection, crossover, replacement and TA are used. The AMLNS differs from typical MAs in that adaptive mechanism, large neighbourhood search, and TA are used. The AMLNS differs from typical TAs in terms of being population-based and using crossover, tournament, replacement and large neighbourhood search operators. Moreover, some specialised mechanisms are incorporated such as a cut-off mechanism

and roulette wheel partitioning, as described in Section 4.5.7. We demonstrated the flowchart of the AMLNS in Figure 4.1 .

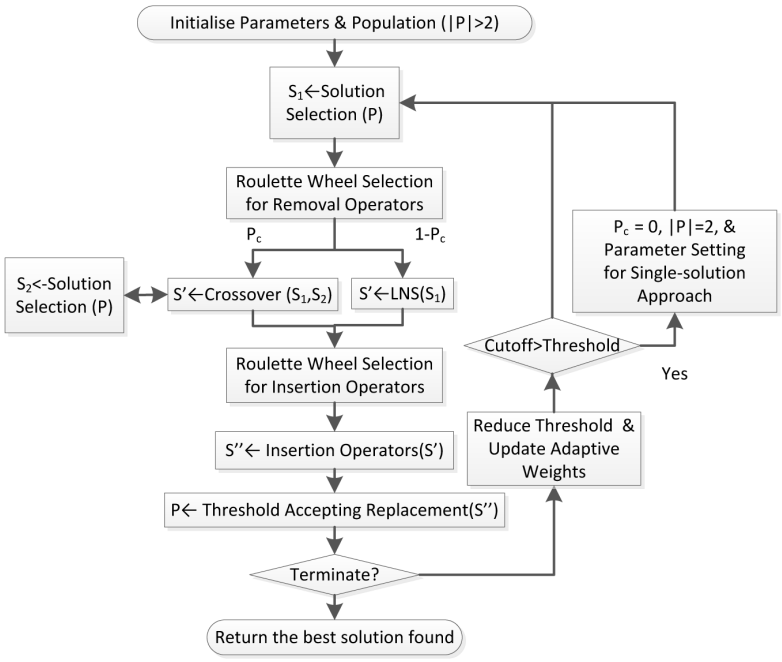


Figure 4.1: Flowchart of the AMLNS

According to Figure 4.1, the AMLNS first initialises the number of solutions, Threshold, and other parameters. One solution is selected by a tournament selection. Then, the AMLNS chooses a removal operator using roulette wheel selection. If a crossover is applied, second tournament selection chooses another solution. After that, the AMLNS chooses insertion operator using roulette wheel selection. A new solution $x^{'}$ is generated from x using the chosen removal and insertion operators. If x' can be accepted by Threshold, then set $x := x^{'}$. If $f(x) < f(x')$, set $x* = x$, where x^* is the best known solution of the search so far. The threshold and adaptive weights are updated. If Threshold is smaller than Cutoff Value, 2nd set parameters are used. The best solution of the population at the cut-off point is selected for further search. Until stop criteria is met, $x*$ is returned.

4.5.1 Removal Operators

The combination of different neighbourhood operators contributes to diversification. Each local search operator introduces a different search direction. The large neighbourhood search with a large number of requests to remove can also be considered as a heavy mutation in the context of GAs or MAs. The heavy mutation can drive a solution away from its current location in the search space.

In the AMLNS, we categorise the removal operators into unary and binary operators. The unary operators are those of Pisinger and Ropke (2007) while the binary operators are the crossover operators developed in this study. For additional details of the removal operators of the original ALNS, we refer the reader to Pisinger and Ropke (2007). However, we summarised these methods accordingly. The first seven removal heuristics return the number of pre-defined requests (q) to remove. In this study, we define the upper and lower bound q in both large- and medium-sized problems. The upper and lower bound of q in large-sized problems are $q_{l,up}$ and $q_{l,low}$. Similarly, the upper and lower bound of q in medium-sized problem are $q_{m,up}$ and $q_{m,low}$, respectively.

Random Removal (1)

This simple removal heuristic randomly selects q requests to remove. This removal heuristic aims to diversify the search.

Worst Removal (2)

This greedy heuristic removes requests with high costs and then inserts them at another place in the solution to obtain a better solution value. Intuitively, the measure is the difference between the objective function with and without that request. All requests are sorted by descending order according to the difference. The selection of requests

involves randomness parameters, p_w , substituted into p in Equation 4.13. The worst removal heuristic now repeatedly chooses a new request, having the largest cost until all q requests have been removed.

Most removal operators considered in this study apply the selection of randomised requests as described in Ropke and Pisinger (2006). Let L be the ranked list of all requests. $\mid L \mid$ is the number of requests. We choose a random number, y, from the interval $[0, 1)$. $p \geq 1$ is the determinism parameter that introduces some randomness in the selection of the request. It is to note that if $p = 1$, removal heuristics become the random removal operator. A low value of p corresponds to much randomness. We then select the request i, where

$$i = y^p \cdot \mid L \mid, \tag{4.13}$$

Related Removal (3)

Pisinger and Ropke (2007) modified the removal heuristics from Shaw (1997) and Shaw (1998). The authors then proposed Related Removal, Cluster Removal, and Time-oriented Removal. The concept is to remove requests that are similar to other requests and are expected, therefore, to be able to exchange positions easily and perhaps create better solutions. For the related removal, the relatedness is defined in terms of distance. We present this measure, as stated in Pisinger and Ropke (2007). The relatedness, r_{ij}, of two orders i and j is solely measured by the distance between the requests. Since, each request i consists of a pickup node i and a delivery node, $i+n$, then the relatedness r_{ij} is expressed in terms of

$$r_{ij} = \frac{1}{D}\left(d^{'}(i,\, j) + d^{'}(i,\, j+n) + d^{'}(i+n,\, j) + d^{'}(i+n,\, j+n)\right), \tag{4.14}$$

Where the distance measure $d'(u,v)$ between two nodes in this context is defined as

$$d'(u,v) = \begin{cases} d_{uv} & if\ u\ and\ v\ are\ not\ located\ at\ a\ terminal \\ 0 & if\ u\ or\ v\ is\ located\ at\ a\ terminal \end{cases}$$

Pisinger and Ropke (2007) discussed that the motivation for neglecting the distance from a terminal is that the terminal is going to be visited in any case, and thus should not contribute to the relatedness measure of two requests. The denominator D is set to the number of non-zero term i.e. number of pickups and deliveries taking place at a site different from a terminal. Therefore, for the PDPTW, the denominator D is set to 4. The lower r_{ij} is, the more related are the two requests.

All planned requests are sorted in ascending order. The algorithm initially selects a request i by random. Then, it repeatedly chooses an already selected request j and selects a new request which is most related to j. The algorithm stops when q requests have been chosen. The selection of requests is controlled by a randomisation parameter, p_r, used in Equation 4.13.

Cluster Removal (4)

Pisinger and Ropke (2007) tried to remove clusters of related requests from a few routes, since, with a route grouped into two geographical clusters, it is better to remove one of these clusters. The insertion heuristics would otherwise likely insert the single removed request back into the route. The Kruskal's algorithm for the minimum spanning tree problem (using r_{ij} for the edge distances) is used and terminated when two connected components remain. One of these clusters is chosen at random and the requests from the selected cluster are removed. If less than q requests have been selected, we randomly pick a removed request and choose the most related request from a different route. Then, the route of the new request is partitioned into two clusters and the process

continues until q has been removed.

Time-oriented Removal (5)

Pisinger and Ropke (2007) stated that this heuristic tries to exchange the requests that are expected to exchange easily, namely those that are served at somewhat the same time as these requests. A request $\widetilde{r}$ is selected at random and the B requests that are closest to $\widetilde{r}$ according to 4.15 are marked. Time-related measure is the arrival time of two pickup-and-delivery requests as shown in Equation 4.15.

$$\triangle t_{ij} = \mid t_{p_i} - t_{p_j} \mid + \mid t_{d_i} - t_{d_j} \mid, \tag{4.15}$$

where t_{p_i} and t_{d_i} are the times of the pickup and delivery of request i in the current solution. Among the B marked requests we select the $q-1$ that are nearest to$\widetilde{r}$according to $\triangle t_{ij}$. The request selection process is also controlled by a randomisation parameter, p_t, used in 4.13.

Historical Node-pair Removal (6)

Pisinger and Ropke (2007) stated that, in this heuristic, the historical success of visiting two nodes right after each other in a route is recorded. With each pair of nodes $(u, v) \in A$, a weight $f^*_{(u,v)}$ indicating the best solution value found so far, in a solution which used edge (u, v). Initially, $f^*_{(u,v)}$ is set to infinity. Each time a new solution is found, the weights $f^*_{(u,v)}$ of all edges used in the given solution are updated. $f^*_{(u,v)}$ is used to remove requests that seem to be misplaced. The heuristic sums the weights of edges incident to i and $i+n$. The most costly request is removed. The randomness parameter, p_{np}, is also introduced to ensure some variation.

Historical Request-pair Removal (7)

Pisinger and Ropke (2007) stated that this heuristic uses the historical success of placing pairs of requests in the same routes. For this operator, the weight $h_{(a,b)}$ for each pair of requests $(a, b) \in \{1, \ldots, n\} \times \{1, \ldots, n\}$ is introduced. The weight $h_{(a,b)}$ denotes the number of times the two requests a and b have been served by the same vehicle in the B best unique solutions observed so far. Initially, $h_{(a,b)}$ is set to zero, and each time a new unique top-B solution is observed, the weights are incremented and decremented according to the solutions entering and leaving the top-B solution. The graph is used to define the relatedness between two requests, such that two requests are considered to be related if the weight of the corresponding edge in the request graph is high. This relatedness measure is used as in the related removal heuristic. From experiments, we also set the B value to 100 as in Pisinger and Ropke (2007) and Ribeiro and Laporte (2012). The randomness parameter, p_{rp}, is also used.

4.5.2 Identical Vehicle Crossover (IVX)

In this metaheuristic hybrid, we introduce a crossover operator that transfers good routes from one parent to another. The MD-PDPTW-SR is a highly constrained problem. Nagata and Kobayashi (2010) confirmed that the existence of the pickup and delivery constraint makes the design of an effective crossover operator more difficult. Since, after applying an appropriate repair operation, the constraint violation may be eliminated, however the resulting solutions will no longer inherit meaningful building blocks from the parents. The authors designed a Selective Route Exchange Crossover (SREX) by combining routes from two parents in such a way that the amount of constraint violation from these routes is approximately minimised by a specialised local search. However, we experimented the SREX to the MD-PDPTW-SR and found that the constraints in terms of special requests, capacity and time windows in the MD-

PDPTW-SR creates some difficulties to the SREX due to constraint violations and repair. It is important to note that the violation of time windows possibly occurs due to relocating requests to another farther depot because different routes in different vehicles from different depots can be recombined. Moreover, the specialised local search is rather complex and requires some computational effort. However, the concept of combining routes still remains. Berger et al. (2003), Berger and Barkaoui (2004), Pankratz (2005) and Hosny (2010) applied different ways of recombining routes to form an offspring, but all aim to preserve the orientation of the route. Recall that we aim to design a metaheuristic that is easy to understand while providing flexibility, accuracy and robustness. To achieve this, we simply transfer the routes from the same vehicles in both parents. In this way, we avoid the concern of violating the constraint related to special requests. Then, we term this operator as the Identical Vehicle Crossover (IVX). The IVX can be broken down into 5 steps.

Main steps of the IVX

Step (1): The number of routes within the interval $[r_1, r_2]$ is selected randomly.

Step (2): The "good" routes of the second parent are selected. We determine the good routes similar to Berger et al. (2003), Berger and Barkaoui (2004) and Hosny (2010). The authors rank routes according to the number of nodes in descending order. Ties are broken by the route travelled distance. Moreover, when employing Grefenstette's (1987) use of probabilistic selection for crossover, we found that introducing some randomness can improve the search performance for the IVX. We adapt the randomness parameter from Equation 4.13 and consider L as the ranked list of non-empty routes instead. In order to rank non-empty routes, we introduce two different route selection rules: average distance and average time and distance. These rules are used to measure the route quality. To illustrate, the average distance is the average value of distance separating the consecutive locations in a route. The average distance reflects

the total distance of the route. The average time and distance reflects the total distance and total time, as considered in the objective function. The randomness parameters p_c are also introduced. To probabilistically choose the good routes, we select route i according to Equation 4.13 where $\mid L \mid$ is the number of non-empty routes.

Step (3): The selected routes in the second parent are checked to avoid duplicate offspring. Hereafter, we refer to selected routes or inserting routes interchangeably. The replaced routes refer to the same vehicle in the firstly selected parent. In order to prevent duplication, the inserting routes and replaced routes must not be identical, since we transfer the requests to the same vehicle. If the route is identical, it is not selected. Instead, it is removed from the list for selection, and the selection of routes continues.

Step (4): We remove all requests in the replaced routes to make room for inserting the selected routes from the second parent. That means the requests contained in both the replaced and inserting routes are not removed, while the requests that are in the replaced routes, but not the inserted routes, are removed to the request bank. If some requests appear twice in the solution, we delete them from the routes that originally belonged to the first parent.

Step (5): We reproduce the second offspring in **Step 2-4** with the parents in reversed roles. An illustrative example of the recombination process of the IVX is demonstrated in Figure 4.2.

In Figure 4.2, assume that one route is selected in step 1. Step 2 is to select good route(s), e.g. the route in vehicle 4 of P2. Then, in step 3, the duplication is checked at the same vehicle of P1. In step 4, the request 5 is removed to the request bank because it is neither request 3 nor 1. Then, requests 3 and 1 of the original P1 are deleted. Some requests may, therefore, be removed to the request bank and reinserted back to the solution, which in this situation is called *implicit mutation*.

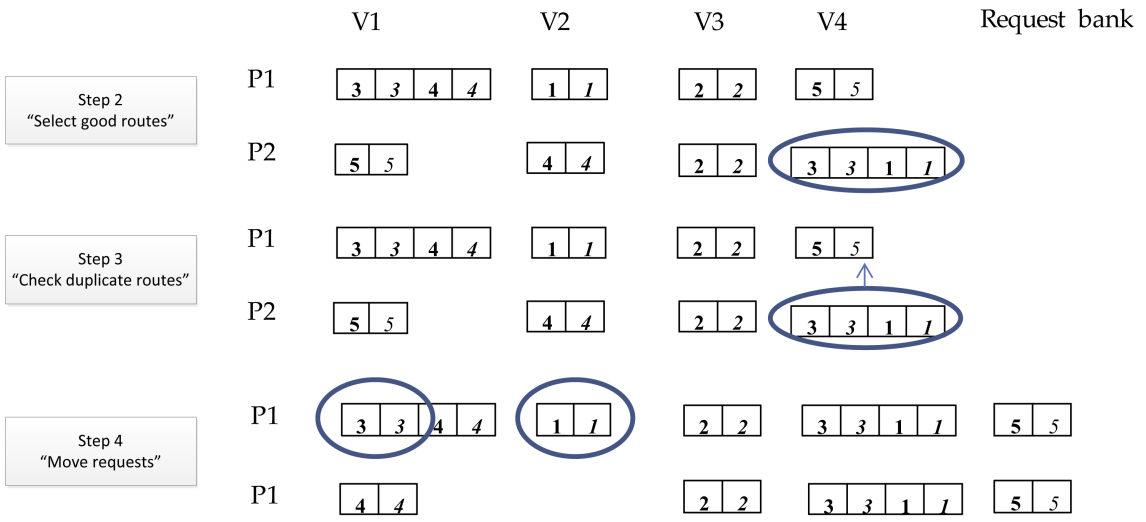


Figure 4.2: The Identical Vehicle Crossover (IVX)

4.5.3 Insertion Operators

Once some requests are removed by the removal operators described in Section 4.5.1, these requests are reinserted by the insertion operator, according to Pisinger and Ropke (2007), as follows:

Given a number of k partial routes in a solution, where $k \in R$ and a number of unassigned requests, U, is in the request bank. The regret-k heuristics are considered for parallel insertion, which construct several routes at the same time. Denote $\triangle f_i^q$ the change in the objective value when inserting request i into its best position in the q^{th} cheapest route for request i. At each iteration, the regret heuristic selects the request i according to

$$i := arg \max_{i \in U} (\sum_{h=2}^{q} \triangle f_i^h - \triangle f_i^1)$$

Ties are broken by selecting the request with lowest insertion cost. Then, the request i is inserted at its minimum cost position, in its best route. The heuristics tries to insert a request on the q best routes and select the requests whose cost difference between inserting it into the best route and the $q-1$ best routes is largest. This type of insertion

heuristics incorporates look-ahead information when selecting the request to insert. Otherwise, the placement of difficult requests in the last iterations or myopic behaviour normally happens in a basic greedy heuristics i.e. when $q = 1$. In our proposed heuristic, we apply regret-1, regret-2, regret-3, regret-4 and regret-m ($m =\mid K \mid$).

4.5.4 Adaptive Mechanism

The adaptive mechanism is used to keep track and adjust the use of each operator according to their historic success. It consists of using a roulette wheel selection and adaptive weight adjustment.

Roulette Wheel Selection

The adaptive mechanism uses the roulette wheel selection for choosing the pre-defined rules in each of the removal heuristics, insertion heuristics, noise methods and IVX's route selection in the AMLNS. The mechanism controls the selection of pre-defined rules, according to their past performance (score). Let π_i be the past score of a pre-defined rule i and ω pre-defined rules in each method. The roulette wheel selects pre-defined rule j with probability:

$$Prob.\ Roulette = \frac{\pi_j}{\sum_{i=1}^{\omega} \pi_i} \tag{4.16}$$

Note that the removal and insertion heuristics are selected independently by a separate roulette wheel, thus, the noise method and IVX's route selection. Moreover, due to being a population-based approach, the AMLNS allocates separate roulette wheels to each solution.

Adaptive Weight Adjustment

According to Pisinger and Ropke (2007), the roulette wheel selection mechanism is based on the scores π_i of pre-defined rules in each method. A successful heuristic gains a high score and, as a result, the heuristics should be selected with a larger probability. In this study, the scores are collected every 100 iterations as in Ropke and Pisinger (2006). The observed score $\bar{\pi}_{i,j}$ of a pre-defined rule i in generation j is increased with the following values depending on the new solution $x^{'}$.

σ_1 : The last remove-insert operation produced by a new global best solution $x^{'}$.

σ_2 : The last remove-insert operation produced a solution $x^{'}$ that its cost is better than the cost of the current solution.

σ_3 : The last remove-insert operation produced a solution $x^{'}$ that its cost is worse than the cost of the current solution, but still acceptable by Threshold.

At the end of each segment, the *smoothened* scores are calculated as follows:

$$\pi_{i,j+1} = \rho\frac{\bar{\pi}_{i,j}}{a_i} + (1-\rho)\pi_{i,j}$$

where a_i is the number of times the heuristics are used in each segment. The reaction factor ρ controls how quickly the weight adjustment reacts to changes in the scores.

Pisinger and Ropke (2007) further observed that a mixture of good and less good heuristics leads to better solutions than solely using good heuristics. It is necessary that well-performing heuristics are given most influence, but still all heuristics participate in the solution process. Therefore, in this study, we set a minimum value of probability of each pre-defined rule (*Prob. Roulette*), in the roulette wheel selection, to 0.1 to remain useful to all operators.

4.5.5 Applying Noise to Objective Function

Noise is applied at the insertion heuristics, which are either with noise or without noise. The objective function corresponds to the weighted sum of distance travelled, time travelled and the number of requests in the request bank. We also set the coefficients of the objective function in Equation 4.1, $\alpha = \beta = 1$ and $\gamma = 100000$ as in Ropke and Pisinger (2006).

An alternative diversification procedure is to apply noise to the objective function. The insertion cost C can be modified with some noise δ. The modified insertion cost $C' = max\{0, C + \delta\}$. The noise is randomly selected as a random number in the interval $[-N_{max}, N_{max}]$, where $N_{max} = \eta \cdot max_{i,j \in V}\{d_{ij}\}$, where η is a parameter that controls the amount of noise. The clean or the noise imposed insertion is selected by the roulette wheel mechanism.

4.5.6 Initialisation

In the AMLNS, the solution structure contains location sequences served by vehicles as described in Section 3.6.2. We avoid the encoding of chromosomes in other formats that cannot apply local search operators and which cause a violation after using operators.

Chu (1997), Hart et al. (2005) and Ho et al. (2008) pointed out that using problem-specific knowledge as the heuristic initialisation is superior to the use of random generation. Grefenstette (1987) suggested that, in contrast to a single solution approach, the population diversity and the quality of the initial solutions are essential for the search performance.

As a result, we generate an initial population that are diverse and good quality. The regret insertion heuristics are the core of the population initialisation. Different seeding criteria can differentiate the individuals from one another, resulting in diverse solutions. We apply two seeding strategies, namely, no seeding and a single-request

seeding. Then, the regret-1, regret-2, regret-3, regret-4 and regret-m construct the solutions by inserting requests from the request bank.

In terms of no seeding, all regret heuristics are used. The initialisation criterion for single-request seeding is to determine the single request for insertion in any vehicle. According to Li and Lim (2001), we select two criteria: minimal combined latest bound of time windows and minimal combined period of time windows because these requests seem to be difficult to insert. For each seeding strategy, all regret heuristics are sequentially applied. It is important to note that these no-seeding and 2 single-request seeding criteria with regret-k insertion can produce up to 15 different feasible solutions, depending on whether or not solutions are duplicated. If there is no duplication at all, the set of solutions seeded from minimal combined latest bound are generally used before those from a minimal combined period of time windows. From the experiment, we found that this method produces a diverse population.

4.5.7 Master Local Search Framework

Ropke and Pisinger (2006) and Pisinger and Ropke (2007) applied the Simulated Annealing as the master local search framework for the ALNS. In this Chapter, we hybridise the ALNS with other metaheuristics. Therefore, the ALNS framework is modified and incorporated with a specially designed mechanism for this hybridisation.

Modified Threshold Accepting

Dueck and Scheuer (1990) simplified the SA procedure by leaving out the stochastic element when accepting worse solutions. Instead, they introduced a deterministic threshold, $Thres$, and always accept a worse solution if the percentage difference to the incumbent solution is smaller or equal to the Threshold.

Pisinger and Ropke (2007) stated that the start temperature control parameter of different problem sizes should be divided by the number of requests in that instance. One advantage of TA is that there is no need to determine the start temperature control parameter that corresponds to the problem size. Dueck and Scheuer (1990) emphasised that the advantages of the TA are its simplicity and efficiency. In our opinion, the exponential probability function of the SA makes the ALNS difficult to be manipulated due to its temperature and stochasticity.

By using the TA, if the objective function of the new solution is less than $(1+Thres)\times$ the objective function of the current solution, the modified solution is accepted. In this definition, $Thres$ is typically a fraction. Due to its determinism, it is rather easy to be manipulated for hybridisation and implemented. We denote the starting Threshold as $St.Thres$. According to the experiments in Chapter 5, we modify the original Threshold Accepting (TA) in terms of threshold reduction by using the exponential cooling rate, c_{exp}, according to Equation 4.17.

$$Thres = Thres \cdot c_{exp} \tag{4.17}$$

Tournament Selection

Selection pressure plays a vital role to ensure increased proportions for good routes. Pankratz (2005) pointed out that the binary tournament selection mechanism has low time complexity. In addition, a simple comparison of the objective values is sufficient. From the literature, we selected the binary tournament selection according to Ombuki and Hanshar (2009), due to its flexibility. To begin with, two individuals, i.e. tournament set, are randomly selected from the population. A random number, r, between 0 and 1 is chosen at random. If r is less than $Prob_T$, the fittest individual in the tournament set is then selected for reproduction. Otherwise, the solution is selected at

random from the two solutions. The typical binary tournament approach is a special case of this tournament approach with tournament probability = 1.0.

In the AMLNS, the tournament selection is different from conventional selection, which selects two or more solutions consecutively. We separate the selection of two solutions by using two binary tournament selection wheels. The two binary tournament selection wheels are applied at the different time and or even possibly different probabilities. The first tournament selection wheel is applied when selecting one solution from the population. Then, the roulette wheel selection determines the removal operator to apply, according to Equation 4.16. If the roulette wheel selects the crossover, the second tournament selection is then called to select another solution in the population for mating and reproduction. The first and second tournament selection probabilities are denoted as $Prob_{T,1^{st}}$ and $Prob_{T,2^{nd}}$ respectively.

Crossover Probability

The selection pressure from route quality measures in crossover and tournament selection reinforces the intensification effect. However, whole route (s) removal and implicit mutation causes a change in the large number of requests and results in diversification. From the experiments in Chapter 5, the significant change in the objective function due to the crossover often exceeds the Threshold and is relatively less accepted than the removal operators of the original ALNS. Without a mechanism to prevent the adaptive weights of the original operators that overtake that of the crossover, the crossover cannot compete.

Therefore, we modify the roulette wheel selection in Section 4.5.4 to control the use of different neighbourhood operators. To simplify the implementation and take advantage of the roulette wheel selection for removal operators, we partition the wheel between the ALNS's original removal operators and the proposed crossover. In the

IVX, two route measures are applied. These are treated as two operators. We refer this partition value to crossover probability, $prob_{cross}$. The $prob_{cross}$ separates the probabilities in the roulette wheel into two intervals e.g. $[0, prob_{cross})$ and $[prob_{cross}, 1)$ for two crossover operators and original ALNS operators, respectively. Even though, all original ALNS operators performed very well, the range of probabilities is still restricted to $[prob_{cross}, 1)$. In each partition, the probabilities of operators are adaptive according to their historic performance.

Replacement Strategy

A replacement strategy plays an important role in preserving the diversity of the population in GAs and MAs. According to the surveyed literature and the discussion described earlier, we redesigned the replacement strategy as similar to that of Nagata and Kobayashi (2010). The similarity is that the parent is replaced by its offspring. Nagata and Kobayashi (2010) generated a number of offspring by crossover and local search operators. Only an improved offspring replaces its parents. In the AMLNS, we generate one offspring at a time. According to the experiments, we observe that accepting only improved offspring may obstruct the traversal of modified solutions to explore different regions of search space. Therefore, in the AMLNS, the parent is replaced by its offspring only if the offspring is accepted by the Threshold. Acceptance of non-improving solutions can diversify the search for the continued innovation of a crossover. However, the best solution thus far for each solution is always kept. The experiments confirmed that this proposed replacement strategy helps maintain population diversity, preventing premature convergence, and allowing the exploration of the search space to some extent.

Cut-off Mechanism

As suggested by Pisinger and Ropke (2007), the number of requests to remove should be reduced when the algorithm rarely accepts non-improving solutions. We then propose a mechanism or so-called cut-off mechanism to schedule the use of smaller moves. For simplicity, we monitor the Threshold whether or not reducing down to a percentage, $Coff$, of the $St.Threshold$. The rationale of using the percentage of the start Threshold is that the acceptance of changes in the objective function depends on the Threshold, as described in Equation 4.17. The lower the Threshold is, the lower the acceptance will be. Then, it may be appropriate to avoid using the IVX and large number of requests to remove for the LNS. Therefore, the percentage of the start threshold is one of the suitable indicators to initially determine the cut-off point for smaller moves.

The cut-off mechanism separates the AMLNS into two stages. Before the cut-off point, the number of solutions, tournament selection, crossover and the original ALNS's operators are used. The second stage of the algorithm is applied after the number of iterations, where the Threshold is reduced down to the value of $Coff \times St.Threshold$. In this stage, the Threshold becomes too small: the crossover operators and LNS with large q are rarely accepted, but the LNS with applying a small number of requests to remove is often accepted. Therefore, we avoid using the crossover, and only use LNS with a smaller number of requests to remove. As we assume that a good basin of attraction is located by the best current solution among the population after the first stage, this solution is individually selected to intensify the search. It is important to note that the selected solution for the second stage, which is selected from the solutions gained at the $Coff \times St.\,Threshold$ iteration, is not necessarily the best solution found thus far. In this stage, the tournament selection is not used anymore. We also set $prob_{cross}$ to zero. Recall that the range of crossovers and original ALNS operator are $[0, prob_{cross})$ and $[prob_{cross}, 1)$ respectively. Therefore, the crossover operators cannot be used but only original ALNS operators. The new solution is only accepted according

to the Threshold. The number of requests to remove is then reduced to $\theta\%$ for both of their upper and lower bounds.

By determining the two stages of the AMLNS, before and after the cut-off point, we then obtain the aggregate diversification (D>I) and intensification phases (I>D), respectively. In other words, the first stage aims at diversifying the search by using population, crossover, LNS operators and large Threshold. The second stage aims at intensifying the search by using smaller *Threshold* and a LNS with a small number of requests to remove, but still applying multiple local search operators, each with its own search direction. The use of multiple local search operators and randomised parameters can still help to diversify the search while other diversification mechanisms such as the population, crossover, and LNS with large q are omitted. Roulette wheel selection gives larger intensification force than diversification force.

4.5.8 Reduction Rules for Improving Computational time

One of the most time-consuming parts in the AMLNS is the regret-k heuristics. In order to find the minimum cost position of one route, the objective function of all possible insertion must be calculated. A known time complexity of $O(n^2)$, when all possible insertion of a request in a route containing n nodes, is quantified. Due to being a highly constrained problem, the MD-PDPTW-SR restricts the feasibly inserted locations according to several constraints. If a reduction rule is found, this can help reduce computational time without leaving out any feasible solution. Two examples are the reduction rules for precedence and capacity constraints, as described in Chapter 3. The MD-PDPTW-SR can be decomposed into several sub-problems such as time windows and special requests.

Time Windows

In this problem, the violation of time windows in each customer location is not allowed. Jaw et al. (1986), Solomon (1987), and Diana and Dessouky (2004) applied a similar concept of time feasibility checking. The authors took advantages of slack, idle or waiting time in order to seek feasibly inserted locations. According to Solomon (1987), the service at a customer, say i, $i = 1, \ldots, n$, involving pickup and/or delivery of goods or services for s_i units of time, can begin at time b_i, within a time window defined by the earliest time e_i and the latest time l_i that customer i will permit the start of service. Hencem if a vehicle travels directly from customer i to customer j and arrive too early at j, it will wait, that is, $b_j = max\{e_j, b_i + s_i + t_{ij}\}$, where t_{ij} is the direct travel time between i and j. Solomon (1987) examined the necessary and sufficient conditions for time feasibility when inserting a customer, say u, between the customers i_{p-1}, and i_p, $1 \leq p \leq m$, on partially constructed feasible route, $(i_0, i_1, i_2, \ldots, i_m)$, $i_0 = i_m = 0$, for which the times to begin service, b_{i_r}, for $0 \leq r \leq m$, are known. It is assumed that initially each vehicle leaves the depot at the earliest possible time, e_0. After the complete vehicle schedules have been created, we can adjust the depot departure time separately for each vehicle to eliminate any unnecessary waiting time.

Denote by $b_{i_p}^{new}$ the new time when service at customer i_p, begins, given the insertion of customer u. Also, let w_{i_r}be the waiting time at i_r for $p \leq r \leq m$. If we assume that the triangle inequality holds both for travel distances and times, this insertion defines a push forward in the schedule at i_p :

$$PF_{i_p} = b_{i_p}^{new} - b_{i_p} \geq 0$$

Furthermore,

$$PF_{i_{r+1}} = max\{0, PF_{i_r} - w_{i_{r+1}}\},\ p \leq r \leq m-1$$

If $PF_{i_p} > 0$, some of the customers i_r, $p \leq r \leq m$, could become infeasible. It should then examine these customers sequentially for time feasibility until we find some customer, say i_r with $r < m$, for which $PF_{i_r} = 0$, or i_r is time infeasible, or, in the worst case all the customers i_r, $p \leq r \leq m$ are examined.

Solomon (1987) proved that:

Lemma 1 *The necessary and sufficient conditions for time feasibility when inserting a customer, say u, between i_{p-1} and i_p, $1 \leq p \leq m$, on a partially constructed feasible route $(i_0, i_1, i_2, \ldots, i_m)$, $i_0 = i_m = 0$ are*

$$b_u \leq l_u, \tag{4.18}$$

$$b_{i_r} + PF_{i_r} \leq l_{i_r},\ p \leq r \leq m \tag{4.19}$$

The Fix-forward Insertion using Reduction Rule for Time Windows is described in Appendix A. We also validated the TW reduction rule by comparing with the explicit testing of time feasibility at each customer location. The feasibly inserted locations are equivalent, yet a substitution can significantly reduce computational time.

Special Requests

In the multi-depot problems, customers may be geographically dispersed. Due to the latest time windows of the vehicles, some customers can not even be served by some vehicles, even, for the first request. These customer requests correspond to the vehicles and can be recorded to incorporate with the special request lists and constraints.

Calculation of the Incremental Distance

The change in distance can occur due to either a removal or insertion operator. When calculating the distance in each route, we only calculated changed distances for changed edges. We only calculated the changed edges because of the application of the removal and insertion operators. For the insertion operator, recall that the insertion of a customer, say u between i_{p-1} and i_p, $1 \leq p \leq m$, on a partially constructed feasible route, $(i_0, i_1, i_2, \ldots, i_m)$, $i_0 = i_m = 0$. Let d_{i_{p-1},i_p} be the travelling distance from i_{p-1} to i_p. Denote $d_{i_{p-1},u}$ the distance from i_{p-1} to u and d_{u,i_p} the distance from u to i_p. The incremental distance, $\triangle_{in}$, is $d_{i_{p-1},u} + d_{u,i_p} - d_{i_{p-1},i_p}$. The insertion of both pickup and delivery location must be calculated. When the consecutive node from pickup node is its corresponding delivery node, the overlapped edges will cancel out each other. Therefore, we can apply the same calculation.

Calculation of the Incremental Time

The time calculation is different from the distance calculation due to waiting time. The incremental time is the push forward time toward the end depot. The push forward is mentioned in Appendix A. The reduction rule for both distance and time can speed up the computational time of the insertion heuristics. The objective function of the new route is the objective of the previous route with an incremental change.

Calculation of the Objective Function

We mark the change of each route after removal and insertion. For this problem, the sum of the distance and time of each route refers to the sub-objective function values. The sub-objective function values of only changed routes are recalculated. Other routes'

sub-objective function values remain the same. This rule can considerably improve the speed of calculating the objective function, especially in large-sized problems since only a few routes are changed.

4.6 Computational Experiments

Pisinger and Ropke (2007) coded the MD-PDPTW-SR and ran on an AMD Opteron 250 (2.4 GHz). The algorithm measures the solution cost using double precision floating point number. The objective function value is then rounded to two decimals. The AMLNS was run on a single-thread of Intel Core I7 (3.5 GHz). It is important to note that the computer languages used for our heuristic and that of Pisinger and Ropke (2007) heuristic are different: while they used C++, we coded our heuristic using the high-level computer language, C# in Visual Studio 2010.

4.6.1 Small-sized Test Instances

We derived the small test instances containing the partial requests from the benchmark test instances of Ropke and Pisinger (2006) for validating the AMLNS by CPLEX. In Ropke and Pisinger (2006), there are 12 types of test instances available with the problem sizes varied from 50 to 500 requests (100 to 1000 locations). The 12 small-sized problems are also characterised by route type, request type and geographical distribution. The Mixed Integer Linear Programming presented in Section 4.3, is coded into CPLEX and optimally solved the problems of up to 18 requests or 36 locations, within a two-day limit imposed. Using CPLEX, we found that the number of requests significantly affects the computational time. In addition, the computational time is varied according to different problem types. From the experiments, while the CPLEX

optimally solved the problem size of 18 requests in a few hours, the AMLNS can solve to optimality in seconds.

4.6.2 Medium-sized and Large-sized Test Instances

To the best of our knowledge, we only found one set of standard benchmark test instances for medium-sized and large-sized problems of Ropke and Pisinger (2006). The description of the test instances is illustrated in Section 4.4.3 of Ropke and Pisinger (2006). The benchmark test instances and their computational results for MD-PDPTW-SR are available from www.diku.dk/~sropke, and updated by the ALNS proposed by Pisinger and Ropke (2007). The computational results were tested on 10 runs. There are 48 test instances varied by route type, request type, geographical distribution and the number of requests (50, 100, 250, 500). Each request contains a pickup and corresponding delivery location. Therefore, the problems size of 500 requests contains 1000 customer locations. The three problems' features, according to Ropke and Pisinger (2006), are shown below:

- Route type: (1) A route starts and ends at the same location, (2) a route starts and ends at different locations.
- Request type: (1) All requests are normal requests, (2) 50% of the requests are special requests. The special requests can only be served by a subset of the vehicles. In the test problems each special request could only be served by between 30% to 60% of the vehicles.
- Geographical distributions: (1) uniform, (2) clustered, and (3) semi-clustered.

Table 4.1 shows the problem types (A-L) arising from the combination of geographical

distribution, route type and request type. In term of abbreviations, Same dep. and Diff. dep. refer to the problem that a route starts and ends at the same location, and a route starts and ends at different locations, respectively. Norm. req. and Spec. req. refer to normal requests and special requests, respectively. In terms of geographical distribution, U., C, and SC. stand for uniform, clustered, and semi-clustered respectively.

	Route type		Request type		Geographical distributions		
Type	Same dep.	Diff. dep	Norm. req.	Spec. req.	U.	C.	SC.
A	√		√		√		
B	√			√	√		
C		√	√		√		
D		√		√	√		
E	√		√			√	
F	√			√		√	
G		√	√			√	
H		√		√		√	
I	√		√				√
J	√			√			√
K		√	√				√
L		√		√			√

Table 4.1: The features of the benchmark test instances used in Ropke and Pisinger (2006)

Tuning Instances

Since, design changes and parameter tuning require numerous experiments, only some instances are experimented, so called tuning instances. The tuning instances are some of instances whose characteristics and sizes represent the benchmark instances targeted. In this study, the set of representative tuning instances contains twelve instances in 50, 100, 250, and 500 requests. Each instance is applied five times and its average values were recorded. The tuning instances, according to Table 4.2, are used to represent all problem types in different problem sizes. These tuning test instances are used in both tuning parameters and developing the AMLNS in Chapter 4 and 5.

	Problem size			
Geo. Distribution	50	100	250	500
Uniform	A	B	C	D
Clustered	E	F	G	H
Semi-Clustered	I	J	K	L
Route type	Same dep.	Same dep.	Diff dep.	Diff dep.
Request type	Norm req.	Spec req.	Norm req.	Spec req.

Table 4.2: Tuning Instances for Problem Type in Each Problem Size

We abbreviate Geographical Distribution, depots, and special requests as Geo. Distribution, dep., and Spec req. respectively. According to Table 4.2, the following problem sizes and types are used as the tuning instances: 50A, 50E, 50I, 100B, 100F, 100J, 250C, 250G, 250K, 500D, 500H and 500L.

Parameter Tuning

In order to keep parameter tuning to a minimum, we adopted some original parameters as empirically set in Ropke and Pisinger (2006) and Pisinger and Ropke (2007). The setting of some parameters is also obtained from the literature. For example, Merz and Freisleben (1999) showed that a population size of 10 up to 40 is common in MAs because the local search in MAs is time-consuming. In addition, we experimented with the randomness parameters of some removal operators that are not described in Ropke and Pisinger (2006) and Pisinger and Ropke (2007). Moreover, some parameters were obtained from the development of the AMLNS in Chapter 5.

Design and Tuning Process

Gendreau and Tarantilis (2010) stated that, in effectiveness and efficiency analysis, the solution quality and computational time can be viewed as the performance measures for a multi-objective optimisation. To tackle multi-objective optimisation, one can apply the weighted sum of these performance measures to perform comparative analysis

of objective values. Ropke and Pisinger (2006), Pisinger and Ropke (2007), and Vidal et al. (2013) compared the performance of their algorithms to other state-of-the-art heuristics by best solution, average solution, and average computational time of 10 runs for each instance. We distinguish the measure of solution quality into preciseness and reliability that are reflected by deviation from best known solution and average solution respectively. In this study, the improved computational time is measured by the deviation of the average time obtained from the first design, the mimicked algorithm, or the benchmark algorithm. The objective of the problem is to minimise a weighted sum, ($Obj\ fn$), consisting of the following three components: (1) the percentage deviation of average of average solutions using the new design/setting, (2) percentage deviation of average of best solutions using the new design/setting, and (3) percentage deviation of the average time using the new design/setting, from the mimicked benchmark algorithm. The three terms are weighted by the coefficients $\phi, \upsilon, \varsigma$, respectively. To illustrate, the weighted sum, $Obj\ fn$, is defined as:

$$Obj\ fn = \phi \cdot Gap_{av/BA}(\%) + \upsilon \cdot Gap_{b/BA}(\%) + \varsigma \cdot Avg.\ Time(\%) \tag{4.20}$$

where

- $Gap_{av/BA}(\%)$: Percentage deviation of average of average solutions obtained by the AMLNS compared to that of the benchmark algorithm
- $Gap_{b/BA}(\%)$: Percentage deviation of average of best solutions obtained by the AMLNS compared to that of the benchmark algorithm
- $Avg.\ Time(\%)$: Percentage deviation of the average time from the first design or the benchmark algorithm
- $\phi, \upsilon, \varsigma$: Weights of $Gap_{av/BA}(\%)$, $Gap_{b/BA}(\%)$, $Avg.\ Time(\%)$ respectively

In this study, the benchmark algorithm is the ALNS proposed by Ropke and Pisinger 2006. However, the computational time is difficult to be compared, due to using different computing environments. In the development stage, we used the same computer throughout the experiments in order to investigate significant changes at their computational time in the same computing environment. Therefore, the $Avg.Time(\%)$ measures percentage deviation of the average time from the first design, providing a comparative value to the later design .

When designing the AMLNS, we systematically developed the AMLNS by three stages: design changes, parameter scanning and parameter fine-tuning. The design changes and parameter scanning, as shown in Chapter 5, are experimented prior to parameter fine-tuning. We observe that design changes and parameter scanning can influence both solution quality and computational time. However, parameter fine-tuning mainly focuses on the solution quality. Design changes, parameter scanning, and parameter tuning are carried out by allowing one parameter to change at a time, while the rest of the parameters are kept fixed. Then, we select the setting resulting the most improved $Obj\,fn$ (the largest negative value). As we deal with the minimisation problem, the negative values of $Gap_{av/BA}(\%)$ and $Gap_{b/BA}(\%)$ demonstrate the improvement over the benchmark algorithm. Also, the negative value of $Avg.Time(\%)$ means the improved computational time. After selection of the best parameter setting at one experiment, we move on to the next parameter and run the AMLNS on tuning instances again. The development of the tuning principle is similar to the method of parameter tuning in Ropke and Pisinger 2006. It is noted that some (new) best known solutions were obtained and also recorded during the experiments. To avoid the reading interruption due to the extensive development of the AMLNS, which will be shown in Chapter 5, we next demonstrate the parameter fine-tuning.

Parameter Fine-tuning

Due to focusing on solution quality, in the parameter-fine tuning, we set $\phi, \upsilon = 1$ and $\varsigma = 0$. In this experiment, we attempt to fine-tune the parameters that are significant or novel due to hybridisation among MAs, ALNS and TAs.

Table 4.3 shows the weighted sum, $Obj\,fn$, of the interaction between $StThres$ and c_{exp}.

$c_{exp} \backslash StThres$	0.0175	0.015	0.0125	0.01	0.0075	0.005	0.0025
0.99990	2.27	1.44	0.75	0.19	0.20	**-0.19**	0.58
0.99985	0.05	**-0.08**	**-0.01**	**-0.1**	**-0.21**	**-0.22**	0.64
0.99980	**-0.12**	**-0.12**	**-0.03**	**-0.3**	0.11	0.27	1.32
0.99975	0.04	0.01	0.17	0.42	0.74	0.28	1.41

Table 4.3: Experiments between Cooling Rate and Start Threshold

From the experiments, according to Figure 4.3, we observed that the higher $StThres$ and c_{exp} give the diversification effect. Since, the AMLNS allows the long period of searching non-improving solutions. Inversely, the lower $StThres$ and c_{exp} provides the intensification effect. Ideally, we must balance between diversification and intensification. The bold numbers in Table 4.3 potentially reflect the appropriate range of balancing $StThres$ and c_{exp}. According to Table 4.3, the $StThres = 0.01$ and $c_{exp} = 0.99980$ are selected for further experiments due to producing the largest negative value.

Table 4.4 shows the $Obj\,fn$ of the interaction between first and second binary tournament selection probability.

$Prob_{T,1^{st}} \backslash Prob_{T,2^{nd}}$	0.6	0.7	0.8	0.9	1.0
0.5	0.01	0.22	0.07	0.19	0.34
0.6	0.06	0.42	0.05	-0.33	0.08
0.7	0.03	-0.09	**-0.33**	-0.01	0.01
0.8	-0.18	0.15	-0.3	0.05	-0.25
0.9	0.05	0.21	0.02	-0.33	0.0

Table 4.4: Experiments between First and Second Tournament Selection Probability

From the experiments, we observed that the higher $Prob_{T,1^{st}}$ and $Prob_{T,2^{nd}}$ give the intensification effect. Since, the tournament selection of the AMLNS gives the pressure to exploit relatively good solutions. Among the equivalent value of -0.33, we selected

0.7 and 0.8 as they resulted in larger negative $Gap_{av}(\%)$ than that of 0.6 and 0.9.

Table 4.5 show the weight sum, $Obj\ fn$, of the interaction between probability and randomness of crossover.

$prob_{cross} \backslash p_c$	3	6	9	12	15
0.2	0.02	0.06	0.00	0.18	-0.09
0.3	0.11	-0.12	0.00	0.57	-0.16
0.4	0.13	-0.01	-0.04	**-0.45**	0.03
0.5	0.05	-0.33	-0.06	0.02	0.19
0.6	0.22	0.31	0.04	0.10	0.05
0.7	0.10	-0.04	0.25	0.19	-0.21

Table 4.5: Experiments between Probability and Randomness of Crossover

According to the Equation 4.13, the higher randomness parameter or p_c for the IVX gives the intensification effect. Also, the higher $prob_{cross}$ value results in a faster convergence or giving intensification effect. $prob_{cross}$= 0.4 and p_c = 12 produced the best result in Table 4.5. From Table 4.3 to 4.5, the values of $Obj\ fn$ are improved from -0.3 to -0.45.

Parameter Setting

Table 4.6 shows the complete set of parameters used for the AMLNS. In Table 4.6, it is to note that $q_{l,low}$ is relatively lower than that of the original ALNS of Pisinger and Ropke (2007). Since, we already equipped the AMLNS with the diversification features by using the number of solutions and crossover, we must then require the large neighbourhood search operators to sometimes act as the local search operators, similar to the concept of MAs.

4.6.3 Analysis of Typical Search

Ideally, in GAs or MAs, the population diversity should be maintained in order that the crossover avoids recombining two identical solutions and continues generating new

Parameters	Setting
p_w,p_r, p_t, p_{np}, p_{rp}	3,6,6,6,6
η	0.025
ρ	0.1
$\sigma_1, \sigma_2, \sigma_3$	33,9,13
$Coff$	30%
Population size	10
$[r_1, r_2]$	$[1, 2]$
$[q_{m,low}, q_{m,up}]$	$[0.1n, 0.4n]$
$[q_{l,low}, q_{l,up}]$	$[5, 40]$
θ	75
$St.Thres$	0.0100f
c_{exp}	0.9998f
$Prob_{T,1^{st}}$	0.7
$Prob_{T,2^{nd}}$	0.8
$Prob_{cross}$	0.4
p_c	12
# of iterations	25,000

Table 4.6: Parameters and settings used throughout the development

solutions. Zhu and Liu (2004) measured the hamming based population diversity of population for TSP, VRP, VRPTW.

Let s be an integer sequence that represent a genotype. The authors denote $A(s)$ to be a set of arcs in s. The edge distance between genotype u and v is defined as:

$$D_e(u, v) =\mid A(u) \setminus A(v) \mid$$

In other words, edge distance is defined as the number of arcs in u but not in v, which is equivalent to that in v but not in u. If $D_e(u, v) = 0$, it means that both solutions are identical. The hamming based population diversity is measured as:

$$gtype(P) = \frac{\sum_{i \neq j} D_e(P[i], P[j])}{(K-1)(N-1)N} \tag{4.21}$$

where $P[i]$ and $P[j]$ are i^{th} and j^{th} genotypes in P and K is the number of customer locations. N is the population size. $(N-1)N$ is the number of possible comparisons excluding itself. To illustrate, Zhu (2003) encoded the representation that depots are

```
Route 1: A -> 3 -> 2 -> 4 -> 5 -> A'
Route 2: B -> 10 -> 6 -> 1 -> 12 -> 11
         -> B'
Route 3: C -> 9 -> 8 -> 7 -> C'
```

Figure 4.4: Route structures for MD-PDPTW-SR

not coded in as delimiters, so that ordinary crossover operations can be used. Figure 4.3 shows the encoding of one chromosome's VRPTW according to Zhu (2003).

```
Route 1: 0 -> 3 -> 2 -> 4 -> 5 -> 0
Route 2: 0 -> 10 -> 6 -> 1 -> 12 -> 11
         -> 0
Route 3: 0 -> 9 -> 8 -> 7 -> 0
```

can be encoded into:

$$3-2-4-5-9-8-7-10-6-1-12-11 \quad (11)$$

Figure 4.3: Zhu (2003)'s encoding for VRPTW (Zhu 2003,p.3)

In Figure 4.3, given K= 12, the number of edges is then equal to 11 in the encoding. However, in the multi-depot PDPTW, the edges from start and end terminal can differentiate two solutions and we also apply natural route representation as shown in Figure 4.4. Figure 4.4 is illustrated to gain insight into the slight modification of population diversity from the original measure.

In Figure 4.4, we applied the same example as described in Figure 4.3 to compare the difference due to problem domains and chromosome representations. The customer location is equal to 12 or 6 requests. The number of routes or vehicles is equal to 3. To illustrate, in vehicle 1, A and A' represent the start and end location since a start and end location can be different. It is then coded in as delimiters. As can be seen from Figure 4.4, the number of edges is equal to $K + \#\,of\,Veh$, or 15, where $\#\,of\,Veh$ is the number of vehicles. Therefore, for the MD-PDPTW-SR in this Chapter, the diversity is measured by the sum of the edges' distance between any two genotypes or solution structures as follows:

$$gtype(P) = \frac{\sum_{i \neq j} D_e(P[i], P[j])}{(K + \#\,of\,Veh)(N-1)N} \tag{4.22}$$

We demonstrated the search trajectories and diversity of population, solved by the AMLNS for Problem 50A, obtaining a new best known solution in Figure 4.5. This new best known solution of problem 50A has the objective value of 62833.33, while www.diku.dk/~sropke reported the best known solution with its objective value of 63414.76.

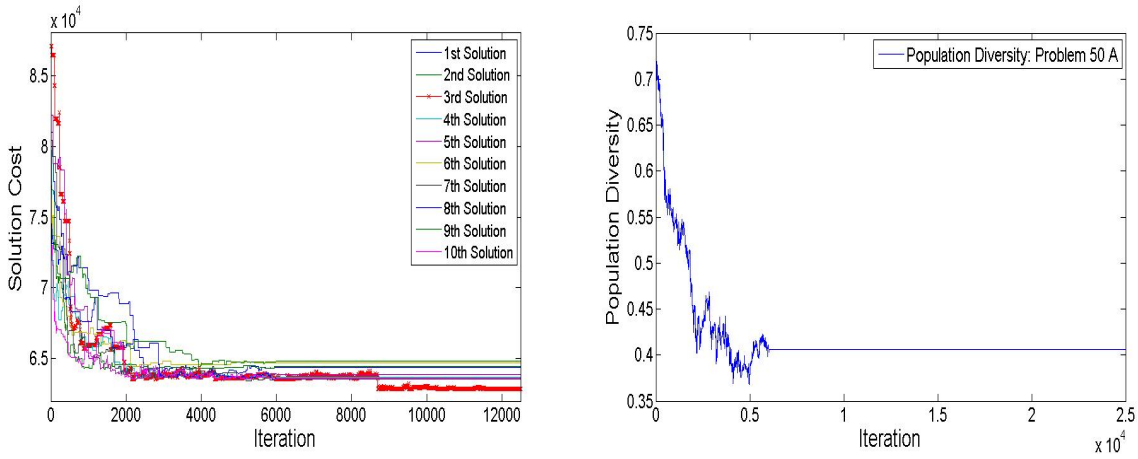


Figure 4.5: Search Trajectories (left) and Population Diversity (right) of Solutions by the AMLNS for Problem 50A

In Figure 4.5 (left), we only showed the search for 12,300 out of 25,000 iterations to enlarge the detail of the search trajectories. To begin with, the population is initialised by different greedy heuristics, providing good diverse solutions. It is interesting to observe that the solution selected for further search (red line) does not need to be the best solution from the beginning. Figure 4.5 (right) shows that the initial population is diverse. After the use of tournament selection and crossover for a number of iterations, the population diversity is dramatically reduced from above 0.7 to below 0.4, while the solution quality also improves rapidly in 5000 iterations. After applying the cooling rate to the Threshold for a number of iterations, the Threshold becomes too small for the large change in objective function due to crossover. As a result, it is sensible to ignore

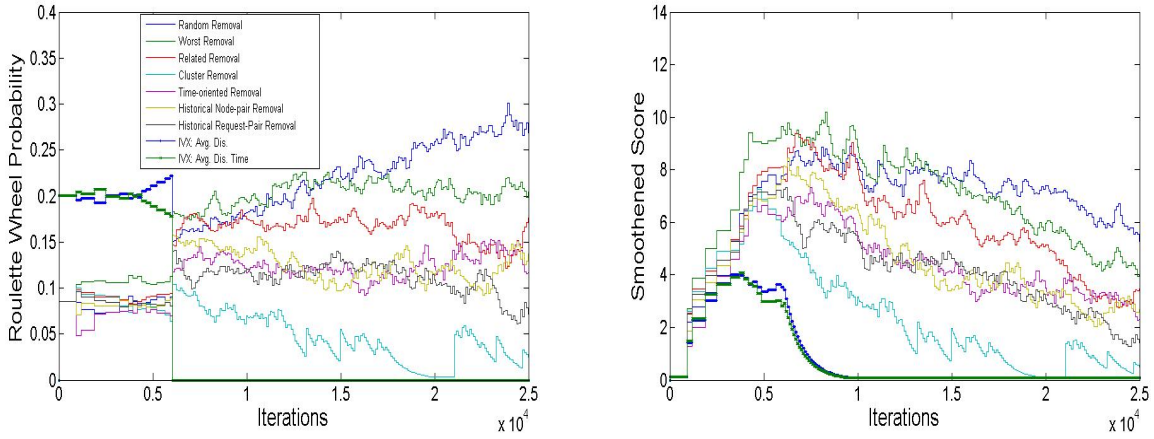


Figure 4.6: Roulette Wheel Probability (left) and Smoothened Score (right) of the Selected Solution using AMLNS for Problem 50A

the use of crossover after the pre-determined cut-off point. The fluctuation of population diversity before cut-off point mainly results from the use of large neighbourhood search operators by choosing a large number of requests to remove, considered as heavy mutation, and the implicit mutation caused by the insertion of route(s). It is important to note that, in the AMLNS using sequential search (opposed to parallel search), the population diversity is not necessarily measured after the cut-off point because only one solution is modified.

Figure 4.6 (left) and (right) show the adaptive weights and probabilities of the separated roulette wheel for the solution that the search continues, respectively. Each coloured line represents each removal operator.

In Figure 4.6 (left), the probabilities of the IVX with average distance, and IVX with average distance and time are higher than all of the large neighbourhood operators due to the partition mechanism, $prob_{cross} = 0.4$. That means each IVX operator takes the initial probability of 0.2.

Figure 4.6 (right) shows the adaptive weights from the corresponding probabilities in Figure 4.6 (left). As the adaptive weights reflect the success of operators, Figure 4.6 (right) shows that the crossover with two measures cannot compete with the large

neighbourhood search operators because significant changes of objective function occur from transferring a large number of requests in route(s) and from implicit mutation. Each IVX differentiated by route measure has an initial probability of 0.2, and each original ALNS operator has the initial probability of 0.086. This should not be interpreted as IVX is unsuccessful and unnecessary because the IVX can improve solutions and diversify the search, as seen from the increased adaptive weights in Figure 4.6 (left), to some extent and be viewed as a removal operator. Moreover, the IVX can recombine different good routes from different good solutions, an ability that the original ALNS operators cannot take advantage from several good solutions in the search space. Among the ALNS operators, the worst removal and the random removal operators are relatively successful.

We observed that the search trajectories and diversity of population are different for the large-sized problems due to the size of its search space. We then showed the search trajectories and diversity of population by using the AMLNS for problem size 500A in Figure 4.7.

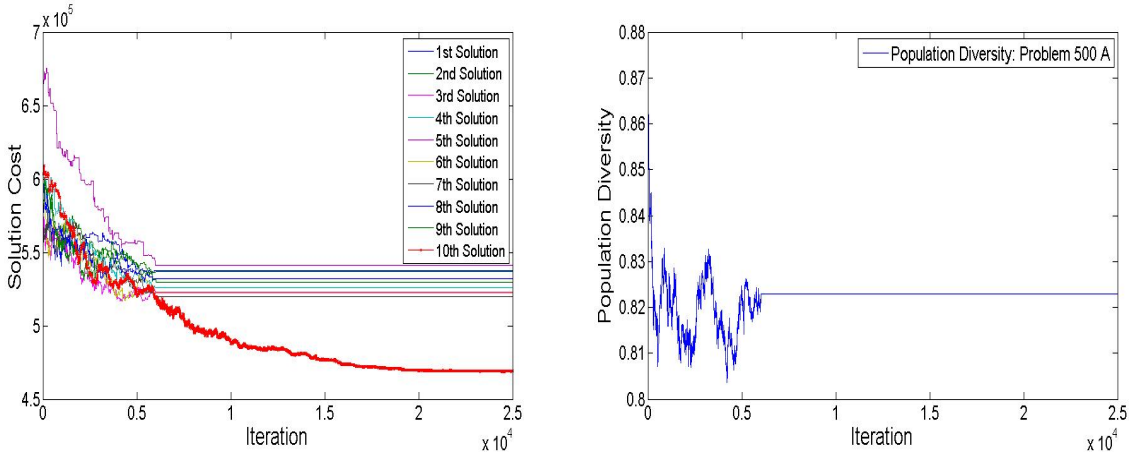


Figure 4.7: Search Trajectories (left) and Population Diversity (right) of Solutions by AMLNS for Problem 500A

Figure 4.7 (left) shows that the selected solution for further search (red bold line) is also not necessarily the best solution, from all solutions, among the population since

the beginning. However, during this process, the objective value of each solution reduced dramatically. We believe that this is the effect and pressure of using tournament selection, heuristic crossover, and LNS with small q. In Figure 4.7 (right), the population is very diverse(at 0.86) due to the use of different construction heuristics and resulting in the different solution structures. Then, the population diversity sharply reduces and fluctuates until reaching the cut-off point. Due to its problem size and replacement strategy that well maintains the diversity, the population diversity rarely converges. It is obvious that the trend of population diversity is not continuously reduced as in Figure 4.5 (right) due to using tournament selection and crossover. However, the tournament selection and the IVX can considerably improve the search. Then, the Threshold rarely accepts a large change in the objective function when reaching the cut-off point. Therefore, it makes sense to ignore the use of heuristic crossover after passing the cut-off point. We believe that the fluctuation is caused by the effects of large neighbourhood search and implicit mutation.

After the cut-off point, the purpose of this stage is to intensify the search. In Figure 4.7 (left), after cut-off point, the cost of a single solution is considerably reduced by the ALNS. The fluctuation of solution costs is also caused by the Threshold to allow the exploration of the search space to some extent.

To illustrate the search behaviour of the AMLNS, we illustrate the hamming distance between accepted solution and the currently best known solution of the AMLNS in Figure 4.7.

In Figure 4.7, we set the $N = 2$ in Equation 4.22, one is accepted solution and the other is the currently best known solution. That means, the hamming distance is normalised by the maximum number of edges from two solutions, $2(K + \#\,of\,Veh)$ in Equation 4.22. The cut-off point of the AMLNS is at approximately 6,000 iterations. From the beginning of the search toward the cut-off points, the population of solutions are recombined, mutated and searched by different LNS operators. Figure 4.7 shows the normalised hamming distances of the accepted solutions to the currently

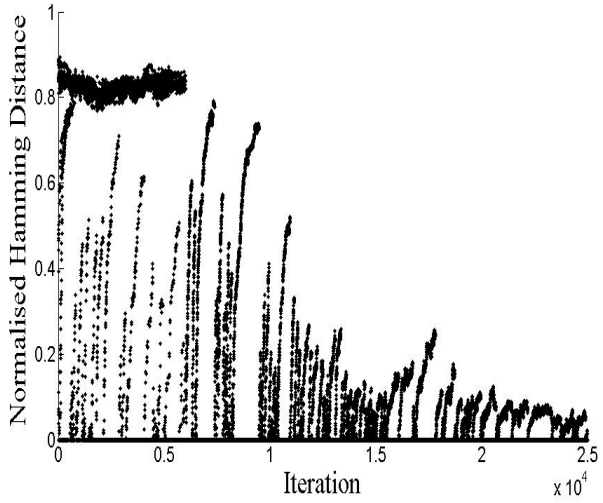


Table 4.7: Hamming distance between accepted solutions and the currently best known solution, using the AMLNS for Problem 500A .

best known solution are highly concentrated between 0.78 and 0.9 approximately. This observation shows that the use of population enables the AMLNS to frequently explore other regions of search space. Moreover, the AMLNS quickly searches far away from the currently best known solution as a result of exploring a number of diverse solutions, while tournament selection tries to choose good solutions for crossover or LNS as it is expected to reproduce a new and better solution. We believe that this mechanism of the AMLNS can enhance the search capability on the ALNS.

4.6.4 Computational Results

The standard benchmark instances of the multi-depot PDPTW in Ropke and Pisinger (2006) is used to test the AMLNS. The ALNS used in Pisinger and Ropke (2007) reproduced the updated computational result in www.diku.dk/~sropke. The table shows the average and best solution found in 10 runs and its average time. In order to evaluate the performance of the AMLNS, we applied these measures accordingly. In Table 4.8 and Table 4.9, the following information is shown for each problem:

- z : Currently best known solutions obtained either from www.diku.dk/~sropke or

the AMLNS presented in this Chapter over all experiments

- z_{av}, z_b : Values of the average and best solutions in 10 runs, respectively
- $Gap_{av}(\%)$, $Gap_b(\%)$: Percentage deviation of the

average and best solution found from current best known solutions, computed as $100\times(z_{av}-z)/z$ and $100\times(z_b-z)/z$, respectively

- Avg. time (s): the average time (in seconds) of 10 runs
- Ref : RP refers to the computational results reported in Ropke (2009a), based on the research in Ropke and Pisinger (2006), and CH refers to the AMLNS developed in this Chapter.

Table B.1 of Appendix B shows the scaling factor that converts the approximate computational time of the AMLNS used in this Chapter relative to the computational time of ALNS reported in Ropke (2009a), due to different computing environments. The scaling factor of 1.44 is multiplied by the computational time of the AMLNS for each instance. However, it is also known that C# programs, used to code the AMLNS in this Chapter, are usually slower than C++ programs. According to Gutin and Karapetyan (2008), these instructions are 1.1 to 4 times slower in C# than in C++. Other factors such as efficiency of data structure and coding affect the computational performance of the algorithm. Nevertheless, these issues are not taken into account in the scaling factor.

One may view AMLNS as an extension of the ALNS that a number of solutions, tournament selection operators, and adaptive crossover operators have incorporated at an early stage. We believe that these operators have low time complexity relative to some removal operators used in Pisinger and Ropke (2007). The IVX only requires the sorting of the number of routes and duplication checking. Initialising a number of solutions as population is expected to only increase a fraction of computational time, but it is essential to the exploration of the search space and is considered as the memory of the solutions. Moreover, the smaller number of requests to remove can reduce computational time, as expected in Pisinger and Ropke (2007). With optimised

Problem	Best Known Cost		ALNS	25K				AMLNS	25K			
			Avg.	Best.	Avg.	Avg.	Avg.	Avg.	Best.	Avg.	Avg.	Avg.
			Sol.	Sol.	gap	gap	time	Sol.	Sol.	gap.	gap	time
Notation	z	Ref	z_{av}	z_b	Gap_{av}	Gap_b	s	z_{av}	z_b	Gap_{av}	Gap_b	s
50-A	**62833.33**	CH	63551.30	63414.76	1.14	0.93	6	63401.72	62833.33	0.90	0.00	15
50-B	**70781.86**	CH	72568.82	71294.23	2.52	0.72	5	71762.89	70781.86	1.39	0.00	12
50-C	**63584.60**	RP/CH	64323.99	63584.60	1.16	0.00	6	64083.38	63584.60	0.78	0.00	15
50-D	**74374.67**	CH	74936.80	74484.31	0.76	0.15	5	74649.64	74484.31	0.37	0.15	12
50-E	**49807.59**	CH	51332.50	50266.03	3.06	0.92	6	51270.40	50553.77	2.94	1.50	16
50-F	**51840.50**	CH	52414.18	51870.79	1.11	0.06	6	53276.99	51873.39	2.77	0.06	14
50-G	**51821.42**	CH	52243.21	51827.63	0.81	0.01	6	52224.96	51973.08	0.78	0.29	16
50-H	**56761.36**	CH	57776.54	56865.13	1.79	0.18	6	57933.15	57067.76	2.06	0.54	15
50-I	**54531.95**	RP/CH	54673.73	54531.95	0.26	0.00	6	54830.31	54531.95	0.55	0.00	17
50-J	**62990.50**	RP/CH	63801.53	62990.50	1.29	0.00	5	64097.81	62990.50	1.76	0.00	13
50-K	**57847.34**	RP/CH	58559.75	57847.34	1.23	0.00	6	58606.38	58065.52	1.31	0.38	16
50-L	**64936.76**	CH	65478.17	64952.83	0.83	0.02	5	65058.55	64981.47	0.19	0.07	13
100-A	**118510.39**	CH	122114.29	119608.14	3.04	0.93	20	121169.25	119678.10	2.24	0.99	57
100-B	**106248.99**	CH	109104.76	107525.63	2.69	1.20	19	108433.69	107325.16	2.06	1.01	47
100-C	**113920.89**	CH	115209.99	114019.54	1.13	0.09	21	115123.84	114440.69	1.06	0.46	58
100-D	**120078.65**	CH	122371.97	120750.57	1.91	0.56	18	122086.50	120142.55	1.67	0.05	45
100-E	**78406.77**	CH	80907.26	79085.24	3.19	0.87	23	80589.83	79093.33	2.78	0.88	63
100-F	**86491.66**	CH	88039.00	87054.86	1.79	0.65	20	88596.00	86491.66	2.43	0.00	50
100-G	**87832.62**	CH	89176.58	88317.18	1.53	0.55	22	88644.83	87832.62	0.92	0.00	63
100-H	**91768.85**	CH	93042.24	91806.09	1.39	0.04	20	92808.57	91948.66	1.13	0.20	50
100-I	**104404.38**	CH	106789.41	105939.13	2.28	1.47	21	106084.63	105636.10	1.61	1.18	55
100-J	**103868.36**	CH	108251.09	107510.53	4.22	3.51	19	107407.96	106098.23	3.41	2.15	45
100-K	**100898.38**	CH	102186.38	101073.71	1.28	0.17	22	102405.88	101966.97	1.49	1.06	60
100-L	**114336.64**	CH	116419.21	115626.96	1.82	1.13	19	116394.86	115655.10	1.80	1.15	46

Table 4.8: Computational results for Multi-Depot PDPTW with Special Requests, 50 and 100 requests

Problem	Best Known Cost		ALNS	25K				AMLNS	25K			
			Avg. Sol.	Best. Sol.	Avg. gap	Best. gap	Avg. time	Avg. Sol.	Best. Sol.	Avg. gap	Best. gap	Avg. time
Notation	z	*Ref*	z_{av}	z_b	Gap_{av}	Gap_b	s	z_{av}	z_b	Gap_{av}	Gap_b	s
250-A	**255186.42**	CH	260643.89	258627.91	2.14	1.35	129	260546.98	258339.06	2.10	1.24	129
250-B	**242372.54**	CH	247857.88	245064.60	2.26	1.11	109	247141.49	245069.68	1.97	1.11	108
250-C	**244974.17**	CH	250624.57	248793.10	2.31	1.56	133	248956.41	247667.02	1.63	1.10	133
250-D	**262290.69**	CH	265083.64	263146.19	1.06	0.33	109	265491.68	262584.55	1.22	0.11	109
250-E	**169588.24**	CH	174147.48	172979.68	2.69	2.00	128	173921.36	172966.00	2.56	1.99	149
250-F	**187906.54**	CH	192151.07	190662.02	2.26	1.47	109	190938.59	189294.69	1.61	0.74	113
250-G	**188265.02**	CH	191844.50	190370.51	1.90	1.12	124	191157.46	189480.33	1.54	0.65	146
250-H	**198267.23**	CH	201505.88	199596.15	1.63	0.67	108	202291.27	200385.66	2.03	1.07	116
250-I	**224297.68**	CH	228099.46	226048.88	1.69	0.78	126	227846.58	226214.92	1.58	0.85	134
250-J	**239555.05**	CH	245589.63	240868.58	2.52	0.55	120	245057.58	242553.82	2.30	1.25	109
250-K	**229261.23**	CH	234942.36	231432.50	2.48	0.95	125	232721.85	231521.69	1.51	0.99	134
250-L	**246739.85**	CH	251344.51	248030.05	1.87	0.52	115	251459.66	248919.46	1.91	0.88	115
500-A	**463345.19**	CH	469805.72	466780.92	1.39	0.74	314	468802.09	465504.96	1.18	0.47	271
500-B	**470814.92**	CH	476944.14	471652.82	1.30	0.18	273	476117.35	474705.09	1.13	0.83	218
500-C	**466278.82**	CH	474111.27	471017.44	1.68	1.02	334	473382.12	469051.24	1.52	0.59	284
500-D	**482608.05**	CH	490795.34	487647.34	1.70	1.04	277	491207.50	486629.87	1.78	0.83	229
500-E	**340001.49**	CH	345685.80	343950.63	1.67	1.16	327	345682.54	342535.04	1.67	0.75	302
500-F	337737.03	RP	342741.79	337737.03	1.48	0.00	284	342811.84	340331.74	1.50	0.77	238
500-G	**376816.71**	CH	382154.10	378502.19	1.42	0.45	326	381714.15	379331.33	1.30	0.67	308
500-H	**389335.78**	CH	393676.16	389949.61	1.11	0.16	291	394879.19	393203.67	1.42	0.99	250
500-I	**431648.00**	CH	438489.43	434295.33	1.58	0.61	337	438987.65	436147.69	1.70	1.04	276
500-J	**452549.30**	CH	458567.79	455011.94	1.33	0.54	286	462468.21	457264.61	2.19	1.04	221
500-K	**451402.01**	CH	458774.35	452319.94	1.63	0.20	341	458631.37	456020.92	1.60	1.02	291
500-L	**466806.55**	CH	473541.96	470132.02	1.44	0.71	290	473545.95	467644.44	1.44	0.18	238
Avg.					1.77	0.70	113			1.64	0.69	113
Lan./CPU			C++/ AMD Opteron		250		2.4GHz	C# / Intel Core i7				3.5GHz

Table 4.9: Computational results for Multi-Depot PDPTW with Special Requests, 250 and 500 requests

coding, and the efficient data structure used in C++, the computational time of the AMLNS could be comparable or faster than the ALNS.

From the computational results in Table 4.8 and 4.9, the Avg. of Gap_{av} between the ALNS and the AMLNS are 1.77 and 1.64 respectively. In other words, we improved the Avg of Gap_{av} of 0.13% from the original ALNS. The average of 10 runs determines the robustness of an algorithm. Double-underlined numbers indicate the average value of best solutions of 10 runs, which is obtained from the AMLNS, and shows a better result than the ALNS of Pisinger and Ropke (2007). Single-underlined numbers indicate the best solution out of 10 runs, which is obtained from the AMLNS, and is better than the numbers from that in Ropke (2009a). Bold numbers mark the best known solution from all experiments conducted in this Chapter. We obtained 47 best known solutions out of 48 test instances during all experiments. According to the improved average values, we then conclude that the AMLNS is competitive to the ALNS in terms of solution quality in this set of test instances.

In Figure 4.8, we illustrate the network structure of a new best known solution obtained for Problem 50A with the new objective function value = 62833.33.

The squares represent depot. Each coloured arrow represents each vehicle. The pickup is symbolised by a triangle with an even number. Its corresponding delivery location is that even number plus one. It is to note that not every vehicle has to be used. In other words, a vehicle may not leave the depot, for instance in Depot 4 (rectangle with number 4). The detailed schedule of the solution in Figure 4.8 is shown from Table 4.10 to 4.12.

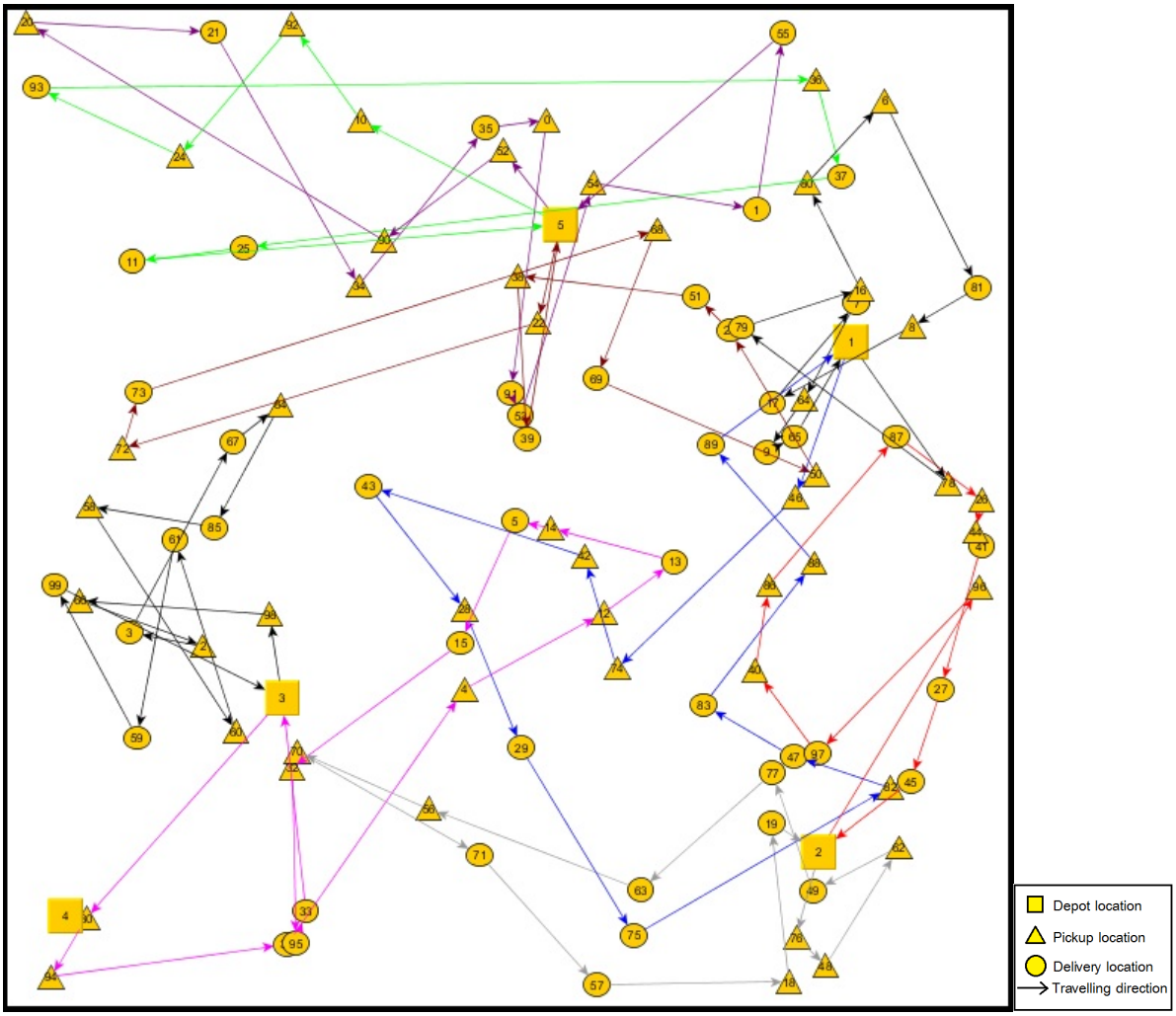


Figure 4.8: New Best Known Solution of Problem 50A (100 locations) obtained by the AMLNS

Seq.	Arr. t	Dep. t	S/D	L.
V1				
D2	-	0.00	0	-
76	90.96	1192.00	14	14
48	1233.73	1443.73	17	31
62	1587.46	1879.00	14	45
49	1978.73	2091.73	-17	28
77	2220.76	2406.76	-14	14
63	2588.79	2830.79	-14	0
56	3065.35	3176.35	16	16
70	3324.60	3531.60	9	25
71	3748.78	3862.78	-9	16
57	4043.33	4145.33	-16	0
18	4343.35	4476.35	9	9
19	4641.34	4826.34	-9	0
D2	4883.79	-	0	0

Seq.	Arr. t	Dep. t	S/D	L.
V4				
D5	-	0.00	0	-
52	97.01	291.01	15	15
90	445.21	594.21	6	21
20	1026.39	1165.00	6	27
21	1358.26	1761.00	-6	21
34	2062.90	2239.90	10	31
35	2449.02	2580.02	-10	21
0	2643.52	2823.52	15	36
91	3107.94	3248.94	-6	30
53	3274.02	3400.02	-15	15
54	3651.47	3859.47	7	22
1	4030.61	4183.61	-15	7
55	4367.60	4498.60	-7	0
D5	4802.09	-	0	0

Seq.	Arr. t	Dep. t	S/D	L.
V5				
D1	-	0.00	0	-
46	169.85	758.00	15	15
74	1013.31	1175.31	14	29
42	1296.19	1483.19	13	42
43	1717.53	1892.53	-13	29
28	2053.56	2175.56	6	35
29	2330.25	2584.00	-6	29
75	2810.04	2992.04	-14	15
82	3298.03	3502.03	13	28
47	3607.03	3780.03	-15	13
83	3887.07	4012.07	-13	0
88	4198.09	4311.09	12	12
89	4472.71	4681.71	-12	0
D1	4860.52	-	0	0

Table 4.10: Detailed Schedule of the New Best Known Solution for Problem 50 A

Seq.	Arr. t	Dep. t	S/D	L.
V7				
D3	-	0.00	0	-
98	87.97	681.00	19	19
66	877.50	1005.50	15	34
2	1140.92	1320.00	12	46
3	1397.28	1636.00	-12	34
67	1860.18	2013.18	-15	19
84	2075.19	2220.19	16	35
85	2364.25	2498.25	-16	19
58	2629.11	2878.11	17	36
60	3154.92	3328.92	6	42
61	3536.71	3732.71	-6	36
59	3941.38	4112.38	-17	19
99	4291.80	4458.80	-19	0
D3	4721.31	-	0	0

Seq.	Arr. t	Dep. t	S/D	L.
V9				
D5	-	0.00	0	-
22	103.59	925.00	19	19
72	1373.26	1483.26	17	36
73	1543.70	1722.00	-17	19
68	2282.76	2384.76	18	37
69	2551.15	2788.15	-18	19
50	3035.79	3185.79	12	31
23	3359.35	3579.35	-19	12
51	3628.85	3784.85	-12	0
38	3970.04	4097.04	14	14
39	4265.34	4507.34	-14	0
D5	4730.94	-	0	0

Seq.	Arr. t	Dep. t	S/D	L.
V10				
D1	-	0.00	0	-
78	177.53	1228.00	9	9
79	1495.80	1712.80	-9	0
16	1841.84	2015.84	7	7
80	2138.82	2285.82	6	13
6	2402.55	2595.55	12	25
81	2811.10	2991.10	-6	19
8	3069.13	3226.13	7	26
17	3391.26	3639.26	-7	19
7	3774.08	3973.08	-12	7
64	4085.85	4310.85	10	17
9	4378.28	4613.28	-7	10
65	4646.40	4810.40	-10	0
D1	4924.28	-	0	0

Table 4.11: Detailed Schedule of the New Best Known Solution for Problem 50 A (cont.)

Seq.	Arr. t	Dep. t	S/D	L.
V11				
D2	-	0.00	0	-
96	320.36	878.00	13	13
97	1117.74	1277.74	-13	0
40	1385.54	1543.54	11	11
86	1632.81	1906.00	13	24
87	2107.42	2306.42	-13	11
26	2415.82	2657.82	12	23
44	2691.36	3123.00	19	42
41	3139.16	3641.00	-11	31
27	3794.84	3988.84	-12	19
45	4088.77	4194.77	-19	0
D2	4314.58	-	0	0

Seq.	Arr. t	Dep. t	S/D	L.
V12				
D3	-	0.00	0	-
30	303.86	499.86	5	5
94	571.21	1183.00	8	13
31	1429.36	1677.36	-5	8
4	1997.76	2244.76	12	20
12	2409.01	2545.01	7	27
13	2635.22	2829.22	-7	20
14	2961.92	3106.92	19	39
5	3144.57	3311.57	-12	27
15	3449.46	3695.46	-19	8
32	3912.06	4145.06	12	20
95	4326.10	4562.10	-8	12
33	4596.59	4760.59	-12	0
D3	4981.89	-	0	0

Seq.	Arr. t	Dep. t	S/D	L.
V14				-
D5	-	0.00	0	
10	232.59	889.00	16	16
92	1010.83	1210.83	12	28
24	1389.58	1517.58	19	47
93	1681.73	1902.00	-12	35
36	2707.04	2892.04	5	40
37	2995.36	3436.00	-5	35
25	4057.42	4179.42	-19	16
11	4295.27	4493.27	-16	0
D5	4936.90	-	0	0
	Tot. Dis.	**19677.03**		
	Obj. fn.	**62833.33**		

Table 4.12: Detailed Schedule of New Best Known Solution for Problem 50 A (cont.)

From Table 4.10 to Table 4.12, the details of 9 vehicles are shown in terms of arrival time, departure time and loading. The first row of each table represents the vehicle id. The following rows are the location sequences. The location with D represents the depot id. It can be used to validate the solution of the benchmark instances of Ropke and Pisinger (2006). The objective value of the new best known solution of problem

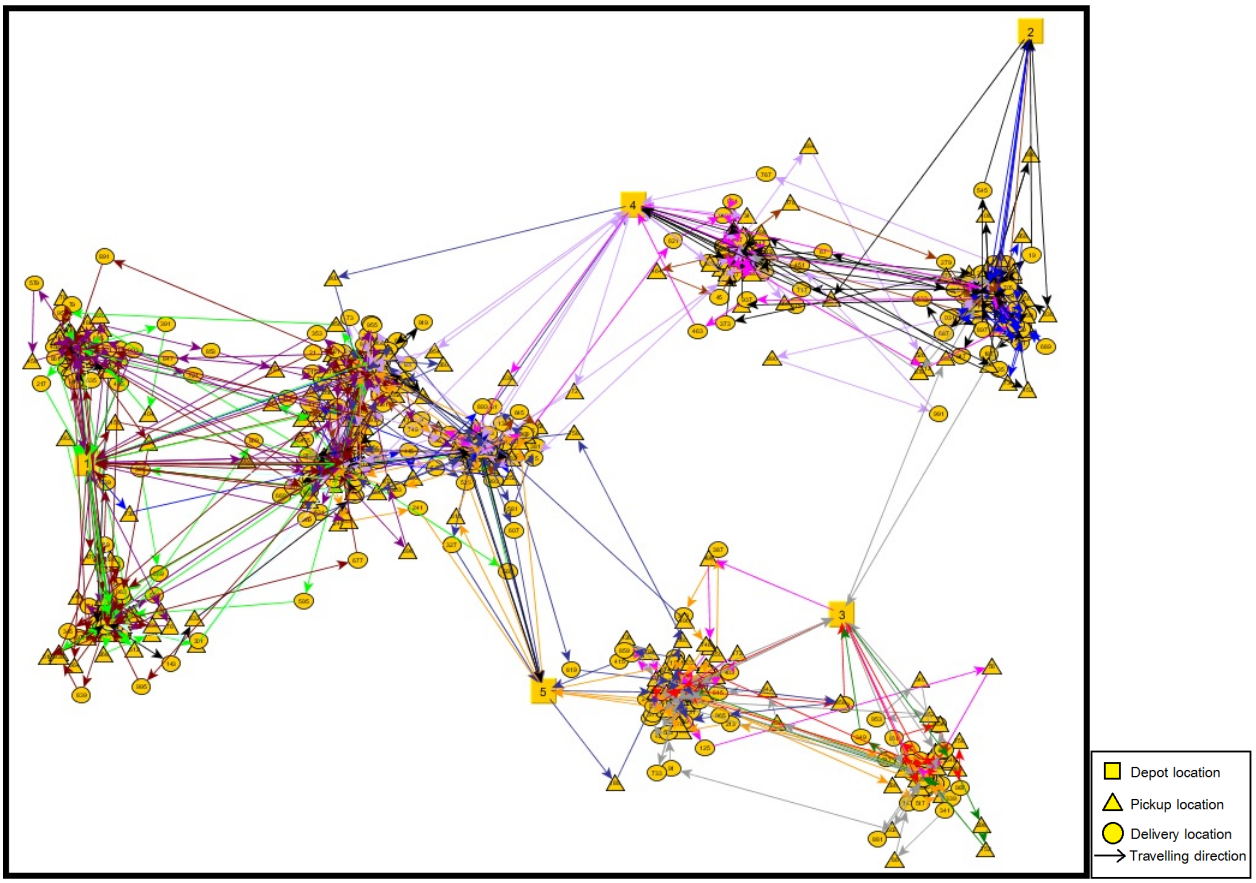


Figure 4.9: New Best Known Solution of Problem 500E (1000 locations)

50A is 62,833.33 which is comprised of the total distance, 19,677.03, and the total time, 43,156.30.

We also demonstrate the size and complexity of problem 500 E in Figure 4.9. This problem is characterised by the same depots, normal requests, and clustered geographical distribution. In this problem, we also obtain a new best known solution with objective function = 340001.49. Moreover, the new best known solutions of Problem 50K, 100B, 250C, 500D and the network topology of the new best known solution of problem 50F, 50L, 50H are shown in Appendix C.

4.7 Discussion

From the GAs' point of view, Goldberg (1989) and Goldberg (2002) emphasised the importance of Building Blocks (BBs) toward the design of competent GAs. It is assumed that recombining the best sub-structures from the good individuals may result in reproducing an individual with higher fitness. The author discussed that a schema is a similarity template describing a subset of strings with similarities at certain string positions. It seems perfectly reasonable to play mix and match with some of the substrings that are highly correlated with the past success. Nagata and Kobayashi (2010) suggested that meaningful building blocks from combining both parents must be inherited. Falkenauer (1998), Pankratz (2005), Rekiek et al. (2006) discussed that, for grouping problems, each gene represents a group of objects instead of a single object. Thus, the groups are the building blocks that are sampled and recombined by genetic operators. In this problem, it is also assumed that the genes, subassemblies, groups and building blocks are defined as routes over a specified number of vehicles. Similar to the concepts by these authors, given a finite number of vehicles, the same vehicle id between two parents represents the same string position. The route measure of IVX increases the selection pressure for highly fit building blocks for a crossover. Similarly, it seems reasonable to mix and match good routes to other solutions. Goldberg (1989) and Goldberg (2002) also confirmed that, in competent GAs, it is ensured that the building blocks will propagate from generation to generation without using special memory other than the population. This refers to *implicit parallelism*. For the AMLNS, the IVX also ensures the propagation of good routes by tournament and IVX, resulting in the implicit parallelism. The IVX can also inherit good building blocks from one parent to the other with the view to interrupting the structure of solutions at minimum.

The proposed IVX can prevent any feasibility violation from vehicle capacity constraints. In the IVX, if the vehicles from both parents have the same sequence of routes but selected by the route measure, then the exchange operation for that vehicle is ig-

nored as it can generate the same offspring. This technique is implemented because, sometimes, if the pressure of tournament selection, crossover and replacement is so high that the population prematurely converges, many routes become identical. This technique should alleviate the concern of reproducing the same offspring as its parents. Moreover, the ALNS with the large q can be viewed as the heavy mutation avoiding the premature convergence due to a small population size, and the pressure from selection and IVX. When the small q is used, the ALNS act as local search operators to exploit the search space or intensify the search. Adaptive features help decide the competing operators. Using several large neighbourhood operators and the adaptive mechanism that collect the adaptive weights of worse solutions within Threshold can diversify the search and also correspond to the concept of Adaptive Memetic Algorithm or Meta-Lamarckian Learning in MAs in Ong and Keane (2004) and Ong et al. (2006). Applying randomness parameters and noises can also be seen as one feature of the mutation operator.

From ALNS's perspective, population can be considered as a memory collecting the partial feasible routes. Population contains different referent points in the search space, thus restoring the search of promising solutions using binary tournament selection. The IVX can be seen as a removal operator or neighbourhood structure that removes whole good route(s) at a time. Modified TA is a deterministic version of SA which posses SA's behaviour and helps scheduling the smaller q after a simple criteria, the cut-off point. The separate tournament selection and wheel partitioning is a novel mechanism that helps bridge the population based and single-solution approach into one algorithm.

The design of the hybrid meta-heuristics requires the synergy effect of all recombined components. From the experiments, we observe that each metaheuristic has its own mechanism for intensification and diversification. All functional components must be well investigated in terms of the effect of intensification and diversification, and the theoretical aspects of how each component of each state-of-the art metaheurisitic works to solve the problem must be understood. They can be recombined if they can strengthen

the search, but they should be avoided if they obstruct each other in each stage. All functional components should work together to synergise the different promising parts from the considered metaheuristics. One has to understand the ideal situation that the search of the solution must be efficient. Many authors claim that the balance between diversification and intensification is important. However, Sorensen (2012) confirmed that it is difficult to quantify the optimal balance. This is still open to research.

In this experiment, the aggregate diversification and intensification phases are separated into two stages: before and after cut-off points respectively. When the problem is large, the solution space is also large, but only a limited time is given. In order to cope with these issues, the early state should diversify the search but still obtain good solutions from intensification. The later stage should intensify more than diversify the search in order to thoroughly search for the good basin of attraction. Toward the end of the search, the TA nearly rejects non-improving change. Then, due to the reduction of q, the number of different neighbourhood operators plays an important role in diversification, providing different search directions. We believe that operators and their interaction in the diversification and intensification phases must be efficient. The balance between diversification and intensification is empirically approximated by design changes, parameter scanning and parameter tuning. The synergy of diversification and intensification is essential in designing hybrid metaheuristics.

In order to apply the AMLNS to several VRP problems, one may consider that in some problems, the number of customer locations in each route is too large or too small. In these cases, we suggest that the number of routes to remove may be re-adjusted or remain the same upon empirical investigation of problem domains. With different objective functions, the route quality measure(s) should correspond to the objective function. They should not only look for the long routes but also short routes with good quality. However, these also require empirical investigation for implementation comparing the solution quality of the AMLNS against the standard benchmark test instances taken from the literature.

4.8 Summary

In this Chapter, we designed a new Adaptive Memetic Large Neighbourhood Search (AMLNS). The AMLNS is the hybrid metaheuristic between the ALNS, GAs and TA. A new adaptive crossover operator was designed for highly constrained problems such as MD-PDPTW-SR. The separate tournament selection and modification at the roulette wheel selection can simply incorporate the crossover operator into ALNS. The cut-off mechanism is designed to separate the search into two stages: population-based and then single solution approaches. The first stage gives the diversification effect. The second stage has the intensification effect. In other words, the aggregate diversification and intensification phases are separated by the cut-off mechanisms. The proposed partitioning mechanism is useful in organising the crossover and LNS operators in each stage, before or after cut-off point. The Modified Threshold Accepting replaces the Simulated Annealing at the master local search framework to schedule the small q. The synergy effect and contribution of all operators in both stages are essential. The AMLNS hybridises the population-based and single solution approaches into one hybird metaheuristic.

We evaluated using small- to large-sized test instances from Ropke and Pisinger (2006). From all of the experiments, we obtained 12 feasible solutions of 50 requests, which had objective function values equal to the best known solutions from Pisinger and Ropke (2007). Overall, we obtained 47 best known solutions out of 48 test instances from all experiments. In 10 runs, for all 48 test instances, we improved the Avg of Gap_b 0.01 %, compared to the computational results of Pisinger and Ropke (2007). We also improved the average values of 10 runs, Gap_{av}, 0.13 % compared to the computational result of Pisinger and Ropke (2007). Moreover, we expect that, with optimised coding, the computational time of the AMLNS is comparable to the ALNS. We then concluded that the computational results of the AMLNS are competitive to those of the ALNS in this set of test instances. We believe that implementing the AMLNS is worthwhile for

improving robustness, speed and accuracy.

The concept of parallelising the ALNS by the AMLNS seems promising and can avoid the concern of PALNS in terms of solution cycling. A further experiment is to use the AMLNS by parallel computing with slight modifications. We expect that the AMLNS applied in parallel computation can improve both efficiency and effectiveness. We suggest that, when using multi-threading technology, each thread represents the individual solution. The number of individuals in the population is equal to the number of threads. The Memetic operators, excepting the first tournament selection, are applied until the cut-off point. Instead of applying the ALNS to the best solution found, the ALNS is used for all threads (solutions) containing its own roulette wheel and Threshold. In this way, it is easy to extend the AMLNS to parallel computing that can speed up the search further. By using the principle of the AMLNS, we believe that searching from the larger number of solutions, depending on the number of threads, can help improving both solution quality and computational time.

Chapter 5

Development of the Adaptive Memetic Large Neighbourhood Search: Implementational Aspects

This Chapter shows the development of the Adaptive Memetic Large Neighbourhood Search (AMLNS), as used in Chapter 4. The development of the AMLNS in this Chapter is based on the computational experience derived from the experiments using the MA presented in Chapter 3 The development of MA in Chapter 3 provides the in-depth investigation of implementing a population-based approach to the variants of the MDPDP. In addition, components of the MA from Chapter 3 including tournament selection, chromosome representation, fixed forward insertion method and a reduction rule in terms of vehicle capacity are the core basis for further implementation to solve the MD-PDPTW-SR.

Comparing the ALNS and MA, recently, Ribeiro and Laporte (2012) investigated the cumulative capacitated vehicle routing problem (CCVRP). The authors presented an Adaptive Large Neighbourhood Search (ALNS) for the CCVRP and compared it with two recently published MAs, proposed by Ngueveu et al. (2010). Even though

MAs provide some better solutions than ALNS, the ALNS overall outperforms the two MAs in terms of computational time and robustness.

The computational comparison between the ALNS and MAs from Ribeiro and Laporte (2012) and Blum and Roli (2003)'s confirmation, "mixing and hybridising is often better than purity", have encouraged a hybrid metaheuristic study rather than a study focusing exclusively on MAs. . Blum et al. (2011) pointed out that the hybridisation of different algorithms is to exploit the complementary character of different optimisation strategies, that is, hybrids are believed to benefit from "*synergy*". Choosing an adequate combination of complementary algorithmic concepts can be the key for achieving top performance in solving many hard optimisation problems. However, the contribution of key components must be thoroughly investigated.

According to the principles described earlier of designing hybrid metaheuristics, the selected state-of-the-art metahueristics are broken down according to functional components and analysed in terms of diversification and intensification. It is important to understand how each metaheuristic works and why they are successful. Figure 5.1 shows the design matrix we developed for hybridising metaheuristics and the AMLNS.

In Figure 5.1, the key components of each state-of-the-art metaheuristic selected are shown with the analysis of its effect toward intensification (I) and diversification (D) in brackets. The unified framework of the hybrid metaheuristic presented in Raidl (2006) and some other operators gives ideas of the key components required for metaheuristics, as shown in the top row in Figure 5.1.

Each metaheuristic has its own concepts, philosophies and operators. Blum and Roli (2003) suggest that although different metaheurictics are different in terms of concepts, the mechanisms for efficiently exploring a search space are all based on intensification and diversification. It is important to identify "sub-tasks" or functional components in the search process where some metaheuristics perform better than others.

In Figure 5.1, from our analysis, some components may be solely contributing to diversification or intensification. However, it is also possible that some operators have

	Pool	S	OF	SCM	IM	IF	Restart	Stopping Criteria
ALNS (Pisinger and Ropke, 2007)	2(-)	1(-)	Current solution(-)		Large Neighbourhood Search (D) Different Neighbourhood Search(D) Randomised Parameters (D) Greedy removal+ Regret Insertion (I>D) Random removal + Regret insertion (D>I) Adaptive features (I>D) Noising (D)	SA's Acceptance (D->I)		Max. Iterations(-)
MA (Nagata and Kobayashi, 2010)	>2(D)	>1(D)	Random selection (D)	SREX (D>I)	Local Search (I)	Replacement by comma strategy (I)		Max. iteration of. stagenates of the best solution (-)
UHGA (Vidal et al., 2013)	>2(D)	>1(D)	Tournament selection (I>D)	AIX (D>I)	Education and Repair (I)	Population with diversity management (D) Replacement by plus strategy (I)	Diversification Phase(D)	Max.Iterations or time (-)
AGES (Mester and Braysy,2005)	2(-)	1(-)	Current solution(-)		Large Neighbourhood Search (D) Local Search (I)	Replace if better (I) GLS-objective function (D)	Return to the so-far best solution (I->D)	No improvemet in both stages for Max. Iterations (-)
MSLS (Braysy et al., 2004)	>2(D)	1(-)	Current solution(-)		Reduce number of route (I) Improve distance(I) GENICROSS and IOPT(I)	Replace if better(I) Threshold Accepting (I)	Multi-start from new solutions(D)	Max. Iterations(-)

Figure 5.1: Design Matrix for Hybridising Metaheuristics

both diversification and intensification or one of them is higher than the other. The possible combinations of designs of all state-of-the-art metaheuristics are numerous. Enumerating and experimenting on all designs is prohibitive. Designing a hybrid metaheuristic from many metaheuristics can be viewed as solving a combinatorial optimisation problem with infinite number of solutions. One way to limit the possible combinations of designs is through the selection of the state-of-the-art metaheuristics. Once limited, it can be solved by a metaheristic, for example a number of promising designs may be explored and the promising ones are further improved upon. The resulting design may not be the global best design, but it should provide a good design with a certain level of robustness, preciseness and speed. Then, the designer should understand the underlying principle of the successful reintegration and promising hybridisation. The principles of hybridisation can be viewed in Raidl (2006), Talbi (2009) and Blum et al. (2011). In this Chapter, the design concept for a hybrid metaheuristic is similar to that of a Memetic Algorithm. The process of design involves selection + recombination and selection + improvement. After good metaheuristics are selected, the functional components are recombined. If the offspring obtains a higher solution quality, then it replaces its parents. The improvement can be carried out both by design changes from problem-specific knowledge and observation etc. Moreover, the improvement can be carried out by parameter scanning and parameter fine-tuning. The models are compared on a pairwise basis, reasoning about the direction of the desirable results. In order to reintegrate, new operators or mechanisms may be required to synergise the metaheuristics or modify some operators to tackle the nature of the problems. It is also possible to replace an existing component that seems weak in terms of the functional requirement, from other metaheuristics with the same function.

As mentioned in Section Design and Tuning Process, we categorised the development into three stages: design changes, parameter scanning and parameter fine-tuning. In this section, we will show the development of designs and parameter scanning. We observe that the design change can have a significant effect on both solution quality

and computational time. According to Equation 4.20, we therefore set $\phi, \upsilon, \varsigma = 1$ and follow the Design and Tuning Process. Due to page width limit, hereafter, the $Gap_{av/BA}(\%)$ and $Gap_{b/BA}(\%)$ are represented by $Gap_{av}(\%)$ and $Gap_{b}(\%)$, respectively.

In order to investigate the effect of each design change or significant changes in parameters, we only modify one design or parameter at a time. We initially mimicked the original ALNS of Pisinger and Ropke (2007) as a point of departure. Therefore, the benchmark algorithm is the ALNS of Pisinger and Ropke (2007). However, the randomness parameters of some removal operators were not reported in either Pisinger and Ropke (2007) and Ropke and Pisinger (2006). These parameters are empirically set as shown in Table 4.6. For matters of clarity the selected design changes are grouped into single-solution, population-based, and hybrid approaches.

5.1 Single-solution Approach

From Pisinger and Ropke (2007), the SA uses an exponential cooling rate from the start temperature, T_{start}, and decreasing temperature, T, according to the expression $T = T \cdot c$, where c is the cooling rate, $0 < c < 1$. Given the current solution x, a candidate solution $x^{'}$ is accepted with probability:

$$e^{-(f(x^{'})-f(x))/T}$$

Ropke and Pisinger (2006) stated that, by using Simulated Annealing, the start temperature is set such that a solution that is $w\%$ worse than the current solution is accepted with probability 0.5. The authors empirically set the $w\%$ to 0.05.

5.1.1 ALNS using Threshold Accepting

In this design, the Simulated Annealing used in the ALNS is replaced by the Threshold Accepting because Yagiura and Ibaraki (2001) suggested the superior performance of TA over SA. Once the start Threshold, $St.Thres$, is initialised. The $Thres$ is reduced

by $\triangle E$ in every iteration, as shown in Equation 5.1.

$$Thres = Thres - \triangle E \tag{5.1}$$

We denote $\triangle E$ as a linear cooling rate. The $w\%$ in Pisinger and Ropke (2007) can be viewed as $St.Thres$ from the TA's point of view. Therefore, the TA always accepts the solutions if its w % worse objective function is better than the Threshold, or with probability =1, in contrast to = $e^{-(f(x^{'})-f(x))/T}$ of SA. Therefore, we initially set the $w\%$ to half i.e. 0.025 or $\frac{0.05}{2}$.

As suggested by Pisinger and Ropke (2007), the number of requests to remove should be reduced at the latter half of iterations. We then introduce the cut-off point to determine where the smaller number of q should be applied. The percentage of Start Threshold is used to determine the cut-off points, $Coff(\%)$. The number of requests to remove is then reduced to $\theta\%$ of its original size. We term this algorithm as Threshold Accepting and Adaptive Large Neighbourhood Search (TA-ALNS). The first design was developed on the ALNS with some modifications. The Threshold Accepting (TA) replaces the Simulated Annealing (SA) in the original ALNS, so called TA-ALNS 1. We experimented by considering design changes and parameter scanning, as shown in Table 5.1.

Design	$St.Thres$	$\triangle E$	$Coff(\%)$	$\theta(\%)$	$Gap_{av}(\%)$	$Gap_b(\%)$	$Avg.Time(\%)$	$Obj.Fn$
TA-ALNS 1	0.025	0.000001	100	100	0.94	1.23	0.00	2.17
TA-ALNS 2	0.025	0.000002	100	100	0.75	0.78	-0.33	1.20
TA-ALNS 3	0.005	0.0000002	100	100	-0.22	-0.04	-0.02	-0.28
TA-ALNS 4	0.025	0.000001	15	75	1.11	1.27	-0.014	2.39
TA-ALNS 5	0.025	0.000001	15	50	1.29	1.42	-0.04	2.67
TA-ALNS 6	0.025	0.000001	30	75	1.17	1.45	-0.081	2.54

Table 5.1: Experiments on design and parameters for TA-ALNS

In Table 5.1 , the cut-off mechanism is not used in the design TA-ALNS 1 to 3. It is important to note that the Threshold of TA-ALNS 2 reaches zero at the 12,500th iteration due to the linear cooling rate,$\triangle E$, at 0.000002. After that, we set the Threshold

equal to zero toward the end of the search. We demonstrated the parameter setting and also computational results of the TA-ALNS designs in Table 5.1. The following remarks consider the knowledge gained from our experiment, and its potential implications for further research:

- By comparing TA-ALNS 1 and TA-ALNS 2, TA-ALNS 1 is shown to be more diversified than TA-ALNS 2 due to lower $\triangle E$.
- By comparing TA-ALNS 1 and TA-ALNS 3, the interaction between the Threshold and cooling rate was revealed. The smaller $St.Thres$ gives a higher intensification effect. While the smaller the cooling rate applied, the less the intensification affect. Therefore, TA-ALNS 3 is more intensified at the beginning due to the smaller start Threshold but more diversified during the search due to the smaller cooling rate. We also observed that the interaction between $St.Thres$ and $\triangle E$ makes a significant impact on solution quality.
- By comparing TA-ALNS 4 and 5 when using small moves, too large reduction in the number of requests, $\theta\%$, to remove may give worse solution quality.

We observed that the exponential cooling rate of the SA reduces acceptance probability sharply in the early stage. Therefore, we modified the TA to possess that behaviour.

5.1.2 ALNS using Modified Threshold Accepting

In this design, we modify the TA by applying an exponential cooling rate, c_{exp}, and updating $Thres$ in every iteration, as shown in Equation 5.2.

$$Thres = Thres \cdot c_{exp} \tag{5.2}$$

We refer to this algorithm as Modified Threshold Accepting and Adaptive Large Neighbourhood Search (MTA-ALNS). The experiments on parameter scanning and their computational results are given in Table 5.2.

Design	$St.Thres$	c_{exp}	$Coff(\%)$	$\theta(\%)$	$Gap_{av}(\%)$	$Gap_b(\%)$	$Avg.Time(\%)$	$Obj.Fn$
MTA-ALNS 1	0.025	0.99975	100	100	0.24	0.23	-0.15	0.32
MTA-ALNS 2	0.025	0.99985	100	100	0.12	0.39	-0.13	0.38
MTA-ALNS 3	0.005	0.99985	100	100	-0.10	0.16	-0.30	-0.24
MTA-ALNS 4	0.025	0.99975	15	75	0.05	0.33	-0.23	0.15
MTA-ALNS 5	0.025	0.99975	15	50	0.41	0.40	-0.36	0.45
MTA-ALNS 6	0.025	0.99975	30	75	0.17	0.18	-0.26	0.09

Table 5.2: Experiments on designs and parameters for TA-ALNS

According to the experiments in Table 5.2, we draw the remarks for the MTA-ALNS design as follows:

- By comparing MTA-ALNS 1, 2 and 3, the interaction between $St.Thres$ and c_{exp} has impact on solution quality.
- By comparing MTA-ALNS 4, 5, and 6, the interaction between $Coff$, and θ also has impact on solution quality.

In order to explain the differences between MTA-ALNS and TA-ALNS, we showed the different Threshold values due to the different implementation of the cooling rate in Figure 5.2 (left).

According to the TA-ALNS and MTA-ALNS, we represented the typical search trajectory when applying the linear and exponential cooling rate by TA-ALNS 3 and MTA-ALNS 3, as demonstrated in Figure 5.2 (right).

According to Figure 5.2 (right), given the same $St.Thres$ for TA-ALNS 3 and MTA-ALNS 3, $Thres$ of MTA-ALNS 3 is dramatically reduced in the early stage, resulting in less accepting non-improving solutions. In other words, MTA-ALNS 3 initially possesses a more intensified behaviour, while, TA-ALNS 3 is rather diversified and potentially

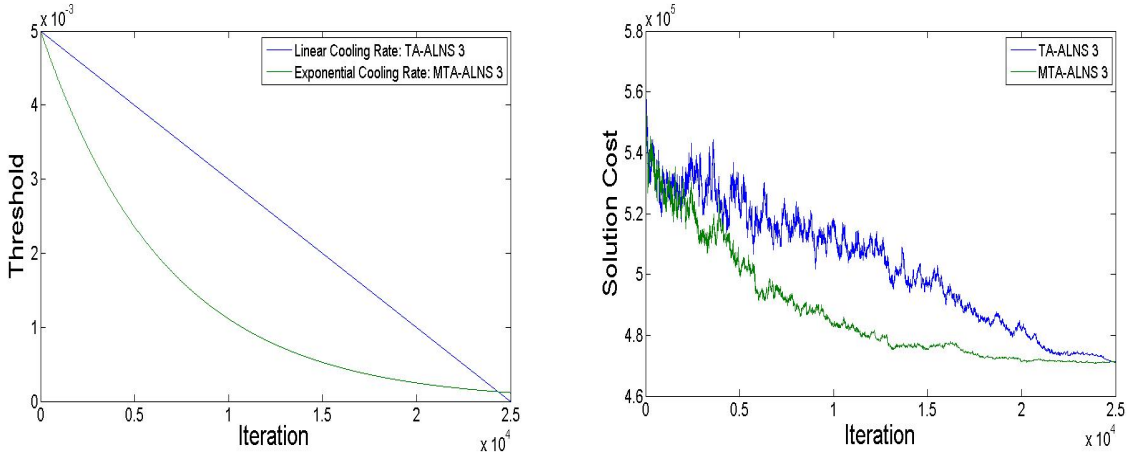


Figure 5.2: (Left) Thresholds between a linear and exponential cooling rate for TA-ALNS 3 and MTA-ALNS 3, respectively. (Right) Search trajectories between TA-ALNS 3 and MTA-ALNS 3

explores a larger search space. In this section, up to now, we gain the improved designs from the cycle of selection + recombination and selection + improvement. We also learnt the impact of solution quality due to several key components. In order to further improve the designs, we then repeated the cycle of selection + recombination and selection + improvement again.

Ropke (2009b) attempted to take advantage of parallel computing using the Parallel ALNS (PALNS). However, the PALNS seems to work against the SA principle. In terms of parallel computing, the concept of Genetic Algorithm and Memetic Algorithms are widely used. We therefore experimented with the ALNS embedded in population-based approaches in Section 5.2.

5.2 Population-based Approach

Rodriguez-Diaz et al. (2010) claimed that the design of hybrid metaheuristics, combining the simulated annealing and evolutionary algorithms, provides a fruitful research line. The authors proposed a GA-based Multiple SA (GAMSA), whose search process

simulates several parallel simulated annealing processes. They performed an empirical study comparing the behaviour of a representative set of the hybrid approaches based on evolutionary algorithms and simulated annealing found in the literature. The GA-based multiple SA (GAMSA) is the best performing hybrid metaheuristics between evolutionary algorithms and simulated annealing (HM-EA/SA). The GAMSA considers the execution of multiple SA processes that share a unique steady-state EA. Several SA processes promote diversification by exploring different regions of the search space. On the other hand, the population of the EA allows the SA agents to communicate with one another in order to explore the search space. One can view that GAMSA uses a population of SA processes that cooperate by employing EA's operators to explore the search space. The steady-state EA creates one single candidate solution at each iteration by crossing over the solution of the master SA and another one from the population. GAMSA can be classified as teamwork collaborative in Talbi (2009). Therefore, the GA is one promising metaheuristic for hybridising with the SA.

In this experiment, due to the original ALNS implementing on a single solution, we implemented a single-threading program by modifying the ALNS for the population-based approach. In other words, several solutions of ALNS are executed in a sequential manner, one after another. We begin with applying the MTA-ALNS to a number of solutions. It is important to note that, in hybrid principles, the selection of GA and MTA-ALNS is due to their historic success. The key components of both metaheuristics are recombined.

5.2.1 Multiple ALNS

In this design, we initially construct a number of solutions by using different regret insertion heuristics. Merz and Freisleben (1999) showed that due to the computation times consumed by local search operators, the population size of a memetic algorithm is typically small compared to genetic algorithms. According to Rodriguez-Diaz et al. (2010), 4 solutions was an appropriate number for GAMSA. We also set the number of

solutions accordingly. In this design, we attempt to take advantage of operators widely used in GAs. We firstly adopted the binary tournament selection operator in order to investigate its effect with the number of solutions. The TA and MTA were both experimented on together using the binary tournament selection. We choose the binary tournament selection according to Ombuki and Hanshar (2009) due to its flexibility and adjustable sensitivity. Then, we adopted the replacement strategy similar to Nagata and Kobayashi (2010) to maintain the population diversity.

The reason for using this type of tournament and replacement strategy is due to its flexibility, synergy effect, and, with a small population size, its prevention of premature convergence. The Threshold is shared throughout the solution due to the tournament selection operator. When the Threshold belongs to each solution, and it is reduced only when generating that solution: the worse solutions are kept diversified and rarely selected by the tournament selection. The roulette wheel is separated for each solution in order to trace the efficiency of operators corresponding to the time state of each solution. According to the experiments conducted on the shared wheel and the separate wheel of roulette wheel selection for operators, smoothened scores and roulette wheel probabilities were found to be different toward the end of the search. From our experiments, separating the roulette wheel and allocating to each solution can measure the goodness of operators better than sharing the roulette wheel with all solutions. Since, we believe that each solution has a different stage of exploration or exploitation at time t. Therefore, the roulette wheel should apply the operators to suit a particular stage of the search by each solution.

As shown in Table 5.3, MALNS 1 to 3 applied TA and, MALNS 4 to 6 applied MTA. The $St.Thres$ is equal to 0.005 for MALNS 1 to 6.

According to the experiments, we draw the remarks from Table 5.3 as follows:

- By comparing the MALNS 1 and MALNS 3, and MALNS 4 and MALNS 6, using tournament selection mechanism, we can improve the solution quality.

Design	Cooling Rate	$Coff$	$Tour.Prob.$	$Gap_{av}(\%)$	$Gap_b(\%)$	$Avg.Time(\%)$	$Obj.Fn$
MALNS1	$\triangle E = 0.0000002$	0.15	0.0	0.60	1.04	-0.07	1.57
MALNS2	$\triangle E = 0.0000002$	0.3	0.0	0.70	0.92	-0.09	1.53
MALNS3	$\triangle E = 0.0000002$	0.15	0.8	0.36	0.57	-0.08	0.85
MALNS4	$c_{exp} = 0.99985$	0.15	0.0	0.65	0.65	-0.12	1.18
MALNS5	$c_{exp} = 0.99985$	0.3	0.0	0.87	1.25	-0.17	1.95
MALNS6	$c_{exp} = 0.99985$	0.15	0.8	0.31	0.48	-0.14	0.65

Table 5.3: Experiments on design and parameters with computational results for MALNS

- By comparing the MALNS 3 and MALNS 6, the MTA is more suitable than the TA for the MALNS.

In the MALNS design, we showed that the tournament selection has a significant impact on the solution quality. Goldberg (1989) claimed that the power of GAs is the result of synergy effect between tournament selection and crossover operator. Therefore, it is investigated in the next section.

5.2.2 Memetic Algorithm and ALNS

In this experiment, we attempt to incorporate the crossover operator into the MALNS. The operators in ALNS can be viewed as the local search or heavy mutation operators depending on the number of requests to remove, q. Therefore, the integration of tournament selection, crossover and ALNS's operators can be considered as a variant of MAs. We, therefore, refer to this design as Memetic Algorithm and ALNS (MA-ALNS).

Pankratz (2005), Nagata and Kobayashi (2010) and Hosny (2010) applied GAs and MAs to variants of VRP and PDPTW. Some of their computational results are competitive to the existing state-of-the-art heuristics. The crossovers they used share some similarities in that the whole route(s) can be selected from one solution and transfered to another. The multi-depot PDPTW, however, is different from those problems in terms of assumptions and constraints, as described in $SCM(S)$ *and* $IM(S)$ at Section

4.4.2. Therefore, we propose a new crossover operator for the MDPDPTW. We expect that the new operator introduced can search in different directions of the search space and increase the level of diversification while improving solution quality.

To integrate the use of crossover into the ALNS, we design the partition mechanism into the roulette wheel selection. The partition value can be seen as the crossover probability, $Prob_{cross}$. As described in Section 4.4.2, one reason to configure the partition is that the crossover moves a large number of requests in the route(s) at a time, while the ALNS can remove the range of number from small to large of misplaced requests and usually improve the solutions. Often, the replacement of long routes, implemented by the IVX, results in non-improving solutions. The partition mechanism protects the crossover's adaptive scores from being overtaken by ALNS's operators. The proposed crossover can be differentiated by measures corresponding to the route quality. From the surveyed literature and the corresponding objective function of this problem, we chose four rules of route selection to experiment on, according to Table 5.4.

Rule	First Measure	Second Measure	Notation
1	The number of locations	total distance	Req. Dis.
2	The number of locations	total distance and time	Req. Dis. & Time
3	Avg. distance	-	Avg. Dis.
4	Avg. time and distance	-	Avg. Dis. & Time

Table 5.4: Crossover rules to measure route quality

In Rule 1 and 2, the first measure is determined by the number of locations and ties are broken by the second measure. Rule 3 and 4 take either route distance or time and divide them by the number of locations. To illustrate, the average distance is the average value of distance separating the consecutive locations in a route. For MA-ALNS, we designed the crossover to be adaptive according to the accumulated performance of each rule by adopting the original ALNS's adaptive mechanism. Also, the randomness parameters of crossover, p_c, is empirically set and used as in the original ALNS.

In order to recombine one solution with another, we again use the binary tournament

selection for mating, according to Ombuki and Hanshar (2009). In the following experiments, we set $Prob_{T,1^{nd}} = 0.8$ and $Prob_{cross}= 0.5$. The number of routes to remove $[r_1, r_2]$ is [1,2].

Design	Measure	$Prob_{T,2^{nd}}$	p_c	$Gap_{av}(\%)$	$Gap_b(\%)$	$Avg.Time(\%)$	$Obj.Fn$
MA-ALNS1	1	0.0	6	0.91	0.94	-0.51	1.34
MA-ALNS2	2	0.0	6	0.71	0.92	-0.52	1.11
MA-ALNS3	3	0.0	6	0.72	1.03	-0.49	1.26
MA-ALNS4	4	0.0	6	0.64	0.87	-0.49	1.02
MA-ALNS5	1+ 2	0.0	6	0.84	0.93	-0.50	1.27
MA-ALNS6	1+ 3	0.0	6	0.79	0.95	-0.50	1.24
MA-ALNS7	3 +4	0.0	6	0.52	0.71	-0.48	0.75
MA-ALNS8	2 + 4	0.0	6	0.60	0.69	-0.43	0.86
MA-ALNS9	1+2+3+4	0.0	6	0.81	1.03	-0.46	1.38
MA-ALNS10	1+2+3+4	0.0	1	1.02	1.06	-0.45	1.63
MA-ALNS11	1+2+3+4	0.8	6	0.59	0.61	-0.45	0.75

Table 5.5: Experiments on design and parameters for MA-ALNS

We give the concluding remarks as follows:

- By comparing from MA-ALNS 1 to 9, the combination of Avg. Dis. +Avg. Dis. & Time used in MA-ALNS 7 shows the best result in terms of $Obj\ Fn$.
- By comparing MA-ALNS 9 and 10, using $p_c = 6$ and $p_c = 1$ respectively, some greediness of route selection is useful to the solution quality
- By comparing MA-ALNS 9 and 11, using $Prob_{T,2^{nd}}$ can improve the solution quality.

From the experiments, we observed that when the Threshold becomes too small. The use of crossover is rarely accepted due to the replacement of large routes and the implicit mutation. We therefore resort to the only use of the original removal operators from the ALNS, which use a fine-grain search in terms of the number of requests to remove, q, in the next section.

5.3 Hybridisation between Population- and Single- Solution Approaches

In this section, we propose a hybrid metaheuristic based on the experience presented in Section 5.1 and Section 5.2. We again repeat the selection + recombination and selection+improvement. The concept of this design is to combine strengths and counteract limitations between population-based and single solution approaches.

We view the strength of the population-based approach, such as MALNS and MA-ALNS, in terms of diversification due to the use of diverse solutions and gathering good information from different solutions. However, the population-based approach requires good local search operators to refine the search.

We view the strength of ALNS in terms of the use of several large neighbourhood operators and its adaptive mechanism. However, the single solution may locate a false peak or require a more diversified mechanism to search thoroughly, but still with the limited computational time.

Therefore, we synergise the use of population, tournament selection, crossover, large neighbourhood search, and its adaptive mechanism. It is important to note that the number of requests to remove also determines the large neighbourhood search whether the operator intensifies or diversifies the search. The wide range of q enables the large neighbourhood search to act as a local search and heavy mutation operators when using together with the crossover.

5.3.1 Adaptive Memetic Large Neighbourhood Search (AMLNS)

From the observation in the MA-ALNS 7 design, we attempt to modify the MA-ALNS 7 so that:

Firstly, the best solution of the population, at the cut-off point, should continue the search and stop the rest of the solutions due to the limited computational time for this

single-thread computing.

Secondly, the IVX removes a large number of requests in the good routes (long routes or well-sequenced locations). Moreover, its implicit mutation when replacing the routes potentially occurs. Therefore, the IVX is good at diversifying the search while sometimes contributes the improvement due to the use of route quality measure. However, when the Threshold is small, the large change produced by crossover is rarely accepted by the Threshold. We should then resort to the use of original ALNS's operators containing both diversification and intensification mechanisms with smaller q.

We coin this modified design as the Adaptive Memetic Large Neighbourhood Search (AMLNS). The AMLNS hybridises the operators between the single-solution and population-based approaches into the same search to take advantage of their strengths. Moreover, we pay attention to details about the synergy effect of this hybridisation. We compare the search between MA-ALNS7 and AMLNS1 in Figure 5.3.

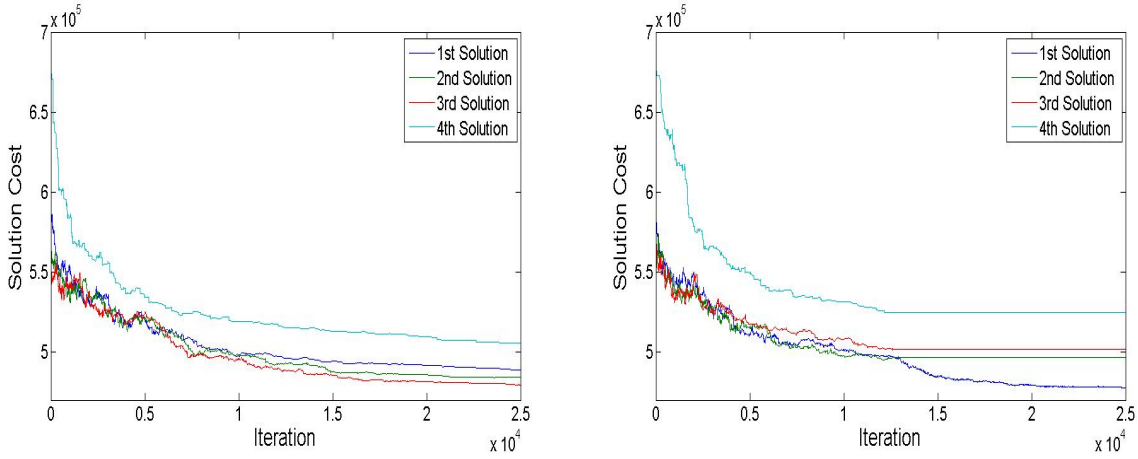


Figure 5.3: Search Trajectories of Solutions by MA-ALNS 7 (left) and AMLNS (right) for Problem 500L

In Figure 5.3 (left), after $10,000^{th}$ iteration, it is quite clear that the 3^{rd} solution is the best solution in terms of solution quality. When tournament selection is used, the better solution is frequently selected. Then, tournament selection continues to frequently sample this solution, while the others are less carried out. However, the other

solutions somehow keep improving but never overtake the best solution. Therefore, it seems that the search through the other solutions, apart from the best solution, is fruitless. In order to evaluate their overall efficiency, Table 5.6 shows the computational results between MA-ALNS 7 and AMLNS1.

Design	$Gap_{av}(\%)$	$Gap_b(\%)$	$Avg.Time(\%)$	$Obj.Fn$
MA-ALNS 7	0.52	0.71	-0.48	0.75
AMLNS1	0.22	0.09	-0.30	0.01

Table 5.6: Computational results for MA-ALNS 7 and AMLNS 1 designs

In Table 5.6, AMLNS1 shows superior results over the designs of MA-ALNS and MALNS according to $Obj.Fn$. We attempt to apply a single parameter set to all problem sizes and types. From the experiments on population size, the constant size of population of 10 gives promising results for all problem sizes. In terms of the number of routes to remove $[r_1, r_2] = [1, 2]$ routes are applied.

From Section 5.1 to 5.3, we showed the development process from the algorithms in this Chapter. In this study, even though the $Obj.Fn$ of AMLNS1 may be worse than those of some designs in the single-solution approach, in this study, we attempt to find an appropriate metaheuristic for parallelising the ALNS, instead of using SA as in Ropke (2009b). We also seek the large improvement from the hybridisation between the single-solution and the population-based approaches in term of solution quality and computational time. Among the population-based approaches, the AMLNS1 is one of the most promising approaches. We therefore select the AMLNS1 for further parameter fine-tuning in Section 4.6.2.

Chapter 6

An Integrated Truck and Semi-trailer Routing Problem: A Practical Multi-Depot Pickup and Delivery Problem for Road Freight Transport

6.1 Introduction

An automated routing and scheduling software is one of the most important optimisation tools in the fast changing and competitive environments of Freight Forwarders, Logistics Service Providers (LSPs) and Third-Party Logistics Providers (3PLs). In order for a 3PL to gain a competitive advantage in terms of cost leadership, an efficient optimisation tool is required. The problem in this Chapter is inspired by a real-life routing problem of road freight transport in Thailand. A medium-sized third-party logistics provider is analysed to understand and represent an illustration of current practices in the industry. The problem can be viewed as a variant of the MD-PDPTW-SR presented in Chapter 4 with additional constraints and characteristics incorporated. In this Chapter, the metaheuristics used to solve the MD-PDPTW-SR in Chapter 4, the

ALNS and the AMLNS, will be applied to this problem. Problem overview, description and formulation will be presented. Finally, a description of the algorithms and their computational results will be shown.

6.2 Problem Overview

A 3PL for freight transportation in Thailand is analysed in order to understand the current practice of its operational planning. The company selected specialises in transporting heavy weight, containerised, and large-volume goods. The complexity of routing and scheduling arises from the number of locations served, the constraints and from the number of service providers involved. Full-truck load and less-than-truck load transportation are provided to customers from pickup to their corresponding delivery locations.



Figure 6.1: (Left) Truck with 3 axles (Right) Semi-trailers with 3 axles

In this study, a vehicle is defined as the combination of truck and semi-trailer, according to Figure 6.1 (left) and Figure 6.1 (right) respectively. The articulated truck can be equipped with different semi-trailers allowing the transport of various types of shipments. Therefore, vehicles, truck with interchangeable bodies, can have variable capacities. There are two types of trucks namely 'normal' trucks and a 'Genset', those

equipped with a power generator. In this study, for each type of truck, there are two numbers of axles: two and three. Figure 6.1 (left) shows the truck with three axles or ten tyres. The truck with two axles typically contains six tyres. The trucks with a different number of axles usually have not only a different performance outcome, but also a different weight.

In this type of problem, the number of semi-trailers typically exceeds the number of trucks in order to provide flexibility in serving various types of products. There are two types of semi-trailers: platform and skeleton. Figure 6.1 (right) shows the skeleton semi-trailer with three axles. In this study, each trailer type has two or three axles.

The facilities include depot, port, intermodal facilities etc. Each of customers/facilities also imposes time windows constraints. Waiting occurs when a truck arrives at a customer location before the earliest time window. The truck cannot arrive after the latest time window of the requests. Moreover, the availability of the truck is imposed by temporal availability.



Figure 6.2: (Left) 20ft Reefer container (special request for truck) (Right) 20ft Dry container

There are also special requests that can only be served by particular types of trucks. Figure 6.2 (left) is the reefer container and Figure 6.2 (right) is the dry container. In this study, the refrigerated container or reefer container is the special request for truck, since the truck requires the power generator or Genset to supply electricity to reefer container, while dry containers can be served by any type of truck. Therefore, a dry

container is considered a normal request for a truck.



Figure 6.3: (Left) Normal request for semi-trailer (Right) Special request for semi-trailer

Figure 6.3 (left) shows the articulated truck using 40ft skeleton semi-trailer with three axles carrying two 20ft dry containers. Some types of goods or freight are required to be transported by platform trailer such as palletised goods or non-standardised shipments. Figure 6.3 (right) demonstrates the articulated truck using 40ft platform semi-trailer with two axles transporting non-standardised shipments. The heavy package freight, non-standardised shipments and palletised goods requires platform trailers. These kinds of goods are therefore considered the special request for trailers. Since, the skeleton trailer, according to Figure 6.1 (right), has no flatbed to support the goods, unlike the platform trailer, the transportation of empty or laden containers is, therefore, the normal request for trailers as they both can be served by the platform and skeleton trailers. The right assignment of semi-trailers to trucks and to requests can reduce the operating costs, as different semi-trailers have different weights and, thus, variable cost rates.

The truck is capacitated by its performance on pulling the weight of goods and a semi-trailer. The semi-trailer is capacitated by volume. Moreover, the total weight of a truck, semi-trailer, and goods is capacitated by the road regulations. The different combinations between truck axles and semi-trailer axles also affect different weight restriction, as shown in Figure 6.4.

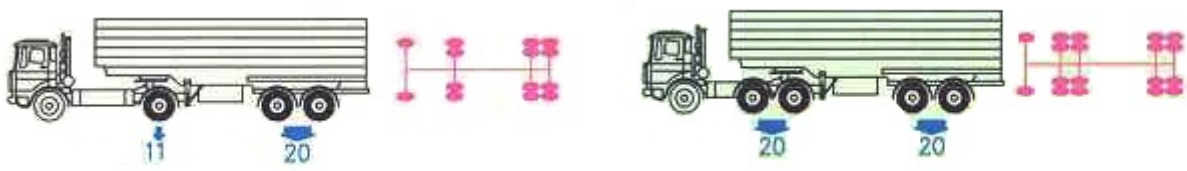


Figure 6.4: (Left) The combination of 2-axles truck and 2-axles trailer (Right) the combination between 3-axles truck and 2-axles trailer

In Figure 6.4 (left), the 2-axle truck and 2-axle trailer have a total weight restriction of 35 tons according to Thai road regulations. In Figure 6.4 (right), the 3-axle truck and 2-axle trailer have a total weight restriction of 45 tons. The 3-axle truck and 3-axle trailer have total weight restriction of 50.5 tons. The 2-axle truck and 3-axle trailer has total weight restriction of 40.5 tons.

The company selected for this study usually uses its own fleet of trucks and has one depot to serve a large geographical coverage area of customers. However, the company sometimes had to outsource some requests to external carriers or subcontractors due to the following reasons: the fluctuating demand of the transportation market, a limited number of available vehicles, low revenue of requests and urgency of service. There are also different types of sub-contraction. In this study, two typical types of sub-contraction for freight transportation are investigated. First, a subcontractor is directly called and the cost is paid per request. The sub-contraction cost of a request is deducted from the price by a pre-determined percentage. This type of sub-contraction is so-called sub-contraction on a request basis. Second, owing to the online logistics market that encourages the reduction of empty miles or the so-called backhauling system, a scheduler can subcontract a request subject to the availability of backhaul trips. sub-contraction costs of this type depend upon the agreed tariff rate per travel unit and the distance travelled. This type of sub-contraction cost is usually cheaper from the backhauling system approach than from the request basis. Since, the vehicle has high possibility of having empty trip and the subcontractor always seeks to fulfil transportation requests.

As a result, the subcontractor has low bargaining power and often accepts lower rate, which is still better than empty trucks travelled.

In terms of costing, the fixed and variable costs of each truck and semi-trailer are varied. The costs of own fleet and subcontractor are also different. Moreover, the different types of subcontractors have different ways of calculating the costs. The efficient routing and assignment of requests to their own fleet or subcontractors are essential for operational planning in terms of cost reduction. In order to investigate the problem, information was gathered from interviews with front-line staff, interviews of top management and from historical data from the case-study company.

6.3 State-of-the-art Reviews of Related Problems

Real-life routing problems become more sophisticated. Hasle and Kloster (2007) pointed out that rich VRP models captures several problem characteristics from industrial setting. The efficiency of a routing tool is highly dependent on the quality of its solver. First, the applicability and flexibility of the tool are determined by the richness of the underlying model. Second, the algorithmic performance depends on the solution quality and computational time. The authors discussed the generic VRP solver named SPIDER, a concrete software product of SINTEF. The SPIDER can handle a large number of constraints and complexities such as VRPTW, Fleet Size and Mix VRP (FSMVRP), Multi-depot VRP (MDVRP), Pickup and Delivery Problem (PDP), Periodic (PVRP), Inventory Routing (IRP), Real-Time Time-dependent VRP(RTTDVRP), Multiple Time Windows VRP (MTWVRP), Compatibility Constraints, Dynamic VRP (DVRP) etc. The SPIDER heuristic approach is based on Local Search: Construction, Tour Depletion and Iterative Improvement. A hybrid of Variable Neighbourhood Descent (VND) and Iterated Local Search (ILS) is used as the overall strategy. The computational results show that SPIDER is robust and efficient over a large variety of

VRPs.

Goel and Gruhn (2008) studied a rich vehicle routing problem incorporating various complexities found in real-life applications. The authors refer the problem to the General Vehicle Routing Problem (GVRP), a combined load acceptance and routing problem which generalises the well-known VRPs. The real-life requirements of this problem include time windows, heterogeneous fleet with different travel times, travel costs, capacity, multi-dimensional capacity constraints, order/vehicle compatibility constraints, orders with multiple pickup, delivery and service locations, different start and end locations for vehicles and route restrictions for vehicles. The authors were motivated to model GVRP by a practical problem arising in air-cargo transport or Road Feeder Service (RFS). The Reduced Variable Neighbourhood Search (RVNS) and Large Neighbourhood Search (LNS) are applied as the meta-heuristics strategy. Good computational results were obtained.

Wen (2010) dedicated his Ph.D. thesis to investigating rich VRPs namely, VRPs with cross-docking options, Dynamic VRPs (DVRP) and integrated vehicle routing and driver scheduling problems. The rich VRP models are formulated from real-life scenarios. The meta-heuristics namely, Tabu Search, Three-Phase Rolling Horizon Heuristics and Multi-level Variable Neighbourhood Search, were used to solve these problems efficiently.

In this Chapter, a third-party logistics provider for freight transportation is investigated. The problem simultaneously considers several real-life constraints and characteristics as a complex routing problem, i.e. a Multi-Depot Heterogeneous Fleet Pickup and Delivery Problem with Time Windows, Special Requests for truck and semi-trailer, Multi-dimensional Capacity Constraints, Assignment of semi-trailers, and sub-contraction. The problem can be broken down into sub-problems. To the best of my knowledge, some of these sub-problems receive little attention. Moreover, the integration of these sub-problems has not been tackled in the literature. The problem characteristics and methodologies are surveyed. Due to the computational complexity,

meta-heuristics are also considered for implementation in competitive business environments.

6.3.1 Multi-Depot Pickup and Delivery Problem with Time Windows and Special Request

The company faces a problem of how to serve its customers: either using their own fleet or subcontractors. The multiple-depot problem arises from this characteristic. Time windows are imposed. Some customers also require a particular type of vehicles. Therefore, this set of constraints and characteristics is modelled as the Multi-depot Pickup and Delivery Problem with Time Windows and Special Requests (MD-PDPTW-SR) as investigated in Chapter 4. Ropke and Pisinger (2006) presented the mathematical formulation for solving PDPTW. The authors applied an Adaptive Large Neighbourhood Search (ALNS) to solve these problems efficiently. The authors also generated the new test instances for MD-PDPTW-SR and reported the computational results. Pisinger and Ropke (2007) modified the ALNS of Ropke and Pisinger (2006) by introducing a larger number of removal heuristics, and adjusting parameters. The computational results of MD-PDPTW-SR were updated in www.diku.dk/~sropke. Chapter 4 of this thesis proposed the Adaptive Memetic Large Neighbourhood Search (AMLNS) for solving the MD-PDPTW-SR efficiently and competitive to the ALNS.

6.3.2 Multi-dimensional Capacity Constraints

In this problem, the heterogeneous fleet of vehicles has a simultaneously limited capacity: both weight and volume are restricted. This situation arises due to heavy weight of goods transported and the road regulations. In the terminology of VRPs, this problem is known as multi-dimensional capacity constraints. Confessore et al. (2008) considered

a VRP with heterogeneous fleet with different capacity and multi-dimensional capacity constraints. The authors applied an evolutionary algorithm based on the combination of a genetic algorithm and local search heuristics. They investigated the performance of the implemented algorithm in large-scale retail and waste collection industries.

6.3.3 Truck and trailer routing problem

In the Truck and Trailer Routing Problem (TTRP), trailers are said to be used when customers are serviced by a truck pulling a trailer. In addition, due to practical constraints, some customers may only be serviced by a truck. Lin et al. (2009) claimed that the truck and trailer routing problem (TTRP) is computationally more difficult to solve than the vehicle routing problem (VRP). In TTRP, the number of available trucks is typically greater than or equal to the number of available trailers. The authors proposed an SA heuristic for the TTRP and show competitive results to the existing approaches using benchmark instances from the literature.

Derigs et al. (2011) studied the vehicle routing problem with the multiple use of tractors and trailers. The authors solved variants of the TTRP problems by using local search and large neighbourhood search as well as standard metaheuristic control strategies. This approach can solve the standard benchmark instances effectively.

Lee et al. (2003) investigated a local logistic company that provides a transportation service for moving empty and laden containers within Singapore. The authors presented a vehicle capacity planning system (VCPS). There are three major types of container movement: importation, exportation and empty container movement. In the study of Lee et al. (2003), the company holds a large number of semi-trailers, and the ratio of trucks to semi-trailers can be as high as 1: 9. Therefore, it is assumed that the right type of semi-trailers is always available at every exchange point. In other words, the semi-trailer type feasibility constraints are not considered in the model. The problem

is modelled as VRPTW and solved by Tabu Search (TS). The authors devised some new rules on how to assign jobs for outsourcing.

Tan et al. (2006) extended the model of Lee et al. (2003) with the detailed manoeuvring of semi-trailers in a routing plan. The authors presented a transportation solution for trucks and semi-trailers vehicle routing problem (TTVRP) containing multiple objectives and constraints. In TTVRP, the semi-trailers are resources with certain limitations that are similar to real world scenarios and the allocation of semi-trailers in different locations could affect the routing planes. Unlike TTRP, the TTVRP requires the trucks to visit semi-trailer exchange points for picking up the correct semi-trailer types depending on the jobs to be served. A hybrid multi-objective evolutionary algorithm (HMOEA) featured with specialised genetic operators, variable-length representation and a local search heuristic is applied to search for the Pareto optimal routing solutions for the TTVRP. The route-exchange crossover allows a good sequence of routes or genes in a chromosome to be shared with other chromosomes in the population. The operation consists of two steps: (1) two random routes are selected and swapped between two chromosomes; (2) the route with the highest number of tasks from each chromosome is swapped. The number of semi-trailers must be up-to-date and a routing plan must include supplementary information of semi-trailer availability at every semi-trailer exchange point. The computational results have shown that the HMOEA is effective in solving multi-objective combinatorial optimization problems.

6.3.4 sub-contraction

Lee et al. (2003) investigated the VCPS as stated in Section 6.3.3. Outsourcing is considered due to the limited capacity of its own fleet of vehicles. The authors assume that it is often impossible for the company to wait for all the orders to come in before contacting other companies to outsource jobs. They therefore devised some rules to

guide the planner in how many jobs should be outsourced and how to select jobs for outsourcing. A Tabu search heuristic has been chosen to solve the problem. The proposed new rules can save a total cost of up to 8.14%.

Tan et al. (2006) studied the TTVRP as described in Section 6.3.3. In terms of cost-related issues, there is no hard rule to specify whether the cost for the internal fleet is cheaper than the outsource fleet or vice versa, i.e. the cost merely depend on the type of jobs to be served. In the HMOEA, any task that is not assigned to a route is considered for outsourcing. All the outsourced tasks are contained in a list. After applying the operators, the approach checks their feasibility in the routes so that any tasks violating the constraints are deleted and later considered as outsourced tasks.

Goel (2008) stated that after an order is received, the carrier has to decide whether to confirm or reject it. The load acceptance problem is the problem of effectively choosing a subset of transportation requests to confirm; it has significant effect on the profitability and efficiency of the carrier's operations. A decision on the load acceptance problem is based on the cost estimate of providing service and on the expectation about future requests. In exceptional cases, some of the confirmed orders can neither be assigned to self-operated vehicles nor subcontracted by external carriers. If the previously confirmed orders have to be rejected or postponed, a penalty fee has to be paid to the shipper. Once an order is subcontracted, the actual transportation process is under control of that carrier. The cost of certain subcontracts may be lower than the company's costs of providing the service themselves. The author modelled the rich vehicle routing problem with subcontractors. It is assumed that external carriers can be employed at double costs of the cheapest vehicle capable of transporting the shipments, i.e. the cost for the transport plus the costs for an empty trip.

Krajewska and Kopfer (2009) investigated a medium-sized freight forwarding company using its own vehicles and external carriers for its operations in several regions of Germany. The authors determined that only about 30% of the requests were fulfilled by the company's own fleet. They pointed out that, typically, planning decisions are

made hierarchically by the dispatchers. At first, the most attractive requests are assigned to their own fleet based on the profit contribution. Next, the requests which are not planned to be performed by self-fulfilment are forwarded to subcontractors. The authors modelled particular types of sub-contraction, namely tour basis, daily basis and freight consolidation. The shipment is calculated on the tour basis using an agreed tariff rate per travel unit and the length of the transferred tour. When using an external carrier is based on a daily basis. The maximal tour length cannot exceed the pre-determined amount of distance and time. The third sub-contraction type consists of forwarding some requests to independent freight carriers. The payment is determined by freight consolidation as a function of load and distance. In practice, there are further, not cost-oriented aspects of deciding for and against using a company's own fleet, e.g. service aspects and flexibility are important arguments to use an own fleet.

6.3.5 Capacity-Driven Activity Based Costing (CDABC)

One reason for incorporating CDABC is to determine the cost of each order and then to simulate its price. This is because the amount of money paid to subcontractors on a request basis is calculated using the price.

Vehicle Routing Problems (VRP) are daily operational planning problems. The VRP objective function depends on the problem's characteristics such as a minimal number of vehicles. Examples of objective function are shown in Figure 2.1. Also, it can be comprised of fixed cost per day and variable cost per travel unit. For other variants of VRPs such as the long-haul routing problem, the fixed costs per day and variable costs still apply. From the management accounting perspective, according to Drury (2005); Atrill and McLaney (2009), fixed cost remains constant over a wide range of activities for a specified time period, while variable costs vary with the volume of activity. The fixed cost is one that recurs regardless of utilisation, while the variable

cost recurs only when the equipment is used. The fixed and variable costs make up the full cost. However, the full cost can also be viewed according to direct and indirect costs. The direct costs are those costs that can be specifically and exclusively identified with a particular cost object, while the indirect cost cannot be identified exclusively with a given cost object. The direct and indirect costs are used for calculating the full cost of each order, as a basis of pricing.

The complexity of management accounting arises when the indirect costs or overheads are apportioned to individual cost units. The widely used method to apportion costs to a cost unit is Activity-Based Costing (ABC), proposed by Cooper and Kaplan (1988). Atrill and McLaney (2009) described that the ABC aims to overcome the problem of tracing the cost of all support activities particular products and services. The factor that causes a change in the costs of each support activity is the cost-drivers. They have a cause-and-effect relationship with activity costs and are used as a basis for attaching activity costs to a particular product or service.

Atrill and McLaney (2009) showed the relationship between the direct, indirect, variable and fixed costs of a particular job in Figure 6.1.

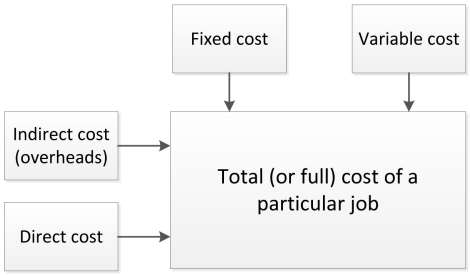


Table 6.1: The relationship between direct, indirect, variable and fixed costs of a particular job

In Figure 6.1, the total cost is the sum of direct and indirect cost. Also, it is the sum of fixed and variable costs. These two facts are independent of one another. The fixed and variable costs are sometimes applied in the objective function of the VRP. For the problem studied in this Chapter, the relationship between the fixed/variable

cost and the direct/indirect cost is analysed and shown in Appendix D.

Moolman et al. (2010) proposed the use of ABC toward VRP. Kaplan and Anderson (2004); (2007) updated the ABC, considering time as the activity cost driver called Time-Driven Activity-Based Costing (TDABC). The principle for measuring the cost of other capacities remains the same as the time for other cost drivers. Therefore, in more general terms, the authors referred to it as Capacity-Driven Activity Based Costing (CDABC). To elaborate, Kaplan and Anderson (2007) assumed that capacity is measured by the time usually available for people and equipment. They opted for time because it represents the great majority of resources. The authors proposed the Time-Driven Activity-Based Costing (TDABC) to improve the Activity-Based Costing (ABC). TDABC simplifies the costing process of the ABC by eliminating the need to interview and survey employees for allocating resource costs to activities before driving them down to cost objects. The TDABC assigns resource costs directly to the cost objects (orders or products) in two steps. First, it calculates the cost of supplying resource capacity. Second, TDABC uses the capacity cost rate to drive process resource costs to cost objects by estimating the demand for resource capacity (typically time) that each cost object requires. TDABC allows the time estimate to vary on the basis of the specific demands of particular orders or orders from a new customer without an existing record.

From the analysis of management accounting methods and routing problems, in CDABC terminology, one supplying resource is the fleet of vehicles. The cost rate of the vehicle per time unit can be calculated according to the principle of TDABC. The travelled time, which is the demand to be served by a vehicle for each request, can be obtained from a digital map. Then, the full cost of each request, order costing, can be estimated for pricing.

In routing terminology, the function of time or distance is the objective function as the measure of solution quality. The routing cost is calculated from the sum of the product of cost rates per travel unit, travel units, and 0-1 binary variables. The

objective function of the routing problems and TDABC also share some cost rates e.g. variable costs per travel unit.

6.3.6 Pricing

Krajewska and Kopfer (2009) modelled the objective function for self-fulfilment using fixed and variable costs. For decisions on sub-contraction, the author applies tour basis, daily basis, and freight consolidation. Typically, variants of Vehicle Routing Problems do not consider the price of the requests. However, in this study, when requests were subcontracted by a request basis, the price of each request must be known. According to Atrill and McLaney (2009), one simple approach widely used for pricing is called cost-plus pricing. In this approach, an amount of profit is calculated as a percentage of the total cost (full cost), and the proposed price of the service is the sum between the amount of profit and the total cost. The proposed price is then negotiated by customers upon their bargaining powers. The advantage of this method is its simplicity.

Powell (2003) and Goel (2008) described that there are a number of methods which show how transportation requests are priced, for example, static pricing, contract pricing, and spot pricing. Static pricing is the standard prices a carrier demands for moving freight between locations. This method of pricing is not specific to a contract and is set by the carrier in advance. They are generally the highest price a carrier will quote. For contract pricing, prices can be set on a contract basis. Transportation costs have to reflect the possibility of combining the load with other loads. For spot pricing, the price can be negotiated. The transportation request is usually demanded near the time the operation has begun and should achieve the yield required to compensate the cost of the decision. In this study, we consider the static pricing corresponding to the practice of the case-study company.

Sukhotu (2011) applied cost-plus pricing to determine the offered price for heavy-

cargo or freight transportation. In practice, this required profit is often set in relation to the amount of capital invested in the business. Drury (2005) explained that the return on capital employed or the return on investment is calculated by dividing the average annual profits from products from a project into the average investment costs. Therefore, the profit loading on full cost should reflect the business's target profit.

Atrill and McLaney (2009) confirmed that the cost-plus can be used as a basis of negotiating a price in advance, which then becomes the fixed price. They found that cost plus is regarded as important in determining selling prices by most businesses, but many businesses only use it for a small percentage of their total sales. Cost-plus pricing tends to be particularly important in service businesses, where many businesses are quite small.

6.3.7 Gaps in the Literature

1. Ropke and Pisinger (2006) considered the Multi-Depot Pickup and Delivery Problem with Time Windows and Special Requests (MD-PDPTW-SR). In their study, special requests can only be served by a subset of the vehicles. However, in this study, some requests required a particular type of both trucks and semi-trailers. Moreover, the problem considered sub-contraction and multi-dimensional capacity constraints. This Chapter considers a more sophisticated problem than that in Ropke and Pisinger (2006).

2. Lee et al. (2003) and Tan et al. (2006) assume that once the container is picked up, it will be directly sent to the destination location. However, in this study, there were some cases where vehicle capacity is not exceeded, and it is possible to consolidate the laden and empty container in one trip. This results in potential savings. This assumption can be tackled by one-to-one pickup and delivery prob-

lems, unlike as VRPTW considered by Lee et al. (2003) and Tan et al. (2006). Moreover, the weight and variable costs of semi-trailers are different in this study. The assignment of correct semi-trailers contributes the overall routing costs.

3. Krajewska and Kopfer (2009) considered three types of sub-contraction: tour basis, daily basis, and consolidation. However, from the investigation of practices in freight transportation, the subcontractor paid on request basis is widely used. Moreover, the problem considered in this Chapter incorporates the complexity in terms of constraints and characteristics.

4. Derigs et al. (2011) modelled the vehicle routing problem with multiple use of tractors and trailers. The tractor and trailer assignment was considered with their compatibility to requests. The objective function was to minimise the number of required tractors. The problem studied in this Chapter is the Pickup and Delivery Problem, an extension of VRPs. The truck and semi-trailer assignment also influenced the routing cost in the objective function.

5. Goel (2008) considered the generalised VRP that can incorporate several real-life constraints and characteristics. In this study, we further investigate the characteristics of trailer assignment, two types of sub-contraction cost and multi-dimensional capacity constraints.

6. Generally, the objective functions of routing problems contained the cost elements. The total costs in a planning horizon are possibly the sum of the fixed and variable costs, as described in Krajewska et al. (2008). However, when the cost of each request must be determined, it is difficult to accurately estimate the cost of each request from the fixed and variable costs of serving all requests in the planning horizon. Instead, it should be viewed as the sum of direct and indirect cost. Especially in real-life scenarios, the accurate order costing method such as CDABC from management accounting's point of view should be investigated. The inte-

gration of optimisation and management accounting perspectives should enhance the effectiveness of operational planning.

6.3.8 Problem Complexity

The problem studied in this Chapter is considered an extension of the VRP, where several real-life constraints and characteristics are incorporated. Therefore, this is an NP-hard problem. These constraints and characteristics include multiple depots, pickups and deliveries, time windows, special requests for trucks, special requests for semi-trailers, a heterogeneous fleet of vehicles, the assignment of semi-trailers, sub-contraction and multi-dimensional capacity constraints. In addition, where the number of semi-trailers is greater than or equal to the number of trucks, it is required to assign exactly one semi-trailer to each truck in such a way that the objective function is minimised.

Pankratz (2005) mentioned that the PDPTW is a combination of two interdependent sub-problems. On the one hand, of clustering requests and assigning them to a vehicle has to be solved. On the other hand, for each cluster of requests, constraints in each route have to be satisfied.

The problem considered in this study adds further complexities. First, the combination of truck and semi-trailers with the view to minimising the cost of their utilisation can be seen as the assignment problem. Second, when the requests are subcontracted, the requests must be outsourced with a view to minimising the sub-contraction cost. The sub-contraction cost is different to the cost structure of one's own fleet in the objective function. Therefore, vehicles may have different cost structures. Last, the capacity constraints consist of several restrictions namely volume, weight due to truck performance and weight due to road-regulation. These complexities make this problem tightly constrained and difficult to solve.

6.4 Problem Description

6.4.1 Requests

For freight transportation, the operations involve the transportation of intermodal containers, pallets and non-standardised units etc. The intermodal containers are 2.44 metres (8ft) wide and either 6.1 metres (20ft) or 12.2 metres (40ft) long. These intermodal containers can be reefer or normal. As mentioned earlier, this is the special request for the truck. For pallets, the goods are placed on the top of the pallets. The pallets can be placed only on the platform semi-trailer but not the skeleton semi-trailer. Therefore, the transportation of palletised goods can be considered as the special request, while the intermodal containers can use both platform and skeleton semi-trailers. There are several types of intermodal transportation: importation, exportation, empty container movement. For each request, a shipper sends a proposal to logistics service providers. The given information includes pickup and delivery locations, truck and semi-trailer type requirements, weight and volume, total units, time windows and estimated loading and unloading time.

6.4.2 Trucks

According to the problem studied in this Chapter, the types of trucks are classified into truck with and without a power generator set (Genset). The truck with Genset allows the transportation of both refrigerated and normal containers. However, the truck without Genset cannot transport refrigerated containers. Trucks without Genset type fall into two axle categories: (1) two axles; (2) three axles, while the truck with Genset has only one category, three axles. Different combinations of truck axles to each type of semi-trailer allow different road weight limits. It is also possible that different types of trucks or individual trucks are different in terms of weight, fixed costs and variable costs. In this study, it is assumed that one driver is responsible for one truck.

6.4.3 Semi-trailers

There are two types of semi-trailers: skeleton and platform. In this study, each type of semi-trailer has two lengths: 20 and 40 feet. The size of a semi-trailer is compatible with the intermodal containers. In addition, for a 40-feet semi-trailer, there are two categories of axles: (1) two-axle; (2) three-axle. These categories allow the different truck and semi-trailer combination, resulting in a different road weight limit.

6.4.4 Costing

Accurate costing of the objective function is required in order to reflect correct decision making values on variables. The fixed and variable costs of truck and trailers are the main costs of own fleet. The fixed costs of trucks includes the depreciation, insurance, license, taxes etc. The departmental cost, rent and employee salary can be considered as fixed cost and allocated to each truck, since, the income is derived from the truck utilisation. These fixed costs must be covered and can be calculated in the format of monetary units per month. The fixed cost per day is simply obtained from the total fixed costs per month divided by the number of working days. It is also noted that even though a request is served by a subcontractor, the fixed costs still recur but not the variable costs. It is noted that, as the number of vehicles in the own fleet cannot be changed on the operational planning level, these costs do not influence the short-term planning process. However, as we design the algorithm for not only solving the operational planning but also tactical planning for the fleet composition and strategic planning for alternative depot configurations, the block of fixed costs remains importance. Therefore, these costs are included in our further assumptions for cost modeling.

One significant difference between the transportation of heavy goods and the mall package freight is the amount of fuel used when serving a request. The amount of fuel used for heavy goods significantly depends on both distance and weight. Thus, the

variable cost must take the fuel consumption of the truck and the requests in terms of distance and weight into account. The fuel consumption is defined as the ratio of litre per ton per km. The difference between distance and load status affects the amount of fuel used along the paths in each route. Typically, in freight transportation, the fuel is approximately accounted 30-60 per cent of the total costs. Therefore, the cost of fuel consumption is one of the main cost elements to be minimised. The costs of tyres and maintenance, and the driver's wage per trip are elements of the variable costs. For semi-trailers, fixed costs and variable costs are relatively low compared to the trucks' costs. The cost of each semi-trailer type or even individual semi-trailer can be different. However, with the large number of trailers, relative to that of trucks, the correct trailer type, size, weights and costs should be efficiently assigned since the substantial savings in long-run can be gained from efficient assignment of semi-trailers to trucks in daily operational planning.

6.4.5 Pricing

When the sub-contraction on a request basis is introduced, the price of each request must be known. By using the cost-plus pricing, the total cost of a pickup-and-delivery request must be calculated. The CDABC can be used for estimating the full cost of each request. From the analysis of the problem in this study and from the CDABC, the demand elements that each request requires are namely time, distance, weight-distance and pickup-delivery point. The costs of supplying resource capacity in terms of time are for example, the vehicle's depreciation, the employee's salary and the departmental cost. These can be considered fixed costs. The time cost rate can be obtained from the total fixed costs per month divided by the amount of practical working hours. Tyre and maintenance are the resource costs in relation to the travelled distance. The fuel is the resource cost mainly corresponding to the weight and distance. The time and distance are measured from the depot to pick up-and-delivery points and back to the depot because the potential backhauling in advance or consolidation is unknown. The

driver wage is, for example, the resource cost depending on pickup-delivery points. The details of these resource costs and order costing are explained in Appendix D. The total cost of each request is the sum of these major resource costs utilised for serving the request.

In this study, it is assumed that static pricing applies. The price is set by the carrier in advance and not to a specific contract. The price is fixed over a certain period of time unless the costs i.e. petrol price or cost structures change. The fixed and variable costs of each truck and semi-trailer can be different. However, the trucks and semi-trailers can be grouped according to their costs for the matter of simplicity in terms of data collection. In order to determine the price, the average cost of each group of trucks and semi-trailers must be calculated. The selection of a truck and semi-trailer for pricing is the best combination resulting in minimum costbetween a company's own truck and semi-trailer, while all constraints are satisfied.

In this study, cost-plus pricing is used. The offered price is usually negotiated by customers. For the accepted price, the profit margin is randomly generated in the interval of the maximum expected return on investment (ROI_{max}) and the minimum expected return on investment (ROI_{min}). Let o_i be the percentage of full cost for profit of request i. The o_i is selected at random at interval $[ROI_{min}, ROI_{max}]$.

$$Price_i = Cost_i \times \frac{(100 + o_i)}{100}$$

Even though, the consolidation of requests by optimisation result in lower costs, the static price do not take into account the possibility of combining load with other loads. Since, the arrival of requests is unknown and uncertain. Therefore, the requests are not always consolidated.

6.4.6 Subcontractors

In this study, there are two types of sub-contraction, namely sub-contraction on a

request basis and sub-contraction on a tour basis. For the sub-contraction on a request basis, each subcontractor may have a different sub-contraction rate for a request depending on negotiations within its own company. Denote $K^{'}$ the set of subcontractors on request basis. Let k be the subcontractor id where $k \in K^{'}$. The rate, $Y_{i,k}$, refers to the percentage deducted from the price for subcontractor k for request i. If subcontractor k cannot serve request i, the $Y_{i,k}$ is then equal to large negative number or -1000. Therefore, the sub-contraction cost of subcontractor k for request i, $c_{i,k}$, is equal to

$$c_{i,k} = Price_i \times \frac{(100 - Y_{i,k})}{100}$$

In order to capture the practices and generate test instances, $Y_{i,k}$ is selected at random in the interval $[\phi_{min}, \phi_{max}]$ where ϕ_{min} and ϕ_{max} are the acceptable minimum and maximum percentage deducted for sub-contraction.

For the sub-contraction on tour basis, the cost is the product of travelled distance and cost rate per distance. The start location and end location of this type of sub-contraction is usually different. The constraints are also imposed as the company's own fleet of vehicles.

The cost of using a subcontractor on a request basis is usually higher than using a company's own fleet. However, it is also possible that the cost of using their own fleet is higher than that of the subcontractors. For example, some requests are located far away from their own depot(s) but close to the subcontractor hired by tour basis. The sub-contraction costs on request and tour basis are mutually agreed earlier. The subcontractors are called only when required. This variable cost of truck from sub-contraction by tour basis is always higher than that of own fleet, as observed from the current practices, because it must cover a part of fixed costs of the subcontractor. However, the total sub-contraction cost on tour basis is generally lower than the total cost of using own fleet due to being backhaul trips.

The current state of the vehicles from subcontractor on a request basis is unknown.

Therefore, it is assumed that the start location of vehicles for this sub-contraction type is from the subcontractors' depots. The vehicle's capacity of this sub-contraction type depends on the historical information of subcontractors including the fleet size and mix. If sub-contraction is required and subcontractors are communicated, the status of subcontractors' vehicles can be updated and re-optimised.

For sub-contraction on a tour basis, owing to the backhauling system, the current state and information of vehicles used are given. The information about the subcontractors' vehicles can be imported to the optimisation engine. A typical process of routing with sub-contraction is shown in Figure 6.5.

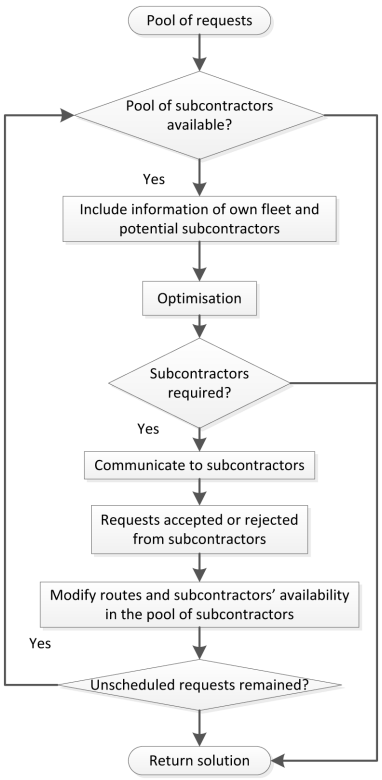


Figure 6.5: Iterative sub-contraction Process

Figure 6.5 is the typical scenario of sub-contraction. It is assumed that the pool of subcontractors at the beginning period is always available. After the optimisation process, subcontractors are approached if their services are needed, and their availability is confirmed. If no unserved requests exist, the process is terminated. Otherwise, with

the existence of unserved requests and no available subcontractors, the penalty cost for the unserved requests must be paid to their corresponding shippers.

Chapter 7

Solution Methods for the Integrated Truck and Semi-trailer Routing Problem

7.1 Problem Formulation

The Integrated Truck and Semi-trailer Routing Problem (ITSRP) investigated in this Chapter consists of several sub-problems. It extends the MD-PDPTW-SR of Ropke and Pisinger (2006) by several assumptions, constraints and characteristics. Hereafter, the problem is called the Integrated Truck and Semi-trailer Routing Problem (ITSRP). The ITSRP forms the core of logistics planning and hence of practical interest. The entire problem considers three fulfilment modes, namely, self-fulfilment, sub-contraction on a request basis and sub-contraction on tour basis. All sub-problems have the same structure in terms of constraints. In addition, the self-fulfilment requires the assignment of semi-trailers. All sub-problems differ in terms of cost structures in the objective function. We formulated the ITSRP as the Mixed Integer Linear Program (MILP).

7.1.1 Assumptions

All transportation requests have their own pickup and corresponding delivery locations. The quantity of goods and the location of the pickup and delivery are known in advance. Both full-truckload and less-than truck load shipments are carried out in the operation. The self-fulfilment of requests is to service all requests using its own fleet. The own fleet of vehicles is heterogeneous. The number and types of trucks and semi-trailers in the own fleet are known in advance. Due to the limited number of vehicles and the large penalty cost of unserved requests, the sub-contraction of requests is sometimes required. The own fleet of vehicle must depart from and return to the own depot. The vehicles' locations of subcontractors paid by request basis are unknown and assumed to depart from and return to a depot. For those subcontractors paid on a tour basis, the vehicles may start and end at different locations. At a company's own depot, the number of semi-trailers is known, and semi-trailers are available for interchanging with base trucks. The capacity restriction for a truck with the assigned semi-trailer is imposed by road regulations. However, information concerning semi-trailers of subcontractors paid on a request basis is unknown. Therefore, we assume that the capacity of each truck and semi-trailer is large. After communicating, the subcontractor will decide whether their truck and semi-trailers are available or not and reply.

Nevertheless, the information about semi-trailers of subcontractors paid on a tour basis is known. For the company's own fleet, the number, weight, and capacity restrictions of trucks and trailers are known. Earliest and latest time windows are also known in both pickup and delivery locations. In case of early arrival at the location, the vehicle has to wait until the earliest time window. A truck might not be able to serve all requests; for example, a request might require that the truck has a power generator for a reefer container. Also, some goods such as palletised goods require a platform semi-trailer. These requests are called special requests. The fleets of trucks and trailers are heterogeneous in terms of ownership, start and end locations, truck types, maxi-

mum weight capacities, truck weight, fixed costs per day, capacity cost rate in terms of distance, fuel consumption per truck and request compatibility. The fleets of semi-trailers are also heterogeneous in terms of ownership, semi-trailer types, semi-trailer axle types, weight, fixed costs per day, capacity cost rates in terms of distance and request compatibility. All requests must be served. The one-day operational planning horizon is investigated.

7.1.2 Notations

n	number of transportation requests
K	set of all vehicles
K_o	set of own trucks
$K^{'}$	set of subcontractors' vehicles on request basis
$K^{''}$	set of subcontractors' vehicles on tour basis
m	number of vehicles, $m =\mid K \mid$
P	set of pickup nodes, $P = \{1, \ldots, n\}$
D	set of delivery nodes, $D = \{n+1, \ldots, 2n\}$
l_i	volume demand/supply at vertex i: pickup vertices are associated with a positive value, delivery vertices with a negative value; at the start depot and end depot the demand/supply is zero.
$l_i^{'}$	weight demand/supply at vertex i: pickup vertices are associated with a positive value, delivery vertices with a negative value; at the start depot and end depot the demand/supply is zero.
a_i	earliest time to begin service at vertex i

b_i	latest time to begin service at vertex i
s_i	service duration at vertex i
K_i	set of vehicles that are able to serve request i,
P_k	set of pickups that can be served by truck k;$P_k \subseteq P$
D_k	set of deliveries that can be served by truck k;$D_k \subseteq D$
τ_k	set of start terminal of truck k;
$\tau_k^{'}$	set of end terminal of truck k;
cf_k	the fixed cost of the company's own truck k
cf_t	the fixed cost of the company's own semi-trailer t
cd_k	the variable cost rate of the company's own trucks k; monetary unit per km
cd_t	the variable cost rate of the company's own semi-trailer t: monetary unit per km
cdw_k	the variable cost rate of the company's own truck k on distance and weight
w_t	the weight of semi-trailer t
u_k	the weight of truck k
g_i	the wage paid per request i
d_{ij}	distance from vertex i to j
t_{ij}	travel time from vertex i to j
$a_{i,t}$	the coefficient matrix of compatible semi-trailers t for request i
$c_{i,k}$	the sub-contraction cost in request basis for request i for vehicle k
$cd_k^{'}$	the variable cost rate of the truck k on tour basis; monetary unit per km

V_t Volume capacity of semi-trailer t

W_k Weight capacity of truck k

E the number of semi-trailer types according to semi-trailer' axles

T set of semi-trailers, T=$\{t \mid 0 \leq t < E\}$.

H the number of truck type depending on axles

χ set of truck types, χ=$\{h \mid 0 \leq h < H\}$

$R_{t,\chi(k)}$ Road weight capacity of semi-trailer t and truck k

x_{ijkt} $=\begin{cases} 1 & , if\ arc\ (i,j)\ is\ traversed\ by\ truck\ k\ with\ semi-trailer\ t \\ 0 & , else \end{cases}$

z_i $=\begin{cases} 1 & , if\ request\ i\ is\ placed\ in\ the\ request\ bank \\ 0 & , else \end{cases}$

S_{ikt} a non-negative number that indicates when truck k with semi-trailer t starts the service at location i

L_{ikt} a non-negative number that indicates space of truck k with semi-trailer t when leaving vertex i

L'_{ikt} a non-negative number that indicates load of truck k with semi-trailer t when leaving vertex i

h_{ijkt} a non-negative number that indicate load of truck k with semi-trailer t if arc (i,j) is traversed

Define $N = P \cup D$ and $N_k = P_k \cup D_k$. Let $\tau_k = 2n + k$ and $\tau'_k = 2n + m + k$, $k \in K$. The ITSRP is modelled on complete graphs $G = (V, A)$ that consists of the nodes

$V = N \cup \{\tau_1, \ldots, \tau_m\} \cup \{\tau_1', \ldots, \tau_m'\}$ and the arcs $A = V \times V$. For each vehicle, due to special requests, we have subgraph $G_k = (V_k, A_k)$ where $V_k = N_k \cup \{\tau_k\} \cup \{\tau_k'\}$ and $A_k = V_k \times V_k$. We assume that distance and time from vertex i to j, are given by d_{ij} and t_{ij}. For each edge $(i, j) \in A$, we assign a distance $d_{ij} \geq 0$ and $t_{ij} \geq 0$. Both d_{ij} and t_{ij} satisfy the triangle inequality. Further, we also assume that $t_{i,n+i} + s_i > 0$ to eliminate sub-tours. Each node $i \in V$ has a time window $[a_i, b_i]$. For one-to-one pickup and delivery problem, for each node $i \in N$, l_i is the amount of goods that must be loaded onto the vehicle at the particular node, $l_i \geq 0$ for $i \in P$ and $l_i = -l_{i-n}$ for $i \in D$. For practical reasons, the arc set can be reduced to $A_k' = \{(i, j) : i, j \in V_k, i \neq \tau_k', j \neq \tau_k, i \neq j, i \in P_k \Rightarrow j \neq \tau_k', i = \tau_k \Rightarrow j \notin D_k, i \in D_k \Rightarrow j \notin P_k \ \ where\ i = j + n\}$. The following six decision variables are used in the mathematical model.

7.1.3 Objective Function

The ITSRP consists of finding a feasible routing and scheduling plan with minimal execution costs. The objective function (C) comprises the costs from self-fulfilment (C_o), the cost from sub-contraction (C_s) the cost from penalty due to unserved requests (C_p).

$$Min\, C = C_o + C_s + C_p \tag{7.1}$$

The costs C_o from self-fulfilment of requests consist of the combination of the fixed costs and the variable costs for both trucks and semi-trailers. A semi-trailer must be attached to a truck for serving a transportation request. The costs are distinguished according to resources, corresponding to the CDABC. There are four cost elements involved in serving requests by own fleet namely, fixed cost ($C_{o,f}$), variable cost on distance ($C_{o,d}$) , variable cost on distance weight ($C_{o,dw}$) and variable cost on requests ($C_{o,r}$) as shown in Equation. 7.2.

$$C_o = C_{o,f} + C_{o,d} + C_{o,dw} + C_{o,r} \tag{7.2}$$

where,

$$C_{o,f} = \sum_{k \in K_o} cf_k + \sum_{t \in T_o} cf_t \tag{7.3}$$

$$C_{o,d} = \sum_{k \in K_o} \sum_{t \in T_o} (cd_k + cd_t) \cdot \sum_{(i,j) \in A'_{k,t}} d_{ij} \cdot x_{ijkt} \tag{7.4}$$

$$C_{o,dw} = \sum_{k \in K_o} cdw_k \cdot \sum_{t \in T_o} \sum_{(i,j) \in A'_{k,t}} d_{ij} \cdot (w_t \cdot x_{ijkt} + h_{ijkt}) \tag{7.5}$$

$$C_{o,r} = \sum_{k \in K_o} \sum_{t \in T_o} \sum_{i \in P_k} (g_i \cdot x_{ijkt}) \tag{7.6}$$

The costs for sub-contraction (C_s) consist of the sum of the sub-contraction on request basis $C_{s,req}$ and tour basis $C_{s,tour}$.

$$C_s = C_{s,req} + C_{s,tour} \tag{7.7}$$

where,

$$C_{s,req} = \sum_{k \in K'} \sum_{t \in T'} \sum_{i \in P} c_{i,k} \cdot x_{ijkt} \tag{7.8}$$

$$C_{s,tour} = \sum_{k \in K''} cd'_k \cdot \sum_{t \in T''} \sum_{(i,j) \in A'_{k,t}} d_{ij} \cdot x_{ijkt} \tag{7.9}$$

The penalty cost (C_p) arises when a request is not served according to service agreement. This cost is relatively high when compared to other cost elements.

$$C_p = \gamma \sum_{i \in P} z_i \tag{7.10}$$

Altogether, the objective function is given by

$$C = \sum_{k \in K_o} cf_k + \sum_{t \in T_o} cf_t + \sum_{k \in K_o} \sum_{t \in T_o} (cd_k + cd_t) \cdot \sum_{(i,j) \in A'_{k,t}} d_{ij} \cdot x_{ijkt}$$

$$+ \sum_{k \in K_o} cdw_k \cdot \sum_{t \in T_o} \sum_{(i,j) \in A'_{k,t}} d_{ij} \cdot (w_t \cdot x_{ijkt} + h_{ijkt}) + \sum_{k \in K_o} \sum_{t \in T_o} \sum_{i \in P_k} (g_i \cdot x_{ijkt})$$

$$+ \sum_{k \in K'} \sum_{t \in T'} \sum_{i \in P} c_{i,k} \cdot x_{ijkt} + \sum_{k \in K''} \sum_{t \in T''} \sum_{(i,j) \in A'_{k,t}} cd'_k \cdot d_{ij} \cdot x_{ijkt} + \gamma \sum_{i \in P} z_i$$

7.1.4 Constraints

The feasibility of the ITSRP is assured if each request is assigned to exactly one fulfilment type and all constraints are satisfied. Constraints are similar to the formulation of PDPTW presented in Desaulniers et al. (2002) and Ropke and Pisinger (2006). However, the flow index of semi-trailers is introduced in the decision variables. In addition, the sets of subcontractors' vehicles are considered.

Subject to

$$\sum_{k \in K} \sum_{t \in T} \sum_{j:(i,j) \in A'_{k,t}} x_{ijkt} + z_i = 1 \quad \forall i \in P \tag{7.11}$$

$$\sum_{j:(i,j) \in A'_{k,t}} x_{ijkt} - \sum_{j:(n+i,j) \in A'_{k,t}} x_{n+i,j,kt} = 0 \quad \forall k \in K, \forall t \in T, \forall i \in P_k \tag{7.12}$$

$$\sum_{j \in P_k \cup \{\tau'_{k,t}\}} x_{\tau_k,j,kt} = 1 \quad \forall k \in K, \forall t \in T \tag{7.13}$$

$$\sum_{i \in D_k \cup \{\tau_{k,t}\}} x_{i,\tau',kt} = 1 \quad \forall k \in K, \forall t \in T \tag{7.14}$$

$$\sum_{i:(i,j) \in A'_{k,t}} x_{ijkt} - \sum_{i:(i,j) \in A'_{k,t}} x_{jikt} = 0 \quad \forall k \in K, \forall t \in T, \forall j \in N_k \tag{7.15}$$

$$x_{ijkt} = 1 \Rightarrow S_{ikt} + s_i + t_{ij} \leq S_{jkt} \quad \forall k \in K, \forall t \in T, \forall (i,j) \in A_k \tag{7.16}$$

$$a_i \leq S_{ikt} \leq b_i \quad \forall k \in K, \forall t \in T, \forall i \in V_k \tag{7.17}$$

$$S_{ikt} \leq S_{n+i,kt} \quad \forall k \in K, \forall t \in T, \forall i \in P_k \tag{7.18}$$

$$x_{ijkt} = 1 \Rightarrow L_{ikt} + l_j \leq L_{jkt} \quad \forall k \in K, \forall t \in T, \forall (i,j) \in A_k \tag{7.19}$$

$$L_{ikt} \leq V_t \quad \forall k \in K, \forall t \in T, \forall i \in V_k \tag{7.20}$$

$$L_{\tau_k kt} = L_{\tau_k kt} = 0 \quad \forall k \in K, \forall t \in T \tag{7.21}$$

$$x_{ijkt} \in \{0,1\} \quad \forall k \in K, \forall t \in T, \forall (i,j) \in A_k \tag{7.22}$$

$$z_i \in \{0,1\} \quad \forall i \in P \tag{7.23}$$

$$S_{ikt} \geq 0 \quad \forall k \in K, \forall t \in T, \forall i \in V_k \tag{7.24}$$

$$L_{ikt} \geq 0 \quad \forall k \in K, \forall t \in T, \forall i \in V_k \tag{7.25}$$

Constraint 7.11 ensures that each pickup location is visited or that the corresponding request is placed in the request bank. Constraint 7.12 ensures that both pickup and corresponding delivery locations must be served by the same vehicle with semi-trailer. Constraints 7.13 and 7.14 ensure that a vehicle with semi-trailer leave every start terminal and enter every end terminal. Note that this does not mean that every vehicle has to be used. A vehicle and semi-trailer may only use the arc (τ_k, τ'_k), i.e. it does not leave the depot. Constraint 7.15 ensures that consecutive paths between τ_k and τ'_k are formed for each $k \in K$. Constraint 7.16 is the sub-tour elimination constraint by

time variables, given that $(t_{ij} + s_i) > 0$. Constraint 7.17 ensures that the time windows of each location is obeyed. Constraint 7.18 ensures that each pickup occurs before the corresponding delivery. Constraint 7.19, 7.20 and 7.21 ensure that a variable is set correctly along the paths and capacity constraints of the semi-trailer in terms of space is respected. Non-linear constraints, given in 7.16 and 7.19, can be linearised using a big M formulation and the computational time speed up. These constraints are similar to the MD-PDPTW-SR in Chapter 4. The following are constraints due to additional characteristics of the ITSRP, as an extension of the MD-PDPTW-SR.

Assignment of semi-trailers

In order to efficiently assign a semi-trailer to a vehicle in the own fleet, the assignment problem then arises. The following constraints must be satisfied.

$$\sum_{t \in T} x_{\tau_k jkt} = 1 \quad \forall k \in K \tag{7.26}$$

$$\sum_{k \in K} x_{\tau_k jkt} \leq 1 \quad \forall t \in T \tag{7.27}$$

Constraint 7.26 states that, for each truck, only one semi-trailer is assigned in the planning horizon. Constraint 7.27 states that, for each semi-trailer, it is assigned to one vehicle or not assigned in the planning horizon.

Multi-dimensional capacity constraints

The MD-PDPTW-SR in Chapter 4 has only one capacity constraint. However, the problem in this study considers real-life capacity constraints which simultaneously involve a number of capacity constraints. In this problem, the space and weight of goods are both considered. The space capacity constraint is already stated in Constraint 7.20. The weight capacity is both restricted according to truck performance and road

regulation. In terms of truck performance, the summed weight between semi-trailer and goods must not exceed the weight capacity in terms of truck performance. In terms of road regulation, the summed weight is combined from trucks, semi-trailers, and goods and must not exceed the road-regulation according to the combination of truck's and trailer's axles. An additional decision variable for load variable, $L^{'}_{ikt}$, are stated in Equation 7.28.

$$L^{'}_{ikt} \geq \ 0 \quad \forall k \in K \cup K^{'}, \forall i \in V_k \tag{7.28}$$

The following constraints must be satisfied.

$$x_{ijkt} = 1 \Rightarrow L^{'}_{ikt} + l^{'}_{j} \leq \ L^{'}_{jkt} \quad \forall k \in K, \forall i : (i,j) \in A^{'}_{k}, \forall t \in T \tag{7.29}$$

$$L^{'}_{ikt} \leq Min\{ \ (R_{t,\chi_{(k)}} - w_k - w_t), (W_k - w_t)\} \quad \forall k \in K, \forall i : (i,j) \in A^{'}_{k}, \forall t \in T \tag{7.30}$$

$$L^{'}_{\tau_k k} = L^{'}_{\tau_k k} = 0 \quad \forall k \in K \tag{7.31}$$

Constraints 7.29, 7.30 and 7.31 ensure that load variable is set correctly along the paths and the capacity constraints of the truck, in terms of weight, are respected. These constraints are similar to those of space capacity in Constaint 7.19, 7.20 and 7.21.

Path Loading

It is to note that $L^{'}_{ikt}$ is the decision variable of location, i. However, in order to estimate the amount of fuel used, the loading between path, i to j, must be used. Therefore, one decision variable is introduced in Equation 7.32 for calculating the amount of fuel used due to the distance and weight between locations.

$$h_{ijkt} \geq \ 0 \quad \forall k \in K, \forall i : (i,j) \in A^{'}_{k}, \forall t \in T \tag{7.32}$$

If the truck k and trailer t traverse from vertex i to j, then the loading status must

be greater than or equal to L'_{ikt} according to Equation 7.33. The h_{ijkt} is modelled also in the objective function in Equation 7.5 for minimisation of fuel use.

$$x_{ijkt} = 1 \Rightarrow L'_{ikt} \leq \ h_{ijkt} \quad \forall k \in K, \forall i \in P_k, \forall t \in T \tag{7.33}$$

Special requests for trailers

For the special requests in terms of truck, the decision variable x_{ijkt} are restricted to the network $A_k = V_k \times V_k$. In addition, in this problem, the special requests for semi-trailers are introduced. Therefore, the decision variable x_{ijkt} is also restricted on the compatibility relationship between vertice i and trailer t.

$$x_{ijkt} \leq a_{it} \quad \forall k \in K, \forall i \in P_k \tag{7.34}$$

In Constraint 7.34, the coefficient matrix a_{it} has value either 0 and 1. If the $a_{i,t}$ is equal to zero, all decision variables x_{ijkt} that traverse from vertex i by trailer t is not used.

7.1.5 Illustrative Example

In order to illustrate the problem, a small example is presented. In this example, it is assumed that 6 requests are considered. For each request, location coordinate, weight, volume, time windows, service time, profit and pickup-delivery expenses are given. For special requests, the compatible trucks and trailers are determined. The pickup and delivery locations of these requests are shown in Figure 7.1.

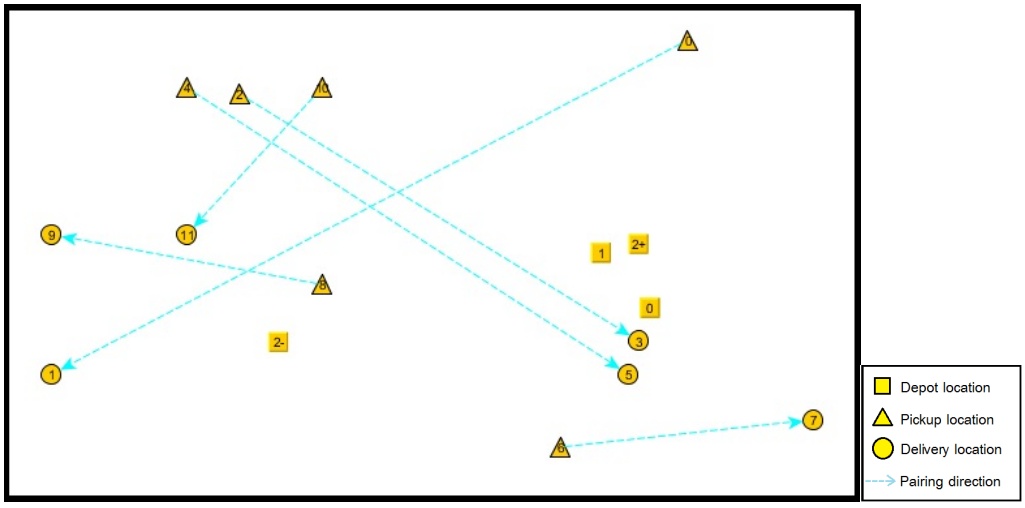


Figure 7.1: Pickup and corresponding delivery locations of 6 requests

From Figure 7.1, the pickup and delivery locations are represented by a triangle and circle, respectively. The locations are paired by dashed line with an arrow. A number of trucks and semi-trailers are located in each depot. The company's own depot, and the depot of the subcontractor paid on request basis are represented by rectangles with the numbers 0 and 1, respectively. The start and end depot of a subcontractor paid by tour basis is represented by rectangles with number 2+ and 2-, since, this type of sub-contraction has a different start and end location. In this example, the truck k_0 and trailer t_0, t_1 are available in the company's own depot. The trucks k_1 and t_2 are provided by the subcontractor paid on a request basis. The truck k_2 and t_3 are available from the subcontractor paid by tour basis.

For all trucks, the following details are given: depot location, truck type id, weight capacity due to truck performance, temporal availability of trucks. The truck type id is used to classify the types of trucks. For the company's own fleet, the following additional details are determined: truck weight, fixed cost, variable cost on distance and variable cost on distance-weight. For the subcontractor for tour basis, the variable cost for distance is also given. The sub-contraction rates for request basis are given by the relationship between request and subcontractor.

For each trailer, type id, weight, fixed cost, variable cost by distance, volume capacity, weight capacity due to road regulation for the combined truck and trailer type are determined.

In order to calculate the total cost of each request, the average values of each truck type id are, namely, truck type, truck's axles, weight, time cost rate, variable costs on distance and variable costs on distance-weight. For each trailer, the average value of each trailer id includes: trailer type, weight capacity due to weight capacity, weight, volume capacity, time cost rate and variable costs depending on distance.

Figure 7.2 shows the optimal solution obtained from CPLEX 12.5 for this illustrative example.

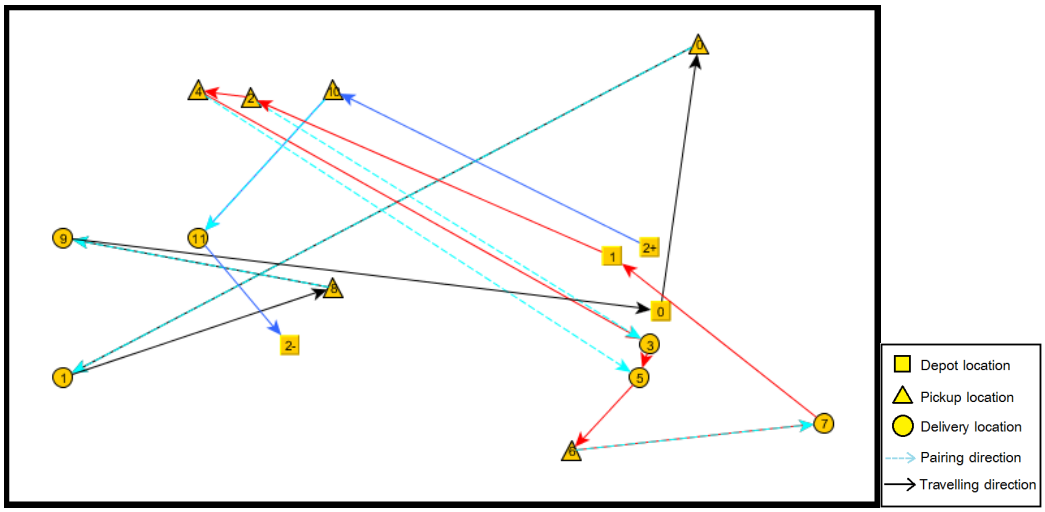


Figure 7.2: Optimal solution of the illustrative example

In Figure 7.2, all available trucks are used. For the company's own fleet, one truck is articulated with one semi-trailer while one semi-trailer is left unused. The pickup and corresponding delivery locations are the even number and that even number plus one, respectively. Different colours of solid lines with arrows indicate the routes of the vehicles. The assumption that all requests are served is satisfied. The optimal routes and costs of the company's own and hired vehicles are shown in Table 7.1.

k	t	Tour	Dist.	Dist.·W.	Objective function	Costs
k_0	t_0	$0, 1, 8, 9$	238	3616	$cf_{k_0} + cf_{t_0} + cf_{t_1} + 238 \cdot (cd_{k_0} + cd_{t_0}) + 3616 \cdot cdw_{k_0} + \Sigma g_i$	3825
k_1	t_2	$2, 4, 3, 5, 6, 7$			$c_{2,k_1} + c_{4,k_1} + c_{6,k_1}$	4317
k_2	t_3	$10, 11$	77		$77 * cd'_{k_2}$	1769

Table 7.1: Routes and costs of own and subcontractor's vehicles

Table 7.1 shows a fulfilment plan for all 6 requests. The cost of the company's own fleet's vehicle depends on both distance and distance-weight. The cost of sub-contraction on a request basis depends on the amount subtracted from the request price. The cost of sub-contraction on a tour basis depends on the distance. The following constraints are satisfied: (1) precedence, (2) request time windows, (3) weight capacity due to road regulation, (4) weight capacity due to truck performance, (5) truck time windows, (6) volume capacity of the trailer, (7) special requests for trucks and (8) special requests for semi-trailers.

We also vary the parameters and constraints of the illustrative example in order to investigate their sensitivities toward objective function and design and to validate proposed meta-heuristics. The following parameters and constraints are used in the experiment: special request due to trucks, special requests due to semi-trailers, cost rates, weight capacity, volume capacity, time windows, price and number of subcontractors.

The MILP of the ITSRP is solved by CPLEX 12.5 with default algorithm. Exact methods can guarantee that the optimisation solution is found if the method is given sufficient time. However, the ITSRP is NP-hard and rapidly changing business environments require the solution within reasonable time frame. While, heuristics are fast but the optimality is not guaranteed. With an industrial sized problem, a scheduler has to select an appropriate optimisation method for solving the problem efficiently. For the problem size studied in this Chapter, a heuristic has to be developed with the view to finding a near-optimal solution in a timely manner. We introduce the ALNS

and AMLNS for this Chapter in order to solve this problem efficiently.

7.2 Adaptive Memetic Large Neighbourhood Search (AMLNS)

The Adaptive Memetic Large Neighbourhood Search (AMLNS) developed in Chapter 4 is extended to solve the ITSRP in this Chapter for two reasons: (i) the MD-PDPTW-SR is a related problem, and (ii) the computational results produced by the AMLNS and presented in Chapter 4 were promising. In order to demonstrate the applicability of AMLNS to other routing problems, the design of the AMLNS is changed at minimum. The operators and adaptive mechanisms are described in the following sections.

7.2.1 Removal Operators

The number of removal operators applied in the ITSRP is identical to those in Chapter 4, with one exception: the Identical Vehicle Crossover (IVX) is altered in terms of route measures. Moreover, unlike the MD-PDPTW-SR, the ITSRP involves the assignment of a trailer. Therefore, a Semi-Trailer Removal Heuristic is introduced to tackle this characteristic.

Identical Vehicle Crossover (IVX)

The IVX procedure is similar to that in Chapter 4. However, the conceptual design of the IVX is to select good routes corresponding to the objective function. Due to the difference of objective function and fulfilment modes, several route measures are experimented on and compared to the existing measures in the original IVX. The average distance and average distance-time are replaced by two new measures. First, the objective value of each route is divided by the number of locations. The lower the average value is, the better each vehicle is routed. For the second measure, the objective

value of the company's own vehicle excludes the fixed cost of trucks and semi-trailers. Then, this objective value is divided by the number of locations. Intuitively, the first measure used in the company's own fleet estimates the average fixed cost and differentiates the routes with a large number of requests. The second measure assumes that the make-or-buy decision of whether or not requests are served by either their own or by a subcontractors' fleet depends on the variable costs of the company's own fleet and the objective value of a subcontractors' fleet. From our observations, the route that has a few requests, but is also efficient in terms of variable costs, can be obtained from the second measure, while the first measure prefers having a large number of requests. These two measures provide some variations to the route selection.

The experiments are conducted by replacing the average distance and average distance-time in all possible ways for each new measure. The computational results of tuning instances show that replacing these two new measures provide promising results, as shown in Table 7.4 and 7.5. This confirms our assumption that the route measure should correspond to the objective function. Since, improving or non-improving moves are evaluated from the objective function. In order to select the ranked non-empty routes, some randomness is introduced as in the original AMLNS in Chapter 4.

In contrast to the MD-PDPTW-SR, the ITSRP models the truck with interchangeable trailers or variable capacities. The concept of an Identical Vehicle Crossover (IVX) is to remove the good sequences of locations to the other solution without feasibility checking. In order to apply the IVX to the ITSRP, the requests and semi-trailers must be transferred together, as the selected identical vehicle for both solutions may be attached with different trailers, and thus capacities. The requests in inserting routes may then violate the capacity constraints.

Once, the requests and semi-trailers are transferred, the repeated requests are repaired similarly to the request repair carried out in Figure 4.2 in the Chapter 4. To illustrate, let $T_{i,j}$ be the semi-trailer of i^{th} truck in the j^{th} solution. Assume that $j = 1$ is selected from the first binary tournament selection or $j = 1$ is called replaced solution,

while, $j = 2$ is selected from the second tournament selection and refers to inserting solution. According to the IVX in Chapter 4, the ranked non-empty routes with route measures are selected. The chosen truck, i, in $j = 2$ uses the inserting trailer, $T_{i,2}$. The corresponding truck, i, in $j = 1$, uses the replaced trailer, $T_{i,1}$. For all semi-trailers in $j = 1$, the truck id, i, which pulls $T_{i,2}$ is identified or this semi-trailer is called "repeated semi-trailer". All available semi-trailers are either attached to the truck or left unused. The pool of unused semi-trailers is hereafter called free semi-trailer list. We show the Semi-trailer Repair Operator in Figure7.1.

Algorithm 7.1 Semi-trailer Repair Operator

1. Delete all repeated semi-trailers in the replaced solution.
2. Remove all requests in these repeated trailers to the request bank.
3. For each inserting vehicle, if the inserting trailer is different from the replaced trailer, then the replaced trailer is removed to the free semi-trailer list.
4. Insert the inserting trailer to the corresponding truck
5. Record the volume capacity of all replaced semi-trailers
6. Use Semi-trailer Insertion Heuristics to insert trailers from free semi-trailer list

In Algorithm 7.1, the Semi-trailer Insertion Heuristic is described in Algorithm 7.3 in Section 7.2.2. Step 2 in Algorithm 7.1 is similar to the group-oriented mutation operator for the PDPTW, used by Pankratz (2005). Once a cluster or route is selected and removed, each request is reinserted again to the insertion location that causes minimal additional cost.

Semi-trailer Removal

To cope with the ITSRP, trailer re-assignment plays a vital role in improving the solution quality because each trailer has different weight, weight capacity, volume capacity and variable cost. The semi-trailer removal is only applied for the company's own fleet and consists of three operators for the attached trailer: the random removal of

Algorithm 7.2 Heuristic selection of semi-trailer

1: Calculate the average fuel used per location for each vehicle

2: For all vehicles, sort the average fuel used per location in descending order

3. Select one vehicle, k, with some randomness, p_{sr}, according to Equation 4.13

4. Record the volume capacity of the selected semi-trailer

5. Remove all requests in k to the request bank

6. Remove the selected trailer to the free-trailer list.

one trailer, the random removal of two trailers and the heuristic removal of one trailer. The removed requests from the random removal of one and two trailers are placed in the request bank. The heuristic selection is shown in Algorithm 7.2.

It is important to note that the heuristic removal operator only requires the heuristic insertion operator as described in Algorithm 7.3 in Section 7.2.2. These three semi-trailer removal operators are applied to every pre-defined number of iterations, N_{SR}.

7.2.2 Insertion Operators

The same regret-heuristics as in Ropke and Pisinger (2006) are used to solve the ITSRP. The detail of regret-1,2,3,4,m are shown in Section 4.5.3 of Chapter 4.

In order to maintain the feasibility for the ITSRP, each insertion requires constraint checking for special requests and multi-dimensional capacity constraints in addition to those in the MDPDPTW. Before insertion, the compatibility between requests and the truck/semi-trailer must be checked. This is carried out in the same way as the special requests for vehicle in Chapter 4.

Moreover, two additional capacity constraints are modelled in the ITSRP. These two capacity constraints are simultaneously considered when inserting a request. The special requests and multi-dimensional capacity constraints are embedded in the fix forward insertion developed in Chapters 3 and 4.

Semi-trailer insertion heuristic (SIH)

In the ITSRP, the trailer re-assignment is essential to improve the solution quality. Therefore, the semi-trailer insertion heuristic (SIH) was proposed after a number of experiments for several semi-trailer insertion heuristics were conducted. The SIH is applied for both the semi-trailer removal and the trailer repair operator. Hereafter, the volume of removed trailers in a semi-trailer removal and trailer repair operator is called the record volume. The pseudo code of the SIH is outlined in Algorithm 7.3.

Algorithm 7.3 Procedure of Semi-trailer Insertion Heuristic (SIH)

1. Recall the recorded volume
2. Record the set of semi-trailers, T_v, in the free semi-trailer list, where T_v have the volume equivalent to the recorded volume
3. For all T_v, sort the variable cost in ascending order
4. Select one semi-trailer, $T^{'}$, with some randomness, $p_{sih,}$ according to Equation 4.13
5. Insert semi-trailer, $T^{'}$, to k
6. Insert the requests in the request bank to the solution by randomly selecting an insertion heuristic

In Algorithm 7.3, the reason we prioritise the semi-trailer equivalent to recorded semi-trailer earlier is that using lower semi-trailer capacity may result in unscheduled requests left in the request bank. The IVX also helps compacting the semi-trailer assignment. The combination of the Semi-trailer Removal and Semi-trailer insertion Heuristic can be seen as trailer re-assignment operators.

7.2.3 Adaptive Mechanism

The adaptive mechanism of the AMLNS for the ITSRP is the same as that used in the AMLNS in the Chapter 4. It is also important to note that the roulette wheel selection is not required for semi-trailer removal and SIH. From the experiments, the fixed

probability of one-third for each semi-trailer removal operator is simple and sufficient.

7.2.4 Applying Noise to Objective Function

The method of applying noise to the objective function remains the same as in Ropke and Pisinger (2006).

7.2.5 Initialisation

The population is initialised according to the AMLNS in Chapter 4. For each initial solution, the trailer is assigned to each truck randomly. A number of rules were designed for initialising trailers to trucks. However, from the experiments, the random trailer insertion shows promising results over the designed rules. It is believed that the random trailer initialisation can generate diverse populations as essentially required for the AMLNS.

7.2.6 Master Local Search Framework

The master local search framework in the AMLNS for the MD-PDPTW-SR is applied to the ITSRP. Regarding the reduction rules namely the calculation of the objective function, the calculation of incremental distance, and calculation of incremental time, described in Chapter 4, are removed in the ITSRP due to changes in the structure of the objective function.

7.3 Computational Experiments

The AMLNS for solving ITSRP was coded in high-level computer language, C#, in the Visual Studio 2010. The proposed AMLNS was run on a single-thread of Intel core I7 (3.5 GHz). The objective values were rounded to a double precision floating point number. The aim of the experimental study was to compare the performance between the proposed AMLNS and original ALNS with slight modifications for solving

the ITSRP.

7.3.1 Development of Sets of Problem Instances

Up to present, there is no suitable test sets for the ITSRP. For computational experiments, 36 test instances were therefore developed for testing for the ITSRP using the ALNS and AMLNS. The generated ITSRP instances that contain features of the model were not used in Ropke and Pisinger (2006). These features include different fulfilment modes, special requests for trucks and semi-trailers and multi-dimensional capacity constraints. The problem features of the test instances developed for the ITSRP are shown in Table 7.2. When developing these instances, the cost rates and parameters, which should be related, are obtained from the real-life business operations of the case-study company. For the matter of generalisation, we develop the test instances according to the literature. Several geographical distributions: uniform, clustered, and semi-clustered are experimented on. These three types of problem were inspired from Ropke and Pisinger (2006). The small-test instances for validating the ALNS and AMLNS by CPLEX are also developed. In this experiment, the problem size of 50 and 100 requests are considered. It is to note that the problem sizes of 50 and 100 requests have 100 and 200 customer locations, respectively. For each problem size, we generated 18 problems according to every combination of the three problem features shown below:

- **Proportion of fleets**: for the case study company's own fleet and for the subcontractor paid by request basis, a route starts and ends at the same location. However, the subcontractor paid tour basis, a route starts and ends at different location. In this study, to simulate the possible scenarios occurring in the real-life operation, three different mixed percentages of fleets were experimented on for own, sub-contraction on request basis, and sub-contraction on tour basis.
- **Request types**: (1) all requests are normal requests: the normal request is for

the 20-feet dry container that can be served by all type of vehicles. (2) 50 % of the requests are special requests. For special requests, they can be served by a subset of trucks or semi-trailers.

- **Geographical distribution**: (1) uniform, (2), clustered and (3) semi-clustered.

For all problems, the ratio of the number of the company's own trucks and own semi-trailers is 1: 2. It is important to note that the network structure of the real-life data of the case-study company is usually clustered. This geographical distribution is already included in the developed test instances. The possible combination of problem types A to R is shown in Table 7.2.

Type	Mix fleet type			Request type		Geographical distribution		
	Own	*Own* + 1^{st}Sub	*Own* + 1^{st} + 2^{nd} Sub	Norm. req.	Spec. req.	U.	C.	SC.
A	√			√		√		
B	√				√	√		
C		√		√		√		
D		√			√	√		
E			√	√		√		
F			√		√	√		
G	√			√			√	
H	√				√		√	
I		√		√			√	
J		√			√		√	
K			√	√			√	
L			√		√		√	
M	√			√				√
N	√				√			√
O		√		√				√
P		√			√			√
Q			√	√				√
R			√		√			√

Table 7.2: The features of the test instances for the ITSRP

In Table 7.2, *Own*, *Own* + 1^{st}*Sub*, and *Own* + 1^{st} + 2^{nd}*Sub* represent the use of (1) own fleet, (2) own fleet and sub-contraction on request basis, and (3) own fleet and sub-contraction on request basis and sub-contraction on tour basis, respectively.

Norm. req. and Spec. req. refers to the request types of normal request and special request, respectively. U., C., and UC. stand for uniform, clustered, and semi-clustered geographical distribution, respectively.

7.3.2 Tuning instances

First, a set of representative tuning instances is identified. In order to perform numerous experiments, the set of tuning instances must have a fairly limited size and related to the problem targeted. The set of tuning instances consists of 6 instances namely, 50A, 50I, 50Q, 100B, 100J, and 100R. In each instance, the name, the number and letter indicate the problem size, followed by type.

7.3.3 Parameter tuning

A number of experiments are conducted to find a good set of parameters. The initial set of parameters is adopted from Chapter 4 and new parameters are produced by an ad-hoc trail-and-error phase. Design and Tuning Process as in Chapter 4 is also applied. The parameter setting is tested by running the algorithms five times. The potential range of each parameter is tested. The parameter tuning is improved by allowing one parameter to take a number of values, while the rest of the parameters remain fixed. The best known solution for each problem is kept and updated. The next parameter is tested by applying the values found so far and the values of the parameters that have not been considered yet. This procedure continues until all parameters have been tuned. The development process of the AMLNS for the ITSRP is also similar to that of Chapter 4 but with minimum changes of design. Therefore, we mainly focus on solution quality, instead of computational time. Thus, the problem is to minimise the $Obj\,fn$, as described in Section Design and Tuning Process in Chapter 4, using $\phi, \upsilon = 1$ and $\varsigma = 0$.

In order to compare the AMLNS with the existing metaheuristics, the ALNS was developed. In order to cope with the ITSRP, the semi-trailer removal and insertion operators are also added to the ALNS with the low rate of N_{sr} ($N_{sr} = 5$), applying them every 5 iteration, to acheive good results. For this ALNS, all operators are used according to Pisinger and Ropke (2007), instead of Ropke and Pisinger (2006), since, in the table of computational results shown in www.diku.dk/∼ropke is obtained from the updated heuristic. However, the start temperature control parameter and cooling rate are not given in Pisinger and Ropke (2007). Therefore, the start temperature control parameter and cooling rate shown in Ropke and Pisinger (2006) are applied. According to Pisinger and Ropke (2007), in order to cope with different problem sizes, the start temperature control parameter should be divided by the number of requests in the instance. From the experiments of the ALNS for the ITSRP, this implementation also works well for the start temperature and cooling rate obtained from Ropke and Pisinger (2006).

The AMLNS for solving the ITSRP is extended from the AMLNS for the MD-PDPTW-SR. In order to keep the parameter tuning to a minimum, the set of parameters used in the AMLNS in Chapter 4 is also used here. However, the ITSRP is different from the MD-PDPTW-SR in terms of objective function and constraints. The number of parameters and designs are thus experimented on. One of the experiments that has had a significant impact on the performance of the heuristic is the pre-defined number of iterations for using the semi-trailer removal, N_{SR}. Moreover, this mechanism is not used in the AMLNS in Chapter 4. The range of N_{SR} from 5 to 500 over 25000 iterations is tested in the AMLNS.

N_{SR}	2	5	10	20	50	100	200	500	None
$Obj\ fn$ (%)	-0.02	**-0.11**	0.04	0.39	0.89	1.35	1.6	1.61	1.9

Table 7.3: Parameter N_{SR} vs. $Obj\ fn$

Table 7.3 shows the influence of N_{SR} toward the $Obj\ fn$. The results show that the

N_{SR} is essential in improving the solution quality. Since, initial random assignment of trailer may not be efficient. The re-assignment of trailer is then required. However, the frequency of N_{SR} should be also limited to a certain value so that the removal and insertion of requests can also work on routing the locations, instead of only using the semi-trailer removal. According to Table 7.3, $N_{SR} = 5$ is selected. In addition to N_{SR}, the number of parameters was further experimented in terms of measures in the IVX for the ITSRP. Moreover, the number of designs was proposed. Table 7.4 and 7.5 show the experiments of the first and second measures used in IVX in Section 7.2.1, respectively, on the following designs:

- AMLNS_1 uses the original measures as in Chapter 4.
- AMLNS_2 replaces a new measure to average distance
- AMLNS_3 replaces a new measure to average distance-time
- AMLNS_4 add the new measure to average distance and average distance-time
- AMLNS_5 uses only the new measure
- AMLNS_6 uses random selection of good routes and the new measure

	AMLNS_1	AMLNS_2	AMLNS_3	AMLNS_4	AMLNS_5	AMLNS_6
$Obj\ fn$ (%)	-0.29	-0.31	-0.27	-0.27	-0.3	-0.32

Table 7.4: First measure of IVX in different designs vs. $Obj\ fn$

From AMLNS_ 2 to AMLNS_6, the new measure can refer to either first measure or second measure, as stated in Section 7.2.1. In Table 7.4, the new measure refers to the first measure. The solution quality of IVX's first measure on design AMLNS_6 is slightly better than the other design. The second measure was tested on the same set of

designs but also the combinations between first and second measure are experimented in AMLNS_7, as shown in Table 7.5.

	AMLNS_1	AMLNS_2	AMLNS_3	AMLNS_4	AMLNS_5	AMLNS_6	AMLNS_7
$Obj\ fn$ (%)	-0.29	-0.3	-0.28	-0.31	-0.28	-0.3	**-0.33**

Table 7.5: Second measure of IVX in different designs vs. $Obj\ fn$

From Table 7.4 and 7.5, the combination between first and second measure show the most promising results. Therefore, the AMLNS_7 design in Table 7.5 was selected for further experiments. The randomisation parameters p_{sr} and p_{sih} are both empirically set to 6. The rest of parameters are applied according to the AMLNS in Chapter 4.

7.3.4 Analysis of Typical Search

In order to illustrate how the ALNS and AMLNS work, the representative test instance, problem 100L, was selected to visualise the search behaviour of each meta-heuristic for solving the ITSRP. Figure 7.3 (left) and (right) demonstrate the objective value as a function of the iteration count for the ALNS and AMLNS, respectively.

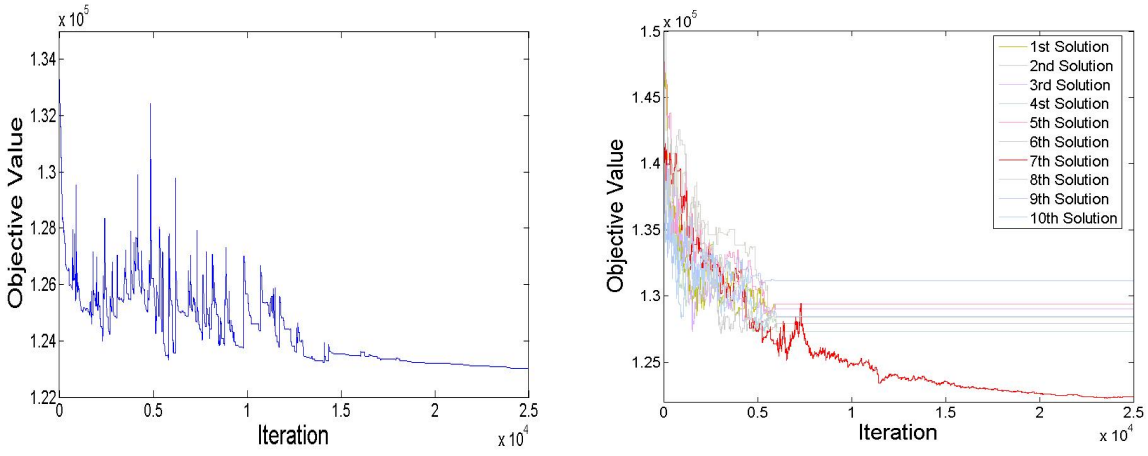


Figure 7.3: (Left) ALNS and (Right) AMLNS used for solving problem 100L

From Figure 7.3 (left), this search behaviour is typical for a simulated annealing

heuristic, and it is also similar to the ALNS search in Pisinger and Ropke (2007). In Figure 7.3 (right), the search behaviour is similar to the AMLNS for the MD-PDPTW-SR in Chapter 4.

7.3.5 Computational results

The computational results are obtained from solving the test instances, as described in Section 7.3.1, by the ALNS and AMLNS, developed for the ITSRP. In order to analyse the performance of the heuristics, a number of notations is introduced, as follows:

• z : Current best known solutions obtained either from the ALNS or the AMLNS during all experiments in this study

• z_{av}, z_b : Values of the average and best solutions in 10 runs, respectively

• $Gap_{av}(\%)$, $Gap_b(\%)$: Percentage deviation of the average and best solution found from the current best known solutions, computed as $100\times(z_{av}-z)/z$ and $100\times(z_b-z)/z$, respectively

• Avg. time (s): the average time (in seconds) of 10 runs

Table 7.6 and 7.7 report the computational results of the ALNS and the AMLNS for 50 and 100 requests, respectively. The AMLNS obtains 32 best known solutions over the 36 test instances while the ALNS obtains 4 best known solutions. Single-underlined numbers indicate the best solution in 10 runs, obtained from the AMLNS that are better than those of the ALNS of Pisinger and Ropke (2007). Double-underlined numbers indicate the average values of each problem in 10 runs, obtained from the AMLNS, better than those of the ALNS of Pisinger and Ropke (2007). The average of Gap_{av} to the best known solutions between the ALNS and the AMLNS are 0.79 and 0.37 respectively. The average of Gap_b between the ALNS and the AMLNS are 0.27 and 0.01. Compared in the same computing environment, the AMLNS also spends less computational time than the ALNS. Since the large number of requests is still applied to the ALNS until the last iteration. From the analysis, the AMLNS outperforms the

Problem	Best Known	ALNS	25K				AMLNS	25K			
	Cost	Avg.	Best.	Avg.	Avg.	Avg.	Avg.	Best.	Avg.	Avg.	Avg.
		Sol.	Sol.	gap	gap	time	Sol.	Sol.	gap.	gap	time
Notation	z	z_{av}	z_b	Gap_{av}	Gap_b	s	z_{av}	z_b	Gap_{av}	Gap_b	s
50-A	77986.62	78069.87	77986.62	0.11	0.00	26	78138.82	78034.51	0.21	0.08	19
50-B	82235.44	82830.72	82471.58	0.72	0.29	22	82433.50	82235.44	0.24	0.00	16
50-C	68651.18	69154.75	68714.75	0.73	0.09	25	68909.22	68651.18	0.38	0.00	18
50-D	77127.67	77594.21	77388.03	0.60	0.34	21	77532.17	77127.67	0.52	0.00	16
50-E	66443.59	66719.98	66443.59	0.42	0.00	24	66626.39	66449.69	0.27	0.00	18
50-F	73916.09	74726.53	74380.52	1.10	0.63	21	74373.36	73916.09	0.62	0.00	16
50-G	72514.25	72789.53	72582.95	0.38	0.09	26	72610.91	72514.25	0.13	0.00	19
50-H	73710.72	73985.14	73874.73	0.37	0.22	21	73953.77	73710.72	0.33	0.00	16
50-I	64566.80	65077.28	64670.76	0.79	0.16	24	64724.27	64566.80	0.24	0.00	18
50-J	56646.99	57034.07	56768.03	0.68	0.21	23	56929.51	56646.99	0.50	0.00	17
50-K	59568.41	60181.04	59982.82	1.03	0.70	24	59941.45	59568.41	0.64	0.01	18
50-L	52402.71	53154.7	52545.59	1.44	0.27	26	52604.15	52402.71	0.38	0.00	20
50-M	72771.57	72948.92	72799.04	0.24	0.04	25	72983.15	72771.57	0.29	0.00	19
50-N	74207.51	74630.38	74291.26	0.57	0.11	22	74426.43	74207.51	0.30	0.00	16
50-O	62243.46	62978.12	62494.78	1.18	0.40	24	62495.96	62243.46	0.41	0.00	19
50-P	64073.68	65003.71	64199.68	1.45	0.20	21	64381.91	64073.68	0.48	0.00	16
50-Q	65353.85	66489.91	65916.91	1.74	0.86	24	65700.21	65353.85	0.53	0.00	19
50-R	72425.81	74056.6	73220.7	2.25	1.10	21	72740.99	72425.81	0.44	0.00	16

Table 7.6: Comparison between the ALNS and AMLNS for 50 requests

Problem	Best Known	ALNS	25K				AMLNS	25K			
	Cost	Avg.	Best.	Avg.	Avg.	Avg.	Avg.	Best.	Avg.	Avg.	Avg.
		Sol.	Sol.	gap	gap	time	Sol.	Sol.	gap.	gap	time
Notation	z	z_{av}	z_b	Gap_{av}	Gap_b	s	z_{av}	z_b	Gap_{av}	Gap_b	s
100-A	149555.93	150346.36	149710.02	0.53	0.10	93	150038.05	149555.93	0.32	0.00	69
100-B	159428.02	160363.18	159567.67	0.59	0.09	83	159854.63	159428.02	0.27	0.00	63
100-C	128509.47	129533.94	128918.02	0.80	0.32	94	129072.24	128509.47	0.44	0.00	69
100-D	144746.18	145713.51	145264.57	0.67	0.36	85	145376.11	144746.18	0.44	0.00	63
100-E	132040.15	132939.5	132252.37	0.68	0.16	88	132478.77	132040.15	0.33	0.00	65
100-F	142575.04	143377.54	142786.03	0.56	0.15	79	142979.89	142575.04	0.28	0.00	60
100-G	127296.57	127904.67	127414.75	0.48	0.09	100	127676.68	127296.57	0.30	0.00	74
100-H	136653.94	137454.79	136734.47	0.59	0.06	89	137167.88	136653.94	0.38	0.00	66
100-I	107079.55	108119.79	107391	0.97	0.29	95	107453.72	107079.55	0.35	0.00	71
100-J	118262.23	118990.73	118447.73	0.62	0.16	84	118783.22	118262.23	0.44	0.00	62
100-K	111411.75	112632.96	111870.37	1.10	0.41	94	111780.97	111411.75	0.33	0.00	71
100-L	121957.43	122525.88	122126.43	0.47	0.14	87	122265.34	121957.43	0.25	0.00	65
100-M	147745.90	148465.64	147774.23	0.49	0.02	93	148033.77	147745.90	0.19	0.00	69
100-N	154525.29	155140.12	154525.29	0.40	0.00	84	154984.87	154613.74	0.30	0.06	62
100-O	130862.44	133051.1	132821.73	1.67	1.50	85	131212.18	130862.44	0.27	0.00	65
100-P	128875.70	129832.01	129052.97	0.74	0.14	78	129374.24	128875.70	0.39	0.00	58
100-Q	129232.97	129943.19	129373.57	0.55	0.11	95	129710.82	129232.97	0.37	0.00	71
100-R	134739.65	135794.07	134739.65	0.78	0.00	83	135636.93	135022.06	0.67	0.21	61
Avg.				0.79	0.27	56			0.37	0.01	42

Table 7.7: Comparison between the ALNS and AMLNS for 100 requests

ALNS in this set of test instances of the ITSRP. In order to understand the solution structure of the problem, the network topology of the best known solution of problem 50F is illustrated in Figure 7.4.

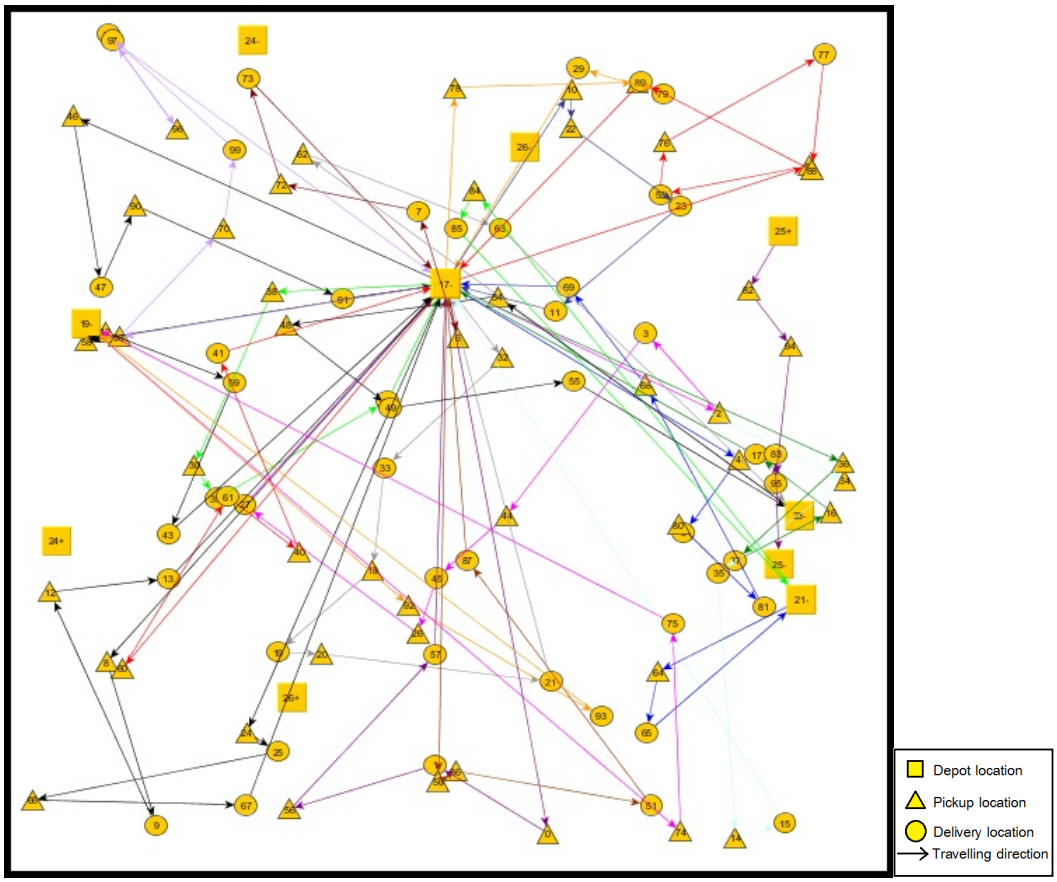


Figure 7.4: Network topology of best known solution: problem 50F (100 locations)

In Figure 7.4, the problem 50F contains the following characteristics: 50 requests (100 locations), special requests for trucks and semi-trailers, uniform geographical distribution. There are 18 company owned trucks, 6 trucks owned by subcontractors who work on a paid by request basis, and 3 trucks owned by subcontractors paid by tour basis. The ratio of own trucks and trailers is 1:2 while that of subcontractors is 1:1. Table 7.8 shows the attached trailer id to each truck of the solution corresponding to Figure 7.4.

k	t		k	t		k	t		k	t
0^o	23		7^o	12		14^o	15		21^{S1}	39
1^o	22		8^o	3		15^o	16		22^{S1}	40
2^o	1		9^o	13		16^o	32		23^{S1}	41
3^o	25		10^o	14		17^o	29		24^{S2}	42
4^o	26		11^o	33		18^{S1}	36		25^{S2}	43
5^o	30		12^o	2		19^{S1}	37		26^{S2}	44
6^o	17		13^o	0		20^{S1}	38			

Table 7.8: Semi-trailer assignment of the best known solution: problem 50F (100 locations)

In order to validate the solution obtained, the schedule of each route is visualised and its feasibility checked. Table 7.9 illustrates the schedule of the sampled routes. It is important to note that the truck id 17, 21, and 25 belong to the company's own fleet, to subcontractors paid by request basis and to subcontractors paid by tour basis, respectively.

In Table 7.9, the schedules of representative routes for each fulfilment mode are illustrated. The first column is the truck id, followed by semi-trailer id. Seq. stands for the sequence of location id. The following abbreviations indicate travelling time (tra. t.), arrival time (arr.t.), time windows (TWs), service time (ser.t), departure time (dep.t.), supply load (S_L), demand load (D_L), loading status (L), combined load between trailer and goods ($L^{'}$), total load ($Tot.L$), supply volume (S_v), demand volume (D_v) and volume status (V) are reported. The first line of each route for TWs, $L^{'}$(T+G), $Tot.L$ and V are their constraints. The last line of each route shows the cost detail of each truck and trailer. For the company's own fleet, the truck fixed cost (Truck FC), trailer fixed cost (Trailer FC.), fuel cost (Fuel C.) distance cost (Dist. C.) and expense for all pickup-delivery requests (PD.C.) are given. For the subcontractors' fleet, the cost of sub-contraction on request basis (Sub.Req.C.) and the cost of sub-contraction on tour basis (Sub.Tour.C.) are also shown. The details and costs of all routes are illustrated in Appendix E. Moreover, the network topology of the best known solution obtained from the AMLNS for problem 100Q is also shown in Figure 7.5.

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	$L^{'}$(T+G)	$Tot.L$	S_V/D_V	V
17	D0	-	-	0-660	0	0	0	0		75	45		80
29	60	106.09	106.09	98-374	0	62	168.09	6	6	13.216	20.986	70	70
	61	42.51	210.6	129-325	0	42	252.6	-6	0	7.22	14.99	-70	0
	40	19.09	271.69	211-470	0	81	352.69	9	9	16.22	23.99	80	80
	41	45.14	397.83	221-444	0	49	446.83	-9	0	7.22	14.99	-80	0
	D0	50.48	497.31	0-660	0	0	0	0	0				
	Truck FC.	491.4	Trailer FC.	361.8	Fuel C.	954.34	Dist. C.	1057.76	PD. C.	520			
21	D2	-	-	0-660	0	0	0	0		70	45		80
39	64	33.93	46	46-307	12.07	40	86	20	20	27.508	34.978	40	40
	65	13.03	99.03	357-579	257.97	82	439	-20	0	7.51	14.98	-40	0
	D2	43.11	482.11	0-660	0	0	0	0	0				
	Sub.Req C.	1478.9											
25	D5	-	-	0-660	0	0	0	0		70	45		80
43	82	14.44	267	267-471	252.56	64	331	13	13	20.07	27.54	60	60
	94	14.74	345.74	219-398	0	65	410.74	13	26	33.07	40.54	10	70
	95	29.04	439.77	248-485	0	37	476.77	-13	13	20.07	27.54	-10	60
	83	6.22	483	320-598	0	62	545	-13	0	7.07	14.54	-60	0
	D5	23.17	568.17	0-660	0	0	0	0	0				
	Sub.Tour	1810.64											

Table 7.9: Schedules of truck 17, 21, and 25 of best known solution: problem 50F

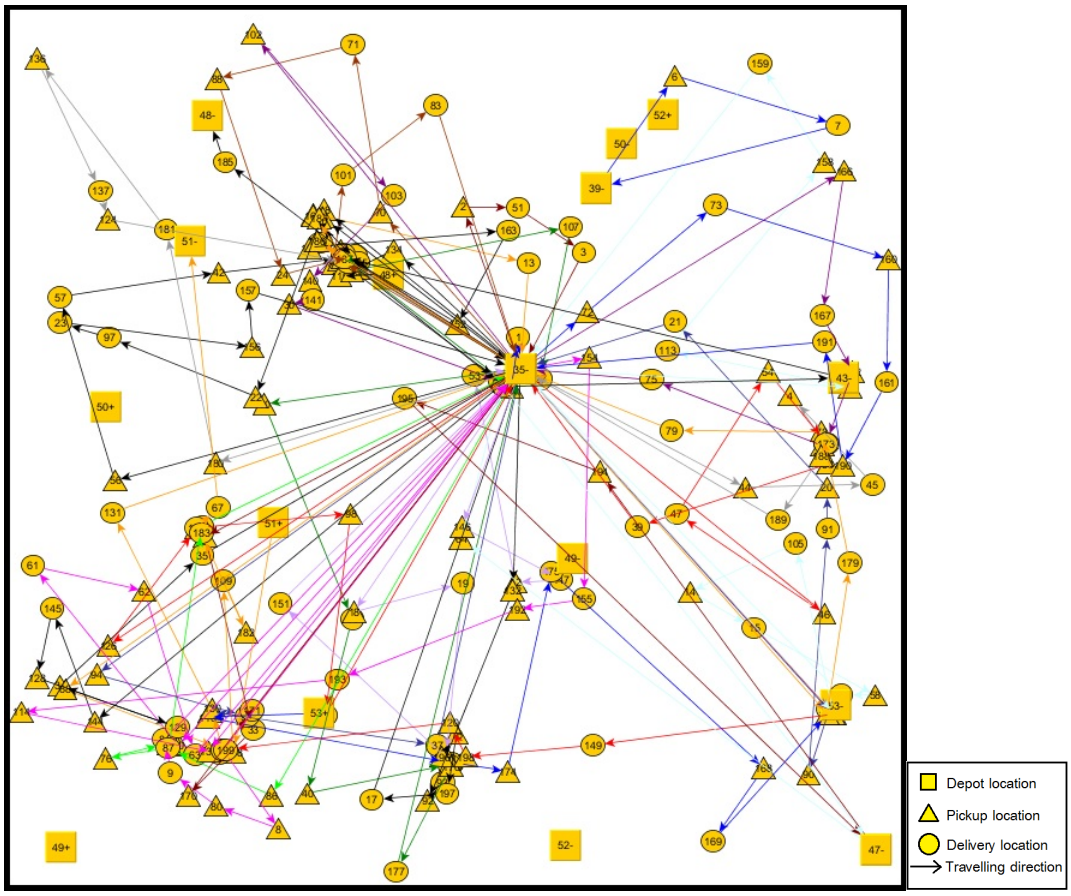


Figure 7.5: Network topology of best known solution: problem 100Q (200 locations)

In order to demonstrate problem size, and complexity, in Figure 7.5, the best known solution of the problem 100Q (200 locations) geographically distributed by semi-clustered are illustrated. For this problem type, own fleet, subcontractors paid by request basis, and subcontractors paid by tour basis are all used.

7.4 Discussion

7.4.1 Algorithmic Perspectives

The ITSRP is a complex combinatorial optimisation problem. To be precise, the assignment of trailers to truck, sub-contraction, and multi-dimensional capacity constraints are incorporated in to the MD-PDPTW-SR. In Chapter 4, the solution quality obtained from the AMLNS was slightly better than that from the ALNS. However, for the ITSRP, the AMLNS shows very promising results.

In Ropke and Pisinger (2006) and Pisinger and Ropke (2007), the ALNS was developed for solving various routing problems. In this Chapter, the ALNS for the ITSRP was slightly modified by adding the random initial trailer assignment of trailer and trailer re-assignment operator. The AMLNS originally developed from the ALNS in Chapter 4 was also proposed for solving the ITSRP. The random initial trailer assignment and trailer re-assignment operator were also incorporated in the AMLNS.

For the AMLNS, the random initial trailer assignment is applied to all solutions in the population. The different seeding and insertion heuristics are also used for generating the initial population, as originally applied in Chapter 4. These operations provide diversity in terms of the assignment of semi-trailers to trucks and, then, different configurations of locations in routes. The IVX operator transfers both sequenced locations and their attached trailers together. According to the design concept of the AMLNS, the good routes and its semi-trailers are selected according to the route quality mea-

sures corresponding to the objective function of the ITSRP. It is, therefore, believed that these additional operators and features naturally tackle the semi-trailer assignment problem, and thus enhance the performance of the AMLNS. With these operators, only few changes are made to from the original AMLNS. In order to improve the performance of the ALNS for solving the ITSRP, specially designed operators for trailer assignment may be required.

7.4.2 Managerial Perspectives

By analysing the cost structures from the ITSRP, the use of own fleet's vehicles is generally cheaper than hiring a vehicle from a subcontractor paid by request basis. Since, the sub-contraction rate, $Y_{i,k}$ is normally lower than the percentage of profit margin, o_i added to the total cost of each request, resulting the higher cost of request execution. In addition, the distance saving of request consolidation is not taken into account as the costing of the company's own fleet. The more the sub-contraction on request basis is used, the more possible it is for expensive total costs to occur. Moreover, if there are vehicles of the company's own fleet left unused, and vehicles of subcontractors paid by request basis are utilised instead, due to inefficient routing. From subcontractors' point of view, the fixed costs of the company's own fleet are sitll incurred. The cost structure of sub-contraction on request basis is also comprised of a fixed cost and variable cost. Thus, in other words, the fixed costs are approximately paid twice i.e. for the vehicles of the company's own fleet and the subcontractor's. This situation is undesirable, but is usually faced by third-party logistics providers. To avoid this, priority should be given to the use of the company's own fleet, if sub-contraction is used on a request basis, corresponding the result of the AMLNS.

In this study, a vehicle of a subcontractor on tour basis is inputted from the backhauling system, which seeks to reduce empty backhauling among logistics service providers. Therefore, the cost of sub-contraction is rather cheap because some of its fixed costs and variable costs to the pickup location are already covered from the line-haul customer.

However, the requests served should align between the start and end location of the vehicle and the diversion time from this aligned route is limited.

In practice, there are issues with using subcontractors, for example, the service aspects, control, monitoring and flexibility. These aspects may still convince the use of a company's own fleet as much as possible. However, the demand varies over the year and logistics providers normally cover the base load of demand through their own fleet, while inevitably using hired vehicles over peak periods. Nevertheless, if the requests are always subcontracted in every period, an appropriate number of the vehicles and equipment should be re-calculated. The AMLNS developed in this Chapter are the essential basis for daily operational planning. It optimises the total cost and determine the most effective mix of a company's own and hired vehicles. Then, this historical information daily produced by the algorithm can be used for finding the optimal fleet size or least cost mix of fleet over a period of time. The company should revise the fleet capacity, which is part of tactical level decisions. Moreover, the depot configuration can be analysed using the AMLNS.

7.5 Summary

In this Chapter, the Integrated Truck and Semi-trailer Routing Problem (ITSRP) was presented and one case-study company was investigated accordingly. The cost structure of fulfilment modes for a company's own fleet and their use of sub-contraction were analysed. The ITSRP is formulated as Mixed-Integer Linear Programming. The ITSRP is a complex combinatorial optimisation problem. Due to being NP-hard problem, a meta-heuristic must be developed to efficiently solve the ITSRP within reasonable time to cope with rapidly changing business environments. The Adaptive Memetic Large Neighbourhood Search (AMLNS) and the Adaptive Large Neighbourhood Search (ALNS) were then proposed to solve the ITSRP. The AMLNS and ALNS were successfully used to solve the MD-PDPTW-SR, as shown in Chapter 4. For the ITSRP, the

ALNS and AMLNS were incorporated with additional operators to tackle the semi-trailer assignment problem. Furthermore, the crossover of the AMLNS was slightly modified from that of its original version, according to its design principle of IVX, to cope with the different objective function. The heuristics were tested on the set of test instances simulating the real-life scenarios. The AMLNS provides very promising results to solve the ITSRP.

Chapter 8

Conclusions

8.1 Contributions

Starting with an overview of the current literature on variants of vehicle routing problems, and pickup and delivery problems, the solution methods applied to these problems were then discussed. In Chapter 3, the Multi-depot Pickup and Delivery Problem was formulated by Mixed-Integer Linear Programming (MILP). The problem incorporates several constraints and characteristics over the classical VRP, namely multiple depots, heterogeneous fleet, precedence relationships and maximum route length. The objective function is to minimise the total distance travelled. CPLEX was used to solve the generated test instance. Several CPLEX parameters were tested. However, CPLEX using default Branch-and-Cut can solve small-sized problems only. Due to being NP-hard, the Memetic Algorithm was proposed to tackle the MDPDP. The solution representation used is able to handle complicated constraints and is applicable for crossover and local search. The operators were adapted from those with related problems such as Multi-depot Vehicle Routing Problems and Pickup and Delivery Problem with Time Windows. The MDPDP and its variants are complex and highly constrained. Therefore, a specialised insertion operator, called fixed forward, was developed to reduce

the computational time by taking advantage of infeasibility conditions such as precedence and capacity constraint. The fixed forward can be further embedded in local search and insertion operators. Chapter 3 was presented at 24^{th} European Conference on Operational Research, 11-14 July , 2010 in Lisbon, Portugal.

In Chapter 4, a complex variant of MDPDP was investigated. The problem referred to the Multi-depot Pickup and Delivery Problem with Time Windows and Special Request (MD-PDPTW-SR). Over the MDPDP, this problem incorporates time windows, special requests, multi-depot characteristics- a route starts and ends at different locations (route type)- and maximum route time into consideration. The objective function is to minimise the weighted sum of total travelling distance, travelling time and the number of unserved requests. The solution representation developed from Chapter 3 was adapted to tackle this problem. The reduction rules based on time feasibility and objective calculation were developed and incorporated into the fixed forward. A hybrid metaheuristic, called Adaptive Memetic Large Neighbourhood Search (AMLNS) was proposed. The AMLNS is hybridised among Adaptive Large Neighbourhood Search (ALNS), Memetic Algorithms (MA), and Threshold Accepting (TA). It is important to note that the ALNS is the state-of-the-art metaheuristics in this problem. The design principles for hybridising metaheuristics were surveyed together with the state-of-the-art metaheuristics for its related problems. An adaptive crossover operator, Identical Vehicle Crossover (IVX), was proposed. The proposed hybrid metaheuristics were tested with the 48 standard benchmark test instances taken from the literature, as generated by Ropke and Pisinger (2006). The range of problem sizes are from 50 (100 locations) to 500 (1000 locations) requests. Three different types of problems, namely route types, request types, and geographical distribution problems were tested. The computational results of ALNS and AMLNS were compared. From all experiments, over 48 test instances, 4 best known solutions equivalent to those from the ALNS of Pisinger and Ropke (2007) were retrieved. Moreover, 43 new best known solutions were obtained. The proposed AMLNS is promising in terms of robustness, and is measured

from the overall average gap (%). The problem and solution method in this Chapter were presented at 9^{th} International Conference on Computational Management Science, 18-20 April 2012, Imperial College London, United Kingdom, and also at the 25^{th} European Conference on Operational Research, 8-11 July 2012 in Vilnius, Lithuania. In Chapter 5, extensive computational experiments on developing the AMLNS were carried out. Specialised mechanisms were developed to hybridise the population-based and single solution approach.

In Chapter 6, another complex variant of MDPDP, arising in practice, was studied. Over the MD-PDPTW-SR considered previous, the problem incorporates several characteristics and constraints arising in real-life problems, including sub-contraction, semi-trailer assignment, multi-dimensional constraints and special requests in terms of trucks and/or semi-trailers. A case-study company providing freight transportation service was investigated. This problem is then entitled Integrated Truck and Semi-trailer Routing Problem (ITSRP). A management accounting technique, Capacity-Driven Activity-Based Costing, was applied to obtain the cost structure and pricing method for formulating the problem. The information concerning costs and parameters was obtained from historical data collected from the case-study company. In Chapter 7, the problem was formulated as MILP, as CPLEX can only solve small problem sizes. The 36 test instances generated up to 100 requests (or 200 locations). Three problems characteristics were studied to generalise and simulate the possible scenarios arising in this problem. These include the types of sub-contraction considered, request type for trucks and trailers, and geographical constraints. The ALNS and AMLNS, previously used, were modified at minimum while being capable of handling additional constraints of the ITSRP. The computational experiments show the AMLNS provides very promising results. From analysis, one observation is that the ITSRP involves semi-trailer assignment problems that can naturally be tackled by the IVX. However, the ALNS may require a specialised operator for semi-trailer assignment to deal with the ITSRP.

In terms of fundamental research, this thesis provides the development of Adap-

tive Memetic Large Neighbourhood, which is a hybridisation of Memetic Algorithms, Adaptive Large Neighbourhood Search and Threshold Accepting. The AMLNS cannot be categorised as a population-based or single-solution approach because the AMLNS bridges the strengths and applies the key components of both population-based and single-solution approaches. Specialised mechanisms are designed to integrate the key complementary components from these algorithms. The key components are investigated and discussed in details in this thesis. The AMLNS provides robustness over the test instances of Ropke and Pisinger (2006) for solving the MD-PDPTW-SR and the test instances for the ITSRP. The AMLNS may be an emerging powerful hybrid metaheuristic that requires further investigation for other variants of the VRPs and PDPs.

In terms of applied research, the problems consider several real-life characteristics and constraints arising in logistics, transportation of goods and passengers, and freight transportaion etc. These are categorised by requests, fleet, route structure objectives and scope planning according to Drexl (2012). Figure 8.1 shows the dimension of richness covered in this thesis, according to Drexl's (2012) framework. In addition, in term of cost, sub-contraction on request basis and sub-contraction on tour basis are incorporated. Moreover, the ITSRP considers the compatibility between requests and semi-trailer, referred to as special requests for semi-trailer. All of these are incorporated into the variants of the MDPDPs. The problems are the core basis of the real-life routing problems for logistics businesses. We formulated these variants of MDPDPs by Mixed-Integer Linear Programming (MILP) and solved by CPLEX. They are also solved by the proposed AMLNS for industrial problem sizes.

8.2 Concluding remarks

The variants of MDPDP, including MD-PDPTW-SR and ITSRP are the backbone of several routing problems arising in the real-world applications for freight forwarders,

Figure 8.1: Characteristics of VRPs considered in this thesis

LSPs and 3PLs. These problems are complex in terms of problem characteristics and constraints. Moreover, they are NP-hard problem as extensions of VRP. The models, operators and algorithms considered should be able to tackle the variants with slight modifications. Due to advances in computational power and algorithmic development of exact methods, it is important for schedulers to select the right methodology to tackle the problem at hand in terms of problem size and complexity. Each method has advantages, disadvantages and, thus, trade-off. While the exact methods provide optimal solution, the allowed computational time in rapidly changing business environments is also restricted. The schedulers must evaluate these dimensions of trade-off.

When introducing a heuristic to a new routing problem e.g. arising form a new business model, one important issue is the robustness of the algorithm due to not being guaranteed of obtaining optimal solution. A new problem may take different sub-problems into account as shown in Chapter 6 and 7. The algorithmic design of the heuristic and an in-depth understanding of the problem domain are therefore important. Moreover, several problems scenarios in the test instances must be covered to replicate real scenarios. Due to the unavailability of benchmark values in new test instances, several designs and experiments must be carried out to ensure the robustness of the heuristic considered.

Among implementation issues, the integration of Geographical Information System (GIS) is essential. According to Gruenert (2012), the shortest distance of an origin and destination matrix from some commercial GIS software is not optimal. However, the exact method for solving this problem exists. In addition, most GIS software cannot distinguish if the road can be traversed by car or truck, an important point for its application in freight transportation. These characteristics should also be solved and made publicly available for improving accuracy in routing and scheduling.

The design and implementation of solution methods in solving real-life routing problems still requires considerable attention. There are still gaps between theory and practice. However, substantial skills and knowledge are required for practitioners to

develop a metaheuristic to tackle their routing problems, yet researchers still lack problem data, costs, and parameters in real world applications. Possible reasons for this are commercial sensitivity and accessibility of the information. These issues must be resolved.

8.3 Future work

In terms of algorithmic development, one interesting direction is to apply Parallel Computing to the AMLNS, since one motivation of development of the AMLNS is that Ropke (2009b) discussed was that the Parallel ALNS seems to work against the SA principle. The AMLNS was then developed from GAs widely implemented in parallel computing to speed up the search.

In terms of problem characteristics, it may be possible to take a dynamic feature into account. The problems considered in this thesis assume that all relevant data is known *a priori*, then routes and schedules can be generated using static planning systems. In dynamic problems, some input is unknown at the time of planning, and some input is not known with certainty. The planning horizon cannot be known or be an open-ended process. Algorithms for dynamic planning must have fast response times. Simulation can be used to generate dynamic scenarios.

In terms of implementation in the real world, the integration of fleet telematics must also be considered. The developed algorithm should integrate with the Order & Fleet Management System (OFMS) and Messaging & Fleet Monitoring System. If possible, the electronic freight market must be electronically connected. The system should be user friendly for schedulers. Graphical use interfaces (GUI) should be provided.

References

Aguirre, H. and Tanaka, K.: 2002, Parallel Varying Mutation in Deterministic and Self-adaptive GAs, *in* J. Guervos, P. Adamidis, H.-G. Beyer, H.-P. Schwefel and J.-L. Fernández-Villacañas (eds), *Parallel Problem Solving from Nature -PPSN VII*, Vol. 2439 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 111–121.

Ai, T. J. and Kachitvichyanukul, V.: 2009, A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery, *Computers & Operations Research* **36**(5), 1693–1702.

Applegate, D. L. (ed.): 2007, *The travelling salesman problem: a computational study*, Princeton Series in Applied Mathematics, Princeton University Press.

Atrill, P. and McLaney, E.: 2009, *Management Accounting for Decision Makers*, 6th edn, Prentice Hall.

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

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P. and Vance, P. H.: 1998, Branch-And-Price: Column Generation for Solving Huge Integer Programs, *Oper. Res.* **46**(3), 316–329.

Bent, R. and Van Hentenryck, P.: 2004, A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows, *Transportation Science* **38**(4), 515–530.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I. and Laporte, G.: 2006, Static Pickup and Delivery Problems: A ClassiÂ¯cation Scheme and Survey.

Berger, J. and Barkaoui, M.: 2004, A parallel hybrid genetic algorithm for the vehicle routing problem with time windows, *Computers & Operations Research* **31**(12), 2037–2053.

Berger, J., Barkaoui, M. and Braysy, O.: 2003, A route-directed hybrid genetic approach for the vehicle routing problem with time windows, *Information Systems and Operational Research* **41**, 179–194.

Bhatti, R. S., Kumar, P. and Kumar, D.: 2010, A Fuzzy AHP Model for 3PL Selection in Lead Logistics Provider Scenarios, *in* S. Parthasarathy. Hershey (ed.), *Enterprise Information Systems and Implementing IT Infrastructures: Challenges and Issues*, IGI Global, pp. 261–277.

Blum, C., Puchinger, J., Raidl, G. R. and Roli, A.: 2011, Hybrid metaheuristics in combinatorial optimization: A survey, *Applied Soft Computing* **11**(6), 4135–4151.

Blum, C. and Roli, A.: 2003, Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Comput. Surv.* **35**(3), 268–308.

Braysy, O.: 2012, Discussion at 25th European Conference on Operational Research.

Braysy, O., Berger, J., Barkaoui, M. and Dullaert, W.: 2003, A threshold accepting metaheuristic for the vehicle routing problem with time windows, *Central European journal of operations research* **4**.

Braysy, O., Porkka, P. P., Dullaert, W., Repoussis, P. P. and Tarantilis, C. D.: 2009, A well-scalable metaheuristic for the fleet size and mix vehicle routing problem with time windows, *Expert Systems with Applications* **36**(4), 8460–8475.

Chu, P.: 1997, *A Genetic Algorithm Approach for Combinatorial Optimisation Problems*, PhD thesis, Imperial College of Science, Technology and Medicine, London.

Confessore, G., Galiano, G. and Stecca, G.: 2008, An Evolutionary Algorithm for Vehicle Routing Problem with Real Life Constraints, *in* M. Mitsuishi, K. Ueda and F. Kimura (eds), *Manufacturing Systems and Technologies for the New Frontier*, Springer London, pp. 225–228.

Cooper, R. and Kaplan, R. S.: 1988, Measure Costs Right: Make the Right Decisions, *Harvard Business Review* **September-**, 96–103.

Cordeau, J.-F.: 2006, A Branch-and-Cut Algorithm for the Dial-a-Ride Problem, *Oper. Res.* **54**(3), 573–586.

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

Creput, J.-C., Koukam, A., Kozlak, J. and Lukasik, J.: 2004, An Evolutionary Approach to Pickup and Delivery Problem with Time Windows, pp. 1102–1108.

Dantzig, G. B. and Ramser, J. H.: 1959, The Truck Dispatching Problem, *Management Science* **6**.

Davis, L.: 1991, *Handbook of genetic algorithms*, Van Nostrand Reinhold, New York.

Dawkins, R.: 1976, *The Selfish Gene*, Oxford University Press, Oxford.

Derigs, U., Kurowsky, R. and Vogel, U.: 2011, Solving a real-world vehicle routing problem with multiple use of tractors and trailers and EU-regulations for drivers arising in air cargo road feeder services, *European Journal of Operational Research* **213**(1), 309–319.

Desaulniers, G., Desrosiers, J., Erdmann, A., Solomon, M. M. and Soumis, F.: 2002, VRP with Pickup and Delivery, *The Vehicle Routing Problem*, chapter 9, pp. 225–244.

Diana, M. and Dessouky, M. M.: 2004, A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows, *Transportation Research Part B: Methodological* **38**(6), 539–557.

Dondo, R., Méndez, C. A. and Cerdá, J.: 2007, *Solving Multiple Vehicle Pickup and Delivery Problems in Multisite Systems by a Rigorous Optimization Approach*, Wiley-VCH Verlag GmbH & Co. KGaA, chapter 2, pp. 29–60.

Dorigo, M. and Stutzle, T.: 2004, *Ant Colony Optimization*, The MIT Press, Cambridge, Massachusett.

Drexl, M.: 2012, Rich vehicle routing in theory and practice, *Logistics Research* **5**(1-2), 47–63.

Drury, C.: 2005, *Management Accounting for Business*, 3rd edn, Patrick Bond.

Dueck, G. and Scheuer, T.: 1990, Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing, *Journal of Computational Physics* **90**(1), 161–175.

Falkenauer, E.: 1998, *Genetic Algorithms and Grouping Problems*, John Wiley & Sons, Inc., New York, NY, USA.

Gattorna, J., Ogulin, R. and Selen, W.: 2004, An empirical investigation of 3rd- and 4th-party logistics provider practices in Australia, *2nd ANZAM Operations Management Symposium*, Australia.

Gattorna, J., Selen, W. and Ogulin, R.: 2004, Characteristics, strategies and trends for 3PL/4PL in Australia, *Technical report*, Alpha Research Consortium, Sydney, Australia.

Gendreau, M. and Tarantilis, C. D.: 2010, Solving Large-Scale Vehicle Routing Problems with Time Windows: The State-of-the-Art, *Technical report*, CIRRELT.

Gillett, B. E. and Johnson, J.: 1974, Sweep algorithm for the multiple depot vehicle dispatch problem, *46th National Meeting of ORSA*, San Juan, Puerto Rico.

Glover, F.: 1986, Future paths for Integer Programming and Links to Artificial Intelligence, *Computers and Operations Research* **5**, 533–549.

Glover, F. and Kochenberger, G. (eds): 2003, *Handbook of Metaheuristics*, An Introduction to Tabu Search, Kluwer's International Series.

Goel, A. (ed.): 2008, *Fleet Telematics: Real-Time Management and Planning of Commercial Vehicle Operations*, Springer Sciene+Business Media, LLC.

Goel, A. and Gruhn, V.: 2008, A General Vehicle Routing Problem, *European Journal of Operational Research* **191**(3), 650–660.

Goldberg, D. E.: 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company Inc.

Goldberg, D. E.: 2002, *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*, Kluwer Academic Publishers, Boston/Dordrecht/London.

Golden, B. L. and Assad, A. A. (eds): 1988, *Vehicle Routing: Methods and Studies*, Vol. 16, Amsterdam: North-Holland.

Golden, B., Magnanti, T. and Nguyen, H.: 1977, Implementing Vehicle Routing Algorithm, *Networks* **7**, 113–148.

Goss, S., Aron, S., Deneubourg, J. L. and Pasteels, J. M.: 1989, Self-organized shortcuts in the Argentine ant, *Naturwissenschaften* **76**(12), 579–581.

Grefenstette, J. J.: 1987, Incorporating problem specific knowledge into genetic algorithms, Morgan Kaufmann, Los Altos, California, chapter 4, pp. 42–60.

Greistorfer, P. and Voß, S.: 2005, Controlled pool maintenance in combinatorial optimization, *in* C. Rego and B. Alidaee (eds), *Metaheuristic Optimization via Memory and Evolution-Tabu Search and Scatter Search*, Springer, pp. 382–424.

Gruenert, T.: 2012, GIS and Digital Maps for Routing in Commercial Decision Support Systems.

Gutin, G. and Karapetyan, D.: 2008, A Memetic Algorithm for the Generalized Traveling Salesman Problem, *European Journal Of Operational Research* **174**(1), 15.

Hang, X., Zhi-Long, C., Rajagopal, S. and Arunapuram, S.: 2003, Solving a Practical Pickup and Delivery Problem, *Transportation Science* **37**(3), 347–364.

Hart, W., Krasnogor, N. and Smith, J.: 2005, Memetic evolutionary algorithms, *Recent advances in memetic algorithms* pp. 3–27.

Hashimoto, H., Ibaraki, T., Imahori, S. and Yagiura, M.: 2006, The vehicle routing problem with flexible time windows and traveling times, *Discrete Applied Mathematics* **154**(16), 2271–2290.

Hashimoto, H., Yagiura, M. and Ibaraki, T.: 2008, An iterated local search algorithm for the time-dependent vehicle routing problem with time windows, *Discret. Optim.* **5**(2), 434–456.

Hasle, G. and Kloster, O.: 2007, Industrial Vehicle Routing, *Geometric Modelling, Numerical Simulation, and Optimization*, chapter Industrial, pp. 397–435.

Hillier, F. S. and Lieberman, G. J.: 2010, Introduction to Operations Research.

Ho, W., Ho, G. T. S., Ji, P. and Lau, H. C. W.: 2008, A hybrid genetic algorithm for the multi-depot vehicle routing problem, *Engineering Applications of Artificial Intelligence* **21**(4), 548–557.

Hosny, M. I.: 2010, *Investigating Heuristic and Meta-Heuristic Algorithms for Solving Pickup and Delivery Problems*, PhD thesis, Cardiff University.

Hosny, N. I. and Mumford, C. L.: 2007, Single Vehicle Pickup and Delivery with Time Windows: Made to Measure Genetic Encoding and Operators, *GECCO's 07*, ACM, London, England, UK.

Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T. and Yagiura, M.: 2005, Effective Local Search Algorithms for Routing and Scheduling Problems with General Time-Window Constraints, *Transportation Science* **39**(2), 206–232.

Ibaraki, T., Imahori, S., Nonobe, K., Sobue, K., Uno, T. and Yagiura, M.: 2008, An iterated local search algorithm for the vehicle routing problem with convex time penalty functions, *Discrete Applied Mathematics* **156**(11), 2050–2069.

Jack J. Dongarra: 2012, Performance of Various Computers Using Standard Linear Equations Software, *Technical report*, University of Tennessee Computer Science.

Jaw, J.-J., Odoni, A. R., Psaraftis, H. N. and Wilson, N. H. M.: 1986, A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows, *Transportation Research Part B: Methodological* **20**(3), 243–257.

Jayalakshmi, G. A., Sathiamoorthy, S. and Rajaram, R.: 2001, A Hybrid Genetic Algorithm - A New Approach to Solve Traveling Salesman Problem, *International Journal of Computational Engineering Science* **2**(2), 339.

Jih, W.-r., Kao, C.-Y. and Hsu, J. Y.-j.: 2002, Using family competition genetic algorithm in pickup and delivery problem with time window constraints, *Proceeding of the 2002 IEEE, International Symposium on Intelligent Control*, Vancouver, Canada, pp. 496–501.

Jones, T.: 1995, *Evolutionary algorithms, fitness landscapes and search*, PhD thesis, University of New Mexico, Albuquerque, NM.

Kaplan, R. S. and Anderson, S. R.: 2004, Time-Driven Activity-Based Costing, *Harvard Business Review* pp. 1–9.

Kaplan, R. S. and Anderson, S. R.: 2007, *Time-Driven Activity-Based Costing: A Simpler and More Powerful Path to Higher Profits*, Harvard Business School Press.

Kennedy, J. and Eberhart, R.: 1995, Particle swarm optimization, *Neural Networks, 1995. Proceedings., IEEE International Conference on*, Vol. 4, pp. 1942–1948 vol.4.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P.: 1983, Optimization by simulated annealing, *Science* **220**(4598), 671–680.

Krajewska, M. A. and Kopfer, H.: 2009, Transportation planning in freight forwarding companies: Tabu search algorithm for the integrated operational transportation planning problem, *European Journal of Operational Research* **197**(2), 741–751.

Krajewska, M. A., Kopfer, H., Laporte, G., Ropke, S. and Zaccour, G.: 2008, Horizontal cooperation among freight carriers: request allocation and profit sharing.

Krasnogor, N.: 2005, Towards robust memetic algorithms, *Recent advances in memetic algorithms* pp. 185–207.

Krasnogor, N. and Smith, J.: 2005, A tutorial for competent memetic algorithms: model, taxonomy, and design issues, *Evolutionary Computation, IEEE Transactions on* **9**(5), 474–488.

Kytojoki, J., Nuortio, T., Bräysy, O. and Gendreau, M.: 2007, An efficient variable neighborhood search heuristic for very large scale vehicle routing problems, *Computers & Operations Research* **34**(9), 2743–2757.

Langley, C. J., Allen, G. R. and Dale, T. A.: 2004, Third-Party Logistics Study: Results and Findings of the 2004 Sixth Annual Study, *Technical report*, Georgia Institute of Technology.

Langley, C. J., Allen, G. R. and Dale, T. A.: 2009, The State of Logistics Outsourcing: 2009 Third-Party Logistics-Results and Findings of the 14th Annual Study, *Technical report*, Geogrgia Institute of Technology.

Langley, C. J., Allen, G. R. and Tyndall, G. R.: 2000, Third-Party Logistics Study: Results and Findings of the 2001 Sixth Annual Study, *Technical report*, Georgia Institute of Technology.

Laporte, G.: 2000, Classical and Modern Heuristics for the Vehicle Routing Problem, *International Transactions in Operational Research* **7**, 200–285.

Laporte, G.: 2007, What You Should Know about the Vehicle Routing Problem, *Naval Research Logistics (NRL)* **54**(8), 811–819.

Lau, H. C. and Liang, Z.: 2001, Pickup and delivery with time windows: algorithms and test case generation, *Tools with Artificial Intelligence, Proceedings of the 13th International Conference on*, pp. 333–340.

Lau, H. C. W., Chan, T., Tsui, W. T. and Pang, W. K.: 2010, Application of Genetic Algorithms to Solve the Multidepot Vehicle Routing Problem, *Automation Science and Engineering, IEEE Transactions on* **7**(2), 383–392.

Lawrence, B.: 1983, Routing and scheduling of vehicles and crews, *An international journal: computers and operations research* **10**(2).

Lee, L. H., Tan, K. C., Ou, K. and Chew, Y. H.: 2003, Vehicle capacity planning system: a case study on vehicle routing problem with time windows, *Trans. Sys. Man Cyber. Part A* **33**(2), 169–178.

Lenstra, J. K. and Kan, A. H. G. R.: 1981, Complexity of vehicle routing and scheduling problems, *Networks* **11**, 221–227.

Li, H. and Lim, A.: 2001, A metaheuristic for the pickup and delivery problem with time windows, *Proceedings of 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 01)*, Dallas, USA.

Lin, S.-W., Yu, V. F. and Chou, S.-Y.: 2009, Solving the truck and trailer routing problem based on a simulated annealing heuristic, *Computers & Operations Research* **36**(5), 1683–1692.

Liu, F.-H. F. and Shen, S.-Y.: 1999, A route-neighborhood-based metaheuristic for vehicle routing problem with time windows, *European Journal of Operational Research* **118**(3), 485–504.

Liu, Y.-H.: 2011, A genetic local search algorithm with a threshold accepting mechanism for solving the runway dependent aircraft landing problem, *Optimization Letters* **5**(2), 229–245.

Lokin, F. C. J.: 1978, Procedures for traveling salesman problems with additional constraints, *European Journal of Operational Research* **3**, 135–141.

Lozano, M., Herrera, F. and Cano, J. R.: 2008, Replacement strategies to preserve useful diversity in steady-state genetic algorithms, *Information Sciences* **178**(23), 4421–4433.

Mahfoud, S. W. and Goldberg, D. E.: 1995, Parallel recombinative simulated annealing: A genetic algorithm, *Parallel Computing* **21**(1), 1–28.

Martin, O., Otto, S. and Felten, E. W.: 1991, Large-step Markov chains for the traveling salesman problem, *Complex Systems* **5**(3), 299–326.

Matthew, A. R. and Gary, B. L.: 2005, A genetic algorithm for unmanned aerial vehicle routing.

Merz, P. and Freisleben, B.: 1999, A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem, *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, Vol. 3, pp. 3 vol. (xxxvii+2348).

Mester, D. and Bräysy, O.: 2005, Active guided evolution strategies for large-scale vehicle routing problems with time windows, *Computers and Operations Research* **32**(6), 1593–1614.

Mladenovic, N. and Hansen, P.: 1997, Variable neighborhood search, *Computers & Operations Research* **24**(11), 1097–1100.

Moolman, A. J., Koen, K. and der Westhuizen, J. V.: 2010, Activity-Based Costing for Vehicle Routing Problems, *South African Journal of Industrial Engineering* **21**(2), 161–171.

Moon, C., Kim, J., Choi, G. and Seo, Y.: 2002, An efficient genetic algorithm for the traveling salesman problem with precedence constraints, *European Journal of Operational Research* **140**(3), 606–617.

Moscato, P.: 1989, On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts - Towards Memetic Algorithms.

Moscato, P. and Cotta, C.: 2003, A Gentle Introduction to Memetic Algorithms, *Handbook of Metaheuristics*, chapter 5, pp. 105–144.

Nagata, Y., Bräysy, O. and Dullaert, W.: 2010, A penalty-based edge assembly memetic

algorithm for the vehicle routing problem with time windows, *Computers and Operations Research* **37**(4), 724–737.

Nagata, Y. and Kobayashi, S.: 2010, A Memetic Algorithm for the Pickup and Delivery Problem with Time Windows Using Selective Route Exchange Crossover, *in* R. Schaefer, C. Cotta, J. Kolodziej and G. Rudolph (eds), *PPSN (1)*, Vol. 6238 of *Lecture Notes in Computer Science*, Springer, pp. 536–545.

Nagy, G. and Salhi, S.: 2005, Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries, *European Journal of Operational Research* **162**, 126–141.

Nanry, W. and Barnes, J.: 2000, Solving the pickup and delivery problem with time windows using reactive tabu search, *Transportation Research Part B: Methodological* **34**(2), 107–121.

Ngueveu, S. U., Prins, C. and Calvo, R. W.: 2010, An effective memetic algorithm for the cumulative capacitated vehicle routing problem, *Computers & Operations Research* **37**(11), 1877–1885.

Nguyen, Q. H., Ong, Y. S. and Krasnogor, N.: 2007, A study on the design issues of Memetic Algorithm, *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pp. 2390–2397.

Ombuki, B. and Hanshar, F.: 2009, Using Genetic Algorithms for Multi-depot Vehicle Routing, *Bio-inspired Algorithms for the Vehicle Routing Problem*, pp. 77–99.

Ong, Y.-S. and Keane, A. J.: 2004, Meta-Lamarckian learning in memetic algorithms, *Evolutionary Computation, IEEE Transactions on* **8**(2), 99–110.

Ong, Y.-S., Lim, M.-H., Zhu, N. and Wong, K.-W.: 2006, Classification of adaptive memetic algorithms: a comparative study, *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* **36**(1), 141–152.

Pankratz, G.: 2005, A grouping genetic algorithm for the Pickup and Delivery Problem with Time Windows, *OR Spectrum* **27**, 21–41.

Parragh, S. N., Doerner, K. F. and Hartl, R. F.: 2008, A survey on pickup and delivery problems Part I: Transportation between customers and depot, *Journal für Betriebswirtschaft* **58**, 21–51.

Pellegrini, P., Favaretto, D. and Moretti, E.: 2007, Multiple Ant Colony Optimization for a Rich Vehicle Routing Problem: A Case Study, *Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Computer Science*, Vol. 4693, pp. 627–634.

Pisinger, D. and Ropke, S.: 2007, A general heuristic for vehicle routing problems, *Computers and Operations Research* **34**(8), 2403–2435.

Polacek, M., Hartl, R. F., Doerner, K. and Reimann, M.: 2004, A Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows, *Journal of Heuristics* **10**(6), 613–627.

Potvin, J.-Y.: 1996, Genetic Algorithms for the Traveling Salesman Problem, *Annals of Operations Research* **63**, 339–370.

Potvin, J.-Y. and Rousseau, J.-M.: 1993, A parallel route building algorithm for the vehicle routing and scheduling problem with time windows, *European Journal of Operational Research* **66**(3), 331–340.

Powell, W. B.: 2003, Dynamic Models of Transportation Operations, *in* S.C. Graves and A. G. de Kok (eds), *Supply Chain Management: Design, Coordination and Operation*, Vol. 11 of *Handbooks in Operations Research and Management Science*, Elsevier, pp. 677–756.

Raidl, G. R.: 2006, A unified view on hybrid metaheuristics, *Proceedings of the Third international conference on Hybrid Metaheuristics*, HM'06, Springer-Verlag, Berlin, Heidelberg, pp. 1–12.

Rekiek, B., Delchambre, A. and Saleh, H. A.: 2006, Handicapped Person Transportation: An application of the Grouping Genetic Algorithm, *Engineering Applications of Artificial Intelligence* **19**(5), 511–520.

Renaud, J., Boctor, F. F. and Laporte, G.: 2002, Perturbation heuristics for the pickup and delivery traveling salesman problem, *Computers and Operations Research* **29**(9), 1129–1141.

Ribeiro, G. M. and Laporte, G.: 2012, An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem, *Computers and Operations Research* **39**(3), 728–735.

Rocha, M. and Neves, J.: 1999, Preventing Premature Convergence to Local Optima in Genetic Algorithms via Random Offspring Generation, *Multiple Approaches to Intelligent Systems*, pp. 127–136.

Rodriguez-Diaz, F. J., Carlos, G.-M. and Lozano, M.: 2010, A GA-based multiple simulated annealing, *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pp. 1–7.

Ropke, S.: 2005, *Heuristic and exact algorithms for vehicle routing problems*, PhD thesis, University of Copenhagen.

Ropke, S.: 2009a, Computational Results for Multi-Depot PDPTW.
URL: *http://www.diku.dk/ sropke/*

Ropke, S.: 2009b, Parallel large neighborhood search - a software framework, *MIC 2009: The VIII Metaheuristics International Conference*.

Ropke, S. and Cordeau, J.-F.: 2006, Branch-and-Cut-and-Price for the Pickup and Delivery Problem with Time Windows, *Technical report*, University of Montreal, Canada.

Ropke, S. and Pisinger, D.: 2006, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, *Transportation Science* **40**, 455–472.

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* **50**, 1034–1042.

Salhi, S. and Sari, M.: 1997, A Multi-level composite heuristic for the multi-depot vehicle fleet mix problem, *European Journal of Operational Research* **103**, 95–112.

Savelsbergh, M. W. P. and Sol, M.: 1995, The general pickup and delivery problem, *Transportation Science* **29**, 17–29.

Schmid, V., Doerner, K. F. and Laporte, G.: 2013, Rich routing problems arising in supply chain management, *European Journal of Operational Research* **224**(3), 435–448.

Shaw, P.: 1997, A new local search algorithm providing high quality solutions to vehicle routing problems., *Technical report*, Department of Computer Science, University of Strathclyde.

Shaw, P.: 1998, Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems, *Principles and Practice of Constraint Programming CP98*, pp. 417–431.

Skok, M., Skrlec, D. and Krajcar, S.: 2000, The genetic algorithm method for multiple depot capacitated vehicle routing problem solving, *Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000. Proceedings. Fourth International Conference on*, Vol. 2, pp. 520–526 vol.2.

Solomon, M. M.: 1987, Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research* **35**, 254–265.

Sorensen, K.: 2012, Metaheuristics - the metaphor exposed. Presentation at 25th European Conference on Operational Research, Vilnius, Lithuania, 8-11 July.

Stutzle, T.: 1999, Iterated local search for the quadratic assignment problem, *Technical report*, FG Intellektik, TU Darmstadt.

Subramanian, A.: 2012, *Heuristic, Exact and Hybrid Approaches for Vehicle Routing Problems*, PhD thesis, Universidade Federal Fluminense, Niteroi, Brazil.

Sukhotu, V.: 2011, Logistics Cost Structure, Distributed on 5 Aprill 2011, Lecture Notes in Supply Chain Leaders, Thailand.

Syswerda, G. (ed.): 1991, *Schedule Optimization Using Genetic Algorithm*, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York.

Talbi, E. G.: 2002, A Taxonomy of Hybrid Metaheuristics, *Journal of Heuristics* **8**(5), 541–564.

Talbi, E.-G.: 2009, *Metaheuristics - From Design to Implementation*, Wiley.

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

Thangiah, S. R. and Salhi, S.: 2001, Genetic clustering: An adaptive heuristic for the multi-depot vehicle routing problem, *Applied Artificial Intelligence* **15**, 361–383.

Toth, P. and Vigo, D. (eds): 2002, *The Vehicle Routing Problem*, SIAM monographs on discrete mathematics and applications.

Ulder, N. L. J., Aarts, E. H. L., Bandelt, H.-J., van Laarhoven, P. J. M. and Pesch, E.: 1991, Genetic Local Search Algorithms for the Travelling Salesman Problem, *Proceedings of the 1st Workshop on Parallel Problem Solving from Nature*, PPSN I, Springer-Verlag, London, UK, UK, pp. 109–116.

Vidal, T., Crainic, T. G., Gendreau, M. and Prins, C.: 2013, A Unified Solution Framework for Multi-Attribute Vehicle Routing Problems, *Technical report*, CIRRELT, Montreal, Canada.

Wen, M.: 2010, *Rich Vehicle Routing Problems and Applications*, PhD thesis, Technical University of Denmark.

Yagiura, M. and Ibaraki, T.: 2001, On metaheuristic algorithms for combinatorial optimization problems, *Systems and Computers in Japan* **32**(3), 33–55.

Yu, B., Yang, Z.-Z. and Yao, B.: 2009, An improved ant colony optimization for vehicle routing problem, *European Journal of Operational Research* **196**(1), 171–176.

Zhu, K. Q.: 2003, A Diversity-Controlling Adaptive Genetic Algorithm for the Vehicle Routing Problem with Time Windows.

Zhu, K. Q. and Liu, Z.: 2004, Population Diversity in Permutation-Based Genetic Algorithm, *Machine Learning: ECML 2004*, pp. 537–547.

Appendix A

Lemma and Proof for Reduction Rule of Time Windows

Fix-forward Insertion using Reduction Rule for Time Windows

From Equation 4.18 and 4.19, we designed Fix-forward Insertion using Reduction Rule for Time Window in Figure A.1.

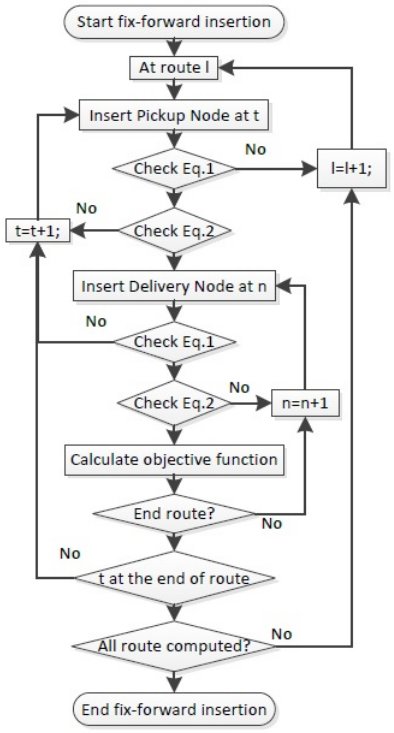


Figure A.1: Fix-forward Insertion using Reduction Rule of Time Windows

From Figure A.1, there are four decisions on time feasibility checking for pickup and delivery problems with time windows:(1) at pickup node checking Eq.1, (2) at pickup node checking Eq.2, (3) at delivery node checking Eq.1, (4) at delivery node checking Eq.2. We describe the rationale of these rules in Figure A.2 and Figure A.3 in form of data structure and network structure.

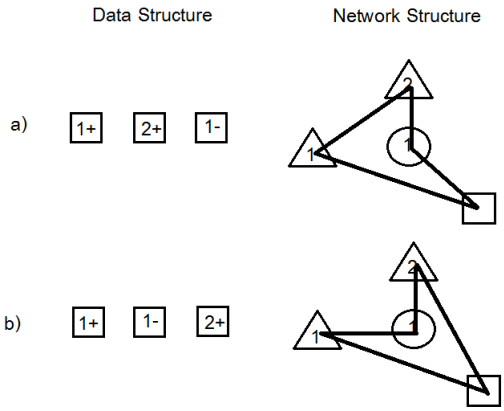


Figure A.2: Figure illustrating for 1^{st} and 2^{nd} decisions

For the 1^{st} decision, in Figure A.2, if the Check Eq.1 at pickup node 2 (triangle No.2) in a) returns NO, then the search starts at the next route. Assume that Eq1. is violated at pickup node 2. After the fix-forward insertion from a) to b), the network structure is shown in network structure of b). As the triangle inequality holds, using fix-forward for pickup node 2 will always violate Eq.1.

For the 2^{nd} decision, in Figure A.2, even though the Check Eq.2 at delivery node 1 in a) is violated, after the fix-forward insertion from a) to b), it is possible to search by fix-forward method further because it can reduces the arrival time at delivery node 1.

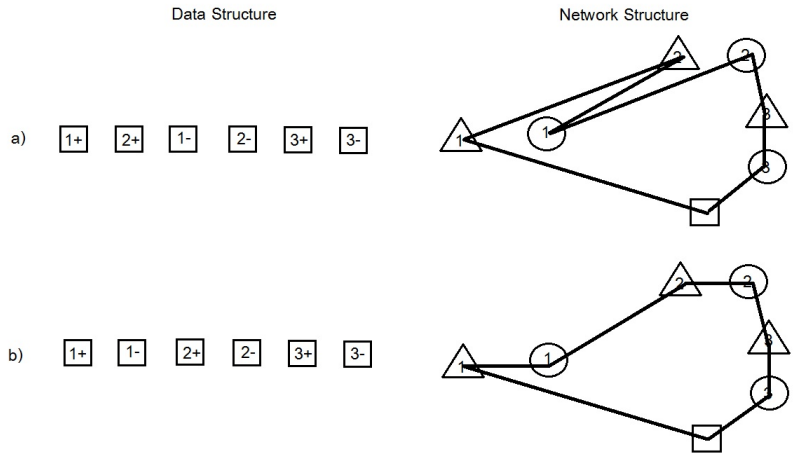


Figure A.3: Figure illustrating for 3^{rd} and 4^{th} decisions

For the 3^{rd} decision, in Figure A.3, assume that the delivery node 2 (circle No.2) is violated by Eq. 1 or return NO at Check Eq.1. It is possible that fix-forward insertion which sequences the route well enable delivery node 2 feasible again as shown in b).

For the 4^{th} decision, in Figure A.3, assume that the pickup node 3 (triangle No.3) is violated by Eq.2. The better sequencing due to fix-forward can reduce the arrival time in pickup node 3.

Appendix B

Comparison of Run Times for the AMLNS

The CPU time of the AMLNS in Chapter 4 can be scaled into the equivalent AMD Opteron 250 2.4GHz. No (flop/s) measure could be found for Intel Core i7 3.5 GHz in Jack J. Dongarra (2012). Therefore, we made the assumption that the processor should be approximately linear with frequency among the processors from the same family. The computational time of the AMLNS is multiplied by 1.44.

Authors	Processor	MFlop/s	Factor
Ropke and Pisinger (2006)	AMD Opteron 250 2.4 GHz	1291	**1.0**
Pisinger and Ropke (2007)	Intel Pentium IV 3.0 GHz	1573	1.22
AMLNS for Chapter 4	Intel Core i7 3.5 GHz	-	**1.44**

Table B.1: Scaling factors for computational time

Appendix C

New Best Known Solutions on Ropke and Pisinger's (2006) Instances

In this appendix, some new best known solutions obtained from the AMLNS are provided for Ropke and Pisinger's (2006) test instances for the multi-depot PDPTW. For other new best known solutions, they can be obtained from us.

Two Best Known Solutions of Problem 50K

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

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

Figure C.1: Two Best Known Solutions of Problem 50K

New Best Known Solution of 100 B

Vehicle	Visit sequence
0	
1	190 191 0 110 4 111 5 188 174 189 1 175
2	
3	
4	56 192 48 122 123 176 177 193 164 165 49 57
5	
6	
7	
8	
9	98 99 54 156 50 55 126 82 51 157 127 70 83 71
10	80 74 52 53 124 166 125 167 81 182 75 183
11	
12	92 66 78 79 58 93 59 22 8 67 186 23 187 9
13	
14	170 40 128 64 129 104 105 41 171 65
15	
16	138 132 2 139 3 114 115 20 90 133 21 91
17	72 76 46 14 15 26 27 77 44 73 47 45
18	84 100 101 160 68 85 69 161
19	62 178 106 179 63 102 172 173 107 103 34 35
20	6 30 134 31 154 148 7 38 39 135 149 155
21	
22	
23	96 136 32 97 137 184 33 185
24	120 158 121 94 95 16 108 118 109 119 159 17
25	116 146 117 88 147 144 196 197 152 89 153 145 142 143
26	150 36 168 151 169 180 18 24 19 25 12 181 13 37
27	198 60 130 10 199 131 86 87 140 61 11 141
28	28 42 29 194 112 162 113 43 163 195
29	
Obj	106248.99

Table C.1: Best Known Solutions of Problem 100B

New Best Known Solution of 250 C

Vehicle	Visit sequence
0	470 254 228 255 296 80 297 471 81 246 86 87 247 229
1	
2	
3	184 222 322 185 396 397 498 476 499 96 477 276 97 323 223 277
4	400 302 401 56 438 303 366 367 439 57
5	28 29 440 474 441 180 294 181 104 475 240 105 295 241
6	414 22 356 164 415 232 357 23 165 233
7	
8	
9	172 14 15 114 480 481 4 173 115 5 252 253
10	48 212 49 442 256 108 257 443 109 402 403 213
11	
12	324 54 325 90 360 98 361 99 55 91 334 335
13	2 3 466 376 242 377 243 154 467 170 155 171
14	62 344 406 332 345 272 407 333 273 63
15	
16	
17	204 88 89 205 326 428 20 450 451 426 429 427 21 327
18	416 244 220 245 206 152 153 221 417 207 64 65
19	
20	178 179 192 262 263 193 216 446 162 447 202 203 217 163
21	
22	134 140 190 226 227 424 425 58 135 260 261 191 141 59
23	368 78 369 200 201 478 166 479 18 167 19 79
24	
25	8 9 490 482 94 236 483 112 430 95 237 113 431 491
26	444 310 311 384 410 411 74 75 116 456 385 117 26 27 457 445
27	488 486 330 364 218 365 219 331 10 11 388 489 389 487
28	
29	300 304 398 208 305 188 496 497 399 301 209 189

Table C.2: Best Known Solutions of Problem 250C: Route 0-29

Vehicle	Visit sequence
30	
31	
32	
33	
34	348 454 12 13 32 274 455 33 275 238 168 210 349 211 239 169
35	158 464 465 346 38 159 39 314 315 347
36	
37	
38	
39	422 198 320 423 199 150 321 174 151 175
40	
41	468 469 362 270 144 363 271 145
42	
43	282 283 186 358 70 187 412 71 359 160 161 413
44	
45	
46	
47	372 374 46 375 373 472 47 404 390 354 473 405 355 391
48	52 318 319 53 452 132 340 341 286 287 453 133
49	
50	394 68 196 395 197 69 352 6 353 492 7 493
51	224 264 328 265 336 225 82 83 329 337
52	386 382 146 147 122 24 25 420 421 448 449 383 123 387
53	380 126 34 278 127 381 136 182 137 35 183 268 269 279
54	284 106 285 128 129 292 266 267 110 293 42 111 43 107
55	16 298 17 342 299 230 231 36 37 343
56	
57	
58	
59	

Table C.3: Best Known Solutions of Problem 250C: Route 30-59

Vehicle	Visit sequence
60	
61	
62	
63	
64	
65	
66	250 0 234 235 1 308 156 251 309 157
67	
68	392 458 393 76 84 77 248 142 143 370 85 371 459 249 60 61
69	44 408 409 130 494 495 432 45 258 131 259 433
70	436 462 92 437 484 463 66 93 485 194 195 148 67 149 214 215
71	378 50 51 290 138 139 379 291
72	
73	
74	338 40 350 288 41 351 460 339 280 461 281 289
Obj	244974.17

Table C.4: Best Known Solutions of Problem 250C: Route 60-74

New Best Known Solution of 500 D

Vehicle	Visit sequence
0	910 800 911 980 252 801 130 981 131 253
1	
2	
3	330 754 286 700 287 906 701 907 72 331 940 73 755 941
4	162 354 163 350 351 258 259 355 154 155 74 760 75 761
5	
6	
7	
8	530 774 426 427 176 390 391 788 789 934 935 531 772 177 773 775
9	
10	
11	
12	714 504 505 446 447 948 362 949 836 837 212 363 715 213
13	734 735 614 262 740 263 741 962 856 576 615 577 963 332 857 333
14	266 684 462 914 463 912 180 913 646 685 181 647 915 267
15	508 470 542 932 933 676 471 677 509 543
16	324 850 851 756 558 746 747 872 325 757 559 873
17	328 329 224 6 225 7 978 108 979 686 546 547 109 687
18	412 612 528 413 344 613 538 650 651 744 745 345 529 539
19	
20	
21	
22	628 40 592 936 937 88 41 682 593 683 629 89
23	
24	
25	554 824 190 398 399 825 406 498 499 968 407 555 191 969
26	904 894 720 905 670 278 444 721 895 736 671 445 737 279
27	322 392 393 323 568 468 469 569
28	810 464 811 2 726 372 727 306 3 373 307 718 465 719
29	

Table C.5: Best Known Solutions of Problem 500D: Route 0-29

Vehicle	Visit sequence
30	
31	
32	500 66 501 67
33	
34	
35	410 492 411 564 876 565 608 877 732 733 493 609
36	
37	
38	870 798 244 871 245 784 428 140 141 310 799 311 429 785
39	
40	
41	708 414 552 415 432 688 433 689 709 553 86 404 172 87 405 173
42	
43	394 54 395 296 476 297 477 368 302 369 562 618 619 563 303 55
44	298 78 638 136 844 299 845 514 137 22 515 79 23 639
45	288 418 312 419 678 274 275 289 70 982 71 983 313 68 679 69
46	
47	890 228 80 92 891 81 920 229 188 189 160 93 921 532 161 533
48	458 574 294 459 575 942 295 304 943 44 590 591 305 45
49	
50	156 50 930 51 931 282 440 441 157 283 728 340 729 341
51	
52	60 512 61 520 521 490 828 82 829 491 454 513 455 83 988 989
53	
54	
55	602 752 603 716 717 556 814 815 790 290 291 76 753 77 557 791
56	178 150 998 402 999 403 634 635 526 527 179 780 781 151
57	42 648 43 616 922 20 923 21 46 617 308 47 309 649
58	
59	366 964 886 482 483 924 367 965 386 387 887 925

Table C.6: Best Known Solutions of Problem 500D: Route 30-59

Vehicle	Visit sequence
60	342 484 896 506 343 507 897 578 485 292 579 293
61	
62	
63	256 644 642 645 610 611 643 738 739 257
64	
65	
66	
67	
68	24 892 893 524 510 511 25 218 525 918 219 919 370 371
69	
70	
71	
72	110 111
73	
74	230 356 226 227 357 380 231 144 928 929 900 901 145 381
75	
76	58 122 778 123 94 596 95 597 534 535 214 215 59 779
77	
78	926 854 992 632 633 927 222 993 450 451 223 855
79	
80	832 840 420 62 794 722 841 723 63 833 884 885 421 795
81	
82	698 580 338 699 842 339 704 581 280 516 281 705 517 238 239 843
83	
84	
85	
86	
87	
88	
89	318 624 625 146 972 973 776 147 194 195 666 777 102 667 319 103

Table C.7: Best Known Solutions of Problem 500D: Route 60-89

Vehicle	Visit sequence
90	
91	
92	358 12 359 182 124 125 566 952 953 183 567 384 28 385 13 29
93	782 30 966 783 802 803 134 135 132 967 796 797 31 242 133 243
94	
95	
96	186 187 858 436 34 96 97 204 205 994 437 35 995 859
97	466 518 100 101 792 793 174 175 467 220 519 221
98	
99	348 652 660 661 326 848 849 148 480 653 349 481 149 327
100	
101	
102	762 763 862 863 882 216 768 486 217 52 769 53 487 883
103	316 198 430 431 694 199 866 695 820 867 184 317 821 185
104	834 18 860 4 861 364 90 91 365 835 990 19 5 991
105	334 8 838 916 839 335 584 917 588 585 448 449 9 589
106	
107	
108	
109	
110	770 494 272 495 248 771 273 236 138 139 237 730 731 249
111	846 847 898 118 119 114 115 38 899 818 39 819 570 571
112	
113	996 128 164 129 944 165 496 945 320 321 497 997
114	
115	626 627 36 260 64 472 473 65 37 360 261 361
116	
117	
118	868 816 664 986 665 987 946 947 817 830 210 211 438 158 439 831 869 159
119	

Table C.8: Best Known Solutions of Problem 500D: Route 90-119

Vehicle	Visit sequence
120	
121	
122	
123	104 826 954 955 827 416 456 105 417 457 478 434 479 435
124	594 662 26 27 710 711 663 196 766 197 595 956 168 169 957 767
125	206 112 346 232 347 113 864 400 233 401 207 865
126	548 120 254 121 255 878 240 241 549 879
127	16 974 106 975 14 17 15 674 675 107
128	976 812 813 276 902 903 10 234 235 977 376 377 277 11 880 881
129	586 587 352 374 166 353 758 759 375 522 658 167 523 659
130	970 971 630 314 804 264 805 56 265 631 250 251 315 208 209 57
131	
132	
133	806 422 423 606 600 807 601 488 192 607 193 489 396 397
134	152 170 874 706 707 750 751 572 573 950 951 171 153 875
135	246 668 938 247 669 640 641 0 1 939 908 742 743 909
136	378 654 379 680 598 681 984 655 808 985 809 599
137	958 474 560 712 959 475 452 536 537 453 822 823 713 561
138	
139	
140	
141	
142	
143	550 692 748 696 697 142 749 656 693 32 33 657 551 852 143 853
144	888 284 889 724 544 285 582 545 725 583 300 301
145	672 98 99 622 604 605 786 787 270 623 271 48 49 673
146	388 460 389 442 443 116 117 690 502 461 424 425 691 503
147	960 126 200 336 337 127 201 268 620 269 621 961 382 383
148	202 408 409 764 702 703 84 203 765 636 540 637 85 541
149	
Obj	482608.05

Table C.9: Best Known Solutions of Problem 500D: Route 120-149

The Structure of New Best Known Solution of 50E

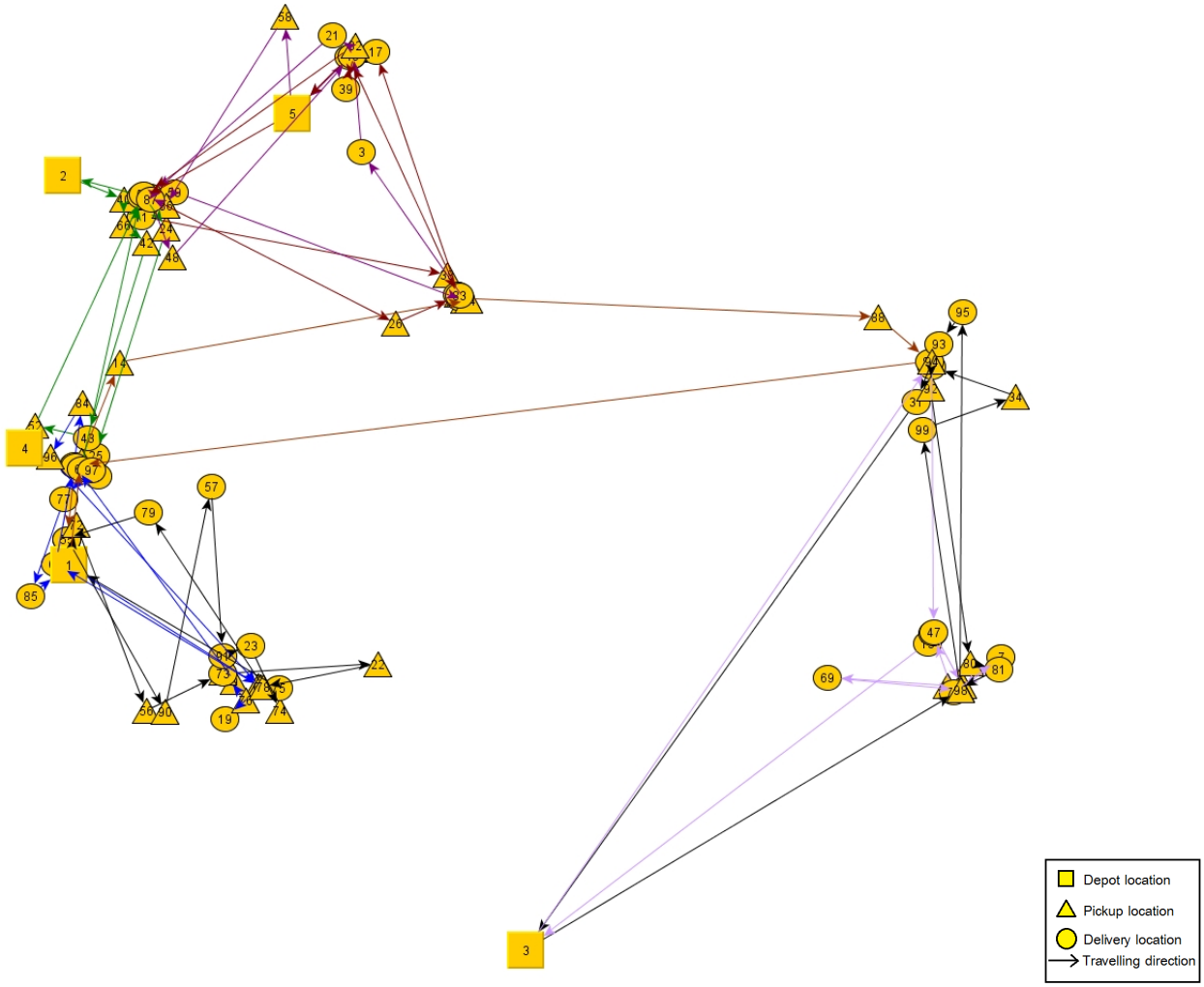


Figure C.2: The New Best Known Solution of Problem 50E (49923.61)

The Structure of New Best Known Solution of 50L

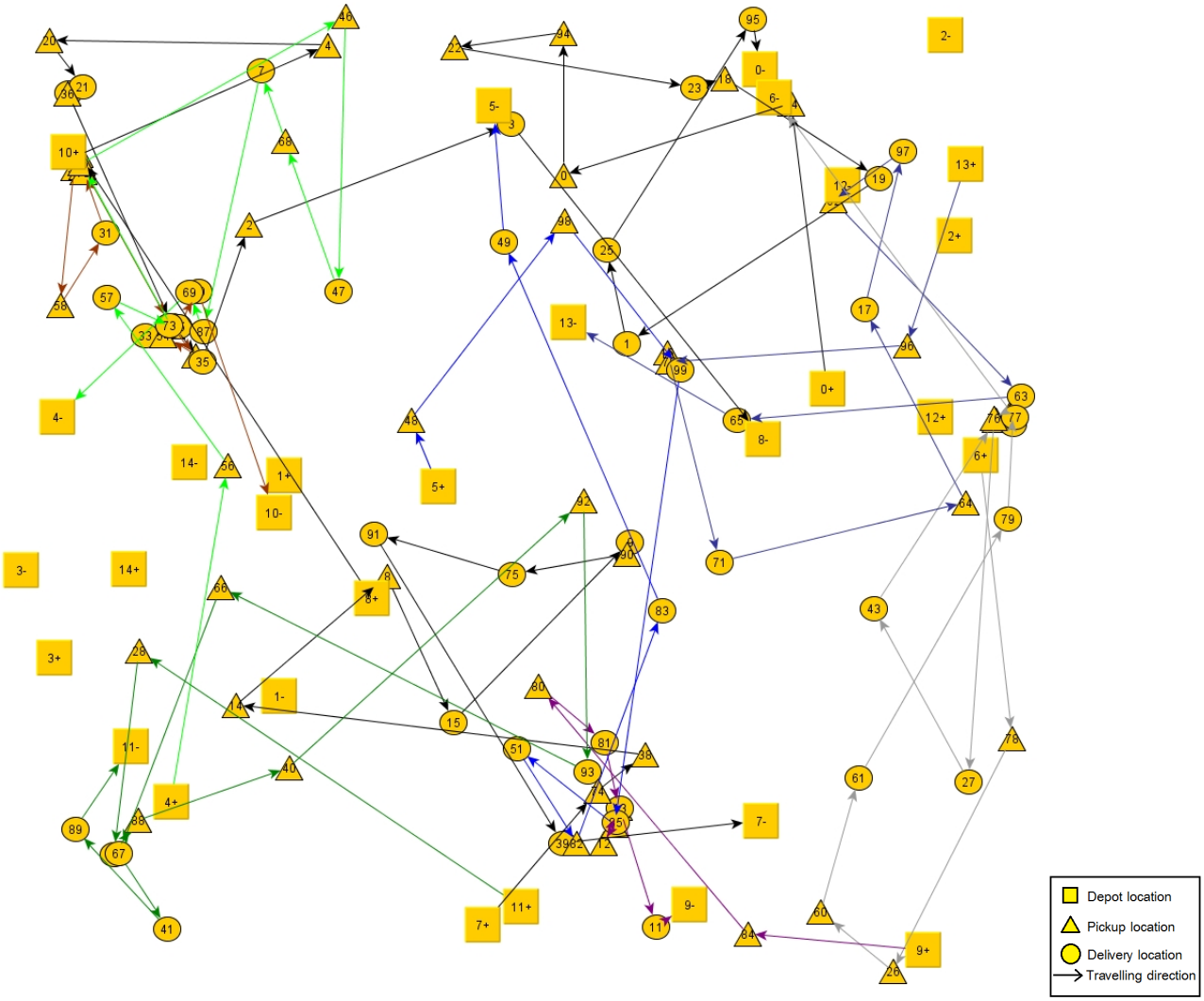


Figure C.3: The New Best Known Solution of Problem 50L (64936.76)

The Structure of New Best Known Solution of 50H

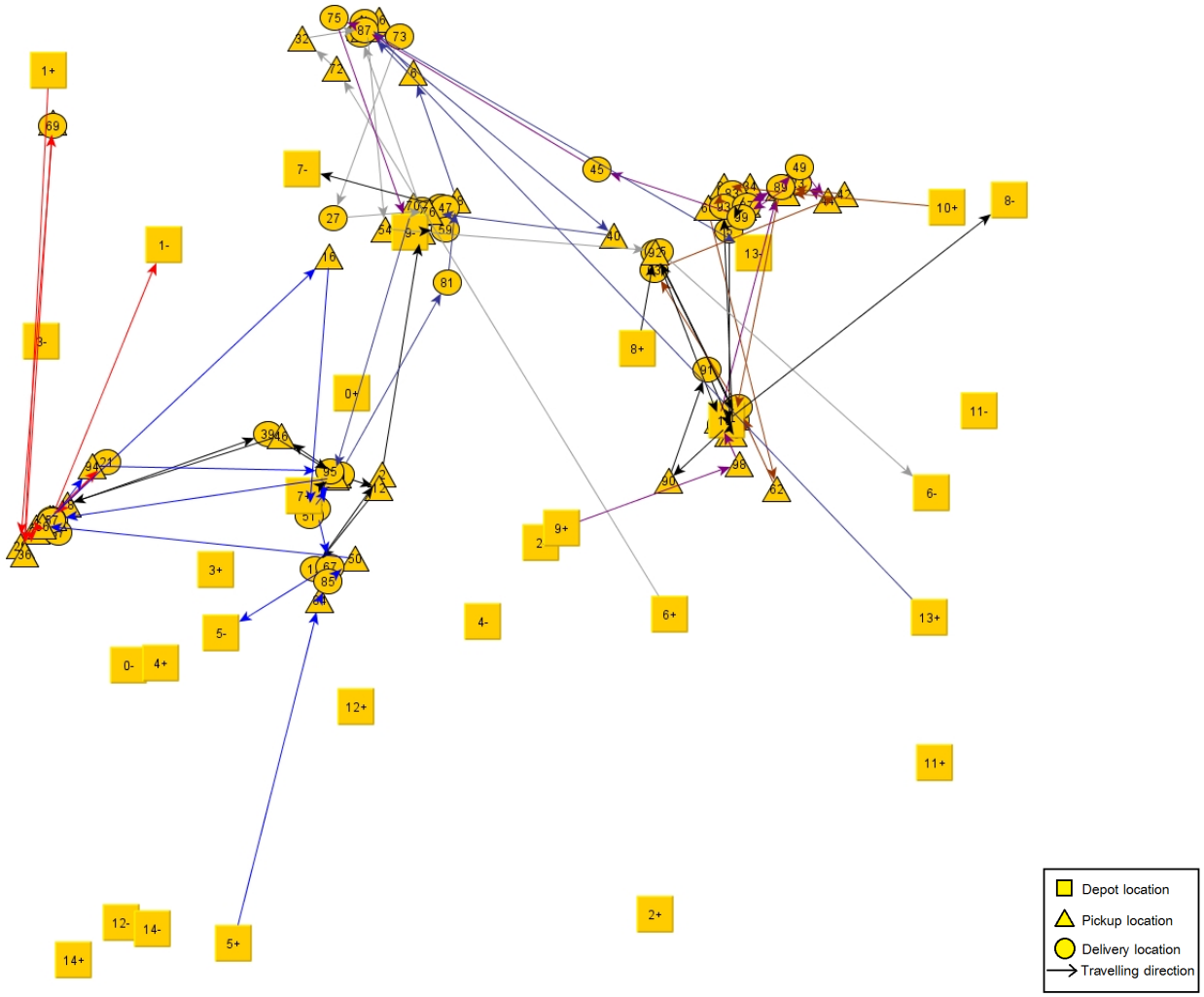


Figure C.4: The New Best Known Solution of Problem 50H (56761.36)

Appendix D

Capacity-Driven ABC for Road Freight Transport

Capacity-Driven ABC

Atrill and McLaney (2009) confirmed that the notion of fixed and variable costs are concerned with cost behaviour related to the changes in the volume of activity. The notion of direct and indirect, on the other hand, are concerned with the extent to which cost elements can be measured in respect of particular cost unit or job.

Kaplan and Anderson (2007) stated that the capacity cost rate is calculated as the ratio of departmental costs to practical capacity, to drive resource costs down on orders, and products. The numerator aggregates all the costs associated with a department, including the compensation of frontline employees and their supervisors. The denominator in the capacity cost rate calculation represents the practical capacity of the resources that perform work in the department. With numerator and denominator determined, the capacity cost rate is calculated by dividing the department's costs by the department's practical capacity.

According to the analysis, we construct the relationship between fixed & variable costs and direct & indirect costs in order to validate the use of CDABC to allocate fixed cost to job costing in the study, as shown in Table D.1.

	Fixed costs	Variable costs
Indirect costs	Rent Driver Monthly Salaries Welfare Equipment Depreciation Monthly Equipment Expense Utilities Insurance and License Maintenance Department General Administration Department Accouting and Finance Department Security Department Insurance and License Department	Truck and semi-trailer accessories Truck and semi-trailer tyres
Direct costs	Expense for each order, Container Lifting	Fuel cost wage per trip

Table D.1: Cost elements of Routing and Truck Operation

In Table D.1, each coordinate has different cost drivers. The cost driver of the elements in the fixed-indirect cost is the time. The cost driver of maintenance and tyres is the distance. The cost driver of fuel is the distance and weight. The cost driver of the expense is specific to each order. These cost drivers are integral parts of order costing. In conclusion, the order costing and route costing may apply different travel paths. In the fixed-indirect cost, the route costing applies the fixed cost per day while the order costing applies TDABC.

Indirect cost	Traveling Time of Each Order	Distance
Direct cost	Each Order	Distance and Weight

Table D.2: Cost rates corresponding to Table D.1 for order costing

Fixed cost	Variable cost
Fixed Cost per day	Distance
Each order	Distance and Weight

Table D.3: Cost rates corresponding to Table D.1 for daily operational planning

The application of Table D.2 and D.3 is different between route costing and order costing. In the order costing, the travelled distance and time is individually calculated from the travel from the depot and return to the depot. Since, the information of other

requests in the same planning horizon may be unknown due to its earliness. Moreover, the request may not be served for the whole day. Therefore, the cost must be allocated to a specific order.

However, when the daily operational routing is conducted, the information of all requests in the same planning horizon is known. Efficient routing methods can combine trips together in one trip and result in financial savings. Therefore, one vehicle may serve a number of requests in each trip. The fixed cost per day, derived from the fixed cost per month, of the vehicle still recurs regardless of its services.

Therefore, the direct and indirect costs are used to calculate the order costing. The fixed cost and variable costs are used to carry out the daily operational planning. Some cost rates of direct & indirect costs and fixed & variable costs are the same. From Table D.2 to Table D.3, the cost rates for distance and for distance and weight are the same for each order. Recall that the sum of all direct costs and indirect costs, as with the full cost, is typically supposed to be equal to the sum of fixed cost and variable cost. However, the savings from efficient algorithms can reduce the full cost for daily operational planning and become the essential tool for logistics companies.

Appendix E

Schedule of the Best Known Solution of Problem 50F (Chapter 7)

This section aims to illustrate the detailed schedules and routes for one test instance. The following constraints are satisfied: (1) precedence, (2) request time windows, (3) load due to road regulation, (4) load due to truck power, (5) volume capacity of trailer and (6) Special Requests. All requests are served. The truck's fixed cost is the sum of the fixed costs of the company's own truck. Also, the trailer fixed cost is the sum of the fixed costs of trailers. Even though some trucks are not used as shown in the Table E.1 to E.6, the fixed costs still incur. Moreover, the fixed costs of trucks and trailers recur even if the subcontractors' vehicles are used. The subcontractors, however, are not paid if they are not contacted to work. Table E.7 illustrates the summary of costs. The detailed schedules are shown from Table E.1 to Table E.6.

Costs	Value
Total Truck Fixed Cost	10200.6
Total Trailer Fixed Cost	11669.4
Total Fuel Cost	15559.13
Total Distance Cost	18264.68
Total PD Cost	7870
Total Requet Subcon Cost	8541.64
Total Tour Subcon Cost	1810.64
Objective Sunction	73916.09

Table E.7: Cost elements of the best known solution of problem 50E

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	L'(T+G)	Tot.L	S_V/D_V	V.
0	D0	-	-	0-660	0	0	0	0		75	45		80
23	46	86.5	86.5	80-330	0	64	150.5	17	17	24.435	33.931	40	40
	47	36.5	187	231-424	44	85	316	-17	0	7.44	16.93	-40	0
	90	18.64	334.64	118-377	0	46	380.64	7	7	14.43	23.93	50	50
	91	48.16	428.8	266-478	0	83	511.8	-7	0	7.44	16.93	-50	0
	D0	22.04	533.83	0-660	0	0	0	0	0				
	Truck FC.	658.8	Trailer FC.	324	Fuel C.	862.27	Dist. C.	928.34	PD. C.	410			
1	D0	-	-	0-660	0	0	0	0		30	35		80
22	52	80.92	80.92	2-252	0	60	140.92	11	11	18.945	24.038	70	70
	53	32.13	173.04	126-375	0	55	228.04	-11	0	7.94	13.04	-70	0
	76	11.1	239.14	161-391	0	45	284.14	12	12	19.94	25.04	70	70
	77	38.54	322.69	296-567	0	74	396.69	-12	0	7.94	13.04	-70	0
	88	24.51	421.2	353-568	0	34	455.2	16	16	23.94	29.04	70	70
	89	40.7	495.9	420-594	0	65	560.9	-16	0	7.94	13.04	-70	0
	D0	59.18	620.08	0-660	0	0	0	0	0				
	Truck FC.	550.8	Trailer FC.	340.2	Fuel C.	591.06	Dist. C.	886.39	PD. C.	500			
2	D0	-	-	0-660	0	0	0	0		70	45		80
1	2	63.92	63.92	14-212	0	87	150.92	20	20	26.706	35.177	70	70
	3	22.85	173.77	148-429	0	34	207.77	-20	0	6.71	15.18	-70	0
	44	48.39	256.17	268-477	11.83	37	305	9	9	15.71	24.18	80	80
	45	19.81	324.81	252-477	0	34	358.81	-9	0	6.71	15.18	-80	0
	26	12.27	371.07	175-384	0	41	412.07	6	6	12.71	21.18	50	50
	27	45.48	457.56	377-584	0	49	506.56	-6	0	6.71	15.18	-50	0
	D0	63.06	569.62	0-660	0	0	0	0	0				
	Truck FC.	599.4	Trailer FC.	318.6	Fuel C.	1031.98	Dist. C.	1128.47	PD. C.	500			
3	D0	-	-	0-660	0	0	0	0		70	50.5		80
25	78	41.22	101	101-322	59.78	76	177	19	19	25.639	34.201	20	20
	28	38.71	215.71	221-432	5.29	88	309	8	27	33.64	42.2	40	60
	79	5.85	314.85	382-594	67.15	87	469	-19	8	14.64	23.2	-20	40
	29	18.74	487.74	306-542	0	52	539.74	-8	0	6.64	15.2	-40	0
	D0	53.31	593.04	0-660	0	0	0	0	0				
	Truck FC.	550.8	Trailer FC.	340.2	Fuel C.	766.5	Dist. C.	782.05	PD. C.	330			

Table E.1: Detailed schedule of the best known solution of problem 50F: truck 0,1,2,3

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	$L^{'}$(T+G)	Tot.L	S_V/D_V	V.
4	D0	-	-	0-660	0	0	0	0		70	50.5		80
26	38	36.82	36.82	0-230	0	81	117.82	16	16	22.091	30.836	40	40
	30	40.1	157.92	0-191	0	87	244.92	8	24	30.09	38.84	30	70
	39	8.37	253.3	87-369	0	85	338.3	-16	8	14.09	22.84	-40	30
	31	41.61	379.91	188-389	0	50	429.91	-8	0	6.09	14.84	-30	0
	D0	27.67	457.57	0-660	0	0	0	0	0				
	Truck FC.	529.2	Trailer FC.	334.8	Fuel C.	816.71	Dist. C.	774.91	PD. C.	340			
5	D0	-	-	0-660	0	0	0	0		75	45		80
30	4	72.25	72.25	14-202	0	56	128.25	14	14	19.014	28.324	30	30
	5	19.32	147.57	8-289	0	59	206.57	-14	0	5.01	14.32	-30	0
	80	2.02	208.59	50-321	0	88	296.59	12	12	17.01	26.32	60	60
	81	25.05	321.64	133-322	0	43	364.64	-12	0	5.01	14.32	-60	0
	68	53	417.64	218-439	0	75	492.64	7	7	12.01	21.32	30	30
	69	26.31	518.95	316-589	0	84	602.95	-7	0	5.01	14.32	-30	0
	D0	25.98	628.93	0-660	0	0	0	0	0				
	Truck FC.	664.2	Trailer FC.	324	Fuel C.	767.36	Dist. C.	960.91	PD. C.	630			
6	D0	-	-	0-660	0	0	0	0		70	45		80
17	32	19.67	53	53-255	33.33	49	102	17	17	24.289	33.489	70	70
	33	34.04	136.04	139-390	2.96	65	204	-17	0	7.29	16.49	-70	0
	18	21.62	225.62	145-383	0	55	280.62	17	17	24.29	33.49	70	70
	19	26.48	307.1	243-440	0	40	347.1	-17	0	7.29	16.49	-70	0
	20	9.24	356.34	315-531	0	56	412.34	7	7	14.29	23.49	70	70
	21	48.99	461.33	270-530	0	55	516.33	-7	0	7.29	16.49	-70	0
	D0	86.65	602.98	0-660	0	0	0	0	0				
	Truck FC.	523.8	Traier FC.	367.2	Fuel C.	1137.42	Dist. C.	986.92	PD. C.	460			
7	D0	-	-	0-660	0	0	0	0		70	45		80
12	8	107.22	107.22	0-227	0	72	179.22	18	18	25.143	33.705	80	80
	9	36.01	215.23	0-256	0	64	279.23	-18	0	7.14	15.7	-80	0
	12	54.07	333.31	274-532	0	63	396.31	13	13	20.14	28.7	70	70
	13	25.25	421.56	279-470	0	89	510.56	-13	0	7.14	15.7	-70	0
	D0	85.51	596.07	0-660	0	0	0	0	0				
	Truck FC.	567	Trailer FC.	340.2	Fuel C.	1139.92	Dist. C.	1327.05	PD. C.	350			

Table E.2: Detailed schedule of the best known solution of problem 50F: truck 4,5,6,7

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	L'(T+G)	Tot.L	S_V/D_V	V.
8	D0	-	-	0-660	0	0	0	0		70	45		80
3	58	77.13	152	152-414	74.87	78	230	9	9	14.318	23.882	30	30
	42	4.44	234.44	60-307	0	68	302.44	8	17	22.32	31.88	40	70
	59	29.33	331.77	225-407	0	58	389.77	-9	8	13.32	22.88	-30	40
	43	34.93	424.7	343-588	0	69	493.7	-8	0	5.32	14.88	-40	0
	D0	78.98	572.68	0-660	0	0	0	0	0				
	Truck FC.	572.4	Trailer FC.	280.8	Fuel C.	807.91	Dist. C.	985.21	PD. C.	330			
9	D0	-	-	0-660	0	0	0	0		70	45		80
13	0	117.73	117.73	14-205	0	64	181.73	17	17	22.268	32.015	80	80
	1	27.95	209.68	64-344	0	62	271.68	-17	0	5.27	15.02	-80	0
	56	32.24	303.92	239-498	0	67	370.92	7	7	12.27	22.02	70	70
	57	44.84	415.76	388-577	0	45	460.76	-7	0	5.27	15.02	-70	0
	D0	78.06	538.82	0-660	0	0	0	0	0				
	Truck FC.	529.2	Trailer FC.	291.6	Fuel C.	934.1	Dist. C.	1205.96	PD. C.	480			
10	D0	-	-	0-660	0	0	0	0		75	45		80
14	50	104.68	104.68	0-137	0	54	158.68	11	11	16.065	23.754	60	60
	86	4.21	162.89	0-201	0	41	203.89	8	19	24.06	31.75	10	70
	51	42.09	245.98	174-406	0	86	331.98	-11	8	13.07	20.75	-60	10
	87	64.66	396.64	375-592	0	83	479.64	-8	0	5.07	12.75	-10	0
	D0	58.31	537.95	0-660	0	0	0	0	0				
	Truck FC.	480.6	Trailer FC.	280.8	Fuel C.	1045.23	Dist. C.	1066.4	PD. C.	370			
11	D0	-	-	0-660	0	0	0	0		75	45		80
33	36	92.25	92.25	0-157	0	31	123.25	8	8	15.07	23.821	70	70
	37	30.73	153.98	210-453	56.02	36	246	-8	0	7.07	15.82	-70	0
	16	22.65	268.65	105-346	0	41	309.65	16	16	23.07	31.82	30	30
	17	19.95	329.6	392-593	62.4	81	473	-16	0	7.07	15.82	-30	0
	D0	74.92	547.92	0-660	0	0	0	0	0				
	Truck FC.	642.6	Trailer FC.	367.2	Fuel C.	891.23	Dist. C.	980.08	PD. C.	240			

Table E.3: Detailed schedule of the best known solution of problem 50F: truck 8,9,10,11

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	$L^{'}$(T+G)	Tot.L	S_V/D_V	V.
12	D0	-	-	0-660	0	0	0	0		70	45		80
2	98	70.11	70.11	22-197	0	67	137.11	11	11	16.116	22.716	10	10
	70	31.82	168.93	26-267	0	37	205.93	14	25	30.12	36.72	70	80
	99	16.8	222.72	135-353	0	31	253.72	-11	14	19.12	25.72	-10	70
	71	36.17	289.89	271-466	0	81	370.89	-14	0	5.12	11.72	-70	0
	96	24.72	395.61	284-502	0	60	455.61	17	17	22.12	28.72	60	60
	97	23.22	478.83	281-483	0	37	515.83	-17	0	5.12	11.72	-60	0
	D0	87.23	603.06	0-660	0	0	0	0	0				
	Truck FC.	550.8	Trailer FC.	286.2	Fuel C.	1107.68	Dist. C.	1047.25	PD. C.	650			
13	D0	-	-	0-660	0	0	0	0		75	50.5		80
0	10	48.76	244	244-497	195.24	73	317	7	7	12.482	22.537	50	50
	22	8.13	325.13	263-486	0	30	355.13	7	14	19.48	29.54	20	70
	23	28.41	383.54	216-450	0	56	439.54	-7	7	12.48	22.54	-20	50
	11	34.52	474.07	228-483	0	67	541.07	-7	0	5.48	15.54	-50	0
	D0	23.92	564.99	0-660	0	0	0	0	0				
	Truck FC.	675	Trailer FC.	361.8	Fuel C.	612.09	Dist. C.	724.23	PD. C.	400			
14	D0	-	-	0-660	0	0	0	0		75	50.5		80
15	6	11.71	215	215-413	203.29	37	252	10	10	16.152	24.717	70	70
	7	27.93	279.93	228-488	0	47	326.93	-10	0	6.15	14.72	-70	0
	72	29.66	356.59	314-523	0	65	421.59	10	10	16.15	24.72	50	50
	73	23.23	444.82	323-499	0	55	499.82	-10	0	6.15	14.72	-50	0
	D0	60.06	559.88	0-660	0	0	0	0	0				
	Truck FC.	610.2	Trailer FC.	356.4	Fuel C.	559.61	Dist. C.	763.7	PD. C.	420			
15	D0	-	-	0-660	0	0	0	0		70	45		40
16	34	94.1	94.1	0-138	0	71	165.1	14	14	18.031	26.502	30	30
	35	33.2	198.3	225-450	26.7	81	306	-14	0	4.03	12.5	-30	0
	14	55.79	361.79	267-526	0	54	415.79	19	19	23.03	31.5	40	40
	15	10.91	426.7	248-455	0	59	485.7	-19	0	4.03	12.5	-40	0
	D0	134.26	619.96	0-660	0	0	0	0	0				
	Truck FC.	518.4	Trailer FC.	264.6	Fuel C.	792.79	Dist. C.	1305.08	PD. C.	510			

Table E.4: Detailed schedule of the best known solution of problem 50F: truck 12, 13,14,15

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	$L^{'}$(T+G)	Tot.L	S_V/D_V	V.
16	D0	-	-	0-660	0	0	0	0		75	45		40
32	24	103.48	103.48	69-343	0	86	189.48	16	16	20.12	27.728	30	30
	25	7.73	197.2	137-328	0	50	247.2	-16	0	4.12	11.73	-30	0
	66	52.94	300.14	190-379	0	78	378.14	9	9	13.12	20.73	20	20
	67	45.16	423.3	233-435	0	38	461.3	-9	0	4.12	11.73	-20	0
	D0	117.61	578.91	0-660	0	0	0	0	0				
	Truck FC.	486	Trailer FC.	286.2	Fuel C.	740.92	Dist. C.	1353.97	PD. C.	430			
17	D0	-	-	0-660	0	0	0	0		75	45		80
29	60	106.09	106.09	98-374	0	62	168.09	6	6	13.216	20.986	70	70
	61	42.51	210.6	129-325	0	42	252.6	-6	0	7.22	14.99	-70	0
	40	19.09	271.69	211-470	0	81	352.69	9	9	16.22	23.99	80	80
	41	45.14	397.83	221-444	0	49	446.83	-9	0	7.22	14.99	-80	0
	D0	50.48	497.31	0-660	0	0	0	0	0				
	Truck FC.	491.4	Trailer FC.	361.8	Fuel C.	954.34	Dist. C.	1057.76	PD. C.	520			
18	D1	-	-	0-660	0	0	0	0		70	45		80
36	74	165.04	165.04	0-231	0	41	206.04	16	16	23.945	31.415	70	70
	75	43.77	249.81	315-509	65.19	52	367	-16	0	7.94	15.42	-70	0
	D1	139.33	506.33	0-660	0	0	0	0	0				
	Sub.Req C.	1739.28											
19	D1	-	-	0-660	0	0	0	0		70	45		80
37	92	90.41	90.41	0-166	0	81	171.41	15	15	22.216	29.686	80	80
	93	47	218.41	311-584	92.59	76	387	-15	0	7.22	14.69	-80	0
	D1	136.82	523.82	0-660	0	0	0	0	0				
	Sub.Req C.	1787.31											
20	D2	-	-	0-660	0	0	0	0		70	45		80
38	84	110.34	110.34	30-250	0	78	188.34	17	17	24.945	32.415	60	60
	85	8.83	197.17	198-406	0.83	76	274	-17	0	7.94	15.42	-60	0
	D2	106.85	380.85	0-660	0	0	0	0	0				
	Sub.Req C.	579.58											
21	D2	-	-	0-660	0	0	0	0		70	45		80
39	64	33.93	46	46-307	12.07	40	86	20	20	27.508	34.978	40	40
	65	13.03	99.03	357-579	257.97	82	439	-20	0	7.51	14.98	-40	0
	D2	43.11	482.11	0-660	0	0	0	0	0				
	Sub.Req C.	1478.9											

Table E.5: Detailed schedule of the best known solution of problem 50F: truck 16,17,18,19,20,21

Id	Seq.	Tra.t.	Arr.t	TWs	Wait.t.	Ser.t.	Dep.t	S_L/D_L	L	L'(T+G)	Tot.L	S_V/D_V	V.
22	D3	-	-	0-660	0	0	0	0		70	45		80
40	62	129.91	129.91	78-338	0	53	182.91	20	20	27.799	35.269	60	60
	63	44.7	227.61	312-560	84.39	62	374	-20	0	7.8	15.27	-60	0
	D3	87.57	461.57	0-660	0	0	0	0	0				
	Sub.Req C.	1189.34											
23	D3	-	-	0-660	0	0	0	0		70	45		80
41	54	78.79	111	111-306	32.21	56	167	18	18	25.143	32.613	30	30
	48	45.21	212.21	208-475	0	86	298.21	12	30	37.14	44.61	30	60
	49	28.1	326.3	126-383	0	31	357.3	-12	18	25.14	32.61	-30	30
	55	39.29	396.59	324-596	0	69	465.59	-18	0	7.14	14.61	-30	0
	D3	55.54	521.14	0-660	0	0	0	0	0				
	Sub.Req C.	1767.24											
25	D5	-	-	0-660	0	0	0	0		70	45		80
43	82	14.44	267	267-471	252.56	64	331	13	13	20.07	27.54	60	60
	94	14.74	345.74	219-398	0	65	410.74	13	26	33.07	40.54	10	70
	95	29.04	439.77	248-485	0	37	476.77	-13	13	20.07	27.54	-10	60
	83	6.22	483	320-598	0	62	545	-13	0	7.07	14.54	-60	0
	D5	23.17	568.17	0-660	0	0	0	0	0				
	Sub.Tour C.	1810.64											

Table E.6: Detailed schedule of the best known solution of problem 50F: truck 22, 23, 25