Construction Heuristics for the Airline Taxi Problem

Ian Michael Dougal Campbell

A thesis submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Doctor of Philosophy.

Johannesburg, October 2013
Declaration

I declare that this thesis is my own, unaided work, except where otherwise acknowledged. It is being submitted for the degree of Doctor of Philosophy in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at any other university.

Signed this ___ day of __________ 20___.

______________________________
Ian Michael Dougal Campbell.
Acknowledgements

Firstly, I would like to thank my parents, my sister and my brothers for many years of love and support.

Secondly, special thanks to my supervisor, Prof. M.M. Ali, for his unwavering and motivational guidance and support, as well as his patience in wading repeatedly through this tome and his resulting comments and corrections.

Thirdly, many thanks to Margaret Campbell and Prof. Robert Reid for their comprehensive proof reading and valuable corrections.
Abstract

A literature review of vehicle routing problems (VRPs) in general, and specifically airline scheduling problems and the airline taxi problem, is provided. A real-world airline taxi scheduling problem is described as experienced by a tourist airline operating in the Okavango Delta, Botswana. In this problem, a daily schedule is drawn up manually by a team of experienced schedulers a few days before the day in question. In this research, a slightly relaxed version of the problem is considered in order to develop heuristics and modelling methods which will be useful for general cases. Various methods and heuristics are proposed for the problem and tested on a small version of the problem as well as the full-sized version. The most promising methods are demonstrated and solutions provided. One of the methods was applied to the actual problem to demonstrate the practical usefulness. In this case a schedule with a cost 12% lower than the manual schedule cost was achieved. All the heuristics and methods are applicable to certain other VRPs, particularly real-world or highly-constrained VRPs. An example is provided of a solution method for a real-world instance of the multi-vehicle capacitated vehicle routing problem (MVCVRP). Another example is provided of a standard, benchmark instance from the internet of a capacitated vehicle routing problem with time windows (CVRPTW).
Contents

Declaration i

Acknowledgements ii

Abstract iii

Contents iv

List of Figures ix

List of Tables x

1 Introduction 1

1.1 Scheduling in Commercial Scheduled Airlines . . . . . . . . . . . . . 1

1.2 On-Demand Airline Scheduling . . . . . . . . . . . . . . . . . . . . 2

1.3 Research Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

2 Objectives 5

3 Literature Review 6

3.1 Vehicle Routing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

3.1.1 Types of Vehicle Routing Problems . . . . . . . . . . . . . . . . 6
7.2.2 Aggregation Validation ........................................ 66
7.3 Geographic Heuristic .............................................. 66
  7.3.1 Description ..................................................... 66
  7.3.2 Evaluation of Geographic Heuristic Effect ................. 68
7.4 MCNFP Solutions .................................................... 69
  7.4.1 Small Schedule ................................................ 69
  7.4.2 Full Schedule ................................................ 72
7.5 Discussion .......................................................... 73

8 Agent Routing Variable Generation ................................ 76
  8.1 Description ....................................................... 76
  8.2 Observations and Results .................................... 83
  8.3 Discussion ....................................................... 87

9 Composite Variable Formulation .................................. 88
  9.1 Description ....................................................... 88
  9.2 Observations and Results .................................... 93
    9.2.1 Small Schedule ............................................ 93
    9.2.2 Full Schedule ............................................. 95
  9.3 Discussion ....................................................... 95

10 The Sefofane Air Scheduling Problem .......................... 97
  10.1 Constrained Problem ......................................... 97
  10.2 Results and Discussion .................................... 100
11 Other VRPs

11.1 Introduction .............................................. 102
11.2 Multi Vehicle CVRP (MVCVRP) .......................... 102
  11.2.1 Problem Description ................................. 102
  11.2.2 Exact ILP Formulation ............................... 103
  11.2.3 Composite Variable Formulation ...................... 106
  11.2.4 Methodology ......................................... 108
  11.2.5 Observations and Results ............................ 109
  11.2.6 Discussion ........................................... 110
11.3 Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) 110
  11.3.1 Problem Description ................................. 110
  11.3.2 Variable Generation ................................ 111
  11.3.3 Observations, Results and Discussion ............... 116

12 Conclusions ................................................. 117

13 Recommendations ........................................... 120

REFERENCES .................................................. 121

APPENDIX A Sefofane Air Problem Data ....................... 134

APPENDIX B Manual Schedule ................................ 140

APPENDIX C Aggregation Heuristic Result .................... 145

vii
List of Figures

3.1 An Insertion Heuristic .................................... 13
3.2 A Savings Heuristic ........................................ 13
3.3 2-Opt Exchange Move ...................................... 16
3.4 Multi Agent Cooperative Search ............................ 23
3.5 GPDP Classification ........................................ 25
3.6 Time-Space Network ........................................ 36
3.7 Time-Space Network with Time Discretisations ......... 38
5.1 Cessna C206 .................................................. 53
5.2 Cessna C208 Grand Caravan ............................... 54
5.3 Map of the Okavango Delta Area ......................... 56
7.1 Geographic Heuristic ........................................ 67
8.1 Agent Routing Method Modelling Process ................. 84
9.1 Linking of Two Flight Variables (Dual Linking) ....... 89
9.2 Composite Variable Modelling Method .................... 91
List of Tables

3.1 VRP Taxonomic Review ............................................... 9
3.2 Metaheuristics .......................................................... 19
3.3 Genetic Algorithm Approaches to the VRPTW ..................... 22
3.4 DARP Heuristics ......................................................... 27
3.5 DARP Metaheuristics .................................................... 32

5.1 Fleet Characteristics ..................................................... 54

7.1 Cost and Computation Times for Different Geographic Heuristic Parameters, Small Schedule ......................... 68
7.2 Effect of Geographic Heuristic on Number of Variables for Large Schedule ...................................................... 69
7.3 Small Schedule Solution ............................................... 71
7.4 Multi-Leg Trips - Unlimited Ground Waiting Time, Small Schedule ................................................................. 71
7.5 Multi-Leg Trips - Minimum Ground Waiting Time, Small Schedule ................................................................. 72
7.6 Schedule Cost Results Summary for Small Schedule .................... 72
7.7 Multi-Legs Trips for Large Schedule .................................... 73
7.8 Schedule Cost Results Summary for Large Schedule ..................... 73

8.1 Full Schedule Solutions .................................................. 85
8.2 Aircraft Changes and Ground Waiting Times ................. 86
8.3 Aircraft Changes and Ground Waiting Times with Passenger Group
   Time Cost Modified Objective Function ...................... 86
8.4 Aircraft Changes and Ground Waiting Times with Individual Passen-
   ger Time Cost Modified Objective Function .................. 86
9.1 Small Schedule Costs Using Composite Variables .............. 94
9.2 Small Schedule Using Composite Variables .................... 94
9.3 Full Schedule Costs Using Composite Variables with Constrained
   End-Of-Day Aircraft Locations ................................ 95
9.4 Comparison Solutions - Unconstrained End-of-Day Aircraft Positions 96
11.1 Customers for Small Version ................................. 108
11.2 Fleet Description ............................................ 108
11.3 MVCVRP Solutions ........................................... 110
A.1 Full Schedule ................................................. 135
A.2 Distance matrix .............................................. 138
A.3 Aircraft and Loadmaster Day Start and End Positions .......... 139
A.4 Airport Codes and Assigned Numbers .......................... 139
B.1 Manual Schedule .............................................. 141
C.1 Aggregated Groups for Full Schedule ........................ 146
E.1 Agent Routing Method Automated Full Schedule .............. 170
F.1 Composite Variable Method Automated Full Schedule ........ 173
1 Introduction

1.1 Scheduling in Commercial Scheduled Airlines

Operations research techniques applied to airline scheduling have been studied extensively. Applications in scheduled commercial applications date back to 1993 when Delta Airlines successfully implemented a fleet assignment problem [1] which saved them many millions of dollars. Since then almost all of the world’s scheduled commercial airlines have followed suit. In many ways the airline planning and scheduling problems are the most demanding in the transportation industry due to the size of the airlines and the possible savings involved.

Given a timetable, the airline scheduling problem can be divided into two subproblems, fleet assignment and aircraft routing [2]. Fleet assignment involves assigning an aircraft type to the flights in the timetable. Aircraft routing (or generating aircraft rotations) refers to the procedure of creating routes for a fleet of aircraft such that all of the flights are provided as required.

Generating the timetable is referred to as schedule design [3]. In this problem, the flight legs are created which are intended to satisfy demand and generate revenue for the airline. Typically commercial airlines generate the schedule, then do the fleet assignment and maintenance routing, followed by the crew scheduling problem. Maintenance routing involves scheduling and routing of maintenance opportunities for every aircraft in the fleet [4]. Theoretically, all four of these problems could be solved in one and this is referred to as the integrated problem. However, the size of many airlines precludes such a large problem from being solved practically, and the problems are addressed sequentially, separately and in an iterative manner.

Fleet assignment problems are often modelled using overlapping time-distance network flow models (or multi-commodity network flow, MCNF) with many variables restricted to binary or integer types. This results in mixed integer programming
(MIP) models with large numbers of variables and constraints. Because of the computational effort involved, solutions can take impractically long times to achieve. Much effort has been spent on techniques to solve these problems within reasonable time limits.

Passenger airlines might only design and optimize their schedule once from scratch. This is called ‘cold start’ [1]. Such models can be huge and take hours if not days to run. In most cases, a previously designed feasible schedule is optimized (‘warm start’) which results in much faster solution times. A schedule sometimes repeats itself daily and/or weekly and does not change for months. Changes that are implemented are normally small.

1.2 On-Demand Airline Scheduling

Certain charter airlines schedule their aircraft to satisfy some demand such as that generated by a once off event, e.g. a sporting event. In this case, the objective is to maximize profit and the demand does not necessarily have to be completely satisfied. Either the group requiring the transport will hire the plane to accommodate all group members, or the charter company will schedule a special trip to meet the demand, for example, from multiple groups.

The airline taxi scheduling problem is different in that demand must be met completely at the lowest cost possible. An airline taxi service is operated by a charter airline company called Sefofane Air in the Okavango Delta region in Botswana. The extensive waterways that make up this geographical feature make land transport difficult and often impossible. In addition the authorities have legislated that passengers must be ferried between the various tourist camps by aircraft to minimize environmental effects. Therefore small groups of tourists must be ferried by aircraft between places of interest or accommodation according to a schedule that has previously been designed by a tour operator and paid for by the client (passengers).

The schedule for such an airline changes completely every day. Often the next day’s schedule needs to be designed from scratch. Currently, these schedules are generated manually by a team of experienced people. Growth of airline traffic and risks with regard to the possibility of losing scheduling staff has created the need to automatically create good schedules and to reduce aircraft operation and scheduling costs. Sefofane Air must satisfy 158 bookings on a busy day, using 14 aircraft of two different types and serving 21 destinations.
There are some other similar or closely-related problems in the literature, namely
the static dial-a-flight problem (SDAFP) [5], air taxi [6] or per-seat, on-demand
(PSOD) air transportation problem [7]. The problems described appear to be largely
similar, therefore the Sefofane Air scheduling problem can be classed in vehicle
routing literature accordingly.

1.3 Research Motivation

This research is intended to complement current methodologies for taxi airline
scheduling. In doing so, it is intended to add to the body of knowledge of solu-
tion methods for vehicle routing problems (VRPs) in general.

This work specifically addresses construction heuristics to create variables for use in
airline taxi minimum cost network flow (MCNF) formulations. As such, most of the
formulations make use of time-space networks, as described in Chapter 3, section
3.2.1.

The problems approached in this work vary in size up to 158 requests (or customers).
For all instances, smaller versions of the actual instances are created and used for
demonstration and, where applicable, parameter tuning.

A customer aggregation heuristic and a geographic heuristic are introduced for the
airline taxi problem, both of which will find applicability in certain instances of other
VRPs such as the dial-a-ride problem (DARP) [8]. These allow a standard MCNF
formulation to be used for small versions of the airline taxi problem.

A framework for using greedy agents to generate variables for an MCNF formulation
is demonstrated. This allows larger-sized versions of the airline taxi problem to be
solved, such as the size of schedules that are generated by Sefofane Air.

A method of creating composite variables which vastly reduces the number of vari-
ables in an MCNF formulation for the airline taxi problem is presented. The ideas
behind this method were first applied to the airline taxi problem and presented by
Silverwood [9], then refined by Lafoyi [10]. In the former, a constraint programming
formulation was presented using a similar composite variable construction method
as described in this work. In the latter, the composite variable method was further
developed to be used in an integer linear program (ILP) formulation. In this work,
the method is refined and a practical solution to the full problem is provided. The
quality of the solution obtained is evaluated.
Finally, it is shown that these methods offer alternative approaches to other VRPs. This is done by tackling a real-world multi-vehicle capacitated vehicle routing problem (MVCVRP) [11] and a standard, benchmark VRP with time windows obtained from the Internet.

The airline taxi problem addressed is an actual, real-world problem, and as such is highly constrained with operational requirements. Similarly, the MVCVRP in Chapter 11 is a real-world problem described by the company concerned. The VRPTW in Chapter 11 is a problem artificially created specifically to be a benchmark problem, and as such it does not have the typical operational constraints of a real-world problem (i.e. it is less highly constrained).

Results are presented showing the usefulness of the various methods proposed.
2 Objectives

This work was inspired by the Sefofane Air scheduling problem, which will be described in more detail in Chapter 5. One objective is to provide practical methods to solve that problem and demonstrate these methods. In order to achieve this, methods to solve the airline taxi problem will be developed. Heuristics offer the most viable methods to achieve this. This work will only deal with construction of solutions, as opposed to solution improvement techniques and heuristics.

Therefore, the objectives are stated as follows:

- To develop and evaluate construction heuristic methods to solve the airline taxi problem in general and the specific problem under consideration.
- To suggest and consider the use of such methods in other vehicle routing problems.
3 Literature Review

3.1 Vehicle Routing

3.1.1 Types of Vehicle Routing Problems

There are a number of standard vehicle routing problems (VRPs) defined in the literature, all classified as \(NP\)-hard and therefore difficult to solve, even for small instances. In practice, such problems become increasingly more difficult to solve as they grow in size, so keeping the number of variables to a minimum is important.

The travelling salesperson problem (TSP) is a problem where the shortest path through a number of locations must be obtained, and each location is only visited once. It is important because, although \(NP\)-hard, a number of efficient algorithms have been developed for it, notably the famous Lin-Kernighan (L-K) heuristic algorithm [12]. VRP solution methods often involve solving some sub-problem/s which are TSPs.

TSPs, like VRPs, can be formulated as integer linear programs (ILPs), and solved as such. If no heuristics are used, the solution is termed exact, since, if well-formulated, the solution obtained is optimal. However the methods for solving ILPs are generally based on branch and bound, which is a form of enumeration and therefore varies in solution time for different instances. Solving can be slow, especially if there are many variables and constraints.

The term VRP often refers to a delivery problem with depot and customers. Because of the difficulties in application of exact methods to practically-sized problems, heuristics and metaheuristics are frequently applied. Variations of the VRP include the capacitated VRP (CVRP) and the multi-vehicle CVRP (MVCVRP).

The CVRP involves using vehicles of one type, but with limited capacity, to deliver goods from a depot to a number of customers. The CVRP with time windows
(CVRPTW) is similar, but with the addition of time windows which define times within which the deliveries must be made to the customers. The MVCVRP is a variation of the CVRP but with multiple vehicle types, each of different capacity.

VRPs may have probabilistic variables. An example is the courier delivery problem, a variant of the VRPTW in which customer arrivals and service times are probabilistic [13].

A class of VRPs deals with situations where goods are transported between pickup and delivery locations. These are called VRPs with pickup and deliveries (VRPPDs) [14]. There are three versions, the pickup and delivery problem (PDP), the pickup and delivery vehicle routing problem (PDVRP) and the dial-a-ride problem (DARP).

The PDP comes in variants such as the PDP with time windows (PDPTW) and the multi-vehicle PDPTW (MV-PDPTW) [15]. The PDP and variants deal with the case of paired pickup and delivery points, as opposed to the PDVRP which has unpaired pickup and delivery points. Paired pickup and delivery points refer to situations where each customer or delivery has a unique pickup and a unique drop-off point.

The dial-a-ride problem (DARP) is a paired vehicle taxi scheduling problem where passengers must be collected from pickup points and delivered to drop-off points. These drop-off points can be anywhere on a route network.

There are many other variations to the VRP and the interested reader is referred to various literature surveys of such problems such as those of Aronson [16], Laporte [17] and Kumar and Panneerselvam [11].

There does not appear to be consensus yet as to what defines a VRP. Eksioglu et al. [18] place the travelling salesperson problem (TSP), the VRP and DARP as separate categories of the generalised routing problem, along with the shortest path problem, the Chinese postman problem, the rural postman problem and the arc routing problem.

A VRP taxonomic review has been compiled recently [18] and is shown in Table 3.1. Broadly, there are five classifications, these being type of study, scenario characteristics, problem physical characteristics, information characteristics and data characteristics. The variety of attributes that can be encompassed in a VRP, and the possibility for subtle differences between problems, is evident.

When comparing the research to be presented in this thesis with other VRP work
(Table 3.1), it is notable that the work in this thesis addresses relatively under-researched areas including real-world problems, directed network, multiple origins, and multiple depots.

The airline taxi problem or dial-a-flight problem (DAFP) can best be classified under the PDP class of problems. It differs from the DARP in that passengers can only start and end their journeys at a limited number of pre-specified locations (airports), and not at any point on the network. This difference implies that in the DARP, passengers or passenger groups are unlikely to have origin or destination locations matching those of any other passenger groups, whereas in the DAFP, different passengers are likely to have matching locations for origin and destination.

In practical or real world problems, there are a number of often subtle differences between different situations and these make generalised modelling methods difficult to devise, even between problems which are apparently of the same type.

Exact methods mostly refer to large-scale integer linear programming problems (ILPs) and can be difficult to solve. Hasle and Kloster [19] suggest instances of 50-100 bookings cannot be solved consistently using exact methods. This is the reason both heuristics and metaheuristics are of such widespread interest.
<table>
<thead>
<tr>
<th>1. Type of study</th>
<th>2.4 Request times of new customers</th>
<th>2.8 Backhauls</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Theory</td>
<td>2.4.1 Deterministic</td>
<td>2.8.1 Nodes request simultaneous</td>
</tr>
<tr>
<td>1.2 Applied methods</td>
<td>2.4.2 Stochastic</td>
<td>pickups and deliveries</td>
</tr>
<tr>
<td>1.2.1 Exact methods</td>
<td>2.4.3 Unknown</td>
<td>2.8.2 Nodes request either linehaul</td>
</tr>
<tr>
<td>1.2.2 Heuristics</td>
<td>2.5 On-site service waiting times</td>
<td>or backhaul service, but not both</td>
</tr>
<tr>
<td>1.2.3 Simulation</td>
<td>2.5.1 Deterministic</td>
<td>2.9 Node/arc covering constraints</td>
</tr>
<tr>
<td>1.2.4 Real time solution methods</td>
<td>2.5.2 Time dependent</td>
<td>2.9.1 Precedence and coupling constraints</td>
</tr>
<tr>
<td>1.3 Implementation documented</td>
<td>2.5.3 Vehicle type dependent</td>
<td>2.9.2 Subset covering constraints</td>
</tr>
<tr>
<td>1.4 Survey, review or meta-research</td>
<td>2.5.4 Stochastic</td>
<td>2.9.3 Recourse allowed</td>
</tr>
<tr>
<td>2. Scenario characteristics</td>
<td>2.5.5 Unknown</td>
<td></td>
</tr>
<tr>
<td>2.1 Number of stops on route</td>
<td>2.6 Time window structure</td>
<td></td>
</tr>
<tr>
<td>2.1.1 Known (deterministic)</td>
<td>2.6.1 Soft time windows</td>
<td>3. Problem physical characteristics</td>
</tr>
<tr>
<td>2.1.2 Partially known, partially probabilistic</td>
<td>2.6.2 Strict time windows</td>
<td>3.1 Transportation network design</td>
</tr>
<tr>
<td>2.2 Load-splitting constraint</td>
<td>2.6.3 Mix of both</td>
<td>3.1.1 Directed network</td>
</tr>
<tr>
<td>2.2.1 Splitting allowed</td>
<td>2.7 Time horizon</td>
<td>3.1.2 Undirected network</td>
</tr>
<tr>
<td>2.2.2 Splitting not allowed</td>
<td>2.7.1 Single period</td>
<td>3.2 Location of addresses (customers)</td>
</tr>
<tr>
<td>2.3 Customer service demand quality</td>
<td>2.7.2 Multi-period</td>
<td>3.2.1 Customers on nodes</td>
</tr>
<tr>
<td>2.3.1 Deterministic</td>
<td></td>
<td>3.2.2 Arc routing instances</td>
</tr>
<tr>
<td>2.3.2 Stochastic</td>
<td></td>
<td>3.3 Geographical location of customers</td>
</tr>
<tr>
<td>2.3.3 Unknown</td>
<td></td>
<td>3.3.1 Urban (scattered with pattern)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.3.2 Rural (randomly scattered)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.3.3 Mixed</td>
</tr>
<tr>
<td>3. Problem physical characteristics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1 Transportation network design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1.1 Directed network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1.2 Undirected network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2 Location of addresses (customers)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2.1 Customers on nodes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2.2 Arc routing instances</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3 Geographical location of customers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3.1 Urban (scattered with pattern)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3.2 Rural (randomly scattered)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3.3 Mixed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4 Number of points of origin</td>
<td>3.9 Vehicle homogeneity (capacity)</td>
<td>4.1.1 Static</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>3.4.1 Single origin</td>
<td>3.9.1 Similar vehicles</td>
<td>4.1.2 Partially dynamic</td>
</tr>
<tr>
<td>3.4.2 Multiple origins</td>
<td>3.9.2 Load-specific vehicles</td>
<td>4.2 Quality of information</td>
</tr>
<tr>
<td>3.5 Number of points of loading/unloading facilities (depot)</td>
<td>3.9.3 Heterogeneous vehicles</td>
<td>4.2.1 Known (deterministic)</td>
</tr>
<tr>
<td>3.5.1 Single depot</td>
<td>3.9.4 Customer-specific vehicles</td>
<td>4.2.2 Stochastic</td>
</tr>
<tr>
<td>3.5.2 Multiple depots</td>
<td></td>
<td>4.2.3 Forecast</td>
</tr>
<tr>
<td>3.6 Time window type</td>
<td></td>
<td>4.2.4 Unknown (real-time)</td>
</tr>
<tr>
<td>3.6.1 Restriction on customers</td>
<td></td>
<td>4.3 Availability of information</td>
</tr>
<tr>
<td>3.6.2 Restriction on road</td>
<td>3.10 Travel time</td>
<td>4.3.1 Local</td>
</tr>
<tr>
<td>3.6.3 Restriction on depot/hubs</td>
<td>3.10.1 Deterministic</td>
<td>4.3.2 Global</td>
</tr>
<tr>
<td>3.6.4 Restriction on drivers/vehicle</td>
<td>3.10.2 Function dependent (a function of current time)</td>
<td>4.4 Processing of information</td>
</tr>
<tr>
<td>3.7 Number of vehicles</td>
<td>3.10.3 Stochastic</td>
<td>4.4.1 Centralised</td>
</tr>
<tr>
<td>3.7.1 Exactly $n$ vehicles</td>
<td>3.10.4 Unknown</td>
<td>4.4.2 Decentralised</td>
</tr>
<tr>
<td>(TSP in this segment)</td>
<td>3.11 Transportation cost</td>
<td>5 Data characteristics</td>
</tr>
<tr>
<td>3.7.2 Up to $n$ vehicles</td>
<td>3.11.1 Travel time dependent</td>
<td>5.1 Data used</td>
</tr>
<tr>
<td>3.7.3 Unlimited number of vehicles</td>
<td>3.11.2 Distance dependent</td>
<td>5.1.1 Real world data</td>
</tr>
<tr>
<td>3.8 Capacity constraints</td>
<td>3.11.3 Vehicle dependent</td>
<td>5.1.2 Synthetic data</td>
</tr>
<tr>
<td>3.8.1 Capacitated vehicles</td>
<td>3.11.4 Operation dependent</td>
<td>5.1.3 Both real and synthetic data</td>
</tr>
<tr>
<td>3.8.2 Uncapacitated vehicles</td>
<td>3.11.5 Function of lateness</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.11.6 Implied hazard/risk related</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 Information characteristics</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.1 Evolution information</td>
<td></td>
</tr>
</tbody>
</table>
3.1.2 Solving VRPs

Heuristics

Classifications Heuristics are common-sense methods to reduce the size of problems by, for example, removing variables from a problem. As such they are commonly used on their own or in conjunction with other methods for NP-hard problems. They can be classified as construction, improving, mathematical or practical types.

Construction heuristics are applied when constructing the problem (i.e. assembling the variables and constraints) to achieve a first, feasible solution. In contrast, improving heuristics are applied to an existing solution to improve it.

Mathematical heuristics relate to mathematical manipulations of the formulation. They are those heuristics suggested by a numerical examination of the problem, for example, in a maximization problem paying more attention to the variables with relatively larger objective function coefficients, or lower costs, as is done in column generation. Preprocessing done by ILP solvers such as implementing cutting planes (cuts额外 constraints to reduce the feasible region) is another example.

Practical heuristics relate to practical evaluations of the problem setting or environment, and removal of obviously impossible or unlikely solutions. They involve reducing the problem size by specifically eliminating variables which are likely to be zero in practice. For example, in a scheduling problem, eliminate routes (variables) which are comparatively long and expensive. Another example is a valid inequality, which is a cut/additional constraint designed for the problem at hand to decrease the size of the search region.

Heuristic methods can be applied when using mathematical programming techniques. For example, an iterative method could proceed as follows:

- Formulate an easy to solve relaxed problem by removing some constraints.
- Adjust the problem such that it satisfies the dropped restrictions, or add some restrictions, and solve the problem again.
- Repeat until an acceptable, feasible solution is obtained.

Heuristics are useful in conjunction with other techniques since they can reduce the size of a problem substantially. However, since they eliminate possible solutions they
need to be carefully considered. Often useful heuristics can be developed which are specific to a certain problem or problem type.

The best solution procedures are specific to variable types. For example, problems with only continuous variables can be efficiently solved using the simplex method, and special techniques exist for efficiently solving purely binary variable problems [20,21]. Also, numerical heuristic techniques such as Feaspump [22] often work well for general integer variables and less well for binary variables, so much so that the latest version works in two phases, first dealing with the binary variables and then the integer variables. Benders decomposition [23] exploits structure by decomposing the problem by variable type, i.e. the subproblem is purely binary or purely general integer.

**Construction Heuristics Algorithms** Aronson [16] has listed some of the main construction heuristics for the VRP. An obvious one is the greedy or nearest neighbour algorithm [7] used for the TSP, where the closest neighbour is always the one travelled to next. Another, related method involves finding and converting minimum spanning trees (MSTs) into feasible routes [24].

A nearest neighbour algorithm for the VRPTW is described by Solomon [25]. Solomon uses a distance measure consisting of both geographic and temporal measures of distance in his algorithm. This distance is given by 

$$c_{ij} = w_1 d_{ij} + w_2 T_{ij} + w_3 v_{ij},$$

where $d_{ij}$ is the geographical distance between two customers, $T_{ij}$ is the time difference between the completion of the service at $i$ and the start of service at $j$, and $v_{ij}$ is the urgency of servicing customer $j$. This urgency is calculated as the time remaining until the deadline of servicing customer $j$ is reached. $w_1$, $w_2$ and $w_3$ are weights which when summed together add up to 1.

Hosny [26] describes the construction procedure as follows: A route is created with one customer. Thereafter, customers are inserted sequentially into the route until no further insertions can feasibly occur. With reference to Figure 3.1, unrouted customer $k$ is inserted between customers $i$ and $j$. Insertions are done based on the time-modified distance measure.

Subtour patching [16] or insertion procedures [7] for a TSP involve creating a number of subtours, i.e. solving a relaxed version of the TSP without the subtour elimination constraints. The subtours are then merged (patched) into one cycle. The most common insertion heuristic is termed the nearest neighbour insertion algorithm [7]. This proceeds in two steps. It first takes a subtour of nodes and finds a node which
should join the subtour next (selection step), then determines where in the subtour it should be inserted (insertion step).

The savings algorithm is a classic algorithm described by Clarke and Wright [27] for the VRPTW and involves computing the savings (typically in distance) by adding nodes to a subtour, then selecting the largest saving node and adding it to the subtour. Initially, each customer is assigned to one vehicle. Thereafter, the savings achieved by combining two such routes, i.e. taking one such customer $i$ and adding that customer to the route of another customer $j$ will result in a savings in cost of $S_{ij} = c_{io} + c_{oj} - c_{ij}$, as depicted in Figure 3.2.

Two types of construction heuristics are defined by Hosny [26]. These are sequential construction where routes are constructed after each other, and parallel construction.
where routes are constructed at the same time. The parallel version of the savings algorithm involves finding the highest saving among all the customers and executing that change. The sequential version considers one route at a time and implements the best saving by joining another route to it.

The nearest merger algorithm involves setting up a number of subtours which are then merged in a way to reduce costs.

Two-phase algorithms [7,16] are often applied to VRPs. They involve a clustering phase followed by a routing phase. In these methods, each city is assigned to a vehicle, and the TSPs are solved for each vehicle-cluster combination. For example, in the sweep algorithm [28], nodes are first assigned to vehicles, then the order of visitation is assigned. Customers are represented by their polar coordinates [29]. An angle of 0 is assigned to an arbitrary customer and all other customers’ angles calculated relative to that and ranked. The procedure is then:

- Choose an unused vehicle.
- Start from the next customer having the smallest angle, and assign customers to the vehicle until its capacity is reached.
- Optimize each vehicle route using a TSP method.
- Perform vertex exchanges between routes if cost is reduced and re-optimise.

The Fisher and Jaikumar algorithm [30] does the clustering, then uses a TSP to do the routing for each vehicle. A general assignment problem (GAP) is solved to assign customers to vehicles. Thereafter a travelling salesperson problem (TSP) with time windows is solved to optimise the vehicles’ routes. The method is applied to VRPs from 50 to 199 customers.

Hierarchical cycling involves clustering first into clusters no bigger than some parameter, then replacing each cluster with a representative node. The new nodes are clustered in the same way, until only one cluster remains. The node is replaced with its cluster and the shortest path through the cluster is found. The process is repeated.

Route first/cluster second algorithms [16] involve constructing a TSP tour for all nodes except the depot, then breaking the tour into pieces such that all pieces can be assigned to a vehicle.
Many of the heuristics for the multi-vehicle DARP (MVDARP) are two-phase algorithms in which phase 1 selects and clusters users and phase 2 routes the vehicles. Dumas et al. [31] proposed creating mini-clusters of customers, where each mini-cluster is transportable by one vehicle, while respecting time constraints, vehicle capacity, pairing and precedence. The mini-clusters are combined to form feasible routes using column generation. Ioachim et al. [32] showed that an optimisation technique in the clustering phase is advantageous.

**Improving Heuristics Algorithms** These involve taking a solution and searching for improving modifications, often iteratively. A feasible solution must first be found using a construction method. Then, for example, a search will be conducted among neighbouring solutions for an improved one. The definition of the neighbourhood defines how the search is conducted. For example, a savings/insertion algorithm quickly finds an initial solution, sometimes by creating many routes with only one customer, then improves it towards a cheaper solution. This is done by merging routes together as long as this process saves costs.

If a reduced cost solution is found it may be adopted if the first acceptance criteria is used [26]. Otherwise, if using the best acceptance criteria, all the neighbourhood solutions are evaluated first and the best is chosen to be implemented.

A local search involves finding a new solution in the neighbourhood of the current solution [26]. Care must be taken to avoid local optima which is why techniques such as tabu search are used. The neighbourhood size can be varied. If it is increased it becomes a large neighbourhood search (LNS) method. Variable neighbourhood search (VNS) has been suggested by Hansen and Mladenovic [33]. This involves gradually increasing the neighbourhood size until a stopping condition is met. VNS has been successfully applied to a number of VRPs, for example, the VRPTW [34], the PD travelling salesperson problem (PDTSP) [35], the periodic VRPTW [36] and the multi-depot VRPTW [37]). It was hybridised with simulated annealing (SA) to solve the PDP by Hosny [26].

Improvement/exchange algorithms involve finding an initial solution, then improving/exchanging edges, nodes and/or vehicles to find improvements (edges are sometimes also referred to as links, arcs or legs in a routing network). The most common are the $k$-optimal methods involving the deletion of $k$ edges and replacement by $k$ other edges. 2-opt and 3-opt versions are most common, since using more edges typically expands the neighbourhood too much and involves too many options, making the algorithm slow. An example of a 2-opt exchange move is shown in Figure 3.3.
The famous Lin-Kernighan (LK) algorithm [12] for the TSP decides dynamically how many edges to exchange at each iteration. In an iteration it may swap a subtour of 2 paths (2-opt) or 3 paths (3-opt) for another to make the tour shorter.

Or-opt is a special case of 3-opt where up to 3 edges are removed and replaced at a different location in the same route. Because there are much fewer options, it is computationally less demanding than 3-opt and provides comparable quality solutions.

A 2-opt exchange swaps paths which belong to the same route. Potvin and Rousseau [38] introduced 2-opt* which tries to combine two different routes by joining the last customers of the second route after the first customers on the first route.

Inter-route operators were introduced by Savelsbergh [39]. These include:

- Re-locate: Moves a customer from one route to another.
- Exchange: Swaps two customers in different routes.
- Cross: Similar to 2-opt*.

Other neighbourhood operators include:

- $\lambda$-exchange [40]: Replaces a set of customers on a route with another set, possibly of a different size.
• CROSS-exchange [41]: Swaps two groups of customers from one route to another.
• GENI-exchange [42]: An extension of the re-locate operator to allow for customer insertion between non-consecutive customers on another route.
• Ejection chains [43]: Exchange of customer sets, but operates on more than two routes.
• Cyclic $k$-transfers [44]: Transfer of customers from one route to another.
• Modified ejection chains [45]: Includes re-ordering of routes.

Most heuristic methods are specific to a certain type of problem. Pisinger and Ropke [46] describe a heuristic which can solve five variants of the VRP; the vehicle routing problem with time windows (VRPTW), the capacitated vehicle routing problem (CVRP), the multi-depot VRP (MDVRP), the site-dependant VRP (SDVRP) and the open VRP (OVRP). An adaptive large neighbourhood search (ALNS) framework is described which involves a number of simple, fast algorithms competing to modify the current solution. Each iteration involves choosing an algorithm to destroy the current solution and then choosing an algorithm to repair the solution.

Large neighbourhood search (LNS) algorithms [15] might remove and replace a large number of customers (30% - 40%) in an iteration. They are computationally more expensive than faster, simpler algorithms but can provide good results [47] [46]. It has been suggested that the success of neighbourhood search algorithms obviates the need for sophisticated construction algorithms which are generally time consuming, parameter dependent and hard to implement [15].

**Metaheuristics**

Metaheuristics have become popular for VRPs because they allow smaller, non-linear and discrete variable model formulations, at the expense of slow processing and convergence. Genetic algorithms (GA), tabu search (TS), particle swarm optimization (PSO), ant colony optimization (ACO) and simulated annealing (SA) have been applied to vehicle transport scheduling problems.

A summary of metaheuristic techniques for the VRPTW is shown in Table 3.2[26]. Practitioners often apply a cheapest insertion method for initial feasible solution construction such as that proposed by Solomon [25]. Solomon’s I1 insertion heuristic starts by initialising a route, then inserts a new customer between two others in the route. The cost of insertion is minimised while retaining feasibility. Solution improvement generally uses a k-opt method. Heuristics are designed for the specific
problem to reduce the search area. Specifically, Garcia et al. [48] only allow inclusion of the relatively shorter edges.
<table>
<thead>
<tr>
<th>Reference</th>
<th>Algorithm</th>
<th>Solution Construction</th>
<th>Solution Improvement</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Garcia et al. [48]</td>
<td>TS</td>
<td>Solomons insertion</td>
<td>2-Opt, Or-Opt</td>
<td>Parallel implementation. Only allows moves involving arcs close in distance.</td>
</tr>
<tr>
<td>Rochat and Taillard [49]</td>
<td>TS</td>
<td>Solomons insertion and 2-Opt</td>
<td>2-Opt, relocate</td>
<td>Uses adaptive memory containing routes obtained from best solutions visited during the search, with the purpose of providing new starting solutions.</td>
</tr>
<tr>
<td>Taillard et al. [41]</td>
<td>TS</td>
<td>Solomons insertion</td>
<td>CROSS</td>
<td>Decomposes solutions into subsets of routes, based on a polar angle associated with the centre of gravity of each route. TS applied to each subset separately. Diversification by penalizing frequently performed exchanges. Intensification by reordering customers within best routes using I1 insertion.</td>
</tr>
<tr>
<td>Lau [50]</td>
<td>TS</td>
<td>Relocate from holding list</td>
<td>Exchange relocate</td>
<td>Allows violation of constraints for a penalty in the objective function. Penalty parameters adjusted dynamically.</td>
</tr>
<tr>
<td>Chiang and Russell [51]</td>
<td>SA</td>
<td>Russell’s parallel construction</td>
<td>Interchange and k-node interchange</td>
<td>Enhancing of the annealing process via a varying size tabu-list.</td>
</tr>
<tr>
<td>Li and Lim [52]</td>
<td>Tabu-embedded SA</td>
<td>Solomon’s insertion and sweep heuristics</td>
<td>Shifting and exchange of customer segments between and within routes</td>
<td>SA restarts from current best solution several times. Reduce routes by reordering customers and inserting them into other routes. Diversification by random shifts and exchange of customer segments.</td>
</tr>
<tr>
<td>Bent and Van Hentenryck [53]</td>
<td>SA and LNS</td>
<td>Not specified</td>
<td>2-Opt, Or-Opt, relocate, exchange and 2-Opt*</td>
<td>2-phase approach: SA to minimize number of routes and LNS to minimize total distance.</td>
</tr>
</tbody>
</table>
Table 3.2 contd.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Algorithm</th>
<th>Solution Construction</th>
<th>Solution Improvement</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gambardella et al. [54]</td>
<td>ACO</td>
<td>Nearest-neighbour heuristic with probabilistic rules</td>
<td>CROSS</td>
<td>2 colonies: first one minimizes number of vehicles, and the second minimizes total distance. Both cooperate in updating best solution.</td>
</tr>
<tr>
<td>Braysy et al. [55]</td>
<td>TA and GLS*</td>
<td>Savings heuristic</td>
<td>SPLIT (of routes). Limited CROSS. Limited Or-Opt</td>
<td>Construction algorithm only considers selected routes and selected customers for merging. Initial phase to minimize number of routes, based on simple customer re-insertion. Further improvement using TA and GLS - TA allows local search moves that worsen the objective function, provided it is within a certain threshold limit - GLS penalizes certain solutions based on some solution features (e.g. long edges), not considered likely to be part of a near-optimal solution.</td>
</tr>
</tbody>
</table>

* TA - threshold accepting, GLS - guided local search
Simulated annealing (SA) is a search procedure successfully applied to various VRPs including those devised by Osman [40]. Nanry and Barnes [56], Cordeau and Laporte [8] and Gendreau et al. [57] apply tabu search for the multiple VRP with pickup and delivery (VRPPD), DARP and urban courier service problems (UCSP) respectively. Cordeau and Laporte [58] state that tabu search is the most successful metaheuristic for the VRP, having outperformed alternative methods in a number of benchmark studies.

Ant colony optimization (ACO) has been applied to certain VRPs such as the VRPPD [59,60], the CVRP [61,62], VRPTW [60], the TSP [63] and the standard VRP [64]. It has not been applied to the MVCVRP or MVCVRPTW, since there appear to be issues dealing with the multi-vehicle constraint.

Like most metaheuristics, ACO can be adjusted in various ways for a specific problem. Gambardella et al. [54] used it to solve the VRPTW and use global pheromone updating, as opposed to the more usual local updating.

Hybrid metaheuristics refer to combining two or more methods and have become popular for VRPs. A taxonomy of such methods has been presented by Talbi [65]. Hybridisations can be classified according to whether the different methods occur sequentially (relay) or whether agents are parallel (or teamwork/cooperating). Local optimisations can be carried out by different metaheuristics (low-level), or the metaheuristic for the global optimisation (high-level) could be varied. Methods can include deterministic methods such as ILPs and often heuristic methods are also included. Hosny [26] has compiled a table of genetic algorithm (GA) approaches to the VRPTW (Table 3.3) and found almost all GA techniques are combined with some other heuristic, local search or other metaheuristic for solution construction and/or improvement.
<table>
<thead>
<tr>
<th>Reference</th>
<th>Initial Population</th>
<th>Crossover</th>
<th>Mutation</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thangiah[66]</td>
<td>A set of K seed angles are planted, where K is the number of vehicles. Sector rays are drawn originating from the depot to each seed angle.</td>
<td>Exchange a randomly selected portion of the bit string between chromosomes Routing done using cheapest insertion of Golden and Stewart [67].</td>
<td>Randomly change bit value.</td>
<td>Cluster first route second approach called GIDEON. Each chromosome is a set of possible clustering schemes and the GA is used to improve the clustering. Routes improved by λ-interchange.</td>
</tr>
<tr>
<td>Potvin and Bengio [68]</td>
<td>Chromosome is a problem solution created using Solomon’s cheapest insertion.</td>
<td>Sequence-Based Crossover (SBX): two sub-tours from two parents are linked together. Route-Based Crossover (RBX) replaces a route of one parent by another route from the second parent. Repair operator applied for infeasible offspring.</td>
<td>Reduce number of routes by inserting customers from a randomly selected short route into another. Local search mutation using Or-Opt exchanges.</td>
<td>The approach is named GENEROUS.</td>
</tr>
<tr>
<td>Blanton and Wainwright [69]</td>
<td>Solution coded as a permutation of customers. Insertion heuristic used to construct solution.</td>
<td>Specialized crossover that uses global precedence relationship among customers (TW/distance).</td>
<td>Randomly exchange 2-edges.</td>
<td>The GA searches for a good ordering of customers, while construction of feasible solutions is handled by a greedy heuristic.</td>
</tr>
<tr>
<td>Berger et al. [70]</td>
<td>Random insertion heuristic. Chromosomes are problem solutions. Two populations: POP1 used to minimize distance and POP2 to minimise constraint violations.</td>
<td>Insertion-based crossover (IBX): combining k routes from parent1 (one at a time) with subsets of customers from parent2. Insert within route crossover (IRNX): simultaneously combines k routes from parent1 with subsets of customers from parent2.</td>
<td>Large Neighbourhood Search (LNS) mutation. Each customer considered for reinsertion in an alternate route. Eliminate shortest route. Reorder customers within a route.</td>
<td>Simultaneous evolution of two populations (minimize distance and constraint violations). POP2 has smaller number of routes than POP1. When a new better solution is obtained from POP2, POP1 is replaced by POP2.</td>
</tr>
<tr>
<td>Repoussis et al. [71]</td>
<td>Discrete arc-based representation of individuals combined with a binary vector of strategy parameters. Individuals created using greedy insertion from a Restricted Candidate List (RCL), with a probabilistic element.</td>
<td>Multiple-parent recombination applied only to strategy parameters, not to problem solutions.</td>
<td>Mutation rate determined by strategy parameters evolved during the search. Mutation done using ruin and recreate.</td>
<td>(λ + µ) Evolution Strategy (ES): Starting from µ-individuals, at each iteration a new intermediate population of λ individuals is produced by mutation. Evolutionary search is based only on mutation. Each offspring is further optimised using route elimination and a local search (based on TS) to improve total distance.</td>
</tr>
</tbody>
</table>
The “ants” in ACO schemes could be termed “agents” and compared to the use of intelligent agents in, for example, agent-based simulation (ABS). In fact, ABS has been successfully used to solve VRPs [72–74].

Barbucha [73] describes a multi-agent co-operative search, where agents can exchange information about states or sub-problems via a shared memory. All co-operative search schemes share two features, namely a set of autonomous programs, each executing a particular solution method, and a cooperation scheme, which allows them to use each other’s information. These methods are parallel and hybrid, in that they most often combine other methods. Barbucha’s strategy is outlined in Figure 3.4, where “NSP” stands for “number of search procedures”. Search procedures proposed are $k$-opt or other neighbourhood exchange methods.

In highly constrained problems, agents can be taught, either online or from previous schedules, what constitutes a good decision at any juncture. Weightings can then be adjusted to enhance the likelihood of previously learned good choices being used in the future. Such learning processes are performed by weight updating processes in neural networks (NN), or pheromone trail updating in ACO.

Vokřínek et al. [74] propose using 3 different types of agents:

1. a task agent for processing of demands and allocation invocation,
2. an allocation agent to maintain allocation and the improvement process, and
3. a vehicle agent for route planning and optimisation.

The allocation agent must apply a defined strategy to allocation of customers to
routes/vehicles with a cost minimising objective. It works in two phases, an allocation phase and an improvement phase. The vehicle agent solves a TSP to find the best route.

Neural networks were employed by Potvin et al. [75] in competitive form to improve the construction phase of a parallel insertion heuristic for a VRPTW. In another study [76], neural networks are used in the selection of the best heuristic for a VRP based on the problem characteristics. Torki et al. [77] describe achieving good results using a competitive neural network to solve the TSP, the $m$-TSP and the CVRP. The TSP normally involves one “salesman” travelling to all the cities. $m$-TSP is the version of the TSP with multiple salesmen available to visit the cities.

**Pickup and Delivery Problems** General PDPs (GPDPs) occur frequently in practice in areas such as courier services, transportation of raw materials from suppliers to factories, food collection and delivery and newspaper delivery [26]. Parragh et al. [78] provide a survey and classification of GPDPs as shown in Figure 3.5. The following acronyms are used [79]:

- VRPB: VRP with backhauls.
- TSPCB: TSP with clustered backhauls.
- VRPCB: VRP with clustered backhauls.
- TSPMB: TSP with mixed linehauls and backhauls
- VRPMB: VRP with mixed linehauls and backhauls.
- TSPDDP: TSP with divisible delivery and pickup.
- VRPDDP: VRP with divisible delivery and pickup.
- TSPSDP: TSP with simultaneous delivery and pickup.
- VRPSDP: VRP with simultaneous delivery and pickup.
- VRPPD: VRP with pickups and deliveries

Some problems have full truckloads and others less than full truckloads. Delivery locations may be paired where no other customer can be visited in-between. A single commodity or multiple commodities may be involved, or the problem may involve the transportation of people.

Important GPDP constraints include [26]:
1. A precedence constraint to enforce the delivery to occur after the pickup.

2. A coupling constraint ensuring the pickup and delivery points are visited by the same vehicle.

3. A vehicle capacity constraint.

4. A time window constraint.

5. A maximum ride time constraint.

According to Hosny [26], the methods of choice for this class of problems are the GA, LNS, adaptive LNS (ALNS) and grouped GA.

In the PDVRP, every unit picked up can be used to satisfy every customer’s demand. In the PDP, every pickup is associated with a delivery point and both origin and destination must be served by the same vehicle. Delivery can only occur after pickup. In the DARP, constraints related to user inconvenience are also required. Constraints for TWs and maximum trip length can be included. Single vehicle cases are denoted SDARP and SPDP. For the DARP, static and dynamic versions occur in the literature.

The MV-PDPTW is a grouping and routing problem, where grouping refers to assigning vehicles to bookings, and routing involves the best routing for a vehicle to service the assigned bookings. For this reason these two aspects are handled separately by clustering first and then routing, or vice versa. Solutions are also generated using a two stage methodology [15]. Stage 1 involves constructing a
solution and stage 2 improving the solution. A typical construction methodology is a least-cost insertion algorithm.

The vehicle routing problem with backhauls (VRPB) involves the transportation of goods from the depot to customers and vice versa [26]. This type of problem can be one of four sub-types:

1. Vehicle routing problem with clustered backhauls (VRPCB): All delivery customers must be visited after all pickup customers.

2. Vehicle routing problem with mixed linehauls and backhauls (VRPMB): Mixed visiting of pickup and delivery customers is allowed.

3. Vehicle routing problem with divisible deliver and pickup (VRPDDP): Each customer is both a pickup and a delivery customer and two visits to the same customer are allowed.

4. Vehicle routing problem with simultaneous delivery and pickup (VRPSDP): Each customer requires a pickup and a delivery, but only one visit is allowed.

According to Hosny [26], for this class of problems the large neighbourhood search algorithm (LNS) is the heuristic of choice.

DARP services can be operated as static or dynamic, static being the case where bookings are known in advance, and dynamic where the bookings change on the day of operation and during execution.

Dynamic programming was used to achieve an exact solution to the DARP by Psaraftis [80] and Desrosiers [81] for the single vehicle case and up to 40 requests. Cordeau [82] uses a branch and cut algorithm for the static DARP. Valid inequalities previously developed for VRPs and PDPs, as well as new ones are used. Cordeau also applies various other rules for removing infeasible variables prior to solving and reducing problem size based on specific features of the problem.

For the multiple vehicle case, most algorithms are heuristics or metaheuristics [82]. A summary of heuristic literature for the static DARP has been compiled by Parragh et al. [78] and is shown in Table 3.4. Ranges of problem sizes and methods used are shown. Anything special about the problem is listed under column “Type”, and special constraints for the problem are listed under column “Constraints”. Insertion heuristics and clustering-based (mostly cluster first route second) methods are the dominant choices. The number of references in this field shows that heuristics have been the preferred solution methods in the past.
Table 3.4: DARP Heuristics [78]

<table>
<thead>
<tr>
<th>Reference</th>
<th>Type</th>
<th>Objective</th>
<th>Constraints</th>
<th>Algorithm</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Single vehicle case</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Psaraftis [83]</td>
<td></td>
<td>Minimise routing cost</td>
<td></td>
<td>MST heuristic, local interchanges</td>
<td>Up to 50 requests</td>
</tr>
<tr>
<td>Psaraftis [84]</td>
<td></td>
<td>Minimise routing cost</td>
<td></td>
<td>Adapted 2-opt and 3-opt</td>
<td>Up to 30 requests</td>
</tr>
<tr>
<td>Sexton and Bodin [85][86]</td>
<td></td>
<td>Minimise customer inconvenience</td>
<td>Desired delivery time</td>
<td>Routing and scheduling algorithm based on Bender’s decomposition</td>
<td>Up to 20 requests</td>
</tr>
<tr>
<td>Healy and Moll [87]</td>
<td></td>
<td>Minimise routing cost</td>
<td></td>
<td>2-opt improvement, optimising/sacrificing phases</td>
<td>Up to 100 requests</td>
</tr>
<tr>
<td><strong>Multi-vehicle case</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cullen et al. [88]</td>
<td></td>
<td>Minimise routing costs</td>
<td></td>
<td>Cluster first, route second, column generation</td>
<td>Up to 50 requests</td>
</tr>
<tr>
<td>Roy et al. [89][90]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing costs,</td>
<td>Time windows</td>
<td>Parallel insertion</td>
<td>Up to 578 requests</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minimise customer inconvenience</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bodin and Sexton [91]</td>
<td></td>
<td>Minimise customer inconvenience</td>
<td>Desired delivery time</td>
<td>Cluster first, route second</td>
<td>Up to 85 requests</td>
</tr>
<tr>
<td>Jaw et al. [92]</td>
<td></td>
<td>Minimise routing costs,</td>
<td>Time windows, ride time</td>
<td>Sequential feasible insertion algorithm</td>
<td>Up to 2617 requests</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minimise customer inconvenience</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reference</td>
<td>Type</td>
<td>Objective</td>
<td>Constraints</td>
<td>Algorithm</td>
<td>Size</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------</td>
<td>----------------------------------------</td>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Alfa [93]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing costs</td>
<td>Time windows, ride time</td>
<td>Adaption of heuristics used by Jaw et al. [92]</td>
<td>Up to 49 requests</td>
</tr>
<tr>
<td>Psaraftis [94]</td>
<td></td>
<td>Minimise routing costs, Minimise customer inconvenience</td>
<td>Time windows, ride time</td>
<td>Comparison of Jaws heuristic and grouping-clustering routing heuristic</td>
<td></td>
</tr>
<tr>
<td>Desrosiers et al. [95], Dumas et al. [96]</td>
<td>Multi depot</td>
<td>Minimise routing costs</td>
<td>Time windows</td>
<td>Mini-clustering algorithm, column generation</td>
<td>Up to 200 requests</td>
</tr>
<tr>
<td>Kikuchi and Rhee [97]</td>
<td></td>
<td>Max. number of customers served</td>
<td>Time windows</td>
<td>Sequential insertion</td>
<td>Up to 200 requests</td>
</tr>
<tr>
<td>Desrosiers et al. [98]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing costs</td>
<td>Time windows, ride time</td>
<td>Improved mini-clustering algorithm of Desrosiers et al. [95]</td>
<td>Up to 2411 requests</td>
</tr>
<tr>
<td>Potvin and Rousseau [38]</td>
<td></td>
<td>Minimise routing costs</td>
<td>Time windows, ride time</td>
<td>Constraint-directed search (beam search)</td>
<td>Up to 90 requests</td>
</tr>
<tr>
<td>Ioachim et al. [32]</td>
<td>Heterogeneous fleet, multi depot, service time</td>
<td>Minimise number of vehicles, Minimise ride time</td>
<td>Time windows, ride time</td>
<td>Mini-clustering, column generation</td>
<td>Up to 2545 requests</td>
</tr>
<tr>
<td>Reference</td>
<td>Type</td>
<td>Objective</td>
<td>Constraints</td>
<td>Algorithm</td>
<td>Size</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>-----------------------------------------------</td>
<td>---------------------------------</td>
<td>-----------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Madsen et al. [99]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing cost, minimise number of vehicles, minimise total waiting time, minimise deviation from promised service</td>
<td>Time windows, ride time, route duration</td>
<td>REBUS insertion based algorithm</td>
<td>Up to 300 requests</td>
</tr>
<tr>
<td>Toth and Vigo [100]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing cost, minimise number of vehicles</td>
<td>Time windows, ride time</td>
<td>Parallel insertion, improved by trip insertion, exchange, double insertion moves</td>
<td></td>
</tr>
<tr>
<td>Bornsforder et al. [101]</td>
<td>Heterogeneous fleet, multi depot, service time</td>
<td>Minimise routing cost, minimise number of vehicles</td>
<td>Time windows, route duration, (ride time)</td>
<td>Cluster first route second, set partitioning, branch and bound</td>
<td>Up to 1771 requests</td>
</tr>
<tr>
<td>Fu [102]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing cost, minimise customer inconvenience</td>
<td>Time windows, ride time</td>
<td>Parallel insertion, stochastic travel times</td>
<td>Up to 2800 requests</td>
</tr>
<tr>
<td>Diana and Dessouky [103]</td>
<td></td>
<td>Minimise routing cost, minimise customer inconvenience, minimise idle times</td>
<td>Time windows, ride time</td>
<td>Parallel regret insertion heuristic</td>
<td>Up to 1000 requests</td>
</tr>
<tr>
<td>Xiang et al. [104]</td>
<td>Heterogeneous fleet</td>
<td>Minimise routing cost, minimise number of vehicles</td>
<td>Time windows, route duration, break time between trips</td>
<td>Construction improvement; clustering by time windows, ideas of sweep heuristics, local search improvement</td>
<td>Up to 150 requests</td>
</tr>
<tr>
<td>Reference</td>
<td>Type</td>
<td>Objective</td>
<td>Constraints</td>
<td>Algorithm</td>
<td>Size</td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>-----------</td>
<td>-------------</td>
<td>-----------</td>
<td>------</td>
</tr>
<tr>
<td>Wong and Bell [105]</td>
<td>Heterogeneous fleet, service time</td>
<td>Minimise routing cost, Minimise customer inconvenience</td>
<td>Time windows, ride time, route duration</td>
<td>Parallel insertion, improved by trip insertion</td>
<td>Up to 150 requests</td>
</tr>
<tr>
<td>Wolff Calvo and Coloni [106]</td>
<td>Max. number of customers served, max. service level</td>
<td></td>
<td>Time windows</td>
<td>Cluster first route second, assignment heuristic, vertex reinsertions</td>
<td>Up to 180 requests</td>
</tr>
</tbody>
</table>
A large scale (500 to 1000 requests) DARP was solved using a parallel regret insertion heuristic by Diana and Dessouky [103]. The problem is described as extremely highly constrained with tight time windows to ensure a high quality of service. A review of past research suggested to the authors that insertion methods are the best to be used on routing problems with time windows. Parallel insertion was chosen as it tends to outperform sequential insertion. The first part of the procedure involved selecting a seed customer. One way is to choose customers with the earliest pickup times; however, for better performance the spatial positioning also needs to be taken into account. Choosing less decentralised customers prevents inconvenient and difficult to insert requests being left to last.

The regret insertion algorithm starts by finding a best insertion cost for each un-routed request. A table is created with requests as rows and routes as columns. The request regret is the difference for each route from the minimum in the row. The request with the largest regret is inserted.

Jaw et al. [92] propose a sequential insertion procedure in which customers are first ordered by increasing earliest pickup time. Then they are inserted according to the cheapest feasible insertion criterion.

The DARP is addressed using a cluster first, route second approach by Borndorfer et al. [101]. Both problems are modelled as set partitioning problems, with the routing problems only being solved approximately using a branch and bound algorithm.

Recently, interest in metaheuristics for DARP has increased significantly [107]. Notably, Cordeau and Laporte [8] use tabu search with great success. Toth and Vigo [100] use a local search based heuristic in the form of a tabu thresholding algorithm for the multi-vehicle DARP. Initial construction uses parallel insertion.

Table 3.5 [78] shows the use of metaheuristics for DARPs. Jorgensen [108] used a space-time network to define the nearest neighbour. Tabu search and genetic algorithms are the methods of choice. Most recent work appears to be concentrated on these two metaheuristic methods.

Comparisons between cases in the literature are complicated due to the fact that many of the problems have been inspired by real-world instances, and none have been applied to standardised data sets [14]. When similar data sets are used, different objectives are considered. In general, metaheuristic methods give better solutions, but heuristic methods are faster.
<table>
<thead>
<tr>
<th>Reference</th>
<th>Type</th>
<th>Objective</th>
<th>Constraints</th>
<th>Algorithm</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Colorni <em>et al.</em> [109]</td>
<td>Multi Vehicle Case</td>
<td>Max. number of customers served, Minimise customer inconvenience</td>
<td>Route duration</td>
<td>Simulated annealing</td>
<td>Up to 100 requests</td>
</tr>
<tr>
<td>Toth and Vigo [100]</td>
<td>Heterogeneous fleet, multi depot</td>
<td>Minimise routing cost</td>
<td>Time windows, ride time</td>
<td>Parallel insertion algorithm, tabu thresholding</td>
<td>Up to 300 requests</td>
</tr>
<tr>
<td>Baugh <em>et al.</em> [110]</td>
<td></td>
<td>Minimise number of vehicles, Minimise routing cost, Minimise customer inconvenience</td>
<td>Time windows, ride time</td>
<td>Simulated annealing</td>
<td>Up to 300 requests</td>
</tr>
<tr>
<td>Uchimura <em>et al.</em> [111]</td>
<td></td>
<td>Minimise routing cost</td>
<td>Ride time, route duration</td>
<td>Genetic algorithm</td>
<td>10 requests</td>
</tr>
<tr>
<td>Cordeau and Laporte [8]</td>
<td>Service time</td>
<td>Minimise routing cost</td>
<td>Time windows, ride time, route duration</td>
<td>Tabu search</td>
<td>Up to 295 requests</td>
</tr>
<tr>
<td>Aldaihani and Dessouky [112]</td>
<td>Mix with fixed route transit</td>
<td>Minimise routing cost, Minimise customer inconvenience</td>
<td>Time windows</td>
<td>Tabu search</td>
<td>Up to 155 requests</td>
</tr>
<tr>
<td>Ho and Haugland [113]</td>
<td>Probabilistic</td>
<td>Minimise routing cost</td>
<td>Time windows, ride time</td>
<td>Tabu search, hybrid GRASP-tabu search</td>
<td></td>
</tr>
</tbody>
</table>
Table 3.5 Contd.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Type</th>
<th>Objective</th>
<th>Constraints</th>
<th>Algorithm</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Multi Vehicle Case</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Melachrinoudis et al. [114]</td>
<td>Heterogeneous fleet, multi depot</td>
<td>Minimise routing cost, Minimise customer inconvenience</td>
<td>Time windows</td>
<td>Tabu search</td>
<td>Up to 8 requests</td>
</tr>
<tr>
<td>Rekiek et al. [115]</td>
<td>Service time, multi depot, heterogeneous fleet</td>
<td>Minimise number of vehicles, Minimise customer inconvenience</td>
<td>Time windows, ride time, route duration</td>
<td>Grouping genetic algorithm</td>
<td>Up to 164 requests</td>
</tr>
<tr>
<td>Jorgensen et al. [108]</td>
<td>Service time</td>
<td>Minimise routing cost, Minimise ride time, Minimise time window violations</td>
<td>Time windows, ride time, route duration</td>
<td>Genetic algorithm, space-time nearest neighbour heuristic</td>
<td></td>
</tr>
<tr>
<td>Parragh et al. [116]</td>
<td>Service time</td>
<td>Minimise routing cost, mean ride time</td>
<td>Time windows, ride time</td>
<td>Variable neighbourhood search, path relinking</td>
<td></td>
</tr>
</tbody>
</table>
Exact Methods  There is a long history of research in exact methods for VRPs. Due to the fact that these problems are NP-hard, complex solution methods are devised, including those involving decomposition, column generation, row-generation, or combinations thereof.

A survey of exact methods for the VRP has been done by Laporte [29]. His definition of a VRP covers situations with a depot and customers. Exact methods can be classified as direct tree search methods, dynamic programming or integer linear programming (ILP).

Laporte et al. [117] [118] describe a formulation for CVRPs which exploits the fact that an $m$-TSP can be transformed into a 1-TSP by introducing artificial depots. The problem can be solved using a branch-and-bound process where the sub-problems are assignment problems, as in the assignment-based solution to the TSP (as described in [119]). Several variations of the basic VRP have been solved.

The $k$-degree center tree algorithm for the MCVRP was developed by Christofides et al. [120]. They divide the edges into four subsets and devise a special formulation accordingly.

Set partitioning has become popular in exact algorithms. In these formulations, routes are chosen from sets of depots which make up feasible routes with a cost minimizing objective [119]. Because the variable sets can be huge, column generation is the natural way to address these types of problems. In such formulations, any feasible solution is generated with only a few variables, then new routes are added depending on the marginal cost of other possible routes. Finding these new routes to add can, in some cases, be a challenge, and dynamic programming has been used or shortest path [119] has been used, depending on problem features. Because of the integer variable constraints, the procedure must be used in conjunction with a branch-and-bound algorithm.

Three-index formulations refer to formulations where each variable $x_{ijk}$ refers to a specific vehicle or fleet $k$ using an arc $ij$, where the variable is binary and equal to 1 if the arc is used [119]. Two-index formulations are where variables $x_{ij}$ refer to an arc $ij$ only.

A number of valid inequalities (cutting planes) have been proposed for various VRPs. One such approach is that of Cordeau [82]. Chandran and Raghavan [121] demonstrate the use of tree structures to solve CVRPs and other VRPs. In special cases, when a problem can be represented in tree form, a number of valid inequalities and
special formulations are applicable.

Exact methods for VRPPDs have been surveyed by Cordeau et al. [122]. Formulations are provided for the VRPPD with time windows (VRPPDTW). Early DARP exact formulations made use of dynamic programming (DP). Thereafter, few if any exact formulations have been proposed, with heuristics and metaheuristics the most commonly used techniques. According to Parragh et al. [14], the reason exact methods are not used for the DARP is that the concept of “optimal solutions” becomes debatable for these types of problems.

3.2 Airline Scheduling

3.2.1 Scheduled Airlines

Schedule Planning and Generation

The airline schedule planning problem can be disassembled into the schedule design problem, the fleet assignment problem, the maintenance routing problem and the crew scheduling problem [123]. Because of the size of typical scheduled flight (or commercial) airline companies, these problems are most often each addressed separately and often iteratively. Some work has been done on integrating some or all of these problems. Much work has been done on solution methodologies for optimising the resultant large models. Solutions are most often generated using linear programming models of various types, including network flow and integer linear programming (ILP). Because of the massive sizes of models, heuristics are often an important part of the modelling and solution process. An overview of formulations and solution methods is provided by Gronkvist [2].

Schedule Design Problem

Schedule planning involves maximizing revenues [124] by providing services to meet expected demand. Constraints include gate facilities, available slots, connections, etc. The schedule should be flyable with available resources. Flight legs are designed, each leg consisting of one aircraft flight, and with an associated origin and destination (O-D) and time of departure and time of arrival. Typically airlines must consider peak passenger times and frequency of service, size of aircraft, competition and the effect the provision of the services will have.
After generation of the schedules, fleet assignment and aircraft routing can be performed.

**Fleet Assignment Problem**

**Description**  This is the process of assigning the optimum aircraft type to each leg in a schedule, to maximize revenues and minimise costs. This type of problem is often formulated as a multi-commodity network flow problem (MCNFP) with aircraft flowing through the network. The formulation can be done using a timeline network (Figure 3.6), sometimes also called a time-space network, with arcs representing flights. The different aircraft types or fleets are then the commodities.

Unfortunately such MCNF models cannot use the computationally efficient network simplex method and must be solved using integer programming methods. Maintenance considerations can be taken into account by adding constraints accordingly. Although crew scheduling is a separate and extensive modelling effort, crew can be taken into account in the fleet assignment phase such that more feasible and cheaper crew schedules are likely to be available in the crew scheduling phase [2].

Kontogiorgis [125] describes using A Mathematical Programming Language (a commercial mathematical modelling language, AMPL) to assign fleets for US Airways. US Airways typically flies 2500 flights to more than a hundred markets using 400
aircraft of 14 different types. The model maximizes revenue and reduces costs. The solution procedure involves solving the linear programming (LP) relaxation, fixing variables and finally solving the reduced mixed integer program (MIP) with a branch and bound solver to find the remaining variables. According to Rushmeier et al. [124], such a problem could have 75000 binary variables, 1000 integer variables and 50000 constraints. Normally a solution is required within a day. The solution process involves preprocessing, aggregating of nodes and other size reduction heuristics. Then, based on an LP relaxation, a number of variables are fixed. After this, a model may have 12% of the rows and 20% of the variables remaining. Subsequently an integer solution is sought using a branch and bound method.

In some cases, fleet assignment is not necessary, for example, if the airline uses one fleet type [126], and only aircraft routing and crew pairing is done. These are solved iteratively to achieve a series of feasible solutions. Ray and Tomlin [127] address the aircraft routing problem by using an IP model with time-space network and time windows. A heuristic and LP relaxation plus rounding technique are suggested to find good solutions.

**Time-Space Networks** Timespace networks have been used many times in literature to represent aircraft fleet assignment and/or routing problems and thereby create model formulations. An example is the network used by Barnhart [3]. The most commonly used time-space network representation is shown in Figures 3.6 and 3.7. Each node has a geographical position and time associated with it and arcs represent movement of aircraft between nodes. Arcs can represent flight legs or they may be “ground arcs” and represent time when an aircraft is stationary on the ground at a specific location.

For such a model, aircraft may be grouped by fleet, such that a fleet consists of similar, interchangeable aircraft, in which case a fleet will represent a commodity in a multi-commodity network flow problem. The addition of passengers represents another type of commodity, where passenger groups with different origin-destination (O-D) pairs will each represent a different commodity. Passenger movement must then be linked to aircraft movement using aircraft capacity constraints [128].

A “passenger leg” will typically refer to a passenger group travelling on a single, specific flight leg. Similarly, “passenger ground arcs” represent time spent by passenger groups waiting on the ground at some location.
**Time-Discretised Networks**  The most commonly used formulation method uses time discretisations. In these formulations, time is discretised into suitably sized time steps and a node is placed at each step, in each location, as shown in Figure 3.7.

Depending on problem requirements, these networks are required for each of the individual aircraft, each fleet type, and/or each passenger or each passenger group.

Using this method the number of variables for a MCNFP can be calculated as follows:

- Number of locations \( N_c \)
- Number of time steps \( N_{ts} \)
- Number of fleets \( N_f \)
- Number of passenger groups \( N_{pg} \)
- Number of nodes
  \[
  N_n = N_c \times N_{ts} + N_c \tag{3.1}
  \]
- Number of fleet legs
  \[
  N_{fl} = N_n \times (N_c - 1) \times N_f \tag{3.2}
  \]
- Number of passenger legs
  \[
  N_{pl} = N_{fl} \times N_{pg} \tag{3.3}
  \]
• Number of fleet ground arcs

\[ N_{fr} = (N_n + N_c) \times N_f \]  

(3.4)

• Number of passenger group ground arcs

\[ N_{pr} = (N_n + N_c) \times N_{pg} \]  

(3.5)

• Number of variables in problem

\[ N_v = N_{fl} + N_{pl} + N_{fr} + N_{pr} \]  

(3.6)

The model time may consist of a single day, or longer periods, such as a week. For this work, a single day is used. It should be noted that the number of fleet ground arcs calculated \( N_{fr} \) includes ground arcs for each fleet between every node, as well as a ground arc into the first node at every location at the start of the day, and a ground arc out of the last node at every location at the end of the day.

With reference to Figure 3.7, the number of flight legs starting from a single node is equal to \( N_c - 1 \) for each fleet type. The arrival time of a flight leg is the departure node time plus the flight time. This flight time will fall between the times of two nodes at the destination location. The later of these two nodes is assigned as the leg arrival node.

It should be noted that flight legs that depart near the end of the day may have arrival times which are beyond the end-of-day time, so no nodes can be assigned as arrival nodes. These flight legs are not included in the problem. There is no way to predict how many flight leg variables will be left out for this reason before the problem parameters are known. The equation shown for number of fleet legs (3.2) includes these legs and will therefore produce a higher figure than will actually be the case for any instance.

Unfortunately in practice the size of these problems quickly grows impractically large in terms of variables for even small-sized problems. For example, using the formulas (3.1) to (3.6), a 7 city, 26 passenger group, 2 fleet type, 12 hour day problem with 10 minute steps will result in 177604 variables, 157248 of which are passenger group leg variables.

In many such problems most of the variables are aircraft flight legs or passenger legs and will never be used. The obvious solution is to use heuristics to leave out as many unlikely-to-be-used variables as possible. These heuristics must be developed after careful study of the specific problem.
**Maintenance Routing Problem**

The aircraft maintenance routing problem involves assigning a particular aircraft to each flight leg [123]. Consistency with the fleet assignment problem must be maintained. Each aircraft must be assigned a feasible sequence of flight legs such that the periodic maintenance checks for that specific aircraft can be undertaken within legal requirements.

**Crew Scheduling Problem**

This problem involves determining the cost-minimising assignments of crew to each leg in the aircraft schedule [123]. The schedule will specify activities for a crew member typically for a period of a month. All labour regulations must be satisfied.

The crew scheduling problem was the first of the airline scheduling sub-problems to receive attention in 1969, see Arabeyre et al [129] for details. This was because of the large size of such problems in practice. Typically, many possible pairings (i.e. crew-aircraft combinations) are generated and the best combination (minimum cost) is sought using integer programming methods [124]. Since then, many others have considered the problem using various techniques.

**Integrated Models**

Integrated scheduling refers to combining routing, scheduling, fleet assignment and crew scheduling [130]. This has been addressed by Barnhart [131] using flight strings. Papadakos [130] presents an integrated model including fleet assignment, aircraft maintenance routing and crew scheduling. The problem is solved using enhanced Benders decomposition and accelerated column generation.

A warm-start type of model was developed by Stojkovic et al [132]. The model re-optimizes a schedule after a perturbation (disruption), and therefore must be able to generate real-time data. This problem is often referred to as the day-of-operation changes model [133]. The problem was a linear programming problem, and the dual of the problem was a network problem and therefore relatively easily solvable using the network simplex method.

An integrated scheduling method is devised by Haouari et al. [134]. They use a 2-phase network flow heuristic which starts with clustering flights and assigning
aircraft to flights using a greedy heuristic. Thereafter a series of assignment problems are solved, and a further set of MCNF problems solved in phase 2.

3.2.2 Charter Airlines

Charter airline scheduling has been addressed by a number of researchers. This work has been mostly restricted to the case where demand need not be completely satisfied. Erdman et al. [133] have proposed a schedule generation problem for a charter airline to maximize profits. The model is a capacitated network model with side constraints (i.e. an ILP). Variables relate to aircraft rotations (daily routes or itineraries). Since the model provides weak relaxations, cutting plane methods of relaxation tightening are proposed. Essentially, the constraint causing the weak relaxations was identified as the constraint restricting the number of passengers transported due to the aircraft capacity. This was addressed by adding suitable cutting plane constraints to limit passenger demand to aircraft size.

Kim and Barnhart [135] consider the charter airline service network design problem by developing exact and heuristic network design models. They develop a fast method for a single fleet model and adapt it to create a heuristic approach to the multiple fleet model. Weak relaxations were improved by adding constraints forcing at least one aircraft to be assigned to a group of passengers. For large instances, a column generation scheme is proposed where aircraft itineraries with reduced costs are sought by means of shortest path type sub-problems.

Yan and Tseng [128] developed a demand-driven integrated fleet routing and flight scheduling model for a charter airline company, i.e., where not all demand needed to be met. They use a fleet flow time-space network for each fleet type, as well as a passenger flow time-space network for each demand (O-D) pair. Time discretisations are used. The model is a cost minimising multi-commodity network flow problem. The resulting large-sized problem is solved using Lagrangian relaxation, a sub-gradient method, specialised heuristics and a custom-designed solution sequence. The model was applied to a Taiwanese airline with 11 cities served by 170 daily flights and 2 fleet types. The model had 9504 nodes, 25558 arcs and 10407 constraints.

3.2.3 On-Demand Air Transportation

In 2004, Cordeau et al. [5] compiled a literature review on the transportation on demand (TOD) types of problems. They found little or no literature for the static
dial-a-flight problem (DAFP) and therefore presented an ILP formulation. The static version must schedule a known number of demands in a day. The dynamic version of the DAFP addresses the problem where passengers may arrive and request service immediately, but then may be denied service at some cost to the provider. The suggested time-discretised multi-commodity network flow problem becomes large quickly, and specialised methods are required to solve instances involving 15 to 30 airports and 5 to 10 airplanes. It should be noted that the problem at hand matches this type of problem. Therefore, the problem descriptions “DAFP” and “airline taxi” are used interchangeably in this work.

Courier services operate a type of on-demand delivery service. Armacost et al. [136] describes a system developed to schedule packages for United Parcel Service (UPS), an American courier company. UPS delivers 13 million packages globally every day. The model determines aircraft routes, fleet assignments and package routings. The main problem with the formulation used, as with many such problems, was the weakness of relaxations. A solution involved redefining the decision variables as composite variables. Effectively, instead of using the actual aircraft decision variables, new variables specifically designed to have sufficient capacity to carry the full parcel loads that are needed to be transported are defined. For example, if one aircraft type is too small to carry the load, another decision variable is introduced which represents two of that aircraft type in one variable.

Business and service considerations for the DAFP include how many stops a passenger may experience, passengers changing planes at stops, waiting times and total journey times (Cordeau et al. [5]). In terms of costs, short flights use more fuel per km of distance flown than long flights. Fuel consumption is also dependent on how heavily loaded the aircraft is.

The turnaround time refers to the time it takes for an aircraft to be ready to take off after having landed and stopped to offload and take on new passengers. There are varying amounts of turnaround time necessary, depending on factors such as whether passengers and their baggage must change aircraft, number of passengers in the aircraft, whether refueling is required and aircraft size.

An on-demand air transportation type of problem was addressed by Ronen [137]. A set of revenue trips needed to be partially satisfied by a fleet of aircraft at minimal cost, and any remaining flights were to be sold to other operators. A large set of feasible candidate schedules is generated for each aircraft using heuristics. The best set is then selected using an MIP-type solver.
Lee et al. [6] address the air taxi service problem with probabilistic variables. They model the situation where passengers must request the service in advance, as opposed to simply arriving when transport is needed. Discrete event simulation methods to maximize revenues are used.

The most recent literature dealing with the airline taxi problem is that of Espinoza et al [7]. The research dealt with the hiring of executive jets in the USA and is termed the per-seat, on-demand air transportation problem (PSOD). This problem description closely matches that of the problem at hand, suggesting the PSOD is similar to the airline taxi problem to be addressed.

In such a problem, business people provide a required origin-destination (O-D) pair (demand) and timing window a few days or one day in advance, and the jet provider must assemble a schedule to accommodate all of these bookings. A time-discretised network model resulted in a multi-commodity network flow model with side constraints. Heuristics designed to eliminate obviously infeasible routes are captured in the form of a rolling forward algorithm. These heuristics relate to plane capacity, timing constraints such as time of day, elimination of sequences which involve two consecutive flights without passengers on board, eliminating later departures in flight sequences if earlier departures are available. Time windows are also intelligently chosen such as to coincide with key moments of the day, hopefully eliminating the need for many variables. Aggregation of nodes was carried out to further reduce the network size. This involved eliminating arcs and nodes. Problems of up to 81 requests, 8 jets and 17 airports are solved. A second paper [138] describes improved neighbourhood search techniques and parallelisation of the search to achieve solutions for larger instances.

The on-demand air transportation problem or DAFP or airline taxi problem is to some extent similar to certain vehicle routing problems such as PDPs, since both have pickups and deliveries and time windows. It differs from the DARP in that, in an airline taxi scheduling problem, all passengers must be picked up and dropped off at predefined points, namely airports, whereas in the DARP, passengers can be picked up and dropped off anywhere on the network.

This means there are likely to be fewer locations in a DAFP than a DARP for a similar number of bookings, allowing a smaller-sized formulation for the DAFP. Different heuristics might be applicable. In practice in a DAFP, it is often acceptable from a quality of service point of view to drop off passengers at a location such that they can be collected by another vehicle to continue their journey. This is not only unlikely to be acceptable in a DARP, but, due to the random positioning
of each passenger pickup and delivery points, there should be fewer cost-effective opportunities to do this.

3.3 Solving large MIP problems

Aircraft scheduling problems often have large numbers of variables, including many binary variables. Integer and continuous variables may also be present. Such problems can be easier to solve by, for example, reducing variable count at the expense of more constraints, or by exploiting special structures related to the binary variables [139]. More constraints help with respect to integer solution methods by reducing the size of the search space, but can slow down the solution of the linear programming (LP) relaxations.

In practice, various mathematical solution algorithms and methods are used, including decomposition, column generation, branch, price and cut. The objective function can be modified by means of Lagrangian multipliers to effectively remove difficult constraints and result in a problem with a simple structure. This might be done, for example, to allow the model to be solved using an efficient network algorithm [2]. Typically, the aircraft capacity constraints are dealt with in this manner.

Column generation is a well-established technique which involves progressively adding variables to the problem and proceeding with LP relaxation solutions accordingly [140]. The technique applied to IP is referred to by the term ‘branch and price’. Branch and price has become popular for many VRP problems. Examples include those of Salani, Fukasawa et al. and Prescott-Gagnon et al. [141–143]. There are issues which arise and make these methods more suited to achieving good solutions than optimal solutions. For example, the method normally involves a reformulation of the problem such that variables inherently contain more information. This allows each variable to be priced to determine if it should be in a solution or not. The problem then revolves around how to make up or select feasible integer variables that should be included in the problem. This search procedure is done at each branch of the branch and bound process, and is sometimes done using dynamic programming (DP) or metaheuristics. The technique has been successfully applied to a number of routing-type problems.

Branch and cut involves leaving constraints out of the LP relaxation. Then, if a solution is infeasible, a subproblem (the separation problem) is solved to identify violated inequalities. Some of these are added to the LP to cut the infeasible region
off. Branching occurs when no violated inequalities are found. Both techniques can be used on a problem simultaneously.

Feaspump [22] is a heuristic method which works by considering two solutions, the feasible LP relaxation optimum and an integer solution which is the LP rounded solution and thus may or may not be optimum. The idea is to search in the direction of the LP optimum for a better integer solution. This is done by finding the closest LP feasible point to the current infeasible integer solution. The method first addresses the binary variables, then having fixed them, attends to the general integer variables.

Benders [23] proposed what is now called Bender’s decomposition. Effectively, the MIP problem is decomposed by variable type. This or modifications thereof may have advantages in practice since efficient methods can often be developed to solve problems with purely one kind of variable, but not mixed types. Papadakos [130] uses Benders decomposition in this way in a necessarily large integrated model for airline scheduling.

Kamath et al. [144] and Karger and Plotkin [145] have developed fast approximation algorithms for the minimum cost multi-commodity network flow problem. The algorithm starts with zero flow and at each iteration, identifies incremental flows in the network.
3.4 Conclusions from Literature Review

There has been a substantial amount of work done on heuristics for VRP problems in general. In recent heuristic research, construction heuristics such as greedy heuristics, insertion heuristics and clustering, including cluster-first route-second methods, are favoured.

The latest dial-a-ride problem research is dominated by tabu search and genetic algorithm based methods. A wide range of sizes of problems have been approached with these methods. Exact methods tend to be avoided for multi-vehicle pickup and delivery problems in particular.

Work on agent routing, including ACO, for VRPs is becoming more popular. Apart from ACO and other nature-based methods, there seems to be little, if any, work on training of agents in agent-based routing.

There is very little work on the airline taxi problem in published literature.
4 Methodology

4.1 Overview

The type and size of the problem at hand suggests it could be approached by applying similar techniques as have been used for the DARP, such as the use of improvement heuristics or local search-based metaheuristics. Compared to the large DARP described by Diana and Dessouky [103], this problem is relatively small and less tightly constrained, since many of the customers have wide time window requirements. However, it would be regarded as fairly tightly constrained compared to many VRPTW research problems, such as the VRPTWs used by Solomon [25]. In general, heuristic methods give lower quality solutions than metaheuristic methods [5]. The nature of the problem suggests solution methods for reasonably tightly constrained and medium-sized problems, such as exact or metaheuristic methods. Very little work has been done on construction methods which is the focus of this research. Theoretically, if good solutions can be obtained in a reasonable time using a construction method, improvement heuristics might not be needed, or need less time to improve the solution.

A reason to avoid the use of purely heuristic methods is the cost of the service. Since the service is provided by an expensive vehicle type, the cost of a less than optimal solution is high compared to a road-based taxi service.

In this work, a pure version of the airline taxi problem was addressed using the relevant Sefofane Air data, but leaving out certain constraints. A small problem was created to test the methods and for benchmarking purposes. Thereafter, the problem with all constraints was addressed.
Constraints that were relaxed included:

- No forced end-of-day positions for aircraft for certain versions of the problem.
- No special considerations with regard to the number of crew required or special customer requests (customers can request special arrangements for their journey).

For PDPs, particularly real-world problems, it is considered difficult to construct feasible solutions [15]. However, existing literature is concentrated around improvement heuristics and metaheuristic techniques, which need such a feasible solution to begin. Metaheuristic techniques are beneficial because the actual, constrained feasible region in these problems can be small. Therefore methods to produce good starting feasible solutions for such problems are required.

The full problem, as experienced by Sefofane Air, is first described in detail in Chapter 5. Thereafter various approaches to the solution of the air taxi problem that underlies it are detailed.

The problem described in this work can be addressed using an exact ILP and problem size reduction heuristics, as done by Espinoza et al. [7]. Such an exact method is presented in Chapter 6, together with newly-developed heuristics in Chapter 7. This method involves the use of a multi commodity network flow (MCNF) integer linear program (ILP) with time discretisations (TDs). This method was initially devised purely to obtain benchmark solutions as close to optimality as possible. However, when combined with heuristics devised for this problem, it will be a practical alternative for smaller instances, possibly up to 80 requests. Two heuristics are developed in Chapter 7 for this problem. They are designed to reduce the number of variables in the problem by aggregation of customer groups and by eliminating cost-ineffective routes.

Another approach to the problem is to combine exact ILP methods with construction heuristics, by using heuristics to construct feasible route sections and use those as variables in ILP formulations (as done by Ronen [137] and Armacost et al. [136]). This way, more information is captured in each variable than a typical exact formulation, and fewer variables will result. It is these ideas which motivate the following methods which are proposed in this research:

1. **MCNFP with time discretisations (TDs): Chapters 6 and 7** - A group aggregation heuristic and a geographically-inspired heuristic are designed to
reduce the number of variables associated with the passenger group networks in the standard MCNFP of Chapter 6. Although this method was initially included to obtain a lower bound for the problem, it will be useful for smaller problem instances.

2. **Variable construction using agent routing information combined with MCNFP: Chapter 8** - This method is similar to that described in method 1, except it only uses well-placed nodes and arcs in the networks. This information is provided by the use of agents.

3. **Composite variable MCNFP: Chapter 9** - This method creates variables where each variable inherently contains more information than the variables in methods 1 and 2. Specifically, the variables are created such that no passenger networks are required, greatly reducing the size of the problem, particularly for the problem described here.

In order to evaluate whether these methods may be useful in other VRPs, the composite variable MCNF method (method 3) was applied to an additional VRP, namely the MVCVRP. The agent route variable generation method (method 2) was applied to the CVRPTW.

All models except those in Chapter 5 were generated and solved on a Mecer PC with Intel i7 8 core processor, 8 GB RAM, Windows 7, and with Matlab 7.11, Excel 2010 and CPLEX 12.1 software packages. Chapter 5 models were solved using GUROBI solver provided by NEOS [146].

### 4.2 Small-Scale Benchmark Airline Taxi Problem

For the pure airline taxi problem, a small problem was compiled in addition to the given, relaxed and full-sized problem, such that various techniques could be tested and for benchmark comparison purposes. The problem was formulated from the real data such that it was suitable in terms of size and solution speed for the evaluation of the heuristics and for comparing results from different models. It also had to be representative geographically of the spread of landing strips (or locations) in the full-size (large) problem.

In order to ensure the small problem was representative of real data which may actually occur, the problem was compiled by using Monte Carlo random sampling from the full table of 158 bookings. In doing this, the locations which appear
in the smaller problem should be nearly as scattered and widespread as in the larger problem. If this is not the case, the heuristics devised may not work in the same manner on both problems, particularly if those heuristics are related to the geography of the locations.

The first seven locations in the full city list (see Figure 5.3 and Appendix A, Table A.4) were chosen to be included in the smaller schedule, since they included the hub at Maun and satisfied the requirements with regard to geographic spread. A sampling table was assembled consisting of all bookings from the full list which only included these seven locations.

Individual bookings were randomly sampled using a uniform distribution with replacement from this sampling table. Replacement is done to ensure samples are independent. In this case, many samples are in any case similar and obtaining more than one of the same sample does not affect the obtained sample or the solution process in an adverse manner. The sampling process continued until 40 bookings had been compiled.

4.3 Sefofane Air Solution

One of the solutions obtained for the full-sized model was used to demonstrate how Sefofane Air would use this solution method in practice. The solution is compared with the actual solution used on the day in question.

4.4 Application to Other VRPs

The composite variable MCNF method was applied to a real-world MVCVRP to evaluate whether it may be useful for different types of VRP problems. The MVCVRP was inspired by a real-life logistics problem for a food distributor in Pretoria, South Africa, and as such the problem specifications match the operations of the company in question. Two reduced-size problems were created, a small and medium size version, in order to compare solutions with their exact solution counterparts found using a standard ILP. A large-size problem instance provided by the company was solved. Problems addressed represented a range of customer daily deliveries from 10 to 128 from a single depot. The presence of a single depot means that this type of problem differs significantly from the PDP-type problems and the airline taxi problem.
Another type of problem which shares commonality in terms of having a single depot with the MVCVRP is the VRPTW. Standard, benchmark instances for this type of problem have been artificially generated and are available on the Internet, and such a problem, provided by Solomon [147], was obtained. For this problem, the agent routing method developed for the airline taxi problem was applied. To achieve this, the problem is modelled using a time-space network. A best known solution available on the Internet was used for comparison purposes.
5 The Sefofane Air Scheduling Problem

Description

5.1 Sefofane Air Operations

Sefofane Air is the largest airline operating in the Okavango Swamps, Botswana, and that is where the bulk of their operations occur. Their primary target market is tourist groups. Their operations also include airline services in Namibia, Zimbabwe, Zambia and South Africa.

Tourists book to stay at any of more than 100 tourist safari camps in the Okavango and surrounding area. When booking, they will also book flights between locations in the area.

Tourists may enter Botswana by road or on flights from South Africa or other international locations. Typically a road trip or an international flight will bring tourists to Victoria Falls in Zimbabwe, Livingstone in Zambia or, most likely, Maun in Botswana, which is also where the Sefofane Air head office is located. This can be considered the hub of their operations, since more of their flights arrive and depart from there than anywhere else. Air Botswana international flights land and depart from Maun.

From Maun or other point of access, Sefofane Air transports the tourist groups to their prebooked safari camps. Most safari camps have an airstrip, and most are an un tarred (dust) landing strip. Main centres such as Victoria Falls have tarred landing strips or airports. Often, tourists will stay in a series of different safari camps, in which case Sefofane Air will transport them between these camps, as required.

Sefofane Air uses relatively small aircraft designed to operate to and from such locations. These include the Cessna C206, a 5-seater aircraft (i.e. 5 passenger seats)
often described as the “workhorse of the bush” (Figure 5.1), and the Cessna C208 Grand Caravan, a larger, 12-seater aircraft (Figure 5.2). Both aircraft can be flown by a single pilot. However, if carrying passengers and luggage, the C208 needs a loadmaster, which is an extra crew member required to balance the aircraft load before takeoff, particularly if the aircraft is full or heavily loaded.

Currently, Sefofane Air employs two experienced schedulers who make up the flight schedule and route the aircraft a few days in advance of the day in question. The schedulers are considered to produce good, low cost schedules. However, since the job they do is dependent on their experience, there is a risk to the airline that, if they should not be available, at best schedule costs would increase, and at worst the airline would not be able to operate effectively.

Because schedules are assembled in advance for a known booking list, the problem can be classified as a static DAFP [122].

5.2 Problem Specifications

Sefofane Air’s fleet and aircraft specifications, as used in the Okavango Swamps, are shown in Table 5.1.
Table 5.1: Fleet Characteristics [150], [151]

<table>
<thead>
<tr>
<th>Aircraft Type</th>
<th>Number Available</th>
<th>Passenger Seat Capacity</th>
<th>Cruising Speed (km/hr)</th>
<th>Operating Cost (R/hr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cessna 206</td>
<td>9</td>
<td>5</td>
<td>210</td>
<td>1385</td>
</tr>
<tr>
<td>Cessna 208</td>
<td>5</td>
<td>11</td>
<td>260</td>
<td>3910</td>
</tr>
</tbody>
</table>

The costs in Table 5.1 are given in South African currency, Rands. The exchange rate at the time of this work was approximately 8.50 Rands (R8.50) to the US Dollar. The C208 cost given includes the loadmaster (R255/hr). In the case of the C208, the seating excludes the 2 crew members (pilot and loadmaster). The operating cost includes crew, maintenance costs and fuel.

Sefofane Air supplied a list of bookings for a particularly busy day (18 June 2008) and this list included 21 safari camps. The booking list supplied is included in Appendix A, Table A.1. Note the use of the term “pax” which refers to the number of passengers in the booking group, or the number of passengers being transported on a flight.

A map of the camps which were included in the booking list on that day, and the area of operations is shown in Figure 5.3.
In this research, each safari camp (or location) was arbitrarily assigned a number from 1 to 21 for the purposes of modelling, and these numbers are shown on the map. The safari camp names and numbers are listed in Appendix A, Table A.4. The distances between safari camps is included in Appendix A, Table A.2.
Figure 5.3: Map of the Okavango Delta Area [152]
Sefofane Air supplied the manual schedule that was compiled and actually used for the day in question, and this is included in Appendix B. This schedule shows that the schedulers use a turnaround time of 10 minutes.

Sefofane Air have 3 different types of passengers:

1. Tourists - Usually travelling in groups of mostly 2 but also more.
2. Sefofane and safari camp staff - Usually travelling as a single person.
3. Boxes - A box consists of provisions for the camps and takes up one seat on an aircraft.

It is desirable not to have tourists sitting next to boxes, hence in the schedule shown, the boxes were grouped either in 5s or in 12s such that they would alone take up their own aircraft.

For the C208, no loadmaster is required for “box only” flights. Examination of the manual schedule (Appendix B) reveals an instance where a C208 had no loadmaster, and had to fly to a certain destination to collect one. Also, all the “box only” trips were specifically scheduled without a loadmaster. Presumably this saves costs and allows one extra box to be transported.

Crew may overnight at any safari camp, since the aircraft is merely parked on the airstrip and the crew accommodated in available accommodation. However, it may be advantageous to ensure they are at some specific destination, for example, to start the next day’s flights. In fact, examining the manual schedule, it is apparent that the schedulers did schedule aircraft to fly empty to overnight at specific destinations.

Note that both aircraft have relatively long ranges so refueling during a day’s flying is unlikely to be necessary, and this was actually the case in this work. If necessary, fuel is available at the main hub (Maun) and various other stops such as Victoria Falls and Livingstone.

Pilots and loadmasters are paid per hour of flying time. The loadmaster’s cost is relatively small compared to the C208 operation cost (R255 vs R3610). This, coupled with the fact that the flights requiring a loadmaster far outnumber the flights which do not, suggest the airline should consider always including a loadmaster in every flight. This is because the operational complications of not having a loadmaster on certain flights is likely to outweigh the small cost benefit.

Examination of the manual schedule revealed a special request. This involved a
large tourist group which wanted to be flown in their own aircraft and without any
other passengers. Since the group consisted of 13 people, they requested two C208s
for this purpose.
6 Multi-Commodity Network Flow ILP with Time Discretisations

6.1 Description

As suggested by Cordeau [5], Yan and Tseng [128] and Espinoza [7], the DAFP can be formulated as an ILP with the use of time steps or discretisations (TDs). In this case, the time-space diagram is discretised in the time dimension using 10 minute time steps. A node is placed at each destination at each time step, and all possible flight arcs are included, as shown in Figure 3.7. Nodes are also placed at each passenger origin at the earliest departure time (EDT) and destination at the latest arrival time (LAT). These nodes are used to ensure ground arcs can be included that will allow demand to be met as required by the time windows. Multiple networks are superimposed and solved in the formulation, one network for each fleet type and one for each passenger group.

The formulation is as follows:

Define the following decision variables:

- Set of aircraft flight variables $X$, consisting of elements $x_{ijf}$, each corresponding to a possible flight leg. Each such leg is associated with a fleet type $f$ in the set of fleet types $F$, a starting node $i_{ut}$ in the set of nodes $N$ with an origin $u$ from the set of locations $C$, and a departure time $t$, and an ending node $j_{vt}$ in the set of nodes $N$ with a destination $v \in C$ and an arrival time $t$. These variables are referred to as “flight legs” and are integer.

- Set of fleet ground variables $S$, consisting of elements $s_{ijf}$, each corresponding to a fleet type $f$ in the set of fleet types $F$, a start node $i_{ut}$ and end node $j_{vt}$, both in set of nodes $N$. A ground variable has an associated location $u$
and represents the number of aircraft of a particular fleet type on the ground. These variables are referred to as “ground arcs” and are integer.

- Set of passenger flight variables $Y$, consisting of elements $y_{xg}$, each corresponding to a possible flight leg $x$ and a passenger group $g$ in the set of passenger groups $G$. These variables are binary and indicate whether the associated passenger group is undertaking that flight leg (i.e. on-board the aircraft for that flight leg). They are referred to as “passenger flight legs”.

- Set of passenger ground variables $T$, consisting of elements $t_{ijg}$, each corresponding to a passenger group $g$ in the set of passenger groups $G$. Each passenger ground arc connects two nodes $i_{ut}$ and $j_{ut}$ at a location $u \in C$ and indicates whether the associated passenger group is on the ground. They are referred to as “passenger ground arcs”.

The objective is to minimize costs:

$$\text{minimize } \sum_{x \in X} C_x x_{ijf},$$

(6.1)

where $C_x$ is the cost associated with the flight variable $x$, i.e. the cost of executing the flight leg. It is calculated as the aircraft type hourly cost in Rands times the flight distance in kilometers divided by the aircraft type flight speed in km/hr.

Constraints must ensure conservation of fleet flow and passenger group flow at every node. For fleets,

$$(\text{incoming aircraft}) = (\text{outgoing aircraft}).$$

Therefore

$$\left( \sum_{i \in N} x_{inf} + s_{(n-1)n_f} \right) - \left( \sum_{j \in N} x_{nfj} + s_{n(n+1)f} \right) = 0, \quad \forall n \in N, \forall f \in F,$$

(6.2)

where nodes $n$ are numbered sequentially at each location. Therefore $s_{(n-1)n_f}$ refers to the ground arc leading from the preceding node $(n - 1)$ to the subsequent node $n$, and, likewise, $s_{n(n+1)f}$ refers to the ground arc leading from the preceding node $n$ to the subsequent node $n + 1$. 
For passengers,

\[
\left( \sum_{i \in N} y_{x_{infg}} + t_{(n-1)ng} \right) - \left( \sum_{j \in N} y_{x_{njfg}} + t_{n(n+1)g} \right) = 0, \quad \forall n \in N, \forall g \in G. \quad (6.3)
\]

The passenger networks and the fleet networks are bound by the aircraft capacity constraints as follows:

\[
\sum_{y \in Y} y_{xg} \text{GrpSize}_g \leq x \text{Cap}_x, \quad \forall x \in X, \quad (6.4)
\]

where \(\text{GrpSize}_g\) is the number of people making up group \(g\). \(\text{Cap}_x\) is the passenger capacity of the aircraft type associated with \(x\).

Additional constraints are needed to ensure the number of aircraft of each fleet type are correct at each location at day start and day end, and that the number available is not exceeded. Therefore at the starting node for each location \(u\) in set of all locations \(C\), the ground arc \(s\) for each fleet \(f\) must be set equal to the number of aircraft of that fleet type positioned there at day start:

\[
s_{n,f} = v_{uf}, \quad \forall f \in F, \forall u \in C, \quad (6.5)
\]

where \(s_{n,f}\) is the ground arc for fleet type \(f\) going into the first node (i.e. earliest time) of location \(u\). \(v_{uf}\) is the number of aircraft of fleet type \(f\) at location \(u\) at the start of the day.

Note that this constraint must be used for all locations where there are no aircraft of some fleet type, in which case \(v_{uf} = 0\). This is to ensure the correct number of aircraft are used in the model.

In the case where the aircraft need to be at certain locations at day end, the ground arc \(s\) for each fleet type \(f\) leaving the final node at each location \(u\) must be set to the required number of aircraft of the appropriate fleet type:

\[
s_{nuf} = v_{uf}, \quad \forall f \in F, \forall u \in C, \quad (6.6)
\]

where \(s_{nuf}\) is the ground arc for fleet type \(f\) going out of the last node (i.e. latest
time) of location $u$. $v_{uf}$ is the number of aircraft of fleet type $f$ at location $u$ at the end of the day. This constraint could be used only for locations where aircraft are required at the end of the day if necessary, and not all locations.

A similar constraint is required to ensure the correct number of each passenger group in the problem. This can be done by setting the passenger ground arc into the earliest node at each location (i.e., at day start) to either one if that location matches the group origin, or zero otherwise. This is achieved as follows:

$$t_{ng} = b_{ug}, \quad \forall g \in G, \forall u \in C,$$

(6.7)

where $t_{ng}$ is the passenger group ground arc with end node corresponding to the first node at location $u$ and $b_{ug}$ is either 1 or zero, depending on whether $u$ is the origin for $g$.

Constraints must be included to ensure passenger groups are at the correct locations at the group EDT and LAT of each day to ensure that demand is met and time windows enforced. For the EDT, the passenger ground arcs $t$ for every passenger group $g$ into the correct node of each location $u$ must be set to a value of 1:

$$t_{ng} = 1, \quad \forall g \in G,$$

(6.8)

where $t_{ng}$ is the passenger group ground arc with end node corresponding to EDT and origin location for passenger group $g$.

Similarly, the passenger ground arcs out of the LAT nodes at each location must be set to 1 for each passenger group:

$$t_{mg} = 1, \quad \forall g \in G,$$

(6.9)

where $t_{mg}$ is the passenger group ground arc with start node corresponding to LAT and destination location for passenger group $g$.

Note that this formulation allows passenger groups to swap aircraft during multi-leg journeys. Preventing this effect would require additional variables to be introduced into the problem, adding complexity and slowing computational times. In practice, it was found that in addition to aircraft swapping by passenger groups, certain passenger groups also had to endure long waiting times during journeys. A constraint
was introduced to constrain this intra-journey waiting time. This constraint was formulated to limit the amount of ground arc time endured when the group was not at either their origin or their destination, as follows:

\[
\sum_{t \in \{T: g_{od}^c\}} t \leq 4, \quad \forall g \in G. \tag{6.10}
\]

In equation (6.10), set \( \{T: g_{od}^c\} \) is a subset of passenger ground arcs \( T \) which includes only ground arcs which are not situated at either the passenger group \( g \) origin or destination. This will cause every group to only need to endure ground waiting times of a maximum of 10 minutes at each stop, plus \( 4 \times 10 = 40 \) minutes, i.e. 50 minutes (10 minutes being the size of the time discretisations, TDs). Replacing the 4 by a 0 causes the passenger groups to only have ground waiting times of 10 minutes per stop during a journey.

In practice it was found that passenger leg and ground arc variables were being used unnecessarily since they do not have a cost associated with them. Therefore a nominal cost was applied to all the variables except the flight leg variables to prevent them being used in a solution. A value of 20 was found to be sufficient for this purpose.

The number of variables produced by this formulation for the full-sized problem under consideration in this work was found to be 7.8 million. This is too many to create and manipulate on a personal computer. Therefore, problem size reduction heuristics are described in the next chapter.
7 Heuristics for Problem Size Reduction

7.1 Motivation

The exact formulation MCNFP ILP with time discretisations (TDs) is difficult to solve largely because of the number of passenger groups in the problem. This is because each commodity in the problem requires a separate network, and each passenger group and each fleet effectively represents a different commodity. Since for this problem there are 2 fleet types and 158 passenger groups, the formulation results in 7.8 million variables, which is too many to allow a solution to be obtained. The main problem is the number of passenger groups. Therefore, heuristics are aimed specifically at reducing:

1. The size of the passenger networks,
2. The number of passenger networks required.

In this work, two heuristics were devised to reduce the number of passenger network variables. The first, a passenger group aggregation heuristic, effectively reduced the number of passenger groups in the problem by half. Thus the number of passenger networks required was reduced proportionately. The second, a heuristic based on geographic considerations, reduced the passenger network sizes, such that, in the large instance (full schedule) the remaining problem variables were reduced to a half of their original number. These two heuristics were used together for the standard MCNF described in Chapter 6 and also for the method described in Chapter 8. Only the passenger aggregation heuristic was used in the method described in Chapter 9.
7.2 Group Aggregation Heuristic

7.2.1 Description

Since many bookings involve only one person (staff), the number of items to be scheduled could effectively be reduced using a heuristic to combine smaller groups into larger groups. For this problem, this was the case. This heuristic was devised as follows:

1. Begin with the first booking in the list.

2. Search from the second booking onwards for any other booking which satisfies the following:
   - Has the same origin and destination as the first booking
   - Has an earliest departure time (start of time window, EDT) and latest arrival time (end of time window, LAT) closely matching that of the first booking
   - When the number of passengers in the booking is added to that of the first booking, the aircraft capacity for the largest aircraft (C208) is not exceeded.

3. If found, that booking is combined with the first booking and the EDT and LAT for the combined booking is set to the later of the two in the case of EDT, and the earlier in the case of the LAT.

4. If the plane (the largest) has additional space available, the search is continued to the end of the booking list.

5. Move to the next booking in the list and return to step 2. Repeat until every item in the booking list has been considered for combination with every other item.

Note that item 2 specifies that two groups can be aggregated if their EDTs and LATs closely match. The time within which two group times could be considered “close” can be varied. A larger tolerance would cause more groups to be aggregated and a smaller resultant aggregated schedule. The value used in this work is 15 minutes and was found to produce good results.

Note that groups were simply assembled into as large groups as possible to fit in the largest available aircraft (i.e. maximum group size of 11).
7.2.2 Aggregation Validation

This heuristic was validated by randomly shuffling the full set of booking data, then observing whether different aggregated groups resulted when the heuristic was applied. The data was thus shuffled and the heuristic applied five times, and the result was identical each time, that is, the same groups were aggregated together and the same number of combined groups resulted. It was concluded that, in this instance, the grouping mechanism would always produce repeatable resultant bookings.

The result for the large schedule (158 bookings/passenger groups, 21 locations) was 79 combined groups. These groups consisted of 3 combined groups that had been assembled from 11 of the original groups, 1 combined group from 10 original groups, 1 combined group from 6 original groups, 1 combined group from 5 original groups, 2 combined groups from 4 original groups, 5 combined groups from 3 original groups, and 15 combined groups from 2 original groups. The balance of the groups could not be combined. The aggregated groups are shown in Appendix C. The effective number of bookings was reduced by half for the full-sized schedule. The application of this heuristic to the small-sized problem (40 bookings, 7 locations) resulted in 26 aggregated passenger groups.

It should be noted that, although this heuristic was effective in this instance, some problem parameters could differ in other instances and cause the heuristic to be less effective. For example, for this work the largest aircraft size was 11 passengers in comparison with 5 passengers for the smaller aircraft type. There were 5 of the larger aircraft type available. In a similar case, but where there is only one of the larger aircraft available, the problem may become infeasible due to this heuristic, or give much lower quality results. This is because there could be too many aggregated groups larger than the smaller aircraft capacity of 5 passengers for the single larger aircraft to transport in a day. In such a case, the heuristic would be executed to only allow a limited number of large groups for the large aircraft, and the rest of the groups of sizes up to a maximum of the size of the smaller aircraft.

7.3 Geographic Heuristic

7.3.1 Description

A heuristic based on geographical considerations was devised which limited the generation of passenger legs in the MCNFP formulations. The heuristic is based
on the fact that a passenger group (booking) is most likely only going to travel within a certain geographical region based on its origin and destination, therefore any passenger leg or ground arc not within this geographical region can be left out of the problem. This makes practical sense since a passenger group is likely to be dissatisfied if flown in a direction away from or beyond their destination. This heuristic is advantageous for this particular problem since it specifically reduces the number of passenger legs in the problem.

A geographical area is defined based on the origin and the destination of each passenger group in the booking list. In no case would it be advantageous to transport a passenger group in a direction opposite to their destination. Also, passengers would not need to be flown very far to either side of the line joining the origin and the destination.

Therefore, the geographical area is defined using an angle $\alpha$ and a distance $d$ as shown in Figure 7.1. Only destinations (shown as round dots) within the polygon ABCDEF are included in the passenger group geographic network under consideration. By varying $\alpha$ and $d$ the geographical area can be varied for any particular problem, allowing more or less variables in the problem. The effect of the heuristic on the solution quality can be evaluated by adjusting the geographic control parameters $(\alpha, d)$ for the problem and judging when the solution quality begins to be adversely affected.

Many passenger origin-destination (O-D) pairs are close together, and in these cases, many variables would be removed from the problem. The heuristic will be much less effective in variable generation reduction for those groups flying longer distances.

![Figure 7.1: Geographic Heuristic](image)

Also of note is the fact that this concept can be applied to the time dimension. To do
this, all passenger leg and passenger ground arc variables falling outside a passenger group EDT and LAT should be excluded from the problem. This results in a set of valid inequalities for the problem.

7.3.2 Evaluation of Geographic Heuristic Effect

The geographic heuristic was applied to the small-size schedule and the model subsequently solved using the MCNF described in Chapter 6, in order to evaluate the effect of the heuristic on the solution. The version of the problem with constrained end-of-day aircraft locations was used (equations (6.1) to (6.9)). Note that the aggregation heuristic was applied first to these problems. The results are shown in Table 7.1. Note that objective value \((Z)\) result for the problem without the geographic heuristic is 40388.

Note that “gap” refers to the integrality gap (ILP gap) provided by typical ILP solvers during the branch-and-bound solving process, and before the optimal solution is found. The solver starts from a relaxed solution and progressively adds constraints to encourage variables to resolve to integers. After some time an integer feasible solution is found. At any point in time, the current, best integer solution is retained while the search for better integer solutions proceeds. This solution provides an upper bound (in a minimisation problem). The search process involves finding many solutions which may have a certain number of variables integer, but not all. The best solution obtained from these “unresolved branches” represents a lower bound, or “best bound” to the problem. The gap refers to the percentage gap between these upper and lower bounds. When this gap is zero, optimality has been achieved [153].

From Table 7.1, it is evident that the solution begins to degrade for \(\alpha\) values below \(45^\circ\) and \(d\) below 50\(km\). Therefore those values were used in subsequent problems.

In Table 7.1 the solution times are often unpredictable in relation to the number

<table>
<thead>
<tr>
<th>(\alpha) (degrees)</th>
<th>d (km)</th>
<th>(\text{min } Z) (Rands)</th>
<th>time (s)</th>
<th>\text{No. of Variables}</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>100</td>
<td>40388</td>
<td>453</td>
<td>44852</td>
</tr>
<tr>
<td>45</td>
<td>50</td>
<td>40388</td>
<td>52</td>
<td>34153</td>
</tr>
<tr>
<td>45</td>
<td>40</td>
<td>42564 (0.75% gap)</td>
<td>3695</td>
<td>32351</td>
</tr>
<tr>
<td>45</td>
<td>30</td>
<td>43640 (2.84% gap)</td>
<td>380</td>
<td>32351</td>
</tr>
<tr>
<td>40</td>
<td>50</td>
<td>41405</td>
<td>307</td>
<td>32555</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
<td>44334</td>
<td>61</td>
<td>31113</td>
</tr>
<tr>
<td>35</td>
<td>50</td>
<td>44334</td>
<td>59</td>
<td>31113</td>
</tr>
<tr>
<td>35</td>
<td>40</td>
<td>44334</td>
<td>307</td>
<td>31113</td>
</tr>
</tbody>
</table>

Table 7.1: Cost and Computation Times for Different Geographic Heuristic Parameters, Small Schedule
of variables in the problem. Most of the problems were solved to optimality in a reasonable time. However, the problem for \( d=40 \) km and \( \alpha=45^\circ \) took much longer to solve, such that the solver had to be stopped prior to optimality. For this problem, it was observed that the ILP gap was less than 3% after 6 minutes. Therefore the solver could have been stopped at that point and a good solution obtained. Accordingly, the solution for \( d=30 \) km and \( \alpha=45^\circ \), was stopped after about 6 minutes. The reason for these slow solution times is not apparent, but since acceptable results were achieved, this was not further investigated.

The geographic heuristic was applied to the full size schedule to evaluate the effect of the heuristic on the number of variables in the problem. Note that the effect of the heuristic on the solution quality was not evaluated. The results are shown in Table 7.2. Without the heuristic, the number of variables is approximately 3.84 million. Therefore the problem size has been significantly reduced. However, for an \( \alpha \) value of 45° and a \( d \) of 50 km, the number of variables is still a substantial 235428.

Table 7.2: Effect of Geographic Heuristic on Number of Variables for Large Schedule

<table>
<thead>
<tr>
<th>( \alpha ) (degrees)</th>
<th>( d ) (km)</th>
<th>No. of Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>100</td>
<td>523615</td>
</tr>
<tr>
<td>45</td>
<td>50</td>
<td>235428</td>
</tr>
<tr>
<td>45</td>
<td>40</td>
<td>233474</td>
</tr>
<tr>
<td>45</td>
<td>30</td>
<td>220567</td>
</tr>
<tr>
<td>35</td>
<td>50</td>
<td>179208</td>
</tr>
<tr>
<td>35</td>
<td>40</td>
<td>177718</td>
</tr>
<tr>
<td>35</td>
<td>30</td>
<td>176483</td>
</tr>
</tbody>
</table>

7.4 MCNFP Solutions

7.4.1 Small Schedule

Solutions were obtained for the MCNFP described in Chapter 6 for the small and large schedules. Both the aggregation and geographic heuristics were applied.

For the case where the end-of-day locations of aircraft and passenger group ground arc waiting times (or intra-journey waiting times) are not constrained (equations (6.1) to (6.5), (6.7) to (6.9)), and using geographic heuristic values \( d=50 \) km and \( \alpha=45^\circ \), a best cost of R37906 was achieved in 1882s and with 34153 variables.

The small-size schedule was solved for the case where the end-of-day locations of aircraft are constrained (equations (6.1) to (6.9)) and the results are shown in Table 7.1. The resultant schedule for the chosen geographic heuristic values \( d=50 \) km and
\( \alpha=45^\circ \) is shown in Table 7.3. Note that the passenger groups referred to in the table are aggregated groups, hence there are 26 groups listed.

The passenger groups which experience multiple flight legs and those which are required to swap aircraft during their journey are shown in Table 7.4. The schedule cost was R40388.

Note that the groups listed in column 1 of Table 7.4 are the only groups that experienced a journey which consisted of more than one flight leg. For example, row 1 shows that passenger group 7 consists of two passengers, and their journey from origin to destination involved 3 aircraft flights. The last column shows that the aircraft on which they were placed was a C206, and they started at location 2, went to location 6, then location 4, then ended their journey at location 3.

Four passenger groups (15, 14, 21 and 19) needed to change aircraft and one group changed aircraft twice and endured a total of 390 minutes (6.5 hours) of ground waiting time en-route. Another group had to endure a wait of 580 minutes (9.7 hours) between flight legs.

The amount of intra-journey ground waiting time on multi-leg journeys was then limited to turn around time only (i.e. 10 minutes per stop) by adding the constraint shown in equation (6.10) to the problem. The result (using \( d=50\text{km}, \alpha=45^\circ \)) was a schedule cost of R41338, using 34135 variables and 1462s solution time. Allowing 40 minutes of en-route ground waiting time (excluding 10 minutes turn around time per stop) produced a schedule cost of R41300. With this constraint restricting ground waiting times to the minimum (turn around times only), 5 passenger groups needed to change aircraft en-route (Table 7.5). One group needed to change aircraft twice.

A summary of these results is included in Table 7.6. It was found that the problems with both the end-of-day aircraft position constraints and the ground time limiting constraints took excessively long and therefore impractical times to solve, hence those results are not included in the table.

The effect of the aggregation heuristic was evaluated by solving the problem without using this heuristic, and limiting the ground arc waiting time for passenger groups to the minimum (using \( d=50\text{km}, \alpha=45^\circ \), equations (6.1) to (6.10)). The result was a schedule with a cost of R41907 with a 2.5% gap after 69329s (19.3hrs) of processing time, and using 77089 variables. This is similar to R41338 obtained for the aggregated instance (taking into account the 2.5% ILP gap).

When the non-aggregated model was executed with unlimited passenger ground
time, a schedule cost of R40388 resulted (0.7% ILP gap) after 56324s (15.6 hours) processing using 77089 variables. This is identical to the R40388 obtained for the aggregated instance. The large-size schedule could not be solved in non-aggregated form due to the number of variables involved.

Table 7.3: Small Schedule Solution

<table>
<thead>
<tr>
<th>Fleet Type</th>
<th>Origin Location</th>
<th>Departure Time*</th>
<th>Destination Location</th>
<th>Arrival Time*</th>
<th>Passenger Groups on Board*</th>
</tr>
</thead>
<tbody>
<tr>
<td>C206</td>
<td>3</td>
<td>370</td>
<td>4</td>
<td>420</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>420</td>
<td>7</td>
<td>470</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>530</td>
<td>2</td>
<td>570</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>610</td>
<td>6</td>
<td>640</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>670</td>
<td>4</td>
<td>700</td>
<td>7, 14, 15</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>700</td>
<td>3</td>
<td>750</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>760</td>
<td>5</td>
<td>820</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>820</td>
<td>6</td>
<td>860</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>860</td>
<td>2</td>
<td>890</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>950</td>
<td>7</td>
<td>990</td>
<td>0</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>380</td>
<td>6</td>
<td>410</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>480</td>
<td>3</td>
<td>560</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>610</td>
<td>1</td>
<td>650</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>660</td>
<td>3</td>
<td>700</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>810</td>
<td>4</td>
<td>860</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>860</td>
<td>3</td>
<td>910</td>
<td>2, 14</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>460</td>
<td>2</td>
<td>500</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>500</td>
<td>7</td>
<td>540</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>870</td>
<td>2</td>
<td>910</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>950</td>
<td>7</td>
<td>990</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>950</td>
<td>4</td>
<td>1000</td>
<td>9, 18</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1000</td>
<td>7</td>
<td>1050</td>
<td>16, 19</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>370</td>
<td>6</td>
<td>390</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>420</td>
<td>3</td>
<td>480</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>720</td>
<td>4</td>
<td>760</td>
<td>1, 21</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>830</td>
<td>6</td>
<td>860</td>
<td>8, 17, 21</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>860</td>
<td>7</td>
<td>880</td>
<td>8, 13, 17</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>420</td>
<td>6</td>
<td>440</td>
<td>12, 14</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>440</td>
<td>3</td>
<td>500</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>720</td>
<td>5</td>
<td>770</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>830</td>
<td>6</td>
<td>860</td>
<td>24</td>
</tr>
</tbody>
</table>

* Times are given in minutes in the day, i.e. 370 = 6h10, 420 = 7h00  ** 0 indicates empty aircraft flight

Table 7.4: Multi-Leg Trips - Unlimited Ground Waiting Time, Small Schedule

<table>
<thead>
<tr>
<th>Passenger Groups</th>
<th>No. of Passengers</th>
<th>No. of Legs</th>
<th>Fleet Type and Journey Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>2</td>
<td>3</td>
<td>C206: 2 - 6 - 4 - 3</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>2</td>
<td>C208: 7 - 6, wait: 480min, C206: 6 - 4</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>3</td>
<td>C208: 7 - 6, wait: 230min, C206: 6 - 4, wait: 160min, C206: 4 - 3</td>
</tr>
<tr>
<td>21</td>
<td>3</td>
<td>4</td>
<td>C206: 1 - 3, wait: 20min, C208: 3 - 4 - 6, C206: 6 - 2</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>2</td>
<td>C206: 3 - 4, wait: 580min, C206: 4 - 7</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>2</td>
<td>C208: 4 - 6 - 7</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>2</td>
<td>C208: 4 - 6 - 7</td>
</tr>
</tbody>
</table>
### Table 7.5: Multi-Leg Trips - Minimum Ground Waiting Time, Small Schedule

<table>
<thead>
<tr>
<th>Passenger Groups</th>
<th>No. of Passengers</th>
<th>No. of Legs</th>
<th>Fleet Type and Journey Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>1</td>
<td>3</td>
<td>C206:7 - 6, C206: 6 - 4 - 3</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>2</td>
<td>C206: 7 - 6, C206: 6 - 4</td>
</tr>
<tr>
<td>21</td>
<td>3</td>
<td>4</td>
<td>C206: 1 - 3 C208: 3 - 4 - 6, C206: 6 - 2</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>2</td>
<td>C208: 4 - 6, C206: 6 - 7</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>2</td>
<td>C208: 4 - 6, C206: 6 - 7</td>
</tr>
</tbody>
</table>

### Table 7.6: Schedule Cost Results Summary for Small Schedule

<table>
<thead>
<tr>
<th>Waiting Time Allowed (min)</th>
<th>With Aircraft End-of-Day Location Constraints (Rands)</th>
<th>Without Aircraft End-of-Day Location Constraints (Rands)</th>
</tr>
</thead>
<tbody>
<tr>
<td>unlimited</td>
<td>40388</td>
<td>37906</td>
</tr>
<tr>
<td>40</td>
<td>-</td>
<td>41300</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>41338</td>
</tr>
</tbody>
</table>

* Excluding TATs

#### 7.4.2 Full Schedule

Both the aggregation and geographic heuristic were applied to the full-sized schedule. For the version of the problem with unconstrained end-of-day aircraft locations (equations (6.1) to (6.5), (6.7) to (6.9)), a schedule cost of R79087 (7.02% ILP gap) was obtained, after 104160s (28.9 hours) processing time and using 235428 variables.

For the problem with constrained end-of-day aircraft locations (equations (6.1) to (6.9)), the resultant schedule cost was R85927 (15.4% ILP gap) after 22046s (6.1 hours) processing time. Passenger intra-journey ground time was not constrained. All passenger trips that required swapping of aircraft are shown in Table 7.7. The most en-route flight legs is 3 and a waiting time of more than 10 hours was required for one group. Seven passenger groups needed to swap aircraft out of a total of 79 groups, or 9% of booked passengers.

In order to further reduce the problem size and for the case with constrained end-of-day aircraft locations (equations (6.1) to (6.9)), the geographic heuristic parameters were set to $d=40km$ and $\alpha=35^\circ$. The passenger intra-journey waiting time on the ground was limited to 40 minutes (excluding TATs) and a schedule with cost of R88108 (16.8% ILP gap) resulted after 52746s processing time and using 177718 variables. When the allowed passenger ground time was reduced to the minimum, the resultant schedule cost was R90819 (19.4% ILP gap) after 12930s of processing time and using 177718 variables.

A summary of results is included in Table 7.8.
Table 7.7: Multiple Leg Trips for Large Schedule

<table>
<thead>
<tr>
<th>Passenger Groups</th>
<th>No. of Passengers</th>
<th>No. of Legs</th>
<th>Fleet Type and Journey Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>30, 31, 32</td>
<td>3</td>
<td>4</td>
<td>C206: 1 - 3, C208: 3 - 4 - 6 C206: 6 - 2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3</td>
<td>C206: 2 - 6 - 4 - 3</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>3</td>
<td>C208: 7 - 6 C206: 6 - 4 - 3</td>
</tr>
<tr>
<td>25, 27</td>
<td>2</td>
<td>2</td>
<td>C206: 3 - 4 wait: 610 minutes C206: 4 - 7</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>2</td>
<td>C208: 7 - 6 C206: 6 - 4</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>C208: 4 - 6 - 7</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>2</td>
<td>C208: 4 - 6 - 7</td>
</tr>
</tbody>
</table>

Table 7.8: Schedule Cost Results Summary for Large Schedule

<table>
<thead>
<tr>
<th>Heuristic Values</th>
<th>Waiting Time Allowed (min)</th>
<th>With Aircraft End-of-Day Location Constraints (Rands)</th>
<th>Without Aircraft End-of-Day Location Constraints (Rands)</th>
</tr>
</thead>
<tbody>
<tr>
<td>d=50, α=45</td>
<td>unlimited</td>
<td>85927 (15.4% gap)</td>
<td>79087 (7.02% gap)</td>
</tr>
<tr>
<td>d=40, α=35</td>
<td>40</td>
<td>88108 (16.8% gap)</td>
<td>-</td>
</tr>
<tr>
<td>d=40, α=35</td>
<td>0</td>
<td>90819 (19.4% gap)</td>
<td>-</td>
</tr>
</tbody>
</table>

7.5 Discussion

From Table 7.1, the problem solution begins to degrade for values of $d$ between 20 and 50, and values of $\alpha$ of between 25° and 45°. Therefore, suitable values are $d=50km$ and $\alpha=45^o$ and these values are used in further computations. Notable is the fact that a substantially reduced problem size (31%) can be achieved with tighter geographic heuristic parameters with only a 9.8% reduction in optimality when comparing the worst case calculated in Table 7.1 (schedule cost R44334) with the best case (R40388).

The full schedule models produced numbers of variables exceeding 150 000 and as such could not be solved to optimality in a reasonable time. It was found that the ILP solver slowed down substantially as the ILP gap reduced, requiring many hours of processing time to reduce the ILP gap to between 12% and 20%.

The full schedule solution with constrained end-of-day locations could be found to a 15.4% gap after just over 6 hours, and produced a cost of R85927.

Although good solutions can be achieved using this method, the number of variables generated is still too many for practical purposes for the full-size schedule, unless specialised solution methods and/or processing hardware are developed. However, the method will be practical for moderate size instances (up to 80 requests) of the airline taxi problem.

It should be noted that this formulation (equations (6.1) to (6.9)) allows swapping of groups between different aircraft. Adding constraints to prevent this would require
additional variables and constraints, increasing the problem size significantly and increasing the solution computation time accordingly. Table 7.4 shows the groups that experienced aircraft swapping and ground waits en-route for the small schedule, including some unreasonably long waiting times. Therefore the additional constraint (6.10) was added to limit passenger waiting times on the ground, and this raised the cost of the schedules accordingly. The small schedule went from costing R37906 with unlimited passenger ground waiting time en-route, to R41338 with passenger ground waiting time limited to the minimum. The passenger ground time limiting constraints were observed to have a detrimental effect on processing times.

The constraints on passenger waiting times caused the schedule cost to increase. This means the airline can save further costs by allowing increased sized time windows. This might be possible with boxes and possibly with staff as well, but is unlikely to be acceptable for many tourist groups.

For the small schedule, limiting the passenger group ground time to the minimum reduced the number of groups that experienced multi-leg trips from 7 (27%) to 5 (19%) out of 26 total groups (Table 7.5). However, for unlimited passenger ground time, 4 groups swapped aircraft (15%) vs 5 groups (19%) for the case where ground time was limited to the minimum. For the full-sized schedule, only 11% of passengers needed to swap aircraft.

There are various options open to the airline in terms of quality of service and using this modelling technique. The easiest way to control quality of service provided will be to limit the EDT and LAT (time windows) of all passenger groups which need a higher quality of service than others, e.g. tourist groups vs employees and boxes. Providing tight time windows will reduce multi-leg journeys and aircraft swapping. A further option is to provide passengers with a two-tier pricing structure, where the lower tier has flexible pick-up and delivery times and may be subject to long ground waiting times. This would remove the need for some of the passenger ground time limiting constraints which slow the processing time down.

The small schedule was used to evaluate the effect of the aggregation heuristic. The version of the problem without aggregated groups and with zero passenger ground waiting time produced a cost of R41907 but could only be solved with a 2.5% ILP gap in a reasonable processing time. This compares favourably with the aggregated version of the problem, which produced a schedule cost of R41338, and suggests the aggregation heuristic has a marginal, if any, effect on the quality of the solutions.

Even though the number of variables in the non-aggregated model is not excessive
(77089), the solution times are long. This is due to there being substantially fewer solutions in close proximity in the feasible region in the case of the aggregated model, therefore the branch-and-bound process is far more efficient in pruning.

Although the number of variables was high and processing time long for the full schedule using the MCNF with TD method, good solutions can be obtained. The process speed can be increased by making the TDs longer, by shortening the day, or by tightening the geographic heuristic. The method would be appropriate for smaller instances of this problem.

In terms of portability of the heuristics, both could be applied to certain PDP type VRPs. It is reasonable to assume the geographic heuristic will be applicable to the DARP and other paired PDPs, and can be combined with other construction and improvement methods.
8 Agent Routing Variable Generation

8.1 Description

The ILP with time discretisations is theoretically limited in terms of optimality of solution by the use of time steps, since aircraft are limited to taking off and arriving only at the nodes associated with a time step. Continuous variable node times would be preferable to the use of time steps, but the formulation becomes too complicated, with a substantial number of constraints, and is unwieldy for problems of the size considered in this work.

An alternative method of obtaining nodes which may be better placed in time for improved solutions is by using simulation-type agents to generate routes and thereby also generate node times. Ant colony optimisation (ACO) also utilises agents which use random search to generate routes, but generally is used for simpler, less constrained problems ([154], [61], [60]).

A method related to random search, but without the learning capability of ACO, was devised for this problem. Effectively agents are allowed to search for feasible or partially-feasible solutions. The solutions obtained are used to generate well-positioned nodes and good legs for a mixed integer linear programming (MILP) formulation. The quality of the solution will then depend on how long the agents are given to search. If no good solution is found, an iterative improvement procedure is proposed.

The method described here involves allowing agents (effectively aircraft) to start at the beginning of the day and route themselves through to the end of the day, collecting and delivering as many groups as possible. At the start of each leg, the aircraft must decide where it should fly to, or whether it should stay at its current location. The aircraft considers flying to each possible destination in turn and assigns an attraction measure to each of these possible locations. A filter ensures that locations which have no reason to be visited by the aircraft are assigned a zero
attractiveness value and therefore cannot be selected. The attractiveness of any location is based on eight factors:

- The distance to the location.
- The available seats in the aircraft.
- Whether there are any groups waiting at the location and whether there is space in the aircraft to accommodate them.
- How close the arrival time of the aircraft would be to the waiting groups’ EDT.
- Whether those waiting groups have similar destinations to any groups which are already on board the aircraft.
- If there are any on-board groups, whether their destination is the location under consideration.
- Whether the aircraft could fly to the location under consideration, then still deliver all of the on-board groups on time.
- The size of the waiting group.

If a location has no waiting groups, and if there are no passengers on board with destinations matching that location, the attractiveness of that location is automatically set to zero.

After an attractiveness is assigned to every possible location, a Monte Carlo sampling procedure is used to randomly select a location to fly to.

The aircraft is then moved to the new location and must decide which waiting groups to collect there. If all waiting passengers cannot fit on board the aircraft, another Monte Carlo selection process is used to choose which passenger groups should be taken on board. The aircraft must then again decide which new location to fly to, or whether it should wait on the ground for other passenger groups to become available for collection.

This procedure was written in MATLAB and is included in Appendix D.

Nine parameters were included in the model to be used to adjust how attractive each factor is (or to weigh the importance of the factor). These parameters were:

1. Difference between current time and LAT of on-board groups.
2. Number of on-board groups who have a destination matching the candidate groups’ location (where candidate groups are groups waiting at the candidate location to be picked up).

3. For the candidate groups, the difference between arrival time and EDT.

4. The geographical distance.

5. The number of on-board groups with destinations matching those of waiting groups.

6. A parameter to increase the attractiveness of a location which is the destination of a group taken on board, when considering taking another group on board at the same location.

7. When at a location and having picked up at least one group, a parameter to add attractiveness to take off without taking on any more waiting groups at that location.

8. The time difference between the arrival time at the candidate location and the time at which the aircraft will be forced to prioritise on-board groups which must be delivered before their LAT.

9. The sizes of the candidate groups.

These parameters can be used to strictly enforce feasibility, or to allow some flexibility in terms of group EDTs and LATs. More flexibility would widen the search.

Parameters to provide values for the attractiveness of the geographical distance and the times were based on an exponential distribution. For example, the geographic distance attractiveness was calculated as $1/e^{d/F_1}$, where $d$ is the distance in kilometers and $F_1$ is the factor giving weight to the overall attractiveness. The exponential distribution was used since it allows for a maximum upper limit and decays rapidly initially. It is often used in other applications such as neural networks where weights are required. In this application it proved to work adequately. However, in future work, Gaussian and linear functions could also be investigated.

A penalty cost is given to undelivered schedule items such that the agent routing function can be optimised with regards to cost. This was set at 10000.

Initially, the agent scheduling procedure is executed with no randomness, i.e. the agent is always routed to the most attractive destination. The attractiveness parameters are then adjusted to obtain the best schedule result under these conditions. In
this process, an optimisation method can be used. For this work, the pattern search optimisation method as supplied with MATLAB Global Optimization Toolbox was applied [155]. The MATLAB pattern search algorithm is effectively a multi-variable hill climbing technique, and as such, may return results from local optima.

Significantly, the best results were achieved when higher relative weighting was given to the waiting group sizes than the other factors. The chosen factor values are shown in Appendix D.

The procedure was then executed using randomness, ie, using Monte Carlo sampling to select a destination at each decision point with weights based on the calculated attractiveness of the possible locations.

The method was used to generate a number of schedules with unconstrained end-of-day fleet locations. Most of the schedules thus generated were not feasible, since the aircraft were all used up without all of the groups on the schedule having been delivered. The routes generated by the schedule were used to create the nodes and legs in the same type of formulation as described previously in Chapter 6, but with some modifications. Because all of the flight leg variables thus introduced already had times associated with them, there was no need for time discretisations. Also, the constraint ensuring all groups were delivered to their respective destinations was softened by removing it from the constraints and including it in the objective function. This was done by adding a number to the objective function for every group correctly delivered (on time and at the correct destination), then maximising the objective. This number is subsequently referred to as the group delivery benefit parameter.

This number, which can be interpreted as the maximum cost allowable for a passenger leg, is used as the revenue (negative cost) coefficient for each passenger ground arc which represents a passenger group reaching their destination on time. An alternative is to use the maximum cost allowable for delivery of a single passenger, in which case larger passenger groups (and hence tourist groups) would be favoured for delivery over single passengers.

Since the objective of this model is to maximise the number of groups delivered, the problem objective function is now maximised. The cost of each leg is still included in the objective function.

This technique ensured that, in the case that insufficient variables were available to obtain a good solution, the model could leave out the most expensive groups to
be delivered. Then, new legs could be introduced to the model in order to reduce
the cost of those deliveries, and the problem solved again. This process could be
repeated as many times as required until a suitably low-cost solution was found.

For this work, in the case that a suitably low-cost schedule could not be found which
delivered all of the groups in the first iteration, additional legs were added to the
problem in subsequent iterations as follows:

1. Consider each of the undelivered groups sequentially.

2. Add a leg from the origin of the group at EDT to the destination for all feasible
   aircraft types.

3. Add the associated legs from each location in the geographic network of the
group to the origin at EDT.

4. Add the associated legs from the destination at arrival time to every other
   location in the geographic network of the group.

5. Repeat the above process (2-4) for:
   - A leg from the groups’ latest possible departure from the origin to the
destination.
   - Legs placed at intervals of 20 minutes (or less) in between the origin at
     EDT and the latest possible departure from the origin to the destination.

The effectiveness of this iterative process is dependent on the above-described method
of adding additional legs to the problem to aid a lower cost solution. It therefore
makes sense, as is done above, to supply legs to and from the origin and destination
of undelivered (i.e. expensive to deliver) passenger groups. If, after an iteration,
an improved solution is not obtained, the legs described can be added at shorter
intervals, or legs can be added from and to other locations and times.

Clearly, the problem will grow in magnitude after each iteration, so carrying out
the above procedure for more than four unscheduled groups was found to cause the
problem solution process to slow significantly. This is apparently not due to the
number of variables, since the number of variables never exceeded 60 000. Since the
solution suffers from slow ILP gap reduction as the gap becomes small, the effect is
most likely due to more solutions being introduced to the problem near the optimum
which are difficult to prune in the branch and bound process. One resolution is to
simply stop the solver at a small enough ILP gap.
Note that the heuristics developed in Chapter 7 are applicable here and were used with similar setup parameters as was used for the MCNFP in Chapter 6.

A reasonable measure of the solution quality would be the load factor. This was calculated by taking the load factors for each flight leg (number of occupied seats divided by aircraft capacity) and multiplying them by the flying time for the leg. These load factors are added for all the legs in a schedule, and divided by the total schedule flying time. However, for this work and for the problem instance under consideration, the minimum cost per group delivery was known to be just over R1000 per group. Therefore this number (cost per group delivery) was used as a performance measure and target. A similar value could most likely be used for other instances of the problem.

The number to be deducted from the objective function for each delivered group (group delivery benefit parameter) can be adjusted to encourage more or less groups to be delivered in any iteration, thus ensuring only the most expensive groups are excluded from the solution.

The problem solution was found to produce a number of incidents of aircraft swapping, as well as some wait times which may be considered too long. To counter this effect, the objective function was modified with a cost per minute of time for each passenger group. This was achieved by calculating the time taken for each leg ($R_y$) and including it as the cost coefficient for each passenger leg variable $y_{xg}$ (see equation (8.1)), and similarly, adding the time taken ($R_t$) for each passenger ground arc $t$ to the cost in the objective function. The resultant MCNF model is as follows:

The objective is to maximize the number of groups delivered while deducting costs:

$$\text{maximise} \quad \sum_{g \in G} \text{BEN} \ t_{ng} - \sum_{y \in Y} R_y y - \sum_{t \in T} R_t t - \sum_{x \in X} C_x x \quad (8.1)$$

where $t_{ng}$ is the passenger group ground arc with start node corresponding to LAT and destination for passenger group $g$. There will be one $t_{ng}$ for each group. $\text{BEN}$ is the group delivery benefit parameter (or maximum cost per group delivered). Note that one could easily make this parameter the maximum cost per passenger delivered if required, by multiplying the size of the group to this term. $R_y$ and $R_t$ are the times associated with passenger leg $y$ and passenger ground arc $t$ respectively. As before, $C_x$ is the cost associated with the flight variable $x$.

The aircraft node conservation of flow constraints are as follows:
\[ \sum_{i \in N} x_{in} - \sum_{j \in N} x_{nj} + s_{(n-1)n} - s_{n(n+1)} = 0, \quad \forall n \in N, \forall f \in F. \quad (8.2) \]

For passengers,

\[ \sum_{i \in N} y_{inx} - \sum_{j \in N} y_{njx} + t_{(n-1)n} - t_{n(n+1)} = 0, \quad \forall n \in N, \forall g \in G, \quad (8.3) \]

where \( t_{(n-1)n} \) refers to the passenger ground arc leading into node \( n \) and \( t_{n(n+1)} \) refers to the passenger ground arc leading out of node \( n \).

The passenger networks and the fleet networks are bound by the aircraft capacity constraints as follows:

\[ \sum_{y \in Y} y_{xg} GrpSize_g \leq x \text{Cap}_x, \quad \forall x \in X, \quad (8.4) \]

where \( GrpSize_g \) is the number of people making up group \( g \). \( Cap_x \) is the passenger capacity of the aircraft type associated with \( x \).

Constraints are needed to ensure the number of aircraft of each fleet type are correct at each location at day start and that the number available is not exceeded. Therefore at the starting node for each location \( u \) in set of all locations \( C \), the ground arc \( s \) for each fleet \( f \) must be set equal to the number of fleets positioned there at day start:

\[ s_{n,f} = v_{uf}, \quad \forall f \in F, \forall u \in C, \quad (8.5) \]

where \( s_{n,f} \) is the ground arc for fleet type \( f \) going into the first node (i.e. earliest time) of location \( u \). \( v_{uf} \) is the number of aircraft of fleet type \( f \) at location \( u \) at the start of the day.

Note that this constraint must be used for all locations where there are no aircraft of some fleet type, in which case \( v_{uf} = 0 \). This is to ensure the correct number of aircraft are used in the model.

The following constraints ensure the correct number of each passenger group in the problem:
where $t_{n,g}$ is the passenger group ground arc with end node corresponding to the first node at location $u$ and $b_{ug}$ is either 1 or zero, depending on whether $u$ is the origin for $g$.

The following constraints ensure passenger groups are at the correct locations at the passenger group EDT:

$$t_{n,g} = b_{ug}, \quad \forall g \in G, \forall u \in C,$$

(8.6)

where $t_{n,g}$ is the passenger group ground arc with end node corresponding to the first node at location $u$ and $b_{ug}$ is either 1 or zero, depending on whether $u$ is the origin for $g$.

The modelling process is shown schematically in a flowchart in Figure 8.1. The process begins with application of the aggregation heuristic, which, in this case, effectively halves the booking list size. Thereafter the reduced booking list is presented to the agents which intelligently and randomly produce a number of schedules. Variable information is extracted from these schedules for the problem. The geographic heuristic is then applied, and the model assembled and solved. If some groups are not delivered, more variables must be added to the problem and the problem reassembled and resolved (an iteration). The delivery benefit parameter can also be increased to encourage more group deliveries, but at an increased cost per group. This process is repeated until all groups are delivered at an acceptable cost.

### 8.2 Observations and Results

Ten schedules were randomly generated using the agent scheduling technique described. The parameters relating to the legs were extracted from these schedules and were used to generate variables to describe the MCNFP (nodes, arcs, etc). The variables were then used in the ILP. This was done using three different values for the group delivery benefit parameter in the objective function, in three different solution schedules, as follows:

1. 2000, two iterations.
2. 1500, two iterations.
Figure 8.1: Agent Routing Method Modelling Process
3. 1000 for two iterations, followed by 1100 for a third iteration.

It was noted that the best solution achievable for this problem is expected to be around R80000 for the full sized schedule. Therefore using a fixed group delivery benefit parameter of just over 1000 is likely to achieve the best solution (solution schedule number 3 above). In practice it was found that for the first iteration, four groups were not delivered. Thereafter, in the second iteration, one group could not be delivered using this value of group delivery benefit parameter, hence the parameter was increased to 1100 for the final iteration, and all groups were delivered.

The method is dependent on the quality of the ten randomly-generated schedules, and on the method of generating additional legs at every iteration. Future work to improve the method could focus on these two aspects.

The ILP solver used was Gurobi provided by NEOS [146]. The NEOS server sends the model for execution to any one of a number of participating computers at educational institutions in the USA. It limits the number of threads used for any particular job to 4.

The results are shown in Table 8.1. In the table, the full schedule was solved using the three different solution schedules with various delivery benefit parameters (column 2), resulting in different numbers of iterations and cost achieved per group delivered. The best result was achieved from solution scheme 3, where three iterations were used to arrive at a final result of R85266 with all groups delivered.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5000</td>
<td>83940</td>
<td>50364</td>
<td>232</td>
<td>1</td>
<td>1</td>
<td>1076</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>86028</td>
<td>50413</td>
<td>1167</td>
<td>2</td>
<td>0</td>
<td>1089</td>
</tr>
<tr>
<td>2</td>
<td>1500</td>
<td>83940</td>
<td>50364</td>
<td>342</td>
<td>1</td>
<td>1</td>
<td>1076</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>86028</td>
<td>51574</td>
<td>1609</td>
<td>2</td>
<td>0</td>
<td>1089</td>
</tr>
<tr>
<td>3</td>
<td>1000</td>
<td>72870</td>
<td>50364</td>
<td>161</td>
<td>1</td>
<td>4</td>
<td>972</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>83887</td>
<td>51574</td>
<td>4189</td>
<td>2</td>
<td>1</td>
<td>1075</td>
</tr>
<tr>
<td></td>
<td>1100</td>
<td>85266</td>
<td>52615</td>
<td>7240</td>
<td>3</td>
<td>0</td>
<td>1079</td>
</tr>
</tbody>
</table>

From Table 8.1, the best schedule for problem 7 is included in Appendix E. Groups that needed to swap aircraft are shown in Table 8.2. Twelve groups needed to swap aircraft, and one of those needed to swap twice. Waiting times varied from 19 minutes, and in three cases exceeded 1 hour.

Using cost penalties of R1 for each minute a customer group spends on a flight leg, and R10/minute for each minute they spend waiting on the ground in-transit,
Table 8.2: Aircraft Changes and Ground Waiting Times

<table>
<thead>
<tr>
<th>Group No.</th>
<th>No. Of Aircraft Changes</th>
<th>No. of Legs</th>
<th>Longest Waiting Time (Min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>21</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>2</td>
<td>166</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>2</td>
<td>113</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>2</td>
<td>69</td>
</tr>
<tr>
<td>31</td>
<td>1</td>
<td>2</td>
<td>19</td>
</tr>
<tr>
<td>36</td>
<td>1</td>
<td>2</td>
<td>19</td>
</tr>
<tr>
<td>41</td>
<td>1</td>
<td>2</td>
<td>22</td>
</tr>
<tr>
<td>44</td>
<td>1</td>
<td>2</td>
<td>25</td>
</tr>
<tr>
<td>48</td>
<td>1</td>
<td>2</td>
<td>117</td>
</tr>
<tr>
<td>53</td>
<td>1</td>
<td>2</td>
<td>21</td>
</tr>
<tr>
<td>70</td>
<td>1</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td>73</td>
<td>2</td>
<td>4</td>
<td>46</td>
</tr>
</tbody>
</table>

Table 8.3: Aircraft Changes and Ground Waiting Times with Passenger Group Time Cost Modified Objective Function

<table>
<thead>
<tr>
<th>Group No.</th>
<th>No. Of Aircraft Changes</th>
<th>No. of Legs</th>
<th>Longest Waiting Time (Min)</th>
<th>No. Passengers Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>1</td>
<td>2</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>31</td>
<td>1</td>
<td>2</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>48</td>
<td>1</td>
<td>2</td>
<td>27</td>
<td>1</td>
</tr>
<tr>
<td>69</td>
<td>1</td>
<td>2</td>
<td>27</td>
<td>2</td>
</tr>
<tr>
<td>70</td>
<td>1</td>
<td>2</td>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 8.4: Aircraft Changes and Ground Waiting Times with Individual Passenger Time Cost Modified Objective Function

<table>
<thead>
<tr>
<th>Group No.</th>
<th>No. Of Aircraft Changes</th>
<th>No. of Legs</th>
<th>Longest Waiting Time (Min)</th>
<th>No. Passengers Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>1</td>
<td>2</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>36</td>
<td>1</td>
<td>2</td>
<td>24</td>
<td>2</td>
</tr>
<tr>
<td>44</td>
<td>1</td>
<td>2</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>70</td>
<td>1</td>
<td>2</td>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>

a single iteration schedule with a cost of R93621 was obtained. This schedule is included in Appendix E. For this schedule, only 5 groups (9 passengers) had to swap aircraft and the maximum intra-journey ground waiting time was 30 minutes (see Table 8.3).

Instead of applying cost penalties to passenger groups, cost penalties can be applied to individual passengers, thus providing increased incentive to deliver larger tourist groups relative to individual staff passengers.

The individual passenger cost was set to R1/passenger per minute for flight legs, and to R10/passenger per minute for intra-journey ground waiting time. This resulted in a schedule with cost R95905 but only 4 groups of 8 people in total required to
swap aircraft and a longest waiting time of 20 minutes (see Table 8.4). Using more randomly-generated schedules (20 in total), 2 iterations were required to reduce this cost to R87471 (R1107 per passenger group).

8.3 Discussion

Compared to the MCNF model from Chapter 6, the fewer variables generated by this method make it more practical in that it solves faster while still providing acceptable solutions. The use of a similar MCNF ILP formulation to that used in Chapter 6 means that swapping of aircraft may occur in the final solution. In this case, 12 groups (15%) needed to swap aircraft. Note that, in terms of service quality, the same options are open to the airline for this method as for the previously described method (Chapters 6 and 7).

If a cost is given to intra-journey waiting times for passengers and passenger legs, the swapping of aircraft and length of ground times are positively affected. Options are to apply a cost per passenger, per passenger group, or to apply a higher cost for tourists and lower costs for boxes and staff. A cost could be added to the objective function to encourage aircraft end-of-day locations, as opposed to using constraints.

The explanation of Tables 8.3 and 8.4 include demonstrations of how the costs can be applied to the objective function to change operational characteristics of the solution. In this case, when costs are applied to passenger time on an individual passenger basis, slightly fewer passengers had to swap aircraft and wait, but the schedule cost was higher. This cost was reduced by adding an additional 10 randomly generated schedules and resolving.

This method provides flexibility, in that it could easily be applied to all kinds of routing problems. The only application consideration is how large the search area is, as a smaller area (more tightly constrained problem) will be advantageous in terms of solution quality.
9 Composite Variable Formulation

9.1 Description

An alternative method of solving the problem involves constructing composite variables as Ronen [137] describes. For this problem, the aircraft flight variables can be combined with each other and with passenger flight variables, such that the resultant variables provide feasible chains of flight legs which can be included as part of a route in a schedule. The advantage of such a formulation is that it removes the need to have separate passenger variables completely, vastly reducing the size of the problem, since the passenger flight variables and passenger ground arc variables are the largest component of the problem.

The procedure begins by applying the aggregation heuristic. Thereafter, the procedure for creating the composite variables is as follows:

1. **Create single flight variables:** For every booking and every fleet type, create a flight variable which can satisfy that booking. This means the aircraft type capacity will be large enough to carry the passenger group, and the length of the flight in terms of time will match the aircraft speed and the distance to be flown.

2. **Dual linking:** For every single flight variable, search for other single flight variables which can be joined on to it. This means the fleet types are similar, the destination of the first variable matches the origin of the second variable and the EDT and LAT for both variables will still be met. However, this is only done in cases where at least one more booking could be satisfied by the linked variable, in other words, at least a third passenger group can fit in the aircraft (sufficient capacity) and will fly on both legs.

The process is shown graphically in Figure 9.1 in which three locations are depicted. Two single flight variables have been found which can be joined,
since flight variable 1 has a destination (ABU) which coincides with the origin of flight variable 2. Flight variable 1 transports passenger groups 3 and 5 from HND to ABU. Flight variable 2 transports passenger groups 2 and 7 from ABU to JAO. Both flight variable 1 and 2 have spare seats available. Therefore they are linked together as shown to form a new, 2-leg flight variable. Then a search is done to find other passenger groups that can be assigned to this new variable. Any groups that have an origin HND and destination JAO and can fit in the aircraft as it flies on the existing two legs will be considered. If no groups can be found, the new variable is discarded. In the Figure, group 1 has been found and assigned to the new variable. Therefore the new flight variable is added to the problem. The existing single flight variables are retained in the problem.

Figure 9.1: Linking of Two Flight Variables (Dual Linking)

3. Further linking: For every 2-link variable, search for single variables which could be joined on to create 3 leg and more linked variables. Again, only create such a variable if some other group or groups could also be assigned to the resultant variable. This linking process can be repeated as many times as required to create longer chains of flight variables. However, the problem can quickly become large and unwieldy, and in such cases the process can be stopped after 3 or 4 link variables have been created.
4. **Replicate variables over time windows:** Each variable has a time window over which it can feasibly take off and land, and still all associated groups’ EDTs and LATs are met. To allow the problem to make use of this time window, the time window is discretised into small time steps of a suitable resolution such as 10 minutes. Nodes are placed at each time step at the departure location. The variable is then replicated within the time window as many times as possible, such that all replicated variables are feasible with regards to time, and each variable starts at a time step node. Nodes are also placed at the ending points of variables at the end location, as calculated using their associated flight times.

5. **Introduce aircraft delivery variables:** Additional flight legs are required to deliver empty aircraft from their locations to where they may be needed for a flight variable. Without these empty flight legs, the problem firstly may not be feasible, and secondly, many cost-effective chain variables might not be used due to unavailability of aircraft. For every flight variable, a flight variable from every other location to the origin of the flight variable and of matching fleet type is generated. These variables do not satisfy any bookings so they are effectively empty aircraft flight legs. Unfortunately, this process causes the number of flight variables in the problem to grow by a multiple of the number of locations.

6. **Introduce end-of-day aircraft delivery variables (optional):** For this problem, in many cases, the model will not require specific empty flight variables added from variable end locations to desired ending locations of aircraft at the end of the day since, if the locations have sufficient flight bookings, the method generates enough variables to ensure aircraft are correctly placed at the end of the day. However, for the 40 booking 7 city problem, it was found necessary to add these empty flights to enable feasibility.

7. **Construct flow network:** Ground arcs are introduced between nodes for the purpose of generating fleet flow conservation constraints at the nodes. These ground arcs are variables representing the number of each fleet type waiting on the ground. Each flight variable is replicated within the time window such that the new variables are each assigned feasible starting and ending times associated with nodes and such that their EDTs and LATs will be satisfied.

The modelling procedure is shown in Figure 9.2.

In practice, this produced approximately 52000 variables for the full-size schedule and for the case where linked variables with a maximum of 3 individual flight legs
Figure 9.2: Composite Variable Modelling Method

1. Execute aggregation heuristic
2. Create single leg variables to satisfy each O-D demand
3. Link these variables in 3-leg chains. Only include chains which satisfy some extra booking
4. Discretise time windows
5. Replicate variables over time windows
6. Add empty aircraft delivery flight legs
7. Generate MCNF model variables
8. Solve and extract solution
were included in the problem. This included all variables, single and linked. Note that if linked variables with a maximum of 4 individual flight legs are added to the problem, the number of variables simply increases by the number of those new, 4 link variables that are generated.

It was unnecessary to apply the geographic heuristic, since the full-sized schedule could easily be solved without it. The geographic heuristic (with \( d=50 \text{km} \) and \( \alpha=45^\circ \)) could be used for other, larger problem instances such as multi-day problem instances.

The multi-commodity network flow model was formulated with two commodities, these being the two aircraft types/fleets, \( f \) in \( F \). The model was created to select flight variables at a minimum cost and to ensure every booking is satisfied at least once. If \( X \) is the set of all flight variables \( x \), each with cost \( C_x \), the objective is:

\[
\text{minimize } \sum_{x \in X} C_x x. \tag{9.1}
\]

A subset of flight variables was created for each booking \( x^b \in \{ X : b \} \), this set including only the flight variables which could satisfy the booking in question. Constraints can then be derived to ensure that each booking \( b \) in set of all bookings \( B \) is satisfied at least once:

\[
\sum_{i \in N} \sum_{j \in N} \sum_{f \in F} x^b_{ijf} \geq 1, \quad \forall b \in B. \tag{9.2}
\]

Constraints are required to maintain conservation of fleet flow. This is done by generating an aircraft flow conservation constraint for each fleet at each node. The flow constraints are as follows for each node \( n \) in the set of all nodes \( N \):

\[
\left( \sum_{i \in N} x_{inf} + s_{(n-1)n_f} \right) - \left( \sum_{j \in N} x_{njf} + s_{n(n+1)f} \right) = 0, \quad \forall n \in N, \forall f \in F \tag{9.3}
\]

where variables \( s_{(n-1)n_f} \) and \( s_{n(n+1)f} \) refer to the ground arc into and out of node \( n \) respectively. Note that, as before (Chapter 6), ground arcs are fleet specific, so \( s_{(n-1)n_f} \) and \( s_{n(n+1)f} \) are associated with a fleet type \( f \).

Since the inclusion of a variable in a solution now feasibly satisfies the origin and
destination requirements, as well as time window requirements, for all of the associated passenger groups, no nodal conservation of passenger group flow constraints are required.

Since each node has an associated location and time, the ground arc leading to the first node at any location (i.e. with the earliest time) should be set to the number of aircraft of that fleet type at that location at the start of the day:

\[ s_{n_i f} = v_{u_i f}, \quad \forall u \in C, \forall f \in F, \quad (9.4) \]

where \( v_{u_i f} \) is the number of aircraft starting at location \( u \), and \( s_{n_i f} \) is the ground arc for fleet type \( f \) going into the first node (earliest time) at location \( u \).

Similarly, the ground arc leading from the last node of the day will represent the number of aircraft at a particular destination at the end of the day, and can be constrained if required:

\[ s_{n_o f} = v_{u_o f}, \quad \forall u \in C, \forall f \in F, \quad (9.5) \]

where \( v_{u_o f} \) is the number of aircraft ending at location \( u \), and \( s_{n_o f} \) is the ground arc for fleet type \( f \) going out of the last node (latest time) at location \( u \).

9.2 Observations and Results

9.2.1 Small Schedule

The problem was executed with constrained end-of-day aircraft locations (i.e. including equation (9.5)) and the maximum number of linked variables included in the problem was varied from 3 to 5. The results are shown in Table 9.1. It was found that no further linking occurred after variables with 4 links had been produced.

The best solution achieved (cost R44200) was 6.9% from the best solution using the MCNF ILP with limited passenger group waiting time in Chapter 7 (cost R41338). The resultant schedule is shown in Table 9.2. In this table, the group numbers refer to the original, un-aggregated group numbers. Groups that appear on two different aircraft have a choice of more than one leg. Thirty flights in total are used for the schedule.
Table 9.1: Small Schedule Costs Using Composite Variables

<table>
<thead>
<tr>
<th>No of Links</th>
<th>Min Z (Rands)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>44200</td>
</tr>
<tr>
<td>4</td>
<td>44200</td>
</tr>
<tr>
<td>5</td>
<td>44200</td>
</tr>
</tbody>
</table>

Table 9.2: Small Schedule Using Composite Variables

<table>
<thead>
<tr>
<th>Fleet Type</th>
<th>Origin Location</th>
<th>Departure Time</th>
<th>Destination Location</th>
<th>Arrival Time</th>
<th>Passenger Groups on Board</th>
</tr>
</thead>
<tbody>
<tr>
<td>C206</td>
<td>7</td>
<td>370</td>
<td>6</td>
<td>400</td>
<td>14, 15, 20</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>450</td>
<td>3</td>
<td>530</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>680</td>
<td>1</td>
<td>720</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>720</td>
<td>2</td>
<td>850</td>
<td>30, 31, 32</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>920</td>
<td>7</td>
<td>960</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>970</td>
<td>4</td>
<td>1020</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1020</td>
<td>7</td>
<td>1070</td>
<td>22, 26, 40</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>380</td>
<td>4</td>
<td>430</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>430</td>
<td>7</td>
<td>480</td>
<td>3, 4</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>480</td>
<td>4</td>
<td>530</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>670</td>
<td>3</td>
<td>720</td>
<td>2, 29</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>910</td>
<td>4</td>
<td>970</td>
<td>33, 34, 25, 27</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>970</td>
<td>7</td>
<td>1020</td>
<td>23, 25, 27</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>450</td>
<td>2</td>
<td>490</td>
<td>12, 13, 19</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>490</td>
<td>3</td>
<td>590</td>
<td>8, 19</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>690</td>
<td>5</td>
<td>750</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>870</td>
<td>6</td>
<td>910</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>970</td>
<td>7</td>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>460</td>
<td>2</td>
<td>500</td>
<td>12, 13</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>500</td>
<td>7</td>
<td>540</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>860</td>
<td>2</td>
<td>900</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>960</td>
<td>7</td>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>390</td>
<td>6</td>
<td>420</td>
<td>14, 15, 20</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>480</td>
<td>3</td>
<td>540</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>620</td>
<td>5</td>
<td>670</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>720</td>
<td>6</td>
<td>750</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>810</td>
<td>7</td>
<td>830</td>
<td>16, 17, 18, 39</td>
</tr>
<tr>
<td>C208</td>
<td>6</td>
<td>480</td>
<td>3</td>
<td>540</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>710</td>
<td>4</td>
<td>770</td>
<td>1, 25, 27</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>770</td>
<td>7</td>
<td>820</td>
<td>9, 25, 27</td>
</tr>
</tbody>
</table>
9.2.2 Full Schedule

The full schedule was solved with constrained end-of-day aircraft locations (i.e. including equation (9.5)) and the results are shown in Table 9.3. Column 1 in the table refers to the maximum number of legs that were linked together to form a variable. As longer linked variables are constructed and inserted into the problem, the number of variables in the problem (column 2) increases accordingly. This is because each time longer linked variables are generated, these additional variables are simply added to all of the existing variables. The processing time (column 4) also increases and the schedule cost decreases (column 3, “Min Z”).

Note that if a schedule is produced from variables that have a maximum number of three links, no passenger group will have a trip of more than three aircraft legs.

Table 9.3: Full Schedule Costs Using Composite Variables with Constrained End-Of-Day Aircraft Locations

<table>
<thead>
<tr>
<th>Max. No. of Links</th>
<th>No. of Variables</th>
<th>Min Z (Rands)</th>
<th>Processing Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>42522</td>
<td>88230</td>
<td>18.1</td>
</tr>
<tr>
<td>3</td>
<td>52404</td>
<td>87756</td>
<td>32.9</td>
</tr>
<tr>
<td>4</td>
<td>61728</td>
<td>86787</td>
<td>251.9</td>
</tr>
<tr>
<td>5</td>
<td>82262</td>
<td>86787</td>
<td>665.4</td>
</tr>
</tbody>
</table>

The problem was solved with unconstrained end-of-day aircraft locations. This version produced a schedule cost of R87683 for the problem including a maximum size of 3 link variables (9.8s solution time). This solution cost is similar to that obtained using the method described in Chapter 8 for the equivalent problem (R87471), but has the advantage of no aircraft changes for passengers and is faster to solve (32.9s vs more than 1 hour).

9.3 Discussion

From Table 9.3, the processing time increases and the schedule cost decreases as the maximum allowable number of links increases.

The method solves quickly if the number of variables in the problem is limited. The method can not accommodate swapping of aircraft and limits the amount of passenger ground time en-route. This makes it ideal for this particular problem, since a high quality of service can be maintained. The method retains flexibility in allowing end-of-day aircraft positions to be constrained. Even without providing
empty flights to deliver aircraft to their end-of-day locations, the version with unconstrained locations did not differ considerably from the constrained version (0.6% difference). Quality of service can also be adjusted by allowing variables with fewer linked legs in the problem for tourist groups, and allowing more links in journeys for boxes and staff.

A closer look at the small schedule produced using this method in Table 9.2 shows the schedule uses 30 flights (i.e. the table has 30 rows) in total. Extensive use is made of the C206 (22 flights), and the longest chain of flights any passenger had to endure on one journey was, as expected, three legs.

The composite full-size schedule produces a schedule of similar cost to the agent variable generation method (R87756 vs R87471). For the constrained end-of-day version, the MCNF ILP method suggests the problem lower bound is R76318 with passenger groups swapping aircraft and with limited passenger intra-journey ground waiting time. The best composite schedule cost is therefore 13.7% or less from optimality.

Table 9.4 shows a comparison of all of the methods and results from them. The composite variable method outperforms the others in terms of speed of processing for the airline taxi problem considered, yet still provides a comparable schedule cost. Since processing speed is important in practice, this method is the preferred method for implementation at the airline.

<table>
<thead>
<tr>
<th>Method</th>
<th>Min Z (Rands)</th>
<th>No. of Variables</th>
<th>Processing Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard MCNF ILP</td>
<td>90819 (19.4% gap)</td>
<td>177718</td>
<td>12930</td>
</tr>
<tr>
<td>Agent routing</td>
<td>87471 (1.84% gap)</td>
<td>52172</td>
<td>3630</td>
</tr>
<tr>
<td>Composite variable</td>
<td>87756</td>
<td>52404</td>
<td>32.9</td>
</tr>
</tbody>
</table>
10 The Sefofane Air Scheduling Problem

10.1 Constrained Problem

The results obtained for various methods considered in this work are adequate in comparison to the manual schedule in terms of cost and schedule quality for the problem at hand. This is because they provide comparatively lower cost, practical and usable solutions. However, the composite variable method is superior in terms of speed of processing. In addition, the method produces a schedule directly comparable to the manual schedule since it does not allow passenger groups to swap aircraft en-route. The process of problem setup offers reasonable flexibility in terms of taking further constraints into account, as will be discussed in this section. Therefore it was used to solve the actual problem supplied.

The actual problem faced by Sefofane Air has additional considerations which need to be accounted for before the value of these solution methods can be evaluated for their specific case:

**Fuel, range and maintenance:** The first consideration is fuel use. Sefofane Air uses the C206 with a range of 1352 km and the C208 Grand Caravan with a range of 1700 km [150], both of which can comfortably cover all the routes proposed by all of the models in one day. Additionally, fuel is available at the most visited airstrip (Maun) and a number of other larger airstrips (Victoria Falls, Vumbura and others), so refueling could be done during the execution of almost any routing, particularly since invariably either the day start or day end is at Maun for most, if not all, routings. Time to refuel could be gained by extending the turnaround times, but this is unlikely to be necessary. Sefofane Air schedules 10 minutes for turnaround times. Maintenance is currently done by leaving the aircraft out of the schedule and booking it for maintenance, therefore in this work maintenance did not need to be taken specifically into account. A shortage of aircraft must be handled by hiring additional aircraft and pilots, or refusing bookings. The airline schedule is
only finalized at most two or three days in advance, often only one day prior to the
day in question.

**Multiple day scheduling and end-of-day aircraft location considerations:**
Since the model solves quickly it may be possible to solve for multiple days, especially
if fewer bookings are made, as will often be the case. The geographic heuristic can
be applied if the number of variables grows too large. However, as discussed above,
the schedule may only be finalized the day before the day in question, so this
is not likely to be needed in practice. It may be useful to ensure an aircraft is at a
certain location or choice of locations at the end of the day. This could be the case
when the aircraft will need refueling or maintenance, or when it is undesirable to
leave it in a certain location overnight (As an example, some airstrips have a high
concentration of elephants in the area). Examination of the manual schedule reveals
that of the 13 aircraft used on the day, 5 were transported empty at the end of their
days’ routing, indicating that they were specifically being positioned for the next
day’s flights. The aircraft that were flown with passengers for the last flight of the
day may or may not have been specifically positioned.

In practice, as could be expected and similarly to the manual schedule, the schedules
generated by the model make extensive use of the hub destination, Maun (7, this
number refers to the unique number assigned to this location for modelling purposes,
see Figure 5.3). This is because more bookings have either an origin or a destination
there than any other location. Many daily routings start and end at Maun. In
fact, it was found that when the composite model was solved without constraining
the aircraft end-of-day locations, the locations selected by the model were similar
to those of the manual schedule for most (9) of the 15 aircraft. Therefore the
two versions of the schedule (constrained and not constrained end-of-day aircraft
positions) do not differ by a large cost margin.

**Passenger classes - boxes, staff, customers:** Sefofane Air has three different
types of passengers, these being the actual tourists or customers, staff of the airline
or more likely staff working for the companies that operate safari camps, or boxes.

Most of the single person bookings are staff, and Sefofane Air could consider treating
them differently from tourist groups by allowing them to swap aircraft or have longer
flights with more legs. However, this would require careful consideration as it would
raise the cost to the employer of the staff. Note that the composite schedule could
easily be set up to allow boxes and staff to have more linked legs in their journeys
than the tourist groups.
From the manual schedule, it is clear that Sefofane Air schedulers try to group boxes together that can be taken in a full plane load. This automatically prevents tourists being ferried with boxes and allows special “cargo” delivery flights to be undertaken. From an operational perspective, safari camps could be encouraged to book an entire airplane flight when they require cargo.

**Loadmasters and co-pilots for the C208:** In general, the C208 requires a loadmaster for passenger flights. Therefore there must either be a loadmaster available where the C208 is located, a loadmaster must be flown in, or the C208 must fly a leg to collect a loadmaster. In the manual schedule, two of four loadmasters used were collected at Linyanti (4)(one who had stayed with the aircraft overnight) and one was left with the aircraft to overnight at River Club (18) at the day end. The simple way to deal with this in the case of the automated schedule is to always only allow 11 passengers to be ferried in the C208. The schedule must then be manually adjusted to ensure all the loadmasters are picked up from their relevant locations before embarking on the day’s schedule. Since most loadmasters are collected at the hub (Maun) and mostly accompany the C208, the adjustments will be minor, if any.

In many airline scheduling problems, the amount of flying time a pilot is allowed to undertake in a day is a constraint on the problem. However, in this problem the longest pilot scheduled flying time was 7h40 min, with all of the other flying times being less than 5 hours. Even if a flying time of longer than 8 hours (the maximum allowed as specified by the Federal Aviation Authority [156]) were to be generated, there are almost always stopovers at the main hub at Maun to allow a change of pilot.

**Special requests:** From the data supplied, one large passenger group had a special request, this being that their group alone be ferried in an aircraft, with no other passengers on board. This particular group consisted of 13 passengers, meaning that at least two aircraft were needed to be used to ferry them. Since each C206 carries only 5 passengers, at least one C208 is required. However, since the manual schedule used two equally loaded C208s for this purpose, we must assume that the tourists in question requested C208 transport specifically. The simple way to deal with this problem was to artificially resize the group into two groups of passenger size 11 each, forcing them to be ferried in two C208s, as was done in the manual schedule.
10.2 Results and Discussion

Minor adjustments needed to be made to the automated schedule before it could be compared to the manual schedule. These related to the loadmasters, certain of which were not at the same location as a C208 at the start of the day, and the boxes, which needed to be transported without a loadmaster. Two loadmasters were situated at Linyanti (4) and not at Maun where most C208s started the day, and these loadmasters had to be collected for use. Fortunately, for this particular day, no schedule adjustments were needed to pick up the two loadmasters (see Appendix F). This is because one C208 started the day at Linyanti and therefore could collect the loadmaster there. The other loadmaster could be collected by one of the scheduled C206s en-route and dropped at Maun in time to be used by another C208. This can be seen on the schedule provided by the composite method before adjustments shown in Appendix F, where the 3rd C206 in the schedule has one seat available for the loadmaster from locations 7 to 4.

The airline had assembled boxes into groups of 5 and 12, ensuring a C206 or a C208 with no loadmaster was required for delivery thereof. Fortunately, the automated schedule fitted the trips with boxes into the schedule in much the same way as the manual scheduler had done. For example, the 2nd C208 in the schedule carries the boxes, therefore it can drop the loadmaster at location 20 (Victoria Falls) and collect the loadmaster after having delivered the boxes. Therefore there were no cost adjustments to be made, and in this case, the composite schedule can be compared directly with the manual schedule. The manual schedule cost was R99633, while the composite variable automated schedule cost was R87756, thus a saving of 12% (R11877) for the day was achievable. The automated schedule is included in Appendix F.

The booking list supplied was for a particularly busy day for the airline, and most booking lists consist of about 80 bookings. Therefore, if the average number of bookings per day is assumed to be 80, the annual savings of using the scheduling method proposed here is approximately R2.2m.

The composite method is flexible in that it can be applied to problems in various ways. It can be adjusted for optimal performance for a specific problem type. In this problem, possible adjusting mechanisms include:

1. Longer linked chains for specific passenger classes
2. Modifying the heuristics involved in the linking process, e.g. geographic heuristic in Chapter 7.

3. Different linking mechanism according to aircraft type

In the composite variable method, linked variables represent preferred, high load factor routes for an aircraft. In the larger problem instances, it is clear that more linking is advantageous. However, if longer chains with more legs are allowed then the problem grows in size and becomes slower to solve.

The composite formulation method works well in this specific case because the problem is highly constrained, thus it is advantageous to remove the many unreasonable routes. One might therefore consider applying this technique or similar to other difficult-to-solve highly-constrained vehicle routing problems.

This problem would have been easier to solve if the C208 only had 10 passenger seats, since then groups could be aggregated in groups of up to five, each representing either an aircraft load in the case of the C206, or half an aircraft load in the case of the C208. Even for this problem as it stands, it might be advantageous to solve it as such, then, afterwards, make modifications and rearrange any single passengers to take advantage of the empty seat in the C208, possibly using a cost-savings, improvement type of heuristic.

In the solutions proposed in this report, the option of load splitting has not been included due to quality of service considerations. Tourist groups can obviously not be split, and staff normally travel alone. However, box loads could be split with a relatively minor effect on service quality. Therefore, an improved solution might be obtainable by allowing splitting of boxes. This could be combined with allowing boxes to endure any amount of journey time, giving further improvements. These options should only be considered if it is found that certain days deal with relatively more transport of goods (boxes) than occurred on the day considered in this work. These options could easily be included in an agent routing variable generation method.

It should be noted that the airline could use these models to optimise their fleet. To do this, they would run models (or simulations) for the previous six months of their operations using different numbers of aircraft of various aircraft types, taking into account the capital cost and running cost of the aircraft, and thereby predict the cost of different fleet make-ups.
11 Other VRPs

11.1 Introduction

The methods introduced in this work may be applicable to other VRPs, particularly PDPs, but possibly other problems with time windows. Real-world problems are often highly constrained and require flexibility in the modelling process, which is where these methods should be most useful. For these reasons, the methods are demonstrated on two VRPs, one a real-world MVCVRP, and the other a CVRPTW obtained from the Internet. Note that these methods are merely to demonstrate that the techniques devised in this work can be used to approach other VRP problems.

11.2 Multi Vehicle CVRP (MVCVRP)

11.2.1 Problem Description

Real-world vehicle routing problems are often highly constrained. An example of such a problem is a MVCVRP. In such a problem, different vehicles will have different speeds of travel and different costs, traffic jams are present during certain times of the day, daily travel time is limited, staff may take the lunch hour off and may not receive goods during that time, etc. It is under such conditions that the composite technique is advantageous, since it can deal with many of these constraints in the formulation process.

The MVCVRP is a practical problem experienced by companies in the field of wholesale food distribution, and a typical such company is Hoxies in Pretoria [157]. Hoxies need to carry out up to 128 order deliveries of food per day using their fleet of 30 trucks of 8 different types and capacities, and from their single depot. Each vehicle has a different cost of usage per kilometer and average travelling speed. For this
work, cost of use per kilometer was artificially derived from various sources including rental [158], trucking [159] and AA rates [160]. Travelling speed was estimated. The data is included in Appendix G. Since this MVCVRP is to demonstrate the technique only, the accuracy of these figures is considered adequate.

Although Hoxies could not supply the geographical positioning of their customers, the approximate area within which the customers were situated was known. The number of trucks of each size was supplied, as well as three delivery lists from busy days (Appendix G, section G.2). Trucks can make multiple deliveries in a day. To make test schedules, the positioning of customers for each schedule item was uniformly generated from the size of the delivery area. A day duration of 7 hours and turnaround time of 15 minutes were used. The three schedules used are included in Appendix G.

Note that no information was available regarding when customers would like their deliveries scheduled, but it was known that certain customers did not want deliveries when staff were taking a lunch break. Also, information regarding traffic and times when traffic is slow was not available. Therefore these constraints were ignored. They could easily be incorporated in our formulation and would make the solution process faster.

A small and medium sized schedule were also produced. For these schedules, the customer positions were generated as before, and the order sizes were obtained by randomly sampling Monte Carlo style with replacement from a list. This list consisted of all of the orders obtained for all of the three days supplied. As previously, sampling with replacement ensures samples are independent and having the same sample more than once in a schedule is acceptable. This is because geographic positions for all samples were randomly generated anyway. One large-size (128 customer, as supplied by Hoxies) schedule was also solved.

11.2.2 Exact ILP Formulation

An exact ILP was devised to solve the various problem instances, particularly the smaller instances, for comparison purposes. This MCNF ILP is as follows:

Define the following decision variables:

1. Set of binary leg variables $x_{ijf}$, each corresponding to a trip or leg. Each such leg is associated with a vehicle type $f$ in $F$, a starting node $i$ in node set $N$
and an ending node $j$ in node set $N$.

2. Set of load variables $y_{x}$, each corresponding to a leg $x$. This is the load carried by a vehicle on a leg.

3. Set of auxiliary binary variables $b_{nf}$, each corresponding to a location or node $n$ and vehicle type $f$.

The objective is to minimize costs:

$$\text{minimize } \sum_{x \in X} C_{x} x, \quad (11.1)$$

where $C_{x}$ is the cost or distance of leg $x$.

Constraints must ensure conservation of vehicle and load flow at every network node $n$ in $N$.

For vehicles:

$$\sum_{i \in N} x_{inf} - \sum_{j \in N} x_{njf} = 0, \quad \forall n \in N, \forall f \in F, \quad (11.2)$$

Constraints are needed to enforce conservation of load flow at the nodes. This is done using constraints 11.3 and 11.4, together, but only if auxiliary variable $b_{nf}$ is set to 1:

$$\sum_{i \in N} y_{xin} - \sum_{j \in N} y_{xjn} + M b_{nf} \leq M - L_n, \quad \forall n \in N, \forall f \in F, \quad (11.3)$$

$$\sum_{j \in N} y_{xjn} - \sum_{i \in N} y_{xin} + M b_{nf} \leq M + L_n, \quad \forall n \in N, \forall f \in F. \quad (11.4)$$

$M$ is the “BigM” commonly used in LP formulations, and in this case, set to be 10000. $L_n$ is the load to be delivered to location/node $n$.

If an auxiliary variable $b_{nf}$ is set to 0, no leg $x_{ijf}$ connected to the associated node $n$ may be set to 1. The following constraint enforces this, both for legs entering and legs leaving the node:

$$\sum_{i \in N} x_{in} + \sum_{j \in N} x_{nj} - M b_{nf} \leq 0, \quad \forall n \in N, \forall f \in F, \quad (11.5)$$
Similarly, if an auxiliary variable $b_{nf}$ is set to 0, the load associated with legs $x_{njf}$ leaving that node must be 0:

$$\sum_{j \in N} y_{x_{njf}} - M b_{nf} \leq 0, \quad \forall n \in N, \forall f \in F.$$  \hspace{1cm} (11.6)

If an auxiliary variable $b_{nf}$ is set to 1, at least one associated leg $x_{inf}$ into the node must be set to 1:

$$-\sum_{i \in N} x_{inf} + M b_{nf} \leq M - 1, \quad \forall n \in N, \forall f \in F.$$  \hspace{1cm} (11.7)

Vehicle capacity constraints:

$$y_x - \text{Cap}_x \leq 0, \quad \forall x \in X,$$  \hspace{1cm} (11.8)

where $\text{Cap}_x$ is the load capacity of the vehicle type associated with $x$.

If leg $x$ is unused, route load $y_x = 0$:

$$-Mx + y_x \leq 0, \quad \forall x \in X.$$  \hspace{1cm} (11.9)

Demand constraints are required to ensure each location is visited once:

$$\sum_{i \in N} x_{inf} \geq 1, \quad \forall n \in N, f \in F.$$  \hspace{1cm} (11.10)

Note that this formulation does not constrain working day time, i.e. vehicles could work more than the usual working day time of 7 hours. This is because the formulation becomes too complicated and the solution too slow. The formulation also ignores truck speed and has distance minimizing as the objective. Therefore it can be considered to produce a lower bound for the problem being addressed.
11.2.3 Composite Variable Formulation

Formulation Methodology

In order to simplify the formulation, this problem can be set up with composite variables. An outline of the variable construction procedure is as follows:

- Create variables consisting of trips or legs, with the origin being the depot and the customers being the locations, and vice versa.
- Create variables consisting of legs from one customer to another ("single variables").
- Create chains of variables:
  1. Take selected single variables (for example, the three with origins closest to the depot) and search for other single variables which can be linked to them. Various heuristics can be included in this process.
  2. Repeat step 1, but use each linked variable and search for single variables to link to, to create longer chains, until no more linking is possible. This occurs when no more single variables can be found which can feasibly be connected to the end of the chain, taking into account the duration of the chain trip including the return trip to the depot, which must be less than 7 hours, and the total load the vehicle can carry on one trip.

It should be noted that all variables thus created can be added to a solution and the load capacity and time available constraints will not be violated.

Composite Variable Construction Process

For this problem, a heuristic was necessary to reduce the number of variables produced to a reasonable level. This was simply to only include the shortest arcs in the problem, as proposed by Garcia et al. [48]. The complete and detailed heuristic variable generation process used was as follows:

1. **Generate single legs from the depot to selected locations and back for all fleets:** For each fleet and each customer, a variable from the depot to the customer, and a variable from the customer to the depot, was generated. As proposed by Garcia et al. [48], the customers were selected as the closest
customers only. In this case, the closest three customers were used, since this number ensured the number of variables created was not too large to be manipulated.

2. **Generate single legs between every location (excluding the depot) and the closest other locations:** In this work, the closest three locations were used.

3. **Link the single variables created in step 2 to other single leg variables created in step 2, and repeat this process:** Generate as long chains of single legs as possible, ensuring that the capacity of the vehicle is never exceeded. Each trip was also limited to the length of a working day (7 hours).

All of the variables produced in steps 1, 2 and 3 are then included in a simple, cost-minimising ILP which has three constraints. Two of these are to ensure flow conservation of vehicles, and the third to ensure every customer is visited. The formulation is as follows:

The objective is:

\[
\text{minimize } \sum_{x \in X} C_x x \tag{11.11}
\]

where the set \( X \) contains all of the variables \( x_{ijf} \), where index \( i \) refers to the starting node for variable \( x \), and \( j \) the ending node, both from the set of all nodes \( N \). \( C_x \) is the cost associated with variable or leg \( x_{ijf} \), and is calculated as \( C_x = D_x \times cpk_f \). Here, \( D_x \) is the distance of variable \( x \) and \( cpk_f \) is the cost per kilometer of using vehicle type \( f \).

There are two types of flow constraints. The first is to connect depot-location variables to linked variables, and the second to connect linked variables to location-depot variables:

\[
x_{dnf} - \sum_{j \in N} x_{njf} = 0, \quad \forall n \in N, \ f \in F, \tag{11.12}
\]

and

\[
\sum_{i \in N} x_{inf} - x_{ndf} = 0, \quad \forall n \in N, \ f \in F, \tag{11.13}
\]

where \( d \) refers to the depot location/node.

The demand constraints enforce the requirement that all locations must be visited at least once:
\[ \sum_{i \in N} \sum_{f \in F} x_{inf} \geq 1, \quad \forall n \in N. \] (11.14)

### 11.2.4 Methodology

Three problems were solved, a small, medium and large version, and by both techniques (composite variable formulation (11.11) - (11.14) and exact ILP formulation (11.1)-(11.10)). The geographic positioning of customers was uniformly generated between horizontal and vertical coordinate values of 0 and 80. In all cases, the depot was placed at a horizontal position of 50, and vertical position of 20.

All problems were inspired by the real problem experienced by Hoxies. However, all small and medium instances were generated randomly by sampling from the three actual daily schedules supplied.

The problem has one depot and a number of customers and fleets. The small version has 10 customers (Table 11.1) and 3 fleets (Table 11.2). The medium sized (20 customers) and actual (large sized, 128 customers) data is included in Appendix G.

<table>
<thead>
<tr>
<th>Customer No.</th>
<th>Horizontal (X, km)</th>
<th>Vertical (Y, km)</th>
<th>Load (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>73.7</td>
<td>27.0</td>
<td>207</td>
</tr>
<tr>
<td>2</td>
<td>59.6</td>
<td>20.8</td>
<td>311</td>
</tr>
<tr>
<td>3</td>
<td>10.1</td>
<td>32.6</td>
<td>975</td>
</tr>
<tr>
<td>4</td>
<td>50.2</td>
<td>68.9</td>
<td>173</td>
</tr>
<tr>
<td>5</td>
<td>55.2</td>
<td>5.0</td>
<td>459</td>
</tr>
<tr>
<td>6</td>
<td>32.8</td>
<td>42.1</td>
<td>2737</td>
</tr>
<tr>
<td>7</td>
<td>42.5</td>
<td>60.3</td>
<td>278</td>
</tr>
<tr>
<td>8</td>
<td>19.7</td>
<td>3.7</td>
<td>597</td>
</tr>
<tr>
<td>9</td>
<td>34.1</td>
<td>43.9</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>26.2</td>
<td>7.7</td>
<td>195</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fleet No.</th>
<th>Maximum Load (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1000</td>
</tr>
<tr>
<td>2</td>
<td>2000</td>
</tr>
<tr>
<td>3</td>
<td>6500</td>
</tr>
</tbody>
</table>
11.2.5 Observations and Results

The solution costs and computation times for each problem solved are shown in Table 11.3. The problem instances, except for those indicated, were solved using CPLEX 12.1 solver.
Table 11.3: MVCVRP Solutions

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Method</th>
<th>Cost (R)</th>
<th>Computation Time (s)</th>
<th>No. of Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>ILP</td>
<td>267.4</td>
<td>0.4</td>
<td>578</td>
</tr>
<tr>
<td>10</td>
<td>Composite</td>
<td>286.8</td>
<td>0.03</td>
<td>1001</td>
</tr>
<tr>
<td>20</td>
<td>ILP</td>
<td>616.9</td>
<td>87.1</td>
<td>4888</td>
</tr>
<tr>
<td>20</td>
<td>Composite</td>
<td>647.0</td>
<td>0.03</td>
<td>8801</td>
</tr>
<tr>
<td>128</td>
<td>ILP</td>
<td>1221 (15.1% gap)</td>
<td>7098</td>
<td>133533</td>
</tr>
<tr>
<td>128</td>
<td>Composite</td>
<td>1672.0</td>
<td>3.2</td>
<td>17676</td>
</tr>
</tbody>
</table>

11.2.6 Discussion

For both the small and medium problems, the composite variable technique produces a solution within 7.5% of the exact solution. The composite variable model solution for the large instance is comparably poor. This is due to the exact solution not having a limitation on the time of day for deliveries to be made. The exact ILP solution produced routes with around 15 customers on them which would be too many to do in a 7 hour day.

Without the daytime limitation, the composite variable method produces too many variables to be practically useful. Having restrictions on lunch time deliveries and traffic would further enhance the relative performance of the method for this problem. The method is therefore most appropriate for highly constrained problems.

11.3 Capacitated Vehicle Routing Problem with Time Windows (CVRPTW)

11.3.1 Problem Description

Data describing a CVRPTW was obtained from Solomon via the Internet [147]. The particular instance used is the “RC101” instance, a 100 customer problem with an optimal solution (shortest distance) of 1620. The data is described by Solomon as geographically “a mix of random and clustered structures”, with “a short scheduling horizon and only a few customers per route (approximately 5 to 10)”. The maximum vehicle capacity is 200 and the turnaround time is 10 units. The problem data is included in Appendix I.

Only one instance of a reasonable size was obtained and used, since the intention is to show how the method would be applied to such a problem and to obtain a
comparative result. The problem is not a real world problem, and as such, not sufficiently constrained to suit the method. One would expect little difference in performance for the other instances of this problem.

The problem is modelled using a time-space network for each fleet. The agent routing method of variable generation is used with a MCNF formulation, followed by solution using an ILP solver.

Note that, in this context, EAT refers to the start of the time window for the location, and LDT refers to the end of the time window. Vehicles could, of course, arrive at the location before the EAT and wait, or wait at the location and leave after the LDT.

Note that, for the CVRPTW, vehicles could equally be considered to leave the depot empty and collect loads at customers, or to leave the depot fully loaded, drop loads off at customers, and return empty. In the formulations which follow, the vehicle leaves the depot empty and collects loads at customer premises.

11.3.2 Variable Generation

The first stage of the process is to generate agent routes. In this case, for agent decision making when choosing the next location to move to, three attraction factors are used:

1. The geographical distance
2. The distance in time from current time until the earliest arrival time (EAT)
3. The size of the load to be collected

As before, the routing code (Appendix H) is run with no randomness in order to select the best attractiveness weightings. In this case, the agent always chooses the most attractive next location to route to. The MATLAB optimisation procedure “Pattern Search” was used [155]. Thereafter the agents are routed 20 times and the nodes and legs extracted from these randomly-generated routes and used in an MCNF formulation. Because a time-space network is used, each node has an associated geographical position and time. Ground arc variables $s$ in set $S$ connect all nodes $n$ at each location.
Each route is given a benefit in the objective function if it is visited in a solution, less the cost (distance) to get there. The function is then maximised.

The MCNF formulation is based on a time-space network, as follows:

**Decision variables**

- Arc variables $x_{ijf}$ in set $X$ (binary) - These represent a trip or leg between two locations (nodes $i$ and $j$).
- Ground arc variables $s$ in set $S$ (binary) - These represent a period spent waiting at some location between two nodes with different times.
- Arc load variables $l$ in set $L$ - These represent the total load on a vehicle when traversing an arc. Note that this variable represents the sum of loads from all locations visited up to that point on a journey.
- Ground arc load variables $p$ in set $P$ - These represent the load on a vehicle during waiting associated with a ground arc $s$. Again, this is the sum of the loads from all of the customers previously visited on the journey.

**Objective function**

$$\text{maximize } \sum_{x \in X} B_x - \sum_{x \in X} D_x x$$

where $D_x$ is the distance associated with variable $x$, and $B$ is a parameter representing the benefit of visiting a location.

**Constraints**

Network flow constraints are required for the vehicles:

$$\sum_{i \in N} x_{inf} - \sum_{j \in N} x_{njf} + s_{(n-1)nf} - s_{n(n+1)f} = 0, \ \forall n \in N, \forall f \in F,$$

where $s_{(n-1)nf}$ refers to the ground arc which ends at node $n$, $s_{n(n+1)f}$ refers to the ground arc which starts from node $n$ (and ends at node $n + 1$).

Load flow constraints ensure the load is always conserved and added correctly at each location over a multi-leg journey:
\[
\sum_{i \in N} l_{x_{inf}} - \sum_{j \in N} l_{x_{njf}} + p_{s(n-1)nf} - p_{s(n+1)nf} + q_n = 0, \quad \forall n \in N, \forall f \in F,
\]

(11.18)

where \(l_{x_{inf}}\) refers to load variables associated with arc variables (legs or ground arcs) which end at node \(n\), \(l_{x_{njf}}\) refers to load variables which start from node \(n\). \(f\) refers to fleets in set \(F\). \(p_{s(n-1)nf}\) and \(p_{s(n+1)nf}\) are the loads associated with the ground variable \(s\) entering and leaving node \(n\) respectively. \(q_n\) is the load collected at node \(n\).

Load constraints ensure the vehicle maximum load is not exceeded. This is done by requiring that the loads for all non-depot variables be less than the vehicle maximum load, in this case, 200:

\[
l_x \leq 200, \quad \forall x \in X.
\]

(11.19)

The departure loads from the depot must be set to zero:

\[
l_{x_{djf}} = 0, \quad \forall j \in N, \forall f \in F, \forall d \in D_o,
\]

(11.20)

where \(D_o\) is the set of all nodes at the depot.

The demand constraints must ensure all locations \(u\) are visited once only:

\[
\sum_{u \in C, u \neq d} \sum_{f \in F} x_{iu_jmf} = 1, \quad \forall m \in C, m \neq d,
\]

(11.21)

where \(C\) is the set of locations and \(d\) refers to the depot location.

Timing constraints must ensure the time windows are complied with, that is, the LDT and EAT are met. Therefore the arrival legs need to be constrained by the LDTs as follows:

\[
T_n \leq LDT_u \quad \forall i \in N, \forall u \in C, \forall f \in F,
\]

(11.22)

where \(T_n\) is the time associated with the end node \(n\) for variable \(x\), and \(LDT_u\) is the time window start for location \(u\).
The departure legs need to be constrained by the EATs as follows:

\[ T_n \geq EAT_u x_{n,f} + TAT, \quad \forall u \in C, j \in N, f \in F, \quad (11.23) \]

where \( T_n \) is the time associated with the start node \( n \) for variable \( x_{u} \), and \( EAT_u \) is the time window end for the location \( u \). The turnaround time is \( TAT \) (10 for this instance).

Constraints are required to ensure vehicles start at the depot:

\[ s_{d_1(d_1+1)f} = 1, \quad \forall f \in F, \quad (11.24) \]

where \( s_{d_1(d_1+1)f} \) refers to the ground arc for fleet \( f \) at the start of the day at the depot.

No vehicles should start at any other location:

\[ s_{iu_1jf} = 0, \quad \forall f \in F, \forall u \in C, u \neq d, \forall j \in N, \quad (11.25) \]

where \( s_{iu_1jf} \) refers to the ground arc for fleet \( f \) at the start of the day at location \( u \), and \( d \) is the depot location.

Similarly, all vehicles must end the day at the depot:

\[ s_{(d_2-1)d_2f} = 1, \quad \forall u \in C, u \neq d, \forall j \in N, \forall f \in F, \quad (11.26) \]

where \( s_{(d_2-1)d_2f} \) refers to the ground arc for fleet \( f \) at the end of the day at the depot.

No vehicles should end the day at any other location:

\[ s_{iju_2f} = 0, \quad \forall f \in F, \forall u \in C, u \neq d, \forall i \in N \quad (11.27) \]

where \( s_{iju_2f} \) refers to the ground arc for fleet \( f \) at the end of the day at location \( u \).
If an arc is not used, the load should be set to zero:

\[ l_x \leq Mx, \quad \forall x \in X, \tag{11.28} \]

where M is “BigM” (10000).

Similarly, for ground arcs:

\[ p_s \leq Ms \quad \forall s \in S. \tag{11.29} \]

Initially, the problem was set up and solved without any load-related variables or constraints. It was noted that this produced feasible solutions. Therefore, it can be assumed that the variables generated and supplied to the problem already encouraged load feasibility. This is because they were generated from feasible routes. Also, the maximum vehicle load constraints were found to be relatively easy to satisfy for the instance at hand.

Therefore the extra variables and constraints which were designed to ensure maximum vehicle load feasibility were left out of the problem, resulting in a smaller formulation. The decision variables \( l \) and \( p \), as well as constraints (11.18), (11.19), (11.20), (11.28) and (11.29) were omitted during solving. If other instances of the problem are obtained and solved using this method, these constraints may need to be included, depending on the specific problem characteristics.

In the case that not all locations are visited for any benefit parameter used (i.e. the per location cost of the solution is too high), additional variables (legs or routes) must be added to the problem. This was done by including additional legs for every location not visited as follows:

- From the depot to the location, to arrive at EAT.
- From the location to the depot, to depart at LDT.
- To and from the \( n_{cl} \) closest locations, including both the earliest possible and latest possible trips. \( n_{cl} \) was varied as iterations progressed, between 10 (first iteration) and 30 (fifth iteration).
- To and from each of the other locations which were not visited in the previous iteration.
11.3.3 Observations, Results and Discussion

The best solution achieved was 1849 (14% from optimum solution =1620). The problem used 104970 variables and took 1145s to solve.

The problem is not constrained sufficiently to benefit substantially from this technique. A more traditional solution method will outperform this method.

It should also be noted that this method is less suitable for problems involving a single depot than it is for PDP types of problems. This is because the agent-generated routes are “greedy” in nature, and no mechanism considers the holistic situation at any point in time.

However, the method is still a useful and practical method for solving this problem. The performance could be improved by improving the agent routing code, and by improving the heuristics to insert additional legs between iterations. One idea to be considered would be to incorporate insertion heuristic logic such as that provided by Solomon [25] between iterations.

The technique holds the potential to perform as an adequate routing method for a wide range of difficult-to-solve, highly constrained vehicle routing problems.
12 Conclusions

1. Three methods of solving the airline taxi problem have been developed. Some of the most significant results are summarised in Table 9.4. These methods, which are enabled by the use of a number of innovative heuristics, are as follows:

(a) MCNFP with TDs: This is an exact method. However, since the model as developed created large numbers of variables, heuristics were designed to reduce the problem size. The aggregation and geographic heuristics designed reduced the number of variables to less than 200000 in some cases, making the model solvable.

(b) Agent variable generation with MCNFP: Here it was demonstrated that use of the aggregation heuristic, the geographic heuristic and the use of “agents” and random search can be used to create good variables to be used in an MCNFP model. Time discretisations were not used in the resulting MCNFP model.

(c) Composite variable generation and MCNFP with TDs: A significantly smaller multi-commodity network formulation was developed by firstly using the aggregation heuristic, then, secondly, by creating variables each of which inherently combines passenger network information with the fleet networks, leaving only fleet networks in the model.

2. Each modelling technique was found to have strengths and weaknesses. These will need to be considered by the airline when choosing the most appropriate modelling technique:

   • MCNFP with TDs
     - Produces many variables and therefore has slow convergence.
     - Allows for passenger groups to swap aircraft.
     - Cannot disallow passenger aircraft swapping.
– Allows the maximum amount of intra-journey ground waiting time to be adjusted as required, but with a significant processing time penalty.
– Produces good solutions; longer processing times equates to better solutions.

• Agent variable generation and MCNFP
  – Allows for passenger groups to swap aircraft.
  – Cannot disallow passenger aircraft swapping.
  – Allows the maximum amount of intra-journey ground waiting time to be adjusted as required.

• Composite variable generation and MCNFP
  – Does not allow for passenger groups to swap aircraft.
  – Cannot adjust intra-journey waiting time.
  – Can control the maximum number of legs any passenger group is subjected to.
  – Fast solution time.

3. A good solution to the Sefofane Air problem instance has been produced using the composite variable technique. The proposed method offers a 12% cost improvement over the manual schedule which was actually used on the day. The estimated saving over a year is R2.2m excluding scheduler manpower savings.

4. The composite variable construction technique has been shown to produce fast solutions to medium to large instances of another highly-constrained VRP, namely a real-world MVCVRP. In addition, the use of such variables greatly simplifies the ILP formulation.

5. There is much opportunity to apply both composite variable generation methods and agent-based variable generation methods to many other highly-constrained routing problems.

6. The addition of learning to the agent routing generation method by using, for example, neural network techniques, may speed up the process and improve the quality of solutions. Such trained agents would not need retraining when faced with future scheduling for any specific company, and could be programmed to learn continuously.

7. The method of generating more intelligent variables for a MVCVRP formulation is essentially about transferring processing from the solver to the model
preparation stage. This process gives the user much more control and flexibility over the setup of the model and makes it a simple matter to apply heuristics.

8. The agent routing method may provide the basis for a viable method for producing a solver capable of solving a wide range of VRPs, particularly highly constrained, real-world VRPs.

9. The agent routing method and the composite variable method could be combined for improved overall performance.

10. It is likely that some of the numerous construction and improvement heuristics that have been proposed over the years, such as those of Alfa [93], Potvin and Rousseau [38] or Diana and Dessouky [103], would be usable for this problem. Therefore future work could include investigating how these heuristics can be combined with the work presented here.
13 Recommendations

- The aggregation heuristic is applicable to airline taxi problems and other problems where there are terminals for pickup and delivery, e.g. shipping.

- The geographic heuristic is applicable to PDP problems in general. The analogous heuristic for traditional VRPs (with depots) is the shortest-route only heuristic, i.e. where only the shortest inter-customer trips are included in the problem formulation.

- The MCNF formulation can be used in combination with the aggregation and geographic heuristic to obtain high quality solutions for airline taxi problems of about 80 requests or less, and where customer vehicle swapping is allowable. Similarly, it should be applicable to general PDP problems.

- The agent routing method is a useful framework to apply in combination with heuristics to a wide range of vehicle routing problems. However, it is preferable that problems be highly constrained (e.g. real-world).

- The composite variable method is best applied to highly constrained problems and in combination with other heuristics. It will also perform well for larger problems, such as greater than 100 requests. It is a fast and effective method for the airline taxi problem where aircraft swapping is undesirable and quality of service must be high, i.e. few connecting flight legs.
REFERENCES


[83] Psaraftis H.N. Analysis of an $0-n^2$— heuristic for the single vehicle many-to-

[84] Psaraftis H.N. K-interchange procedures for local search in a precedence-


APPENDIX A  Sefofane Air Problem Data
<table>
<thead>
<tr>
<th>Booking No.</th>
<th>Name</th>
<th>From</th>
<th>To</th>
<th>Pax</th>
<th>EDT</th>
<th>LAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Auster 7nt</td>
<td>JAO</td>
<td>MUB</td>
<td>3</td>
<td>480</td>
<td>540</td>
</tr>
<tr>
<td>2</td>
<td>Bartenev</td>
<td>CBE</td>
<td>VUM</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>3</td>
<td>Bartenev Extra Seat</td>
<td>CBE</td>
<td>VUM</td>
<td>1</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>4</td>
<td>Beckerman</td>
<td>BBK</td>
<td>CBE</td>
<td>4</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>5</td>
<td>Beckerman</td>
<td>LVI</td>
<td>BBK</td>
<td>4</td>
<td>600</td>
<td>865</td>
</tr>
<tr>
<td>6</td>
<td>Bell</td>
<td>VUM</td>
<td>MUB</td>
<td>2</td>
<td>770</td>
<td>840</td>
</tr>
<tr>
<td>7</td>
<td>Bethoney 7nt</td>
<td>CBE</td>
<td>BBK</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>8</td>
<td>Bethoney 7nt</td>
<td>BBK</td>
<td>LVI</td>
<td>2</td>
<td>750</td>
<td>930</td>
</tr>
<tr>
<td>9</td>
<td>Boehmke/1</td>
<td>CBE</td>
<td>MUB</td>
<td>3</td>
<td>420</td>
<td>480</td>
</tr>
<tr>
<td>10</td>
<td>Boehmke/2</td>
<td>CBE</td>
<td>MUB</td>
<td>1</td>
<td>420</td>
<td>480</td>
</tr>
<tr>
<td>11</td>
<td>BOX#69(08)/2</td>
<td>SWI</td>
<td>BBK</td>
<td>5</td>
<td>420</td>
<td>560</td>
</tr>
<tr>
<td>12</td>
<td>BOX#71(08)/2</td>
<td>KWD</td>
<td>SWI</td>
<td>5</td>
<td>660</td>
<td>900</td>
</tr>
<tr>
<td>13</td>
<td>BOX#73(08)/2</td>
<td>BBK</td>
<td>KWD</td>
<td>2</td>
<td>600</td>
<td>900</td>
</tr>
<tr>
<td>14</td>
<td>Carthy</td>
<td>CBE</td>
<td>SLI</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>15</td>
<td>Christenson</td>
<td>CBE</td>
<td>OMD</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>16</td>
<td>Davis</td>
<td>LVI</td>
<td>BBK</td>
<td>2</td>
<td>600</td>
<td>930</td>
</tr>
<tr>
<td>17</td>
<td>Davis</td>
<td>BBK</td>
<td>ABU</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>18</td>
<td>Farley</td>
<td>MUB</td>
<td>TSO</td>
<td>2</td>
<td>770</td>
<td>930</td>
</tr>
<tr>
<td>19</td>
<td>Farley Extra Seat Luggage</td>
<td>MUB</td>
<td>TSO</td>
<td>1</td>
<td>770</td>
<td>925</td>
</tr>
<tr>
<td>20</td>
<td>Freeman Kalm</td>
<td>MUB</td>
<td>CTB</td>
<td>2</td>
<td>770</td>
<td>930</td>
</tr>
<tr>
<td>21</td>
<td>Gilels 7nt</td>
<td>MOM</td>
<td>MUB</td>
<td>2</td>
<td>770</td>
<td>840</td>
</tr>
<tr>
<td>22</td>
<td>Harris</td>
<td>XIG</td>
<td>BBK</td>
<td>2</td>
<td>610</td>
<td>750</td>
</tr>
<tr>
<td>23</td>
<td>Highbee Stacy</td>
<td>JAO</td>
<td>CTB</td>
<td>4</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>24</td>
<td>Holden</td>
<td>CTB</td>
<td>VUM</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>25</td>
<td>Keshavjee Party SOLE 208/1</td>
<td>ABU</td>
<td>MUB</td>
<td>6</td>
<td>540</td>
<td>930</td>
</tr>
<tr>
<td>26</td>
<td>Keshavjee Party SOLE 208/2</td>
<td>ABU</td>
<td>MUB</td>
<td>7</td>
<td>540</td>
<td>930</td>
</tr>
<tr>
<td>27</td>
<td>Kirby 7nt</td>
<td>CTB</td>
<td>JAO</td>
<td>3</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>28</td>
<td>Koeppel</td>
<td>CBE</td>
<td>MUB</td>
<td>2</td>
<td>790</td>
<td>880</td>
</tr>
<tr>
<td>29</td>
<td>Levinson 7nt</td>
<td>BBK</td>
<td>LVI</td>
<td>2</td>
<td>790</td>
<td>930</td>
</tr>
<tr>
<td>30</td>
<td>Levinson 7nt</td>
<td>MOM</td>
<td>BBK</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>31</td>
<td>Levy</td>
<td>MUB</td>
<td>MOM</td>
<td>2</td>
<td>770</td>
<td>930</td>
</tr>
<tr>
<td>32</td>
<td>Long</td>
<td>MUB</td>
<td>CBE</td>
<td>2</td>
<td>950</td>
<td>1070</td>
</tr>
<tr>
<td>33</td>
<td>Luecke</td>
<td>CBE</td>
<td>TSO</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>34</td>
<td>Mahul Shah</td>
<td>MUB</td>
<td>OMD</td>
<td>2</td>
<td>770</td>
<td>930</td>
</tr>
<tr>
<td>35</td>
<td>Marrero</td>
<td>XIG</td>
<td>MUB</td>
<td>2</td>
<td>485</td>
<td>540</td>
</tr>
<tr>
<td>36</td>
<td>McLaughlan</td>
<td>VUM</td>
<td>MUB</td>
<td>3</td>
<td>470</td>
<td>540</td>
</tr>
<tr>
<td>37</td>
<td>Merrill 7nt</td>
<td>MUB</td>
<td>JAO</td>
<td>4</td>
<td>765</td>
<td>930</td>
</tr>
<tr>
<td>38</td>
<td>Michaela</td>
<td>SHN</td>
<td>MOM</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>39</td>
<td>MILLER 7nt</td>
<td>MUB</td>
<td>VUM</td>
<td>2</td>
<td>770</td>
<td>930</td>
</tr>
<tr>
<td>40</td>
<td>NAS-Staff-Isaac</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>41</td>
<td>NAS-Staff-Joanne</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>42</td>
<td>NAS-Staff-July</td>
<td>HND</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>Booking No.</td>
<td>Name</td>
<td>From</td>
<td>To</td>
<td>Pax</td>
<td>EDT</td>
<td>LAT</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------</td>
<td>------</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>43</td>
<td>nas-staff-keowetse</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>44</td>
<td>NAS-Staff-Lebo</td>
<td>HND</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>45</td>
<td>NAS-Staff-Mike</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>46</td>
<td>NAS-Staff-Nelly</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>47</td>
<td>Nas-Staff-Sam</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>48</td>
<td>Pawliczek</td>
<td>VUM</td>
<td>MOM</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>49</td>
<td>Pradere</td>
<td>SLI</td>
<td>XIG</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>50</td>
<td>Ritkin</td>
<td>JAO</td>
<td>CBE</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>51</td>
<td>Saff-Ows-Agatha</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>52</td>
<td>Shappell 7nt</td>
<td>MOM</td>
<td>CBE</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>53</td>
<td>Shields</td>
<td>MUB</td>
<td>HND</td>
<td>2</td>
<td>750</td>
<td>930</td>
</tr>
<tr>
<td>54</td>
<td>Sollek &amp; McNally</td>
<td>MUB</td>
<td>CTB</td>
<td>2</td>
<td>770</td>
<td>930</td>
</tr>
<tr>
<td>55</td>
<td>Spilsbury</td>
<td>CTB</td>
<td>MUB</td>
<td>2</td>
<td>770</td>
<td>840</td>
</tr>
<tr>
<td>56</td>
<td>Staff-Flamingo-Lebopo</td>
<td>CTB</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>57</td>
<td>Staff-Flamingo-Simon</td>
<td>CTB</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>58</td>
<td>Staff-GREAT Explo-Goweditswe</td>
<td>MUB</td>
<td>XIG</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>59</td>
<td>Staff-Great Explo-OT</td>
<td>MUB</td>
<td>XIG</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>60</td>
<td>Staff-Linyanti Explo-Foster</td>
<td>BBK</td>
<td>SLI</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>61</td>
<td>Staff-Nas-Clara</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>62</td>
<td>STAFF-NAS-CLINT</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>63</td>
<td>Staff-Nas-Dona</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>64</td>
<td>Staff-Nas-Graeme</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>65</td>
<td>Staff-OWS-Baeti</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>66</td>
<td>Staff-OWS-Bashie</td>
<td>OMD</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>67</td>
<td>Staff-OWS-Boeti</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>68</td>
<td>Staff-Ows-Boitumelo</td>
<td>OMD</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>69</td>
<td>Staff-Ows-Bole</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>70</td>
<td>Staff-Ows-Bole</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>71</td>
<td>Staff-Ows-Bonolo</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>72</td>
<td>Staff-OWS-Camilla</td>
<td>VUM</td>
<td>MOM</td>
<td>1</td>
<td>660</td>
<td>1080</td>
</tr>
<tr>
<td>73</td>
<td>Staff-OWS-Celia</td>
<td>OMD</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>74</td>
<td>Staff-OWS-Changlu</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>75</td>
<td>Staff-Ows-Disho</td>
<td>VUM</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>76</td>
<td>Staff-ows-Dolly</td>
<td>VUM</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>77</td>
<td>Staff-Ows-Felicia</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>78</td>
<td>Staff-OWS-Gaba</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>79</td>
<td>Staff-OWS-Gale</td>
<td>SRA</td>
<td>OMD</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>80</td>
<td>Staff-OWS-Glorius</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>81</td>
<td>Staff-OWS-Joel</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>82</td>
<td>Staff-OWS-Johannes</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>83</td>
<td>Staff-OWS-John</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>84</td>
<td>Staff-OWS-John</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>Booking No.</td>
<td>Name</td>
<td>From</td>
<td>To</td>
<td>Pax</td>
<td>EDT</td>
<td>LAT</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>85</td>
<td>Staff-OWS-Kelemetse</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>86</td>
<td>Staff-OWS-Kelemetse</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>87</td>
<td>Staff-OWS-Kelly</td>
<td>SRA</td>
<td>OMD</td>
<td>1</td>
<td>360</td>
<td>930</td>
</tr>
<tr>
<td>88</td>
<td>Staff-Ows-Koolebogile</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>89</td>
<td>Staff-OWS-KG</td>
<td>MUB</td>
<td>SWI</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>90</td>
<td>Staff-Ows-Kgakgamatso</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>91</td>
<td>Staff-OWS-KK</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>92</td>
<td>Staff-OWS-KK</td>
<td>MUB</td>
<td>SWI</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>93</td>
<td>Staff-OWS-Koi</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>94</td>
<td>Staff-OWS-KP</td>
<td>SWI</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>95</td>
<td>Staff-OWS-Lalu</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>96</td>
<td>Staff-OWS-Landi</td>
<td>MUB</td>
<td>MOM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>97</td>
<td>Staff-Ows-Lebaho</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>98</td>
<td>Staff-Ows-Lebaho</td>
<td>VUM</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>99</td>
<td>Staff-Ows-Lesego</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>100</td>
<td>Staff-OWS-Letty</td>
<td>SRA</td>
<td>OMD</td>
<td>1</td>
<td>360</td>
<td>930</td>
</tr>
<tr>
<td>101</td>
<td>Staff-OWS-Lindi</td>
<td>SWI</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>102</td>
<td>Staff-OWS-Marriam</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>103</td>
<td>Staff-OWS-Martha</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>104</td>
<td>Staff-OWS-Martha</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>105</td>
<td>Staff-OWS-Mokopi</td>
<td>OMD</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>106</td>
<td>Staff-OWS-Mokopi</td>
<td>SRA</td>
<td>OMD</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>107</td>
<td>Staff-Ows-Lebaho</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>108</td>
<td>Staff-Ows-Motty</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>109</td>
<td>Staff-Ows-Nana</td>
<td>VUM</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>110</td>
<td>Staff-Ows-OB</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>111</td>
<td>Staff-Ows-Olatotswe</td>
<td>OMD</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>112</td>
<td>Staff-Ows-Ortell</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>113</td>
<td>Staff-Ows-Phetso</td>
<td>SRA</td>
<td>OMD</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>114</td>
<td>Staff-OWS-Rob</td>
<td>MUB</td>
<td>MOM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>115</td>
<td>Staff-OWS-Russel</td>
<td>JAO</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>116</td>
<td>Staff-OWS-Russel</td>
<td>MOM</td>
<td>JAO</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>117</td>
<td>Staff-OWS-Sadek</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>118</td>
<td>Staff-OWS-Sam</td>
<td>SWI</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>119</td>
<td>Staff-OWS-Smilely</td>
<td>OMD</td>
<td>SRA</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>120</td>
<td>Staff-Ows-Sylvia</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>121</td>
<td>Staff-OWS-Taps</td>
<td>MUB</td>
<td>MOM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>122</td>
<td>Staff-OWS-Thuto</td>
<td>MUB</td>
<td>BBK</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>123</td>
<td>Staff-Ows-Tirelo</td>
<td>MUB</td>
<td>SWI</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>124</td>
<td>Staff-OWS-TK</td>
<td>MUB</td>
<td>MOM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>125</td>
<td>Staff-Ows-Tshenyego M</td>
<td>SRA</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>126</td>
<td>Staff-Ows-Tsholetlego</td>
<td>SRA</td>
<td>OMD</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
</tbody>
</table>
Table A.2: Distance Matrix (kilometres)

<table>
<thead>
<tr>
<th>Booking No.</th>
<th>Name</th>
<th>From</th>
<th>To</th>
<th>PAZ</th>
<th>EDT</th>
<th>LAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>127</td>
<td>Staff-Ows-Tshubugo</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>128</td>
<td>Staff-Sable-Cathy</td>
<td>MUB</td>
<td>CBE</td>
<td>1</td>
<td>360</td>
<td>930</td>
</tr>
<tr>
<td>129</td>
<td>Staff-Sable-Lee</td>
<td>CBE</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>130</td>
<td>Staff-Sable-Raymond</td>
<td>CBE</td>
<td>MUB</td>
<td>1</td>
<td>660</td>
<td>1080</td>
</tr>
<tr>
<td>131</td>
<td>Staff-Sable-Rocky</td>
<td>MUB</td>
<td>CBE</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>132</td>
<td>Staff-Sefofane-Lekani</td>
<td>BBK</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>133</td>
<td>Staff-Sefofane-Modiri</td>
<td>CBE</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>134</td>
<td>Staff-Sefofane-Ollie</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>135</td>
<td>Staff-Sefofane-Tumo</td>
<td>BBK</td>
<td>MUB</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>136</td>
<td>Staff-Sefofane-Witness</td>
<td>MUB</td>
<td>VUM</td>
<td>1</td>
<td>360</td>
<td>1080</td>
</tr>
<tr>
<td>137</td>
<td>Stahl</td>
<td>SLI</td>
<td>BBK</td>
<td>4</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>138</td>
<td>Steel</td>
<td>MOM</td>
<td>MUB</td>
<td>2</td>
<td>780</td>
<td>840</td>
</tr>
<tr>
<td>139</td>
<td>Stewart 7nt</td>
<td>VUM</td>
<td>CBE</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>140</td>
<td>Thomas</td>
<td>LVI</td>
<td>BBK</td>
<td>2</td>
<td>600</td>
<td>865</td>
</tr>
<tr>
<td>141</td>
<td>Thomas</td>
<td>BBK</td>
<td>SLI</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>142</td>
<td>Thomas &amp; Lorenz</td>
<td>CTB</td>
<td>JAO</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>143</td>
<td>Trimble 7nt</td>
<td>VUM</td>
<td>TSO</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>144</td>
<td>World</td>
<td>MOM</td>
<td>MUB</td>
<td>2</td>
<td>780</td>
<td>840</td>
</tr>
<tr>
<td>145</td>
<td>World Journey Edu/1</td>
<td>HND</td>
<td>VUM</td>
<td>4</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>146</td>
<td>World Journeys Edu/2</td>
<td>HND</td>
<td>VUM</td>
<td>4</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>147</td>
<td>Wright</td>
<td>MUB</td>
<td>XIG</td>
<td>4</td>
<td>830</td>
<td>930</td>
</tr>
<tr>
<td>148</td>
<td>Yoshimoto 7nt</td>
<td>CBE</td>
<td>BBK</td>
<td>2</td>
<td>660</td>
<td>930</td>
</tr>
<tr>
<td>149</td>
<td>Zi-11(08)Guide-Francis</td>
<td>XOR</td>
<td>XIG</td>
<td>1</td>
<td>660</td>
<td>900</td>
</tr>
<tr>
<td>150</td>
<td>Zi-11(08)Macgregor</td>
<td>XOR</td>
<td>XIG</td>
<td>1</td>
<td>660</td>
<td>900</td>
</tr>
<tr>
<td>152</td>
<td>Z3-10(08)Guide Richard</td>
<td>BBK</td>
<td>CBE</td>
<td>1</td>
<td>810</td>
<td>895</td>
</tr>
<tr>
<td>153</td>
<td>Z3-10(08)Kauth</td>
<td>BBK</td>
<td>CBE</td>
<td>2</td>
<td>810</td>
<td>895</td>
</tr>
<tr>
<td>154</td>
<td>BOX#67(05/20)2008</td>
<td>MLO</td>
<td>VFA</td>
<td>12</td>
<td>815</td>
<td>870</td>
</tr>
<tr>
<td>155</td>
<td>BOX#69(05/20)08</td>
<td>VFA</td>
<td>MLO</td>
<td>12</td>
<td>735</td>
<td>795</td>
</tr>
<tr>
<td>156</td>
<td>BOX#69(08)/1</td>
<td>SWI</td>
<td>BBK</td>
<td>12</td>
<td>420</td>
<td>560</td>
</tr>
<tr>
<td>157</td>
<td>BOX#71(08)/1</td>
<td>KWD</td>
<td>SWI</td>
<td>12</td>
<td>660</td>
<td>900</td>
</tr>
<tr>
<td>158</td>
<td>BOX#73(08)/1</td>
<td>BBK</td>
<td>KWD</td>
<td>12</td>
<td>600</td>
<td>900</td>
</tr>
</tbody>
</table>
### Table A.3: Aircraft and Loadmaster Day Start and End Positions

<table>
<thead>
<tr>
<th>Position/Destination</th>
<th>No. of C206 at Day Start</th>
<th>No. of C206 at Day End</th>
<th>No. of C208 at Day Start</th>
<th>No. of C208 at Day End</th>
<th>No. of Loadmasters at Day Start</th>
<th>No. of Loadmasters at Day End</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUB</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>BBK</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VUM</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>HND</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>CBE</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>TSO</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SLI</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ABU</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>LVI</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table A.4: Airport Codes and Assigned Numbers

<table>
<thead>
<tr>
<th>Safari Camp Name</th>
<th>Airport Code</th>
<th>Assigned Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lechwe Island Camp</td>
<td>XOR</td>
<td>1</td>
</tr>
<tr>
<td>Xigera</td>
<td>XIG</td>
<td>2</td>
</tr>
<tr>
<td>Chobe/Kasane</td>
<td>BBK</td>
<td>3</td>
</tr>
<tr>
<td>Linyanti</td>
<td>CBE</td>
<td>4</td>
</tr>
<tr>
<td>Lianshulu</td>
<td>KWD</td>
<td>5</td>
</tr>
<tr>
<td>Moremi Tented Camp</td>
<td>SWI</td>
<td>6</td>
</tr>
<tr>
<td>Maun</td>
<td>MUB</td>
<td>7</td>
</tr>
<tr>
<td>Vumbura South</td>
<td>VUM</td>
<td>8</td>
</tr>
<tr>
<td>Jacana</td>
<td>JAO</td>
<td>9</td>
</tr>
<tr>
<td>Chitabe Trails</td>
<td>CTB</td>
<td>10</td>
</tr>
<tr>
<td>Little Mombo</td>
<td>MOM</td>
<td>11</td>
</tr>
<tr>
<td>Jack’s Camp</td>
<td>TSO</td>
<td>12</td>
</tr>
<tr>
<td>Seronga</td>
<td>SRA</td>
<td>13</td>
</tr>
<tr>
<td>Duba Plains</td>
<td>OMD</td>
<td>14</td>
</tr>
<tr>
<td>Shinde Camp</td>
<td>SHN</td>
<td>15</td>
</tr>
<tr>
<td>Selinda</td>
<td>SLI</td>
<td>16</td>
</tr>
<tr>
<td>Tubu Tree Camp</td>
<td>HND</td>
<td>17</td>
</tr>
<tr>
<td>River Club</td>
<td>LVI</td>
<td>18</td>
</tr>
<tr>
<td>Abu</td>
<td>ABU</td>
<td>19</td>
</tr>
<tr>
<td>Victoria Falls</td>
<td>VFA</td>
<td>20</td>
</tr>
<tr>
<td>Matetsi</td>
<td>MLO</td>
<td>21</td>
</tr>
</tbody>
</table>
APPENDIX B  Manual Schedule
Table B.1: Manual Schedule

<table>
<thead>
<tr>
<th>Aircraft Type</th>
<th>Origin</th>
<th>Departure Time</th>
<th>Destination</th>
<th>Arrival Time</th>
<th>Pax</th>
<th>Group Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>C206</td>
<td>7</td>
<td>14:30</td>
<td>9</td>
<td>15:05</td>
<td>4</td>
<td>Merrill 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>15:15</td>
<td>7</td>
<td>15:50</td>
<td>5</td>
<td>x NAS-Staff-Isaac, x nas-staff-keowetse, x NAS-Staff-Mike, x NAS-Staff-Nelly, x Nas-Staff-Sam</td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>8:30</td>
<td>7</td>
<td>9:00</td>
<td>2</td>
<td>x Marrero</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>10:45</td>
<td>8</td>
<td>11:30</td>
<td>1</td>
<td>x Staff-OWS-Martha</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>11:40</td>
<td>4</td>
<td>12:15</td>
<td>2</td>
<td>x Trimble 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>12:25</td>
<td>12</td>
<td>13:45</td>
<td>4</td>
<td>x Luecke &amp; x Trimble 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>12</td>
<td>13:55</td>
<td>7</td>
<td>14:50</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>17</td>
<td>8:10</td>
<td>9</td>
<td>8:15</td>
<td>1</td>
<td>x NAS-Staff-Lebo</td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>8:25</td>
<td>7</td>
<td>9:00</td>
<td>4</td>
<td>x Auster 7nt, x NAS-Staff-Lebo</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>13:00</td>
<td>12</td>
<td>13:55</td>
<td>3</td>
<td>x Farley, x Farley Extra Seat Luggage</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>7:30</td>
<td>6</td>
<td>7:45</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>8:00</td>
<td>3</td>
<td>9:15</td>
<td>5</td>
<td>x BOX#69(08)/2</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>10:00</td>
<td>5</td>
<td>11:05</td>
<td>2</td>
<td>x BOX#73(08)/2</td>
</tr>
<tr>
<td>C206</td>
<td>5</td>
<td>11:20</td>
<td>6</td>
<td>12:05</td>
<td>5</td>
<td>x BOX#71(08)/2</td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>12:15</td>
<td>10</td>
<td>12:25</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>12:35</td>
<td>9</td>
<td>13:00</td>
<td>3</td>
<td>x Kirby 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>13:10</td>
<td>10</td>
<td>13:35</td>
<td>4</td>
<td>x Higbee Stacy</td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>13:45</td>
<td>7</td>
<td>14:05</td>
<td>4</td>
<td>x Spilsbury, x Staff-Flamingo-Lebopo, x Staff-Flamingo-Simon</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>14:35</td>
<td>2</td>
<td>15:05</td>
<td>4</td>
<td>x Wright</td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>15:15</td>
<td>17</td>
<td>15:25</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>9:00</td>
<td>2</td>
<td>9:30</td>
<td>2</td>
<td>x Staff-GREAT Explo-Goweditswe, x Staff-Great Explo-OT</td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>9:40</td>
<td>3</td>
<td>11:15</td>
<td>2</td>
<td>x Harris</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>11:55</td>
<td>16</td>
<td>12:55</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>16</td>
<td>13:05</td>
<td>3</td>
<td>14:05</td>
<td>4</td>
<td>x Stahl</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>14:10</td>
<td>4</td>
<td>15:05</td>
<td>3</td>
<td>x Z3-10(08)Guide Richard, x Z3-10(08)Kauth</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>15:15</td>
<td>7</td>
<td>16:05</td>
<td>1</td>
<td>x Staff-Sefolane-Lekani</td>
</tr>
<tr>
<td>Aircraft Type</td>
<td>Origin</td>
<td>Departure Time</td>
<td>Destination</td>
<td>Arrival Time</td>
<td>Pax</td>
<td>Group Names</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
<td>----------------</td>
<td>-------------</td>
<td>--------------</td>
<td>-----</td>
<td>-------------</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>7:45</td>
<td></td>
<td>8:00</td>
<td>3</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>8:20</td>
<td></td>
<td>9:15</td>
<td>12</td>
<td>x Mobula, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>10:00</td>
<td></td>
<td>11:45</td>
<td>6</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>11:55</td>
<td></td>
<td>13:10</td>
<td>6</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>12:40</td>
<td></td>
<td>14:20</td>
<td>12</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>13:25</td>
<td></td>
<td>15:00</td>
<td>10</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>14:05</td>
<td></td>
<td>15:40</td>
<td>6</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>14:50</td>
<td></td>
<td>16:30</td>
<td>8</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>15:25</td>
<td></td>
<td>17:05</td>
<td>12</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>16:05</td>
<td></td>
<td>18:30</td>
<td>15</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>16:45</td>
<td></td>
<td>18:45</td>
<td>18</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>17:25</td>
<td></td>
<td>19:00</td>
<td>11</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>18:05</td>
<td></td>
<td>20:00</td>
<td>14</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>18:45</td>
<td></td>
<td>20:45</td>
<td>17</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>19:25</td>
<td></td>
<td>21:00</td>
<td>20</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>20:05</td>
<td></td>
<td>22:00</td>
<td>23</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>20:45</td>
<td></td>
<td>22:45</td>
<td>26</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>21:25</td>
<td></td>
<td>23:00</td>
<td>29</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>22:05</td>
<td></td>
<td>23:45</td>
<td>32</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
<tr>
<td>C208</td>
<td></td>
<td>22:45</td>
<td></td>
<td>24:30</td>
<td>35</td>
<td>x Staff-OWS-KG, x Staff-OWS-DO, x Staff-OWS-Sam</td>
</tr>
</tbody>
</table>

Note: x denotes the presence of a group member, and the number following the x indicates the number of group members.
<table>
<thead>
<tr>
<th>Aircraft Type</th>
<th>Origin</th>
<th>Departure Time</th>
<th>Destination</th>
<th>Arrival Time</th>
<th>Pax</th>
<th>Group Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>C208</td>
<td>7</td>
<td>10:35</td>
<td>20</td>
<td>11:50</td>
<td>1</td>
<td>x Staff-OWS-Thuto</td>
</tr>
<tr>
<td>C208</td>
<td>20</td>
<td>12:30</td>
<td>21</td>
<td>13:15</td>
<td>12</td>
<td>x BOX#69(05/20)08</td>
</tr>
<tr>
<td>C208</td>
<td>21</td>
<td>13:35</td>
<td>20</td>
<td>14:20</td>
<td>12</td>
<td>x BOX#67(05/20)2008</td>
</tr>
<tr>
<td>C208</td>
<td>20</td>
<td>15:00</td>
<td>3</td>
<td>15:20</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>10:15</td>
<td>1</td>
<td>11:00</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>1</td>
<td>11:10</td>
<td>2</td>
<td>11:35</td>
<td>3</td>
<td>x ZI-11(08)Guide-Francis , x ZI-11(08)Macgregor , x ZI-11(08)McLean</td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>11:45</td>
<td>11</td>
<td>11:50</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>12:00</td>
<td>3</td>
<td>13:30</td>
<td>2</td>
<td>x Levinson 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>9:50</td>
<td>8</td>
<td>11:15</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>11:25</td>
<td>4</td>
<td>12:00</td>
<td>2</td>
<td>x Stewart 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>12:10</td>
<td>3</td>
<td>13:05</td>
<td>4</td>
<td>x Bethoney 7nt , x Yoshimoto 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>13:45</td>
<td>18</td>
<td>14:10</td>
<td>4</td>
<td>x Bethoney 7nt , x Levinson 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>13:10</td>
<td>11</td>
<td>13:35</td>
<td>8</td>
<td>x Levy , x MILLER 7nt , x Shields , x Staff-OWS-Landi , x Staff-OWS-Rob</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>13:45</td>
<td>17</td>
<td>13:55</td>
<td>4</td>
<td>x MILLER 7nt , x Shields</td>
</tr>
<tr>
<td>C206</td>
<td>17</td>
<td>14:05</td>
<td>8</td>
<td>14:15</td>
<td>10</td>
<td>x MILLER 7nt , x World Jouney Edu/1 , x World Journeys Edu/2</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>14:25</td>
<td>17</td>
<td>14:35</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>10:25</td>
<td>15</td>
<td>10:55</td>
<td>2</td>
<td>x Staff-OWS-Taps , x Staff-OWS-TK</td>
</tr>
<tr>
<td>C206</td>
<td>15</td>
<td>11:05</td>
<td>11</td>
<td>11:20</td>
<td>4</td>
<td>x Michaela , x Staff-OWS-Taps , x Staff-OWS-TK</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>11:30</td>
<td>9</td>
<td>11:35</td>
<td>3</td>
<td>x Shappell 7nt , x Staff-OWS-Russel</td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>11:45</td>
<td>4</td>
<td>12:25</td>
<td>4</td>
<td>x Rifkin , x Shappell 7nt</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>12:35</td>
<td>19</td>
<td>13:25</td>
<td>4</td>
<td>x Davis , x Koeppe</td>
</tr>
<tr>
<td>C206</td>
<td>19</td>
<td>13:35</td>
<td>7</td>
<td>14:10</td>
<td>2</td>
<td>x Koeppe</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>15:45</td>
<td>4</td>
<td>16:35</td>
<td>4</td>
<td>x Long , x Staff-Sable-Cathy , x Staff-Sable-Rocky</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>16:45</td>
<td>16</td>
<td>16:50</td>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>Aircraft Type</td>
<td>Origin</td>
<td>Departure Time</td>
<td>Destination</td>
<td>Arrival Time</td>
<td>Pax</td>
<td>Group Names</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
<td>----------------</td>
<td>-------------</td>
<td>--------------</td>
<td>-----</td>
<td>-------------</td>
</tr>
<tr>
<td>C208</td>
<td>18</td>
<td>10:40</td>
<td>3</td>
<td>11:00</td>
<td>8</td>
<td>x Beckerman, x Davis, x Thomas</td>
</tr>
<tr>
<td>C208</td>
<td>3</td>
<td>11:40</td>
<td>4</td>
<td>12:20</td>
<td>9</td>
<td>x Beckerman, x Davis, x Staff-Linyanti Explo-Foster, x Thomas</td>
</tr>
<tr>
<td>C208</td>
<td>4</td>
<td>12:30</td>
<td>16</td>
<td>12:35</td>
<td>10</td>
<td>x Bartenev, x Bartenev Extra Seat, x Carthy, x Christenson, x Staff-Linyanti Explo-Foster, x Thomas</td>
</tr>
<tr>
<td>C208</td>
<td>16</td>
<td>12:45</td>
<td>8</td>
<td>13:05</td>
<td>7</td>
<td>x Bartenev, x Bartenev Extra Seat, x Christenson, x Pradere</td>
</tr>
<tr>
<td>C208</td>
<td>8</td>
<td>13:15</td>
<td>14</td>
<td>13:20</td>
<td>9</td>
<td>x Bell, x Christenson, x Pawliczek, x Pradere, x Staff-OWS-Camilla</td>
</tr>
<tr>
<td>C208</td>
<td>14</td>
<td>13:30</td>
<td>11</td>
<td>13:35</td>
<td>7</td>
<td>x Bell, x Pawliczek, x Pradere, x Staff-OWS-Camilla</td>
</tr>
<tr>
<td>C208</td>
<td>11</td>
<td>13:45</td>
<td>2</td>
<td>13:50</td>
<td>10</td>
<td>x Bell, x Gilels 7nt, x Pradere, x Steel, x Word</td>
</tr>
<tr>
<td>C208</td>
<td>2</td>
<td>14:00</td>
<td>7</td>
<td>14:25</td>
<td>8</td>
<td>x Bell, x Gilels 7nt, x Steel, x Word</td>
</tr>
</tbody>
</table>
APPENDIX C  Aggregation Heuristic Result
<table>
<thead>
<tr>
<th>Aggregated Group No.</th>
<th>Number of Passengers</th>
<th>Original Booking List Numbers</th>
<th>Booking Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>Auster 7nt</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2, 3</td>
<td>Bartenev, Bartenev Extra Seat</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>Beckerman1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>5,140</td>
<td>Beckerman2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>6</td>
<td>Bell</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>7, 148</td>
<td>Bethoney 7nt1, Yoshimoto 7nt</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>8</td>
<td>Bethoney 7nt2</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>9, 10</td>
<td>Boehmke1, Boehmke2</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>11</td>
<td>BOX69(08)/2</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>12</td>
<td>BOX71(08)/2</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>13</td>
<td>BOX73(08)/2</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>14</td>
<td>Carthy</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>15</td>
<td>Christenson</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>16</td>
<td>Davis</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>17</td>
<td>Davis</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>18,19</td>
<td>Farley, Farley Extra Seat Luggage</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
<td>20,54</td>
<td>Freeman Kalm, Sollek, McNally</td>
</tr>
<tr>
<td>18</td>
<td>6</td>
<td>21,138,21,144</td>
<td>Gilels 7nt, Steel, Word</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>22</td>
<td>Harris</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>23</td>
<td>Higbee Stacy</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>24</td>
<td>Holden</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
<td>25</td>
<td>Keshavjee Party SOLE 208/1</td>
</tr>
<tr>
<td>23</td>
<td>7</td>
<td>26</td>
<td>Keshavjee Party SOLE 208/2</td>
</tr>
<tr>
<td>24</td>
<td>5</td>
<td>27,142</td>
<td>Kirby 7ntm Thomas, Lorenz</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>28</td>
<td>Koeppel</td>
</tr>
<tr>
<td>26</td>
<td>2</td>
<td>29</td>
<td>Levinson 7nt</td>
</tr>
<tr>
<td>27</td>
<td>2</td>
<td>30</td>
<td>Levinson 7nt</td>
</tr>
<tr>
<td>28</td>
<td>2</td>
<td>31</td>
<td>Levy</td>
</tr>
<tr>
<td>29</td>
<td>2</td>
<td>32</td>
<td>Long</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
<td>33</td>
<td>Lueche</td>
</tr>
<tr>
<td>31</td>
<td>2</td>
<td>34</td>
<td>Mahul Shah</td>
</tr>
<tr>
<td>32</td>
<td>2</td>
<td>35</td>
<td>Marrero</td>
</tr>
<tr>
<td>33</td>
<td>3</td>
<td>36</td>
<td>McLachlan</td>
</tr>
<tr>
<td>34</td>
<td>4</td>
<td>37</td>
<td>Merrill 7nt</td>
</tr>
<tr>
<td>35</td>
<td>2</td>
<td>38</td>
<td>Michaela</td>
</tr>
<tr>
<td>36</td>
<td>3</td>
<td>39</td>
<td>MILLER 7nt</td>
</tr>
<tr>
<td>37</td>
<td>11</td>
<td>40,41,40,43,40, 45,40,46,40,47, 40,61,40,62,40, 63,40,64,40,115</td>
<td>NAS-Staff-Isaac, NAS-Staff-Joanne,  nas-staff-keowetse, NAS-Staff-Mike, NAS-Staff-Nelly,  Nas-Staff-Sam, Staff-Nas-Clara, STAFF-NAS-CLINT, Staff-Nas-Dona, Staff-Nas-Graeme, Staff-OWS-Russel</td>
</tr>
<tr>
<td>38</td>
<td>2</td>
<td>42,44</td>
<td>NAS-Staff-July, NAS-Staff-Lebo</td>
</tr>
<tr>
<td>39</td>
<td>2</td>
<td>48</td>
<td>Pawliczek</td>
</tr>
<tr>
<td>40</td>
<td>2</td>
<td>49</td>
<td>Pradere</td>
</tr>
<tr>
<td>41</td>
<td>2</td>
<td>50</td>
<td>Rifkin</td>
</tr>
<tr>
<td>42</td>
<td>11</td>
<td>51,70,51,71,51, 81,51,86,51,88, 51,90,51,95,51, 99,51,103,51,104</td>
<td>Saff-Ows-Agatha, Staff-Ows-Bole, Staff-Ows-Bonolo, Staff-OWS-Joel, Staff-OWS-Keleemetse, Staff-Ows-Keolebogile , Staff-Ows-Kgakgamatsa, Staff-OWS-Lalu,Staff-Ows-Lesego, Staff-OWS-Martha, Staff-OWS-Martha</td>
</tr>
<tr>
<td>43</td>
<td>2</td>
<td>52</td>
<td>Shappell 7nt</td>
</tr>
<tr>
<td>44</td>
<td>2</td>
<td>53</td>
<td>Shields</td>
</tr>
<tr>
<td>Group No.</td>
<td>Number of Passengers</td>
<td>Original Booking List Numbers</td>
<td>Booking Names</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------</td>
<td>-----------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>45</td>
<td>2</td>
<td>55</td>
<td>Spilsbury</td>
</tr>
<tr>
<td>46</td>
<td>2</td>
<td>56,57</td>
<td>Staff-Flamingo-Lebopo, Staff-Flamingo-Simon</td>
</tr>
<tr>
<td>47</td>
<td>2</td>
<td>58,59</td>
<td>Staff-GREAT Explo-Goweditswe, Staff-Great Explo-OT</td>
</tr>
<tr>
<td>48</td>
<td>1</td>
<td>60</td>
<td>Staff-Linyanti Explo-Foster</td>
</tr>
<tr>
<td>49</td>
<td>10</td>
<td>65,67,69,74,65,77,65,78,65,82,65,83,65,84,65,93,65,109</td>
<td>Staff-OWS-Baelti, Staff-OWS-Boh, Staff-OWS-Changi, Staff-Ows-Felicia, Staff-OWS-Gaba, Staff-OWS-Johannes, Staff-OWS-John, Staff-OWS-John, Staff-OWS-Koi, Staff-Ows-Nana</td>
</tr>
<tr>
<td>50</td>
<td>6</td>
<td>66,68,66,73,66,105,66,111,66,119</td>
<td>Staff-OWS-Bashie, Staff-Ows-Boitumelo, Staff-OWS-Celia, Staff-OWS-Mokopi, Staff-Ows-Olatotswe, Staff-OWS-Smiley</td>
</tr>
<tr>
<td>51</td>
<td>11</td>
<td>69,80,69,85,69,91,69,79,69,102,69,107,69,108,69,110,69,117,69,125</td>
<td>Staff-Ows-Bole, Staff-OWS-Glorius, Staff-OWS-Kelemetse, Staff-OWS-KK, Staff-Ows-Lebako, Staff-OWS-Marriam, Staff-OWS-Motsumi, Staff-OWS-Motty, Staff-OWS-OB, Staff-OWS-Sadek, Staff-OWS-Tshenyego M</td>
</tr>
<tr>
<td>52</td>
<td>1</td>
<td>72</td>
<td>Staff-OWS-Camilla</td>
</tr>
<tr>
<td>53</td>
<td>3</td>
<td>75,76,75,98</td>
<td>Staff-Ows-Disho, Staff-ows-Dolly, Staff-Ows-Lebasho</td>
</tr>
<tr>
<td>54</td>
<td>4</td>
<td>79,106,79,113,79,126</td>
<td>Staff-OWS-Gale, Staff-OWS-Mokopi, Staff-OWS-Phetso, Staff-OWS-Tshotelego</td>
</tr>
<tr>
<td>55</td>
<td>2</td>
<td>87,100</td>
<td>Staff-OWS-Kelly, Staff-OWS-Letty</td>
</tr>
<tr>
<td>56</td>
<td>3</td>
<td>89,92,89,123</td>
<td>Staff-OWS-KG, Staff-OWS-KK, Staff-Ows-Tirelo</td>
</tr>
<tr>
<td>57</td>
<td>3</td>
<td>94,101,94,118</td>
<td>Staff-OWS-KP, Staff-OWS-Lindi, Staff-OWS-Sam</td>
</tr>
<tr>
<td>58</td>
<td>4</td>
<td>96,114,96,121,96,124</td>
<td>Staff-OWS-Landi, Staff-OWS-Rob, Staff-OWS-Taps, Staff-OWS-TK</td>
</tr>
<tr>
<td>59</td>
<td>5</td>
<td>112,120,112,127,112,134,112,136,13</td>
<td>Staff-Ows-Ortell, Staff-Ows-Sylvia, Staff-Ows-Tshubugo, Staff-Sefofane-Ollie, Staff-Sefofane-Witness</td>
</tr>
<tr>
<td>60</td>
<td>1</td>
<td>116</td>
<td>Staff-OWS-Russel</td>
</tr>
<tr>
<td>61</td>
<td>1</td>
<td>122</td>
<td>Staff-OWS-Thuto</td>
</tr>
<tr>
<td>62</td>
<td>1</td>
<td>128</td>
<td>Staff-Sable-Cathy</td>
</tr>
<tr>
<td>63</td>
<td>2</td>
<td>129,133</td>
<td>.129,133</td>
</tr>
<tr>
<td>64</td>
<td>1</td>
<td>130</td>
<td>Staff-Sable-Raymond</td>
</tr>
<tr>
<td>65</td>
<td>1</td>
<td>131</td>
<td>Staff-Sable-Rocky</td>
</tr>
<tr>
<td>66</td>
<td>2</td>
<td>132,135</td>
<td>Staff-Sefofane-Lekani, Staff-Sefofane-Tumo</td>
</tr>
<tr>
<td>67</td>
<td>4</td>
<td>137</td>
<td>Stahl</td>
</tr>
<tr>
<td>68</td>
<td>2</td>
<td>139</td>
<td>Stewart 7nt</td>
</tr>
<tr>
<td>69</td>
<td>2</td>
<td>141</td>
<td>Thomas</td>
</tr>
<tr>
<td>70</td>
<td>2</td>
<td>143</td>
<td>Trimble 7nt</td>
</tr>
<tr>
<td>71</td>
<td>8</td>
<td>145,146</td>
<td>World Jouney Edu/1, World Jounes Edu/2</td>
</tr>
<tr>
<td>72</td>
<td>4</td>
<td>147</td>
<td>Wright</td>
</tr>
<tr>
<td>73</td>
<td>3</td>
<td>149,150,149,151</td>
<td>Z1-11(08)Guide-Francis, Z1-11(08)Macgregor, Z1-11(08)McLean</td>
</tr>
<tr>
<td>74</td>
<td>3</td>
<td>152,153</td>
<td>Z3-10(08)Guide Richard, Z3-10(08)Kauth</td>
</tr>
<tr>
<td>75</td>
<td>12</td>
<td>154</td>
<td>BOX67(05/20)2008</td>
</tr>
<tr>
<td>76</td>
<td>12</td>
<td>155</td>
<td>BOX69(05/20)08</td>
</tr>
<tr>
<td>77</td>
<td>12</td>
<td>156</td>
<td>BOX69(08)/1</td>
</tr>
<tr>
<td>78</td>
<td>12</td>
<td>157</td>
<td>BOX71(08)/1</td>
</tr>
<tr>
<td>79</td>
<td>12</td>
<td>158</td>
<td>BOX73(08)/1</td>
</tr>
</tbody>
</table>
APPENDIX D  Agent Routing MATLAB
Code - Airline Taxi Problem

D.1 Attractiveness Parameters

Fac1(1,1)=30; % time urgency factor 1
Fac1(2,1)=4; % number of on board groups with destinations matching candidate location
Fac1(3,1)=30; % time closeness
Fac1(4,1)=50; % geographic closeness
Fac1(5,1)=4; % on-board group destinations match awaiting group destinations
Fac1(6,1)=0.5; % if subsequent collected groups have the same destination as previously collected groups
Fac1(7,1)=15; % take off and leave any other waiting groups (selection of groups to collect)
Fac1(8,1)=20; % time urgency factor 2 - for on-board groups
Fac1(9,1)=1/20; % awaiting group size

D.2 Agent Routing Function

function [f u]=Fn-Agent_Fixed_V2(Fac1)

%Fac1 is a vector of attractiveness factors

global sch
global fleet
global fleetType
global dest

Fix1=1; % make best decisions only
NIters=1;
Penalty=10000; %cost for every undelivered item

Fac1=Fac1';
u=0;
\[ r1 \times c1 = \text{size}(Fac1); \]
for \( it = 1: r1 \)
  if \( Fac1(it, 1) < 0.01 \)
    \( Fac1(it, 1) = 0.01; \)
  end
end

DayStart = 360;

TAT = 10;

NCities = 21;
\[ \text{[NItems} \times c1 = \text{size}(sch); \]
\[ \text{[NFleets} \times c1 = \text{size}(fleet); \]
fleet = [fleet(:, 1:4) zeros(NFleets, 1)];

AvgSum = 0;
CloseArr = zeros(1, 6);
CACnt = 0;

for ppn = 1:NIters
  \( \text{NAnt} = 1; \)
  Route = zeros(1, 20);
  Routing = Route;
sch = sch(:, 1:5);
sch = [sch zeros(NItems, 2)];
fleet = [fleet(:, 1:4) zeros(NFleets, 1)];

  StickCnt = 0;
  SSCn = 0;
  NRemFleets = NFleets;
  CompleteItems = 0;

  \% start schedule
  SOut = 0;
  while SOut == 0
    \% choose fleet - Choose with uniform random Monte Carlo sampling, more weighting to larger aircraft
    out1 = 0;
    i = 0;
    Rnd1 = rand(1, 1);
    sum1 = 0;
    Max1 = 0;
    for \( k = 1: \)NFleets
      if \( \text{fleet}(k, 5) == 0 \)
sum1=sum1+fleetType(fleet(k,2),3);
if fleetType(fleet(k,2),3)>Max1
    Max1=fleetType(fleet(k,2),3);
end
end
cum1=0;
while out1==0
    i=i+1;
    if fleet(i,5)==0
        if Fix1==0
            cum1=cum1+fleetType(fleet(i,2),3)*(1-fleet(i,5));
            if Rnd1<=cum1/sum1
                out1=1;
            end
        else
            if Max1==fleetType(fleet(i,2),3)
                out1=1;
            end
        end
    else
        if Max1==fleetType(fleet(i,2),3)
            out1=1;
        end
    end
end
end

ant(1,1)=i;
Flt1=i;
NRemFleets=NRemFleets-1;
fleet(i,5)=1;

CurrCity=fleet(ant(1,1),3);
CurrTime=DayStart;

FltSpd=fleetType(fleet(ant(1,1),2),2);
EmptySeats=fleetType(fleet(ant(1,1),2),3);

NOnBrd=0;
OnBrd=[];

%record route
Route(1,1)=CurrCity;
Route(1,2)=DayStart;
Route(1,3)=EmptySeats;
Route(1,5)=NOnBrd;
Route(1,4)=Flt1;
for pp=1:NOnBrd
Route(1,5+pp)=OnBrd(pp,6);
end

%step ant through day
NOut=0;
StepCnt=1;
while NOut==0
    %choose next destination
    Att=zeros(NCities,2);
    Choose1=zeros(NCities,1);
    ArrivalTime=Choose1;
    CanPick=zeros(NCities,50);    % grp number is j,1
    CanPickAttr=CanPick;
    MustGo=zeros(NCities,1);

    for j=1:NCities
        %Attractiveness:
        %distance
        Dist1=distance2(dest,CurrCity,j);
        FltTime=Dist1/FltSpd*60;
        ArrivalTime(j,1)=CurrTime+FltTime;
        % Geographic closeness
        Close3= 1 / exp(Dist1 / Fac1(4,1));

        %On-board grps
        ReasonFlag=0;
        NOnBrdDest=0;
        TimeClose=0;

        for i=1:NOnBrd
            if OnBrd(i,4)==j
                ReasonFlag=1;
                Att(j,1)=1;
                NOnBrdDest=NOnBrdDest+1;

                %how soon must they arrive at dest?
                t1=ArrivalTime(j,1);
                t2=OnBrd(i,2);
                TU=1 / exp((abs(t2 - t1) / Fac1(1,1)));
                % TU=TimeUrgency(Sch(OnBrd(i,1),2),ArrivalTime(j,1);
                TimeClose=TimeClose+TU;

                if (t2-t1)<TU*Fac1(8,1)
                    MustGo(j,1)=1;
                end
            end
        end
    end
151
end

Close1=NOnBrdDest/Fac1(2,1);
if Close1>1
    Close1=1;
end

% Are there waiting grps?
out2=0;
cnt2=0;
Close5=0;
Close2s=0;
Close4s=0;
Close6s=0;

% CanPick=[]; % attractiveness is j,2
CP=1;
SumAttr=0;
for i=1:NItems
    if EmptySeats>0 && sch(i,6)==0 && NOnBrd>0
        stop1=1;
    end

    if sch(i,6)==0 && sch(i,3)==j && EmptySeats>=sch(i,5) % not already picked up, correct city
        % Time to collect
        if ArrivalTime(j,1)>= sch(i,1) % arrive after EAT
            CollectTime=ArrivalTime(j,1);
        else
            CollectTime=sch(i,1);
        end

        % Can this awaiting group and all on-brd grps be delivered on time?
        CanDo=CheckD2(CollectTime,j,i,Flt1, TAT, sch, fleetType, fleet, NOnBrd, OnBrd, dest);
        if CanDo==1 % feasible
            ReasonFlag=1;
            Att(j,2)=1;
            CP=CP+1;
            CanPick(j,1)=CanPick(j,1)+1;
            CanPick(j,CP)=i;

            % how attractive?

            % Time closeness
            t1=ArrivalTime(j,1);
            t2=CollectTime;

            SumAttr=SumAttr+1;
    end
end
TU=1 / exp((abs(t2 - t1) / Fac1(3,1)))
Close2=TU;
%
dest. of waiting grp = on brd grp destinations
OBDMatchCnt=0;
cnt9=0;
out9=0;
for k=1:NOnBrd
    if OnBrd(k,4)==sch(i,4)
        OBDMatchCnt=OBDMatchCnt+1;
    end
end
Close4=OBDMatchCnt/Fac1(5,1);
if Close4>1
    Close4=1;
end
%
size - make larger groups more attractive
Close6=sch(i,5)*Fac1(9,1);
xx=Close2+Close4+Close6;
Close5=Close5+xx;
CanPickAttr(j,CP)=xx;
SumAttr=SumAttr+xx;
Close2s=Close2s+Close2;
Close4s=Close4s+Close4;
Close6s=Close6s+Close6;
end
end

if ReasonFlag==1
    Choose1(j,1)=Close1+Close3+Close5+TimeClose;
else
    Choose1(j,1)=0;
end
CACnt=CACnt+1;
CloseArr(CACnt,1)=Close1;
CloseArr(CACnt,2)=Close3;
CloseArr(CACnt,3)=TimeClose;
CloseArr(CACnt,4)=Close2s;
CloseArr(CACnt,5)=Close4s;
CloseArr(CACnt,6)=Close6s;
CloseArr(CACnt,7)=Close5;
%if urgent matter:
Filter11=0;
for j=1:NCities
    if MustGo(j,1)==1
        NextCity=j;
        Filter11=1;
    end
end
%
if Filter11==0
    AllZeros=1;
    for k=1:NCities
        if Choose1(k,1)>0.01
            AllZeros=0;
        end
    end
end

if AllZeros==0
%
Monte Carlo - sample next city to go to
Rnd1=rand(1,1);
sum1=0;
Max1=0;
for j=1:NCities
    sum1=sum1+Choose1(j,1);
    if Choose1(j,1)>Max1
        Max1=Choose1(j,1);
    end
end
out2=0;
cnt2=0;
cum2=0;
while out2==0
    cnt2=cnt2+1;
    if Fix1==0
        cum2=cum2+Choose1(cnt2,1);
        if Rnd1<=cum2/sum1
            NextCity=cnt2;
        end
    end
end
out2=1;
cnt2=cnt2+1;
if cnt2>NCities
  out2=1;
end
delay = 1;
else
  if Choose1(cnt2,1)==Max1
    NextCity=cnt2;
    out2=1;
  end
end
end
delay = 1;

if AllZeros==0 || Filter11==1 % there is somewhere to go

  % move to next city
  StepCnt=StepCnt+1;
  % record route
  Route(StepCnt,1)=NextCity;
  Route(StepCnt,2)=ArrivalTime(NextCity,1);
  Route(StepCnt,3)=EmptySeats;
  Route(StepCnt,5)=NOnBrd;
  Route(StepCnt,4)=Flt1;
  for pp=1:NOnBrd
    Route(StepCnt,5+pp)=OnBrd(pp,6);
  end

  if CurrCity~=NextCity
    CurrTime=ArrivalTime(NextCity,1)+TAT;
  end
  CurrCity=NextCity;

  % are grps dropped off?
  Dropped=0;
  if Att(CurrCity,1)==1 % yes
    Dropped=1;
    DelCnt=0;
    for i=1:NOnBrd
      if OnBrd(i,4)==CurrCity
        % remove group
        DelCnt=DelCnt+1;
        DelGrp=OnBrd(i,6);
        DelItem(DelCnt,1)=i;
        sch(DelGrp,7)=1;
      end
    end
  end

  % are grps picked up?
  Picked=0;
  if Pick(CurrCity,1)==1 % yes
    Picked=1;
    PickCnt=0;
    for i=1:NOnBrd
      if OnBrd(i,4)==CurrCity
        % remove group
        PickCnt=PickCnt+1;
        PickGrp=OnBrd(i,6);
        PickItem(PickCnt,1)=i;
        sch(PickGrp,7)=1;
      end
    end
  end

  % check for fullness
  if EmptySeats==0
    % drop groups
    for i=1:NOnBrd
      if OnBrd(i,5)==1
        % remove group
        DelCnt=DelCnt+1;
        DelGrp=OnBrd(i,6);
        DelItem(DelCnt,1)=i;
        sch(DelGrp,7)=1;
      end
    end
  end

  % check for overbooked
  if NOnBrd==NSeat
    % overbooked
    % drop groups
    for i=1:NOnBrd
      if OnBrd(i,5)==1
        % remove group
        DelCnt=DelCnt+1;
        DelGrp=OnBrd(i,6);
        DelItem(DelCnt,1)=i;
        sch(DelGrp,7)=1;
      end
    end
  end

  % check for ending
  if CurrCity==EndCity
    % end
  end

end

155
for \( ij = 1: \text{DelCnt} \)
\[
\text{OnBrd}(\text{DelItem}(ij,1),8) = 1;
\]
%update schedule

%update empty seats
\[
\text{EmptySeats} = \text{EmptySeats} + \text{sch}(\text{DelGrp},5);
\]
end

Dum1 = [];
cnt111 = 0;
for \( ij = 1: \text{NOnBrd} \)
if \( \text{OnBrd}(ij,8) = 0 \)
\[
cnt111 = cnt111 + 1;
\]
\[
\text{Dum1}(cnt111,:) = \text{OnBrd}(ij,:);
\]
end
end
\[
\text{OnBrd} = \text{Dum1};
\]
\[
\text{NOnBrd} = \text{NOnBrd} - \text{DelCnt};
\]
\[
\text{CompleteItems} = \text{CompleteItems} + \text{DelCnt};
\]
end

%are grps picked up?
\[
\text{T0Attr} = 0;
\]
if \( \text{Att}(\text{CurrCity},2) = 1 \) % yes
%randomly select grps to collect from CanPick list until plane %is full
%then move time forward till then
\[
\text{TimeL} = \text{zeros}(11,1);
\]
\[
\text{AddG} = 0;
\]
\[
\text{AttrBoost} = \text{zeros}(\text{NCities},1);
\]
\[
\text{TList} = [];
\]
\[
\text{GList} = [];
\]
out6 = 0;
while out6 == 0
\[
\text{NPick} = \text{CanPick}(\text{CurrCity},1);
\]
%boost attr for same destinations of grps put on previously
for \( g = 1: \text{NPick} \)
\[
\text{for } k = 1: \text{NItems}
\]
if \( \text{CanPick}(\text{CurrCity},g+1) = k \)
\[
\text{if } k = \text{sch}(\text{CanPick}(\text{CurrCity},g+1),4)
\]
\[
\text{CanPickAttr}(\text{CurrCity},1+g) = \text{CanPickAttr}(\text{CurrCity},1+g) + \text{AttrBoost}(k,1) * \text{Fac1}(6,1);
\]
end
end
e
156
\texttt{AttrBoost(k,1)=0;}
\texttt{end}

\%remove attractiveness of grps that can't fit
\texttt{if sch(CanPick(CurrCity,g+1),5)>EmptySeats}
\texttt{CanPickAttr(CurrCity,1+g)=0;}
\texttt{end}
\texttt{end}
\texttt{end}

\%add attractiveness to just take off - only if grps were
\%dropped off
\texttt{if Dropped==1}
\texttt{\%min waiting time for grps to collect}
\texttt{Nt1=CurrTime;}
\texttt{if AddG>0}
\texttt{for k=1:AddG}
\texttt{\%TList(k,1)>Nt1}
\texttt{Nt1=TList(k,1);}
\texttt{end}
\texttt{end}
\texttt{MinTOTime=2000;}
\texttt{for g=1:NPick}
\texttt{\%sch(CanPick(CurrCity,g+1),1)<MinTOTime}
\texttt{MinTOTime=sch(CanPick(CurrCity,g+1),1);}
\texttt{end}
\texttt{end}
\texttt{TWait=MinTOTime-Nt1;}
\texttt{if TWait<0}
\texttt{TWait=0;}
\texttt{end}
\texttt{TOAttr=1-1/exp(TWait/Fac1(7,1));}
\texttt{if TOAttr<=0.01}
\texttt{TOAttr=0.01;}
\texttt{end}
\texttt{end}
\texttt{Max1=0;}
\texttt{sum1=0; \%Monte Carlo sample waiting grps}
\texttt{for g=1:NPick}
\texttt{sum1=sum1+CanPickAttr(CurrCity,1+g);}
\texttt{if CanPickAttr(CurrCity,1+g)>Max1}
\texttt{Max1=CanPickAttr(CurrCity,1+g);}
\texttt{end}
\texttt{end}
% if sum1<0.01
%    TOAttr=1;
% end

if sum1>0.01 %there are grps to collect that can fit
    sum1=sum1+TOAttr;
    SSCn=SSCn+1;
    if SSCn>500
        stop1=1;
    end

    out8=0;
    cnt8=0;
    cum1=0;

    Rnd1=rand(1,1);
    while out8==0
        cnt8=cnt8+1;
        if Fix1==0
            if cnt8<=NPick
                cum1=cum1+CanPickAttr(CurrCity,cnt8+1);
            else
                cum1=cum1+TOAttr;
                out8=1;
            end
            if Rnd1<=cum1/sum1
                out8=1;
            end
        else
            if Max1==CanPickAttr(CurrCity,1+cnt8)
                out8=1;
            end
        end
    end

    if cnt8<=NPick %going to pick up a group
        PGrp=CanPick(CurrCity,cnt8+1);
        %check that dupl grp not being put on brd
        find1=0;
        for i=1:NOnBrd
            if PGrp==OnBrd(i,6)
                find1=1;
        end
        ***
%remove grp from CanPick list
CanPick(CurrCity,1)=CanPick(CurrCity,1)-1;
NPick=CanPick(CurrCity,1);
out8=0;
cnt8=1;
while out8==0
    cnt8=cnt8+1;
    if CanPick(CurrCity,cnt8)==PGrp
        CanPick(CurrCity, cnt8)=0;
        for i=cnt8:NPick+1
            CanPick(CurrCity,i)=CanPick(CurrCity,i+1);
            CanPickAttr(CurrCity,i)=CanPickAttr(CurrCity,i+1);
        end
        out8=1;
    end
end
CanPick(CurrCity,i+1)=0;
CanPickAttr(CurrCity,i+1)=0;
NPick=CanPick(CurrCity,1);

%***

end
end

%check that all grps can still be delivered on time
TimeToCollect=sch(PGrp,1);
CanDo=CheckD2(TimeToCollect, CurrCity, PGrp, Flt1, TAT, sch, fleetType, fleet, NOnBrd, OnBrd, dest, TToCollect, City1, GrpToGo, flt1, TATime, sch1, fleetType1, fleet1, NOnBrd1, OnBrd1, dest1);
if CanDo==0
    find1=1;

%remove grp from CanPick list
CanPick(CurrCity,1)=CanPick(CurrCity,1)-1;
NPick=CanPick(CurrCity,1);
out8=0;
cnt8=1;
while out8==0
    cnt8=cnt8+1;
    if cnt8>30
        stop1=1;
    end
    if CanPick(CurrCity,cnt8)==PGrp
        CanPick(CurrCity, cnt8)=0;
        for i=cnt8:NPick+2
            CanPick(CurrCity,i)=CanPick(CurrCity,i+1);
        end
end
CanPick(CurrCity,1)=CanPick(CurrCity,1+1);
CanPickAttr(CurrCity,i)=CanPickAttr(CurrCity,i+1);
end
out8=1;
end
end
NPick=CanPick(CurrCity,1);
end

if find1==0 % put grp on board
EmptySeats=EmptySeats-sch(PGrp,5);
NOnBrd=NOnBrd+1;
OnBrd=[OnBrd; sch(PGrp,1:5) PGrp 0 0];
sch(PGrp,6)=1;
AddG=AddG+1;
TList(AddG,1)=sch(PGrp,1); % save EDT
GList(AddG,1)=PGrp;

%boost attractiveness for grps going to same dest as this %one:
AttrBoost(sch(PGrp,4),1)=AttrBoost(sch(PGrp,4),1)+1;

%remove grp from CanPick list
CanPick(CurrCity,1)=CanPick(CurrCity,1)-1;
NPick=CanPick(CurrCity,1);
out8=0;
cnt8=1;
while out8==0
  cnt8=cnt8+1;
  if CanPick(CurrCity,cnt8)==PGrp
    CanPick(CurrCity, cnt8)=0;
    for i=cnt8:NPick+1
      CanPick(CurrCity,i)=CanPick(CurrCity,i+1);
      CanPickAttr(CurrCity,i)=CanPickAttr(CurrCity,i+1);
    end
    out8=1;
  end
end
CanPick(CurrCity,i+1)=0;
CanPickAttr(CurrCity,i+1)=0;
NPick=CanPick(CurrCity,1);

%add attractiveness to just take off
Dropped=1;

end %putting grp on brd
else
    %have selected to just take off
    out6=1;
end

else
    %no more grps to take on brd
    out6=1;
end
end

%move time forward
TOTime=CurrTime;
for k=1:AddG
    if sch(GList(AddG,1),1)>TOTime
        TOTime=sch(GList(AddG,1),1);
    end
end

CurrTime=TOTime;

end

end %picking up grps

StepCnt=StepCnt+1;
Route(StepCnt,3)=EmptySeats;
Route(StepCnt,1)=CurrCity;
Route(StepCnt,2)=CurrTime;
Route(StepCnt,5)=NOnBrd;
Route(StepCnt,4)=Flt1;
for pp=1:NOnBrd
    Route(StepCnt,5+pp)=OnBrd(pp,6);
end

else
    %nowhere to go - all cities zero attractiveness
    %new ant
    NOut=1;

    [r1 c1]=size(Routing);
    [r2 c2]=size(Route);
    cntn=0;
    for kj=r1+1:r1+r2
        cntn=cntn+1;
        Routing(kj,:)=Route(cntn,:);
    end
    Route=zeros(1,20);
    NAnt=NAnt+1;

    %check if schedule is complete
    if CompleteItems>=NItems
SOut=1;
end

% any remaining agents/planes?
if NRemFleets<0 % No - finish off
SOut=1;
end

end % end - there is nowhere to go

StickCnt=StickCnt+1;
if StickCnt>1000
    stop1=1;
end
end % while NOut - ant out

% set new ant and new time
CurrTime=DayStart;

%SOut=1; % schedule complete
end % schedule complete - SOut

% calculate route cost
[r1 c1]=size(Routing);
TotCost=0;
for i=2:r1
    Flt1=Routing(i,4);
    NOnBrd=Routing(i,5);
    Des1=Routing(i,1);
    Tim1=Routing(i,2);
    EmptyS=Routing(i,3);
    FltCost=fleetType(fleet(Flt1,2),4);

    if i>2
        Or1=Routing(i-1,1);
        TimO=Routing(i-1,2);
        if Des1~=Or1
            if Tim1>TimO
                FlTime=Tim1-TimO;
            end
        end
    end
end
Cost1 = FltCost * FlTime / 60;
if Cost1 < 0
    stop1 = 1;
end
TotCost = TotCost + Cost1;
end
end
end

%xlswrite('routing.xlsx', Routing);

TotCost;
Penal = (NItems - CompleteItems) * Penalty;
TotCost = TotCost + Penal;
u = u + NItems - CompleteItems;
CompleteItems;

AvgSum = AvgSum + TotCost;
end
f = AvgSum / NIters;

%xlswrite('CloseArr.xlsx', CloseArr);
end

function z = CheckD2(TToCollect, City1, GrpToGo, flt1, TATime, sch1, fleetType1, fleet1, NOnBrd1, OnBrd1, dest1)
%function to check whether feasibility can be maintained if a certain city is visited

%city1, TToCollect = where we currently are in space-time
GrpNo = GrpToGo;
Time1 = TToCollect;
NDests = 2;
DestList = [City1 Time1 0]; % dest. city, time to collect (current time)
DestList = [DestList; sch1(GrpNo, 4) sch1(GrpNo, 2) 0]; % grp dest, LAT

fltSpd = fleetType1(fleet1(flt1, 2), 2);

if NOnBrd1 > 0
    for p = 1:NOnBrd1
        NotIn = 0;
        for q = 1:NDests
            % is dest already in dest list?
if OnBrd1(p,4)==DestList(q,1)
    NotIn=1; %dest. already on list
    if OnBrd1(p,2)<DestList(q,2)
        DestList(q,2)=OnBrd1(p,2);
    end
end
end
if NotIn==0
    %add dest to list
    NDest=NDest+1;
    DestList=[DestList; OnBrd1(p,4) OnBrd1(p,2) 0];
end
end

%Travel times table
TTable=zeros(NDest,NDest);
for p=1:NDest
    for q=1:NDest
        dist1=distance2(dest1,DestList(p,1),DestList(q,1));
        flTime=dist1/fltSpd*60;
        TTable(DestList(p,1),DestList(q,1))=flTime;
        TTable(DestList(q,1),DestList(p,1))=flTime;
    end
end

%feasible route search
feasible=1;
outt1=0;
cntt1=2;
CCity=City1;
CTime=Time1;
Ind1=2;
oldCnt(Ind1,1)=1;
DestList(1,3)=1;
StrtPt=zeros(NDest,1);
for i=1:2
    StrtPt(i,1)=i;
end

IndexM=zeros(NDest,2); %prev node index and time
IndexM(1,1)=1;
IndexM(1,2)=CTime;
IndexM(1,3)=CCity;
RowInd=StrtPt;
ReOrder=1;
OrigPt = StrtPt;

while out1 == 0

    if ReOrder == 1
        % reorder lower levels and strtPts
        StrtPt = RowInd;
        for kk = Ind1 + 1:NDests

            outp = 0;

            while outp == 0
                StrtPt(kk,1) = StrtPt(kk,1) + 1;
                if StrtPt(kk,1) > NDests
                    StrtPt(kk,1) = 1;
                end
                OrigPt(kk,1) = StrtPt(kk,1);
            end
            fnd1 = 0;
            for kkk = 1:kk - 1
                if StrtPt(kk,1) == RowInd(kkk,1)
                    fnd1 = 1;
                end
            end
            if fnd1 == 0
                outp = 1;
                RowInd(kk,1) = StrtPt(kk,1);
            end
        end
        ReOrder = 0;
    end

    if DestList(RowInd(Ind1,1), 1) ~= CCity && DestList(RowInd(Ind1,1), 3) ~= 1 % not curr city AND
        NCity = DestList(RowInd(Ind1,1), 1);
        NTime = CTime + TTable(CCity, NCity);
        DestList(RowInd(Ind1,1), 3) = 1;

        IndexM(Ind1,1) = RowInd(cntt1,1);
        IndexM(Ind1,2) = NTime;
        IndexM(Ind1,3) = NCity;

        if NTime > DestList(RowInd(Ind1,1), 2)

            % infeasible
            % move sideways

165
DestList(RowInd(Ind1,1), 3) = 0;

Back1=1;
while Back1==1

outp=0;
while outp==0

    RowInd(Ind1,1)=RowInd(Ind1,1)+1;
    if RowInd(Ind1,1)>NDest
        RowInd(Ind1,1)=1;
    end
    fnd1=0;
    for ii=1:Ind1-1
        if RowInd(ii,1)==RowInd(Ind1,1)
            fnd1=1;
        end
    end
    if fnd1==0
        outp=1;
    end
    %check if RowInd has gone all the way - from StrtPt to
    %StrtPt - if yes, move up a level and reset StrtPt.
    if RowInd(Ind1,1)==OrigPt(Ind1,1)
        Back1=1;
    else
        Back1=0;
    end
    ReOrder=1;
end

else

    %5- backtrack to previous level
    % DestList(RowInd(Ind1,1), 3) = 0;

    %change previous level city
    if Back1==1

        Ind1 = Ind1 - 1;
        DestList(RowInd(Ind1,1), 3) = 0;

        First1=0;
        if Ind1>2
            CTime = IndexM(Ind1-1,2)+TATime;
        else
            if Ind1^=1

166
CTime = IndexM(Ind1-1,2);
else
    CTime = IndexM(Ind1,2);
First1 = 1;
outt1 = 1;
feasible = 0;
Back1 = 0;
end
end
if outt1 == 0
    if Ind1 == 1
        CCity = IndexM(Ind1-1,3);
    else
        CCity = IndexM(Ind1,3);
    end
end
if outt1 == 0
    %DestList(RowInd(Ind1,1), 3) = 0;
outp = 0;
OutCnt = 0;
while outp == 0
    OutCnt = OutCnt + 1;
    RowInd(Ind1,1) = RowInd(Ind1,1) + 1;
    if RowInd(Ind1,1) > NDests
        RowInd(Ind1,1) = 1;
    end
    fnd1 = 0;
    for ii = 1:Ind1-1
        if RowInd(ii,1) == RowInd(Ind1,1)
            fnd1 = 1;
        end
    end
    if fnd1 == 0 || OutCnt > NDests
        outp = 1;
    end
end
%check if RowInd has gone all the way - from StrtPt to %StrtPt - if yes, move up a level and reset StrtPt.
if RowInd(Ind1,1) == OrigPt(Ind1,1)
    Back1 = 1;
    Ind1 = Ind1 - 1;
    DestList(RowInd(Ind1,1), 3) = 0;
if First1==1
    outt1=1;
else
    Back1=0;
end
end
ReOrder=1;
end
end
end

else %move on

    CTime = NTime + TATime;
    CCity = NCity;
    Ind1 = Ind1 + 1;
    oldCnt(Ind1, 1) = 0;
    cntt1 = 1;
end
else
    cntt1 = cntt1 + 1;
    if cntt1>6
        outt1=1;
    end
end
RNu=1;
for tt = 1:NDest
    if DestList(tt, 3) <= 0.1
        RNu = 0;
    end
end
if RNu==1
    outt1=1;
end

z = feasible; %0 no feasible route, 1 feasible

end
end
APPENDIX E  Agent-Generated Variable Schedule
<table>
<thead>
<tr>
<th>Aircraft Type</th>
<th>Origin</th>
<th>Departure Time</th>
<th>Destination</th>
<th>Arrival Time</th>
<th>Pax</th>
<th>Group Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>C206</td>
<td>3</td>
<td>360</td>
<td>7</td>
<td>445</td>
<td>2</td>
<td>66,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>477</td>
<td>3</td>
<td>562</td>
<td>2</td>
<td>61,</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>778</td>
<td>5</td>
<td>834</td>
<td>2</td>
<td>11,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>360</td>
<td>4</td>
<td>407</td>
<td>2</td>
<td>65,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>424</td>
<td>7</td>
<td>471</td>
<td>5</td>
<td>8,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>517</td>
<td>4</td>
<td>563</td>
<td>5</td>
<td>62,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>660</td>
<td>8</td>
<td>695</td>
<td>2</td>
<td>2,</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>742</td>
<td>4</td>
<td>776</td>
<td>0</td>
<td>68,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>790</td>
<td>16</td>
<td>798</td>
<td>0</td>
<td>48, 69,</td>
</tr>
<tr>
<td>C206</td>
<td>16</td>
<td>887</td>
<td>2</td>
<td>923</td>
<td>5</td>
<td>40,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>360</td>
<td>2</td>
<td>390</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>485</td>
<td>7</td>
<td>515</td>
<td>6</td>
<td>32,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>531</td>
<td>2</td>
<td>561</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>610</td>
<td>3</td>
<td>703</td>
<td>9</td>
<td>19,</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>790</td>
<td>18</td>
<td>805</td>
<td>0</td>
<td>7, 26,</td>
</tr>
<tr>
<td>C206</td>
<td>18</td>
<td>815</td>
<td>3</td>
<td>830</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>892</td>
<td>4</td>
<td>881</td>
<td>10</td>
<td>74,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>360</td>
<td>11</td>
<td>392</td>
<td>0</td>
<td>58,</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>660</td>
<td>4</td>
<td>698</td>
<td>2</td>
<td>27, 43,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>708</td>
<td>12</td>
<td>785</td>
<td>6</td>
<td>30,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>360</td>
<td>6</td>
<td>383</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>420</td>
<td>3</td>
<td>490</td>
<td>11</td>
<td>9,</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>500</td>
<td>1</td>
<td>533</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>1</td>
<td>660</td>
<td>2</td>
<td>783</td>
<td>2</td>
<td>73,</td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>870</td>
<td>11</td>
<td>876</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>1039</td>
<td>9</td>
<td>1053</td>
<td>2</td>
<td>60,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>360</td>
<td>10</td>
<td>373</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>660</td>
<td>9</td>
<td>693</td>
<td>0</td>
<td>24,</td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>703</td>
<td>4</td>
<td>754</td>
<td>2</td>
<td>41,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>790</td>
<td>7</td>
<td>837</td>
<td>4</td>
<td>25, 63, 64,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>848</td>
<td>11</td>
<td>880</td>
<td>2</td>
<td>28, 44,</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>883</td>
<td>17</td>
<td>900</td>
<td>4</td>
<td>44,</td>
</tr>
<tr>
<td>C206</td>
<td>17</td>
<td>991</td>
<td>7</td>
<td>1034</td>
<td>4</td>
<td>38,</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>360</td>
<td>9</td>
<td>376</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>480</td>
<td>7</td>
<td>521</td>
<td>5</td>
<td>1,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>531</td>
<td>5</td>
<td>586</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>5</td>
<td>660</td>
<td>6</td>
<td>691</td>
<td>2</td>
<td>10,</td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>701</td>
<td>7</td>
<td>725</td>
<td>8</td>
<td>57,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>793</td>
<td>12</td>
<td>843</td>
<td>0</td>
<td>16,</td>
</tr>
<tr>
<td>C206</td>
<td>17</td>
<td>360</td>
<td>13</td>
<td>376</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>13</td>
<td>386</td>
<td>14</td>
<td>392</td>
<td>0</td>
<td>55,</td>
</tr>
<tr>
<td>C206</td>
<td>14</td>
<td>402</td>
<td>8</td>
<td>406</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>470</td>
<td>7</td>
<td>507</td>
<td>5</td>
<td>33,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>517</td>
<td>4</td>
<td>563</td>
<td>5</td>
<td>62,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>678</td>
<td>14</td>
<td>715</td>
<td>1</td>
<td>13,</td>
</tr>
<tr>
<td>C206</td>
<td>14</td>
<td>720</td>
<td>15</td>
<td>740</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>15</td>
<td>757</td>
<td>11</td>
<td>774</td>
<td>11</td>
<td>35,</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>812</td>
<td>12</td>
<td>894</td>
<td>2</td>
<td>70,</td>
</tr>
<tr>
<td>Aircraft Type</td>
<td>Origin</td>
<td>Departure Time</td>
<td>Destination</td>
<td>Arrival Time</td>
<td>Pax</td>
<td>Groups Onboard</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
<td>----------------</td>
<td>-------------</td>
<td>--------------</td>
<td>-----</td>
<td>----------------</td>
</tr>
<tr>
<td>C206</td>
<td>19</td>
<td>360</td>
<td>9</td>
<td>365</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>660</td>
<td>10</td>
<td>693</td>
<td>2</td>
<td>20,</td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>770</td>
<td>7</td>
<td>783</td>
<td>3</td>
<td>45, 46,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>823</td>
<td>10</td>
<td>835</td>
<td>2</td>
<td>17,</td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>849</td>
<td>8</td>
<td>875</td>
<td>2</td>
<td>21,</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>885</td>
<td>14</td>
<td>889</td>
<td>0</td>
<td>31,</td>
</tr>
<tr>
<td>C206</td>
<td>14</td>
<td>918</td>
<td>8</td>
<td>922</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>923</td>
<td>7</td>
<td>959</td>
<td>0</td>
<td>53,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>1016</td>
<td>4</td>
<td>1062</td>
<td>5</td>
<td>29,</td>
</tr>
<tr>
<td>C208</td>
<td>4</td>
<td>676</td>
<td>16</td>
<td>683</td>
<td>2</td>
<td>12,</td>
</tr>
<tr>
<td>C208</td>
<td>16</td>
<td>693</td>
<td>4</td>
<td>699</td>
<td>10</td>
<td>67,</td>
</tr>
<tr>
<td>C208</td>
<td>4</td>
<td>709</td>
<td>3</td>
<td>749</td>
<td>4</td>
<td>6, 27, 67,</td>
</tr>
<tr>
<td>C208</td>
<td>3</td>
<td>749</td>
<td>5</td>
<td>794</td>
<td>9</td>
<td>79,</td>
</tr>
<tr>
<td>C208</td>
<td>5</td>
<td>804</td>
<td>6</td>
<td>830</td>
<td>5</td>
<td>78,</td>
</tr>
<tr>
<td>C208</td>
<td>6</td>
<td>840</td>
<td>17</td>
<td>869</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>17</td>
<td>884</td>
<td>8</td>
<td>899</td>
<td>3</td>
<td>71,</td>
</tr>
<tr>
<td>C208</td>
<td>8</td>
<td>982</td>
<td>13</td>
<td>990</td>
<td>0</td>
<td>49,</td>
</tr>
<tr>
<td>C208</td>
<td>13</td>
<td>1000</td>
<td>14</td>
<td>1005</td>
<td>11</td>
<td>54,</td>
</tr>
<tr>
<td>C208</td>
<td>14</td>
<td>1012</td>
<td>13</td>
<td>1017</td>
<td>3</td>
<td>50,</td>
</tr>
<tr>
<td>C208</td>
<td>13</td>
<td>1027</td>
<td>8</td>
<td>1035</td>
<td>3</td>
<td>51,</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>360</td>
<td>6</td>
<td>379</td>
<td>6</td>
<td>56,</td>
</tr>
<tr>
<td>C208</td>
<td>6</td>
<td>440</td>
<td>3</td>
<td>497</td>
<td>0</td>
<td>77,</td>
</tr>
<tr>
<td>C208</td>
<td>3</td>
<td>650</td>
<td>18</td>
<td>660</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>18</td>
<td>730</td>
<td>20</td>
<td>735</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>20</td>
<td>735</td>
<td>21</td>
<td>775</td>
<td>0</td>
<td>76,</td>
</tr>
<tr>
<td>C208</td>
<td>21</td>
<td>815</td>
<td>20</td>
<td>855</td>
<td>0</td>
<td>75,</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>465</td>
<td>8</td>
<td>495</td>
<td>0</td>
<td>42,</td>
</tr>
<tr>
<td>C208</td>
<td>8</td>
<td>787</td>
<td>11</td>
<td>792</td>
<td>0</td>
<td>5, 39, 52, 70,</td>
</tr>
<tr>
<td>C208</td>
<td>11</td>
<td>801</td>
<td>7</td>
<td>826</td>
<td>0</td>
<td>5, 18,</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>849</td>
<td>2</td>
<td>873</td>
<td>0</td>
<td>34, 47, 72,</td>
</tr>
<tr>
<td>C208</td>
<td>2</td>
<td>883</td>
<td>9</td>
<td>892</td>
<td>0</td>
<td>34,</td>
</tr>
<tr>
<td>C208</td>
<td>9</td>
<td>1017</td>
<td>7</td>
<td>1050</td>
<td>0</td>
<td>37,</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>617</td>
<td>19</td>
<td>646</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>19</td>
<td>731</td>
<td>7</td>
<td>760</td>
<td>0</td>
<td>22,</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>853</td>
<td>8</td>
<td>883</td>
<td>0</td>
<td>31, 36, 59,</td>
</tr>
<tr>
<td>C208</td>
<td>18</td>
<td>620</td>
<td>3</td>
<td>632</td>
<td>0</td>
<td>4, 14,</td>
</tr>
<tr>
<td>C208</td>
<td>3</td>
<td>733</td>
<td>4</td>
<td>773</td>
<td>0</td>
<td>3, 15, 48, 69,</td>
</tr>
<tr>
<td>C208</td>
<td>4</td>
<td>783</td>
<td>19</td>
<td>824</td>
<td>0</td>
<td>15,</td>
</tr>
<tr>
<td>C208</td>
<td>19</td>
<td>893</td>
<td>7</td>
<td>923</td>
<td>0</td>
<td>23,</td>
</tr>
</tbody>
</table>
APPENDIX F  Automated Composite Schedule
<table>
<thead>
<tr>
<th>Aircraft Type</th>
<th>Origin</th>
<th>Departure Time</th>
<th>Destination</th>
<th>Arrival Time</th>
<th>Pax</th>
<th>Groups Onboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>C208</td>
<td>7</td>
<td>370</td>
<td>8</td>
<td>400</td>
<td>11</td>
<td>42,</td>
</tr>
<tr>
<td>C208</td>
<td>8</td>
<td>400</td>
<td>13</td>
<td>410</td>
<td>10</td>
<td>49,</td>
</tr>
<tr>
<td>C208</td>
<td>13</td>
<td>370</td>
<td>14</td>
<td>420</td>
<td>2</td>
<td>25,</td>
</tr>
<tr>
<td>C208</td>
<td>14</td>
<td>450</td>
<td>13</td>
<td>460</td>
<td>6</td>
<td>50,</td>
</tr>
<tr>
<td>C208</td>
<td>13</td>
<td>530</td>
<td>8</td>
<td>540</td>
<td>11</td>
<td>51,</td>
</tr>
<tr>
<td>C208</td>
<td>8</td>
<td>670</td>
<td>17</td>
<td>600</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>17</td>
<td>690</td>
<td>8</td>
<td>710</td>
<td>8</td>
<td>71,</td>
</tr>
<tr>
<td>C208</td>
<td>8</td>
<td>770</td>
<td>11</td>
<td>790</td>
<td>7</td>
<td>52, 18,</td>
</tr>
<tr>
<td>C208</td>
<td>11</td>
<td>790</td>
<td>7</td>
<td>820</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>860</td>
<td>2</td>
<td>890</td>
<td>4</td>
<td>72,</td>
</tr>
<tr>
<td>C208</td>
<td>2</td>
<td>900</td>
<td>19</td>
<td>910</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>19</td>
<td>910</td>
<td>7</td>
<td>940</td>
<td>6</td>
<td>22,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>380</td>
<td>6</td>
<td>410</td>
<td>3</td>
<td>56,</td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>450</td>
<td>3</td>
<td>530</td>
<td>5</td>
<td>9,</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>610</td>
<td>5</td>
<td>670</td>
<td>2</td>
<td>11,</td>
</tr>
<tr>
<td>C206</td>
<td>5</td>
<td>670</td>
<td>6</td>
<td>710</td>
<td>5</td>
<td>10,</td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>710</td>
<td>16</td>
<td>740</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>16</td>
<td>740</td>
<td>3</td>
<td>800</td>
<td>4</td>
<td>67,</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>860</td>
<td>4</td>
<td>920</td>
<td>5</td>
<td>74, 12,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>920</td>
<td>16</td>
<td>930</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>390</td>
<td>11</td>
<td>420</td>
<td>4</td>
<td>58,</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>490</td>
<td>2</td>
<td>500</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>500</td>
<td>7</td>
<td>540</td>
<td>2</td>
<td>32,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>560</td>
<td>1</td>
<td>680</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>1</td>
<td>690</td>
<td>2</td>
<td>820</td>
<td>3</td>
<td>73,</td>
</tr>
<tr>
<td>C206</td>
<td>2</td>
<td>860</td>
<td>9</td>
<td>880</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>900</td>
<td>10</td>
<td>940</td>
<td>4</td>
<td>20,</td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>1060</td>
<td>7</td>
<td>1080</td>
<td>2</td>
<td>46,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>390</td>
<td>4</td>
<td>440</td>
<td>1</td>
<td>62,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>440</td>
<td>7</td>
<td>490</td>
<td>4</td>
<td>8,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>500</td>
<td>3</td>
<td>640</td>
<td>4</td>
<td>47, 19,</td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>670</td>
<td>19</td>
<td>770</td>
<td>2</td>
<td>15,</td>
</tr>
<tr>
<td>C206</td>
<td>19</td>
<td>770</td>
<td>9</td>
<td>780</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>780</td>
<td>4</td>
<td>840</td>
<td>2</td>
<td>41,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>840</td>
<td>14</td>
<td>940</td>
<td>4</td>
<td>25, 31,</td>
</tr>
<tr>
<td>C206</td>
<td>14</td>
<td>950</td>
<td>13</td>
<td>960</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>13</td>
<td>970</td>
<td>14</td>
<td>980</td>
<td>4</td>
<td>54,</td>
</tr>
<tr>
<td>C206</td>
<td>14</td>
<td>980</td>
<td>17</td>
<td>1000</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>420</td>
<td>6</td>
<td>440</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>6</td>
<td>480</td>
<td>3</td>
<td>540</td>
<td>11</td>
<td>77,</td>
</tr>
<tr>
<td>C208</td>
<td>3</td>
<td>760</td>
<td>18</td>
<td>780</td>
<td>2</td>
<td>7,</td>
</tr>
<tr>
<td>C208</td>
<td>18</td>
<td>780</td>
<td>3</td>
<td>800</td>
<td>6</td>
<td>4,</td>
</tr>
<tr>
<td>C208</td>
<td>3</td>
<td>800</td>
<td>5</td>
<td>850</td>
<td>11</td>
<td>79,</td>
</tr>
<tr>
<td>C208</td>
<td>5</td>
<td>850</td>
<td>6</td>
<td>880</td>
<td>11</td>
<td>78,</td>
</tr>
<tr>
<td>C208</td>
<td>6</td>
<td>900</td>
<td>15</td>
<td>910</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>15</td>
<td>910</td>
<td>11</td>
<td>930</td>
<td>2</td>
<td>35,</td>
</tr>
<tr>
<td>C208</td>
<td>11</td>
<td>930</td>
<td>9</td>
<td>950</td>
<td>1</td>
<td>60,</td>
</tr>
<tr>
<td>C208</td>
<td>9</td>
<td>970</td>
<td>7</td>
<td>1010</td>
<td>11</td>
<td>37,</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>490</td>
<td>7</td>
<td>530</td>
<td>3</td>
<td>33,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>590</td>
<td>8</td>
<td>630</td>
<td>5</td>
<td>59,</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>670</td>
<td>12</td>
<td>800</td>
<td>4</td>
<td>68, 30,</td>
</tr>
<tr>
<td>C206</td>
<td>12</td>
<td>800</td>
<td>3</td>
<td>890</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>890</td>
<td>18</td>
<td>910</td>
<td>2</td>
<td>26,</td>
</tr>
<tr>
<td>C206</td>
<td>18</td>
<td>910</td>
<td>3</td>
<td>930</td>
<td>2</td>
<td>14,</td>
</tr>
<tr>
<td>Aircraft Type</td>
<td>Origin</td>
<td>Departure Time</td>
<td>Destination</td>
<td>Arrival Time</td>
<td>Pax</td>
<td>Groups Onboard</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
<td>----------------</td>
<td>-------------</td>
<td>--------------</td>
<td>-----</td>
<td>----------------</td>
</tr>
<tr>
<td>C206</td>
<td>17</td>
<td>490</td>
<td>9</td>
<td>500</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>9</td>
<td>500</td>
<td>7</td>
<td>550</td>
<td>3</td>
<td>1,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>760</td>
<td>10</td>
<td>780</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>10</td>
<td>780</td>
<td>8</td>
<td>840</td>
<td>4</td>
<td>45, 36,</td>
</tr>
<tr>
<td>C206</td>
<td>8</td>
<td>840</td>
<td>11</td>
<td>850</td>
<td>2</td>
<td>39,</td>
</tr>
<tr>
<td>C206</td>
<td>11</td>
<td>850</td>
<td>3</td>
<td>940</td>
<td>2</td>
<td>27,</td>
</tr>
<tr>
<td>C208</td>
<td>4</td>
<td>660</td>
<td>16</td>
<td>670</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>16</td>
<td>670</td>
<td>2</td>
<td>700</td>
<td>2</td>
<td>40,</td>
</tr>
<tr>
<td>C208</td>
<td>2</td>
<td>700</td>
<td>19</td>
<td>710</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>19</td>
<td>790</td>
<td>7</td>
<td>820</td>
<td>7</td>
<td>23,</td>
</tr>
<tr>
<td>C208</td>
<td>7</td>
<td>870</td>
<td>9</td>
<td>920</td>
<td>9</td>
<td>17, 24,</td>
</tr>
<tr>
<td>C208</td>
<td>9</td>
<td>920</td>
<td>17</td>
<td>930</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>3</td>
<td>710</td>
<td>4</td>
<td>760</td>
<td>4</td>
<td>3,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>760</td>
<td>7</td>
<td>850</td>
<td>5</td>
<td>1, 2, 5,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>850</td>
<td>17</td>
<td>900</td>
<td>2</td>
<td>44,</td>
</tr>
<tr>
<td>C206</td>
<td>17</td>
<td>900</td>
<td>7</td>
<td>950</td>
<td>2</td>
<td>38,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>960</td>
<td>4</td>
<td>1010</td>
<td>2</td>
<td>29,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>1010</td>
<td>6</td>
<td>1040</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>6</td>
<td>1040</td>
<td>7</td>
<td>1070</td>
<td>3</td>
<td>57,</td>
</tr>
<tr>
<td>C208</td>
<td>18</td>
<td>740</td>
<td>20</td>
<td>750</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C208</td>
<td>20</td>
<td>750</td>
<td>21</td>
<td>800</td>
<td>11</td>
<td>76,</td>
</tr>
<tr>
<td>C208</td>
<td>21</td>
<td>830</td>
<td>20</td>
<td>880</td>
<td>11</td>
<td>75,</td>
</tr>
<tr>
<td>C208</td>
<td>20</td>
<td>980</td>
<td>3</td>
<td>1000</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>780</td>
<td>12</td>
<td>840</td>
<td>3</td>
<td>16,</td>
</tr>
<tr>
<td>C206</td>
<td>7</td>
<td>780</td>
<td>4</td>
<td>860</td>
<td>4</td>
<td>28, 43,</td>
</tr>
<tr>
<td>C206</td>
<td>4</td>
<td>920</td>
<td>7</td>
<td>1070</td>
<td>5</td>
<td>6, 66,</td>
</tr>
</tbody>
</table>
APPENDIX G  MVCVRP Data

G.1  Problem Data
Table G.1: Large Instance

<table>
<thead>
<tr>
<th>z position (km)</th>
<th>y position (km)</th>
<th>load (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>39.5</td>
<td>27.2</td>
<td>519</td>
</tr>
<tr>
<td>22.3</td>
<td>36.4</td>
<td>415</td>
</tr>
<tr>
<td>56.3</td>
<td>38.3</td>
<td>1137</td>
</tr>
<tr>
<td>3.4</td>
<td>19.5</td>
<td>446</td>
</tr>
<tr>
<td>58.5</td>
<td>17.6</td>
<td>207</td>
</tr>
<tr>
<td>64.5</td>
<td>2.6</td>
<td>395</td>
</tr>
<tr>
<td>26.9</td>
<td>24.7</td>
<td>1595</td>
</tr>
<tr>
<td>4.0</td>
<td>27.8</td>
<td>3562</td>
</tr>
<tr>
<td>22.1</td>
<td>29.3</td>
<td>176</td>
</tr>
<tr>
<td>19.5</td>
<td>13.1</td>
<td>860</td>
</tr>
<tr>
<td>61.0</td>
<td>27.7</td>
<td>149</td>
</tr>
<tr>
<td>67.4</td>
<td>10.7</td>
<td>84</td>
</tr>
<tr>
<td>72.5</td>
<td>11.5</td>
<td>839</td>
</tr>
<tr>
<td>7.9</td>
<td>47.0</td>
<td>57</td>
</tr>
<tr>
<td>1.2</td>
<td>27.0</td>
<td>62</td>
</tr>
<tr>
<td>7.6</td>
<td>37.5</td>
<td>443</td>
</tr>
<tr>
<td>3.4</td>
<td>13.9</td>
<td>401</td>
</tr>
<tr>
<td>22.3</td>
<td>44.6</td>
<td>654</td>
</tr>
<tr>
<td>29.7</td>
<td>13.1</td>
<td>191</td>
</tr>
<tr>
<td>12.0</td>
<td>30.4</td>
<td>830</td>
</tr>
<tr>
<td>30.5</td>
<td>33.5</td>
<td>278</td>
</tr>
<tr>
<td>46.9</td>
<td>9.8</td>
<td>121</td>
</tr>
<tr>
<td>43.2</td>
<td>3.8</td>
<td>628</td>
</tr>
<tr>
<td>67.0</td>
<td>12.3</td>
<td>214</td>
</tr>
<tr>
<td>28.0</td>
<td>13.6</td>
<td>1695</td>
</tr>
<tr>
<td>46.7</td>
<td>29.5</td>
<td>2136</td>
</tr>
<tr>
<td>7.3</td>
<td>26.4</td>
<td>83</td>
</tr>
<tr>
<td>67.6</td>
<td>39.2</td>
<td>21</td>
</tr>
<tr>
<td>40.2</td>
<td>43.1</td>
<td>399</td>
</tr>
<tr>
<td>50.2</td>
<td>23.6</td>
<td>217</td>
</tr>
<tr>
<td>34.3</td>
<td>16.6</td>
<td>469</td>
</tr>
<tr>
<td>20.0</td>
<td>2.6</td>
<td>278</td>
</tr>
<tr>
<td>72.5</td>
<td>2.9</td>
<td>86</td>
</tr>
<tr>
<td>27.0</td>
<td>23.0</td>
<td>979</td>
</tr>
<tr>
<td>35.1</td>
<td>12.1</td>
<td>79</td>
</tr>
<tr>
<td>40.1</td>
<td>7.3</td>
<td>66</td>
</tr>
<tr>
<td>48.4</td>
<td>23.8</td>
<td>913</td>
</tr>
<tr>
<td>7.9</td>
<td>36.8</td>
<td>586</td>
</tr>
<tr>
<td>55.8</td>
<td>28.0</td>
<td>108</td>
</tr>
<tr>
<td>1.4</td>
<td>9.9</td>
<td>1249</td>
</tr>
<tr>
<td>7.8</td>
<td>15.6</td>
<td>614</td>
</tr>
<tr>
<td>0.0</td>
<td>25.2</td>
<td>1897</td>
</tr>
<tr>
<td>40.3</td>
<td>38.9</td>
<td>692</td>
</tr>
<tr>
<td>14.2</td>
<td>31.9</td>
<td>100</td>
</tr>
<tr>
<td>26.4</td>
<td>7.0</td>
<td>423</td>
</tr>
<tr>
<td>68.7</td>
<td>24.9</td>
<td>601</td>
</tr>
<tr>
<td>56.1</td>
<td>18.9</td>
<td>56</td>
</tr>
<tr>
<td>x position (km)</td>
<td>y position (km)</td>
<td>load (kg)</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
<td>-----------</td>
</tr>
<tr>
<td>36.4</td>
<td>9.8</td>
<td>1051</td>
</tr>
<tr>
<td>7.1</td>
<td>27.7</td>
<td>588</td>
</tr>
<tr>
<td>68.6</td>
<td>4.6</td>
<td>987</td>
</tr>
<tr>
<td>20.2</td>
<td>41.0</td>
<td>100</td>
</tr>
<tr>
<td>20.2</td>
<td>31.7</td>
<td>217</td>
</tr>
<tr>
<td>6.7</td>
<td>1.5</td>
<td>1</td>
</tr>
<tr>
<td>58.5</td>
<td>14.0</td>
<td>202</td>
</tr>
<tr>
<td>35.6</td>
<td>12.0</td>
<td>1</td>
</tr>
<tr>
<td>3.3</td>
<td>22.7</td>
<td>1801</td>
</tr>
<tr>
<td>64.0</td>
<td>27.7</td>
<td>790</td>
</tr>
<tr>
<td>68.7</td>
<td>15.6</td>
<td>1136</td>
</tr>
<tr>
<td>6.0</td>
<td>29.8</td>
<td>514</td>
</tr>
<tr>
<td>71.1</td>
<td>5.4</td>
<td>105</td>
</tr>
<tr>
<td>45.9</td>
<td>16.3</td>
<td>134</td>
</tr>
<tr>
<td>35.5</td>
<td>10.3</td>
<td>425</td>
</tr>
<tr>
<td>9.7</td>
<td>1.4</td>
<td>1087</td>
</tr>
<tr>
<td>40.5</td>
<td>43.4</td>
<td>85</td>
</tr>
<tr>
<td>30.1</td>
<td>39.8</td>
<td>1036</td>
</tr>
<tr>
<td>49.8</td>
<td>33.9</td>
<td>716</td>
</tr>
<tr>
<td>25.1</td>
<td>23.3</td>
<td>358</td>
</tr>
<tr>
<td>51.5</td>
<td>8.4</td>
<td>69</td>
</tr>
<tr>
<td>40.2</td>
<td>38.3</td>
<td>375</td>
</tr>
<tr>
<td>31.6</td>
<td>23.9</td>
<td>99</td>
</tr>
<tr>
<td>45.8</td>
<td>23.0</td>
<td>478</td>
</tr>
<tr>
<td>65.6</td>
<td>17.4</td>
<td>834</td>
</tr>
<tr>
<td>21.7</td>
<td>7.1</td>
<td>162</td>
</tr>
<tr>
<td>16.5</td>
<td>27.5</td>
<td>473</td>
</tr>
<tr>
<td>64.8</td>
<td>22.5</td>
<td>43</td>
</tr>
<tr>
<td>50.6</td>
<td>35.1</td>
<td>826</td>
</tr>
<tr>
<td>57.9</td>
<td>7.6</td>
<td>471</td>
</tr>
<tr>
<td>15.0</td>
<td>45.0</td>
<td>2370</td>
</tr>
<tr>
<td>4.6</td>
<td>37.3</td>
<td>123</td>
</tr>
<tr>
<td>34.3</td>
<td>5.6</td>
<td>1037</td>
</tr>
<tr>
<td>12.9</td>
<td>2.3</td>
<td>140</td>
</tr>
<tr>
<td>39.4</td>
<td>26.4</td>
<td>520</td>
</tr>
<tr>
<td>34.6</td>
<td>35.1</td>
<td>82</td>
</tr>
<tr>
<td>29.5</td>
<td>42.4</td>
<td>507</td>
</tr>
<tr>
<td>6.6</td>
<td>29.8</td>
<td>1031</td>
</tr>
<tr>
<td>1.2</td>
<td>20.3</td>
<td>89</td>
</tr>
<tr>
<td>20.4</td>
<td>46.3</td>
<td>171</td>
</tr>
<tr>
<td>51.5</td>
<td>19.6</td>
<td>98</td>
</tr>
<tr>
<td>20.5</td>
<td>16.8</td>
<td>44</td>
</tr>
<tr>
<td>69.9</td>
<td>5.7</td>
<td>797</td>
</tr>
<tr>
<td>25.7</td>
<td>4.9</td>
<td>180</td>
</tr>
<tr>
<td>5.7</td>
<td>20.3</td>
<td>77</td>
</tr>
<tr>
<td></td>
<td>x position (km)</td>
<td>y position (km)</td>
</tr>
<tr>
<td>-------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>40.1</td>
<td>23.2</td>
<td></td>
</tr>
<tr>
<td>16.1</td>
<td>17.8</td>
<td></td>
</tr>
<tr>
<td>16.1</td>
<td>17.8</td>
<td></td>
</tr>
<tr>
<td>20.8</td>
<td>23.7</td>
<td></td>
</tr>
<tr>
<td>38.3</td>
<td>45.4</td>
<td></td>
</tr>
<tr>
<td>67.3</td>
<td>30.9</td>
<td></td>
</tr>
<tr>
<td>51.3</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>51.8</td>
<td>23.4</td>
<td></td>
</tr>
<tr>
<td>16.6</td>
<td>15.3</td>
<td></td>
</tr>
<tr>
<td>3.7</td>
<td>24.3</td>
<td></td>
</tr>
<tr>
<td>59.2</td>
<td>15.3</td>
<td></td>
</tr>
<tr>
<td>59.5</td>
<td>31.7</td>
<td></td>
</tr>
<tr>
<td>64.8</td>
<td>19.6</td>
<td></td>
</tr>
<tr>
<td>70.6</td>
<td>37.5</td>
<td></td>
</tr>
<tr>
<td>29.7</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>12.2</td>
<td>24.0</td>
<td></td>
</tr>
<tr>
<td>7.9</td>
<td>13.0</td>
<td></td>
</tr>
<tr>
<td>62.8</td>
<td>23.4</td>
<td></td>
</tr>
<tr>
<td>66.3</td>
<td>17.5</td>
<td></td>
</tr>
<tr>
<td>57.0</td>
<td>10.2</td>
<td></td>
</tr>
<tr>
<td>17.5</td>
<td>41.3</td>
<td></td>
</tr>
<tr>
<td>27.7</td>
<td>18.3</td>
<td></td>
</tr>
<tr>
<td>43.4</td>
<td>43.8</td>
<td></td>
</tr>
<tr>
<td>24.4</td>
<td>40.8</td>
<td></td>
</tr>
<tr>
<td>19.2</td>
<td>8.4</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>34.9</td>
<td></td>
</tr>
<tr>
<td>20.6</td>
<td>33.0</td>
<td></td>
</tr>
<tr>
<td>60.1</td>
<td>34.9</td>
<td></td>
</tr>
<tr>
<td>5.8</td>
<td>19.3</td>
<td></td>
</tr>
<tr>
<td>52.6</td>
<td>14.7</td>
<td></td>
</tr>
<tr>
<td>11.2</td>
<td>27.9</td>
<td></td>
</tr>
<tr>
<td>18.0</td>
<td>44.2</td>
<td></td>
</tr>
<tr>
<td>72.8</td>
<td>29.7</td>
<td></td>
</tr>
<tr>
<td>66.8</td>
<td>27.0</td>
<td></td>
</tr>
<tr>
<td>63.7</td>
<td>3.5</td>
<td></td>
</tr>
<tr>
<td>56.2</td>
<td>11.5</td>
<td></td>
</tr>
</tbody>
</table>

Table G.2: Large Instance - Vehicles

<table>
<thead>
<tr>
<th>Vehicle No.</th>
<th>Max. Load (kg)</th>
<th>Average Speed (km/hr)</th>
<th>Cost (R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>850</td>
<td>50</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>50</td>
<td>2.6</td>
</tr>
<tr>
<td>3</td>
<td>2000</td>
<td>45</td>
<td>3.7</td>
</tr>
<tr>
<td>4</td>
<td>3000</td>
<td>45</td>
<td>4.2</td>
</tr>
<tr>
<td>5</td>
<td>3500</td>
<td>40</td>
<td>4.5</td>
</tr>
<tr>
<td>6</td>
<td>4000</td>
<td>40</td>
<td>4.7</td>
</tr>
<tr>
<td>7</td>
<td>5500</td>
<td>35</td>
<td>5.8</td>
</tr>
<tr>
<td>8</td>
<td>6500</td>
<td>35</td>
<td>6.6</td>
</tr>
</tbody>
</table>
Table G.3: Medium Instance - Customers

<table>
<thead>
<tr>
<th>x position (km)</th>
<th>y position (km)</th>
<th>load (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>22.0</td>
<td>28.5</td>
<td>527</td>
</tr>
<tr>
<td>23.2</td>
<td>4.4</td>
<td>686</td>
</tr>
<tr>
<td>70.8</td>
<td>34.5</td>
<td>56</td>
</tr>
<tr>
<td>34.0</td>
<td>21.6</td>
<td>1401</td>
</tr>
<tr>
<td>30.4</td>
<td>33.2</td>
<td>6500</td>
</tr>
<tr>
<td>28.6</td>
<td>11.1</td>
<td>1037</td>
</tr>
<tr>
<td>5.2</td>
<td>37.1</td>
<td>637</td>
</tr>
<tr>
<td>24.6</td>
<td>19.7</td>
<td>6421</td>
</tr>
<tr>
<td>51.7</td>
<td>23.1</td>
<td>6500</td>
</tr>
<tr>
<td>25.2</td>
<td>7.2</td>
<td>6421</td>
</tr>
<tr>
<td>37.8</td>
<td>37.1</td>
<td>6500</td>
</tr>
<tr>
<td>25.6</td>
<td>45.2</td>
<td>2617</td>
</tr>
<tr>
<td>22.5</td>
<td>10.3</td>
<td>1036</td>
</tr>
<tr>
<td>66.1</td>
<td>5.3</td>
<td>6148</td>
</tr>
<tr>
<td>46.0</td>
<td>41.8</td>
<td>6500</td>
</tr>
<tr>
<td>24.3</td>
<td>25.3</td>
<td>5120</td>
</tr>
<tr>
<td>11.3</td>
<td>37.3</td>
<td>6421</td>
</tr>
<tr>
<td>40.2</td>
<td>3.4</td>
<td>1401</td>
</tr>
<tr>
<td>61.8</td>
<td>46.3</td>
<td>222</td>
</tr>
<tr>
<td>48.6</td>
<td>44.8</td>
<td>123</td>
</tr>
</tbody>
</table>

Table G.4: Medium Instance - Vehicles

<table>
<thead>
<tr>
<th>Vehicle No.</th>
<th>Max. Load (kg)</th>
<th>Average Speed (km/hr)</th>
<th>Cost (R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>850</td>
<td>50</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>2000</td>
<td>45</td>
<td>3.7</td>
</tr>
<tr>
<td>3</td>
<td>4000</td>
<td>40</td>
<td>4.7</td>
</tr>
<tr>
<td>4</td>
<td>6500</td>
<td>35</td>
<td>6.6</td>
</tr>
</tbody>
</table>
G.2 Supplied Delivery Lists

G.2.1 List for Day 1

nazareth house 207
Garden village 311
meat good cash & carry 975
tacoma spur 173
warmbad rusoord maatskappy inc 459
supreme chicken centre 2737
bronco spar 278
stone craddle rietvlei 597
silverton sandwich bar 10
les marais spar 195
emperor’s palace 523
ciao baby cucina 47
cullinan sorg- en 1659
willow haven 250
tembisa hospital 1253
gerotek test facilities a 324
makhado correctional service 3943
fruit stop silverton 1080
multi meat & chicken wholesale 806
oasis lodge cc 693
rtt 462
dpt of correctional middelburg 1427
fringes 244
pax convenience stores 975
waterberg ouetehuis 349
medina meats 41
netcare optiklin hospital 154
fish 563
summit on site cc 444
f.h odendaal hospital 876
bergies take aways 87
mark slaghuis 161
goldreef city casino 392
arwyp medical centre 1611
liberty life 929
constantia restaurant 146
automotive supplier park 439
adams takeaways 217
pride of india 115
tara hospital 34
rayton rite value-chip & dip 546
the friendly butchery (pty) ltd 61
warmbath hospital 1731
fairleads old age home 239
makro-struben valley 170
a manjee id 6601315307180 229
kevin poto’ father funeral 113
hunter slaghuis 682
fruit & veg city louis trich 680
clearview rehabilitation centr 329
home style fish & chips 419
herfsland old age home 86
meat mecca 3904
holy cross home 426
waltloo meat & chicken 1324
fire fly foods 817
warmbaths koelkamers 1440
denab 73
s.a army 544
kevin poto - funeral 453
meat city 854
ekklesia park 337
naboom fruit & veg 240
mars africa rosslyn 40
catering school 145
salmas light gm & hardware 87
san salvador 157
ac nielsen marketing & media 50
central kitchen 671
ons herberg npo 001/671 1456
big save supermarket 569
allan woodrow 269
m.d.r 848
putco mamelodi sbu 30
j.r. de abrco t/a blou bull 773
wimpy hatfield 53
tambo memorial hospital 3483
boston deli hazelwood 164
gerbera mess 339
main kitchen 982
mr munchies 776
a mess 5120
polokwane correctional service 7651
africafe 22
180 degrees done to perfection 572
national ceremonial gaurd 206
putco roseville 175
o’hagans 547
mario’s cash & carry 6529
crossroads recovery centre 96
sportlight gourmet foods 313
abedare cable edenvale 216
plot 32 173
4 sai 1362
bolivai lodge 686
geneeering formation 218
queens haven 237
ingwe single quarters 222
meat & veg chicken 283
peters supermarket foodzone 1447
sandfi aventis 478
roodepoort chicken wholesalers 762
kankanje aftree oord 1532
summit college 121
seafood delight 129

G.2.2 List for Day 2

forest farm 285
chatter box coffee cafe 173
magalies park 141
siemens wadeville 161
crawdaddys 96
centurion sub acute 44
s.a.bank note co. (pty)ltd 2021
samro 311
pineslopes spar 274
stone cradle rietvlei 1832
brpm hostel raasimone 29
tiberius fish shop 268
vibe catering 705
randjes park spar 234
promise grill restaurant 478
gallagher gourmet & catering c 1031
panarottis menlyn 141
prime co meat 2376
chamara trading cc (bosman) 2789
les marais spar 236
lafarge 157
la scala 692
big daddys 147
bondi distributors cc 2506
ann harding 125
bj’s the bridge 535
azinaz trading cc 9726
clicks distribution centre 102
millenium 189
clover sa 301
s.a.reserve bank 912
meat empire cc ck96/15394/23 265
gerotek test facilities 306
sizwe tropical disease hosp. 1104
banjaara 448
masakhane cookfreeze 9020
weskoppies hospital main kitch 3870
tumela hostel 3389
sandton foods 466
snoopy snacks 147
goldreef city casino 2352
arwyp medical centre 308
osalito centurion 65
npc staff restaurant 506
the raj indian restaurant 2617
mr salad 118
makro woodmead 224
p v b brits 836
tara hospital 186
dishaba hostel 1462
m s fernandes t/a target foods 1899
tip top meat products 1500
hernic ferrochrome mine 1690
platter xpress cc 222
quality meal services 276
emily hobhousetehuis 458
the friendly butchery (pty)ltd 101
health emporium & produce 1401
j.r. de abrco t/a blou bull 668
pretoria academic hospital 103
the sheiks palace 160
coco bonco 80
scrooge diner 410
baglios cc 506
sahil curry den 483
emperor foods 32
amandelbult hospital 75
papachinos clearwater 149
mars africa rosslyn 594
harry’s fast food centre cc 99
st martins school 305
frontline cash & carry 1557
mlanje hostel 1825
laudium meat supply 1000
bonnies fast foods 58
masonic haven 250
tuscan bbq groenkloof 1094
ons huis 103
lathyser fish and chips 418
central kitchen 216
tshwane university of technolo 278
matador w/salers 3211
clicks upd (roodeplaat) 385
ok value 6421
cosmic gold trd 368 579
pick n pay 422
boston deli hazelwood 637
ciao baby cucina 139
clayton house 121
africafe 579
serene park retirement village 510
kashif’s fusion food restaurant 88
ghazi bryanston 262
nathis catering 297
p.p.m training centre 195
nedbank selby ext 233
s.a.p college winkel 2163
nestle s.a 503
wolhuterskop trading 61
wimpy 1 4538
eugene marais creche 247
unisa florida campus 270
citadela 231
caternet pty ltd 7302
queens haven 289
barneys den supermarket 294
akeso parktown 310
ghazi north indain rest cc 49
king pie 852
casa bella guest 5
montina investment 728
meat net cc 285
parrots kolonade 229
germiston hospital 527
r p m rustenburg 176
desmondi campus catering 156
nampak rosslyn 285
bread basket franchise 135
G.2.3 List for Day 3

nazareth house 519
huis vergendeg 415
ons tuis riveria 1137
fego student centre (main camp 446
north indian tansoor rest 207
setaria spar 395
the classic india restaurant 1595
s.a.p. hondeskool menasie 3562
gallegher gourmet catering 176
pretoria west hospital 860
cf fish market silverlakes 149
putco selby 84
vibe catering 839
the bread basket eastgate 57
panarottis menlyn 62
wong thai - sandton 443
masonic haven 401
chamara trading cc (bosman) 654
serenity aged home 191
les marais spar 830
clicks upd (roodeplaat) 278
deliver to ezulwini 121
detective academy 628
jaffa 214
azani caterers ck995487523 1695
tembisa hospital 2136
avalon 83
thava restaurant 21
gerotek test facilities a 399
oyo-restaurant 217
varsity bakery 469
bj’s the bridge 278
welkom mutchery 86
ghazal sunninghill 979
clicks distribution centre 79
mugg & bean 66
neptunes fishmonger 913
orient restaurant 586
librety life 108
gallagher gourmet & catering 1249
fish 614
igbal meat and deli 1897
a taste of punjab 692
nedbank 100 main 324
dawn 423
amor spyseniers 601
heidelberg graphics 56
goldreef city casino 1051
arwyp medical centre 588
retire@ midstream 987
crescent clinic 100
ebg sherwood trust (pty) ltd 217
wimpy no 1 548
liberty life 202
boom street no 1 1360
the raj indian restaurant 1801
mamelodi hospital 790
m s fernandes t/a target foods 1136
cape town fish market s/down 514
nedbank 105 west 263
excellent meat &chicken 134
platter xpress cc 425
ad-eed super market 1087
lorraine rautenback 85
gineeering formation 1036
staff hostel 716
east lynne spar 358
makro-struben valley 69
chazal bryanston 375
vetkoekden 99
norma & vilma caterers 478
emperor foods 834
rens butchery 162
so yum restaurant 473
papachinos clearwater 43
emperor’s palace 826
alberton meat market 471
bulk pack butchery 2370
a.g gooliman sons 123
fordsburg meat & supermarket c 1037
ekklesia park 140
klerksdorp tehuis vir bejaarde 520
d.c.l foods (pty) ltd 82
stabilis treatment centre 507
vista medi clinic 1031
del to ezulwini 89
van rensburg old age home 171
gafa huis protea home 98
fego lynwood 44
mf seafood distributors 797
js tikka & kebab 180
golden valley chinese rest 77
tambo memorial hospital 1054
a mess 7148
clayton house 38
five star caterers 338
africafe 648
sita 22
serene park retirement village 220
the sheiks palace 526
nathis catering 300
circle catering and 388
anglo platinum amandelbuilt 6190
akeso alberton 421
keg no 1 113
keg no 2 89
chazal north indian rest cc 104
desmondi campus catering 426
john dory’s fish & grill 205
21 signal mess 3952
skill moters 198
meat & veg chicken 150
akeso parktown 173
sandfi aventis 360
mc synty’s no 1 202
charles tiganis food enterpris 135
jubilee wits 139
ogalito centurion 297
armadillo’s coffee shop 65
swartklip mine 6148
the food factor 717
summit college 1010
bpr management services cc 82
ann harding 50
officer commanding 906
bread basket franchise 117
site park build (cafeteria) 636
function z=fn_agentV2(Fac,num1);

global A

Fix=0; %don't sample randomly - choose most attractive city
Fix2=0; %also randomize attr factors

WCap=200;

[r1 c1]=size(Fac);
if c1>1 && r1>1
    Fac=Fac(:,1);
elseif c1>1
    Fac=Fac';
end

ZCnt=0;
DayEnd=240;
TAT=10;

CurrentCity=1;
CurrentTime=0;
CurrentLoad=0;

cnt1=1;
Route1=[];
Route1(cnt1,1)=CurrentCity;

BiggestLoad=max(A(:,4));
BiggestX=max(A(:,2));
BiggestY=max(A(:,3));
[NCities NCols]=size(A);
A=[A zeros(NCities,1)];
SumDis1=0;

if Fac(1,1)<0.5
    Fac(1,1)=0.5;
end
if Fac(2,1)<0.5
    Fac(2,1)=0.5;
end
if Fac(3,1)<5
    Fac(3,1)=5;
end
if Fac(4,1)<0.5
    Fac(4,1)=0.5;
end
if Fac(5,1)<5
    Fac(5,1)=5;
end

out2=0;
while out2==0
    Attr=zeros(NCities,1);
    EDT=zeros(NCities,1);
    DisArr=EDT;
    for j=1:NCities
        if j~CurrentCity

            %calc attractiveness
            x=A(j,1);
            y=A(j,2);
            Ld1=A(j,3);
            T1=A(j,4);
            T2=A(j,5);

            Dist1=sqrt((x-A(CurrentCity,1))^2+(y-A(CurrentCity,2))^2);
            DisArr(j,1)=Dist1;
            TDist1=T1-CurrentTime;
            if TDist1<0
                TDist1=0;
            end
            TTime=Dist1;
DDist1=sqrt((x-A(1,1))^2+(y-A(1,2))^2);
DTime=DDist1;

% can we get there on time?
% is load OK?
% can we get back to depot on time?

ArrTime=CurrentTime+TTime;
if ArrTime<T1
    ArrTime=T1;
end
EDT(j,1)=ArrTime+TAT;

RetTime=EDT(j,1)+DTime;

if EDT(j,1)<T2 && CurrentLoad+Ld1<=VCap && RetTime<=DayEnd && (A(j,7)==0 || j==1)

    % Bigger loads - more attractive
    Close1=Ld1/BiggestLoad*Fac(1,1); % Fac(1) 1
    if Close1<0.000001
        stop1=1;
    end

    % closer - more attractive
    Close2=Fac(2,1)*exp(-DDist1/Fac(3,1)); %Fac(2)~1.5 Fac(3)~10

    % closer in time - more attractive
    TDiff=EDT(j,1)-CurrentTime;
    Close3=Fac(4,1)*exp(-TDiff/Fac(5,1)); %Fac(4)~1.5 Fac(5)~10

    Attr(j,1)=Close1+Close2+Close3;
    if j==1
        Attr(j,1)=0;
    end
end

end
end

AllZero=1;
for j=1:NCities
    if Attr(j,1)>0.00000001
        AllZero=0;
    end
end

if AllZero==1
    ZCnt=ZCnt+1;
end
Attr(1,1)=0.1;
AllZero=0;
end

if AllZero==0
%choose city

if Fix==0
ZCnt=0;

Sum1=0;
for j=1:NCities
    Sum1=Sum1+Attr(j,1);
end
R1=rand();
out1=0;
j=0;
Cum1=0;
while out1==0
    j=j+1;
    Cum1=Cum1+Attr(j,1)/Sum1;
    if R1<=Cum1
        out1=1;
    end
    if j>=NCities
        out1=1;
    end
end

else %Fix=1 - choose most attractive location

Max1=0.00000001;
Best1=1;
for kk=1:NCities
    if Attr(kk,1)>Max1
        Max1=Attr(kk,1);
        Best1=kk;
    end
end
j=Best1;
end
else 
j=1;
end
NextCity = j;

% Move to next city
CurrentCity = NextCity;

cnt1 = cnt1 + 1;
Route1(cnt1, 1) = CurrentCity;
SumDis1 = SumDis1 + DisArr(j, 1);
Route1(cnt1, 2) = DisArr(j, 1);

Route1(cnt1, 4) = SumDis1;

if CurrentCity ~= 1
CurrentLoad = CurrentLoad + A(CurrentCity, 3);

CurrentTime = EDT(j, 1);
Route1(cnt1, 3) = CurrentTime;
else
    CurrentLoad = 0;
    CurrentTime = 0;
    SumDis1 = 0;
    Route1(cnt1, 1) = CurrentCity;
    Route1(cnt1, 2) = DisArr(j, 1);
    Route1(cnt1, 4) = SumDis1;
    Route1(cnt1, 3) = CurrentTime;

    if A(1, 7) == 1
        A(1, 7) = 0;
    end
end
A(CurrentCity, 7) = 1;

AllDone = 1;
for j = 1:NCities
    if A(j, 7) == 0
        AllDone = 0;
    end
end
if AllDone == 1
    out2 = 1;
end
end

if ZCnt>100
    out2=1;
end
end

[r1 c1]=size(Route1);

SumDist=0;
for i=2:r1
    Dist1=sqrt((A(Route1(i-1,1),1)-A(Route1(i,1),1))^2+(A(Route1(i-1,1),2)-A(Route1(i,1),2))^2);
    SumDist=SumDist+Dist1;
end

if ZCnt>100
    SumDist=100000;
end

A(:,7)=[];

%fn_agent=SumDist;
z=SumDist;
tf=isnumeric(z);
if tf==0
    z=[];
    z=100001;
end
[r1 c1]=size(z);
if r1>1 || c1>1
    z=[];
    z=100002;
end

xlswrite(strcat('routing',num2str(num1),'.xlsx'),Route1);
APPENDIX I  CVRPTW Data
<table>
<thead>
<tr>
<th>x position</th>
<th>y position</th>
<th>load</th>
<th>starting time window</th>
<th>ending time window</th>
<th>turnaround time</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>50</td>
<td>0</td>
<td>0</td>
<td>240</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>85</td>
<td>20</td>
<td>145</td>
<td>175</td>
<td>10</td>
</tr>
<tr>
<td>22</td>
<td>75</td>
<td>30</td>
<td>50</td>
<td>80</td>
<td>10</td>
</tr>
<tr>
<td>22</td>
<td>85</td>
<td>10</td>
<td>109</td>
<td>139</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>80</td>
<td>40</td>
<td>141</td>
<td>171</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>85</td>
<td>20</td>
<td>41</td>
<td>71</td>
<td>10</td>
</tr>
<tr>
<td>18</td>
<td>75</td>
<td>20</td>
<td>95</td>
<td>125</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>75</td>
<td>20</td>
<td>79</td>
<td>109</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>80</td>
<td>10</td>
<td>91</td>
<td>121</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>35</td>
<td>20</td>
<td>91</td>
<td>121</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>40</td>
<td>30</td>
<td>119</td>
<td>149</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
<td>40</td>
<td>59</td>
<td>89</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>45</td>
<td>20</td>
<td>64</td>
<td>94</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>35</td>
<td>10</td>
<td>142</td>
<td>172</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>45</td>
<td>10</td>
<td>35</td>
<td>65</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>20</td>
<td>58</td>
<td>88</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>40</td>
<td>20</td>
<td>72</td>
<td>102</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>45</td>
<td>20</td>
<td>149</td>
<td>179</td>
<td>10</td>
</tr>
<tr>
<td>44</td>
<td>5</td>
<td>20</td>
<td>87</td>
<td>117</td>
<td>10</td>
</tr>
<tr>
<td>42</td>
<td>10</td>
<td>40</td>
<td>72</td>
<td>102</td>
<td>10</td>
</tr>
<tr>
<td>42</td>
<td>15</td>
<td>10</td>
<td>122</td>
<td>152</td>
<td>10</td>
</tr>
<tr>
<td>40</td>
<td>5</td>
<td>10</td>
<td>67</td>
<td>97</td>
<td>10</td>
</tr>
<tr>
<td>40</td>
<td>15</td>
<td>40</td>
<td>92</td>
<td>122</td>
<td>10</td>
</tr>
<tr>
<td>38</td>
<td>5</td>
<td>30</td>
<td>65</td>
<td>95</td>
<td>10</td>
</tr>
<tr>
<td>38</td>
<td>15</td>
<td>10</td>
<td>148</td>
<td>178</td>
<td>10</td>
</tr>
<tr>
<td>35</td>
<td>5</td>
<td>20</td>
<td>154</td>
<td>184</td>
<td>10</td>
</tr>
<tr>
<td>95</td>
<td>30</td>
<td>30</td>
<td>115</td>
<td>145</td>
<td>10</td>
</tr>
<tr>
<td>95</td>
<td>35</td>
<td>20</td>
<td>62</td>
<td>92</td>
<td>10</td>
</tr>
<tr>
<td>92</td>
<td>30</td>
<td>10</td>
<td>62</td>
<td>92</td>
<td>10</td>
</tr>
<tr>
<td>90</td>
<td>35</td>
<td>10</td>
<td>67</td>
<td>97</td>
<td>10</td>
</tr>
<tr>
<td>88</td>
<td>30</td>
<td>10</td>
<td>74</td>
<td>104</td>
<td>10</td>
</tr>
<tr>
<td>88</td>
<td>35</td>
<td>20</td>
<td>61</td>
<td>91</td>
<td>10</td>
</tr>
<tr>
<td>87</td>
<td>30</td>
<td>10</td>
<td>131</td>
<td>161</td>
<td>10</td>
</tr>
<tr>
<td>85</td>
<td>25</td>
<td>10</td>
<td>51</td>
<td>81</td>
<td>10</td>
</tr>
<tr>
<td>85</td>
<td>35</td>
<td>30</td>
<td>111</td>
<td>141</td>
<td>10</td>
</tr>
<tr>
<td>67</td>
<td>85</td>
<td>20</td>
<td>139</td>
<td>169</td>
<td>10</td>
</tr>
<tr>
<td>65</td>
<td>85</td>
<td>40</td>
<td>43</td>
<td>73</td>
<td>10</td>
</tr>
<tr>
<td>65</td>
<td>82</td>
<td>10</td>
<td>124</td>
<td>154</td>
<td>10</td>
</tr>
<tr>
<td>62</td>
<td>80</td>
<td>30</td>
<td>75</td>
<td>105</td>
<td>10</td>
</tr>
<tr>
<td>60</td>
<td>80</td>
<td>10</td>
<td>37</td>
<td>67</td>
<td>10</td>
</tr>
<tr>
<td>60</td>
<td>85</td>
<td>30</td>
<td>85</td>
<td>115</td>
<td>10</td>
</tr>
<tr>
<td>58</td>
<td>75</td>
<td>20</td>
<td>92</td>
<td>122</td>
<td>10</td>
</tr>
<tr>
<td>55</td>
<td>80</td>
<td>10</td>
<td>33</td>
<td>63</td>
<td>10</td>
</tr>
<tr>
<td>55</td>
<td>85</td>
<td>20</td>
<td>128</td>
<td>158</td>
<td>10</td>
</tr>
<tr>
<td>55</td>
<td>82</td>
<td>10</td>
<td>64</td>
<td>94</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>82</td>
<td>10</td>
<td>37</td>
<td>67</td>
<td>10</td>
</tr>
<tr>
<td>18</td>
<td>80</td>
<td>10</td>
<td>113</td>
<td>143</td>
<td>10</td>
</tr>
<tr>
<td>x position</td>
<td>y position</td>
<td>load</td>
<td>starting time window</td>
<td>ending time window</td>
<td>turnaround time</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>------</td>
<td>----------------------</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>2</td>
<td>45</td>
<td>10</td>
<td>45</td>
<td>75</td>
<td>10</td>
</tr>
<tr>
<td>42</td>
<td>5</td>
<td>10</td>
<td>151</td>
<td>181</td>
<td>10</td>
</tr>
<tr>
<td>42</td>
<td>12</td>
<td>10</td>
<td>104</td>
<td>134</td>
<td>10</td>
</tr>
<tr>
<td>72</td>
<td>35</td>
<td>30</td>
<td>116</td>
<td>146</td>
<td>10</td>
</tr>
<tr>
<td>55</td>
<td>20</td>
<td>19</td>
<td>83</td>
<td>113</td>
<td>10</td>
</tr>
<tr>
<td>25</td>
<td>30</td>
<td>3</td>
<td>52</td>
<td>82</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>50</td>
<td>5</td>
<td>91</td>
<td>121</td>
<td>10</td>
</tr>
<tr>
<td>55</td>
<td>60</td>
<td>16</td>
<td>139</td>
<td>169</td>
<td>10</td>
</tr>
<tr>
<td>30</td>
<td>60</td>
<td>16</td>
<td>140</td>
<td>170</td>
<td>10</td>
</tr>
<tr>
<td>50</td>
<td>35</td>
<td>19</td>
<td>130</td>
<td>160</td>
<td>10</td>
</tr>
<tr>
<td>30</td>
<td>25</td>
<td>23</td>
<td>96</td>
<td>126</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>20</td>
<td>152</td>
<td>182</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>19</td>
<td>42</td>
<td>72</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>60</td>
<td>17</td>
<td>155</td>
<td>185</td>
<td>10</td>
</tr>
<tr>
<td>45</td>
<td>65</td>
<td>9</td>
<td>66</td>
<td>96</td>
<td>10</td>
</tr>
<tr>
<td>65</td>
<td>35</td>
<td>3</td>
<td>52</td>
<td>82</td>
<td>10</td>
</tr>
<tr>
<td>65</td>
<td>20</td>
<td>6</td>
<td>39</td>
<td>69</td>
<td>10</td>
</tr>
<tr>
<td>45</td>
<td>30</td>
<td>17</td>
<td>53</td>
<td>83</td>
<td>10</td>
</tr>
<tr>
<td>35</td>
<td>40</td>
<td>16</td>
<td>11</td>
<td>41</td>
<td>10</td>
</tr>
<tr>
<td>41</td>
<td>37</td>
<td>16</td>
<td>133</td>
<td>163</td>
<td>10</td>
</tr>
<tr>
<td>64</td>
<td>42</td>
<td>9</td>
<td>70</td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>40</td>
<td>60</td>
<td>21</td>
<td>144</td>
<td>174</td>
<td>10</td>
</tr>
<tr>
<td>31</td>
<td>52</td>
<td>27</td>
<td>41</td>
<td>71</td>
<td>10</td>
</tr>
<tr>
<td>35</td>
<td>69</td>
<td>23</td>
<td>180</td>
<td>210</td>
<td>10</td>
</tr>
<tr>
<td>65</td>
<td>55</td>
<td>14</td>
<td>65</td>
<td>95</td>
<td>10</td>
</tr>
<tr>
<td>63</td>
<td>65</td>
<td>8</td>
<td>30</td>
<td>60</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>60</td>
<td>5</td>
<td>77</td>
<td>107</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>8</td>
<td>141</td>
<td>171</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>16</td>
<td>74</td>
<td>104</td>
<td>10</td>
</tr>
<tr>
<td>60</td>
<td>12</td>
<td>31</td>
<td>75</td>
<td>105</td>
<td>10</td>
</tr>
<tr>
<td>23</td>
<td>3</td>
<td>7</td>
<td>150</td>
<td>180</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>56</td>
<td>27</td>
<td>90</td>
<td>120</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>68</td>
<td>30</td>
<td>89</td>
<td>119</td>
<td>10</td>
</tr>
<tr>
<td>47</td>
<td>47</td>
<td>13</td>
<td>192</td>
<td>222</td>
<td>10</td>
</tr>
<tr>
<td>49</td>
<td>58</td>
<td>10</td>
<td>86</td>
<td>116</td>
<td>10</td>
</tr>
<tr>
<td>27</td>
<td>43</td>
<td>9</td>
<td>42</td>
<td>72</td>
<td>10</td>
</tr>
<tr>
<td>37</td>
<td>31</td>
<td>14</td>
<td>35</td>
<td>65</td>
<td>10</td>
</tr>
<tr>
<td>57</td>
<td>29</td>
<td>18</td>
<td>96</td>
<td>126</td>
<td>10</td>
</tr>
<tr>
<td>63</td>
<td>23</td>
<td>2</td>
<td>87</td>
<td>117</td>
<td>10</td>
</tr>
<tr>
<td>21</td>
<td>24</td>
<td>28</td>
<td>87</td>
<td>117</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>24</td>
<td>13</td>
<td>90</td>
<td>120</td>
<td>10</td>
</tr>
<tr>
<td>24</td>
<td>58</td>
<td>19</td>
<td>67</td>
<td>97</td>
<td>10</td>
</tr>
<tr>
<td>67</td>
<td>5</td>
<td>25</td>
<td>144</td>
<td>174</td>
<td>10</td>
</tr>
<tr>
<td>37</td>
<td>47</td>
<td>6</td>
<td>86</td>
<td>116</td>
<td>10</td>
</tr>
<tr>
<td>49</td>
<td>42</td>
<td>13</td>
<td>167</td>
<td>197</td>
<td>10</td>
</tr>
<tr>
<td>53</td>
<td>43</td>
<td>14</td>
<td>14</td>
<td>44</td>
<td>10</td>
</tr>
<tr>
<td>61</td>
<td>52</td>
<td>3</td>
<td>178</td>
<td>208</td>
<td>10</td>
</tr>
</tbody>
</table>