Master's degree thesis

LOG950 Logistics

Distance based decision support for multi-objective VRP

Jorge Luis Oyola Mendoza

Number of pages including this page: 84

Molde, 25 May 2010

Molde University College

Publication agreement

Title: Distance based decision support for multi-objective VRP

Author(s): Jorge Luis Oyola Mendoza

Subject code: LOG950

ECTS credits: 15

Year: 2010

Supervisor: Arne Løkketangen

Agreement on electronic publication of master thesis

Author(s) have copyright to the thesis, including the exclusive right to publish the document (The Copyright Act §2).
All theses fulfilling the requirements will be registered and published in Brage HiM, with the approval of the author(s).
Theses with a confidentiality agreement will not be published.

I/we hereby give Molde University College the right to, free of charge, make the thesis available for electronic publication: ☒**yes** ☐**no**

Is there an agreement of confidentiality? ☐**yes** ☒**no**
(A supplementary confidentiality agreement must be filled in)
- If yes: **Can the thesis be online published when the period of confidentiality is expired?** ☐**yes** ☐**no**

Date: 25 May 2010

Preface

This thesis is submitted as partial fulfillment of the requirements for the Master's degree in Logistics at Molde University College, Molde, Norway.

It contains a report of the work done during the spring 2010. Professor Arne Løkketangen has been my supervisor. In addition I received help and advice from Professor Johan Oppen, who implemented the original solver that I extended to include some particular aspects required for this research.

Doing this work demanded from me to learn a lot of new things. Professors Løkketangen and Oppen were always available to answer any question that I brought.

The topic of this thesis is the attribute based distance measure between solutions to the vehicle routing problem with route balancing; and how this can support the decisions of a planner by providing him/her with a set of different solutions.

I would like to thank Professor Arne Løkketangen for his patience when working with me. His constructive criticism and suggestions made this work far better than what it would be without him. I am also grateful for all the help and advices that I received from Professor Johan Oppen.

I will always appreciate the support received from the Benitez Londoño Family. Without their help, at that time, I would not come this far.

I also wish to thank Jana; her support has made things goes easier. To my friends who even at the distance have made me feel close to them. To my family, I miss them too. And to Lidis, of course.

Jorge Oyola
Molde, Norway
May 2010

Summary

This master thesis deals with an extension of the vehicle routing problem (VRP), including the balance of the tours length as an additional objective. This particular problem is called vehicle routing problem with route balancing (VRPRB). In general VRPs with more than one objective, multi-objective, describe better real life situations. In the case of the route balancing, legal restrictions or union agreements can limit the differences in the time assigned to the drivers. As part of a Decision Support System, a VRP solver should be able to provide the planner with several different solutions to choose from. An attribute based distance measure between solutions is implemented to quantify how different two solutions are.

An existing VRP solver is extended to include the multi-objective approach to the problem and the attribute based distance measure between solutions. Instances from the literature are tested. It was found that including an extra objective can have a significant cost. Results shows that there is a low/medium correlation between the values from the attribute based distance measure and the difference in the relative importance of the objectives in the solutions, the first being a better way to measure the difference between two solutions. The possibility of providing the planner with dominated solutions was evaluated; finding that in such case, the decision maker will have a more diverse set of solutions to choose from.

Contents

1. Introduction

1.1 Research problem

The purpose of this master thesis is to implement parts of a decision support system (DSS) for the vehicle routing problem, in a multi-objective approach. Two different objectives will be considered. One of them is minimizing the total length which comes from the traditional vehicle routing problem (VRP). The second objective is minimizing the variation in the route lengths. This second objective could be measured by the difference between the longest and the shortest tour and the summation of the square differences between every route length and the average length. The importance associated with each objective will depend on the policy of the decision maker.

A DSS should give the planner different good alternatives for his/her consideration (Marakas 2003, pp 3-6). For the case of the VRP, the idea is to provide the decision maker with structurally different solutions, where the quality level is above a certain threshold, evaluated using the objective function. The measure of how different the solutions are will not depend on the quality, since once they are above the threshold, they are considered good enough. Part of the research will be focused on finding a way of measuring the distance between two solutions in the solution space, considering only the attributes of each solution. In addition, the possibility of providing the decision maker with a solution that is not necessary the best option to select in a multi-objective framework will be evaluated, taking into consideration the diversity of the provided solutions.

The VRP is known to be NP-hard (Toth and Vigo 2002a), so sacrificing optimality in order to find a solution in a "reasonable" time becomes a practical option, and this is achieved by using heuristics methods. Here a variation of the "unified tabu search heuristic for vehicle routing problems with time windows" proposed by (Cordeau, Laporte et al. 2001) is going to be used. This heuristic has been implemented in a VRP solver, with some adjustments and changes, at Molde University College (HiM) by Professor Johan Oppen. The purpose of this research is adding the multi-objective approach and the similarity measure to Professor Oppen's solver.

The test cases used in this research are instances from the literature for the capacitated VRP (CVRP), that is, instances where the vehicle capacity is taken into consideration. In addition the distance between customers is Euclidean. Most of test cases were used before by Professor Oppen. These instances can be found at http://branchandcut.org/VRP/data.

1.2 Tools used

The VRP solver described in this master thesis is based on another solver originally implemented in C++ using Microsoft Visual Studio .NET 2003, by Professor Johan Oppen (Oppen 2004). It has been modified using Microsoft Visual Studio 2008. SVG was used to generate plots of the instances and solutions. SPSS was used for statistical analysis and generating data figures. Microsoft Visio was used for drawing the conceptual model of the solver. The tests were done in regular network PC's at Molde University College.

1.3 Outline of the thesis

The rest of this thesis is divided as follows. In Chapter 2 the classical Vehicle Routing Problem is presented, including some solution methods. Chapter 3 describes the multi-objective optimization problem together with some solution methods. Then, Chapter 4 introduces the distance measure between two solutions. The tabu search heuristic implemented in this master thesis is presented in Chapter 5, followed by Chapter 6 which presents the computational experiments and the obtained results. Finally, Chapter 7 outlines the conclusions and further research in this field.

2. The Vehicle Routing Problem

2.1 The classical VRP

The definition for the VRP used in this research is based on the definition given by (Cordeau, Gendreau et al. 2002) for the case in which the cost matrix is symmetric.

- Given an undirected graph $G=\{V,E\}$, where $V=\{v_0,v_1,...,v_n\}$ is the vertex set, and $E=\{(v_i,v_j):v_i,v_j \in V.i<j\}$ is the edge set.
- v_0 represents the depot, and the other vertices represent the customers, each having a non-negative demand q_i.
- The set E has an associated cost matrix c_{ij}, representing the cost of travelling from vertex i to vertex j $(c_{ij}=c_{ji})$.
- A fleet of m vehicles with equal capacity Q is based at the depot.

The VRP then consists of designing m delivery routes with some constraints:

1. Each route starts and ends at the depot.
2. Each customer is visited once by exactly one vehicle.
3. $\forall i, q_i \leq Q$.
4. The total cost (length) is minimized.

Some authors call the previous definition for the capacitated vehicle routing problem (CVRP) (Toth and Vigo 2002a), because of the capacity constraint of the vehicles. So far just one single objective function has been defined. This will be extended later on.

An example of a VRP is shown in Figure 2.1, which represents the instance A-n32-k5, with one depot, in red, and the other points represent a set of 31 customers. The distance between any pair of customers is Euclidean, the length of a straight line that connects them. The Figure 2.2 shows a feasible solution for the instance mentioned above.

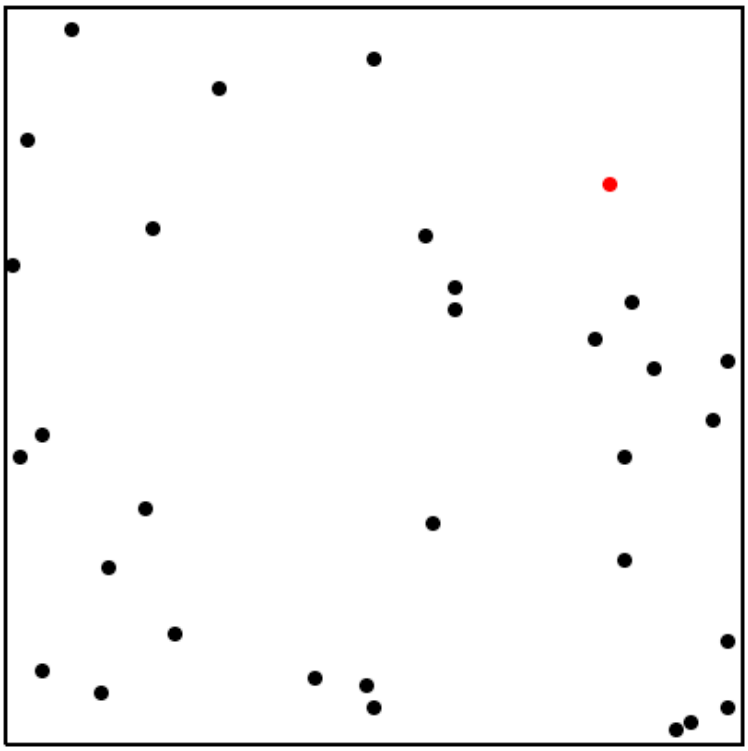

Figure 2.1 Typical VRP used in this research.

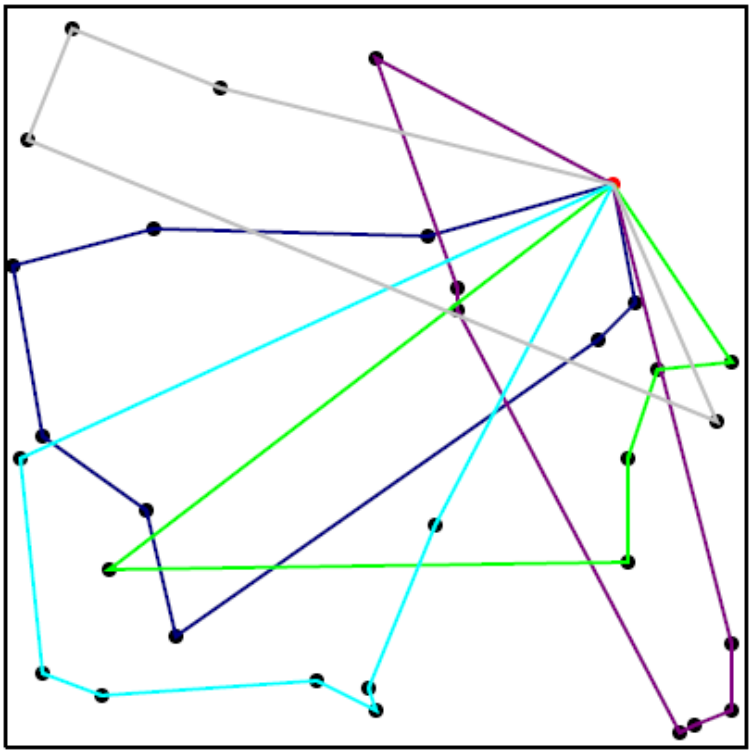

Figure 2.2 Feasible solution of a VRP instance.

2.2 Solution methods for VRPs

2.2.1 Exact methods

There are exact methods able to find the optimal solution of a VRP. But for large VRP instances, with more than 50 customers generally those methods do not perform well (Oppen and Løkketangen 2006), either because they take too long to find the solution or because they need more memory than the available capacity. However nowadays researchers talk about instances with up to 70 customers where exact methods still have a good performance.

A DSS should take into account that in the decision making process, the available processing time is short (Marakas 2003, p 4). There is a large number of exact methods for the VRP, but in general, they can be classified into three categories: direct tree search methods, dynamic programming and integer linear programming (Laporte 1992). Branch and bound algorithms are classified as part of the direct tree search methods (Toth and Vigo 2002a).

Solving the mixed integer programming (MIP) model for the VRP, presented by (Oppen and Løkketangen 2006), requires the use of exact methods. This model is presented here as an example of where these methods can be used.

$$\sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}U_{ij} \qquad (1)$$

Subject to

$$\sum_{j=0}^{n} U_{ij} = 1, i = 1,...,n \qquad (2)$$

$$\sum_{j=0}^{n} U_{ij} = \sum_{j=0}^{n} U_{ji}, i = 0,...,n \qquad (3)$$

$$(1-U_{ij})M + Y_i \geq Y_j + q_j, i = 0,...,n,\ j = 0,...,n \qquad (4)$$

$$0 \leq Y_i \leq Q, i = 0,...,n \qquad (5)$$

$$U_{ji} \in \{0,1\}, i = 0,...,n,\ j = 0,...,n \qquad (6)$$

$$\sum_{j=1}^{n} U_{0j} \leq m \qquad (7)$$

Where n is the number of customers, the vertex 0 is the depot; c_{ij} is the distance from vertex i to vertex j; U_{ji} is the decision variable, it will take the value 1 if the edge (i, j) is used in the solution, 0 otherwise; q_j is the demand of vertex j; Yi is the load of the vehicle when it leaves the vertex i; Q is the capacity of the vehicles; m is the number of vehicles; and M is a big number.

The equation (1) is the objective function, minimizing the total length; the equation (2) ensures that all the vertices, except the depot, are left just once; the equation (3) ensures that all vertices are entered the same number of time that they are left, this includes the depot; the equation (4) ensures that sub tours are avoided, because of M, the inequality will hold for all the edges that are not traversed, for those that are traversed the equality will hold, the load that arrives will be equal to the demand of the vertex plus the load that leaves; the equation (5) ensures that the load of the vehicle will not be higher than its capacity; equation (6) specify that the decision variable should be binary; and the equation (7) establish the number or routes that should be included in the solution.

2.2.2 Heuristic methods

The heuristics methods can not guarantee any quality level on the found solution. But on the other hand, many of them are known because they are able to find a good solution in a short time (Oppen and Løkketangen 2006). The Heuristics used for solving VRP are often derived from the Travelling Salesman Problem (TSP) (Laporte 1992). Two types of heuristics methods can be considered, classical heuristics and metaheuristics (Toth and Vigo 2002b, p 109).

Classical heuristics methods

The classical heuristics methods perform a limited search, but the time consumption is quite modest, in addition they can be extended to include real life constraints. They can be classified into three categories: constructive heuristics, two phase heuristics and improvement methods (Toth and Vigo 2002b, pp 109-110).

Constructive heuristics gradually build up a solution, while the cost is minimized. The Clarke and Wright algorithm (Savings algorithm) is an example of this type of heuristics (Toth and Vigo 2002b, pp 110-113). The two phase algorithms work to find a solution in two different phases, clustering and routing. Christofides – Mingozzi – Toth algorithm can be classified into this category (Laporte 1992). The improvement methods take an initial feasible solution and try to improve it by means of exchanging vertices or edges. This process can be done within the tours, single-route improvements, or among several tours, multiroute improvements (Toth and Vigo 2002b, pp 121-125).

Metaheuristic methods

The term metaheuristics was introduced during the 1980s. This denomination includes a large number of search methods, among those are simulated annealing, tabu search, genetic algorithms, scatter search and neural networks (Eksioglu, Vural et al. 2009). Those methods can eventually escape from a local optimum; however this does not ensure that the global optimum will be found.

There are basically three types of metaheuristic methods. Some methods are based on local search, like tabu search or simulated annealing, where the search is done starting from a initial solution and moving to another solution in the neighbourhood. Other methods are based on populations of solutions, like genetic algorithms, where the idea is to combine solutions. There are some hybrid methods, which are a combination of search methods, e.g. using a local search algorithm to improve the quality of a particular solution, an individual of the population, as memetic algorithms do (Sörensen and Sevaux 2006).

3. Multi-objective optimization

In a multi-objective problem several functions are optimized (minimized or maximized) subject to the same set of constraints. This problem can be stated as $\min F(x)=(f_1(x), f_2(x),..., f_n(x)), s.t. x \in D$, with the number of objective functions being $n \geq 2$; the decision variable vector $x=(x_1, x_2,..., x_n)$; the feasible solution space D; and $F(x)$ is the objective vector (Jozefowiez, Semet et al. 2007). A solution of the problem is given by $y=(y_1, y_2,..., y_n)$, where $y_i = f_i(x)$.

3.1 Pareto optimal solutions

The different objective functions in a multi-objective problem are usually conflicting. Because of this, it is common that not a single optimal solution is able to minimize all the objectives of the problem. Instead of that, a set of solutions is found, which are not expected to be optimal. They will minimize some objectives and will not necessarily be optimal in the others, those solutions are called tradeoff solutions (Collette and Siarry 2003, p 21).

The Pareto optimal solutions are obtained using the concept of domination, which was defined in (Jozefowiez, Semet et al. 2007). A solution $y=(y_1, y_2,..., y_n)$ dominates a solution $z=(z_1, z_2,..., z_n)$, if and only if $\forall i \in \{1, 2,..., n\}\ y_i \leq z_i$, and $\exists j \in \{1, 2,..., n\}$, such that $y_i < z_i$. That is, the solution z is not better than y for any objective function, but it is worse for at least one. The non dominated solutions, Pareto set or Pareto optimal solutions define the solution of a multi-objective optimization problem (Jozefowiez, Semet et al. 2007).

When dealing with metaheuristics, this is when there is no guarantee of finding the optimum, one more concept should be included, the potentially Pareto optimal (PPS). A solution y found by a particular algorithm A is considered potentially Pareto optimal, relative to A, if that algorithm does not find a different solution z that dominates y (Jozefowiez, Semet et al. 2007).

3.2 Solution methods

There are several methods for solving multi-objective optimization problems, as well as classifications for these methods. One possible classification is done according to the characteristics of the method itself: *scalar methods*, *Pareto methods* and *non-scalar, non-Pareto methods* (Jozefowiez, Semet et al. 2008). In addition, if the interaction of the decision maker with the method is considered, then the classification categories would be *a priori*, *progressive* and *a posteriori* methods (Collette and Siarry 2003, p 81). A different type of methods for solving multi-objective optimization problems is *fuzzy methods* (Collette and Siarry 2003, p 105).

3.2.1 Scalar methods

These methods are based on mathematical transformation. The most popular method has been weighted linear aggregation. This is because when the different objectives are aggregated, the result is a new single objective, so the regular methods for the single objective optimization problems can be used with minor modifications (Jozefowiez, Semet et al. 2008).

The weighted linear aggregation method, also known as weighted sum of objective functions method, has some disadvantages (Jozefowiez, Semet et al. 2008). First, the weights show how important one objective is compared with the others, which is not always easy to establish. On the other hand, the method is not able to find all the Pareto optimal solutions, since it can not discover the solutions located in the concavities of the feasible set (Collette and Siarry 2003, p 48). Figure 3.1 illustrates this drawback. But this method has an important advantage, it is relatively simple to implement (Jozefowiez, Semet et al. 2008). In addition, as mentioned before, it allows single objective optimization methods to be used.

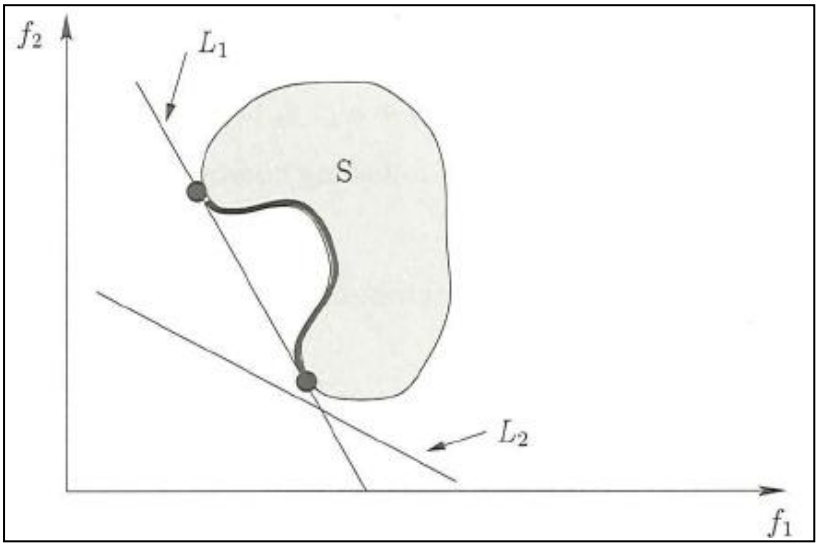

Figure 3.1 Weighted sum of objective function method (Collette and Siarry 2003, p 49).

The goal programming and ϵ-constraint methods are also within this category of methods (Jozefowiez, Semet et al. 2008). In the goal programming methods, a goal is selected, this is a point in the objective space, and then the idea is to minimize the distance between the current solution and the goal. Here the disadvantage comes from establishing the goal, however it has the advantage that it works on concave feasible sets (Collette and Siarry 2003, p 64). In the case of the ϵ-constraint methods, the idea is optimize one of the objectives and the others are considered as constraints $f_i(x) \leq \epsilon_i$. A set of solutions is obtained changing the values for ϵ_i (Jozefowiez, Semet et al. 2008).

3.2.2 Pareto methods

Pareto Methods are based on the domination concept explained before (Jozefowiez, Semet et al. 2008). With Pareto optimal selection criteria, it is not possible to favour one objective over the other. The objective when using this approach, is to generate the complete set of Pareto optimal solutions or an approximation to it (Loukil, Teghem et al. 2007). The concept of domination has often been used within an evolutionary framework, genetic algorithms, when dealing with multi-objective problems. Genetic algorithms work with a population (set) of solutions, which suits the goal of the Pareto methods of finding a particular set of solutions (Jaszkiewicz 2002).

Pareto methods, when they are used with evolutionary algorithms or any other population based method, can deal with problems having any number of objectives. So preliminary information about the optimal Pareto sets can be obtained, which is very useful when dealing with real-life problems. The Pareto dominance selection criteria can be used also with memetic and local search algorithms, where the next current solution is selected from the non-dominated solutions in the neighbourhood (Jozefowiez, Semet et al. 2008).

The main difference between the scalar and Pareto methods is that scalar methods are not used to find the Pareto set, neither an approximation to it. However scalar methods can be used to find solutions that eventually will fit in the set of Pareto optimal solutions.

3.2.3 Non-scalar and non-Pareto algorithms

Some multi-objective routing problems have been solved using methods, which are neither scalar nor Pareto, where every objective is considered separately. These methods are usually based on genetic algorithms, lexicographic strategies, ant colony mechanism, or some specific heuristics designed to solve multi-objective problems (Jozefowiez, Semet et al. 2008).

The genetic algorithms are population based algorithms. The lexicographic methods are based on priority values given to the objective functions, and then the problem is optimized in order of priority. After an objective function is optimized it becomes a constraint of the problem and its value can not be changed (Jozefowiez, Semet et al. 2008). The lexicographic method is also considered as part of the scalar methods (Collette and Siarry 2003, p 71).

Ant colony mechanisms imitate the behaviour of a real ant colony when the ants are looking for a food source. The ants secrete pheromones so they communicate with the other members of the colony. In the algorithm the pheromones secreted by some artificial ants communicate the most attractive and most travelled paths (Doerner, Gutjahr et al. 2006). As an example, in the case of bi-objective optimization problems, where the objectives are total mean transit time and the variance in the transit time, one type of pheromone has been used for each objective (Jozefowiez, Semet et al. 2008).

3.2.4 *A priori, progressive* and *a posteriori* methods

This classification is done taking into account the interaction between the decision maker and the optimization method.

If the decision maker, before starting the search, provides the method some initial tradeoff between the objective functions (a policy) and the result of that search is a single solution, then the method will be classified as *a priori* (Collette and Siarry 2003, p 81).

On the other hand, the method is *progressive* if the optimization method allows the decision maker to reorient the search during the process. This could be done changing the tradeoff between the objective functions. STEP method, Jahn method and simplex method (different than linear programming) are part of the progressive methods (Collette and Siarry 2003, p 81).

A posteriori methods do not interact with the decision maker. In contrast with the previous methods, the result of using these is a set of Pareto optimal solutions. The decision maker will be able to compare the solutions and select the one that he/she prefers. Applying an *a posteriori* method can be quite time consuming (Collette and Siarry 2003, p 9).

3.2.5 Fuzzy methods

These are methods that solve multi-objective problems using fuzzy logic. The functions in classical logic works with binary parameters, this is 0 or 1, FALSE or TRUE. In the case of fuzzy logic the functions work with continuous values between 0 and 1. Then the AND function, $A = B\,and\,C$ in classic logic becomes $A = \min(B,C)$ in fuzzy logic. The OR function, $A = B\,or\,C$, becomes $A = \max(B,C)$. Finally the NOT function, $A = not\,B$, becomes $A = 1 - B$. Two examples of these methods are *Sakawa* and *Reardon* methods (Collette and Siarry 2003, pp 105-114).

3.3 Selected method

The multi-objective optimization method used to do the research in this thesis is the "weighted sum of objective functions method". This method was selected for two main

reasons. First because it was possible to use a modified version of the solver that Professor Johan Oppen already implemented. Second, because this method can be easily generalized to include more objective functions.

To each objective function is associate a multiplicative weight. The weighted objectives are added and then a single function is obtained. The process is repeated changing the values of the weights, producing the tradeoff solution set (Collette and Siarry 2003, pp 47-53). This means that the original multi-objective problem $\min F(x)=(f_1(x), f_2(x),..., f_n(x)), s.t. x \in D$, will be transformed into a single objective problem, $\min F_{eq}(x)=(w_1 f_1(x)+w_2 f_2(x)+...+w_n f_n(x)), s.t. x \in D$, where $\sum_{i=1}^{n} w_i = 1$, this new optimization problem with the objective function $F_{eq}(x)$ can be solved with suitable optimization methods, depending on the complexity of the problem. The set of w represents a policy, and shows the relative importance given to each objective.

3.4 Objective functions

Two objective functions are considered in this work; the balance of the tour lengths and the total length. The balance can be measured in two different ways; one option is the difference between the longest and the shortest tour, another is the summation of the square differences between every tour length and the average tour length. In the last option the main purpose is penalizing big differences more than the small ones. This extension of the VRP has been called vehicle routing problem with route balancing (VRPRB) (Jozefowiez, Semet et al. 2007).

When dealing with route balancing it is important to avoid artificial improvements. If the two solutions (a) and (b) in the Figure 3.2 are considered, probably (a) is better balanced than (b). Solution (a) has a tour with a line crossing, which makes it suboptimal. Here the balance will be minimized, trying to avoid suboptimal tours. The minimization of the total tour length will be used as criteria for selecting the position of the customers to be inserted into the tours. In addition a 2-opt local search is applied to each tour.

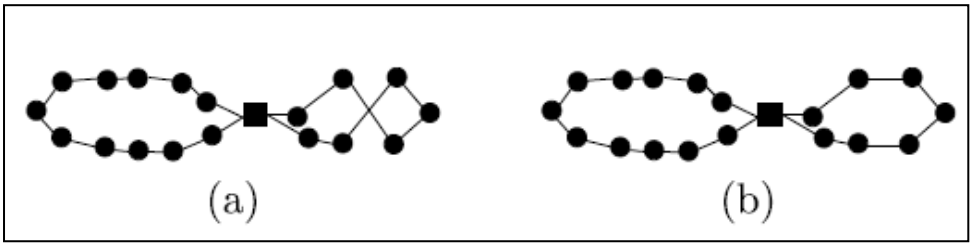

Figure 3.2 Artificial improvement of the route balancing (Jozefowiez, Semet et al. 2007)

Figure 3.3 shows some tradeoff solutions to the instance A-n45-k7 as a CVRPRB. The weights used during the search are between 0 and 1, using a step of 0.1. The weight represents the importance associated to the total length. The difference between 1 and the weight represents, on the other hand, the importance associated to the route balance. Table 3.1 shows the results in more detail. In this case the balance has been measured as the difference between the longest and the shortest tour.

weight	length	1 - weight	balance	weighted objective
0.0	1631	1.0	0	0
0.1	1610	0.9	0	161
0.2	1571	0.8	2	315.8
0.3	1562	0.7	4	471.4
0.4	1563	0.6	4	627.6
0.5	1504	0.5	11	757.5
0.6	1381	0.4	16	835
0.7	1329	0.3	36	941.1
0.8	1146	0.2	138	944.4
0.9	1146	0.1	146	1046
1.0	1155	0.0	148	1155

Table 3.1 Tradeoff solutions to the instance A-n45-k7

Figure 3.4 shows similar results to Figure 3.3, but in this case the step-weight used is 0.01, which results in 101 solutions. The Figure 3.5 shows the result of using 0.001 as step-weight. In both cases, and in the figure 3.3, it is possible to find dominated solutions. This can be explained because of the heuristic, which does not guarantee optimality in the solutions found. However it can be observed that if the step-weight is smaller, more potential Pareto optimal solutions are found and the potential frontier is somehow delineated. In every case the running time was 5 minutes for each weighted search. That is, 55 minutes for the total search if the step is equal to 0.1; 505 minutes when the step is 0.01 and 5005 minutes for a 0.001 step.

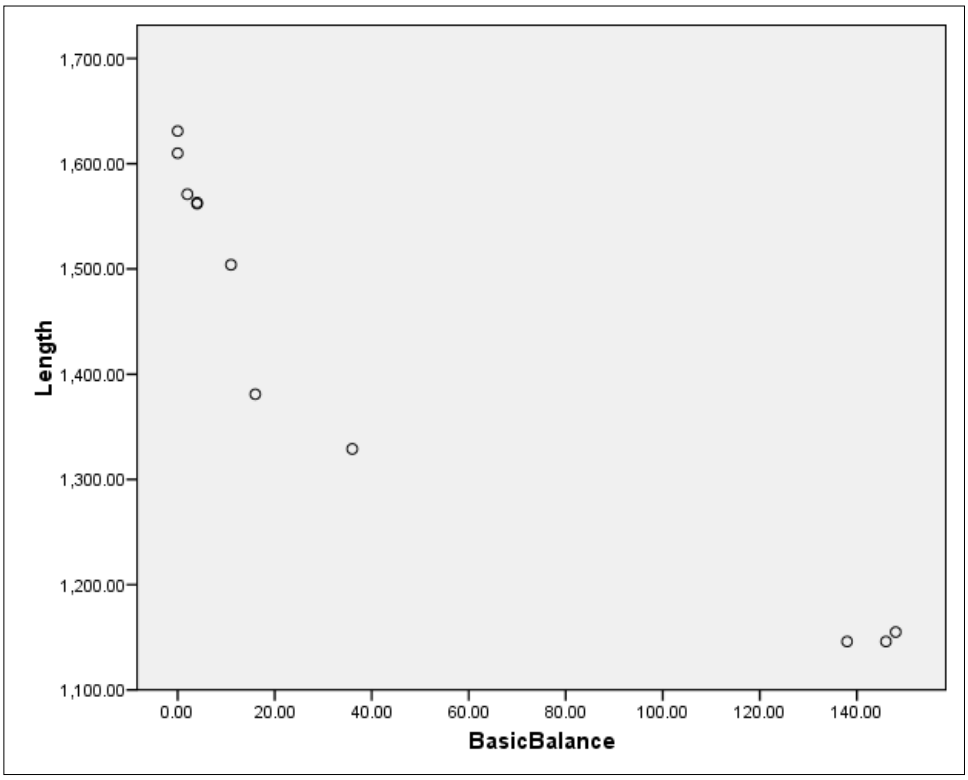

Figure 3.3 Multi-objective solutions to the instance A-n45-k7

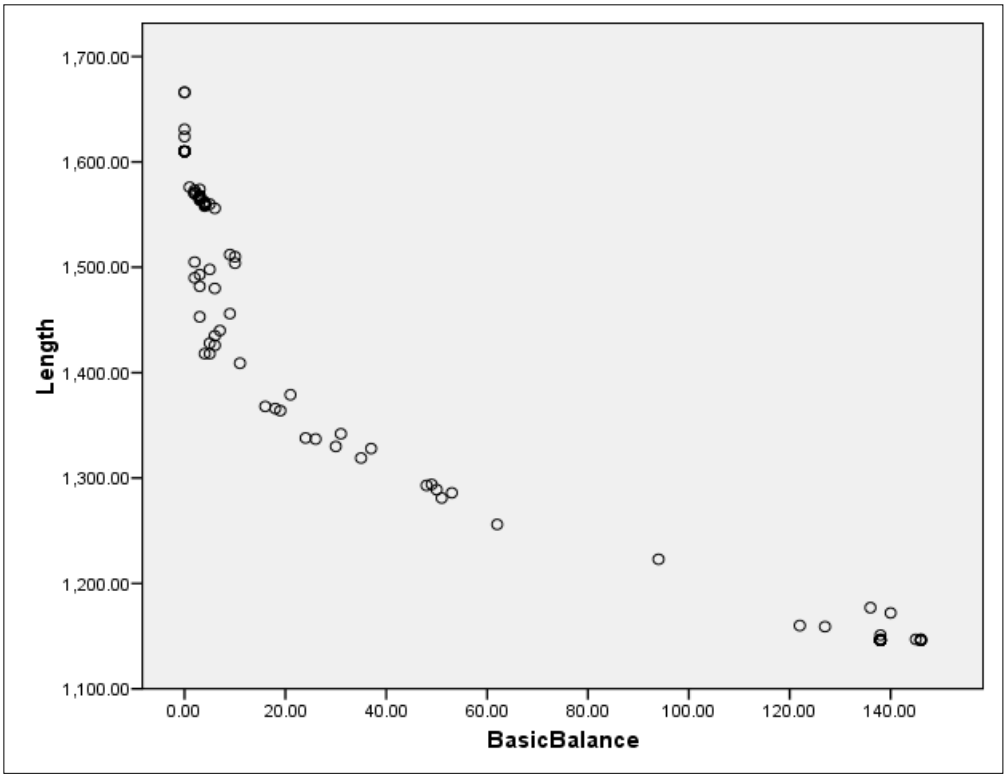

Figure 3.4 Multi-objective solutions to the instance A-n45-k7

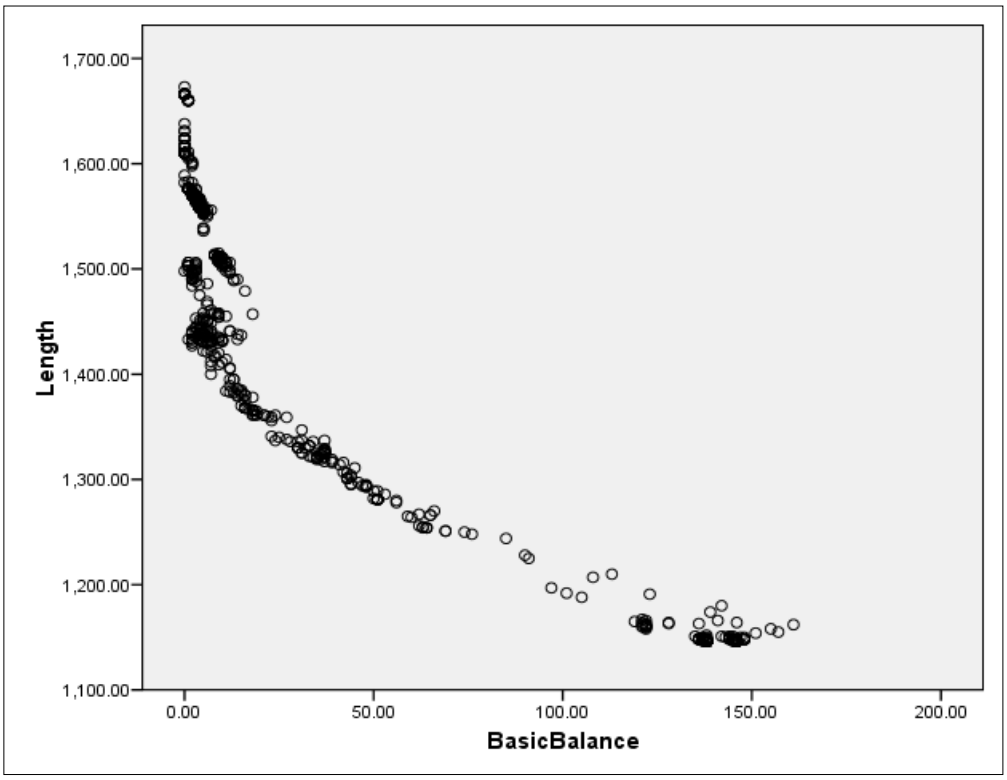

Figure 3.5 Multi-objective solutions to the instance A-n45-k7

The Figure 3.6 shows the tradeoff solutions resulting of doing the same search than the one represented in Figure 3.5, but this time using the square balance as the balance measure during the search. It is possible to see that the solutions are more concentrated in some areas of the trade-off solutions frontier. This could be explained by the deficient normalization of the move evaluation; this concept will be explained in Section 5.1.2.

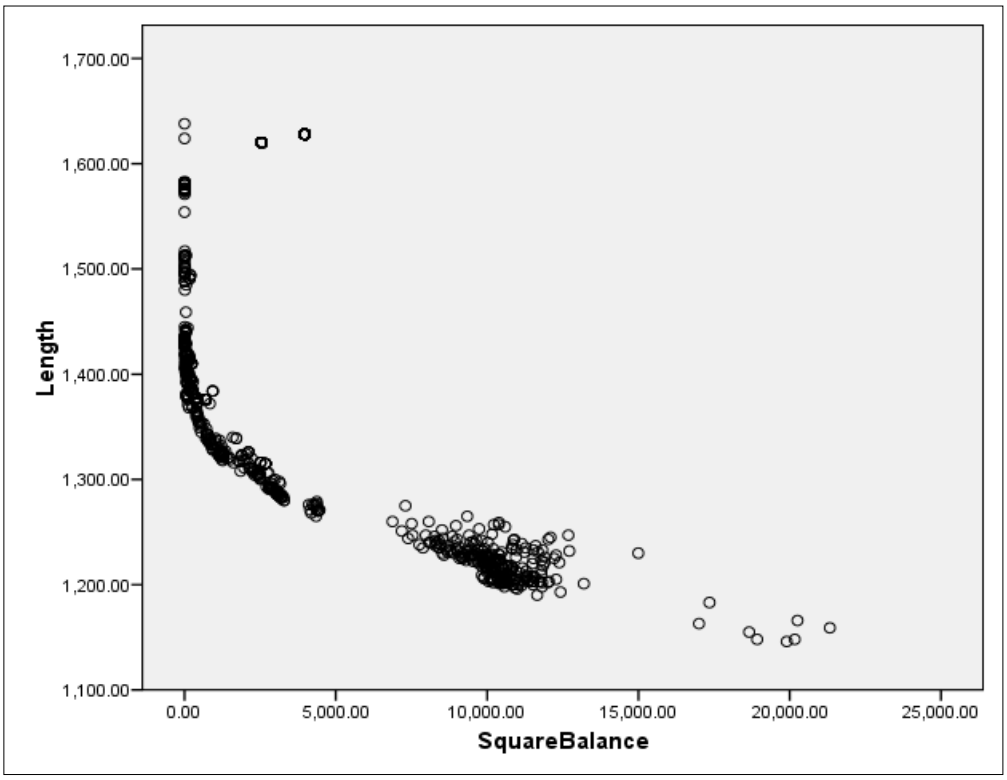

Figure 3.6 Multi-objective solutions to the instance A-n45-k7

4. Distance between solutions

The mathematical model included in a Decision Support System (DSS) is a simplification of the real world, since very often assumptions are made and relevant variables may not be included in the model. Because of this, a good solution given by the DSS will not necessarily fit or perform so well in the real life when it is implemented. As a consequence of this, decisions makers often would like to receive as an output from a DSS a set of different good solutions, instead of a single best solution. The ranking of the first two, or more, best found solutions might not be the better set to show to the decision maker, since these may be very similar. Measure of how different or similar two solutions are should be used (Løkketangen and Woodruff 2005).

The difference between two solutions arranged as binary vectors, total or partially, could be measured using the Hamming distances (Løkketangen and Woodruff 2005). This is the number of elements in which the two solutions are different. Given two solutions $y=(0,1,1,0)$ and $z=(1,0,1,0)$, the Hamming distances will be equal to 2, since the elements if the first and second positions are different in both cases.

The solutions for the VRP are not characterized by having binary vectors. The solutions are heterogeneous and can have different sizes. On the other hand there are some attributes, not easily captured by the Hamming distances (Løkketangen and Woodruff 2005), that makes a solution different from another. e.g. the sequence of stops in a route. Because of that, a different measure has to be used, when the similarity between two different VRP solutions is evaluated.

A VRP solution has some structural properties given by the stops, the arcs and the tours. These properties should be considered by the distance measure between two solutions. Some properties are not necessarily part of the problem; but they can be used just to measure the distance between the solutions. The objective value, on the other hand, should not be taken into consideration, since it is assumed that the solutions that are being compared are all good enough. In the multi-objective approach, and the solutions methods used in this research, some weights are associated to every objective function. These objectives are not part of the structural properties of the solutions, so these weights should not be considered either.

4.1 Attribute based distance measure

For the specific case of two VRP solutions, *x* and *y*, a distance measure with values on [0,1] has been proposed (Oppen 2008, pp 89-108).

$$t(x,y) \equiv \alpha \frac{1}{n}\sum_{i=1}^{n} d(\hat{w};\hat{x}_i,\hat{y}_i) + \beta \frac{1}{n}\sum_{i=1}^{n} d(\tilde{w};\tilde{x}_i,\tilde{y}_i) + \gamma \frac{1}{n}\sum_{i=1}^{n} d(\overline{w};\overline{x}_i,\overline{y}_i)$$

Where α, β and γ are parameters that indicate the relative importance of stops, arcs and tours, respectively. These three parameters should sum up one. $\hat{w}, \tilde{w}$ and $\overline{w}$ are vectors that gives the weights for the different attributes of the stops, arcs and tours respectively. $\hat{x}$ is a *n*-vector of sets that represents the stops in the solution x. $\tilde{x}$ is a *n*-vector of sets that represents the solution x from the arcs point of view. $\overline{x}$ is a *n*-vector correspond to a representation of the tours in x. If in any case the attributes of the stops, arcs or tours have the same weight, then the respective *w* will be a vector of ones, which is the case of this research.

The function $d(\cdot)$ included in the distance measure for two VRP solutions, corresponds to the dissimilarity between two sets of vectors. So given *A* and *B*, two sets of vectors, $d(\cdot)$ function is defined as

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

Where

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

and

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

Given a set *C*, $|C|$ denotes the cardinality of *C*, this is the number of elements of the set. So far the functions considered here compare two set of vectors. But to be able to compute $g(\cdot)$ function, it is required to measure the distance between two vectors $\delta(\cdot)$

$$\delta(w;v,u) \equiv \frac{\sum_{j=1}^{p} \eta_j(v_j,u_j) w_j}{\sum_{j=1}^{p} w_j}$$

Where $\eta_j(\cdot)$ is the distance between the vectors *v* and *u* related to the element *j*. This is

$$\eta_j(v_j,u_j) \equiv \min\left(1, \frac{|v_j - u_j|}{s_j(\cdot)}\right)$$

Here $s_j(\cdot)$ is a measure of the dispersion of the values for the element *j*, this could be the average, the standard deviation or a multiple of this, among others. It is assumed that if $s_j(\cdot)=0$, then $n_j(\cdot)=0$.

When dealing with categorical data, including binary attributes, η_j will be equal to one if values for the element *j* are different, zero otherwise. In case that the element *j* of the vectors correspond to a set, then the Tversky's similarity measure can be used

$$\eta_j \equiv 1 - \frac{|A \cap B|}{|A \cap B| + |A - B| + |B - A|}$$

The distance measure described here is classified as a semi-metric, since it obeys the properties:

1. $t(x,x)=0$,
2. $t(x,y)>0 if\ x \neq y$,
3. $t(x,y)=t(y,x)$

It is not a full metric because the triangle inequality $t(x,z)+t(z,y) \geq t(x,y)$, is not being considered (Løkketangen and Woodruff 2005).

4.2 Solution attributes

The function $t(\cdot)$ that measures the distance between two different solutions, involves the concept of attributes of the stops, arcs and tours (Oppen 2008, pp 89-108). For a particular solution *x*, the attributes of the stops are represented as a vector $\hat{x}$, with a number of elements equal to the number of stops in the solution, without considering the depot. Every element *i* of the vector $\hat{x}$ is related to a particular stop, and gives the attributes of the stops in the same route than *i*, without considering the depot and the stop *i*. In the case of the instances that will be used in this work, the attributes of a stop will be the customer demand and the customer number. The demand is normalized to [0, 1] by dividing it with the capacity of the vehicle.

The attributes of the arcs are represented as a vector $\tilde{x}$, as in the previous case, every element is related to a stop. The element *i* of the vector $\tilde{x}$, contains the attributes of the arcs that are in the same route than *i*. Here the arc is characterized by the pair of nodes, origin – destination, and the arc length. The arc length is measured relative to the average arc length.

The last group of attributes, tour attributes, is represented by the vector $\bar{x}$. To every stop is associated the number of the tour where this belongs. In the cases analyzed here, no difference in the fleet is considered, since all the vehicles have the same capacity and no operational costs are involved. However the tour can be associated to a driver, type of technology that the vehicle is using or any other characteristic, even though this is not considered into the optimization problem, it might introduce some differences between the solutions.

Table 4.1 shows four different solutions (A, B, C and D) to the instance A-n32-k5. The Figures 4.1 to 4.4 represent these solutions in the plane. It is evident that the solutions A and B are similar, since just small variations between them are detected. According to the measure used here, the distance between these two solutions is just 0.184521. On the other hand the Solutions C and D look very different, which is confirmed by the distance of 0.758677 between them. Table 4.2 shows the distances between the four solutions in more detail. The numbers in parenthesis represent the weight associated with the total length in every solution.

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

Table 4.1 Different solutions to the instance A-n32-k5

	Solution 2			
Solution 1	A (0.7)	B (0.8)	C (0.2)	D (0.9)
A (0.7)	0	0.184521	0.535995	0.658771
B (0.8)	0.184521	0	0.418876	0.680456
C (0.2)	0.535995	0.418876	0	0.758677
D (0.9)	0.658771	0.680456	0.758677	0

Table 4.2 Distances between solutions to the instance A-n32-k5

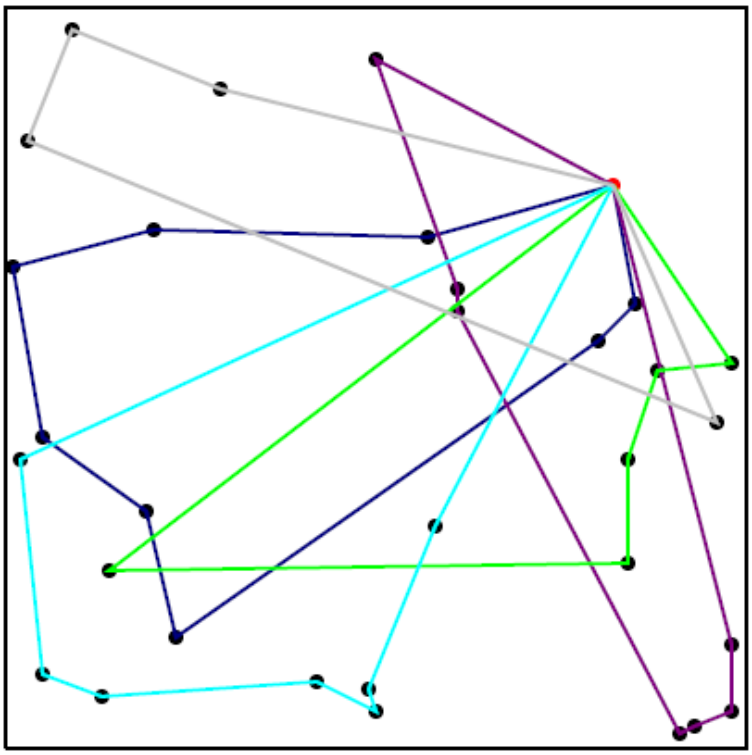

Figure 4.1 Solution A to instance A-n32-k5

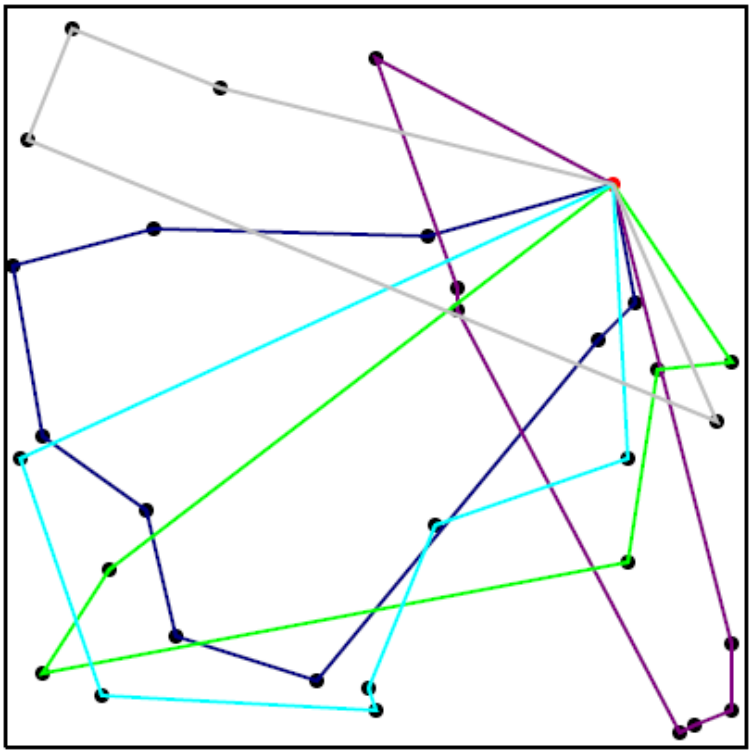

Figure 4.2 Solution B to instance A-n32-k5

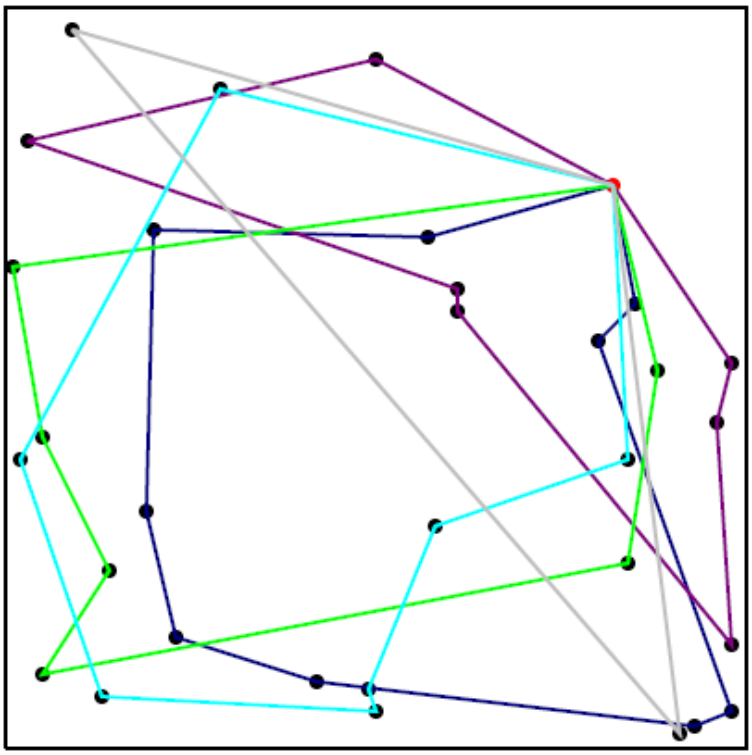

Figure 4.3 Solution C to instance A-n32-k5

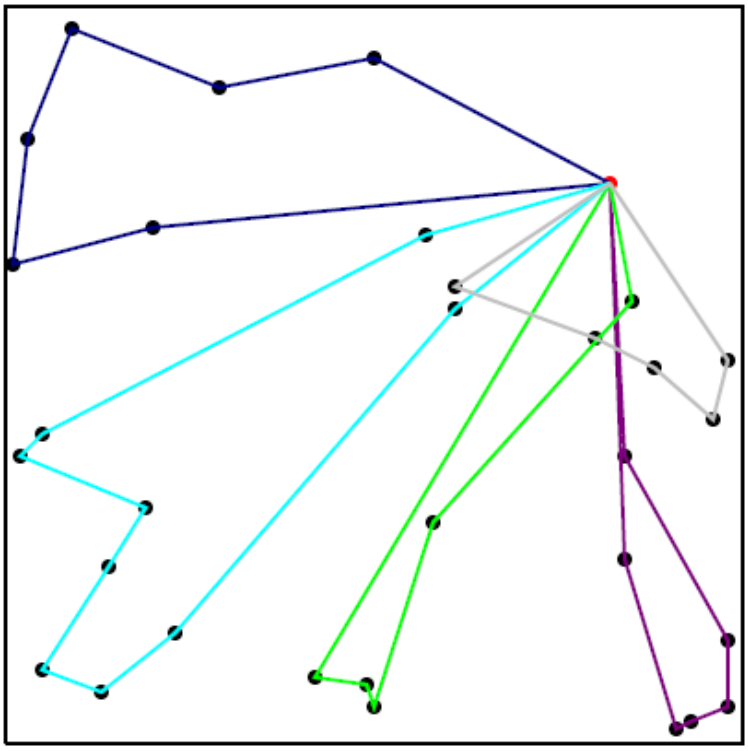

Figure 4.4 Solution D to instance A-n32-k5

5. A tabu search heuristic for the Multi-objective VRP

Given the multi-objective problem, $\min F(x)=(f_1(x), f_2(x),..., f_n(x)), s.t. x \in D$, using the "weighted sum of objective functions method" this is transformed into a single objective problem, $\min F_{eq}(x)=(w_1 f_1(x)+w_2 f_2(x)+...+w_n f_n(x)), s.t. x \in D$, where $\sum_{i=1}^{n} w_i = 1$.

For every combination of weights $(w_1, w_2,..., w_n)$ a different single objective function is obtained. If an exact method is applied, then the solution of this single objective problem is a Pareto optimal solution of the original problem (Collette and Siarry 2003, p 41). Here a heuristic will be used, so the solution that the heuristic will find will be potentially Pareto optimal, relative to the algorithm itself (Jozefowiez, Semet et al. 2007). However it is expected that for some set of weights, it will not be possible to find a non dominated solution. So not every solution will be a potentially Pareto optimal, relative to this heuristic.

Figure 5.1 shows a set of solutions to the instance A-n34-k5. The points in blue indicate the non dominated solutions, while these in green show the dominated solutions. To every point in green is possible to find another in blue that represents a better value for the total length and/or the route balance. These solutions were found assigning weights to the total length between 0 and 1, using a 0.1 step.

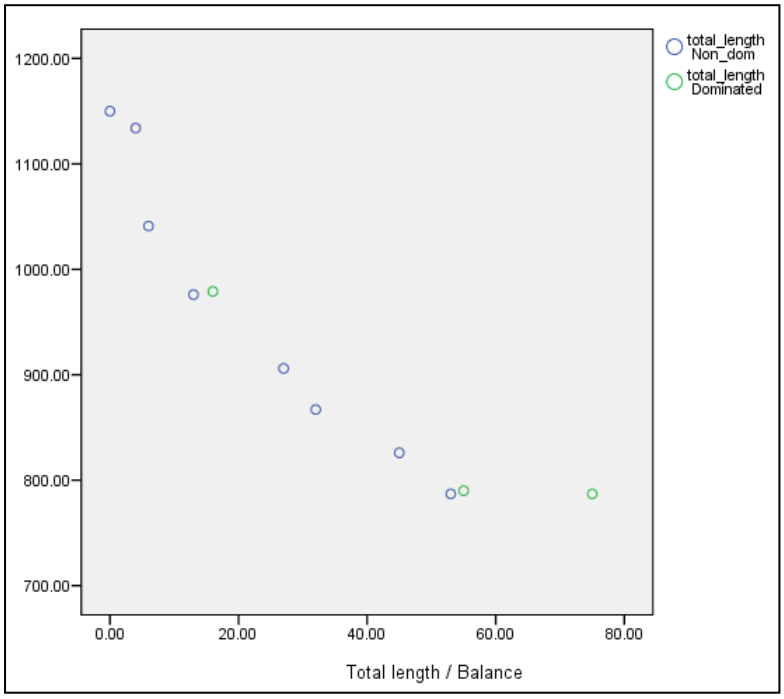

Table 5.1 Set of multi-objective solutions to the instance A-n34-k5

5.1 The basic algorithm

The implemented algorithm is based on the VRP solver developed by Professor Johan Oppen. The original algorithm is based on the "unified tabu search heuristics for vehicle routing problem with time windows" (Oppen 2004), which was proposed by proposed by Cordeau, Laporte and Mercier (Cordeau, Laporte et al. 2001).

5.1.1 Moves

The algorithm accepts infeasible moves, that is, moves that take the search to a solution with a capacity constraint violation in at least one tour. So let's define a set S which is not the set of feasible solutions, but the set of all solutions that satisfy: every tour starts and ends at the depot, and every customer belongs to exactly one tour. Every solution $s \in S$ is characterized by an attribute set, defined as $A(s)=\{(i,k):\textit{customer } i \textit{ is visited by vehicle } k\}$. The neighbourhood $N(s)$ of a solution s is defined by applying an operator that removes an attribute (i,k) from $A(s)$ and replaces it with a different attribute (i,k'), where $k \neq k'$. This gives a neighbourhood size of $|N|=n\times(m-1)$, where n is the number of customers and m corresponds to the number of tours (vehicles). When a customer is removed from a tour, the tour is reconnected by linking the predecessor and successor of the removed

customer. The insertion of a customer into a tour is done in a way that the tour length increasing is minimized, but without changing the order of the customers already in the tour. (Oppen 2004).

5.1.2 Move evaluation and diversification

Given a solution $s \in S$ and a move that takes the search from s to s', the evaluation of the move will depend on the two solutions s and s'. Let $c(s', s)$ be the difference of the total length of all tours in the solution s' and the same length in the solution s. In the same way, let $b(s', s)$ be the difference of the balance of the tour lengths in both solution. And finally let $q(s', s)$ be the difference of total violation of capacity constraints for the tours. The tour balance can be measured either as the difference between the longest and the shortest tour or the summation of the square differences between every tour length and the average length.

The single objective function used in the search can be associated to a weighted average, but in this case every element has different units. This can generate some distortions, since the difference between the values of the total length and the tours balance could be so significant that the weights $w's$ associated to each of them will not affect the search. In some cases the total length could be less than 1000, but the tour balance values can be more than 30000. This situation is more evident when the tour balance is measured as the summation of the square differences between every tour length and the average length. Because of this, the total length and the tour balance were normalized to avoid these significant differences.

A poor or a lack of normalization of the move evaluation function, can lead to tradeoff solutions as the ones represented in the Figures 5.2 and 5.3. In both cases it is easy to see that there is a bias to prefer the balance over the length.

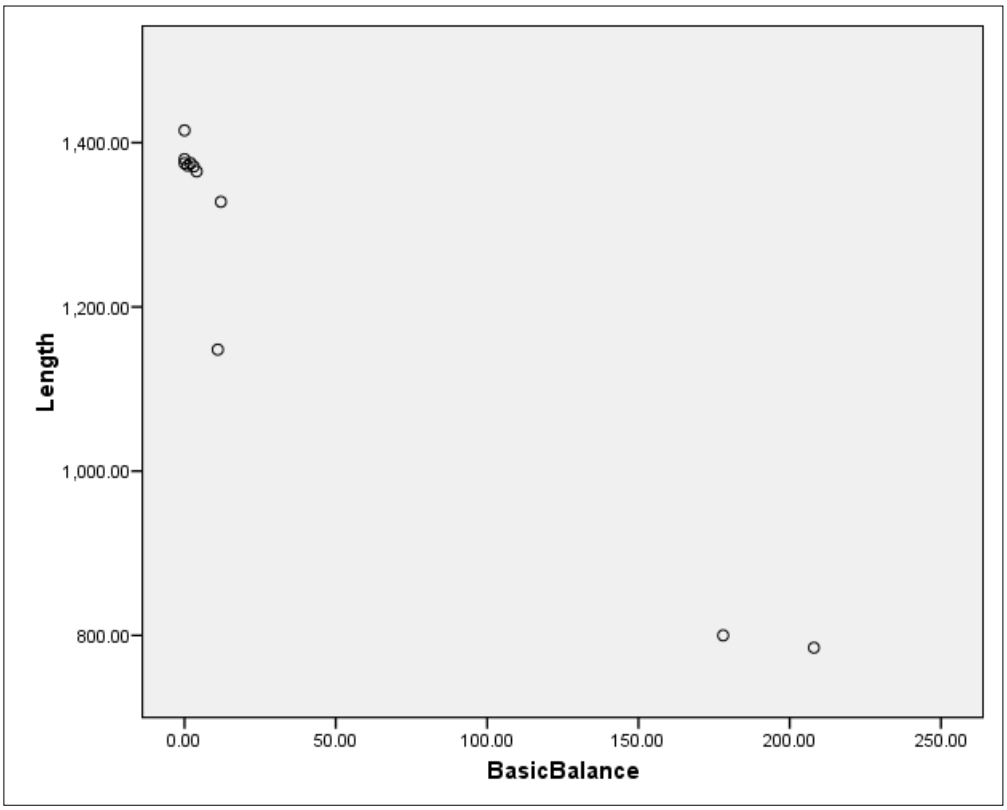

Figure 5.2 Multi-objective solutions to instance A-n32-k5

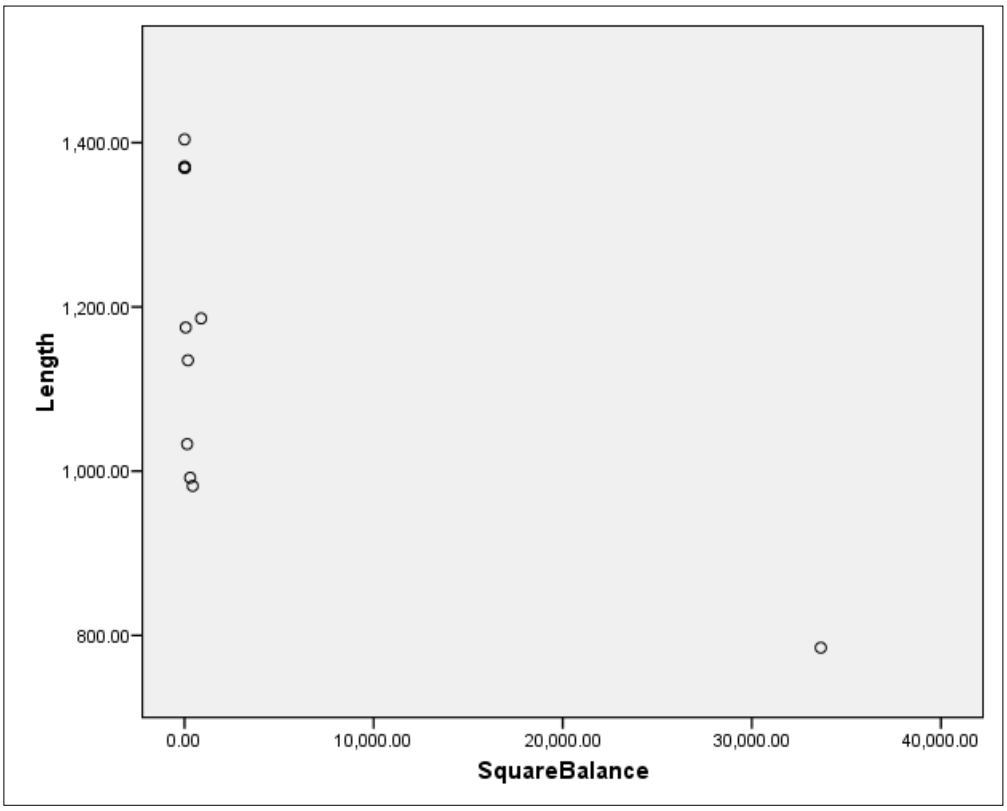

Figure 5.3 Multi-objective solutions to instance A-n32-k5

The normalization of the move evaluation function is done in two different ways, depending on the type of measure used for the tour balance. If this is measured as the difference between the longest and the shortest tour, then the total length is divided by the difference between the longest and the shortest arc in the instance, and the tour balance is divided by half of the average arc length. If on the other hand, the tour balance is measured by summation of the square differences between every tour length and the average length, the total length is then divided by average arc length, and the tour balance is divided by two times the square average arc length. In each case, the total violation of capacity constraints for the tours is divided by the vehicle capacity.

If the corresponding normalization factor associated to each objective value is defined as $s(.)$, then each move can be evaluated using the function $f(s',s)=\left(\frac{c(s',s)}{s(c)}+\alpha_1\frac{q(s',s)}{s(q)}\right)\times w_1+\left(\frac{b(s',s)}{s(b)}+\alpha_2\frac{q(s',s)}{s(q)}\right)\times w_2$, where α_1 and α_2 are positive parameters that are dynamically adjusted during the search.

For diversifying the search only the non-improving moves, $f(s',s)>0$, are considered. A penalty $p(s',s)=\lambda\sum_{(i,k)\in A(s')}\rho_{ik}$ is added to $f(s',s)$. Here, ρ_{ik} is the number of times the attribute (i,k) has been added to every solution. The parameter λ is used to control the intensity of the diversification. These penalties are used to lead the search into less explored parts of the solution space whenever a local optimum is found. If $f(s',s)\leq 0$, the move is improving and the penalty term is not added (Oppen 2004, p 28).

5.1.3 Initial solution

For every set of weights used in the solution method, a complete tabu search is done. In every search the first step is generating a starting solution. This can be done either by a greedy or a random approach. In both cases, all customers are first put into a list of unassigned customers, from which they are removed as they are inserted into tours. A number is assign to the customers which follow the order in which they appear in the instance file (Oppen 2004, pp 28-29).

If the greedy approach is used, the starting solution will be the same for each search. The customers are inserted into tours one at the time, starting with the customer with the highest customer number. Every customer is inserted into the tour with the lowest possible tour number that has enough capacity left. A customer will be inserted into the respective tour, in a way that the increase in distance is minimized. If no tour with enough free capacity is found, the customer is inserted into the last tour. This gives a starting solution where all the tours are feasible, except the last tour, which eventually could be infeasible (Oppen 2004).

If a random approach is chosen, the search related to each set of weights will have a different starting solution. In this case, the customers are randomly selected, one at the time. The selected customer is inserted into a randomly chosen tour. However the position in which it is inserted into the tour is selected using the same criteria than in the greedy approach, this is, minimizing the length increasing. This procedure is repeated until the list of unassigned customers is empty. The random approach will lead to a starting solution where several tours may be infeasible (Oppen 2004). In all the tests done in this research, the search starts from a greedy solution.

5.1.4 Tabu search

For every set of weights (w_1, w_2) a tabu search is performed. The number of sets is given by the user, who will specify a minimum weight, a maximum weight and a step. The first search is done using the minimum weight. The weight related to the next search is obtained by adding the step to the one used in the previous search. This procedure is done until the maximum weight is reached. The number of weights obtained will represent the number of searches that have to be performed.

Every weight will represent a set, since just two objective functions are being considered. Given a particular weight w, during the search this will be associated to the length, whereas the difference between 1 and w is assigned to the balance. This means that a particular set of weights can be expressed just in terms of one of its elements (w_1, w_2).

Every tabu search starts from an initial solution and moves, each iteration, to the non-tabu neighbour that minimizes the weighted objective value for all $s \in N(s)$. This is, the move

with the minimum evaluation is selected; removing the attribute (i,k) from $A(s)$. The attribute is then declared tabu, for a given number of iterations, equal to the tabu tenure. During these iterations, the customer *i* will not be moved back to tour *k*. However, an aspiration criterion allows the selection of a tabu move if it leads to a solution better than the best solution found so far in the search. (Oppen 2004)

After each move, the values of the parameters α_1 and α_2 are adjusted. If none of the tours in the current solution has capacity violations, the values of α_1 and α_2 are adjusted, decreasing them, in order to focus more on the objective function value. If one or more tours in the current solution violate the capacity constraints, α_1 and α_2 are increased in order to lead the search back to the feasible region of the solution space. (Oppen 2004)

If the current solution is feasible, has a total length less than 1.1 times the length of the best feasible solution found so far during the search, and the number of iterations performed has reached 100, the solution is considered to be potentially good. Whenever a potentially good solution is found, a 2-opt procedure is applied to each tour in the solution, trying to find improvements by evaluating all possibilities of replacing 2 arcs within the tour (Oppen 2004).

The search continues until the stopping criterion is reached, which could be either a specific number of iterations or an amount of time, established by the user. Every search will last for the number of iterations or the time established by the user. To solve a problem will require as many searches as the size of the set of weights that the user establishes. So the real number of iterations performed by the solver, when finding a potentially Pareto set of solutions, will depend on the cardinality of the set of weights and the number of iterations established by the user. The same consideration applies when the time is the stopping criteria.

5.2 Solver description

The Figure 5.4 shows a conceptual model of the VRP solver. The model has some original entities included by Professor Johan Oppen in his solver, most of which were modified during this research. A new group of entities were included here, to be able to compute the

distance measure between solutions. In the conceptual model, the new entities are in light blue, the rest of the model is based on the original one (Oppen 2004).

5.2.1 Original entities of the solver

VRPSolver

The solver has an instance object (VRPInstance), so it will know what it has to solve. In addition it has a Parameter object with the parameter values required for the search. It has two vectors of Solution objects, one of them to store the best solutions found during one search, the other is used to store the best solution found on every search, the multi-objective problem solutions. The solver has also a vector of tours and a vector of solutions images. It is responsible for doing the different searches and computing the distance between solutions.

VRPInstance

The Instance knows the available number of vehicles and their capacity. It has a vector of original customers and knows the distance between all of them, including the depot. The instance also has a parameter object.

Solution

A Solution consists of a vector of tours, and knows its total length, the amount of capacity excess and the tour length balance, measure in two different ways. In addition it knows in what iteration the solution was found, the weight and the weighted objective function value.

Tour

A Tour has of a list of customers that define both what customers to visit and in what order. It has pointers to both the solver and the instance. It knows the length, the capacity excess and the free capacity of the tour. It needs to deliver results to the solver and ask the instance for information.

Node

A Node represents a point in the plane, and originally it could be a road crossing or a customer, however here it is considered just as a customer. The node has a Location object.

Customer

A Customer is a particular type of Node, even thought in this version of the solver a Node will be always a Customer. It has the location, inherited from the Node, the demand and a customer number.

Location

A Location knows its coordinates in the plane.

5.2.2 New entities added to the solver

The entities included in this section are related to the representation of the solution by means of attributes, i.e. arcs, stops and tours. This representation will be used to compare the solution with others and being able to measure the distance between them.

SolutionImage

A Solution Image has an Instance and a Solution object, so it knows the solution that it is representing and also the problem to which the solution is associated to. It has a vector of customer images, and it is responsible for creating the images for all customers.

CustomerImage

A Customer Image has a Tour and an Instance object, so it will know the other customers that share the same tour with the one that it is representing. In addition, it also has the index of the customer in the Tour and the tour number. The customer image has a vector of Stops and a vector of Arcs. It is also responsible for creating the stops and arcs attributes.

Stop

A Stop knows the original customer number and the customer demand, not the real one, but standardized by the vehicle capacity.

Arc

The Arc knows the starting point and the ending point of the arc, and the arc length standardized by the average original customer distance.

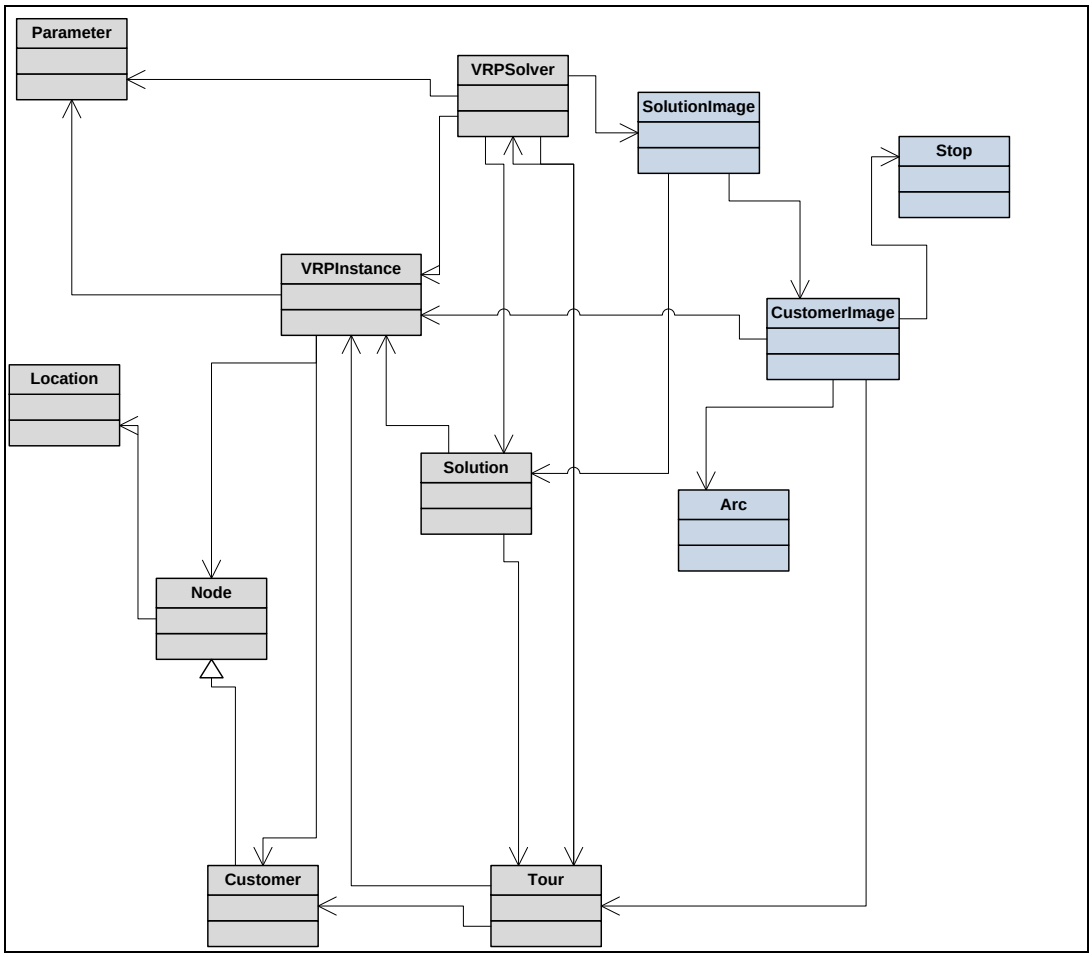

Figure 5.4 Conceptual model of the VRP solver

5.3 Preliminary testing for finding good parameter values

A subset of five instances has been used to do preliminary tests for finding good values for some search parameters. Not all the parameters considered into the original solver, neither all the options were included in this research. The instances selected for the preliminary testing were A-n32-k5, P-n55-k8, B-n67-k10, E-n101-k14 and M-n200-k17. More details about the test cases can be found in Appendix A. These instances were selected looking to have a broad number of customers and vehicles into the problems considered for the testing. The instance with the lower number of customers has 31 (plus the depot) and the one with the higher number has 199. The number of vehicles varies from 5 to 17.

Ten different random seeds will be used for the testing, and each test will last for 5 minutes. The run time limit is used, since most of the significant improvements in the weighted objective value, have been observed to be achieved during the first 5 minutes of the search. The weight used during the search was 0.5, this is just one multi-objective search was performed, giving the same relevance to every objective function.

5.3.1 Tabu tenure

The tabu tenure determines the number of iterations in which the tabu status of an attribute (i,k) will hold. This is, after a customer i has been removed from tour k, the customer cannot be returned back to the tour for a certain number if iterations equal to the tabu tenure. The tabu restriction is the most distinctive feature of a tabu search (Al-Sultan and Fedjki 1997), because of this choosing a good tabu tenure to perform a search is quite important.

Several strategies can be followed to set the tabu tenure. It can be fixed during the whole search, either the same for all the instances or individually for every instance. An additional approach is changing it at every move; in this case it will be randomly selected from an integer interval. When the tabu tenure is assigned individually to every instance, it is necessary to establish a connection between the instance and the tabu tenure that should be selected. Here, just one of these strategies was evaluated. This was setting the tabu tenure from an integer interval, different for every instance, but based on the number of customers.

To find out which interval could be good for every instance, several intervals were used to perform the tests. Then the relation between the number of customers in the instance and the interval that perform better was checked, i.e. those that lead to a lower weighted objective function. It was concluded that a good strategy should be [30, 35] if the customers are less than 50; [30, 40] if the number of customers is between 50 and 99; [60, 100] for instances where the number of customers is greater or equal to 100 and less than 200; finally [100 - 200] for the instances with more than 199 customers.

5.3.2 Weight of infeasibility

The parameters α_1 and α_2 give the weight of infeasibility for each objective function, initially both of them are set equal to 1. So two terms included in every parenthesis in the move evaluation function, $f(s',s)=\left(\frac{c(s',s)}{s(c)}+\alpha_1\frac{q(s',s)}{s(q)}\right)\times w_1+\left(\frac{b(s',s)}{s(b)}+\alpha_2\frac{q(s',s)}{s(q)}\right)\times w_2$ will count equally at the beginning of the search.

Several tests were run to find out a good way to adjust these parameters. During the tests, the parameters α_1 and α_2 were adjusted arithmetically, this is all the combinations among 0.01, 0.1, 0.5, 1 and 2 for add and subtract from the current value of the parameters. The best results for both cases were found when the factor 1 is added / subtracted from the current value of α_1 and α_2. This is if the current solution is feasible then both, α_1 and α_2 are decreased by 1. If on the other hand, it is infeasible, then α_1 and α_2 are increased by 1.

5.3.3 Diversification strategy

Two additional diversifications strategies were tested to see which could give better results. In both cases an additional penalty $p(s',s)=\lambda\sum_{(i,k)\in A(s')}\rho_{ik}$ was added to the non-improving moves. Where λ is a fixed parameter and ρ_{ik} is the number of times than the attribute (i,k) has been added to either every solution or to the *good* solutions. Depending on the criteria used for establishing ρ_{ik}, then the strategy will be different, it could be related to the good solutions or to every solution. The best results were found when λ was set equal to 100 and ρ_{ik} established as the number of times that the attribute (i,k) has been added to every solution.

5.3.4 Running time

The results from the solver are not going to be compared with results from other solvers. But as the solver is being considered as a part of a DSS, the solution time can be considered as a critical resource for the user. It has been observed that the number of iterations per second that the solver is able to perform varies significantly, depending on the number of customers in the instance, as can be seen in Figure 5.5. Then it would not be reasonable to assign the same running time to all the instances.

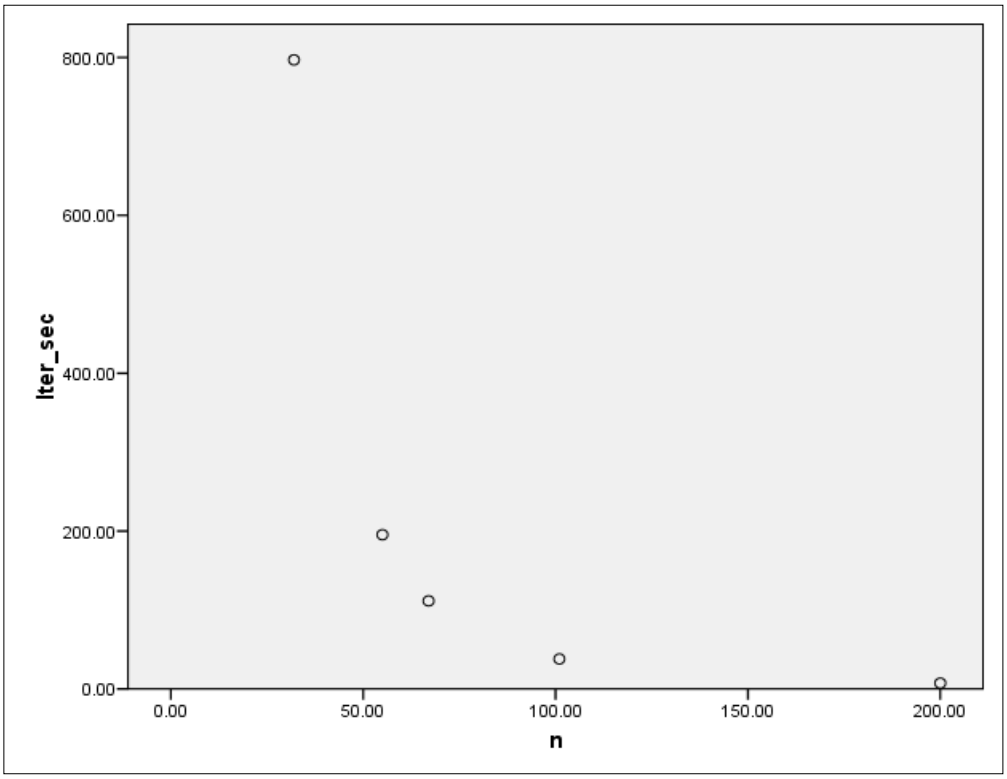

Figure 5.5 Iterations per second as function of the number of customers in the instance

The objective of this research is not to find the best solutions, or not necessarily. On the other hand as many searches, one per each potentially Pareto optimal solution, have to be performed, then establishing a long running time can lead to an extreme situation. This situation can occur in the case that several hours would be required for finding the set of solutions, which in a DSS framework could be considered prohibitive, since the decision maker wish to have the results as soon as possible.

Due to the previous considerations, the running time was established as a function of the numbers of customers in the instance. Where the assigned time should be close to the time expended in reaching the point where the improvements start to be smaller. To determine this time 10 different runs where performed using 3600 seconds for each instance. And then a linear regression was done, finding the relationship between the time expended and the size of the problem, defined as the number of customers.

The Figure 5.6 shows the relation between the best weighted objective value and the search time for the instance M-n200-k17. Where after 450 seconds there was just one improvement better than 2%, for all random seeds.

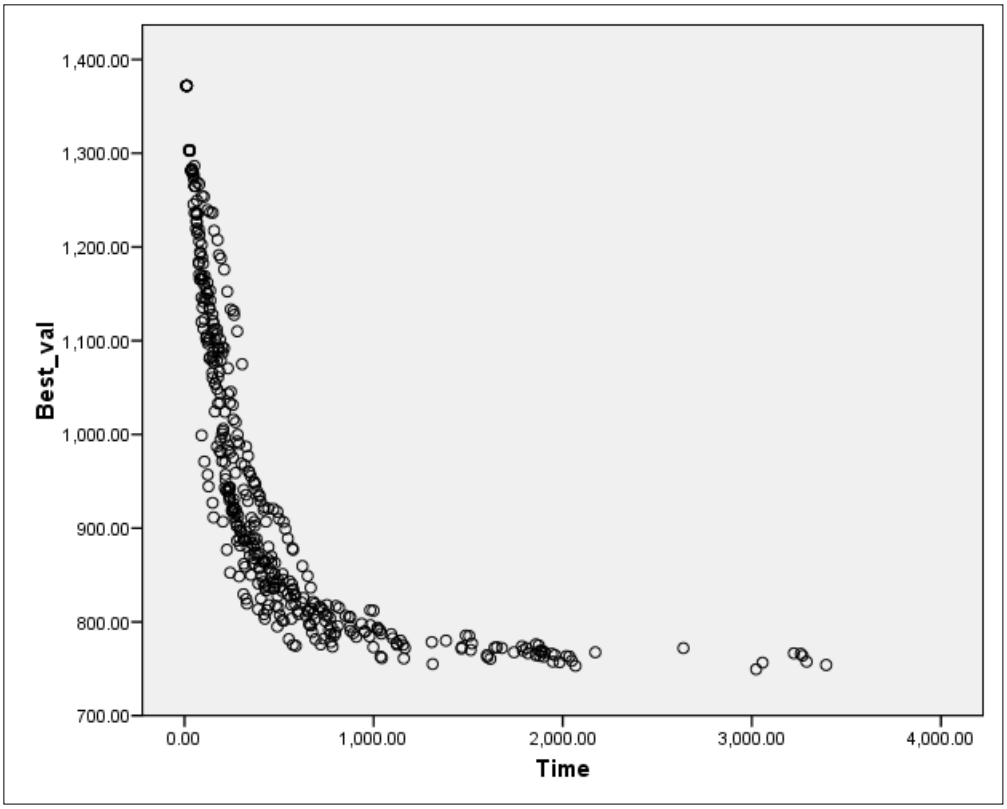

Figure 5.6 Best weighted objective value, as function of the time, for instance M-n200-k17

It was found that a model represented as $T = 1.096 \times N \log N$ would be a fair explanation to the time for the different instances. The R-Square is equal to 0.8974, which was considered good enough. To simplify the problem of assigning the running time to each instance, eight different running times were established. The instances were grouped into eight sets, depending on the number of customers. To every set a running time was assigned. Table 5.1 shows the maximum number of customers of the instances in each set.

Instance set	Number of customers	Running time
1	48	60
2	72	120
3	78	180
4	100	240
5	135	300
6	151	360
7	200	480
8	262	720

Table 6.1 Running time per instance set

5.3.5 Balance measure

It is assumed here that the decision maker would wish to have not necessary equal tour lengths, but none of them longer than the other by certain threshold. Somehow this can be computed dividing the difference between shortest tour and the longest tour by the shortest one. This difference between the longest and the shortest tour, has been one of the ways to measure the balance, it has been called here, basic balance. The other possibility for measuring the balance during the search, has been the summation of the square differences between every tour length and the average tour length, this balance has been called square balance.

The basic balance gives an easier idea to understand of what it is measuring, and it is more related to the measure of balancing that is assumed to be used by the decision maker. However both balance measures could be used during the search, for evaluating the moves, reporting to the decision maker the basic balance, which is easier to understand for him/her. Several tests were run to find which balance measure report better results, defined as the lowest difference between the shortest and the longest tour. For each balance measure 10 different random seeds were used.

For the instances with 32 and 55 customers (including the depot) there was no statistical difference between the longest and the shortest tour in the final solutions, no matter which balance measure was used during the search. However, for the rest of the instances, with 67, 101 and 200 customers, the difference between the longest and the shortest tour was much higher when the balance measure used during the search was the square balance.

Based on these results, it was decided to use the basic balance as the balance measure in the tests.

6. Experiments and results

6.1 Test cases

Eight different sets of test cases were used in this research. These are instances from the literature for the capacitated VRP (CVRP) and they can be found at http://branchandcut.org/VRP/data. However just 92 of the 109 instances reported in the previous source have been used.

The set A consists of 27 instances with 32 to 80 customers and 5 to 10 vehicles. The set B consists of 23 instances with 31 to 78 customers and 5 to 10 vehicles. The set P consists of 24 instances with 16 to 101 customers and 2 to 15 vehicles. These three sets are labelled as Augurat, et al. Just one of the instances in these sets was not used for the testing, P-n55-k15, for which the solver was not able to find a feasible solution after 10000 seconds, so it was removed from the test cases. The tightness (demand/capacity) of this instance is 0.99 which could explain the difficulty in finding the solution, however feasible solutions were found on instances with similar tightness.

The set CE, labelled as Christofides and Eilon, consists of 13 instances with 13 to 101 customers and 3 to 14 vehicles. Two of them have a non Euclidean distance type, so just 11 instances were used in the tests.

The set F, labelled as Fisher, consists of 3 instances with 45 to 135 customers and 4 to 7 vehicles. Two of them have non integer location coordinates, so just one instance was used in the tests.

The set GJ, labelled as Gillet and Johnson, consists of 1 instance with 262 customers and 25 vehicles.

The set CMT, labelled as Christofides, Mingozzi, and Toth, consists of 5 instances with 101 to 200 customers and 7 to 17 vehicles. All of them were used in the tests.

The set V, labelled as Converted TSPLIB Problems, consists of 13 instances with 16 to 48 customers and 3 to 5 vehicles. Just one of these instances has a Euclidean distance type, so it is the one used in the tests.

More details about the selected instances can be found in the Appendix A.

6.2 General tests

Using the 92 chosen instances, a general basic test was done. Eleven multi-objective solutions to every instance were generated, assigning weights to the total length in [0, 1], using 0.1 as step. In addition the distance between every pair of solutions was computed.

6.2.1 Cost analysis

Considering a second objective function into the optimization problem could also be seen as an additional constraint. So it is expected that the total length will increase when a higher weight is assigned to the balance objective. The solver does not guarantee optimality, so even though the shortest tour length is expected to be found when the weight assigned to the length objective function; this will not be always the result. Actually, in 57.6% of the instances, the best total length was found when the weight associated to the length objective function was different than 1. In 45.6% of the instances the best total length was found when the mentioned weight was equal to 0.9. This could mean that including a different objective function into the model will diversify the search, forcing the algorithm to explore different areas of the search space. However, it could also mean that the good search parameters, when the two objectives are considered, do not perform so well when only the total length objective is considered.

The cost of including the balance as an objective function into the optimization problem, in average, was quite significant. If the best total length is compared with the total length found when the weight assigned to it is equal to 0, the total length has an increment of 72.2%. If the comparison is done between the total lengths when the weights are 0 and 1, then the increment in the total length (cost) is equal to 70.3%. The Figure 6.1 shows the average increment in the total length, compared with the best value found during the multi-objective search.

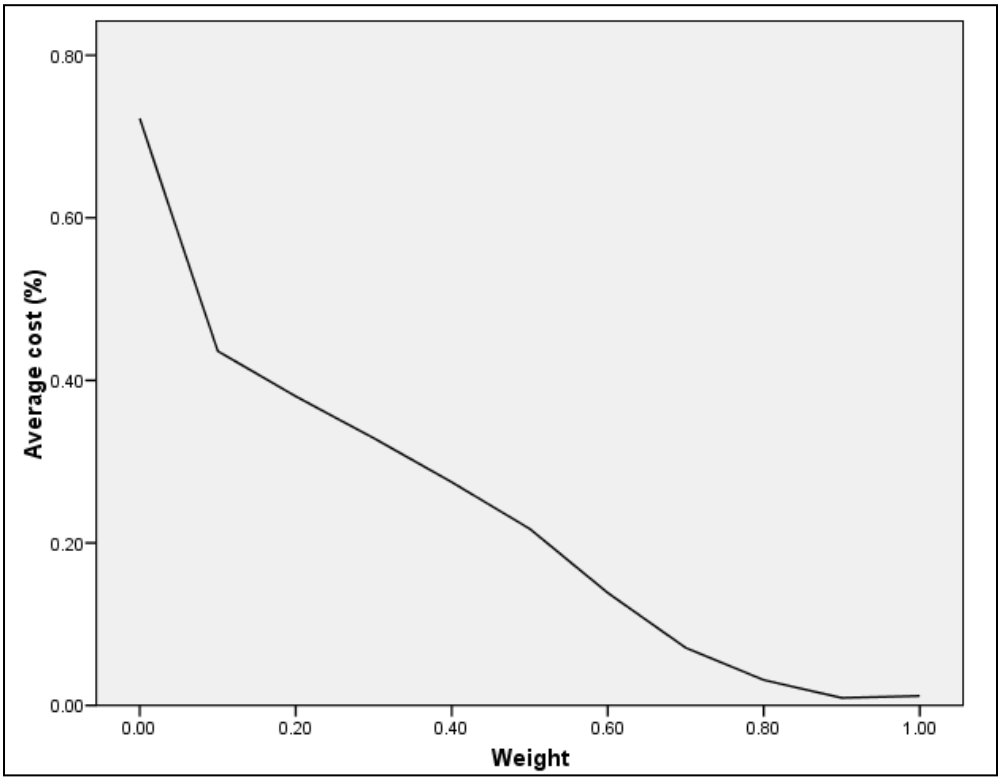

Figure 6.1 Cost of including the route balance in the optimization problem

6.2.2 Solution distances

The distance between solutions, as defined in Section 4.1, was calculated for every pair of solutions to the instances. A linear regression was done to find the relation among the average distance of the solutions to an instance and the number of customers and / or vehicles in the instance. No statistical evidence was found about the relation between the average distance of the solutions and the number of customer in the instance. In addition, even though it was found statistical evidence of the relation between the average distance and the number of vehicles, this last variable does not explain the linear model, since the R-Square of the regression is 0.126, which can be considered too low. Different variables should be considered to explain the distance between the solutions.

Figure 6.2 shows the frequency of the average distance, between all the pairs of solutions in the instances. The frequency shows a high average distance between solutions.

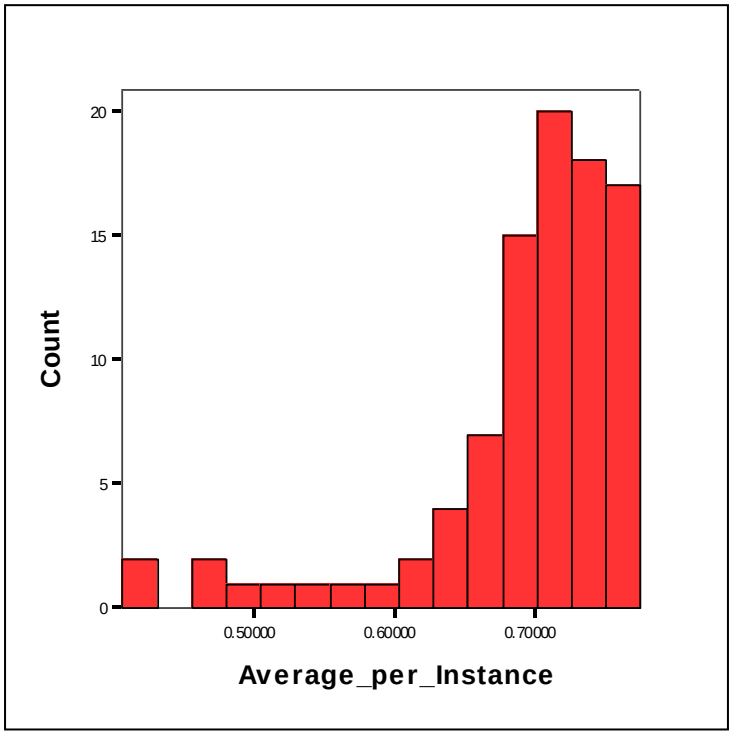

Figure 6.2 Frequency of the average distance between solutions, per instance

An additional analysis to the results was done. The average distance between any pair of solutions for all the instances was computed. The minimum and maximum distance was also found, for every pair of solutions. These results can be found in more detail in Appendix B. Figure 6.3 shows the frequency for the average distance. Figure 6.4 shows the behaviour of the maximum, the minimum and the average distance per pair of solutions. There are 55 pairs of solutions (0,1), (0,2)… (9,10).

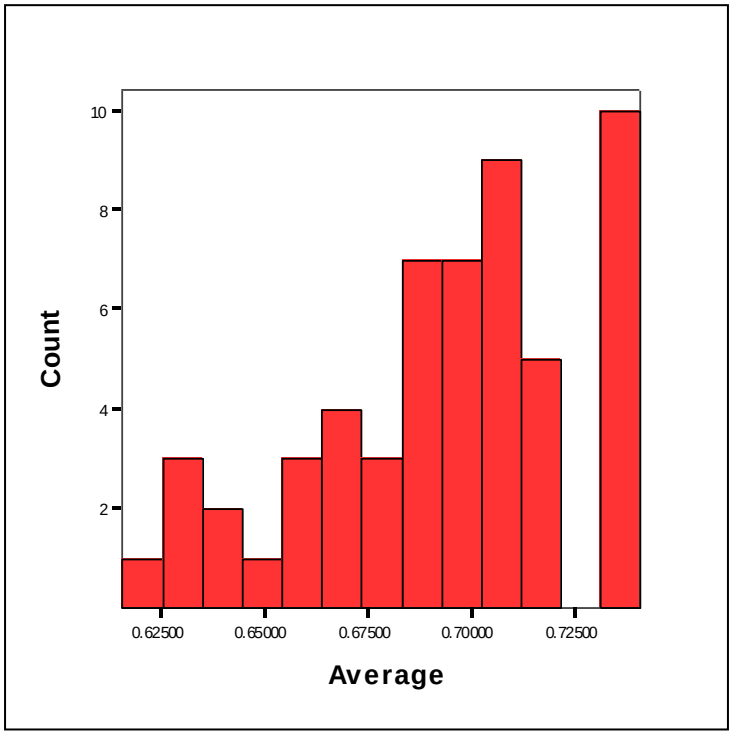

Figure 6.3 Frequency of the average distance between every pair of solutions

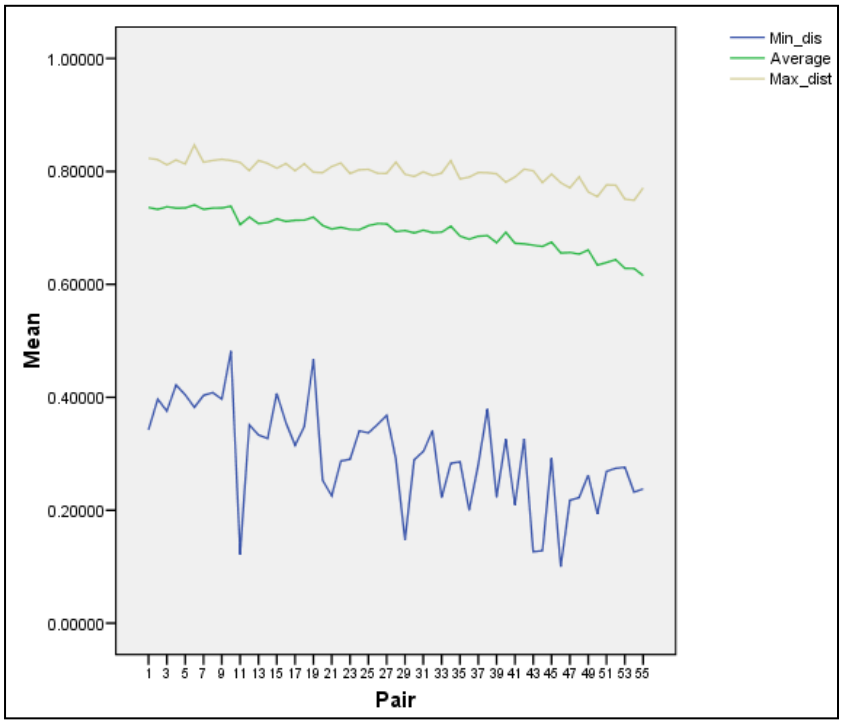

Figure 6.4 Minimum, average and maximum distance per pair of solutions

It was observed that varying the difference in the weights associated with the total length objective function will not necessary change the distance between the found solutions, or at least not as expected. For example, it was found that the distance between two solutions

that were found assigning weights of 0 and 0.1 to the total length objective, could be greater than the distance between the first solution and the other found when just the total length was considered.

Figure 6.5 represents a solution to the instance A-n32-k5, which was found after assigning a weight of zero to the total length objective, this solution will be referred as solution E. Figure 6.6 represents a solution to the same instance, solution F, but in this case the weight associated to the total length objective was 0.1. In addition, the same weight associated to the solution represented in Figure 6.7 is 1, solution G. It is evident that the distance between the weights associated to the solutions E and F is less than the distance between the weights associated to solutions E and G. The result is different when the solutions are compared in the solution space; this is using the distance measure between solutions. The distance between the solutions E and F is 0.755094, which is greater than the distance between solutions E and G, 0.69872. If the difference between the weights associated with the total length is used as a distance measure, the result would not correspond to the previous one. Table 6.2 shows all the distances between the three solutions, the weight difference is in parenthesis.

	Solution 2		
Solution 1	E	F	G
E	0	0.755094 (0.1)	0.69872 (1.0)
F	0.755094 (0.1)	0	0.642543 (0.9)
G	0.69872 (1.0)	0.642543 (0.9)	0

Table 6.2 Distances between solutions to the instance A-n32-k5

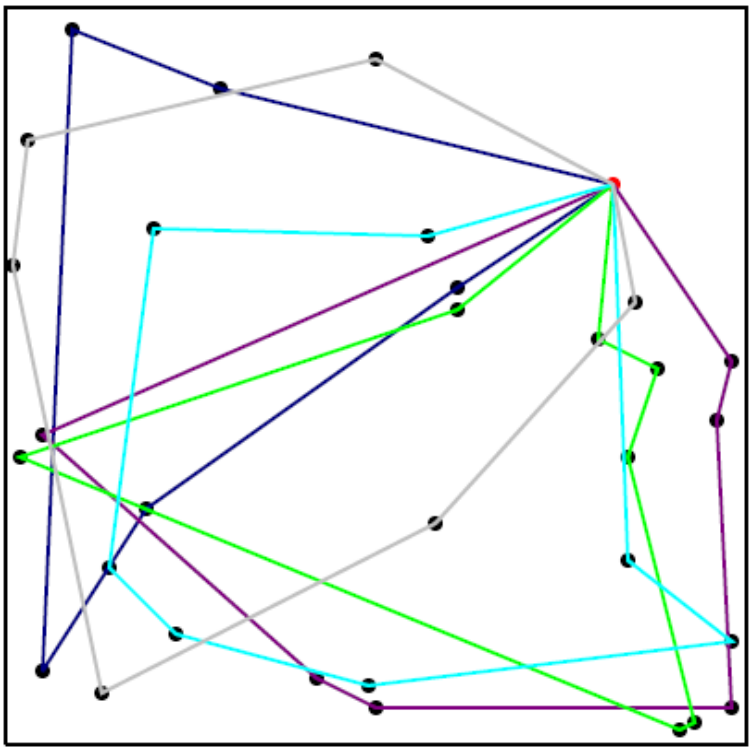

Figure 6.5 Solution E to the instance A-n32-k5

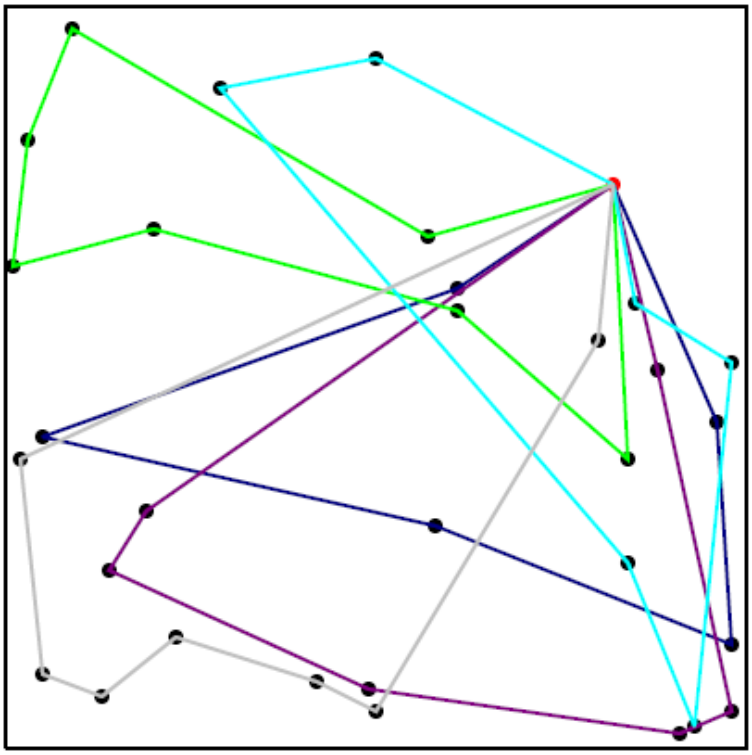

Figure 6.6 Solution F to the instance A-n32-k5

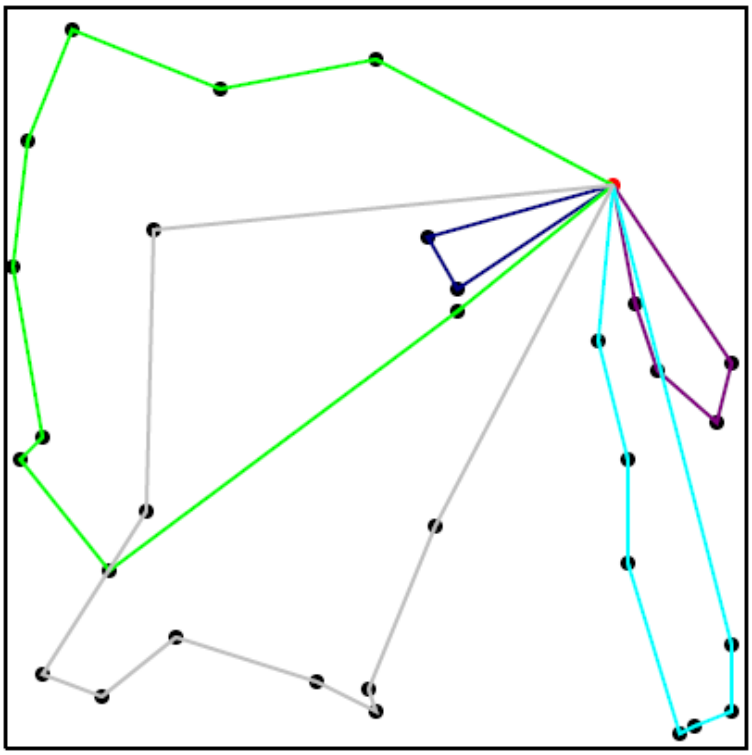

Figure 6.7 Solution G to the instance A-n32-k5

6.3 Specific tests

For running more specific tests, the time factor has to be considered. Since there is a limited time for carrying on this research, only 12 instances were selected for more testing. The instances A-n36-k5, A-n37-k6, A-n45-k7, A-n60-k9, A-n80-k10, B-n41-k6, B-n57-k9, P-n70-k10, P-n101-k4, E-n76-k10, G-n262-k25 and M-n200-k17 were selected, so different problem sizes, in number of customers and number of vehicles, will be considered.

Two different groups of tests were performed. The objective of the first group of tests is to generate several solutions where none of the route lengths is above the others by certain threshold. A 0.1 (10%) threshold was used here. In the second group of tests the objective is to reproduce the general tests, using the instances mentioned above. Ten different random seeds will be used, in any case, so it will be possible to get some statistical conclusions.

6.3.1 Solutions within a threshold

The tests have two different stages. The first stage is to find eleven potentially Pareto optimal solutions, varying the weight associated to the total length from 0 to 1, using 0.1 as step. For every solution the balance is checked, measured as the percentage in which the longest tour overpasses the shortest one. From those solutions which balance is within the threshold given by the decision maker, the highest weight is selected and increased by 0.1. The second stage of the search is performed then, varying the weight associated with the total length from 0 to the value found in the previous stage, using now 0.01 as step. Part of the solutions found during this stage are then classified as either good solutions or non-dominated solutions. The good solutions are those which balance is within the given threshold. The non-dominated solutions are those that, in addition to the previous condition, are not dominated by any of the other good solutions.

A set of solutions is then given to the decision maker from each group, good and non-dominated solutions. The solution with the shortest total length, the solution more distant to that one and the two more different solutions are part of the output. The objective of doing this is to evaluate if the dominated solutions, included in the good solutions set, could be relevant for the decision maker.

Figure 6.8 shows solutions found on every stage of the search for the instance A-n36-k5. A set of eleven solutions are represented in blue, these correspond to solutions found on the first stage of the search. The solutions represented in green correspond to solutions from the second stage of the search. 51 solutions were found during the last stage. 39 of them were good solutions, i.e. none of the tours was longer than any other in more than 10%. 7 solutions were non-dominated.

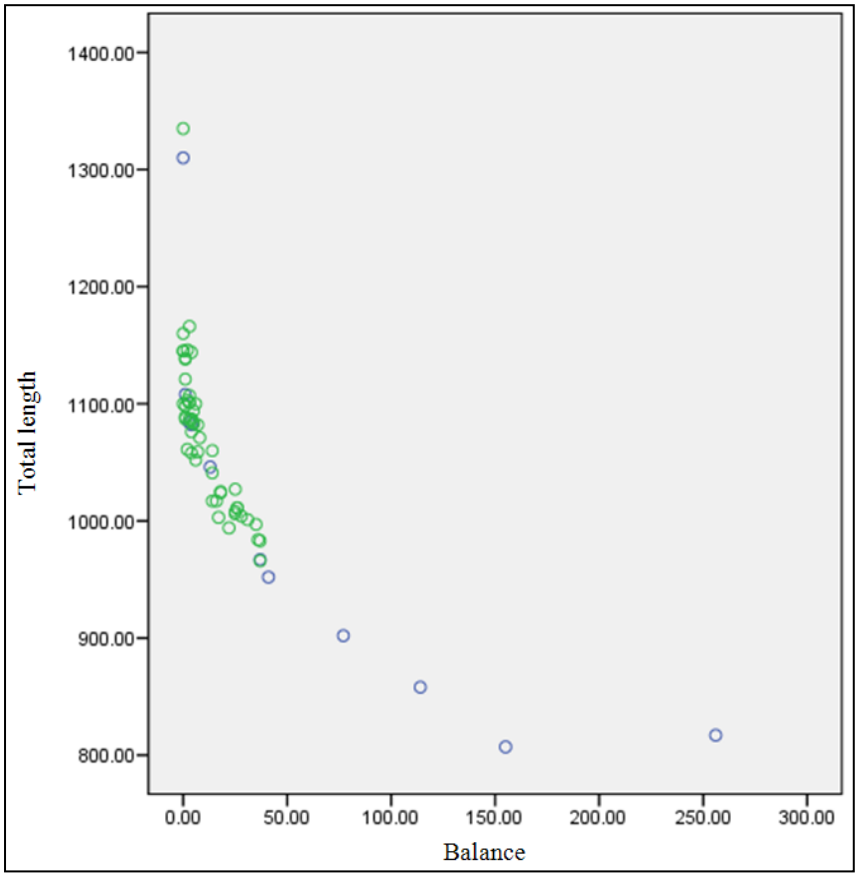

Figure 6.8 Multi-objective solutions for the instance A-n36-k5

Here it is assumed that all the solutions with a balance within the threshold are equally good. So a decision maker can be interested in having different solutions, since the model and the instances include assumptions about the real world, and somehow are a simplification of it. A good solution for the model will not necessarily fit well in the real world, either because it will not be easy to implement or because the quality once is implemented is not the expected. This can create an interest in the decision maker for seeing a set of different solutions, from where to choose the one to implement.

Test results

Two statistics test are performed for every instance. One test will evaluate if there is a difference between the average distance of the two more different solutions on each set, good solutions and non-dominated solutions. The second test will compare the average distance between the solution with the shortest length and the solution more distant to it, in the two sets. The number of solutions given as an output will not necessary always be the

same, since for some cases the solution with the shortest total length is also one of the most different solutions in the set.

Table 6.3 shows the interval confidence for the difference between the average distance obtained from the good solutions set and the one from the non-dominated solution set. As can be seen, there is no evidence to say that the average distances between the solutions in the good solutions set are not greater than those in the non-dominated solutions set.

Instance	95% Confidence Interval of the Difference (2 most different)		95% Confidence Interval of the Difference (including best length)	
	Lower	Upper	Lower	Upper
A-n36-k5	0.02786	0.037277	0.020446	0.042334
A-n37-k6	0.013265	0.026446	0.006951	0.030091
A-n45-k7	0.011534	0.033439	0.003258	0.027043
A-n60-k9	0.010906	0.01768	0.003764	0.020938
A-n80-k10	0.009656	0.01831	0.006818	0.018187
B-n41-k6	0.014714	0.028751	0.01155	0.026128
B-n57-k9	0.017438	0.027275	0.005286	0.020849
P-n70-k10	0.00997	0.034916	0.007876	0.025505
P-n101-k4	0.023153	0.048658	0.018479	0.048621
E-n76-k10	0.008633	0.040852	0.003707	0.027152
G-n262-k25	0.019587	0.031169	0.020724	0.032205
M-n200-k17	0.034734	0.048937	0.028072	0.042864

Table 6.3 Interval confidence for difference between the average distances of the solutions

Here just a threshold for the route balancing is given. The solutions in both sets, good solutions and non-dominated solutions, are considered to be equal, since they are within the given threshold. But the length could be a different issue. For the case of the best length in the good solutions set, it is not worse than the best length in the non-dominated solutions set. What can generate then, a difference for the decision maker, is the length in the solution more different to the previous one, and the two most different solutions. This will be a tradeoff between diversity of the solutions and total length; however this is part of a different analysis. If just the distance between the solutions is considered, then it will be possible to say that including dominated solutions in the set to give to the decision maker, will increase the diversity of the solutions, which according to previous considerations is something that could be interesting for the decision maker.

6.3.2 Solutions to the multi-objective problem

Eleven solutions to the multi-objective problem were generated for each instance, using weights on [0, 1] and 0.1 step. The distance between all the solutions was calculated. This procedure was done using ten different random seeds. The cost of including the route balancing as an objective function into the problem is evaluated. Another aspect that is analyzed from the results is the correlation of the distance between solutions in the solution space (attribute based distance measure) and weight space (difference of weights associated to the total length).

Test results

The average total length was calculated for every weight associated with it. In 9 of the instances, the average best total length was found when the weight associated with the length objective function was 0.9. In the instance G-n262-k25 was found in the search performed with a 0.8 weight. The best total length found in the instances P-n101-k4 and E-n76-k10 was associated with a weight equal to 1. In general, the total length increases when the weight associated with it is decreased. These results are compatible with those found in Section 6.2.1, and shows that considering another objective function can diversify the search. Table 6.4 shows how the average cost as a percentage of the minimum found value is related to the weight, for all the instances.

Instance	weight										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
A-n36-k5	0.636	0.347	0.345	0.290	0.256	0.197	0.158	0.112	0.075	0.000	0.003
A-n37-k6	0.513	0.377	0.324	0.288	0.244	0.204	0.161	0.068	0.014	0.000	0.009
A-n45-k7	0.501	0.321	0.283	0.245	0.223	0.198	0.152	0.107	0.033	0.000	0.003
A-n60-k9	0.689	0.506	0.450	0.393	0.356	0.206	0.090	0.039	0.011	0.000	0.010
A-n80-k10	0.720	0.470	0.424	0.349	0.307	0.224	0.121	0.081	0.023	0.000	0.005
B-n41-k6	0.715	0.509	0.358	0.342	0.305	0.236	0.116	0.058	0.025	0.000	0.005
B-n57-k9	0.474	0.279	0.248	0.171	0.182	0.127	0.102	0.086	0.007	0.000	0.011
P-n70-k10	0.680	0.468	0.415	0.382	0.346	0.285	0.187	0.077	0.030	0.000	0.013
P-n101-k4	0.688	0.394	0.379	0.350	0.327	0.279	0.228	0.151	0.056	0.001	0.000
E-n76-k10	0.795	0.574	0.488	0.459	0.383	0.314	0.145	0.061	0.021	0.006	0.000
G-n262-k25	1.385	0.735	0.735	0.701	0.685	0.565	0.275	0.024	0.000	0.027	0.031
M-n200-k17	1.558	0.777	0.684	0.670	0.598	0.552	0.238	0.038	0.009	0.000	0.030

Table 6.4 Average cost as a percentage of the minimum found value per instance

The highest average cost goes from 50.1% in the instance A-n45-k7 to 155.8% in the instance M-n200-k17. In every case this value is obtained when the weight associated with the total length is 0. It has a significant variation from instance to instance, as a percentage. Figure 6.9 shows the relation between the maximum average cost and the number of customers. Two cases look to be significantly different than the rest. Similar results if the analysis is done with the number of vehicles instead of the customers.

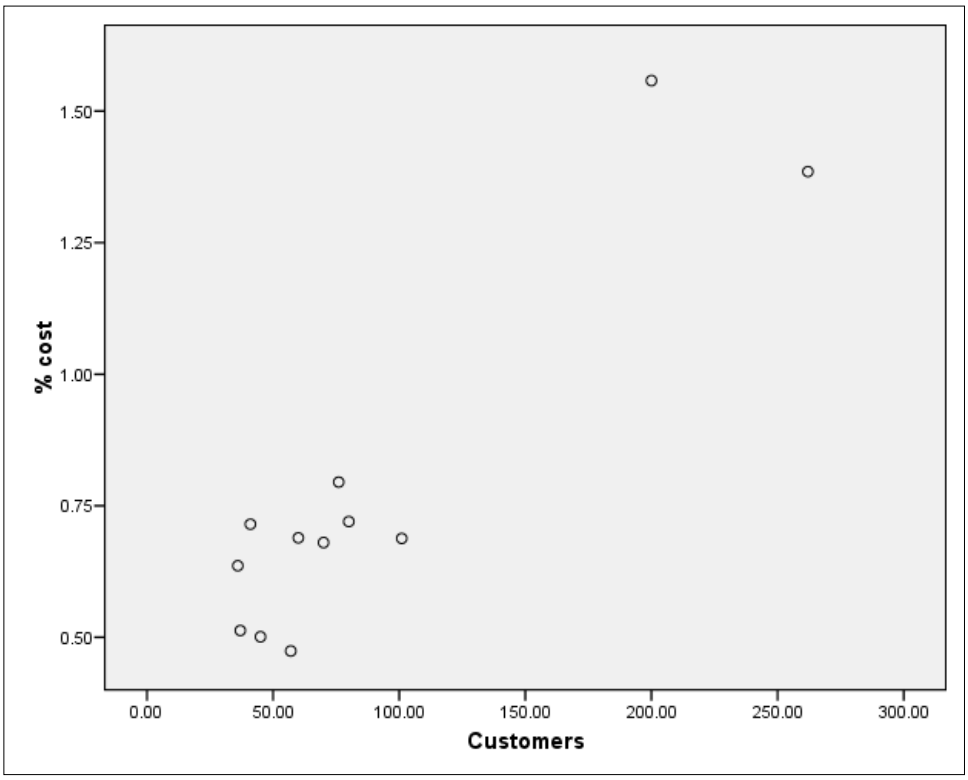

Figure 6.9 Relation between the maximum average cost and number of customers

Table 6.5 shows the correlation coefficients of the distance between solutions in the solution space (attribute based distance measure) and weight space (difference of weights associated to the total length). Two types of correlations were calculated, linear (Pearson) and rank correlation (Spearsman). In each case the correlation was smaller than 0.5, which can be understood as small or medium.

Instance	Pearson	Spearsman
A-n36-k5	0.36515	0.36506
A-n37-k6	0.28984	0.25613
A-n45-k7	0.30539	0.29047
A-n60-k9	0.41144	0.40057
A-n80-k10	0.46281	0.42414
B-n41-k6	0.32033	0.30559
B-n57-k9	0.42060	0.39361
P-n70-k10	0.22111	0.19354
P-n101-k4	0.18378	0.16201
E-n76-k10	0.36878	0.33690
G-n262-k25	0.43685	0.37043
M-n200-k17	0.49172	0.44411

Table 6.5 Correlation coefficients between distances in solution space and weight space

Figure 6.10 shows the relation between the distances in the solution space and the weight space for the instance E-n76-k10. There is some pattern that was common to all the tested instances. The values for the attribute based distance measure are more spread when the difference between the weights is smaller.

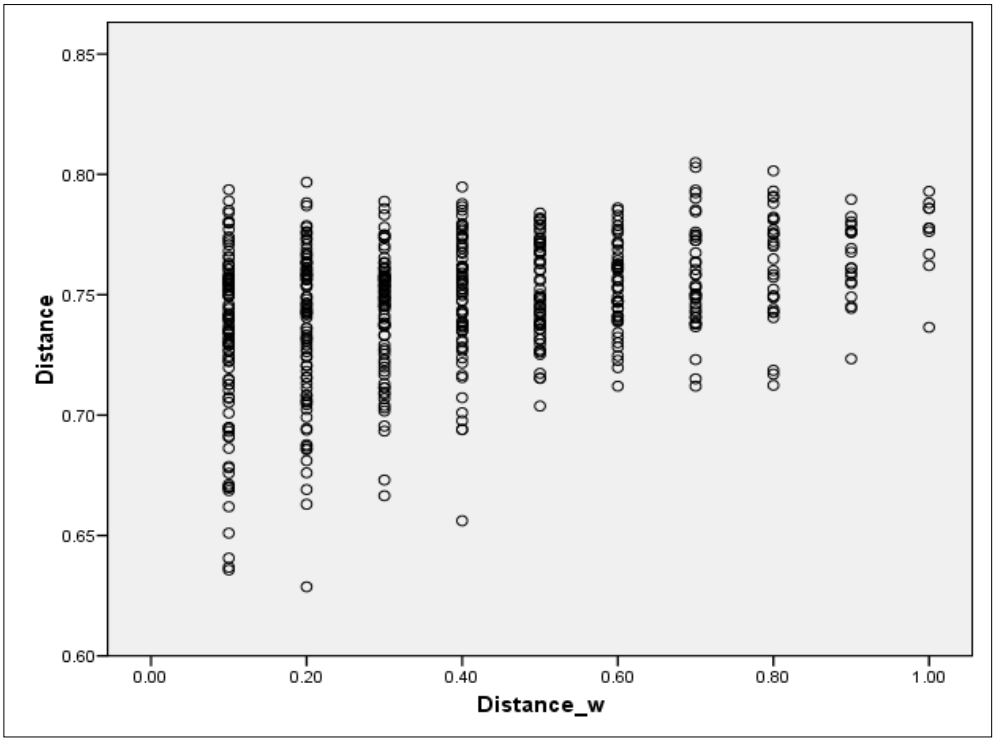

Figure 6.10 Relation between distances of the solutions to instance E-n76-k10

7. Conclusions and further research

Given a problem, several multi-objective solutions were able to be generated. Analyzing the obtained results, it was possible to confirm that including an extra objective to a single objective VRP, will have a similar effect than adding an extra constraint to the optimization problem, this is, it involves a cost.

The distance between multi-objective solutions was computed in the solution space, using the structural characteristics of the solutions. The distance between the solutions calculated like this seems to be, a more effective way to measure how different two solutions are, than using the weights associated to the objective functions. It was found that greater differences in the weights associated to the total length objective function, will not necessary generate more different solutions. In fact, the correlation between the difference of the weights associated with the total length and the attribute based distance measure was no different than small or medium.

It was not found statistical evidence of a strong linear relation between the average distance of the solutions to an instance and the number of vehicles and/or number of customers. Another type of relation or other aspects should be considered for further research.

Including a dominated solution into the set of solutions given to the decision maker would increase the variability of the set. It can have some benefits for the decision maker, since the problem solved by the model includes assumptions and simplifications of the real problem. The solutions will not necessary perform as well as expected, so several alternatives should be considered.

The multi-objective solver can be improved, since not all the strategies for setting the tabu tenure were evaluated, nor the diversification strategies. In addition the distance measure between solutions can be improved, either including more details into the measure or defining in a different way the distance between some attributes.

The tests were also performed using a single threshold for the route balance. It could be interesting to check the results obtained for different thresholds, as well as the importance

given to the arcs, stops and routes in the distance measure, which was set to be the same for all.

Including the balance into the optimization problem has a cost. The average highest cost looks to be significantly higher when the number of customers or vehicles is increased.

This research can be extended to include recovering plans with minimal disruption. When some characteristics of the instance are changed, the best found solution to the instance can have significant changes also. A decision maker can be interested in having a solution to the new problem as close as possible to the best found solution to the initial problem. This can be achieved by focusing the search in minimizing the distance between the solution to the initial problem and a feasible solution to the new problem. This problem could also be considered into a multi-objective approach, with the total length and the distance to the initial solution as part of the objective function.

References

Al-Sultan, K. S. and C. A. Fedjki (1997). "A tabu search-based algorithm for the fuzzy clustering problem." Pattern Recognition **30**(12): 2023-2030.

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

Cordeau, J. F., M. Gendreau, et al. (2002). "A guide to vehicle routing heuristics." The Journal of the Operational Research Society **53**(5): 512.

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

Doerner, K. F., W. J. Gutjahr, et al. (2006). "Pareto ant colony optimization with ILP preprocessing in multiobjective project portfolio selection." European Journal of Operational Research **171**(3): 830-841.

Eksioglu, B., A. V. Vural, et al. (2009). "The vehicle routing problem: A taxonomic review." Computers & Industrial Engineering **57**(4): 1472-1483.

Jaszkiewicz, A. (2002). "Genetic local search for multi-objective combinatorial optimization." European Journal of Operational Research **137**(1): 50-71.

Jozefowiez, N., F. Semet, et al. (2007). "Target aiming Pareto search and its application to the vehicle routing problem with route balancing." Journal of Heuristics **13**(5): 455.

Jozefowiez, N., F. Semet, et al. (2008). "Multi-objective vehicle routing problems." European Journal of Operational Research **189**(2): 293-309.

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

Løkketangen, A. and D. L. Woodruff (2005). "A distance function to support optimized selection decisions." Decision Support Systems **39**(3): 345-354.

Loukil, T., J. Teghem, et al. (2007). "A multi-objective production scheduling case study solved by simulated annealing." European Journal of Operational Research **179**(3): 709-722.

Marakas, G. M. (2003). Decision support systems in the 21st century. Upper Saddle River, N.J., Prentice Hall.

Oppen, J. (2004). Arc routing in a node routing environment. Master thesis. Molde, Molde University College**:** 86 bl.

Oppen, J. (2008). Models and solutions for rich transportation problems. PhD thesis. Molde, Molde University College. **2008:1:** IV, 128 s.

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

Sörensen, K. and M. Sevaux (2006). "MAPM: memetic algorithms with population management." Computers & Operations Research **33**(5): 1214-1225.

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

Toth, P. and D. Vigo (2002b). The Vehicle routing problem. Philadelphia, Pa., Society for Industrial and Applied Mathematics.

Appendix A: Description of test cases

In the following tables are described the instances from the literature that have been used here. More general information about them can be found at the web page http://branchandcut.org/VRP/data.

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
A-n32-k5.vrp	31	5	100	0.82
A-n33-k5.vrp	32	5	100	0.89
A-n33-k6.vrp	32	6	100	0.9
A-n34-k5.vrp	33	5	100	0.92
A-n36-k5.vrp	35	5	100	0.88
A-n37-k5.vrp	36	5	100	0.81
A-n37-k6.vrp	36	6	100	0.95
A-n38-k5.vrp	37	5	100	0.96
A-n39-k5.vrp	38	5	100	0.95
A-n39-k6.vrp	38	6	100	0.88
A-n44-k6.vrp	43	6	100	0.95
A-n45-k6.vrp	44	6	100	0.99
A-n45-k7.vrp	44	7	100	0.91
A-n46-k7.vrp	45	7	100	0.86
A-n48-k7.vrp	47	7	100	0.89
A-n53-k7.vrp	52	7	100	0.95
A-n54-k7.vrp	53	7	100	0.96
A-n55-k9.vrp	54	9	100	0.93
A-n60-k9.vrp	59	9	100	0.92
A-n61-k9.vrp	60	9	100	0.98
A-n62-k8.vrp	61	8	100	0.92
A-n63-k9.vrp	62	9	100	0.97
A-n63-k10.vrp	62	10	100	0.93
A-n64-k9.vrp	63	9	100	0.94
A-n65-k9.vrp	64	9	100	0.97
A-n69-k9.vrp	68	9	100	0.94
A-n80-k10.vrp	79	10	100	0.94

Table A.1 Instances Augerat, et al. Set A

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
B-n31-k5.vrp	30	5	100	0.82
B-n34-k5.vrp	33	5	100	0.91
B-n35-k5.vrp	34	5	100	0.87
B-n38-k6.vrp	37	6	100	0.85
B-n39-k5.vrp	38	5	100	0.88
B-n41-k6.vrp	40	6	100	0.95
B-n43-k6.vrp	42	6	100	0.87
B-n44-k7.vrp	43	7	100	0.92
B-n45-k5.vrp	44	4	100	0.97
B-n45-k6.vrp	44	6	100	0.99
B-n50-k7.vrp	49	7	100	0.87
B-n50-k8.vrp	49	8	100	0.92
B-n51-k7.vrp	50	7	100	0.98
B-n52-k7.vrp	51	7	100	0.87
B-n56-k7.vrp	55	7	100	0.88
B-n57-k7.vrp	56	7	100	1
B-n57-k9.vrp	56	9	100	0.89
B-n63-k10.vrp	62	10	100	0.92
B-n64-k9.vrp	63	9	100	0.98
B-n66-k9.vrp	65	9	100	0.96
B-n67-k10.vrp	66	10	100	0.91
B-n68-k9.vrp	67	9	100	0.93
B-n78-k10.vrp	77	10	100	0.94

Table A.2 Instances Augerat, et al. Set B

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
P-n16-k8.vrp	15	8	35	0.88
P-n19-k2.vrp	18	2	160	0.97
P-n20-k2.vrp	19	2	160	0.97
P-n21-k2.vrp	20	2	160	0.93
P-n22-k2.vrp	21	2	160	0.96
P-n22-k8.vrp	21	8	3000	0.94
P-n23-k8.vrp	22	8	40	0.98
P-n40-k5.vrp	39	5	140	0.88
P-n45-k5.vrp	44	5	150	0.92
P-n50-k7.vrp	49	7	150	0.91
P-n50-k8.vrp	49	8	120	0.99
P-n50-k10.vrp	49	10	100	0.95
P-n51-k10.vrp	50	10	80	0.97
P-n55-k7.vrp	54	7	170	0.88
P-n55-k8.vrp	54	8	160	0.81
P-n55-k10.vrp	54	10	115	0.91
P-n60-k10.vrp	59	10	120	0.95
P-n60-k15.vrp	59	15	80	0.95
P-n65-k10.vrp	64	10	130	0.94
P-n70-k10.vrp	69	10	135	0.97
P-n76-k4.vrp	75	4	350	0.97
P-n76-k5.vrp	75	5	280	0.97
P-n101-k4.vrp	100	4	400	0.91

Table A.3 Instances Augerat, et al. Set P

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
E-n22-k4.vrp	21	4	6000	0.94
E-n23-k3.vrp	22	3	4500	0.75
E-n30-k3.vrp	29	3	4500	0.94
E-n33-k4.vrp	32	4	8000	0.92
E-n51-k5.vrp	50	5	160	0.97
E-n76-k7.vrp	75	7	220	0.89
E-n76-k8.vrp	75	8	180	0.95
E-n76-k10.vrp	75	10	140	0.97
E-n76-k14.vrp	75	14	100	0.97
E-n101-k8.vrp	100	8	200	0.91
E-n101-k14.vrp	100	14	112	0.93

Table A.4 Instances Christofides and Eilon. Set CE

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
F-n72-k4.vrp	71	4	30000	0.96

Table A.5 Instance Fisher. Set F

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
G-n262-k25.vrp	261	25	500	0.97

Table A.6 Instance Gillet and Johnson. Set GJ

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
M-n101-k10.vrp	100	10	200	0.91
M-n121-k7.vrp	120	7	200	0.98
M-n151-k12.vrp	150	12	200	0.93
M-n200-k16.vrp	199	16	200	1
M-n200-k17.vrp	199	17	200	0.94

Table A.7 Instances Christofides, Mingozzi, and Toth. Set CMT

Problem Instance	# of Customers	# of Vehicles	Vehicle Capacity	Tightness (Demand/Capacity)
att-n48-k4.vrp	47	4	15	0.73

Table A.8 Instances Converted TSPLIB Problems. Set V

Appendix B: Computational results (general tests)

Table B.1 shows the minimum, average and maximum distance between every pair of solutions, for all the instances.

Pair	Weight associated to the length objective in the solution		Distance		
	Solution 1	Solution 2	minimum	average	maximum
1	0	0.1	0.342227	0.736129	0.823248
2	0	0.2	0.396639	0.732698	0.820853
3	0	0.3	0.375654	0.737475	0.81144
4	0	0.4	0.421857	0.735068	0.820282
5	0	0.5	0.40442	0.735225	0.81309
6	0	0.6	0.382198	0.74067	0.846737
7	0	0.7	0.403344	0.732763	0.816292
8	0	0.8	0.408474	0.735128	0.819334
9	0	0.9	0.396708	0.735343	0.821525
10	0	1	0.481801	0.738356	0.819226
11	0.1	0.2	0.121748	0.705739	0.815913
12	0.1	0.3	0.351224	0.719213	0.801462
13	0.1	0.4	0.33333	0.707763	0.81918
14	0.1	0.5	0.327138	0.70939	0.814121
15	0.1	0.6	0.406758	0.716007	0.805765
16	0.1	0.7	0.354541	0.711296	0.813871
17	0.1	0.8	0.315076	0.713446	0.800967
18	0.1	0.9	0.348079	0.713621	0.813661
19	0.1	1	0.467467	0.719041	0.798547
20	0.2	0.3	0.252878	0.704203	0.797613
21	0.2	0.4	0.225599	0.698001	0.80876
22	0.2	0.5	0.287409	0.700963	0.8148
23	0.2	0.6	0.290259	0.697213	0.796189
24	0.2	0.7	0.340442	0.696615	0.802846
25	0.2	0.8	0.33706	0.704026	0.803649
26	0.2	0.9	0.351811	0.707606	0.796655
27	0.2	1	0.368008	0.707037	0.796301

Table B.1 Minimum, average and maximum distance between every pair of solutions

Pair	Weight associated to the length objective in the solution		Distance		
	Solution 1	Solution 2	minimum	average	maximum
28	0.3	0.4	0.290689	0.693372	0.816241
29	0.3	0.5	0.147461	0.695342	0.794971
30	0.3	0.6	0.289353	0.690956	0.791046
31	0.3	0.7	0.304091	0.695892	0.799123
32	0.3	0.8	0.340634	0.691588	0.792782
33	0.3	0.9	0.222995	0.692425	0.797026
34	0.3	1	0.283227	0.703008	0.818989
35	0.4	0.5	0.285963	0.685424	0.786677
36	0.4	0.6	0.199948	0.679831	0.78976
37	0.4	0.7	0.281329	0.685259	0.798068
38	0.4	0.8	0.379188	0.686271	0.79772
39	0.4	0.9	0.223192	0.673569	0.79565
40	0.4	1	0.325605	0.692165	0.780814
41	0.5	0.6	0.20941	0.672718	0.790199
42	0.5	0.7	0.326304	0.671726	0.803849
43	0.5	0.8	0.12656	0.669255	0.801095
44	0.5	0.9	0.128463	0.666971	0.780255
45	0.5	1	0.292104	0.6747	0.795272
46	0.6	0.7	0.100188	0.655609	0.780001
47	0.6	0.8	0.217877	0.656409	0.770825
48	0.6	0.9	0.222419	0.653559	0.790453
49	0.6	1	0.26212	0.660843	0.763687
50	0.7	0.8	0.193422	0.634099	0.755229
51	0.7	0.9	0.268758	0.638687	0.776149
52	0.7	1	0.274313	0.643776	0.775624
53	0.8	0.9	0.276221	0.628302	0.75097
54	0.8	1	0.232127	0.628201	0.74878
55	0.9	1	0.237615	0.615494	0.770642
Average			0.298795	0.693191	0.798622

Table B.1 (Continuation) Minimum, average and maximum distance between every pair of solutions

Appendix C: Computational results (specific tests)

The following tables show the distance between two pair of solutions. The first pair corresponds to the two most different solutions. The second pair corresponds to the best found solution, defined as the one with the shortest total length, and the one most different to it. All the solutions are above a 10% threshold for the tour balance, that is there are no tours longer than any other by more than 10%. Each of the ten values, given in the tables, corresponds to the results of a search done with a different random seed.

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.809501	0.783867
	0.80484	0.784143
	0.813494	0.797773
	0.810579	0.789613
	0.800282	0.758113
	0.801493	0.779569
	0.807636	0.795002
	0.813051	0.797122
	0.809263	0.797843
	0.803894	0.792434
Best found and most different to it	0.795563	0.783867
	0.791212	0.753193
	0.787855	0.780257
	0.801877	0.772327
	0.78761	0.746268
	0.797284	0.771927
	0.784217	0.784217
	0.797122	0.797122
	0.784156	0.782794
	0.787418	0.757132

Table C.1 Distance between solutions to instance A-n37-k6

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.790457	0.753156
	0.78486	0.754344
	0.786669	0.752136
	0.789427	0.771517
	0.798316	0.766202
	0.774133	0.736472
	0.795235	0.753945
	0.790379	0.763697
	0.787017	0.754668
	0.787658	0.752327
Best found and most different to it	0.754369	0.735537
	0.766642	0.72939
	0.765028	0.726562
	0.780664	0.771517
	0.790742	0.738298
	0.748463	0.712276
	0.749484	0.696004
	0.769201	0.735338
	0.757822	0.741597
	0.770332	0.752327

Table C.2 Distance between solutions to instance A-n36-k5

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.807094	0.793181
	0.814565	0.788209
	0.814211	0.760125
	0.808829	0.784169
	0.807615	0.78666
	0.815547	0.789005
	0.814688	0.814688
	0.81853	0.788646
	0.802362	0.800428
	0.810768	0.784232
Best found and most different to it	0.787745	0.774402
	0.784143	0.769051
	0.791773	0.746129
	0.780775	0.777302
	0.789053	0.780055
	0.788503	0.778958
	0.80905	0.80905
	0.799208	0.754557
	0.781634	0.781634
	0.786036	0.775279

Table C.3 Distance between solutions to instance A-n45-k7

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.819917	0.811121
	0.816874	0.80635
	0.816904	0.802804
	0.819839	0.806226
	0.815229	0.808319
	0.813125	0.800431
	0.824546	0.804525
	0.818441	0.799729
	0.814649	0.79776
	0.819867	0.799197
Best found and most different to it	0.819917	0.811121
	0.803266	0.802434
	0.80902	0.790593
	0.79848	0.794425
	0.808319	0.808319
	0.813125	0.792713
	0.812291	0.798807
	0.798149	0.798015
	0.809751	0.772579
	0.805061	0.784865

Table C.4 Distance between solutions to instance A-n60-k9

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.818427	0.795015
	0.818946	0.798958
	0.818142	0.804779
	0.815723	0.802317
	0.818844	0.802817
	0.815131	0.803137
	0.820849	0.80354
	0.813472	0.812208
	0.816639	0.807412
	0.824063	0.810225
Best found and most different to it	0.812275	0.795015
	0.803674	0.791493
	0.814918	0.791965
	0.810471	0.788339
	0.803979	0.79917
	0.800551	0.780073
	0.804898	0.794231
	0.801447	0.792386
	0.807672	0.802191
	0.809678	0.809678

Table C.5 Distance between solutions to instance A-n80-k10

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.80543	0.785617
	0.805242	0.785002
	0.808195	0.791622
	0.813874	0.780119
	0.81186	0.793135
	0.808082	0.793724
	0.810596	0.773943
	0.799994	0.79677
	0.812433	0.787083
	0.812076	0.783443
Best found and most different to it	0.799463	0.785617
	0.780014	0.772132
	0.793716	0.779394
	0.786467	0.760975
	0.804247	0.774066
	0.795891	0.781314
	0.802	0.773943
	0.778056	0.752515
	0.787083	0.787083
	0.802419	0.773929

Table C.6 Distance between solutions to instance B-n41-k6

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.829325	0.811857
	0.833305	0.821878
	0.835868	0.802763
	0.838528	0.818486
	0.838487	0.8122
	0.827519	0.814243
	0.831718	0.804399
	0.833505	0.81158
	0.832503	0.807936
	0.833557	0.805407
Best found and most different to it	0.819157	0.789944
	0.812555	0.782166
	0.80602	0.802763
	0.817079	0.814168
	0.815043	0.807763
	0.813243	0.809535
	0.808195	0.801091
	0.814239	0.79902
	0.814462	0.806683
	0.81579	0.791976

Table C.7 Distance between solutions to instance B-n57-k9

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.797926	0.7839
	0.800601	0.778976
	0.798194	0.784241
	0.808938	0.766586
	0.801869	0.789284
	0.804737	0.802145
	0.795685	0.77554
	0.801712	0.752048
	0.808501	0.806611
	0.805075	0.759476
Best found and most different to it	0.786798	0.774652
	0.800601	0.778976
	0.784241	0.784241
	0.789183	0.766586
	0.789284	0.789284
	0.804737	0.802145
	0.789411	0.757062
	0.780765	0.752048
	0.808501	0.790049
	0.787902	0.759476

Table C.8 Distance between solutions to instance P-n70-k10

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.722803	0.692585
	0.724105	0.682484
	0.721837	0.677634
	0.724318	0.698225
	0.721311	0.685018
	0.724377	0.646486
	0.720889	0.71119
	0.725771	0.687972
	0.728506	0.696451
	0.719191	0.696008
Best found and most different to it	0.695975	0.645675
	0.704788	0.640656
	0.710855	0.655662
	0.709064	0.698225
	0.701276	0.685018
	0.706648	0.646486
	0.708078	0.688853
	0.704754	0.687972
	0.705593	0.685101
	0.702124	0.680008

Table C.9 Distance between solutions to instance P-n101-k4

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.796049	0.72498
	0.802302	0.802302
	0.803339	0.788769
	0.802864	0.746378
	0.801753	0.783966
	0.787499	0.782867
	0.795185	0.78005
	0.807378	0.779778
	0.800835	0.773772
	0.79428	0.781196
Best found and most different to it	0.775163	0.72498
	0.788116	0.774158
	0.790439	0.788769
	0.782472	0.746378
	0.783499	0.783499
	0.775326	0.756163
	0.793496	0.78005
	0.777516	0.771816
	0.787853	0.773772
	0.781196	0.781196

Table C.10 Distance between solutions to instance E-n76-k10

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.817319	0.789715
	0.81447	0.789189
	0.815384	0.790825
	0.815936	0.783197
	0.810214	0.798085
	0.813767	0.789721
	0.815004	0.800622
	0.815968	0.787644
	0.812573	0.788142
	0.82029	0.780004
Best found and most different to it	0.817319	0.772078
	0.806183	0.787475
	0.812413	0.790825
	0.810885	0.783197
	0.804931	0.780599
	0.809721	0.781308
	0.81454	0.795641
	0.807576	0.782949
	0.806312	0.784661
	0.813021	0.779523

Table C.11 Distance between solutions to instance G-n262-k25

	Solutions set	
Pair of solutions	Good solutions	Non-dominated
Two most different	0.807181	0.754767
	0.811436	0.774051
	0.810818	0.753141
	0.798855	0.77096
	0.808036	0.777034
	0.802962	0.761626
	0.803305	0.766787
	0.805248	0.750969
	0.801943	0.761027
	0.802721	0.763788
Best found and most different to it	0.797525	0.754767
	0.796366	0.756375
	0.802711	0.753141
	0.777814	0.761475
	0.800506	0.763031
	0.794295	0.749145
	0.784321	0.76153
	0.79046	0.750969
	0.79011	0.760148
	0.794941	0.763788

Table C.12 Distance between solutions to instance M-n200-k17