Magisterarbeit

Titel der Magisterarbeit

„A heuristic and exact solution procedure for Period Traveling Salesman Problems”

Verfasserin:

Bak. Elitsa Karaeneva Devrnja

angestrebter akademischer Grad

Magistra der Sozial- und Wirtschaftswissenschaften

Mag. rer. soc. oec

Wien, im Mai 2009

Studienkennzahl lt. Studienblatt: A 066 915

Magisterarbeitsgebiet lt. Studienblatt: Betriebswirtschaft

Betreuer: o. Univ.-Prof. Dipl.-Ing. Dr. Richard F. Hartl

Contents

List of Figures

List of algorithms

List of abbreviations

% - shows the difference between the 2optE and 2optH.

2optE - optimized solution of the exact approach obtained with improvement 2-opt heuristic

2optH - optimized solution of the NNS approach obtained with improvement 2-opt heuristic

a_0 - service duration at the depot = 0

a_j - service duration in order to reach customer j

b – frequency of visit

ba - number of possible visit combinations

BIP - binary integer programming

C - the number of customers

coor-x – x coordinate

coor-y – y coordinate

d_{ij} - distance between customer i and customer j; $i,j \in V_0$. The distance is given as coordinates between the points.

D – maximum route duration per day

D_2 – equals D, free distance capacity at the end of the day, and is needed for the repair procedure

E – edges i,j

Ex – obtained solution with an exact approach, without improvement step

f(h) – function, where *h* is restricted to some feasible region

G - complete, undirected graph with all distances of d_{ij} on edge $(i,j) \in E$. G = (V0, *E*)

Gap% - percentage left till obtaining an optimal solution. Stopping condition of 7200 seconds was applied.

H – obtained solution with an Nearest Neighbor Search approach, without improvement step

i, j – customer

i_b – coordinate of customer i on the graph

i_c - coordinate of customer i on the graph

ILP - integer linear programming

IP - integer programming

k – number of the vehicle

l – number of the day (or depot or vehicle type)

m – number of vehicles

n – number of customers

NNS - Nearest Neighbor Search

NP-complete - nondeterministic polynomial-time hard

PTSP - Period Traveling Salesman Problem

r - is the number of edges exchanged at each iteration by *r-opt* procedures

r_{it} - auxiliary variable, representing a rank of a customer in tour

q_0 – demand in depot = 0

q_i – demand needed for each customer

Q – maximum capacity of the vehicle per day

Q_2 – equals Q, free load capacity at the and of the day, and is needed for the repair procedure

s_i – a sequence of days on which the customer $i \in V_2$ can be visited by the traveling salesman.

S(*i*) – a set S(*i*) of combinations specified for each customer $i \in V_2$, and the visit days are assigned to the customer by selecting one of these combinations.

T - number of days (tours) in which we have to visit the customers, $t \in T$

TSP - Traveling Salesman Problem

V – vertex of all customers, belonging to subsets V_1, V_2 *and* V_3 , $V \subset V_1 \cup V_2 \cup V_3$

V_0 – includes all customers, belonging to vertex V, and Depot

V_1 – a subset of customers, including depot, who have to be visited each day, $V_1 \neq V_2 \neq V_3$

V_2 – a subset of customers, who have to be visited every second day

V_3 – a subset of customers, who have to be visited just one day

x_{ijt} – indicates whether the distance from customer *i* to customer *j* on day *t* is traveled and if they are visited after another $i, j \in V_0, t \in T$

y_{jt} – indicates whether customer *j* will be visited on that particular day

Chapter 1

1. Introduction

At present a lot of companies should make their decisions in a fast way and respond quickly to their customer's demands in order to deal with the rapid growing competitiveness. Time is becoming one of the most valuable company's characteristics – not only the time of taking a decision, but also the time of fulfilling it.

In response to the current market, the delivery time for goods has been drastically shortened. This fact is turning into a challenge for logistic companies, major corporations, as well as for little stores.

The movement of the goods through the entire supply-chain includes the delivery from the production facilities to depots or to the end users directly. In the course of the Globalization, more than 75% of the goods are transported using the European road network.[1]

The classic Traveling Salesman Problem is that is has only one transportation vehicle available, i.e. the Salesman services several customers without capacity constraints. No other restrictions than retuning to his depot of departure affect the route building process.

[1] compare Füllerer et al., (2008)

The Period Traveling Salesman Problems, NP-hard[2] optimization problems, provide a natural generalization of the classic Traveling Salesman Problem[3] and include both time and space decisions at the same time[4].

A brief definition of the Period Traveling Salesman Problems can be stated as fallow: “Customers with known demands and visiting times are served by one or more vehicles of limited capacity. Routes are assumed to start and end at exactly one depot. The primary objective of traveling salesman problems is the minimization of costs.” [5]

Different customers usually require different numbers of visits in a certain timeframe. For example customers with larger demands or smaller storage capacity require more visits than customers with smaller demands or larger storage capacity. This type of problem occurs e.g.:

- in grocery distribution[6]
- soft drink industry[7]
- waste collection[8]
- mail delivery[9]
- lawn-care services[10] and others.

However, to satisfy actual demands additional constraints are complicating the development of appropriate methods. As complexity grows with increasing problem size efficient algorithms are essential to find solutions in an acceptable time period.

Exact algorithms, e. g., described in Laporte and Nobert[11] become highly time intensive as soon as problem instances are increasing. Therefore route construction as well as route improvement heuristics emerged. The latter consist of rules which are applied until no

2 nondeterministic polynomial-time hard
3 see Lawler et al., (1985)
4 see Bodin et al., (1983)
5 see Lawler et al., (1985)
6 see Carter et al., (1996) and see Hemmelmayr et al., (2007)
7 see Golden and Wasil, (1987)
8 see Beltrami , (1974), Russel and Igo, (1979)
9 see Cordeau et al., (1997)
10 see Cordeau et al., (1997)
11 see Laporte and Nobert, (1987)

better solution is found. The proceeding of the successive generation of combinations which will only be accepted if an improvement occurs are leading to usable solutions in an adequate runtime.

The aim of this thesis is to develop algorithms for the Period Traveling Salesmen Problem with two different computer programs: XPRESS and C++. The first algorithm uses the exact method of solving the problem, based on branch-and-bound algorithm. The second algorithm uses heuristic approaches, based on the philosophy of Nearest Neighbor Search (NNS), a simple heuristic, which searches for the closest point. The difficulty of this work lies in the development of those two algorithms. The main goal is thereby the achieved results and their comparison.

The remainder of this work is organized as follows. The fallowing Chapter 2 gives more detailed definition of the Period Traveling Salesman Problem and its extensions regarding to capacity and tour length constraints. In Chapter 3 are presented possible solution methods. In Chapter 4 are described the proposed algorithm for the PTSP, developed with XPRESS, and the heuristic algorithm, developed with C++. In Chapter 5 are represented the obtained computational experiments and conclusions are drawn. Finally, Chapter 6 concludes the paper.

Chapter 2

2. Period Traveling Salesman Problems

The term “Traveling Salesman Problem” may have been used in mathematical circles in 1931-1932. But in 1832, a book was printed in Germany entitled “The Traveling Salesman, how he should be and what he should do to get Commissions and to be Successful in his Business. By a veteran Traveling Salesman”. The book reaches the essence of the TSP... by a proper choice and scheduling of the tour, one can often gain so much time that we have to make some suggestions....The most important aspect is to cover as many locations as possible without visiting a location twice.[12]

The interest in Period Traveling Salesman Problem is motivated by its practical relevance as well as its considerable difficulty. “In the Period Traveling Salesman Problem, a traveling salesman must visit each city a fixed number of times over a given *m*-day planning period. Each city specifies a set of sequences of visit days and the visit days are assigned to the city by selecting one of these sequences. Moreover, for each day of the planning period, a not empty tour must be generated by connecting the salesman home city and the cities that must be visited on that day. The salesman objective is to minimize the total distance traveled over the entire *m*-day period.” [13]

---

[12] see Lawler et al., (1985)
[13] see Paletta (2002)

The Period Traveling Salesman Problem can also be seen as a special case of the Vehicle Routing Problem, where only one vehicle is available[14].

The problem involves designing the optimal set of routes for a vehicle for the purpose of serving a given set of customers, such that the requirements of the customers are fulfilled, all the operational constraints are satisfied, and the global transportation cost is minimized.

Typical characteristics of customers are:

- coordinates of the road graph where customers are located
- amount of goods that should be delivered
- periods when the customers can be served
- times or additional distances required to deliver the goods

On the other side traveling salesman faces given constraints for his operational plan:

- location of the depot
- capacity of the vehicle
- distance capacity

The visit of the customers has to fulfill the following requirements:

- the vehicle has a restriction of the distance it can cover per day
- additional distance, called service duration, must be added to the existing distance
- the packing of the vehicle must not exceed its capacity
- all demand of the customers must be satisfied at once
- only one tour per day is allowed, so that the vehicle can not visit some customers, drive back to depot, load more and then visit other customers

A customer does not necessarily require one visit on every day of the period, and the traveling salesman must visit at least one customer every day, i.e., no daily tour must be empty. In the problem solved in this work there are three different visit-day-combinations.

The first visit-day-combination includes customers $i \in V_1$. Those customers must be visited each single day by the traveling salesman.

[14] see Hammelmayr et al., (2007)

The second visit-day-combination for the customers $i \in V_2$, is a sequence of s_i days on which the customer $i \in V_2$ can be visited by the traveling salesman. Each customer $i \in V_2$ specifies a set S(i) of combinations, and the visit days are assigned to the customer by selecting one of these combinations. The salesman must, thus, visit the customer $i \in V_2$ on the days belonging to the selected combination. The problem described in this work concerns a 4-day planning period, if the customer $i \in V_2$ specifies the two visit-day-combinations {1, 3} and {2, 4}, then the salesman must visit the customer $i \in V_2$ on the days 1 and 3 if the combination {1, 3} is selected or on the days 2 and 4 if the combination {2, 4} is chosen.

The third visit-day-combination consists of customers $i \in V_3$, which must be visited only one time within the m-day visit.

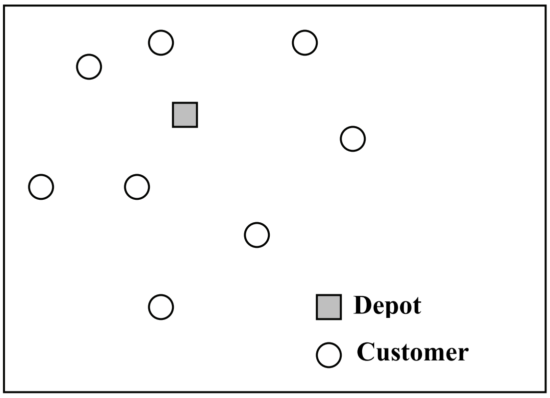


Figure 2.1 Instance of a Traveling Salesman Problem

2.1 Problem Formulation

In order to clarify the problem description a binary integer programming (BIP) model for the PTSP is presented.

If the unknown variables are all required to be integers, then the problem is called an integer programming (IP) or integer linear programming (ILP) problem. In contrast to linear programming, which can be solved efficiently in the worst case, integer programming problems are in many practical situations (those with bounded variables) NP-hard. 0-1 integer programming or binary integer programming (BIP) is the special case of integer programming where variables are required to be 0 or 1 (rather than arbitrary integers). This problem is also classified as NP-hard problem.[15]

The PTSP consist of designing one vehicle route on graph G such that:

- every route starts and ends at the depot
- the total duration of a route serviced by the vehicle does not exceed *D*
- the total load of the vehicle does not exceed *Q*
- for each customer *i*, the visits are on particular days and an additional service time is added

Let G = (V_0, *E*) be a complete, undirected graph with all distances of d_{ij} on edge $(i,j) \in E$. Equivalently one can use distances of d_{ji} on each edge $(j,i) \in E$ due to the fact that the PTSP is symmetric in distances. V_0 – customers, *E* - edges. The vertex set $V_0 = V \cup \{0\} = \{0,1,......,n\}$ denotes the vertices of all customers and vertex 0 corresponds to the depot. If G is a directed graph, the distance matrix is asymmetric, and the corresponding problem is called Asymmetric PTSP.

The location of a customer *i* can result from coordinates (i_b, i_c) representing the customer as points and the calculated Euclidian distance d_{ij} for each edge $(i, j) \in E$ between the two points corresponding to customers *i* and *j*.

[15] compare en.wikipedia.org/wiki/Linear_programming, 25.02.2009

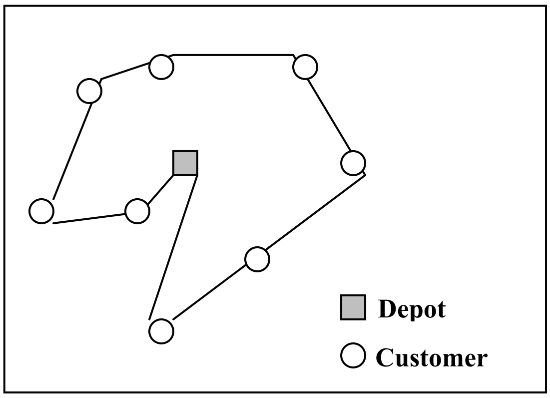


Figure 2.2 Possible solution of a Traveling Salesman Problem

The vertex V has three subsets so that $V \subset V_1 \cup V_2 \cup V_3$.

V_0 – includes all customers and depot $V_0 \subset V \cup Depot$

V_1 – a subset of customers, who have to be visited each day, $V_1 \neq V_2 \neq V_3$

V_2 – a subset of customers, who have to be visited every second day

V_3 – a subset of customers, who have to be visited just one day

T - number of days (tours) on which the customers have to be visited, $t \in T$

d_{ij} - distance between customer i and customer j; $i,j \in V_0$. The distance is given as coordinates between the points

a_j - additional distance, service duration, in order to reach customer j

q_j – demand needed for each customer

Variables:

x_{ijt} – indicates whether the distance from customer *i* to customer *j* on day *t* is traveled and if they are visited after another $i, j \in V_0, t \in T$

y_{it} – indicates whether customer *i* will be visited on that particular day

The objective function can be formulated as follows:

$$\min \sum_{i \in V_0, i \neq j} \sum_{j \in V_0} \sum_{t \in T} x_{ijt} * (d_{ij} + a_j) \tag{1.1}$$

subject to:

$$\begin{aligned} y_{jt} &= 1 && \forall j \in V_1, \forall t \in T \\ y_{0t} &= 1 && \forall t \in T \end{aligned} \tag{1.2}$$

$$\begin{aligned} y_{j1} + y_{j2} &= 1 && \forall j \in V_2 \\ y_{jt} &= y_{jt+2} && \forall j \in V_2 \end{aligned} \tag{1.3}$$

$$\sum_{t \in T} y_{jt} = 1 \qquad \forall j \in V_3 \tag{1.4}$$

$$\sum_{j \in V_0, i \neq j} x_{ijt} = y_{it} \qquad \forall i \in V_0, \forall t \in T \tag{1.5}$$

$$\sum_{i \in V_0, i \neq j} x_{ijt} = y_{jt} \qquad \forall j \in V_0, \forall t \in T \tag{1.6}$$

$$x_{ijt} \in \{0,1\} \qquad \forall i, j \in V_0, t \in T \tag{1.7}$$

$$y_{jt} \in \{0,1\} \qquad \forall j \in V_0, \forall t \in T \tag{1.8}$$

The objective function is the minimization of the total traveling time from the Euclidean distance and the service duration according (1.1). The constraint (1.2) ensures that customers, belonging to subset V_1 and the depot, will be visited every day. Corresponding constraint (1.3) tells us that every customer, belonging to subset V_2, has to be visited exactly 2 times in the given period of days. In the given problem, for example, it should be on the 1^{st} and 3^{rd} day, or on the 2^{nd} and 4^{th} day. The constraint (1.4) tells that every customer, belonging to subset V_3, has to be visited exactly ones. The so-called in- and out-degree constraints (1.5) and (1.6) ensure that if the distance to customer j from no matter which customer is traveled on a particular day, then the customer j must be visited. Constraints (1.7) and (1.8) impose that all edges connecting two customers are binary (equal to one if customer i goes to customer j on day t, respectively if after customer i is visited on day t, and zero otherwise).

2.1.1 Capacity restriction

In the classic TSP there is not capacity restriction. Satisfying real life demands the capacity constraint is imbedded in the PTSP. All customers correspond to deliveries with deterministic demands, known in advance, and may not be split. The vehicle can be based in one and only depot, and the capacity restriction for the vehicle is imposed.

Each customer i = 1,...,n is associated with a known nonnegative demand q_j whereas the depot has a demand $q_0 = 0$. The capacity restriction can be formulated as follows:

$$\sum_{j \in V} y_{jt} * q_j \leq Q \qquad \forall t \in T \tag{1.9}$$

The Q (1.9) constraint guarantees that the capacity restriction will not be exceeded. The total demand can not be higher than the maximum capacity Q of the vehicle. The vehicle can not serve more customers than its capacity permits.

2.1.2 Duration restriction

Customers *i* and *j* = 1,...n are associated with a known nonnegative service time d_{ij}, denoting the distance for the vehicle to travel from customer *i* to customer *j* and a service duration a_j , denoting the time period for which the vehicle must stop at customer *j*. The service duration at the depot a_0 is considered as 0.

$$\sum_{i \in V_0} \sum_{j \in V_0, i \neq j} x_{ijt} * (d_{ij} + a_j) \leq D \quad \forall t \in T \tag{1.10}$$

The *D* (1.10) constraint guarantees that the distance restriction will not be exceeded. The duration of a route serviced by a vehicle, results from the accumulated Euclidean distances d_{ij} and the service duration a_j at each customer *j*. The total amount of service time and service duration may not exceed the maximum route duration *D*.

2.1.3 Sub-tour elimination constraints

There are different ways to introduce sub-tour elimination constraints. The sub-tour elimination constraints are introduced by Miller, Tucker & Zemlin (1960)[16]. The generalized sub-tour elimination constraints (1.11) impose the connectivity of the routes so that all customers are connected between each other.

For each customer $i \neq 1$ is introduced a real auxiliary variable r_{it}, which represents a rank of a customer in tour. The constraint eliminates possible routes, which do not contain customer 1 and builds only routes, which represent acceptable round trips.

$$r_{it} + n * x_{ijt} \leq r_{jt} + n - 1 \qquad \forall i \in \{0,...n\}, j \in \{1,...,n\}, i \neq j, \forall t \in T$$

if $x_{ijt} = 1$ then $r_{it} + n \leq r_{jt} + n - 1$

if $x_{ijt} = 0$ then $r_{it} \leq r_{jt} + n - 1$ (1.11)

[16] see Miller et al., (1960)

Chapter 3

3. Solution methods

3.1. Exact approach

In the context of faster computers, more efficient commercial Linear Problem Solvers (like XPRESS 2007 from DashOptimization) researchers are also interested in providing exact solutions to the PTSP. A PTSP problem solved with XPRESS obtains its globally optimal solution.

According to the book of Toth and Vigo[17] and more resent approaches exact methods can be divided in:

- branch-and-bound algorithms
- branch-and-cut algorithms
- set-covering-based algorithms
- branch-and-cut-and-price algorithms

3.2. Heuristic approach

[17] see Toth and Vigo (2002)

For the heuristic approach there is also variety of algorithms. Some to mention are:

- Nearest Neighbor Search algorithm
- savings algorithm by Clarke and Wright[18]
- sweep algorithm by Gillett and Miller[19]
- Fisher and Jaikumar algorithm by Fisher and Jaikumar[20].

3.2.1. Nearest Neighbor Search (NNS)

Algorithms for Nearest Neighbor Search may be divided into two major groups: partitioning algorithms and graph-based algorithms. Partitioning algorithms partition the data space (or the actual data set) recursively and store information about the partitions in the nodes. Graph-based algorithms recalculate some nearest neighbors of points, store the distances in a graph and use the recalculated information for a more efficient search.[21]

Nearest Neighbor Search (NNS) is a simple heuristic, which searches for the closest point. It is an optimization problem for finding closest points in metric spaces. The problem is: given a set *V* of points in graph *G* and a query point $i \in G$, find the closest point in *V* to *i*. In many cases, *G* is taken to be *i*-dimensional Euclidean space and distance is measured by Euclidean distance or Manhattan distance.

3.2.1.1. Euclidean distance

The **Euclidean distance** for two-dimensional points, $I = (i_b, i_c)$ and $J = (j_b, j_c)$, is computed as: $\sqrt{(i_b - j_b)^2 + (i_c - j_c)^2}$ [22]

[18] see Clarke and Wright (1964)
[19] see Gillett and Miller (1974)
[20] see Fisher and Jaikumar (1981)
[21] see Berchtold et. al., (1997)
[22] compare en.wikipedia.org/wiki/Euclidean_distance, 25.02.2009

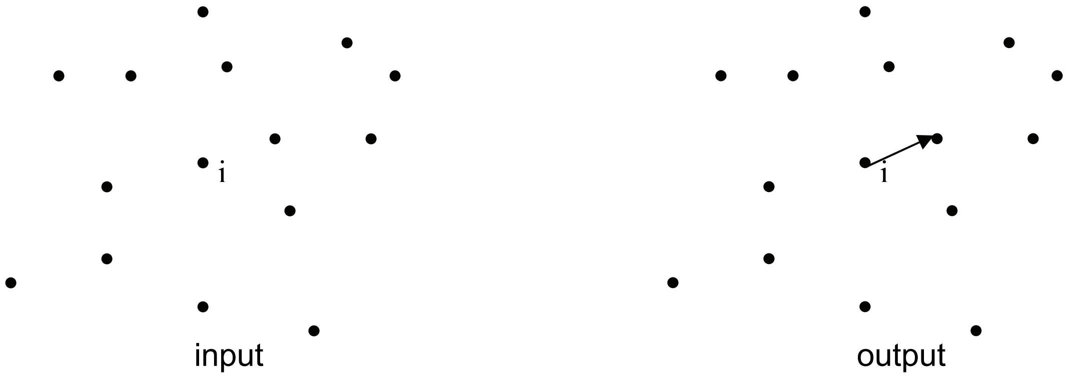


Figure 3.1 Euclidean distance – the distance between point i and nearest neighbor

3.2.1.2. Manhattan distance

Manhattan distance, considered by Hermann Minkowski in the 19th century, is a form of geometry in which the usual metric of Euclidean geometry is replaced by a new metric in which the distance between two points is the sum of the (absolute) differences of their coordinates. The name, **Manhattan distance,** alludes to the grid layout of most streets on the island of Manhattan, which causes the shortest path a car could take between two points in the city to have length equal to the points' distance in taxicab geometry.[23]

3.2.1.3. Manhattan distance versus Euclidean distance

The red, blue, and yellow lines have the same length (12) in both Euclidean and Manhattan geometry. In Euclidean geometry, the green line has length $6\times\sqrt{2} \approx 8.48$, and is the unique shortest path.

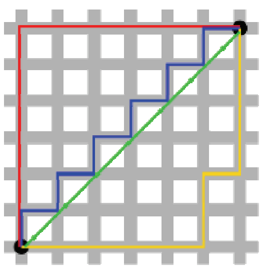


Figure 3.2 Manhattan versus Euclidean distance [24]

[23] see en.wikipedia.org/wiki/Taxicab_geometry, 25.02.2009
[24] see en.wikipedia.org/wiki/Taxicab_geometry, 25.02.2009

3.2.1.4. Nearest Neighbor Search Algorithm

Algorithm 3.1 Nearest Neighbor Search (see Lawler et al., 1985, pp. 150)

1. Start with a partial tour consisting of a single arbitrarily chosen customer *i*
2. calculate the distance d_{ij} for all customer pairs $i, j = 1,...,n; i \neq j$
3. Select the customer *j* with the shortest route to customer *i*
4. Merge these two customers
5. Halt when the current tour contains all customers or constraints fulfilled

"One obvious drawback of this algorithm is the fact that, although all earlier edges are in a sense 'minimal', the final edge *{j,i}* may be quite long. However, as a consequence of the triangle inequality, it can be no longer than the total length of the tour, and hence one might still hope for some meaningful bound on the length of the overall tour. Unfortunately, by being 'greedy' at every step along the way, the nearest neighbor algorithm can get into trouble well before its last edge, with unpleasantly cumulative consequences"[25].

3.2.2. Improvement Heuristics

"Heuristics are criteria, methods, or principles for deciding which among several alternative courses of action promises to be most effective in order to achieve some goal. They represent compromises between two requirements: the need to make such criteria simple and, at the same time, the desire to see them discriminate and correctly between good and bad choices"[26]. Because heuristics are restricted to their rules, these regulations can prevent them to reach the global optima. They alone are often trapped in local optima.

All improvement heuristics known from the TSP can be used for the PTSP, because each route represents a TSP problem. Apart from this intra-route improvement heuristics, there are also inter-route improvement heuristics.

---

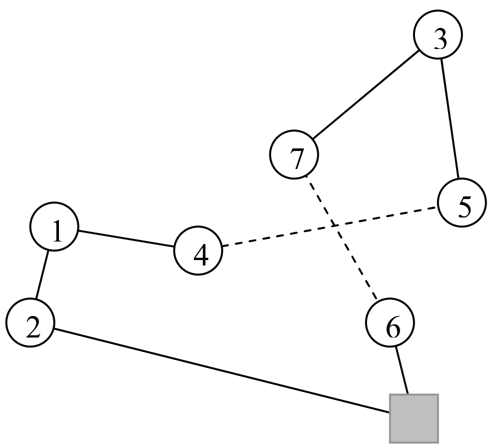
[25] compare Lawler et al., (1985)
[26] see Pearl (1984)

3.2.2.1 Intra-route Improvement Heuristic

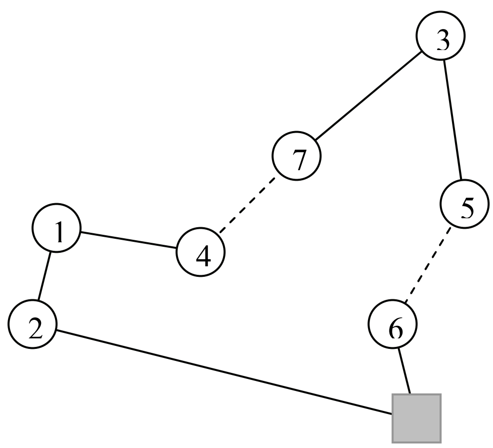
In the general case, *r* edges in a feasible tour are exchanged for *r* edges not in that solution as long as the results remains a tour and the length of that tour is less than the length of the previous tour. Exchange procedures are referred to as *r-opt* procedures where *r* is the number of edges exchanged at each iteration. In r-opt algorithm, all exchanges are tested until there is no feasible exchange that improves the current solution. The solution it then said to be *r-optimal*.[27]

Among simple local search optimization algorithms, the most famous are 2-opt and 3-opt.

- 2-opt procedure is a simple local search algorithm first proposed by Croes in 1958 for solving the TSP, although the basic move had already been suggested by Flood (1956). This move deletes two edges, thus breaking the tour into two paths, and then reconnects those paths in the other possible way. The main idea behind it is to take a route that crosses over itself and reorder it so that it does not. In the example of Figure 1.1 the edges from customer 4 to 5 and 6 to 7 are removed (D – 2 – 1 - 4 x 5 – 3 – 7 x 6 – D) and the tour is reconnected with customer 4 to 7 and 5 to 6 (D – 2 - 1 – 4 – 7 – 3 – 5 – 6 – D). Note that the 2-opt removes all crossings when the triangular inequality holds $(d_{ij} \le d_{ik} + d_{kj})$. Figure 3.3 illustrates the concept.



(a) Solution before 2-opt

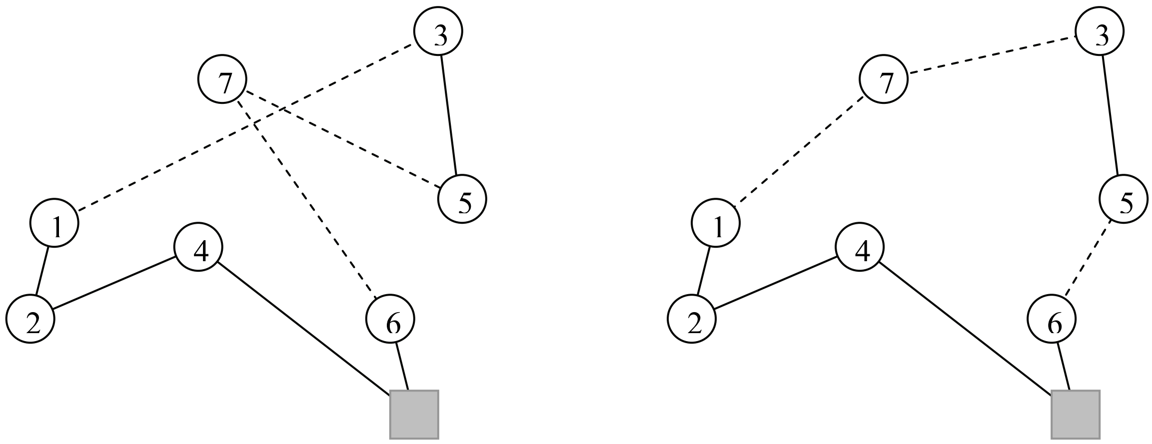


(b) Solution after potential 2-opt

Figure 3.3 Example for 2-opt

[27] see Lawler et al., (1985)

- 3-opt removes 3 edges and reconnects the route fragments. Figure 3.4 shows an example of it. The edges (1,3), (5,7) und (6,7) are removed and replaced by edges (1,7), (3,7) and (5,6). Figure 3.4 illustrates the concept.



(a) Solution before 3-opt (b) Solution after potential 3-opt

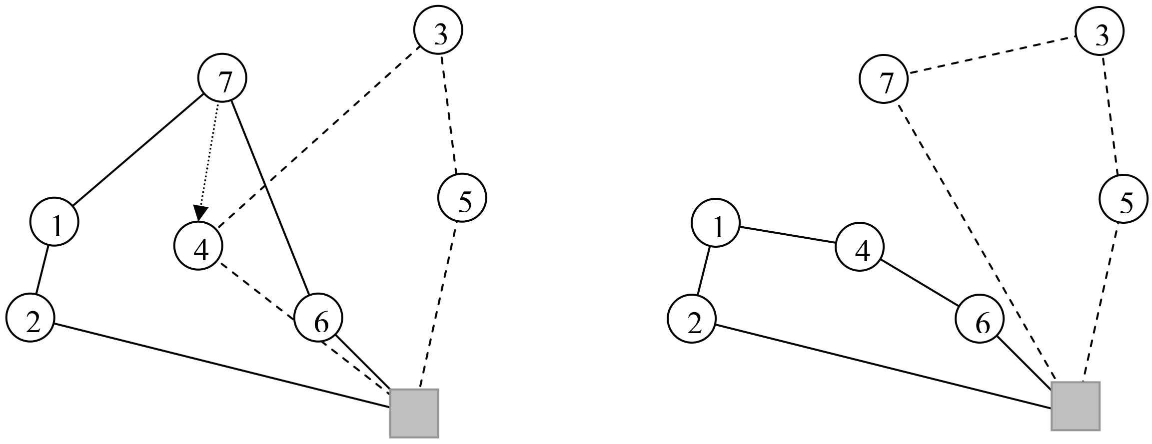
Figure 3.4 Example for 3-opt

3.2.2.2. Inter-route improvement heuristics

Detailed overview on the inter-route improvement heuristics is given from Kindervater and Savelsbergh (1997)[28]. As feasibility check of the loading for one route requires a lot of computational time only inter-route improvement heuristics with simple and small neighborhoods are used within local search:

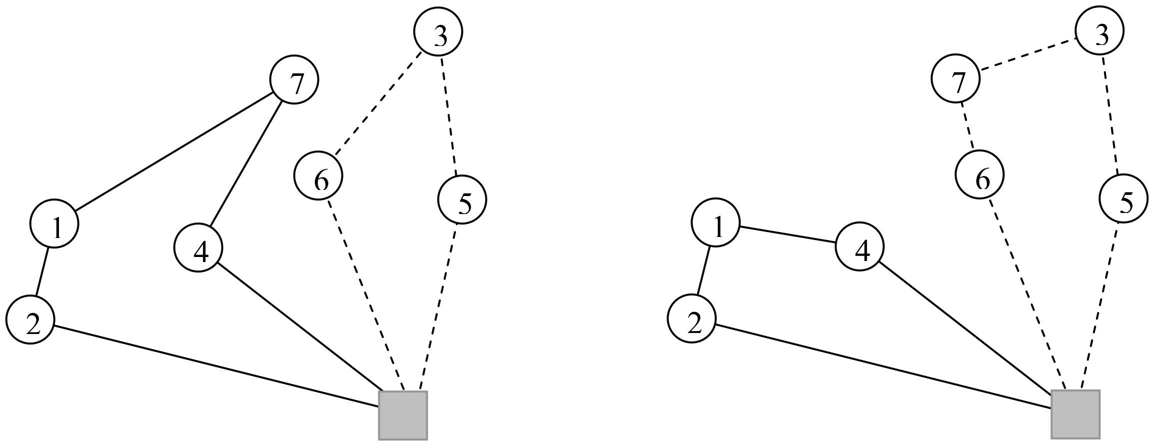
- Swap – the swap is characterized by swapping (exchanging) two customers of two different routes. Figure 3.6 illustrates the concept.
- Move – the move tries to move one customer from one route to another. Figure 3.7 illustrates the concept.

[28] see Kindervater and Savelsbergh (1997)



(a) Solution before Swap (b) Solution after potential Swap

Figure 3.6 Example for Swap



(a) Solution before Move (b) Solution after potential Move

Figure 3.7 Example for Move

Chapter 4

4. Implementation

This chapter explains in details which algorithms are used for solving Period Traveling Salesman Problem. The description comprehends the methods used by the exact and the heuristic algorithms.

4.1. Exact method

Branch and Bound is a general exact search method, used from XPRESS program, for minimizing a function *f(h)*, where *h* is restricted to some feasible region (defined, e.g., by explicit mathematical constraints). To apply branch and bound, one must have a means of computing a lower bound on an instance of the optimization problem and a means of dividing the feasible region of a problem to create smaller subproblems. There must also be a way to compute an upper bound (feasible solution) for at least some instances. The method starts by considering the original problem with the complete feasible region, which is called the root problem. The lower-bounding and upper-bounding procedures are applied to the root problem. If the bounds match, then an optimal solution has been found and the procedure terminates. [29]

[29] compare www.cs.sandia.gov/opt/survey/mip.html, 25.02.2009

The Branch-and-Bound tree, presented on figure 4.1, shows a root problem and a tree of subproblems, which partition the roof problem and cover the whole feasible region. The green nodes on the figure represent an optimal solution found to a subproblem, which is a feasible solution to the full problem, but not necessarily globally optimal. This solution can be used to prune the rest of the tree. The red nodes on the figure represent infeasible solutions. The lower bound of these nodes had exceeded the best known feasible solution and therefore these solutions are removed from consideration. The blue nodes are still not explored.

The search proceeds until all nodes have been solved or pruned, or until some specified threshold is meet between the best solution found and the lower bounds on all unsolved subproblems.[30]

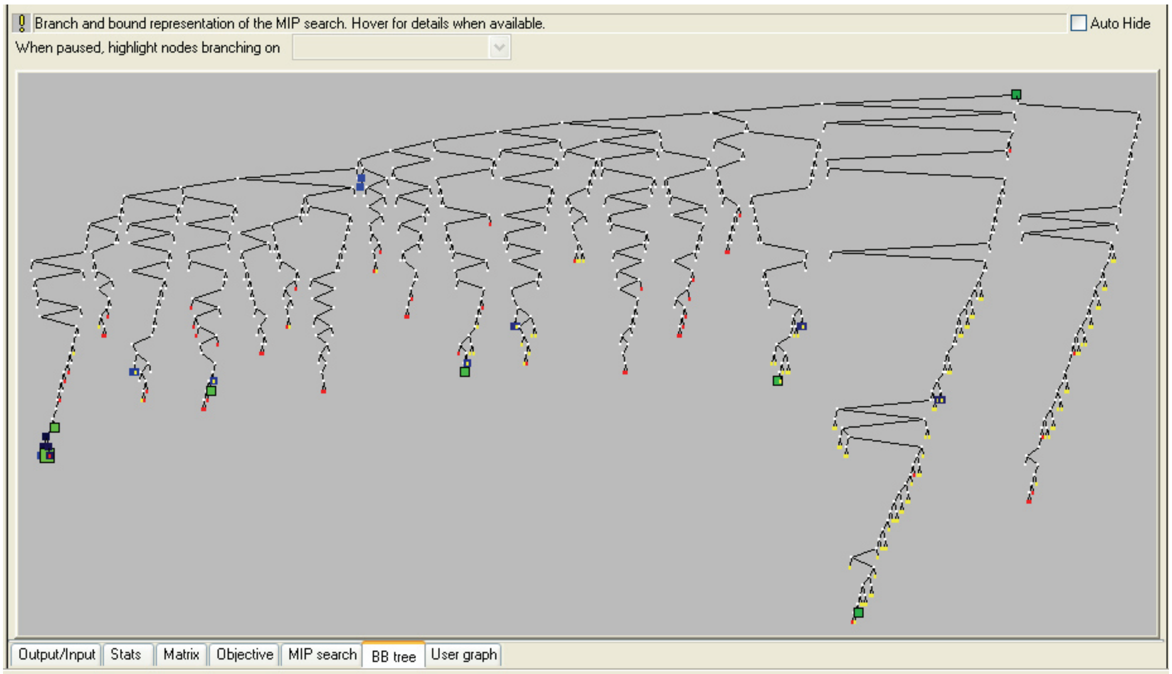


Figure 4.1. Branch and Bound representation of MIP search
■ feasible solution ■ infeasible solution ■ still unexplored nodes

The exact search methods are classified as NP-hard problems (nondeterministic polynomial-time hard), because of the difficulty of solving them. There is still no known

[30] compare www.cs.sandia.gov/opt/survey/mip.html, 25.02.2009

algorithm, that can solve them fast – the most common resources they need are time and space (computational memory). The time required increases extremely quickly as their size grows. To solve large versions of them can take millions of years. Therefore other methods are developed, that help solving those NP-hard problems. The heuristic methods are particularly used to rapidly come to a solution that is reasonably close to the best possible answer, or “optimal solution”.[31]

4.2. Heuristic method

4.2.1. Nearest Neighbor Search

In the core of the heuristic lies the Nearest Neighbor Search, by which the nearest customer is selected for building the route. At the end of the day a procedure checks if all customers, who have to be visited every day, are already in. If not, the program builds another route, where the first customer from the old route is skipped. The visit to the customer, who was dropped, will not be repeated for this day exactly. Nevertheless, not every customer could be skipped, but only the one, who does not belong to the set of V_1 (the set of customers, who have to be visited every single day) and the depot. If customer from this set is at the beginning of the route, the algorithm can not move him, but goes to the next customer, so that the position of this one is already fixed.

The procedure for customers, who have to be visited every second day, differs from the previous one. It registers the visits on the 1st day, but checks not until the end of the second day if all customers, belonging to the set of customers V_2 (those, who have to be visited every second day) are actually visited in these two days. A drawback at this point may be that not all customers are visited on the designated days. Reasons could be no enough distance or capacity available. This case is one of the worse possible cases. The algorithm goes in closed loop and no feasible solution is found. Escape from the situation could be a change in the set of chosen customers.

[31] compare en.wikipedia.org/wiki/Heuristic, 28.02.2009

The 3rd set of customers V_3 (those, who have to be visited once during the planning period) has no specific constraints. Those customers are visited in between. Possible bad situation could be that a customer from this set stays without a visit. To make sure such case does not accrue a check over the sum of the needed capacity is made. If diversity appears an additional modification – repair procedure – is needed.

4.2.2 Repair procedure

Minor adaptations of the Nearest Neighbor Search are proposed in order to fix some of the problematic instances. The applied repair procedure tells that if a customer from set V_3 was dropped as unvisited at the beginning of the tour, it should be proved one more time if the vehicle could still visit him at the end of the day on its way back to depot.

Algorithm 4.1 Additional repair procedure for the excluded customers

---

Initialisation: on the way back to depot if there is still free distance and load capacity the vehicle tries to visit customers, which were already excluded from the tour on this day, by selecting them from a list in the order in which they were excluded

Repeat the fallowing until distance or load capacity exceeded

1. Set $D_2 = D$ (free distance capacity at the end of the day), $Q_2 = Q$ (free load capacity at the end of the day)
2. **while** $D_2>0$ & $Q_2>0$ do
    a. **for** all excluded customers on a particular day
        i. take the last customer on the route as first customer
        ii. take customer from the list with excluded customers if not already visited in this procedure
        iii. connect them
        iv. calculate the new route distance and load capacity
        v. **if** enough distance and load capacity --> **break**
    b. the added customer becomes the last customer on the route
3. D_2 & Q_2 will be updated

4.2.3 2-opt improvement procedure

When feasible solution is found, the next step is the implementation of an improvement heuristic. The 2-opt procedure, as detailed described in 3.2.2.1 Intra-route Improvement Heuristic, exchanges two edges and accepts only an improvements of the distances. The local search restarts immediately after an improving move is found.

An example of the 2-opt local search exchanges is given in Figure 4.2. Here the route consists of 6 customers and the maximal sequence length is 2. In addition, involved customers are marked with a darker background, whereas the dashed line indicates the edge where the 2 sequences are swapped.

2	1	3	4	5	6
1	3	2	4	5	6
1	2	4	3	5	6
1	2	3	5	4	6
1	2	3	4	6	5

3	2	1	4	5	6
1	4	3	2	5	6
1	2	5	4	3	6
1	2	3	6	5	4

Figure 4.2 Example of 2-opt local search

Algorithm 4.2 2-opt improvement procedure algorithm (compare Paletta, 2004)

- take a feasible solution – current route
- set new route < temporary route
- while new route <= temporary route do

  for customer 2 to customer n-1

  take the current customer and change place with current customer-1

  calculate the new distance and save in array

  for customer 3 to customer n-1

  take the current customer and change place with current customer-2

  calculate the new distance and save in array

find shortest route and compare with temporary one
if new route < temporary route take new route as temporary
else break

- if new route < current route show new route

Chapter 5

5. Results and Comparison

In this chapter are presented two outputs of the PTSP implementation. Each version will be described with regard to the applied approaches.

The test cases used in this thesis originate from Cordeau and are available on the internet at http://neumann.hec.ca/chairedistributique/data. Additionally, a description of the instance format is attached in the appendix.

Cordeau has performed the best results through tabu search heuristic. If interested in this topic details can be found in the paper “A tabu search heuristic for periodic and Multu-depot vehicle routing problems” by Cordeau, Gendreau and Laporte[32]. Also Hemmelmayr[33] in her paper “A variable neighborhood search heuristic for periodic routing problems” have given interesting results, based on Variable Neighborhood Search, which even outperform existing solution procedures proposed in the literature.

The exact algorithm in this work was coded in XPRESS IVE 2007. The XPRESS experiments were performed on a PC with 3.2 GHz. The heuristic algorithm was coded in C++ 2005 express edition and performed on laptop with 1.6 GHz.

A comparison of the instances and the results are given in Table 5.1. For each test problem, Table 5.1 shows instance description of set for the PTSP, where:

[32] see Cordeau et al., (1997)
[33] see Hemmelmayr et al., (2007)

- column Instance shows the test problem number,
- C is the number of customers,
- V is the version used. There are either different set of customers in a problem with same number or different distances and capacity.
- n is the number of customers that must be visited *i* times
- D is the maximum duration of a route

Instance	C	V	n	D	Q	Ex	Gap%	2optE	H	2optH	%
1	12		29	500	200	606.388	0,0%	606.388	639.893	634.288	0,9560
2	13	a	28	500	200	1393,81	0,0%	1393,81	1544,06	1535,4	0,9078
3	13	b	29	500	200	1563,9	0,0%	1563,9	1658,69	1619,32	0,9658
4	13	c	37	500	200	1585,97	18,6%	1578,5	1720,71	1627,33	0,9700
5	13	d	29	500	200	1283,12	0,0%	1283,12	1462,42	1402,83	0,9147
6	21		49	500	200	1711,99	29,8%	1648,9	1885,53	1854,92	0,8889
7	22		51	500	200	1727,99	27,0%	1704,53	1910,7	1892,33	0,9008
8	**23**		**53**	**500**	**200**	**1791,9**	**27,2%**	**1759,27**	**1924,13**	**1906,46**	**0,9228**
9	24	a	55	500	200	1613,74	29,2%	1582,04	1711,55	1662,19	0,9518
10	24	b	54	500	200	1819,98	24,9%	1791,55	1944,94	1927,52	0,9295
11	25	a	56	500	200	1595,49	21,2%	1578,87	1702,63	1693,04	0,9326
12	26		57	500	200	1659,23	24,6%	1624,06	1728,62	1705,03	0,9525
13	27		58	500	200	1680,69	23,7%	1645,87	1759,21	1735,63	0,9483
14	**29**		**61**	**500**	**200**	**1791,21**	**25,4%**	**1735,43**	**1887,15**	**1852,51**	**0,9368**
15	**30**		**63**	**500**	**200**	**1913,93**	**27,4%**	**1863,84**	**1925,02**	**1908,68**	**0,9765**
16	31		64	700	450	1976,15	27,4%	1927,19	2036,85	2008,42	0,9596
17	32	a	66	700	450	2119,86	32,3%	2069,77	2175,05	2125,72	0,9737
18	32	b	66	700	450	2124,71	33,0%	2009,69	2136,69	2115,84	0,9498
19	33	a	67	700	450	2073,84	25,7%	2017,6	2196,85	2173,24	0,9284
20	33	b	67	700	450	2191,1	33,2%	2064,87	2147,87	2095,97	0,9852
21	34	a	69	700	450	2241,65	29,3%	2154,08	2238,62	2198,89	0,9796
22	34	b	68	700	450	2286,95	31,9%	2159,2	2199,57	2147,67	1,0054
23	35	a	70	700	450	2198,48	22,9%	2111,13	2303,8	2271,91	0,9292
24	35	b	70	700	450	2410	28,3%	2335,34	2357,64	2303,58	1,0138
25	36		71	700	450	2369,49	29,6%	2260,39	2337,4	2293,5	0,9856
26	37		72	700	450	2414,37	26,5%	2259,21	2449,85	2379,54	0,9494
27	48	a	100	1600	1500	4282,73	55,6%	4018,68	3721,76	3654,77	1,0996
28	48	b	100	1500	1600	4018,41	45,8%	3729,23	3759,83	3668,11	1,0167
Average value						23.651,02	25%	23.580,65	24.882,86	24.644,58	0,9582

Table 5.1: Computational results obtained from 30 PTSP test problems

- Q is the maximum capacity of the vehicle
- Ex is the solution obtained with an exact approach, without improvement step
- Gap% is the percentage left till obtaining an optimal solution. Stopping condition of 7200 seconds was applied.
- 2optE is the optimized solution of the exact approach obtained with improvement 2-opt heuristic
- H is the solution obtained with an Nearest Neighbor Search approach, without improvement step
- 2optH is the optimized solution of the NNS approach obtained with improvement 2-opt heuristic
- % shows the difference between the 2optE and 2optH.
- The number of the vehicles is not presented, because it stays always the same. There is only one vehicle available
- Average value represents the average of the sum of the results

Since the problem of 48 customers is a time-consuming and difficult problem the following steps are preceded. The given problem has initially *D* = 500 (distance restriction per day) and *Q* = 200 units (load restriction for the vehicle also per day). It is impossible to visit all 48 customers respectively 100 visits within these constraints. Simple mathematical calculation shows that the capacity needed for all customers is 1229 units and with one vehicle, that has capacity of 200 units it will be impossible to fulfill the need. Moreover, needed distance for service duration in 4 days is 1025 kilometers, but there is also the distance from customer to customer that should be taken into account. In order to find out how many of the customers can be visited within these restrictions the search was started with a small amount of chosen customers which was extended constantly. Additionally, combinations of the same number of customers (only different sets of them) were also possible.

By comparing the results of both searches for greater instances the heuristic approach shows a significant improvement of run time. Thus, the run time of the exact algorithm was limited by 7200 seconds for many instances it was impossible an optimized solution to be obtained. After running 7200 seconds the gap between the lower and the upper bound of the objective function, performed with branch-and-bound algorithm, lies between 30-45%. Even with so huge gap, it was decided to accept these results for further tests.

Although only twenty eight instances of the exact and heuristic approaches are shown in this chapter more than hundred of test runs have been made to evaluate the algorithms and parameters primarily regarding to the set of neighborhood structures and the value of the chosen distance and capacity restrictions.

The first step was to choose customers randomly. After running many tests it was considered to select for further work customers who lie, if possible, near to each other. For this thesis experiments were made with various combinations of 12 customers. The next step was to increase the number of customers until a point is reached where no additional customer can be added.

As can be seen on Table 5.1 for smaller instances the exact algorithm shows much better results. Though, the time for obtaining them is rather long – in most cases over 2 minutes.

On the smallest instances the branch-and-bound algorithm outperforms the Nearest Neighbor Search algorithm and especially by the instances, where the optimal solution is achieved. The difference in the best solutions found from the two algorithms is about 6-7% in average. This difference can be seen as minimal if compared to the run time needed for obtaining the solution. The branch-and-bound algorithm runs minimum 4 minutes for the smallest instances 1, 2, 3 and 5, since the Nearest Neighbor Search algorithm takes less than few seconds to propose a possible solution.

For greater instances the gap between the exact and heuristic approaches is in some cases even smaller. It is about 2-3%. For example instances 17, 20, 21 and 25. Here the run time for branch-and-bound algorithm has reached the performed limit of 7200 seconds and the solution still lies by average of 31% from the optimal solution.

Moreover, interesting solutions are achieved by the greatest instances as 22, 24, 27 and 28, where the Nearest Neighbor Search algorithm even outperforms the branch-and-bound one, not only by running time, but also by better solutions.

On Table 5.1 there are some reported instances, which showed specific problems. By the instances 8, 14 and 15 some customers from set V_3 were left without a visit. The difference was noted by comparing the results from the branch-and-bound with Nearest Neighbor Search in the part with the delivered units. After investigating the cases was found out that left over customers were the ones, who lay near to the depot. The algorithm starts with visiting them and after running the day it notices that it can not visit all customers from V_1. Therefore the algorithm skips the customers from V_3. This situation repeats every day and at the end some customers stay without a visit. For those three instances an additional modification, repair procedure, described in Chapter 4 was added. This modification though was not applied for the rest of the instances, because it deteriorates the best solution, found till this moment.

Comparison of the results from tabu search by Cordeau and Nearest Neighbor Search and Branch-and-Bound for Instance 28 given in this paper:

Results with Tabu Search			Results with NNS			Results with BB		
Day	Distance	Load	Day	Distance	Load	Day	Distance	Load
1	664.21	334	1	1437.05	635	1	942.19	286
2	887.40	294	2	700.71	157	2	1057.14	380
3	742.58	324	3	919.84	299	3	713.38	206
4	795.66	277	4	610.52	138	4	1016.51	357
Total:	**3089,84**	**1229**		**3668,11**	**1229**		**3729,23**	**1229**

Table 5.2: Comparison of the results with Tabu Search, NNS and BB

The comparison of the two applied approaches in this work can be systemized as advantages and disadvantages for each of them.

Advantages of the branch-and-bound algorithm, representing the exact method:

- gives the optimal solution when running till optimality
- needs no additional control for proving if all customers are visited
- the difference in tour length for each day according to the rest of the days is minimal

Disadvantages of the branch-and-bound algorithm, representing the exact method:

- time-consuming
- computational capacity consuming
- needs an additional optimization heuristic if interrupted
- needs fast computer system
- licensed product
- the student version of the program has no the capacity to solve the problem

Advantages of the Nearest Neighbor Search algorithm, representing the heuristic method:

- time-saving
- computational capacity-saving
- acceptable solutions
- can be applied on slower computers
- the capacity of the student version of the program is enough for solving problems of such instance

Disadvantages of the Nearest Neighbor Search algorithm, representing the heuristic method:

- needs implicitly improvement steps
- results must be proved if all customers are visited
- the heuristic has a drawback, which leads to infeasible solutions in some cases
- The difference in tour length per day is obvious. Usually at the first day the tour length is the longest.
- licensed product

Chapter 6

6. Conclusion

In this thesis were presented two different approaches, exact and heuristic, for solving Period Traveling Salesman Problem. By smaller instances the exact method gives optimal results and the run time is still acceptable. Nevertheless, the reposted heuristic has proven itself as capable method for achieving results of high grade in remarkable short run time.

The advantage of the heuristic method lies primarily in its simplicity maybe even in its primitiveness. In fact the procedure is exempted from elaborate calculations and a lot of memory. So the operating expense is focused on generating new combinations and comparing the solution values thus obtained.

The major difficulty in the development of the heuristic procedure is to find a well balanced composition between an efficient set of neighborhood structures and a method able to escape from the restrictions of this set to avoid being trapped in local optima or performing the drawback.

The exact and heuristic variants have to be analyzed in more details because much room of improvement can be assumed. Especially the exact method that was interrupted after two hours may be left running until the end in order to locate the optimal solution.

Consequently, the implemented heuristic procedure fulfills the qualifications and can be used for practical applications.

Chapter 7

7. Bibliography

[1] Arya S.; *Nearest Neighbor Searching and Applications;* Ph.D. thesis, University of Maryland, College Park, MD, 1995.

[2] Beltrami E.J., Bodin L.D.; *Networks and vehicle routing for municipal waste collection;* Networks, 1974; 4; Pp. 290-306.

[3] Berchtold S., Böhm C., Keim D.A., Kriegel H.P.; *A Cost Model For Nearest Neighbor Search in High-Dimensional Data Space*; University of Munich, Germany, 1997; 2.

[4] Bodin L.D., Golden B. Assad A., Ball M.; *Routing and scheduling of vehicles and crews: The state of the art;* Computers and Operations Research, 1983; 10: Pp. 149-180.

[5] Carter MW, Farvolden J.M., Laporte G., Xu J.; *Solving an integrated logistics problem arising in grocery distribution;* INFOR, 1996; 34; Pp. 290-306.

[6] Clarke G. and Wright J.W.; *Scheduling of vehicles from a central depot do a number of delivery points*; INFORMS, 1964.

[7] Cordeau J.F., Gendreau M., Laporte G.; *A Tabu Search Heuristic for Periodic and Multi-Depot Vehicle Routing Problems*; Networks, 1997; 30; Pp. 105-119.

[8] en.wikipedia.org/wiki/Euclidean_distance

[9] en.wikipedia.org/wiki/Taxicab_geometry

[10] en.wikipedia.org/wiki/Linear_programming

[11] en.wikipedia.org/wiki/Heuristic

[12] Fisher, M. and Jaikumar R.; *A Generalized Assignment Heuristics for Vehicle Routing*; Networks, 1981; 11, 109-124.

[13] Füllerer G., Hartl R., Walter G.; *Vehicle Routing with Multi-Dimensional Loading Constraints*; 2008, Pp. 1

[14] Gillett B.E. and Miller L.R.; A Heuristic Algorithm for the Vehicle-Dispatch Problem; INFORMS, 1974.

[15] Golden B.L., Wasil E.A.; *Computerized vehicle routing in the soft drink industry;* Operations Research, 1987; 35; Pp. 6-17.

[16] Hemmelmayr V.C., Doerner K.F., Hartl R.F.; *A variable neighborhood search heuristic for periodic routing problems*; Department for Business Administration, University of Vienna, Austria, 2007; 2

[17] Kindervater G.A.P. and Savelsbergh M.W.P.; *Vehicle routing: handling edges exchanges windows*. In: *Local Search in Combinatorial Optimization;* Ed. by E. Aarts ans Lenstra JK. John Wiley and Sons Ltd., 1997; Pp. 337-360.

[18] Laporte G. and Nobert Y.; *Exact algorithms for the vehicles routing problem*; Annals of Discrete Mathematics, 1987; 31.

[19] Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G., Shmoys D.B.; *The traveling salesman problem: a guided tour of combinatorial optimization*; New York: John Wiley & Sons, 1985; Pp. 150.

[20] Miller C., Tucker A. and Zemlin R.; *Integer programming formulations and travelling salesman problems*; JACM 7, 1960; Pp. 326–329

[21] Paletta G.; *The Periodic Traveling salesman problem: a new heuristic algorithm;* Computers & Operations Research, 2002; 29; Pp. 1343-52.

[22] Paletta G., Bertazzi L., Speranza M.G.; *An improved heuristic for the period traveling salesman problem*; Computer and Operations Reasearch, 2004; 31; Pp1215-1222.

[23] Pearl J.; *Heuristics: Intelligent Search Strategies for Computer Solving*; Addison-Wesley Publishing Company, 1984; Pp. 3.

[24] Russel R.A. and Igo W.; *An assignment routing problem*; Networks, 1979; 9; Pp. 1-17.

[25] Toth, P. and Vigo D.; *Branch-and-bound algorithms for the capacitated VRP*. In *The vehicle routing problem*; Ed. by P. Toth and Vigo D.; Philadelphia: Society for Industrial and Applied Mathematics, 2002; Pp. 29-51.

[26] www.cs.sandia.gov/opt/survey/mip.html

Appendix A

A. Abstract in German

Heuristisches und exaktes Verfahren zur Lösung von Period Traveling Salesman Probleme

Das Ziel dieser Magisterarbeit ist die Entwicklung von Algorithmen für die Period Traveling Salesman Probleme mit zwei verschiedenen Computer-Programmen: XPRESS und C++. Der Algorithmus, entwickelt mit XPRESS, nutzt die genaue Methode zur Lösung des Problems, basierend auf der Grundlage des Branch-and-bound-Algorithmus. Der Algorithmus, entwickelt mit C++, verwendet heuristische Ansätze, die auf der Philosophie der Nearest Neighbor Suche ("NNS") beziehen, eine einfache Heuristik, die nach dem nächstgelegenen Punkt sucht. Das Hauptziel der Arbeit ist, die erzielten Ergebnisse zu vergleichen.

Der genaue Algorithmus in dieser Arbeit ist in XPRESS IVE 2007 codiert. Die XPRESS Experimente wurden auf einem PC mit 3,2 GHz durchgeführt. Der heuristische Algorithmus ist in C++ 2005 Express Edition codiert und auf Laptop mit 1,6 GHz implementiert.

Der Vorteil der heuristischen Methode liegt in erster Linie in ihrer Einfachheit. In der Tat ist das Verfahren von aufwändigen Berechnungen und von Anforderung viel Arbeitsspeicher ausgenommen. Die betrieblichen Aufwendungen sind auf die Generierung von neuen Kombinationen und den Vergleich der so erhaltenen Werte konzentriert.

Appendix B

B. Abstract in English

„A heuristic and exact solution procedure for Period Traveling Salesman Problems"

The aim of this thesis is to develop algorithms for the Period Traveling Salesmen Problem with two different computer programs: XPRESS and C++. The algorithm developed with XPRESS uses the exact method of solving the problem, based on branch-and-bound algorithm. The algorithm developed with C++ uses heuristic approaches, based on the philosophy of Nearest Neighbor Search (NNS), a simple heuristic, which searches for the closest point. The main goal is thereby the achieved results and their comparison.

The exact algorithm was coded in XPRESS IVE 2007. The XPRESS experiments were performed on a PC with 3.2 GHz. The heuristic algorithm was coded in C++ 2005 express edition and performed on laptop with 1.6 GHz.

The advantage of the heuristic method lies primarily in its simplicity. In fact the procedure is exempted from elaborate calculations and a lot of memory. So the operating expense is focused on generating new combinations and comparing the solution values thus obtained.

Appendix C

C. Curriculum vitae

LEBENSLAUF

Name: Bak. Elitsa Karaeneva Devrnja
Geburtsdatum: 27.3.1977
Geburtsort: Kazanlak, Bulgarien
Familienstand: verheiratet
E-Mail: karaeneva@gmx.net
Nationalität: Bulgarien

Ausbildung

1992 - 1996	**Gymnasium für Marketing und Management,** Bulgarien - Schwerpunkt „Marketing und Management“
1996 - 2000	**Wirtschaftsuniversität,** Sofia, Bulgarien - Schwerpunkt „Buchhaltung und Controlling“
2003 – 2005	**Wirtschaftsuniversität Wien,** Schwerpunkt „Logistikmanagement“
2005 – 2009	**Universität Wien** - Schwerpunkt „Logistikmanagement“ Magisterarbeit „A heuristic and exact solution procedure for Period Traveling Salesman Problems“

Beruflicher Werdegang

2001 – 2003	*Buchhalterin*, **Versicherungsgesellschaft QBE GmbH, Sofia, Bulgarien**
2006 – laufend	*Projektleiterin und Graphikdesignerin,* **Zeitschrift „Bulgaren in Österreich“ (www.bulgaren.org), Wien**
2007 – 2008	*Buchhalterin*, **Elektro- und Solartechnik Hobiger, Wien**
2008 – 2009	*Kostenrechnung und Controlling MOEL,* **Gebrüder Weiss Gesellschaft m.b.H., Wien**

Sonstiges

Fremdsprachen	Deutsch (Sehr gute Kenntnisse in Wort und Schrift) English (Sehr gute Kenntnisse in Wort und Schrift) Russisch (Gute Kenntnisse)
EDV	Microsoft Office (Word, Excel, Power Point), MS-Project, SAP, XPress, Adobe Photoshop, Adobe In-Design
Interessen	Yoga, Schwimmen, Fotographie, Kulturreisen

Appendix D

D. Import Data Structure

The format of data and solution files in all directories is as follows:

A) DATA FILES

The first line contains the following information:

type m n t

where

type = 0 (VRP)
1 (PVRP)
2 (MDVRP)
3 (SDVRP)
4 (VRPTW)
5 (PVRPTW)
6 (MDVRPTW)
7 (SDVRPTW)

m = number of vehicles

n = number of customers

t = number of days (PVRP), depots (MDVRP) or vehicle types (SDVRP)

The next t lines contain, for each day (or depot or vehicle type), the following information:

D Q

where

D = maximum duration of a route

Q = maximum load of a vehicle

The next lines contain, for each customer, the following information:

i coor-x coor-y d q f a list e l

where

i = customer number

coor-x = x coordinate

coor-y = y coordinate

d = service duration

q = demand

b = frequency of visit

ba = number of possible visit combinations

list = list of all possible visit combinations

e = beginning of time window (earliest time for start of service), if any

l = end of time window (latest time for start of service), if any

Each visit combination is coded with the decimal equivalent of the corresponding binary bit string. For example, in a 5-day period, the code 10 which is equivalent to the bit string 01010 means that a customer is visited on days 2 and 4. (Days are numbered from left to right.)

B) SOLUTION FILES

The first lines contain, for each route, the following information:

l k d q list

where

l = number of the day (or depot or vehicle type)

k = number of the vehicle

d = duration of the route

q = load of the vehicle

list = ordered sequence of customers (with start-of-service times, if applicable)

The last line contains the cost of the solution (total duration including service time) and the total load of the vehicle.

1 1 48 4
0 0
0 0
0 0
0 0

i	coor-x	coor-y	d	q	b	ba	list			
0	-10.442	19.999	0	0	0	0				
1	-29.730	64.136	2	12	4	1	15			
2	-30.664	5.463	7	8	4	1	15			
3	51.642	5.469	21	16	4	1	15			
4	-13.171	69.336	24	5	4	1	15			
5	-67.413	68.323	1	12	4	1	15			
6	48.907	6.274	17	5	4	1	15			
7	5.243	22.260	6	13	4	1	15			
8	-65.002	77.234	5	20	4	1	15			
9	-4.175	-1.569	7	13	4	1	15			
10	23.029	11.639	1	18	4	1	15			
11	25.482	6.287	4	7	4	1	15			
12	-42.615	-26.392	10	6	4	1	15			
13	-76.672	99.341	2	9	2	2	5	10		
14	-20.673	57.892	16	9	2	2	5	10		
15	-52.039	6.567	23	4	2	2	5	10		
16	-41.376	50.824	18	25	2	2	5	10		
17	-91.943	27.588	3	5	2	2	5	10		
18	-65.118	30.212	15	17	2	2	5	10		
19	18.597	96.716	13	3	2	2	5	10		
20	-40.942	83.209	10	16	2	2	5	10		
21	-37.756	-33.325	4	25	2	2	5	10		
22	23.767	29.083	23	21	2	2	5	10		
23	-43.030	20.453	20	14	2	2	5	10		
24	-35.297	-24.896	10	19	2	2	5	10		
25	-54.755	14.368	4	14	1	4	1	2	4	8
26	-49.329	33.374	2	6	1	4	1	2	4	8
27	57.404	23.822	23	16	1	4	1	2	4	8
28	-22.754	55.408	6	9	1	4	1	2	4	8
29	-56.622	73.340	8	20	1	4	1	2	4	8
30	-38.562	-3.705	10	13	1	4	1	2	4	8
31	-16.779	19.537	7	10	1	4	1	2	4	8
32	-11.560	11.615	1	16	1	4	1	2	4	8
33	-46.545	97.974	21	19	1	4	1	2	4	8
34	16.229	9.320	6	22	1	4	1	2	4	8
35	1.294	7.349	4	14	1	4	1	2	4	8
36	-26.404	29.529	13	10	1	4	1	2	4	8
37	4.352	14.685	9	11	1	4	1	2	4	8
38	-50.665	-23.126	22	15	1	4	1	2	4	8
39	-22.833	-9.814	22	13	1	4	1	2	4	8
40	-71.100	-18.616	18	15	1	4	1	2	4	8
41	-7.849	32.074	10	8	1	4	1	2	4	8
42	11.877	-24.933	25	22	1	4	1	2	4	8
43	-18.927	-23.730	23	24	1	4	1	2	4	8
44	-11.920	11.755	4	3	1	4	1	2	4	8
45	29.840	11.633	9	25	1	4	1	2	4	8
46	12.268	-55.811	17	19	1	4	1	2	4	8
47	-37.933	-21.613	10	21	1	4	1	2	4	8
48	42.883	-2.966	17	10	1	4	1	2	4	8

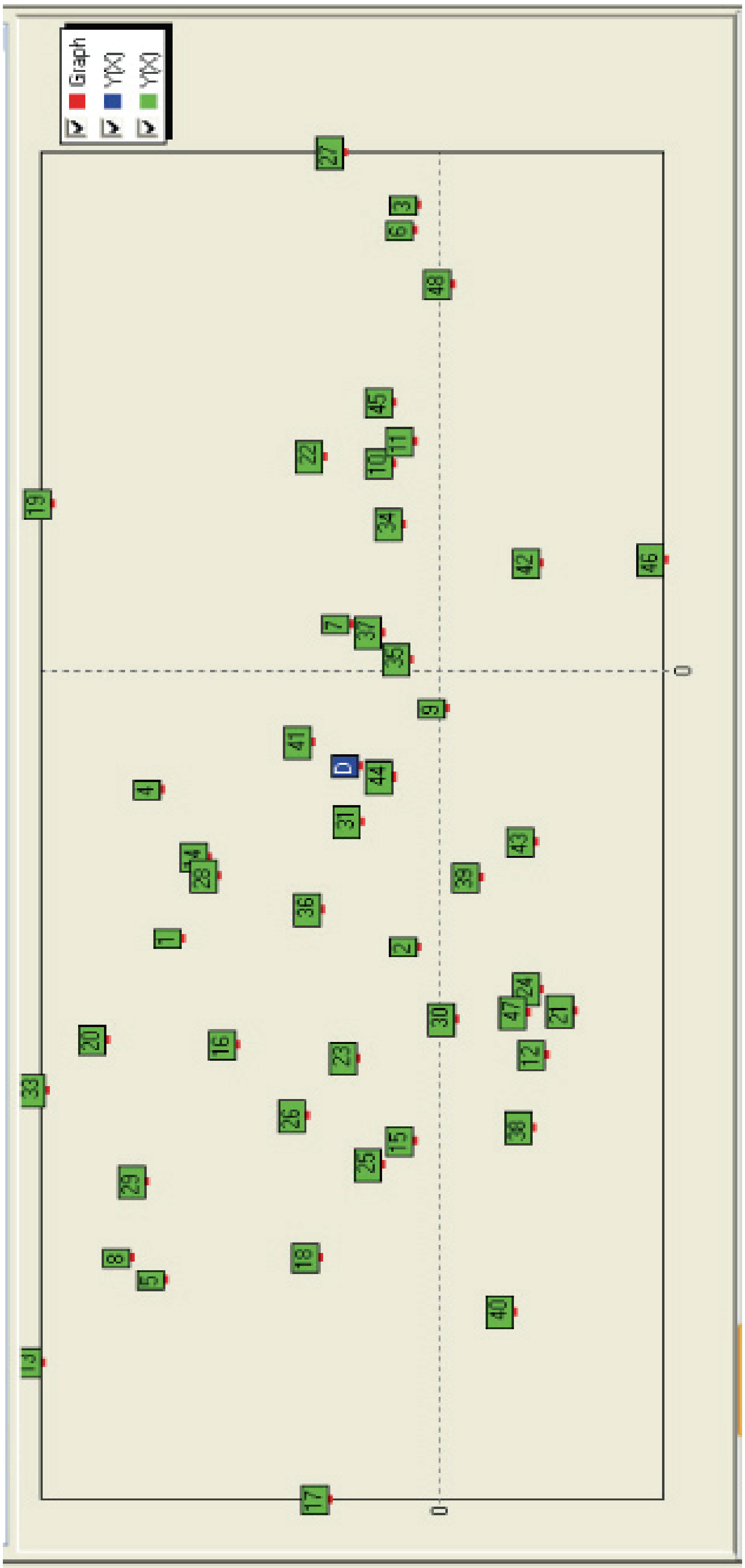


Figure A.1 shows the position of customers on the graph

Appendix E

E. Detailed Route Results

E. 1. Instance 1 – 12 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	136.455	59	0 - 41 - 7 - 10 - 11 - 9 - 0
2 1	182.019	133	0 - 7 - 22 - 45 - 11 - 10 - 34 - 35 - 9 - 0
3 1	134.838	81	0 - 44 - 32 - 9 - 11 - 10 - 37 - 7 - 0
4 1	153.075	72	0 - 9 - 11 - 10 - 22 - 7 – 0
	606.388	**345**	gap: 0%

Heuristic approach

1 1	259.638	171	0-44-32-35-37-7-41-22-10-11-45-34-9-0
2 1	113.59	51	0-7-10-11-9-0
3 1	153.075	72	0-7-22-10-11-9-0
4 1	113.59	51	0-7-10-11-9-0
	639.893	**345**	

2opt for heuristic approach

1 1	254.033	171	0-44-32-35-37-7-41-22-45-11-10-34-9-0
2 1	113.59	51	0-7-10-11-9-0
3 1	153.075	72	0-7-22-10-11-9-0
4 1	113.59	51	0-7-10-11-9-0
	634.288	**345**	

E. 2. Instance 2 – 13 customers

Variant a)

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	458.669	97	0-39-24-38-40-15-17-5-4-14-0
2 1	221.705	42	0-16-5-4-0
3 1	477.777	75	0-14-4-5-17-15-24-48-37-0
4 1	235.657	52	0-4-5-16-36-0
	1393.81	**266**	**gap:0%**

Heuristic approach

1 1	493.876	115	0-37-36-16-14-4-5-17-15-38-24-0
2 1	454.163	55	0-39-40-5-4-48-0
3 1	392.651	79	0-14-4-16-5-17-15-24-0
4 1	203.369	17	0-4-5-0
	1544.06	**266**	

2opt for heuristic approach

1 1	486.913	115	0-37-36-4-14-16-5-17-15-38-24-0
2 1	454.163	55	0-39-40-5-4-48-0
3 1	390.952	79	0-4-14-16-5-17-15-24-0
4 1	203.369	17	0-4-5-0
	1535.4	**266**	

E. 3. Instance 3 – 13 customers

Variant b)

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	416.876	63	0-5-4-19-27-35-9-0
2 1	319.36	52	0-4-5-17-18-9-0
3 1	364.59	56	0-9-44-31-36-5-4-19-0
4 1	463.071	95	0-9-46-43-18-17-5-4-0
	1563.9	**266**	**gap:0%**

Heuristic approach

1 1	444.641	66	0-9-27-19-4-5-18-0
2 1	480.191	78	0-9-43-46-4-5-17-0
3 1	421.264	87	0-31-44-35-9-36-18-5-4-19-0
4 1	312.589	35	0-9-4-5-17-0
	1658.69	**266**	

2opt for heuristic approach

1 1	444.641	66	0-9-27-19-4-5-18-0
2 1	453.518	78	0-9-43-46-17-5-4-0
3 1	420.005	87	0-31-44-9-35-36-18-5-4-19-0
4 1	301.159	35	0-9-17-5-4-0
	1619.32	**266**	

E. 4. Instance 4 – 13 customers

Variant c)

MAXDIST	500
MAXLOAD	200

Exact approach

1 1	366.489	109	0-10-11-3-6-22-1-8-31-0
2 1	411.353	97	0-1-8-24-6-3-11-10-0
3 1	377.004	128	0-1-8-29-28-22-3-6-11-10-0
4 1	431.128	126	0-32-10-11-3-6-24-30-8-1-0
	1585.97	**460**	**gap: 18,6108%**

2opt for exact approach

1 1	366.328	109	0-10-11-6-3-22-1-8-31-0
2 1	410.418	97	0-1-8-24-11-3-6-10-0
3 1	371.559	128	0-8-29-1-28-22-3-6-11-10-0
4 1	430.193	126	0-32-10-6-3-11-24-30-8-1-0
	1578.5	**460**	

Heuristic approach

1 1	457.409	141	0-10-11-22-6-3-28-1-29-8-30-0
2 1	448.587	123	0-31-32-10-11-6-3-24-1-8-0
3 1	390.058	99	0-10-11-22-6-3-1-8-0
4 1	424.657	97	0-10-11-6-3-24-1-8-0
	1720.71	**460**	

2opt for heuristic approach

1 1	424.572	141	0-10-11-3-6-22-28-1-29-8-30-0
2 1	434.837	123	0-31-32-10-11-6-3-24-8-1-0
3 1	357.013	99	0-10-11-3-6-22-1-8-0
4 1	410.907	97	0-10-11-6-3-24-8-1-0
	1627.33	**460**	

E. 5. Instance 5 – 13 customers

Variant d)

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	469.161	133	0-1-8-25-15-2-24-43-42-48-0
2 1	222.343	49	0-2-8-1-14-0
3 1	357.349	99	0-1-8-15-40-47-24-2-0
4 1	234.271	57	0-41-14-1-8-2-0
	1283.12	**338**	**gap:0%**

Heuristic approach

1 1	447.817	161	0-41-14-1-8-25-15-2-47-24-43-42-0
2 1	443.598	65	0-2-40-1-8-48-0
3 1	356.717	72	0-2-15-24-14-1-8-0
4 1	214.284	40	0-2-1-8-0
	1462.42	**338**	

2opt for heuristic approach

1 1	447.817	161	0-41-14-1-8-25-15-2-47-24-43-42-0
2 1	411.566	65	0-2-40-8-1-48-0
3 1	339.186	72	0-2-15-24-8-1-14-0
4 1	204.26	40	0-2-8-1-0
	1402.83	**338**	

E. 6. Instance 6 – 21 customers

MAXDIST	500
MAXLOAD	200

Exact approach

1 1	386.909	114	0-4-14-1-8-5-2-12-24-30-31-0
2 1	442.251	131	0-12-21-2-15-23-8-5-16-1-4-0
3 1	415.44	147	0-2-47-24-12-38-29-8-5-1-14-4-0
4 1	467.39	170	0-36-28-1-4-8-5-16-26-23-25-15-12-21-2-0
	1711.99	**562**	**gap: 29,8427%**

2opt for exact approach

1 1	379.068	114	0-14-4-1-8-5-2-30-12-24-31-0
2 1	413.046	131	0-2-21-12-15-23-16-5-8-1-4-0
3 1	398.461	147	0-2-47-24-12-38-5-8-29-1-4-14-0
4 1	458.319	170	0-36-28-4-1-8-5-16-26-23-25-15-12-21-2-0
	1648.9	**562**	

Heuristic approach

1 1	493.693	182	0-2-25-26-16-1-14-28-4-29-8-5-38-12-47-0
2 1	485.252	138	0-2-30-15-23-1-4-8-5-12-24-21-0
3 1	437.655	117	0-31-36-2-12-16-1-14-4-8-5-0
4 1	468.932	125	0-2-23-15-12-24-21-1-4-8-5-0
	1885.53	**562**	

2opt for heuristic approach

1 1	490.643	182	0-2-25-26-16-1-28-14-4-29-8-5-38-12-47-0
2 1	477.653	138	0-2-30-15-23-1-4-8-5-12-21-24-0
3 1	425.283	117	0-31-12-2-36-16-1-14-4-8-5-0
4 1	461.344	125	0-2-23-15-12-21-24-1-4-8-5-0
	1854.92	**562**	

E. 7. Instance 7 – 22 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	427.693	162	0-24-21-12-2-23-26-16-5-8-1-4-36-0
2 1	461.383	138	0-31-14-1-4-20-8-5-47-12-38-15-2-0
3 1	442.105	188	0-4-28-1-16-29-8-5-23-12-21-24-30-2-0
4 1	396.805	106	0-1-14-4-20-8-5-25-15-12-2-0
	1727.99	**594**	**gap: 27,0428%**

2opt for exact approach

1 1	427.693	162	0-24-21-12-2-23-26-16-5-8-1-4-36-0
2 1	455.147	138	0-31-14-4-1-20-8-5-47-12-38-15-2-0
3 1	436.364	188	0-28-4-1-16-29-8-5-23-12-21-24-30-2-0
4 1	385.324	106	0-14-4-1-20-8-5-25-15-12-2-0
	1704.53	**594**	

Heuristic approach

1 1	499.521	177	0-2-25-26-16-1-14-28-4-20-29-8-5-38-12-0
2 1	480.062	146	0-2-23-15-47-24-12-21-1-4-8-5-0
3 1	462.183	146	0-31-36-2-30-12-16-1-14-4-20-8-5-0
4 1	468.932	125	0-2-23-15-12-24-21-1-4-8-5-0
	1910.7	**594**	

2opt for heuristic approach

1 1	496.471	177	0-2-25-26-16-1-28-14-4-20-29-8-5-38-12-0
2 1	473.031	146	0-2-23-15-47-24-12-21-4-1-8-5-0
3 1	461.481	146	0-31-36-2-12-30-16-1-14-4-20-8-5-0
4 1	461.344	125	0-2-23-15-12-21-24-1-4-8-5-0
	1892.33	**594**	

E. 8. Instance 8 – 23 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	435.062	165	0-31-2-24-21-12-23-5-8-20-4-14-1-28-0
2 1	423.565	142	0-2-30-12-15-18-5-29-8-16-1-4-0
3 1	492.98	182	0-2-21-12-38-24-47-23-14-4-20-8-5-1-0
4 1	440.296	139	0-36-26-16-1-4-8-5-18-25-15-12-2-0
	1791.9	**628**	**gap: 27,1852%**

2opt for exact approach

1 1	424.566	165	0-31-2-24-21-12-23-5-8-20-1-4-14-28-0
2 1	412.652	142	0-2-30-12-15-18-5-8-29-16-1-4-0
3 1	481.751	182	0-2-38-12-21-24-47-23-14-4-20-8-5-1-0
4 1	440.296	139	0-36-26-16-1-4-8-5-18-25-15-12-2-0
	1759.27	**628**	

Heuristic approach with additional improvement step

1 1	498.861	184	0-2-12-38-18-16-1-14-28-4-20-29-8-5-31-0
2 1	493.398	156	0-2-23-15-47-24-12-21-1-4-8-5-36-0
3 1	462.935	163	0-2-30-12-25-18-26-16-1-14-4-20-8-5-0
4 1	468.932	125	0-2-23-15-12-24-21-1-4-8-5-0
	1924.13	**628**	

2opt for heuristic approach with additional improvement step

1 1	495.811	184	0-2-12-38-18-16-1-28-14-4-20-29-8-5-31-0
2 1	486.367	156	0-2-23-15-47-24-12-21-4-1-8-5-36-0
3 1	462.935	163	0-2-30-12-25-18-26-16-1-14-4-20-8-5-0
4 1	461.344	125	0-2-23-15-12-21-24-1-4-8-5-0
	1906.46	**628**	

E. 9. Instance 9 – 24 customers

Variant a)

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	356.358	190	0-7-10-45-6-3-11-35-9-24-12-21-47-2-0
2 1	412.845	133	0-31-2-12-9-11-10-22-3-6-27-7-0
3 1	409.996	191	0-2-30-12-38-21-24-9-37-7-11-3-6-10-34-0
4 1	434.546	149	0-7-11-48-3-6-10-22-32-9-2-12-39-44-0
	1613.74	**663**	**gap: 29,1707%**

2opt for exact approach

1 1	352.646	190	0-7-10-45-6-3-11-35-9-24-21-12-47-2-0
2 1	404.411	133	0-31-2-12-9-11-10-22-27-3-6-7-0
3 1	401.847	191	0-2-30-38-12-21-24-9-37-7-11-3-6-10-34-0
4 1	423.137	149	0-7-11-48-3-6-10-22-32-9-39-12-2-44-0
	1582.04	**663**	

Heuristic approach

1 1	356.358	190	0-7-10-45-6-3-11-35-9-24-12-21-47-2-0
2 1	412.845	133	0-31-2-12-9-11-10-22-3-6-27-7-0
3 1	409.996	191	0-2-30-12-38-21-24-9-37-7-11-3-6-10-34-0
4 1	434.546	149	0-7-11-48-3-6-10-22-32-9-2-12-39-44-0
	1613.74	**663**	

2opt for heuristic approach

1 1	352.646	190	0-7-10-45-6-3-11-35-9-24-21-12-47-2-0
2 1	404.411	133	0-31-2-12-9-11-10-22-27-3-6-7-0
3 1	401.847	191	0-2-30-38-12-21-24-9-37-7-11-3-6-10-34-0
4 1	423.137	149	0-7-11-48-3-6-10-22-32-9-39-12-2-44-0
	1582.04	**663**	

E. 10. Instance 10 – 24 customers

Variant b)

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	495.844	189	0-2-12-21-24-30-31-26-28-14-4-1-5-8-16-36-0
2 1	443.558	129	0-2-38-12-15-23-18-5-8-20-1-4-0
3 1	451.984	195	0-21-12-47-24-39-2-16-5-8-29-1-14-4-0
4 1	428.595	128	0-2-12-15-25-23-18-5-8-20-4-1-0
	1819.98	**641**	**gap: 24,8782%**

2opt for exact approach

1 1	489.782	189	0-2-38-12-15-23-18-5-8-20-1-4-0
2 1	438.135	129	0-2-12-38-15-23-18-5-8-20-1-4-0
3 1	442.708	195	0-47-12-21-24-39-2-16-5-8-29-1-4-14-0
4 1	420.921	128	0-2-12-15-25-23-18-5-8-20-1-4-0
	1791.55	**641**	

Heuristic approach

1 1	489.9	174	0-2-12-38-18-16-1-14-28-4-20-29-8-5-0
2 1	493.795	179	0-2-39-47-24-12-21-15-25-23-26-1-4-8-5-0
3 1	492.312	163	0-31-36-2-30-12-18-16-1-14-4-20-8-5-0
4 1	468.932	125	0-2-23-15-12-24-21-1-4-8-5-0
	1944.94	**641**	

2opt for heuristic approach

1 1	486.85	174	0-2-12-38-18-16-1-28-14-4-20-29-8-5-0
2 1	487.014	179	0-2-39-47-24-21-12-15-25-23-26-1-4-8-5-0
3 1	492.312	163	0-31-36-2-30-12-18-16-1-14-4-20-8-5-0
4 1	461.344	125	0-2-23-15-12-21-24-1-4-8-5-0
	1927.52	**641**	

E. 11. Instance 11 – 25 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	399.667	142	0-7-10-22-27-6-3-11-9-12-2-44-32-0
2 1	388.302	189	0-2-39-24-12-21-43-9-34-11-6-3-10-7-0
3 1	377.472	157	0-2-12-9-10-11-45-6-3-22-7-37-35-0
4 1	430.045	199	0-31-2-30-38-47-21-12-24-9-10-48-6-3-11-7-0
	1595.49	**687**	**gap: 21,2038%**

2opt for exact approach

1 1	396.658	142	0-7-10-22-27-3-6-11-9-12-2-44-32-0
2 1	388.208	189	0-2-39-24-12-21-43-9-34-11-3-6-10-7-0
3 1	377.472	157	0-2-12-9-10-11-45-6-3-22-7-37-35-0
4 1	416.531	199	0-31-2-30-38-12-21-24-47-9-10-6-3-48-11-7-0
	1578.87	**687**	

Heuristic approach

1 1	406.981	179	0-7-10-11-6-3-9-2-30-47-24-12-21-38-0
2 1	492.751	195	0-7-22-10-11-45-48-6-3-27-9-39-43-12-2-0
3 1	395.427	196	0-44-32-35-37-7-34-10-11-6-3-9-2-24-12-21-0
4 1	407.469	117	0-31-2-9-7-22-10-11-6-3-12-0
	1702.63	**687**	

2opt for heuristic approach

1 1	400.528	179	0-7-10-11-6-3-9-2-30-47-24-21-12-38-0
2 1	492.751	195	0-7-22-10-11-45-48-6-3-27-9-39-43-12-2-0
3 1	393.231	196	0-44-32-35-37-7-34-10-3-6-11-9-2-24-12-21-0
4 1	406.888	117	0-31-2-9-7-22-10-3-6-11-12-0
	1693.4	**687**	

E. 12. Instance 12 – 26 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	410.59	159	0-2-24-12-9-35-7-6-3-48-34-11-10-41-0
2 1	409.543	188	0-44-32-43-21-12-30-2-9-10-11-6-3-22-7-0
3 1	359.577	164	0-7-10-11-45-3-6-9-39-24-12-47-2-0
4 1	479.522	184	0-31-2-12-21-38-9-3-6-27-22-10-11-37-7-0
	1659.23	**695**	**gap: 24,552%**

2opt for exact approach

1 1	390.945	159	0-2-12-24-9-35-7-6-3-48-11-10-34-41-0
2 1	409.543	188	0-44-32-43-21-12-30-2-9-10-11-6-3-22-7-0
3 1	358.83	164	0-7-10-11-3-6-45-9-39-24-12-47-2-0
4 1	464.737	184	0-31-2-38-12-21-9-6-3-27-22-10-11-37-7-0
	1624.06	**695**	

Heuristic approach

1 1	406.981	179	0-7-10-11-6-3-9-2-30-47-24-12-21-38-0
2 1	492.751	195	0-7-22-10-11-45-48-6-3-27-9-39-43-12-2-0
3 1	422.089	185	0-41-7-37-35-9-34-10-11-6-3-2-24-12-21-0
4 1	406.797	136	0-31-44-32-9-7-22-10-11-6-3-2-12-0
	1728.62	**695**	

2opt for heuristic approach

1 1	400.528	179	0-7-10-11-6-3-9-2-30-47-24-21-12-38-0
2 1	492.751	195	0-7-22-10-11-45-48-6-3-27-9-39-43-12-2-0
3 1	419.318	185	0-41-7-37-35-9-34-10-3-6-11-2-24-12-21-0
4 1	392.436	136	0-31-44-32-9-7-22-10-11-6-3-12-2-0
	1705.03	**695**	

E. 13. Instance 13 – 27 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	458.86	191	0-2-47-12-9-35-7-11-27-6-3-45-10-22-41-0
2 1	434.159	196	0-44-32-9-2-24-21-12-38-48-6-3-11-34-10-7-0
3 1	404.369	140	0-31-36-2-12-39-9-3-6-11-10-22-7-0
4 1	383.3	178	0-2-30-12-21-24-43-9-10-11-3-6-37-7-0
	1680.69	**705**	**gap: 23,6659%**

2opt for exact approach

1 1	452.576	191	0-2-12-47-9-35-7-11-6-3-27-45-10-22-41-0
2 1	414.632	196	0-44-32-9-2-38-12-21-24-48-3-6-11-10-34-7-0
3 1	397.424	140	0-31-36-2-12-39-9-10-11-6-3-22-7-0
4 1	381.236	178	0-2-30-12-21-24-43-9-10-6-3-11-37-7-0
	1645.87	**705**	

Heuristic approach

1 1	437.575	189	0-7-10-11-6-3-9-2-30-47-24-12-21-38-36-0
2 1	492.751	195	0-7-22-10-11-45-48-6-3-27-9-39-43-12-2-0
3 1	422.089	185	0-41-7-37-35-9-34-10-11-6-3-2-24-12-21-0
4 1	406.797	136	0-31-44-32-9-7-22-10-11-6-3-2-12-0
	1759.21	**705**	

2opt for heuristic approach

1 1	431.122	189	0-7-10-11-6-3-9-2-30-47-24-21-12-38-36-0
2 1	492.751	195	0-7-22-10-11-45-48-6-3-27-9-39-43-12-2-0
3 1	419.318	185	0-41-7-37-35-9-34-10-3-6-11-2-24-12-21-0
4 1	392.436	136	0-31-44-32-9-7-22-10-11-6-3-12-2-0
	1735.63	**705**	

E. 14. Instance 14 – 29 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	434.445	199	0-2-30-47-12-21-9-11-10-6-3-45-7-22-41-0
2 1	436.091	189	0-36-23-25-2-12-24-43-9-11-3-6-10-34-7-0
3 1	491.397	187	0-7-11-22-27-6-3-48-10-35-9-38-21-12-2-0
4 1	429.273	172	0-44-32-37-7-10-11-3-6-9-24-12-39-2-23-31-0
	1791.21	**747**	**gap: 25,3809%**

2opt for exact approach

1 1	405.967	199	0-2-30-47-12-21-9-10-11-45-6-3-22-7-41-0
2 1	436.091	189	0-36-23-25-2-12-24-43-9-11-3-6-10-34-7-0
3 1	474.417	187	0-7-22-11-27-3-6-48-10-35-9-21-12-38-2-0
4 1	418.952	172	0-44-32-37-7-10-6-3-11-9-39-24-12-2-23-31-0
	1735.43	**747**	

Heuristic approach with additional improvement step

1 1	434.395	193	0-7-10-11-6-3-9-2-30-47-24-12-21-38-25-0
2 1	499.791	174	0-7-22-10-11-6-3-27-9-39-43-12-2-23-0
3 1	457.349	197	0-7-34-10-11-45-48-6-3-9-2-36-24-12-21-0
4 1	495.618	183	0-44-32-35-37-7-41-22-10-11-6-3-9-2-23-12-31-0
	1887.15	**747**	

2opt for heuristic approach with additional improvement step

1 1	427.942	193	0-7-10-11-6-3-9-2-30-47-24-21-12-38-25-0
2 1	499.791	174	0-7-22-10-11-6-3-27-9-39-43-12-2-23-0
3 1	443.732	197	0-7-34-10-11-45-48-6-3-9-36-2-24-12-21-0
4 1	481.043	183	0-44-32-35-37-7-41-22-10-11-6-3-9-12-23-2-31-0
	1852.51	**747**	

E. 15. Instance 15 – 30 customers

MAXDIST 500
MAXLOAD 200

Exact approach

1 1	497.298	194	0-31-2-30-21-38-12-9-10-27-6-3-11-22-7-41-0
2 1	498.152	198	0-48-3-6-45-11-10-7-9-32-36-23-25-15-12-24-2-0
3 1	460.06	193	0-2-12-47-21-43-39-44-9-7-22-3-6-11-10-0
4 1	458.42	170	0-37-34-6-3-11-10-7-9-35-23-15-12-24-2-0
	1913.93	**755**	**gap: 27,4302%**

2opt for exact approach

1 1	480.017	194	0-31-2-30-38-12-21-9-10-27-3-6-11-22-7-41-0
2 1	498.152	198	0-48-3-6-45-11-10-7-9-32-36-23-25-15-12-24-2-0
3 1	441.551	193	0-2-47-12-21-43-39-9-44-7-22-3-6-11-10-0
4 1	444.122	170	0-37-34-11-3-6-10-7-35-9-23-15-12-24-2-0
	1863.84	**755**	

Heuristic approach with additional improvement step

1 1	477.3	198	0-7-10-11-6-3-9-2-23-25-15-38-12-47-24-21-0
2 1	496.33	193	0-7-22-10-11-48-6-3-27-9-39-43-12-30-2-31-0
3 1	454.962	198	0-7-34-10-11-45-6-3-9-2-23-15-12-24-21-44-0
4 1	496.427	166	0-41-7-37-35-9-2-36-22-10-11-6-3-12-32-0
	1925.02	**755**	

2opt for heuristic approach with additional improvement step

1 1	469.036	198	0-7-10-11-6-3-9-2-23-25-15-38-12-21-24-47-0
2 1	496.33	193	0-7-22-10-11-48-6-3-27-9-39-43-12-30-2-31-0
3 1	447.471	198	0-7-34-10-11-45-6-3-9-2-23-15-12-21-24-44-0
4 1	495.846	166	0-41-7-37-35-9-2-36-22-10-3-6-11-12-32-0
	1908.68	**755**	

E. 16. Instance 16 – 31 customers

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	534.845	209	0-32-44-36-26-23-15-12-38-47-30-2-9-10-11-3-6-22-7-0
2 1	445.287	173	0-7-41-24-21-12-6-3-45-10-11-9-2-31-0
3 1	572.055	211	0-9-10-11-48-3-6-22-37-7-35-39-43-12-25-15-23-2-0
4 1	423.959	168	0-2-24-21-12-27-3-6-11-10-34-9-7-0
	1976.15	**761**	**gap: 27,4379%**

2opt for exact approach

1 1	523.998	209	0-32-44-36-26-23-15-38-12-47-30-2-9-10-11-6-3-22-7-0
2 1	436.5	173	0-41-7-24-12-21-3-6-45-11-10-9-2-31-0
3 1	548.555	211	0-9-10-11-48-3-6-22-7-37-35-39-43-12-15-25-23-2-0
4 1	418.14	168	0-2-12-21-24-6-3-27-11-10-34-9-7-0
	1927.19	**761**	

Heuristic approach

1 1	692.681	338	0-7-34-10-11-45-22-6-3-48-27-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	543.479	168	0-31-44-32-35-37-7-41-36-26-15-2-9-10-11-6-3-12-0
3 1	440.16	165	0-7-22-10-11-6-3-9-2-23-24-12-21-0
4 1	360.531	90	0-7-10-11-6-3-9-2-15-12-0
	2036.85	**761**	

2opt for heuristic approach

1 1	680.228	338	0-7-34-10-11-45-22-6-3-27-48-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	542.898	168	0-31-44-32-35-37-7-41-36-26-15-2-9-10-3-6-11-12-0
3 1	437.964	165	0-7-22-10-3-6-11-9-2-23-24-12-21-0
4 1	347.331	90	0-7-10-11-6-3-9-12-15-2-0
	2008.42	**761**	

E. 17. Instance 17 – 32 customers

Variant a)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	637.44	242	0-36-16-23-15-30-38-12-25-2-9-34-48-3-6-11-10-22-7-41-0
2 1	432.729	222	0-37-11-10-3-6-45-7-9-43-21-12-24-39-2-44-32-0
3 1	609.329	176	0-7-10-11-22-3-6-27-9-12-31-16-23-15-2-0
4 1	440.358	171	0-7-9-35-10-11-3-6-24-21-12-47-26-2-0
	2119.86	**811**	**gap: 32,2725%**

2opt for exact approach

1 1	631.872	242	0-36-16-23-15-38-12-30-25-2-9-34-6-3-48-11-10-22-7-41-0
2 1	425.817	222	0-37-10-11-3-6-45-7-9-43-21-12-24-39-2-44-32-0
3 1	589.585	176	0-7-22-10-11-6-3-27-9-12-31-16-23-15-2-0
4 1	422.497	171	0-9-35-7-10-6-3-11-24-21-12-47-2-26-0
	2069.77	**811**	

Heuristic approach

1 1	692.681	338	0-7-34-10-11-45-22-6-3-48-27-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	599.235	193	0-31-44-32-35-37-7-41-36-26-16-15-2-9-10-11-6-3-12-0
3 1	440.16	165	0-7-22-10-11-6-3-9-2-23-24-12-21-0
4 1	442.971	115	0-7-10-11-6-3-9-2-15-12-16-0
	2175.05	**811**	

2opt for heuristic approach

1 1	680.228	338	0-7-34-10-11-45-22-6-3-27-48-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	583.442	193	0-31-44-32-35-37-7-41-36-16-26-15-2-9-10-11-6-3-12-0
3 1	437.964	165	0-7-22-10-3-6-11-9-2-23-24-12-21-0
4 1	423.888	115	0-7-10-11-6-3-9-2-12-15-16-0
	2125.52	**811**	

E. 18. Instance 18 – 32 customers

Variant b)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	546.827	231	0-2-24-21-12-15-18-26-23-35-9-10-45-11-3-6-22-7-0
2 1	420.058	134	0-44-11-10-48-6-3-43-12-2-9-37-7-0
3 1	590.601	244	0-2-24-21-12-47-30-18-25-15-23-36-9-10-6-3-11-22-7-0
4 1	567.221	186	0-41-7-34-10-11-32-27-6-3-38-12-39-9-2-31-0
	2124.71	**795**	**gap: 32,9705%**

2opt for exact approach

1 1	532.479	231	0-2-24-21-12-15-18-26-23-9-35-10-11-45-6-3-22-7-0
2 1	405.949	134	0-44-10-11-6-3-48-43-12-2-9-37-7-0
3 1	562.449	244	0-2-24-21-12-47-30-15-25-18-23-36-9-10-11-6-3-22-7-0
4 1	508.813	186	0-32-41-7-34-10-11-27-3-6-12-38-39-9-2-31-0
	2009.69	**795**	

Heuristic approach

1 1	692.681	338	0-7-34-10-11-45-22-6-3-48-27-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	574.659	185	0-31-44-32-35-37-7-41-36-26-18-15-2-9-10-11-6-3-12-0
3 1	440.16	165	0-7-22-10-11-6-3-9-2-23-24-12-21-0
4 1	429.185	107	0-7-10-11-6-3-9-2-15-18-12-0
	2136.69	**795**	

2opt for heuristic approach

1 1	680.228	338	0-7-34-10-11-45-22-6-3-27-48-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	574.078	185	0-31-44-32-35-37-7-41-36-26-18-15-2-9-10-3-6-11-12-0
3 1	437.964	165	0-7-22-10-3-6-11-9-2-23-24-12-21-0
4 1	423.57	107	0-7-10-11-6-3-9-2-18-15-12-0
	2115.84	**795**	

E. 19. Instance 19 – 33 customers

Variant a)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	528.589	181	0-7-11-45-10-9-42-3-27-6-30-24-12-2-0
2 1	631.397	259	0-32-44-10-22-3-6-11-7-37-35-9-43-21-12-2-15-18-23-26-36-41-0
3 1	390.656	171	0-2-39-12-47-24-9-34-10-11-48-6-3-7-0
4 1	523.201	206	0-31-23-18-25-15-38-12-21-2-9-10-11-6-3-22-7-0
	2073.84	**817**	**gap: 25,6583%**

2opt for exact approach

1 1	502.055	181	0-7-10-45-11-9-42-6-3-27-24-12-30-2-0
2 1	606.787	259	0-44-32-22-3-6-11-10-7-37-35-9-43-21-12-2-15-18-26-23-36-41-0
3 1	385.553	171	0-2-39-47-12-24-9-34-10-11-48-3-6-7-0
4 1	523.201	206	0-31-23-18-25-15-38-12-21-2-9-10-11-6-3-22-7-0
	2017.6	**817**	

Heuristic approach

1 1	692.681	338	0-7-34-10-11-45-22-6-3-48-27-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	634.823	207	0-31-44-32-35-37-7-41-36-26-18-15-2-9-42-11-10-6-3-12-0
3 1	440.16	165	0-7-22-10-11-6-3-9-2-23-24-12-21-0
4 1	429.185	107	0-7-10-11-6-3-9-2-15-18-12-0
	2196.85	**817**	

2opt for heuristic approach

1 1	680.228	338	0-7-34-10-11-45-22-6-3-27-48-9-39-43-24-47-12-21-38-30-2-23-25-0
2 1	631.478	207	0-31-44-32-35-37-7-41-36-26-18-15-2-9-42-11-3-6-10-12-0
3 1	437.964	165	0-7-22-10-3-6-11-9-2-23-24-12-21-0
4 1	423.57	107	0-7-10-11-6-3-9-2-18-15-12-0
	2173.24	**817**	

E. 20. Instance 20 – 33 customers

Variant b)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	694.102	237	0-7-9-34-3-11-10-6-18-23-2-24-21-12-38-40-15-25-26-0
2 1	449.542	184	0-2-30-12-47-48-3-6-11-22-10-7-9-35-32-44-0
3 1	586.032	225	0-7-27-3-6-10-11-9-2-43-21-12-24-15-18-23-36-31-0
4 1	461.429	164	0-37-7-10-11-6-3-45-22-41-2-9-12-39-0
	2191.1	**810**	**gap: 33,2132%**

2opt for exact approach

1 1	634.218	237	0-7-9-34-10-6-3-11-2-23-18-24-21-12-38-40-15-25-26-0
2 1	422.964	184	0-2-30-12-47-48-3-6-11-10-22-7-35-9-32-44-0
3 1	577.819	225	0-7-27-3-6-11-10-9-2-43-24-21-12-15-18-23-36-31-0
4 1	429.873	164	0-7-37-10-11-45-6-3-22-41-2-12-39-9-0
	2064.87	**810**	

Heuristic approach

1 1	697.564	296	0-7-10-11-6-3-9-39-43-24-47-12-21-38-40-15-25-23-26-18-36-2-30-0
2 1	597.48	242	0-31-44-32-35-37-7-41-22-10-11-45-34-9-2-12-48-6-3-27-0
3 1	497.473	165	0-7-10-11-6-3-9-2-23-15-18-12-24-21-0
4 1	355.355	107	0-7-22-10-11-6-3-9-2-12-0
	2147.87	**810**	

2opt for heuristic approach

1 1	683.508	296	0-7-10-11-6-3-9-39-43-24-47-12-21-38-40-15-25-18-26-23-36-2-30-0
2 1	591.876	242	0-31-44-32-35-37-7-41-22-45-11-10-34-9-2-12-48-6-3-27-0
3 1	478.435	165	0-7-10-11-6-3-9-2-23-18-15-12-24-21-0
4 1	342.155	107	0-7-22-10-11-6-3-9-12-2-0
	2095.97	**810**	

E. 21. Instance 21 – 34 customers

Variant a)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	508.265	167	0-31-2-9-24-12-38-27-6-3-22-10-11-7-0
2 1	609.09	211	0-2-21-12-9-10-11-3-6-34-7-41-23-15-25-17-18-26-36-0
3 1	550.855	262	0-6-3-48-45-11-37-10-22-7-32-9-39-43-24-12-47-30-2-44-0
4 1	573.438	180	0-7-10-11-3-6-9-35-23-18-17-15-40-21-12-2-0
	2241.65	**820**	**gap: 29,2861%**

2opt for exact approach

1 1	474.622	167	0-31-9-2-38-12-24-6-3-27-11-10-22-7-0
2 1	601.601	211	0-2-12-21-9-10-6-3-11-34-7-41-23-15-25-17-18-26-36-0
3 1	516.396	262	0-48-3-6-45-11-10-22-7-37-32-9-39-43-24-12-47-30-2-44-0
4 1	561.465	180	0-7-10-6-3-11-35-9-23-18-17-15-40-12-21-2-0
	2154.08	**820**	

Heuristic approach

1 1	671.13	278	0-7-10-11-6-3-9-43-24-47-12-21-38-40-15-25-23-26-18-17-30-2-0
2 1	672.502	265	0-31-44-32-35-37-7-41-36-2-39-9-34-10-11-45-22-6-3-48-27-12-0
3 1	539.637	170	0-7-10-11-6-3-9-2-23-15-18-17-12-24-21-0
4 1	355.355	107	0-7-22-10-11-6-3-9-2-12-0
	2238.62	**820**	

2opt for heuristic approach

1 1	668.389	278	0-7-10-11-6-3-9-43-24-47-21-12-38-40-15-25-23-26-18-17-30-2-0
2 1	656.311	265	0-31-44-32-35-37-7-41-36-2-39-9-34-10-11-45-22-6-3-27-48-12-0
3 1	532.037	170	0-7-10-11-6-3-9-2-23-15-18-17-12-21-24-0
4 1	342.155	107	0-7-22-10-11-6-3-9-12-2-0
	2198.89	**820**	

E. 22. Instance 22 – 34 customers

Variant b)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	594.439	206	0-3-6-27-38-12-24-21-43-2-9-11-22-10-7-0
2 1	590.677	229	0-39-12-47-15-2-23-26-18-25-44-9-11-3-6-10-7-37-35-32-31-0
3 1	503.757	188	0-9-2-24-21-12-46-48-6-3-22-11-10-7-41-0
4 1	598.075	206	0-15-18-36-23-2-30-12-40-9-11-45-6-3-10-34-7-0
	2286.95	**829**	**gap: 31,8652%**

2opt for exact approach

1 1	560.055	206	0-27-3-6-24-12-38-21-43-2-9-11-10-22-7-0
2 1	575.867	229	0-39-47-12-2-23-15-25-18-26-44-9-11-3-6-10-7-37-35-32-31-0
3 1	476.667	188	0-9-2-24-12-21-46-48-3-6-11-10-22-7-41-0
4 1	546.61	206	0-36-23-18-15-2-30-40-12-9-11-3-6-45-10-34-7-0
	2159.2	**829**	

Heuristic approach

1 1	697.564	296	0-7-10-11-6-3-9-39-43-24-47-12-21-38-40-15-25-23-26-18-36-2-30-0
2 1	649.174	261	0-31-44-32-35-37-7-41-22-10-11-45-34-9-2-12-46-48-6-3-27-0
3 1	497.473	165	0-7-10-11-6-3-9-2-23-15-18-12-24-21-0
4 1	355.355	107	0-7-22-10-11-6-3-9-2-12-0
	2199.57	**829**	

2opt for heuristic approach

1 1	683.508	296	0-7-10-11-6-3-9-39-43-24-47-12-21-38-40-15-25-18-26-23-36-2-30-0
2 1	643.57	261	0-31-44-32-35-37-7-41-22-45-11-10-34-9-2-12-46-48-6-3-27-0
3 1	478.435	165	0-7-10-11-6-3-9-2-23-18-15-12-24-21-0
4 1	342.155	107	0-7-22-10-11-6-3-9-12-2-0
	2147.67	**829**	

E. 23. Instance 23 – 35 customers

Variant a)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	458.076	176	0-31-36-2-12-24-39-9-11-34-10-3-27-6-7-0
2 1	674.106	255	0-7-11-48-3-6-22-10-35-9-43-21-12-40-17-18-15-25-23-26-2-0
3 1	425.483	181	0-45-10-3-6-11-37-7-44-2-9-12-24-47-32-0
4 1	640.815	217	0-41-28-23-18-17-15-38-21-12-30-2-9-11-10-6-3-22-7-0
	2198.48	**829**	**gap: 22,8665%**

2opt for exact approach

1 1	429.868	176	0-31-36-2-12-24-39-9-34-10-11-6-3-27-7-0
2 1	657.776	255	0-7-11-48-3-6-22-10-35-9-43-21-12-40-17-18-26-25-15-23-2-0
3 1	387.29	181	0-10-45-6-3-11-7-37-9-44-2-47-12-24-32-0
4 1	636.193	217	0-41-28-23-18-17-15-38-12-21-30-2-9-10-11-6-3-22-7-0
	2111.13	**829**	

Heuristic approach

1 1	671.13	278	0-7-10-11-6-3-9-43-24-47-12-21-38-40-15-25-23-26-18-17-30-2-0
2 1	686.231	245	0-41-7-37-35-9-39-2-36-28-22-10-11-45-34-48-6-3-27-12-0
3 1	591.079	199	0-31-44-32-9-7-10-11-6-3-2-23-15-18-17-12-24-21-0
4 1	355.355	107	0-7-22-10-11-6-3-9-2-12-0
	2303.8	**829**	

2opt for heuristic approach

1 1	668.389	278	0-7-10-11-6-3-9-43-24-47-21-12-38-40-15-25-23-26-18-17-30-2-0
2 1	677.888	245	0-41-7-37-35-9-39-2-36-28-22-10-34-45-11-48-6-3-27-12-0
3 1	583.479	199	0-31-44-32-9-7-10-11-6-3-2-23-15-18-17-12-21-24-0
4 1	342.155	107	0-7-22-10-11-6-3-9-12-2-0
	2271.91	**829**	

E. 24. Instance 24 – 35 customers

Variant b)

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	518.313	209	0-7-22-6-3-11-10-9-30-2-32-44-12-21-24-26-36-31-0
2 1	694.335	211	0-23-2-12-38-15-18-14-10-45-11-9-35-6-27-3-37-7-0
3 1	505.596	194	0-7-11-10-22-6-3-46-12-24-21-43-2-9-0
4 1	691.753	233	0-14-18-25-23-2-15-40-47-12-39-9-7-34-11-10-3-6-48-41-0
	2410	**847**	**gap: 28,2528%**

2opt for exact approach

1 1	512.647	209	0-7-22-3-6-11-10-9-30-2-44-32-24-21-12-26-36-31-0
2 1	677.484	211	0-23-2-12-38-15-18-14-45-10-35-9-11-6-3-27-7-37-0
3 1	474.175	194	0-7-22-10-11-6-3-46-21-12-24-43-9-2-0
4 1	671.034	233	0-14-18-25-23-2-15-40-12-47-39-9-7-34-10-11-48-3-6-41-0
	2335.34	**847**	

Heuristic approach

1 1	697.564	296	0-7-10-11-6-3-9-39-43-24-47-12-21-38-40-15-25-23-26-18-36-2-30-0
2 1	668.777	254	0-31-44-32-35-37-7-41-14-22-10-11-45-34-9-2-12-46-48-6-3-0
3 1	550.059	181	0-7-10-11-6-3-27-9-2-23-15-18-12-24-21-0
4 1	441.242	116	0-7-22-10-11-6-3-9-2-12-14-0
	2357.64	**847**	

2opt for heuristic approach

1 1	683.508	296	0-7-10-11-6-3-9-39-43-24-47-12-21-38-40-15-25-18-26-23-36-2-30-0
2 1	663.173	254	0-31-44-32-35-37-7-41-14-22-45-11-10-34-9-2-12-46-48-6-3-0
3 1	531.021	181	0-7-10-11-6-3-27-9-2-23-18-15-12-24-21-0
4 1	425.873	116	0-7-22-10-11-6-3-9-12-2-14-0
	2303.58	**847**	

E. 25. Instance 25 – 36 customers

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	662.162	254	0-41-28-36-23-26-17-18-25-15-2-30-21-12-24-43-9-10-3-6-11-7-0
2 1	428.199	145	0-12-32-10-22-6-3-11-34-7-9-2-0
3 1	691.869	276	0-2-23-18-17-15-38-21-24-47-12-31-44-9-42-3-6-48-45-10-11-7-0
4 1	587.264	176	0-2-39-12-40-6-3-27-9-11-10-22-7-37-35-0
	2369.49	**851**	**gap: 29,5788%**

2opt for exact approach

1 1	650.427	254	0-41-28-36-23-26-18-17-25-15-2-30-12-21-24-43-9-10-6-3-11-7-0
2 1	421.318	145	0-32-12-10-22-3-6-11-34-7-9-2-0
3 1	663.774	276	0-2-23-18-17-15-38-12-21-24-47-31-44-9-42-48-3-6-45-11-10-7-0
4 1	524.869	176	0-2-40-12-39-6-3-27-22-10-11-9-35-37-7-0
	2260.39	**851**	

Heuristic approach

1 1	671.13	278	0-7-10-11-6-3-9-43-24-47-12-21-38-40-15-25-23-26-18-17-30-2-0
2 1	692.651	277	0-44-32-35-37-7-41-36-2-39-9-34-10-11-45-22-6-3-48-27-42-12-0
3 1	618.268	189	0-31-2-23-15-18-17-12-24-21-9-7-10-11-6-3-28-0
4 1	355.355	107	0-7-22-10-11-6-3-9-2-12-0
	2337.4	**851**	

2opt for heuristic approach

1 1	668.389	278	0-7-10-11-6-3-9-43-24-47-21-12-38-40-15-25-23-26-18-17-30-2-0
2 1	671.019	277	0-44-32-35-37-7-41-36-2-39-9-34-10-11-45-22-6-3-27-48-42-12-0
3 1	611.941	189	0-31-2-23-15-18-17-12-21-24-9-7-10-11-6-3-28-0
4 1	342.155	107	0-7-22-10-11-6-3-9-12-2-0
	2293.5	**851**	

E. 26. Instance 26 – 37 customers

MAXDIST 700
MAXLOAD 450

Exact approach

1 1	637.231	219	0-7-10-45-6-3-48-11-9-32-44-31-23-25-18-15-12-40-17-2-0
2 1	579.927	210	0-22-3-6-27-10-11-7-46-21-12-24-43-9-2-0
3 1	600.943	193	0-11-3-6-10-35-37-7-9-2-12-47-38-15-26-18-17-23-0
4 1	596.272	248	0-7-22-34-3-6-10-11-9-42-39-21-24-12-30-2-36-28-41-0
	2414.37	**870**	**gap: 26,4725%**

2opt for exact approach

1 1	583.358	219	0-7-10-45-6-3-48-11-9-32-44-31-23-15-25-18-17-40-12-2-0
2 1	559.068	210	0-22-27-3-6-11-10-7-46-21-12-24-43-9-2-0
3 1	557.324	193	0-11-3-6-10-7-37-35-9-2-47-12-38-15-17-18-26-23-0
4 1	559.461	248	0-7-22-34-10-11-6-3-42-9-39-24-21-12-30-2-36-28-41-0
	2259.21	**870**	

Heuristic approach

1 1	671.13	278	0-7-10-11-6-3-9-43-24-47-12-21-38-40-15-25-23-26-18-17-30-2-0
2 1	638.076	212	0-7-34-10-11-45-22-6-3-48-27-9-39-2-36-28-12-0
3 1	686.107	225	0-7-35-9-2-23-15-18-17-12-24-21-42-46-11-10-6-3-0
4 1	454.535	155	0-31-44-32-9-37-7-41-22-10-11-6-3-2-12-0
	2449.85	**870**	

2opt for heuristic approach

1 1	668.389	278	0-7-10-11-6-3-9-43-24-47-21-12-38-40-15-25-23-26-18-17-30-2-0
2 1	609.794	212	0-7-34-10-11-45-22-6-3-48-27-9-39-2-12-28-36-0
3 1	661.182	225	0-7-35-9-2-23-15-18-17-12-24-21-46-42-11-10-6-3-0
4 1	440.174	155	0-31-44-32-9-37-7-41-22-10-11-6-3-12-2-0
	2379.54	**870**	

E. 27. Instance 27 – 48 customers

Variant a)

MAXDIST 1600
MAXLOAD 1500

Exact approach

1 1	1055.56	264	0-12-24-2-30-8-13-5-21-45-10-1-28-14-4-31-6-11-3-36-9-7-0
2 1	1228.55	357	0-23-18-17-5-8-20-19-4-1-16-26-44-2-15-25-27-3-6-40-12-47-38-32-9-10-22-11-37-7-0
3 1	812.544	293	0-41-7-34-11-10-48-3-6-42-1-13-5-8-29-14-4-35-9-21-24-12-2-0
4 1	1186.07	315	0-9-10-11-6-3-4-19-20-33-8-5-17-18-16-1-46-39-12-43-23-15-2-22-7-0
	4282.73	**1229**	**gap: 55,5569%**

2opt for exact approach

1 1	941.135	264	0-2-24-12-30-8-13-5-21-10-45-4-1-14-28-31-11-3-6-7-9-36-0
2 1	1179.28	357	0-23-18-17-5-8-20-19-4-1-16-26-44-2-15-25-27-3-6-47-12-38-40-32-9-11-10-22-7-37-0
3 1	782.984	293	0-41-7-34-10-11-6-3-48-42-1-5-13-8-29-4-14-35-9-24-21-12-2-0
4 1	1115.29	315	0-9-10-11-6-3-19-4-20-33-8-5-17-18-16-1-12-46-43-39-2-15-23-22-7-0
	4018.68	**1229**	

Heuristic approach

1 1	1535.53	654	0-31-44-32-35-37-7-41-36-23-25-15-30-2-39-43-24-47-12-21-38-40-18-26-16-1-14-28-4-20-33-29-8-5-13-17-9-34-10-11-45-22-6-3-48-27-42-46-0
2 1	624.926	138	0-7-10-11-6-3-9-2-12-1-4-19-8-5-0
3 1	936.378	299	0-7-22-10-11-6-3-9-2-23-15-18-17-5-8-20-1-14-4-16-13-12-24-21-0
4 1	624.926	138	0-7-10-11-6-3-9-2-12-1-4-19-8-5-0
	3721.76	**1229**	

2opt for heuristic approach

1 1	1513.9	654	0-31-44-32-35-37-7-41-36-23-25-15-30-2-39-43-24-47-12-21-38-40-18-26-16-1-14-28-4-20-33-29-8-5-13-17-9-34-10-11-45-22-6-3-27-48-42-46-0
2 1	610.518	138	0-7-10-11-6-3-9-12-2-1-4-19-8-5-0
3 1	919.836	299	0-7-22-10-11-6-3-9-2-23-15-18-17-5-8-20-1-14-4-13-16-12-24-21-0
4 1	610.518	138	0-7-10-11-6-3-9-12-2-1-4-19-8-5-0
	3654.77	**1229**	

E. 28. Instance 28 – 48 customers

Variant b)
MAXDIST 1500
MAXLOAD 1600

Exact approach

1 1	963.451	286	0-4-19-1-8-13-5-3-6-45-10-11-35-46-42-9-2-39-12-43-34-7-0
2 1	1213.59	380	0-41-36-4-5-20-8-29-1-16-28-14-25-21-9-15-23-26-18-17-2-30-12-24-22-11-3-48-6-7-10-0
3 1	736.99	206	0-7-3-6-11-10-9-2-12-47-4-1-8-5-13-33-19-44-32-0
4 1	1104.38	357	0-31-2-21-12-24-38-40-17-5-16-14-1-4-20-8-27-3-6-22-10-11-9-37-7-18-23-15-0
	4018.41	**1229**	**gap: 45,8233%**

2opt for exact approach

1 1	942.192	286	0-19-4-1-13-8-5-3-6-45-10-11-42-46-9-35-2-39-12-43-34-7-0
2 1	1057.14	380	0-41-36-4-20-29-8-5-16-1-14-28-9-21-15-25-23-26-18-17-2-30-12-24-22-11-48-3-6-10-7-0
3 1	713.382	206	0-7-11-3-6-10-9-47-12-2-4-1-5-8-13-33-19-32-44-0
4 1	1016.51	357	0-31-2-24-21-12-38-40-17-5-16-14-1-8-20-4-27-3-6-11-10-22-7-37-9-15-18-23-0
	3729.23	**1229**	

Heuristic approach

1 1	1458.68	635	0-31-44-32-35-37-7-41-36-23-25-15-30-2-39-43-24-47-12-21-38-40-18-26-16-1-14-28-4-20-33-29-8-5-13-17-9-34-10-11-45-22-6-3-48-27-42-0
2 1	739.844	157	0-7-10-11-6-3-9-2-12-46-1-4-19-8-5-0
3 1	936.378	299	0-7-22-10-11-6-3-9-2-23-15-18-17-5-8-20-1-14-4-16-13-12-24-21-0
4 1	624.926	138	0-7-10-11-6-3-9-2-12-1-4-19-8-5-0
	3759.83	**1229**	

2opt for heuristic approach

1 1	1437.05	635	0-31-44-32-35-37-7-41-36-23-25-15-30-2-39-43-24-47-12-21-38-40-18-26-16-1-14-28-4-20-33-29-8-5-13-17-9-34-10-11-45-22-6-3-27-48-42-0
2 1	700.71	157	0-7-10-11-6-3-9-46-12-2-1-4-19-8-5-0
3 1	919.836	299	0-7-22-10-11-6-3-9-2-23-15-18-17-5-8-20-1-14-4-13-16-12-24-21-0
4 1	610.518	138	0-7-10-11-6-3-9-12-2-1-4-19-8-5-0
	3668.11	**1229**	