UNIVERSITY OF THE AEGEAN

SCHOOL OF BUSINESS

DEPARTMENT OF FINANCIAL AND MANAGEMENT ENGINEERING

POSTGRADUATE PROGRAMME IN ECONOMICS AND MANAGEMENT FOR ENGINEERS

"Single vehicle routing with predefined client sequence and multiple depot returns: The case of two depots"

Georgios Dikas

Chios, December 2010

UNIVERSITY OF THE AEGEAN

SCHOOL OF BUSINESS

DEPARTMENT OF FINANCIAL AND MANAGEMENT ENGINEERING

POSTGRADUATE PROGRAMME IN ECONOMICS AND MANAGEMENT FOR ENGINEERS

Dissertation Submitted
by
Georgios Dikas
in partial fulfillment of the requirements for the
postgraduate programme

Single vehicle routing with predefined client sequence and multiple depot returns: The case of two depots

Dissertation Committee:

Georgios Dounias, Professor (Supervisor)
Department of Financial and Management Engineering

Nikolaos Ampazis, Assistant Professor
Department of Financial and Management Engineering

Dimosthenis Drivaliaris, Assistant Professor
Department of Financial and Management Engineering

Chios, 2010

EXECUTIVE SUMMARY

This thesis examines three interesting cases of the single vehicle routing problem with a predefined client sequence and two load replenishment depots. The cases studied vary with respect to the inventory availability at each depot. Given the location and demand of the clients, we seek the minimum cost route, which includes optimal load replenishment at the depots in order to fully satisfy the client demand. For each case, two solution approaches have been developed: i) A Dynamic Programming algorithm, which obtains the optimal solution for all cases, and ii) a Labeling Algorithm that obtains the optimal solution for the first two cases, and efficient solutions for the third, and most complex, one. The computational efficiency of the two algorithms is studied by solving a wide range of problem instances.

Keywords

Single Vehicle Routing; Multiple Depot Routing, Dynamic Programming for the VRP, Labeling Algorithm for the VRP

Table of Contents

List of Figures

Chapter 1. Introduction

The Vehicle Routing Problem (VRP) is a very well studied problem in the literature over a span of almost six decades. Its variations are closely related to transportation and distribution operations, telecommunication network operations, the airline industry, etc. (see Golden *et al.* (2008), Toth and Vigo, (2002), or Laporte, (1992)).For characteristic examples of exact, heuristic and metaheuristic solution algorithms see Laporte *et al.* (1985), Fisher *et al.*, (1997), Clarke and Wright, (1964), Osman (1993),Kirkpatrick *et al.* (1983), Gendreau *et al.* (1994), Rego and Roucairol, (1996), and Reinmann and Doerner (2004).

The (single) Vehicle Routing with Depot Returns Problem (VRDRP), as defined by Tsirimpas *et al.* (2008), considers the case of a single vehicle serving clients with a predefined visit sequence and known product demand. Due to load and capacity constraints, the vehicle must return one or more times to the depot for replenishment. The problem is to find the optimal visits to the depot so that the total distance traveled is the minimum possible and all clients are served. Tsirimpas *et al.* (2008) studied three cases of this problem and used dynamic programming to identify the optimal route for each case. Yang *et al.* (2000) and Tatarakis and Minis (2009) consider the case of the stochastic vehicle routing problem with predefined client order. In the first work, the authors investigate problems of a single and multiple-vehicle distributing a single product, and minimize the total routing cost and the probability of route failure. Bianchi *et al.* (2006) study the performance of metaheuristics for solving the Yang *et al.* problem. Tatarakis and Minis consider the delivery of multiple products by a single vehicle to clients with random demand and they developed dynamic programming algorithms to determine the minimum *expected* routing cost.

In the present thesis we consider the case of two depots in three VRDRP variants of increasing complexity. In the first variant (UU) both depots have Unlimited product availability (inventory); in the second variant (UL), one depot has Unlimited product inventory and the other Limited inventory, while in the third variant (LL) both depots have Limited product inventory. In all three variants considered a single product is distributed.

Note that in UU, after each visit to a client, the decision is whether to return -or not- to the nearest depot facility for a full replenishment. In LU or LL, since there exists at least one depot with limited inventory, the decisions to be made after each visit to a client are i) return -or not- to

a depot facility, ii) which depot facility to return to, and iii) how many product units will be loaded, in case the selected depot is the one with the limited inventory.

Figure 1.1, demonstrates the LU variant of the VRDRP. In this simple example, we assume that the vehicle can carry up to two product units; depot D^1 has an inventory of two product units, depot D^2 has unlimited inventory, and each client has a demand of 1 product unit. In Fig. 1.1.a the vehicle departs from its starting position, visits D^1 and departs fully loaded (2 product units). Then, it serves two clients before visiting D^2. Subsequently, it departs from D^2, visits two more clients and, then, since it has exhausted its load, it should return to a depot facility for stock replenishment. However, in D^1 there is no stock left and, thus, the vehicle must travel a longer distance to D^2. Another solution is demonstrated in Fig. 1.1.b. The vehicle departs from its starting position, visits D^1 and loads just 1 product unit (instead of the two available in D^1). After departing from D^1 it visits one client, visits D^2, serves two clients and returns to D^2 for a full replenishment of two product units. Subsequently, after serving one more client, it returns to D^1 to load the one remaining product unit in that facility. Finally, the vehicle serves the last two clients. The two solutions in Figs 1.1.a, 1.1.b below illustrate that the quantity of the product units loaded at depotD^1 (facility with limited availability), affects the total distance traveled by the vehicle.

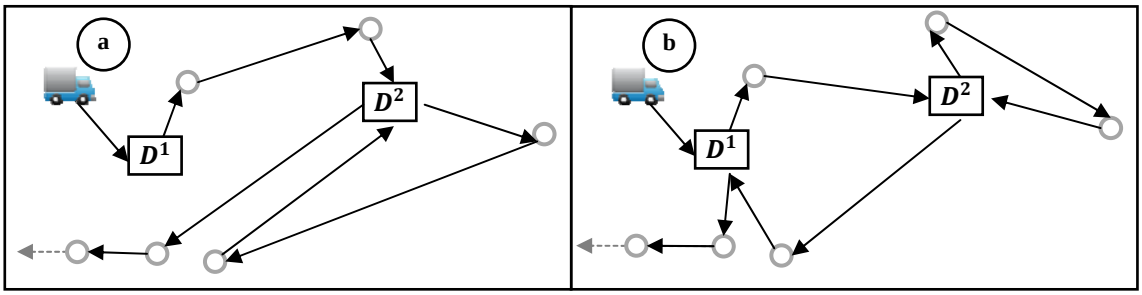

Figure 1.1: Example routes in the LU variant

The problems addressed in this thesis have interesting applications. For instance, material handling systems in a manufacturing shop often operate along a fixed pathway that connects material warehouses with workcenters. Note that in addition to the main pathway connecting the workcenters in a predefined sequence, there are spurs connecting each workcenter with the material warehouses, allowing the direct return of the material handling device (e.g. AGV).

Similar applications may be found in the healthcare industry concerning the transportation of meals, linen, and waste. Another example concerns the distribution of a single product (e.g. bottled water), and the case in which one of the delivery vehicles is incapacitated. In this case the vehicle which, in addition to its own delivery plan, will be assigned to complete the delivery plan of the incapacitated one, may use both the depot and the incapacitated vehicle as replenishment points (see LU problem above).Similar problems have been studied by Li *et al.* (2009 a, b), Mu *et al.* and Mamassis *et al.* (2010).

In this thesis we model, solve and study the complexity of the three new problems (UU, LU and LL) described above. Our solution approaches are based on Dynamic Programming (DP). Furthermore, to deal with the increased complexity of problems LU and LL we also develop efficient labeling algorithms and compare the related computational times with those of the DP ones.

Chapter 2. Problem models

Consider the following notation:

- The vertices in set $C = \{1, \dots, n\}$ correspond to the clients of the network
- Vertex 0 corresponds to the starting position of the vehicle, and vertex $n + 1$ corresponds to the ending position of the vehicle.Note that, without loss of generality, we assume that the vehicle departs empty from its starting position, and, thus, it must visit one of the depots to load product, prior to serving any client
- For modeling purposes, two *virtual depots* are defined for every client, as well as for the starting position. The set of these virtual depots is denoted as $D = \{D_i^1, D_i^2 : i \in C \cup \{0\}\}$. All *virtual depots* D_i^1 are located at the geographical point of Depot 1; similarly all virtual *depots* D_i^2 are located at the geographical point of Depot 2. The inventory level of each *virtual depot* is updated depending on the vehicle visits and the related quantities loaded onto the vehicle.

Consider now the complete directed graph $G(V, A)$, where $V = C \cup D \cup \{0, n + 1\}$ is the vertex set and $A = \{(i, i + 1), (i, D_i^1), (i, D_i^2), (D_i^1, D_i^2) : i \in V \cup \{n + 1\}\}$ is the arc set. The vehicle must serve all clients with a predefined sequence $1, \dots, n$. Each client i has a known positive demand $d_i, \forall\, i \in C$, and $c_{ij} \geq 0$ denotes the cost (distance) of traversing arc (i, j). The product availability in the *virtual depots* after serving client i is denoted as P_i^1 and P_i^2, respectively. Similarly, the initial product availability is Π^1 and Π^2, respectively. The vehicle capacity is Q product units and the stock on the vehicle leaving client i is denoted as q_i.

The solution seeks a vehicle route serving all clients at the predefined sequence with minimal total distance. We present first the model for problem LL (in which both depots have limited inventory), since this is the most general of the three problems.

Problem LL

The decision variables of the model are the following:

$$x_{i,i+1}, x_{i,D_i^1}, x_{i,D_i^2}, x_{D_i^1,D_i^2}, x_{D_i^2,D_i^1}, x_{D_i^1,i}, x_{D_i^2,i} \in \{0,1\}, i = 0, \dots, n$$

Each of the above variables assumes the value 1 if the corresponding arc is traversed and 0 otherwise.

The variables:

$$d_{D_i^1} \in \{0,1,\dots,P_{i-1}^1\}, i = 0,\dots,n$$

and,

$$d_{D_i^2} \in \{0,1,\dots,P_{i-1}^2\}, i = 0,\dots,n$$

represent the quantity of products loaded onto the vehicle while it visits *virtual depots* D_i^1 and D_i^2, respectively.

The objective function of problem LL is defined as follows:

$$minTC = \sum_{i=0}^{n-1}\left(x_{i,i+1}c_{i,i+1} + x_{i,D_i^1}c_{i,D_i^1} + x_{D_i^1,i+1}c_{D_i^1,i+1} + x_{i,D_i^2}c_{i,D_i^2} + x_{D_i^2,i+1}c_{D_i^2,i+1} + x_{D_i^1,D_i^2}c_{D_i^1,D_i^2} + x_{D_i^2,D_i^1}c_{D_i^2,D_i^1}\right) + c_{n,n+1} \quad (2.1)$$

Subject to:

$$x_{i,i+1} + x_{i,D_i^1} + x_{i,D_i^2} = 1 \quad ,i = 0,\dots,n \quad (2.2)$$

$$x_{D_i^1,i+1} + x_{D_i^2,i+1} = x_{i,D_i^1} + x_{i,D_i^2} \quad ,i = 0,\dots,n \quad (2.3)$$

$$x_{i,D_i^1} + x_{D_i^2,D_i^1} = x_{D_i^1,i+1} + x_{D_i^1,D_i^2} \quad ,i = 0,\dots,n \quad (2.4)$$

$$x_{D_i^1,D_i^2} + x_{D_i^2,D_i^1} \le 1 \quad ,i = 0,\dots,n \quad (2.5)$$

$$q_i = q_{i-1} - d_i + \left(\left(x_{i-1,D_{i-1}^1} + x_{D_{i-1}^2,D_{i-1}^1}\right) \times d_{D_{i-1}^1}\right) + \left(\left(x_{i-1,D_{i-1}^2} + x_{D_{i-1}^1,D_{i-1}^2}\right) \times d_{D_{i-1}^2}\right) \quad ,i = 0,\dots,n \quad (2.6)$$

$$q_i + d_i \le Q \quad ,i = 0,\dots,n \quad (2.7)$$

$$P_0^1 = \Pi^1 - \left(x_{0,D_0^1} + x_{D_0^2,D_0^1}\right) \times d_{D_0^1} \quad (2.8)$$

$$P_0^2 = \Pi^2 - \left(x_{0,D_0^2} + x_{D_0^1,D_0^2}\right) \times d_{D_0^2} \quad (2.9)$$

$$P_i^1 = P_{i-1}^1 - \left(x_{i,D_i^1} + x_{D_i^2,D_i^1}\right) \times d_{D_i^1} \quad ,i = 1,\dots,n \quad (2.10)$$

$$P_i^2 = P_{i-1}^2 - \left(x_{i,D_i^2} + x_{D_i^1,D_i^2}\right) \times d_{D_i^2} \quad ,i = 1,\dots,n \quad (2.11)$$

$$P_i^1, P_i^2, q_i \ge 0 \quad ,i = 0,\dots,n \quad (2.12)$$

$$x_{i,i+1}, x_{i,D_i^1}, x_{i,D_i^2}, x_{D_i^1,D_i^2}, x_{D_i^2,D_i^1}, x_{D_i^1,i}, x_{D_i^2,i} \in \{0,1\} \quad ,i = 0, \dots, n \quad (2.13)$$

$$d_{D_i^1} \in \{0,1, \dots, P_{i-1}^1\} \quad ,i = 0, \dots, n \quad (2.14)$$

$$d_{D_i^2} \in \{0,1, \dots, P_{i-1}^2\} \quad ,i = 0, \dots, n \quad (2.15)$$

Equation (2.2) ensures that the vehicle departing from one client will be directed towards the next client, or towards depot 1, or depot 2. Equations (2.3) and (2.4) ensure that if the vehicle arrives at a depot, it will either travel to the other depot or to the next client in the sequence. Inequality (2.5) ensures that the vehicle can only travel once from one depot to the other (thus avoiding cyclical tours between depots). Equations (2.3) to (2.4) and inequality (2.5) also ensure that when the vehicle departs from one client, it will follow only one of the five possible paths to the next customer. Equation (2.6) ensures that upon visiting client i, the vehicle serves exactly its demand. Inequality (2.7) ensures that the vehicle capacity cannot be exceeded at any point along the route. Equations (2.8) and (2.9) define the initial inventory of the two depots. Equations (2.10) and (2.11) define the change in inventory each time a vehicle visits the depot facility. Inequality (2.12) ensures that that the product quantities (at the depots and those carried by the vehicle) cannot be negative at any point along the route. Constraint (2.13) states that the arc decision variables are binary. Finally, Constraints (2.14) and (2.15) specify the range of the decision variables related to the quantities loaded onto the vehicle from the depots.

The above model can be simplified to describe problems UU and LU, as described below:

Problem UU

In this case the two depots have unlimited product availability ($\Pi^1, \Pi^2 \to \infty$) and hence, Constraints (2.8) to (2.12) are no longer needed. Note also that the vehicle will be fully loaded upon each visit to a depot facility. Hence,

$$d_{D_i^1}, d_{D_i^2} = Q - q_{i-1} \quad ,i = 0,1, \dots, n \quad (2.16)$$

Furthermore, visiting two depot facilities in sequence is not appropriate due to a) the fact that the necessary product quantity can always be loaded by any (single) depot facility, and b) the triangular inequality.

In this case, therefore, the set of arcs can be redefined as follows:

$$A' = \{(i, i+1), (i, D_i^1), (i, D_i^2)\} : i \in V \cup \{0\}\}.$$

Obviously,

$$A' \subseteq A.$$

Consequently, the mathematical model for problem UU can be written as follows:

$$minTC = \sum_{i=0}^{n-1} \left(x_{i,i+1} c_{i,i+1} + x_{i,D_i^1} c_{i,D_i^1} + x_{D_i^1,i+1} c_{D_i^1,i+1} + x_{i,D_i^2} c_{i,D_i^2} + x_{D_i^2,i+1} c_{D_i^2,i+1} \right) + c_{n,n+1} \quad (2.17)$$

Subject to:

$$x_{i,i+1} + x_{i,D_i^1} + x_{i,D_i^2} = 1 \quad , i = 0, \dots, n \quad (2.18)$$

$$x_{i,D_i^1} = x_{D_i^1,i+1} \quad , i = 0, \dots, n \quad (2.19)$$

$$x_{i,D_i^2} = x_{D_i^2,i+1} \quad , i = 0, \dots, n \quad (2.20)$$

$$q_i = q_{i-1} - d_i + (x_{D_i^1} + x_{D_i^2}) \times (Q - q_{i-1}) \quad , i = 0, \dots, n \quad (2.21)$$

$$q_i + d_i \leq Q \quad , i = 0, \dots, n \quad (2.22)$$

$$q_i \geq 0 \quad , i = 0, \dots, n \quad (2.23)$$

$$x_{i,i+1}, x_{i,D_i^1}, x_{i,D_i^2}, x_{D_i^1,i+1}, x_{D_i^2,i+1} \in \{0,1\} \quad , i = 0, \dots, n \quad (2.24)$$

Problem LU

In this case we assume that the second depot (D^2) is the one with the finite product inventory. This increases the complexity compared to the UU case. Again, all decision variables associated with D^1 can be excluded from the model. However, the decision variables associated with quantity $d_{D_i^2}$that can be loaded from D^2 to the vehicle is retained. Similarly to the UU case the vehicle will never visit sequentially the two depots, thus the arc set for this case also is A'.

$$minTC = \sum_{i=0}^{n-1} \left(x_{i,i+1} c_{i,i+1} + x_{i,D_i^1} c_{i,D_i^1} + x_{D_i^1,i+1} c_{D_i^1,i+1} + x_{i,D_i^2} c_{i,D_i^2} + x_{D_i^2,i+1} c_{D_i^2,i+1} \right) + c_{n,n+1} \quad (2.25)$$

Subject to:

$$x_{i,i+1} + x_{i,D_i^1} + x_{i,D_i^2} = 1 \qquad ,i = 0, \dots, n \qquad (2.26)$$

$$x_{i,D_i^1} = x_{D_i^1,i+1} \qquad ,i = 0, \dots, n \qquad (2.27)$$

$$x_{i,D_i^2} = x_{D_i^2,i+1} \qquad ,i = 0, \dots, n \qquad (2.28)$$

$$q_i = q_{i-1} - d_i + \left(x_{D_i^1} \times (Q - q_{i-1})\right) + \left(x_{D_i^2} \times d_{D_i^2}\right) \qquad ,i = 0, \dots, n \qquad (2.29)$$

$$q_i + d_i \leq Q \qquad ,i = 0, \dots, n \qquad (2.30)$$

$$P_0^2 = \Pi^2 - x_{D_0^2} \times d_{D_0^2} \qquad (2.31)$$

$$P_i^2 = P_{i-1}^2 - x_{D_i^2} \times d_{D_i^2} \qquad ,i = 1, \dots, n \qquad (2.32)$$

$$P_i^2, q_i \geq 0 \qquad ,i = 0, \dots, n \qquad (2.33)$$

$$x_{i,i+1}, x_{i,D_i^1}, x_{D_i^1,i}, x_{i,D_i^2}, x_{D_i^2,i} \in \{0,1\} \qquad ,i = 0, \dots, n \qquad (2.34)$$

Chapter 3. Solution Methods

In this Chapter, we propose two alternative methods to solve the above problems: a) A Dynamic Programming algorithm (DP), which provides the optimal solution in all three problems. b) A Labeling Algorithm (LA), which solves them in more efficient computational times.

3.1. Problem UU

Dynamic Programming (DP) algorithm

Let's assume that the minimum distance $C_i(q_i)$ from client i until the ending point $(n+1)$ is a function of the load q_i of the vehicle as it departs from client i. Then the DP Equations are formulated as follows:

If $i = n$, then:

$$C_i(q_i) = c_{n,n+1} \qquad , q_i = 0, \dots, Q \qquad (3.1)$$

For clients $i = n-1, n-2, \dots, 1$ and for vehicle load $q_i = 0, \dots, Q$:

$$C_i(q_i) = min \begin{cases} c_{i,i+1} + C_{i+1}(q_i - d_{i+1}) \\ c_{i,D_i^1} + c_{D_i^1,i+1} + C_{i+1}(Q - d_{i+1}) \\ c_{i,D_i^2} + c_{D_i^2,i+1} + C_{i+1}(Q - d_{i+1}) \end{cases} \qquad , q_i \geq d_{i+1} \qquad (3.2)\ (a)$$

$$C_i(q_i) = min \begin{cases} c_{i,D_i^1} + c_{D_i^1,i+1} + C_{i+1}(Q - d_{i+1}) \\ c_{i,D_i^2} + c_{D_i^2,i+1} + C_{i+1}(Q - d_{i+1}) \end{cases} \qquad , q_i < d_{i+1} \qquad (3.2)\ (b)$$

For the starting position $i = 0$, and $q_0 = 0$:

$$C_0(q_0) = min \begin{cases} c_{0,D_0^1} + c_{D_0^1,1} + C_1(Q - d_1) \\ c_{0,D_0^2} + c_{D_0^2,1} + C_1(Q - d_1) \end{cases} \qquad (3.3)$$

Equation (3.2a) ensures that if the vehicle load is adequate to satisfy the demand of the next client, then $C_i(q_i)$ will be the minimum distance considering all three possible paths the vehicle may follow. In Eq. (3.2b), the path that represents the direct visit from client i to $i+1$ is excluded due to Eq. (2.23). The minimum distance $minTC = C_0(q_0)$ is provided by Eq. (3.3).

Labeling Algorithm (LA)

The labeling algorithm creates partial paths iteratively; that is, paths that begin from the starting position and reach each client according to the predefined client sequence. Each partial path incident to a client i can be extended to the next client $i+1$ by: a) direct visit to the next client, b) visit through D_i^1, or c) visit through D_i^2. The algorithm extends all partial paths reaching each client, determines the *quality* of these paths, and discards those of inferior *quality*. The paths of inferior *quality* are identified by applying the, so called, dominance rules. The procedure continues until the ending positionn $+1$. The complete path (from the starting to the ending position) with the minimum total cost is selected as the optimal one.The labeling algorithm is an extension of the classic Dijkstra's algorithm for the directed single-source shortest path problem (Dijkstra, 1959). It has been applied to a multitude of problems, such as the VRP with time windows (Desrochers*et al.*, 1992), in the airline crew pairing Lavoie *et al.* (1988) and in flight crew scheduling Graves *et al.* (1993). More recent works for the Elementary Shortest Path Problem with Resource Constraints (ESPPRC) include Feillet *et al.* (2004) and Chabrier (2005); focus on the importance of the dominance rules.

Let $K = (0,1,2,\dots,n+1)$ be the predefined sequence of serving the clients. Also let Z_i be the path set that contains all partial paths reaching client (node) $i \in K$. Each partial path $z \in Z_i$is characterized by:

- $\bar{C}_z$: the total distance from the starting position (0) to client i, the last node of partial path z
- $\bar{q}_z$: the load of the vehicle after serving client i, the last node of partial path z.

The steps of the algorithm are as follows:

Step 1: Start from first node of the sequence $(i = 0)$ with the initial partial path $z \in Z_0$. where:

$$\bar{C}_z = 0, \bar{q}_z = 0, z \in Z_0$$

Step 2: Set $i = i + 1$

Step 3: [*Partial path extension*]: For each partial path $z, z \in Z_{i-1}$, connect node (client) $i-1$ to i, by the three possible ways and create, if feasible, three new partial paths $z'_1, z'_2, z'_3 \in Z_i$. Overall, $3 \times |Z_{i-1}|$ new paths will be created. Figure 3.1

shows an example for extending a partial path in Z_2. In case a partial path z exists for which the load of the vehicle is adequate to satisfy all clients until the end of the client sequence, i.e. $\bar{q}_z \geq \sum_{y=i}^{n+1} d_y$, then only partial path z'_1 is constructed by visiting node i directly, since any visit to a depot would unnecessarily increase the distance travelled by the vehicle.

Step 4: [*Computation of the path variables*]: For each new created partial path $z' \in Z_i$ the following are computed, based on the corresponding variables of the parent partial path $z \in Z_{i-1}$(note that the index k=1, 2, 3 of z'_k has been dropped for convenience)

i. The total distance:

$$\bar{C}_{z'} = \bar{C}_z + x_{i-1,i}c_{i-1,i} + x_{i-1,D^1_{i-1}}\left(c_{i-1,D^1_{i-1}} + c_{D^1_{i-1},i}\right) + x_{i-1,D^2_{i-1}}\left(c_{i-1,D^2_{i-1}} + c_{D^2_{i-1},i}\right)$$

ii. The load $\bar{q}_{z'}$of the vehicle after serving client i. This amount is calculated due to the visits of the vehicle to a depot as:

$$\bar{q}_{z'} = \begin{cases} \bar{q}_z - d_i \quad , x_{i-1,D^1_{i-1}} + x_{i-1,D^2_{i-1}} = 0 & \text{(3.4) (a)} \\ \mathrm{Q} - d_i \quad , x_{i-1,D^1_{i-1}} + x_{i-1,D^2_{i-1}} = 1 & \text{(3.4) (b)} \end{cases}$$

Equation (3.4a) states that during the direct visit to client i the vehicle load is decreased by exactly the amount corresponding to the demand of client i. Equation (3.4b) states that when the vehicle visits a depot (D^1or D^2), then it is loaded up to its capacity.

Step 5: [*Discarding of partial paths*]: If any of the partial paths in set Z_i is infeasible due to constraints (2.19) to (2.25), then it is discarded (pruned). For the remaining paths, we identify and discard those of inferior quality. For this, consider two paths $z'_1, z'_2 \in Z_i$. Then partial path z'_1 dominates partial path z'_2, and z'_2 can be discarded if the following hold (simultaneously):

i. $\bar{q}_{z'_1} \geq \bar{q}_{z'_2}$ (3.5)

ii. $\bar{C}_{z'_1} \leq \bar{C}_{z'_2}$ (3.6)

If Inequality (3.5) is satisfied, and if the vehicle traverses partial path z'_1 after serving client i, it may return to a depot fewer -or at most the same- number of times for replenishment, and, thus, may travel a shorter (or at most equal) distance from

client i to the end of the route. If Eq. (3.6) is satisfied as well, then at least one complete path created from partial path z'_1 will correspond to a lower-or at most equal- distance as compared to *all* complete paths created from partial path z'_2. Thus partial path z'_2 can be discarded safely.

Step 6: Steps 2, 3,4 and 5 are repeated until $i = n + 1$

Step 7: From the set of pathsZ_{n+1}, the one with the minimum distance is the optimal route to satisfy all clients with $minTC = \min_{z \in Z_{n+1}} \bar{C}_z$.

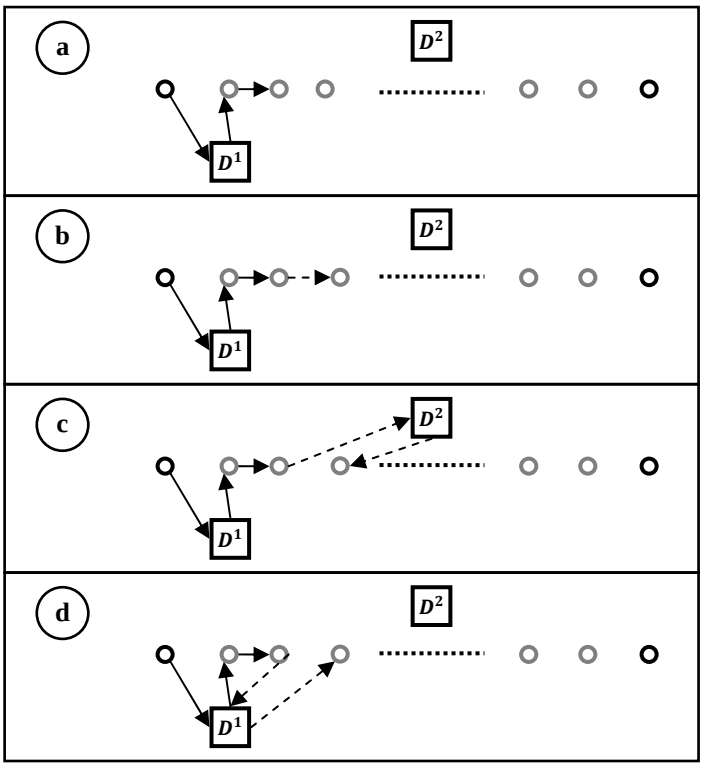

Figure 3.1: Partial path expansion example: (a) The initial partial path. (b), (c), (d) The partial path extensions

3.2. Problem LU

Dynamic Programming (DP) algorithm

The DP developed for this case starts from the last client ($i = n$), and for each client i under consideration it defines the minimum cost $C_i(q, P_i^2)$ from this client to the ending position (n + 1) of the sequence. We assume that this cost depends on i) the vehicle's load $q_i$and ii) the inventory $P_i^2$of depot D^2 (the depot with limited stock) after serving client i.

If $i = n$, then:

$$C_i(q_i, P_i^2) = c_{n,n+1} \qquad , \forall\, q_i = 0, \dots, Q, \forall P_i^2 = 0, \dots, \Pi^2 \qquad (3.7)$$

For $i = n, n-1, .., 1, q_i = 0,1, \dots, Q, P_i^2 = 0,1, \dots, \Pi_2$

$$C_i(q_i, P_i^2) = min \begin{cases} c_{i,i+1} + C_{i+1}(q_i - d_{i+1}, P_i^2) \\ c_{i,D_i^1} + c_{D_i^1,i+1} + C_{i+1}(Q - d_{i+1}, P_i^2) \\ c_{i,D_i^2} + c_{D_i^2,i+1} + \min\limits_{0 \le d_{D_i^2} \le \min\{P_i^2, Q - q_i\}} \{C_n\left(q_i + d_{D_i^2} - d_{i+1}, P_i^2 - d_{D_i^2}\right)\} \end{cases} \qquad , q_i \ge d_{i+1} \qquad (3.8)\ (a)$$

$$C_i(q_i, P_i^2) = min \begin{cases} c_{i,D_i^1} + c_{D_i^1,i+1} + C_n(Q - d_{i+1}, P_i^2) \\ c_{i,D_i^2} + c_{D_i^2,i+1} + \min\limits_{q_i + d_{D_i^2} - d_{i+1} \le d_{D_i^2} \le \min\{P_i^2, Q - q_i\}} \{C_n\left(q_i + d_{D_i^2} - d_{i+1}, P_i^2 - d_{D_i^2}\right)\} \end{cases} \qquad \begin{matrix} , q_i < d_{i+1}, \\ q_i + P_i^2 \ge d_{i+1}, \end{matrix} \qquad (3.8)\ (b)$$

$$C_i(q_i, P_i^2) = c_{i,D_i^1} + c_{D_i^1,i+1} + C_n(Q - d_{i+1}, P_i^2) \qquad \begin{matrix} , q_i < d_{i+1}, \\ q_i + P_i^2 < d_{i+1}, \end{matrix} \qquad (3.8)\ (c)$$

And for the starting position $i = 0, P_0^2 = \Pi^2$ and $q_0 = 0$:

$$C_0(0, P_0^2) = min \begin{cases} c_{0,D_0^1} + c_{D_0^1,1} + C_1(Q - d_1, P_0^2) \\ c_{0,D_0^2} + c_{D_0^2,1} + \min\limits_{d_{D_0^2} = 0,.., \min\{\Pi^2, Q\}} \{C_1\left(d_{D_0^2} - d_1, P_0^2 - d_{D_0^2}\right)\} \end{cases} \qquad (3.9)$$

Equation (3.8a) ensures that if the vehicle's load is adequate to satisfy the demand of the next client, then the minimum cost $C_i(q_i, P_i^2)$will be the minimum distance of the three possible paths in this Equation. In Eq. (3.8b) the path that represents the direct transition from client i to $i+1$ is excluded due to Eq. (2.33). Equation (3.8c) includes only the case in which the vehicle visits D^1 to satisfy the demand of client $i+1$, since there is not enough combined stock on board the vehicle and in D^2. The optimal distance from the starting position is the lowest distance to serve all clients, $minTC = C_0(q_0, P_0^2)$.

Labeling Algorithm (LA)

The LA for the UL problem is an extension of the LA for the UU problem. As in the previous case, let Z_i be the set of all partial paths reaching client (node) $i \in K$. As before, each partial path $z \in Z_i$ is characterized by $\bar{C}_z$, and $\bar{q}_z$; additionally it is also characterized by the inventory level of the depot with limited inventory, that is:

- $\bar{P}_z^2$: the inventory in D^2 after serving the last client of partial path z

At each visit toD^2, the appropriate product quantity $d_{D_i^2}$ should be loaded onto the vehicle (note that $d_{D_i^2}$ is a decision variable of problem LU). Note that this quantity is not known at the time of

the visit to D^2. This difficulty is handled in DP by examining all possible values of the load quantity$d_{D_i^2}$ in order to obtain the optimal solution. In LA, in order to reduce the number of paths, we first consider that the maximum feasible quantity of products is loaded onto the vehicle when it visits the D^2.As the path(s) extend downstream we update this quantity as well as the inventory of D^2 by returning any unused quantity back to it. Each time after the visit to D^2, the vehicle visits D^1(obviously after serving some intermediate clients), no extra load is needed on the vehicle upon the visit to D^1, since it can load up to capacity in D^1. Thus, any extra load remaining should be returned to D^2 by updating its inventory.

In addition, we use the following ancillary variable in the dominance criteria:

- H_z^2: the maximum possible inventory in D^2 after serving the last client of partial path z

This quantity is updated at each expansion of a partial path to the next client by computing the exact required quantity that should be loaded onto the vehicle during its last visit to D^2 to satisfy all clients until the last one served.

The steps of the algorithm are similar to the steps of LA for problem UU. The differences are summarized as follows:

1. In step 1, quantity H_z^2 is set to: $H_z^2 = \Pi^2, z \in Z_0$
2. In step 4, $\bar{P}_z^2$ and H_z^2 are updated as discussed above:

    i. The remaining inventory $\bar{P}_{z'}^2$ in D^2after serving client $i$is computed according to the previous visits of the vehicle to the depots. Consider the last visit to $D^2$for stock replenishment just prior to client i. Let j be the client after which this visit occurred. Also let $p_j$be the remaining inventory in D^2 after serving client j by path $z \in Z_{i-1}$. Then, $\bar{P}_{z'}^2$ ($z' \in Z_i$ is the extension of z) is calculated as follows:

$$\bar{P}_{z'}^2 = \begin{cases} \bar{P}_z^2 - \min\{\bar{P}_z^2, Q - \bar{q}_z\} & , x_{i-1,D_{i-1}^2} = 1 \quad (3.10)\text{ (a)} \\ \bar{P}_z^2 & , x_{i-1,D_{i-1}^1} + x_{i-1,D_{i-1}^2} = 0 \quad (3.10)\text{ (b)} \\ \min\{p_j, \bar{P}_z^2 + \bar{q}_z\} & , x_{j,D_j^2} \times x_{i-1,D_{i-1}^1} \times \left(1 - \min\left\{1, \sum_{m=j+1}^{i-2}\left(x_{m,D_m^1} + x_{m,D_m^2}\right)\right\}\right) = 1 \quad (3.10)\text{ (c)} \\ \bar{P}_z^2 & , x_{j,D_j^1} \times x_{i-1,D_{i-1}^1} \times \left(1 - \min\left\{1, \sum_{m=j+1}^{i-2}\left(x_{m,D_m^1} + x_{m,D_m^2}\right)\right\}\right) = 1 \quad (3.10)\text{ (d)} \end{cases}$$

Equation (3.10a) provides the remaining inventory in D^2 in the case in which the vehicle reached client i from $i-1$ through D^2. In this case, the maximum possible quantity is loaded onto the vehicle. Equation (3.10b) states that if the vehicle does not visit any depot between clients $i-1$ and i, there is no change in the inventory of D^2. In Eq.(3.10c),p the inventory level of D^2 is updatedby adding the quantity of products on board ($\bar{q}_z$), while ensuring that the updated inventory level of D^2 will not exceed the inventory level before the last visit to it, that is p_j.Equation (3.10d) states that between two sequential visits to D^1 there is no change to the inventory of D^2.

ii. The maximum inventory level in D^2after serving client i by partial path $z'(z' \in Z_i$and is the extension of $z \in Z_{i-1})$ is calculated as follows:

$$\bar{H}_{z'} = \begin{cases} \min\{p_j, \bar{P}_z^2 + \bar{q}_z\} & , x_{j,D_j^2} \times \left(1 - \min\left\{1, \sum_{m=j+1}^{i-1}\left(x_{m,D_m^1} + x_{m,D_m^2}\right)\right\}\right) = 1 \quad (3.11)\text{ a)} \\ \bar{H}_z^2 & , x_{j,D_j^1} \times \left(1 - \min\left\{1, \sum_{m=j+1}^{i-1}\left(x_{m,D_m^1} + x_{m,D_m^2}\right)\right\}\right) = 1 \quad (3.11)\text{ (b)} \end{cases}$$

In Eq. (3.11a) the maximum inventory level is defined as in Equation (3.10c), by adding (returning) the remaining quantity of products on-board, but not exceeding the previous inventory level (p_j) ofD^2.Equation (3.11b) states that between two sequential visits to D^1 there is no change to the maximum inventory level of D^2.

3. In step 4, $\bar{q}_z'$is updated as follows:

$$\bar{q}_{z'} = \begin{cases} \bar{q}_z + \min\{\bar{P}_z^2, Q - \bar{q}_z\} & , x_{i-1,D_{i-1}^2} = 1 \quad (3.12)\text{(a)} \\ \bar{q}_z - d_i & , x_{i-1,D_{i-1}^1} + x_{i-1,D_{i-1}^2} = 0 \quad (3.12)\text{(b)} \\ \mathrm{Q} - d_i & , x_{i-1,D_{i-1}^1} = 1 \quad (3.12)\text{(c)} \end{cases}$$

Equation 3.12(a) states that when the vehicle visits D^2, then it loads the maximum feasible quantity of products. Equation (3.12b) states that during the direct visit to client i the vehicle load is decreased by exactly the amount corresponding to the demand of client i. Equation (3.12a) states that when the vehicle visits D^1, then it is loaded up to its capacity.

4. In step 6, consider the two paths $z_1, z_2 \in Z_i$. Then partial path z_1 dominates partial path z_2 (and, thus, z_2 can be discarded) if the following holds additionally to Eqs. (3.5) and (3.6):

$$\bar{H}_{z_1} \geq \bar{H}_{z_2} \quad (3.13)$$

If Eq. (3.13) is satisfied, then in case of the partial path $z_1$the inventory level in D^2,is equal or higher, and, thus, the vehicle may return more (or the same number of) times to D^2,for replenishment (instead of D^1), if appropriate in order to minimize the total distance travelled. As a result, partial path $z_2$can be discarded without further extending it.

3.3. Problem LL

This is the most complex among the three VRDRP variants studied in this thesis. The DP algorithm developed for this case initiates from the last client ($i = n$); for each client i under consideration it defines the minimum cost $C_i(q, P_i^1, P_i^2)$from this client to the ending position (n + 1) of the sequence. We assume that $C_i$this cost depends on i) the vehicle's load q_i, ii) the inventory$P_i^1$of D^1, and iii) the inventory$P_i^2$of D^2.

Dynamic Programming (DP) algorithm

For $i = n, q_i = 0,1, \dots, Q, P_i^1 = 0,1, \dots, \Pi_1, P_i^2 = 0,1, \dots, \Pi_2$:

$$C_i(q_i, P_i^1, P_i^2) = c_{i,n+1} \qquad , q_i = 0, \dots, Q, P_i^1 = 0, \dots, \Pi^1, P_i^2 = 0, \dots, \Pi^2 \tag{3.16}$$

For $i = n-1..,1, q_i = 0,1, \dots, Q, P_i^1 = 0,1, \dots, \Pi_1, P_i^2 = 0,1, \dots, \Pi_2$

$$C_i(q_i, P_i^1, P_i^2) = \min \begin{cases} c_{i,i+1} + C_{i+1}(q_i - d_{i+1}, P_i^1, P_i^2) \\ c_{i,D_1} + c_{D_1,i+1} + \min\limits_{0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\}} C_{i+1}\left(q_i + d_{D_i^1} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2\right) \\ c_{i,D_2} + c_{D_2,i+1} + \min\limits_{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\}} C_{i+1}\left(q_i + d_{D_i^2} - d_{i+1}, P_i^1, P_i^2 - d_{D_i^2}\right) \\ \min \begin{cases} c_{i,D_1} + c_{D_1,D_2} + c_{D_2,i+1} \\ c_{i,D_2} + c_{D_2,D_1} + c_{D_1,i+1} \end{cases} + \min\limits_{\substack{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\} \\ 0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\} \\ 0 \le d_{D_i^1} + d_{D_i^2} \le Q - q_i}} C_{i+1}\left(q_i + d_{D_i^1} + d_{D_i^2} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2 - d_{D_i^2}\right) \end{cases} \qquad , q_i \ge d_{i+1} \tag{3.17 (a)}$$

$$C_i(q_i, P_i^1, P_i^2) = \min \begin{cases} c_{i,D_1} + c_{D_1,D_2} + c_{D_2,i+1} \\ c_{i,D_2} + c_{D_2,D_1} + c_{D_1,i+1} \end{cases} + \min\limits_{\substack{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\} \\ 0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\} \\ d_{i+1} - q_i \le d_{D_i^1} + d_{D_i^2} \le Q - q_i}} C_{i+1}\left(q_i + d_{D_i^1} + d_{D_i^2} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2 - d_{D_i^2}\right) \qquad \begin{matrix} , q_i < d_{i+1}, \\ q_i + P_i^1 < d_{i+1}, \\ q_i + P_i^2 < d_{i+1} \end{matrix} \tag{3.17 (b)}$$

$$C_i(q_i, P_i^1, P_i^2)$$

$$= \min \begin{cases} c_{i,D_1} + c_{D_1,i+1} + \min\limits_{0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\}} C_{i+1}\left(q_i + d_{D_i^1} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2\right) \\ c_{i,D_2} + c_{D_2,i+1} + \min\limits_{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\}} C_{i+1}\left(q_i + d_{D_i^2} - d_{i+1}, P_i^1, P_i^2 - d_{D_i^2}\right) \\ \min \begin{cases} c_{i,D_1} + c_{D_1,D_2} + c_{D_2,i+1} \\ c_{i,D_2} + c_{D_2,D_1} + c_{D_1,i+1} \end{cases} + \min\limits_{\substack{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\} \\ 0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\} \\ d_{i+1} - q_i \le d_{D_i^1} + d_{D_i^2} \le Q - q_i}} C_{i+1}\left(q_i + d_{D_i^1} + d_{D_i^2} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2 - d_{D_i^2}\right) \end{cases} \qquad \begin{matrix} , q_i < d_{i+1}, \\ q_i + P_i^1 \ge d_{i+1}, \\ q_i + P_i^2 \ge d_{i+1} \end{matrix} \tag{3.17 (c)}$$

$$C_i(q_i, P_i^1, P_i^2)$$

$$= \min \begin{cases} c_{i,i+1} + C_{i+1}(q_i - d_{i+1}, P_i^1, P_i^2) \\ c_{i,D_1} + c_{D_1,i+1} + \min\limits_{d_{i+1} - q_i \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\}} C_{i+1}\left(q_i + d_{D_i^1} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2\right) \\ \min \begin{cases} c_{i,D_1} + c_{D_1,D_2} + c_{D_2,i+1} \\ c_{i,D_2} + c_{D_2,D_1} + c_{D_1,i+1} \end{cases} + \min\limits_{\substack{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\} \\ 0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\} \\ d_{i+1} - q_i \le d_{D_i^1} + d_{D_i^2} \le Q - q_i}} C_{i+1}\left(q_i + d_{D_i^1} + d_{D_i^2} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2 - d_{D_i^2}\right) \end{cases} \qquad \begin{matrix} , q_i < d_{i+1}, \\ q_i + P_i^1 \ge d_{i+1}, \\ q_i + P_i^2 < d_{i+1} \end{matrix} \tag{3.17 (d)}$$

$$C_i(q_i, P_i^1, P_i^2)$$

$$= \min \begin{cases} c_{i,i+1} + C_{i+1}(q_i - d_{i+1}, P_i^1, P_i^2) \\ c_{i,D_2} + c_{D_2,i+1} + \min\limits_{d_{i+1} - q_i \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\}} C_{i+1}\left(q_i + d_{D_i^2} - d_{i+1}, P_i^1, P_i^2 - d_{D_i^2}\right) \\ \min \begin{cases} c_{i,D_1} + c_{D_1,D_2} + c_{D_2,i+1} \\ c_{i,D_2} + c_{D_2,D_1} + c_{D_1,i+1} \end{cases} + \min\limits_{\substack{0 \le d_{D_i^2} \le \min\{P_i^2, Q-q_i\} \\ 0 \le d_{D_i^1} \le \min\{P_i^1, Q-q_i\} \\ d_{i+1} - q_i \le d_{D_i^1} + d_{D_i^2} \le Q - q_i}} C_{i+1}\left(q_i + d_{D_i^1} + d_{D_i^2} - d_{i+1}, P_i^1 - d_{D_i^1}, P_i^2 - d_{D_i^2}\right) \end{cases} \qquad \begin{matrix} , q_i < d_{i+1}, \\ q_i + P_i^1 < d_{i+1}, \\ q_i + P_i^2 \ge d_{i+1} \end{matrix} \tag{3.17 (e)}$$

And for the staring position $i = 0$, $P_0^1 = \Pi^1$, $P_0^2 = \Pi^2$ and $q_0 = 0$:

$$C_0(0,P_0^1,P_0^2) = min\begin{cases} c_{0,D_1} + c_{D_1,1} + \min\limits_{0\le d_{D_i^1}\le min\{P_i^1,Q\}} C_1\left(d_{D_0^1}-d_1, P_0^1 - d_{D_0^1}, P_0^2\right) \\ c_{0,D_2} + c_{D_2,1} + \min\limits_{0\le d_{D_0^2}\le min\{P_0^2,Q\}} C_1\left(d_{D_0^2}-d_1, P_0^1, P_0^2 - d_{D_0^2}\right) \\ min\begin{cases} c_{0,D_1} + c_{D_1,D_2} + c_{D_2,1} \\ c_{0,D_2} + c_{D_2,D_1} + c_{D_1,1} \end{cases} + \min\limits_{\substack{0\le d_{D_0^2}\le min\{P_0^2,Q\} \\ 0\le d_{D_0^1}\le min\{P_0^1,Q\} \\ 0\le d_{D_0^1}+d_{D_0^2}\le Q}} C_1\left(d_{D_0^1}+d_{D_0^2}-d_1, P_0^1 - d_{D_0^1}, P_0^2 - d_{D_0^2}\right) \end{cases} \quad (3.18)$$

Equation (3.17a) ensures that if the load of the vehicle is adequate to satisfy the demand of the next client, then $C_i(q_i, P_i^1, P_i^2)$ will be the minimum distance among the five possible paths connecting clients $i-1$ and i. Equation (3.17b) refers to the case in which the vehicle visits both depots sequentially in order to satisfy the demand of client $i+1$, since there is not enough load or stock in D^1 or D^2. In Eq. (3.17c) the path that represents the direct transition from client i to $i+1$ is excluded due to Eq. (2.12). In Eqs. (3.17d) and (3.17e), if the vehicle load plus the inventory available at one depot is not adequate to satisfy the demand of the next client $i+1$, the path that involves only that depot is not considered. The optimal distance from the starting position is the lowest distance to serve all clients, $minTC = C_0(q_0, P_0^1, P_0^2)$.

Labeling Algorithm (LA)

The Labeling Algorithm for the LL problem is again an extension of the algorithm developed for the UU problem. In this case, though, LA does not guarantee optimality. As in the previous cases, let Z_i be the set of all partial paths reaching client (node) $i \in K$. Each partial path $z \in Z_i$ is characterized by $\bar{C}_z$, and $\bar{q}_z$ and, since in this case both depots have limited inventory, $\bar{P}_z^1, \bar{P}_z^2$ are the inventory levels of D^1 and D^2, respectively, after serving the last client of partial path z. In this case, the variables H_z^i for the maximum dept inventory level are not used nor defined.

The differences from the Labeling Algorithms of the previous cases are summarized as follows:

1. In step 3 each partial path incident to a client i can be extended to the next client $i+1$ by: a) Direct trip to the next client, b) trip through D_i^1, c) trip through D_i^2, and d) trip through both depots. In this case as well, the appropriate quantity to be loaded onto the vehicle during a depot visit is not known at the time of extending the path. Thus, in all depot visit cases, the vehicle (initially) receives the maximum possible load (up to its capacity, or up to the available inventory).

2. A significant variation is necessary for cases (b) and (c) if a different depot was visited prior to the current visit; i.e. D_i^2 for (b) and D_i^1 for (c). In such an event, two path extensions are created, differing only by the inventory level of the depots. In the first extension, it is considered that the maximum possible quantity was loaded onto the vehicle during its previous visit to the depot. In the second extension, it is considered that the vehicle was loaded just to satisfy the demand of the clients between the previous and the current visit; during the current visit the maximum possible quantity is assumed to be loaded onto the vehicle. In both extensions during the current visit the vehicle receives the maximum possible load. This variation is necessary because of the importance of the inventory level in depots. By creating the two extensions differing only by the inventory level of the depots, as described, it is ensured that in both depots (in one depot per extension) the maximum inventory will be available, in case of a future visit. Either case may result in the lower distance travelled by the vehicle in the future.

3. Case (d), is considered only if there is not adequate quantity of products in one depot to load the vehicle up to its capacity; in that case it is necessary to create one more extension, in which the vehicle also visits the other depot before serving the next client. Again, this case may eventually lead to a minimum cost route. Overall, at most $5 \times |Z_{i-1}|$ new paths will be created.
4. In step 4, $P_z^1, P_z^2, q_{z\prime}$ are updated as discussed above and according to Eqs. (3.10a)-(3.10.d) and (3.12a)-(3.12c), respectively.
5. In step 6, the dominance rules of the LA for problem UU are applied, by relaxing the second criteria Eq. (3.6) to Eq. (3.19).

$$\bar{C}_{z\prime_1} < \bar{C}_{z\prime_2} \tag{3.19}$$

Equation (3.19) secures that the two extensions created, as described in 1 above, will not be discarded immediately. Note that these dominance criteria are stringent, and, therefore, some good quality paths may be discarded. As a consequence, the optimal solution of the problem is not guaranteed. Figure 3.2 displays an example for extending a partial path for the LL problem. The extensions shown in Figure 3.2 (b and c) are those that are extended twice.

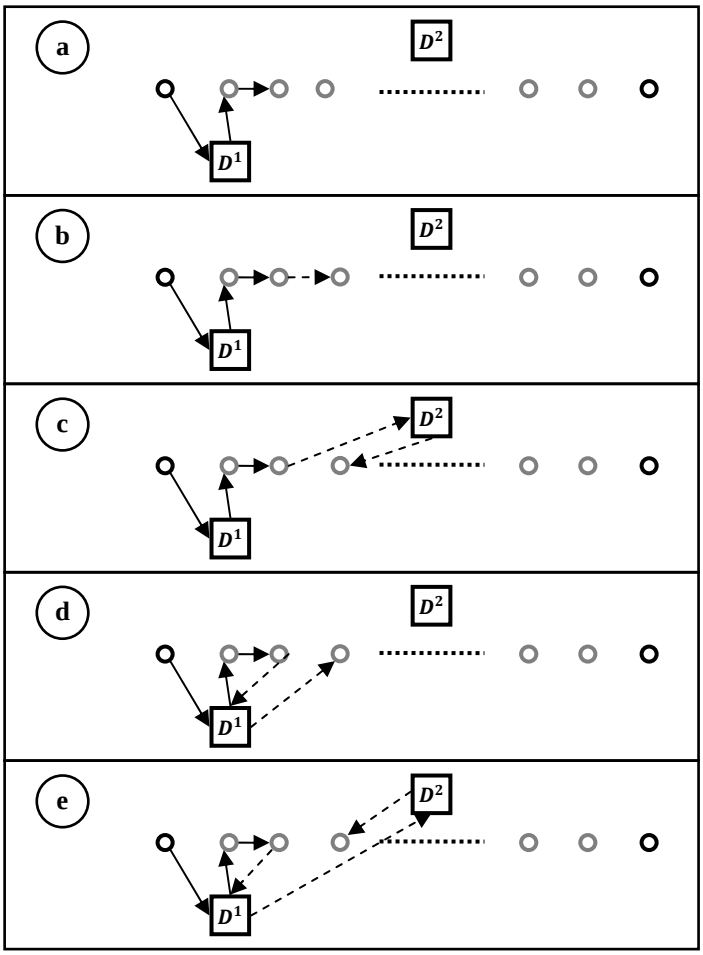

Figure 3.2: Partial path expansion example: (a) The initial partial path. (b), (c), (d), (e) The partial path extensions

Chapter 4. Computational Analysis

A wide range of experiments were conducted in order to evaluate the efficiency of the methods developed to solve the above three VRDRP variants. All methods were implemented in Mathworks' Matlab R2010a and run on a PC equipped with an Intel Core i7 - Q720 CPU, and 4 GB of RAM.

A random problem generator was used to uniformly distribute $n + 4$ points in a 24×24 square space. The first n points represent the locations of the clients to be served; the next two points represent the starting and ending positions of the vehicle, while the last two points represent the locations of the two depots. The capacity of the vehicle is set to Q. The demand of all clients is equal to $m = Q/3$ product units. It has been assumed that the availability of the depots is either unlimited or equal to half the cumulative demand of all clients ($\Pi = {}^{n \times m}/_{2}$). In all problems the vehicle departs empty from the starting position ($q_0 = 0$).

For all experiments (UU, LU, and LL), the following procedure is applied for various values of$n$ and Q:

Step 1: The number of clients is set to n
Step 2: The vehicle capacity is set to Q(and thus the client demand to $m = \frac{Q}{3}$)
Step 3: Ten(10) problem instances are generated for each pair (n, Q)
Step 4: The 10 problem instances are solved using both DP and LA
Step 5: Log computational time and total cost.
This procedure is repeated for increasing values of $n$and for different values of Q.

4.1. Problem UU

As mentioned in Section 3.1 both the DP and LA algorithms solve the UU problem to optimality. In terms of computational efficiency, Figure 4.1 displays the average computational time of DP over the 10 problem instances for the number of clients n, increasing from 10 to 500 by a step of 10. Each line corresponds to a different value of $Q = 12, 24, 48, 96, 192$. It is clear that the computational time of DP increases as the number of clients and the capacity of the vehicle increase. This is expected, since DP computes the optimal distance to the destination point for all possible loads, and, thus, the upper bound of the DP complexity in this case is $O(n \times Q) \cong O(n^2)$.

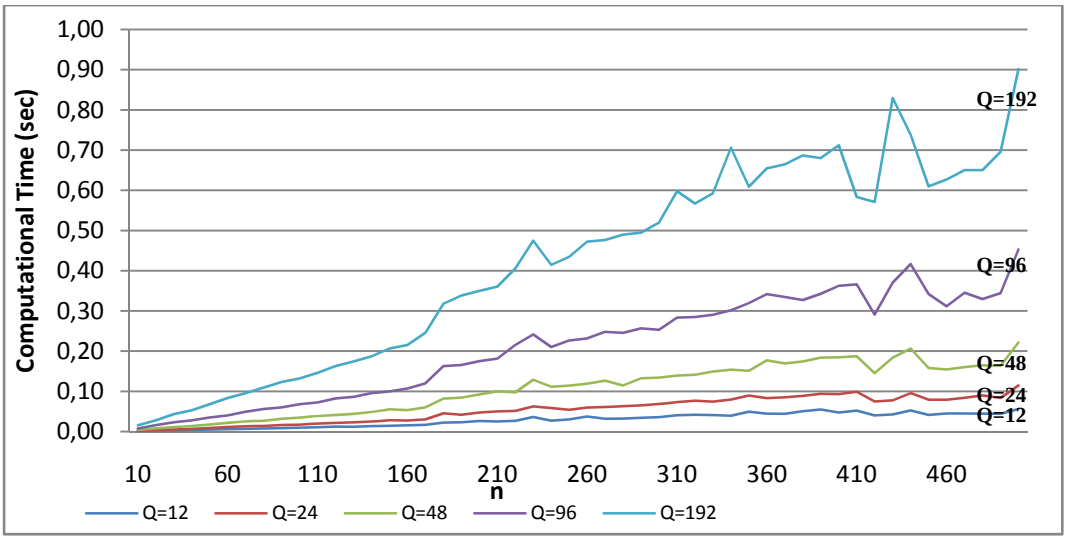

Figure 4.1: Problem UU - Computational time as a function of the number of clients n for five values of the capacity Q

Figure 4.2 displays the average computational time required to solve the same problem instances as in Fig. 4.1 by the Labeling Algorithm. Again, the computational time increases with the number of clients n, butit appears to be independent of Q, since in this case paths that correspond to unfavorable loads are pruned early in the algorithm. Furthermore, since the ratio $\frac{Q}{m}$ is constant the same number of partial paths are expected to be created by the algorithm.

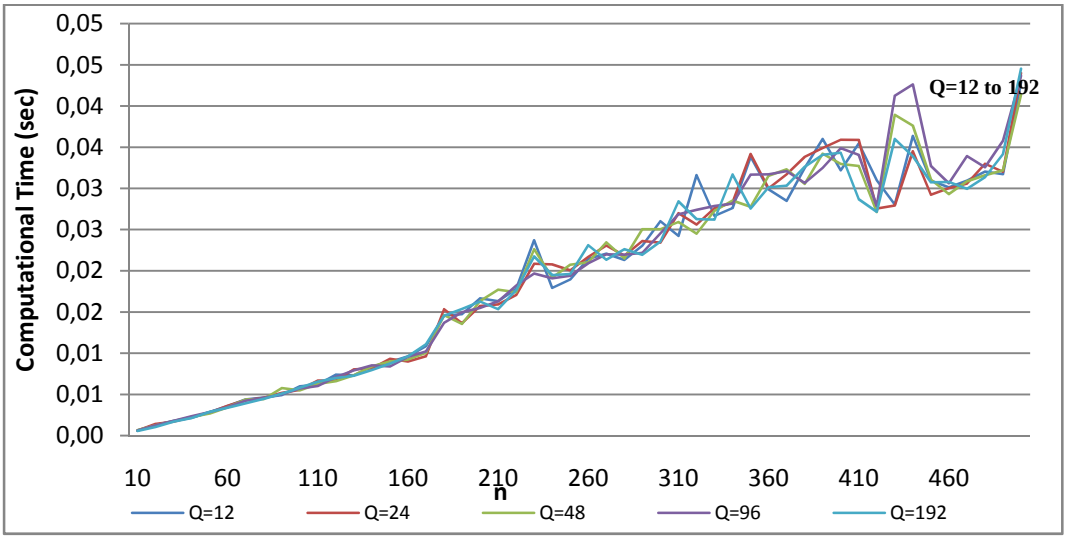

Figure 4.2: Computational time of the labeling algorithm for problem UU

Figure 4.3 presents the ratio r of the average computational times of DP over LA. The Figure validates that LA is more efficient than DP; for example, if $Q = 96$ or $Q = 192$, LA is more than one order of magnitude faster than DP.

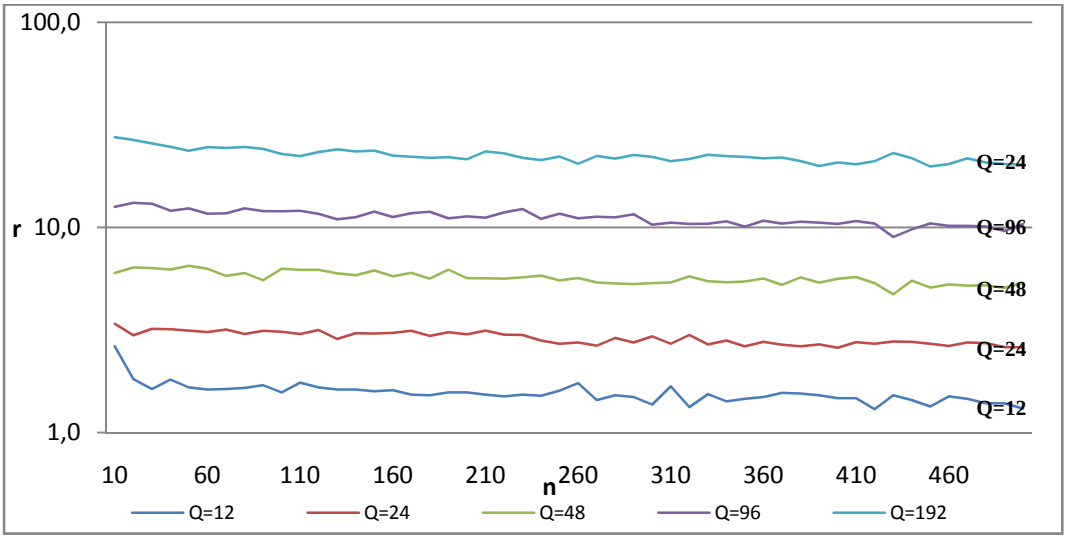

Figure 4.3: Problem UU: The ratio of the average computational times of DP over LA

4.2. Problem LU

In this case as well, both the DP and LA algorithms solve the LU problem to optimality. In terms of computational efficiency, Figure 4.4 shows the average computational time of DP over the 10 problem instances, for the number of clients n, increasing from 10 to 200 by a step of 10. Each line corresponds to a different value of Q= 12, 24, 48. The computational time of DP increases with: a) the number of clients, b) the capacity of the vehicle, c) the inventory of Depot 2, and d) the quantity of products loaded onto the vehicle while it visits Depot 2. Hence, the upper bound of the DP complexity is $O(n \times Q \times P_i^2 \times d_i^2) \sim O(n^4)$

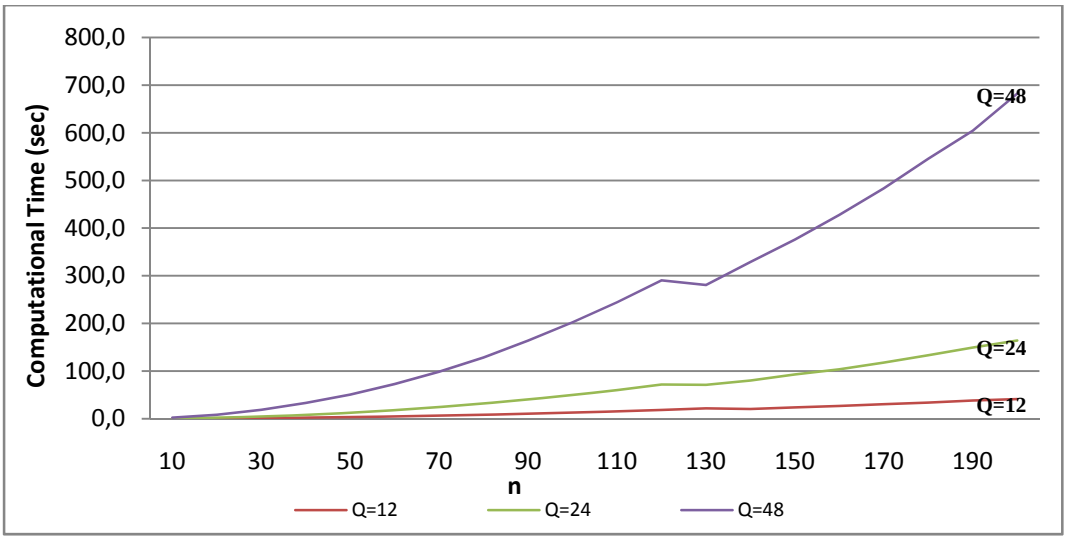

Figure 4.4: Computational time of the dynamic algorithm for problem LU

Figure 4.5 shows the average computational time over the ten instances for LA, and for the same problem instances as in Fig. 4.4. Again, the computational time increases with the number of clients n but, in this case, it appears to be independent of Q. This is because at most three new partial paths are created for any partial path reaching a customer regardless the amount that can be loaded onto the vehicle. Furthermore, many of these path extensions are pruned due to the dominance criteria.

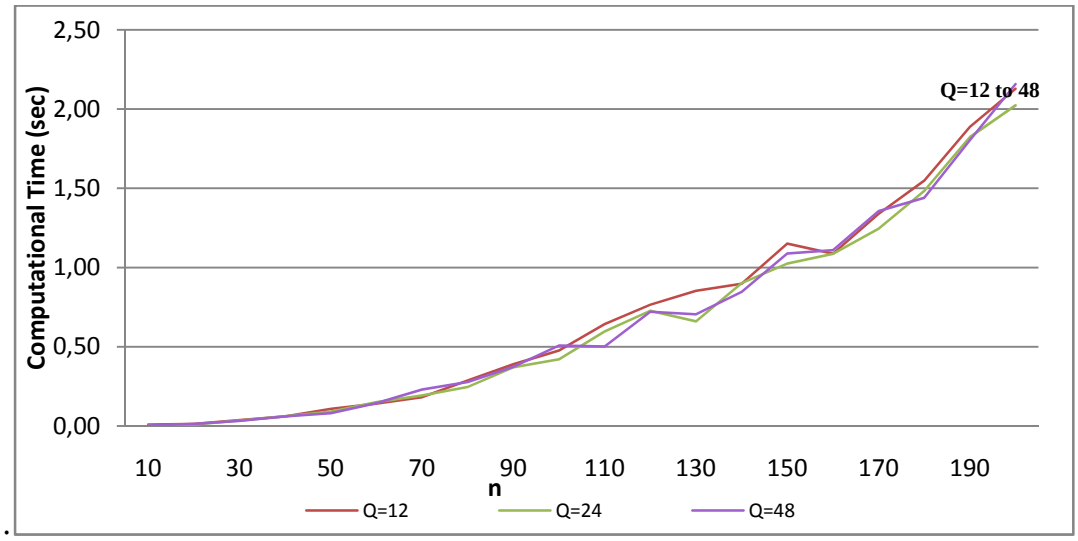

Figure 4.5: Computational time of the labeling algorithm for problem LU

Figure 4.6 presents the ratio r of the average computational times of DP over LA. In this case LA is much more efficient than DP; for instance, in the cases$Q = 24$ and $Q = 48$, LA is more than two orders of magnitude faster than DP.

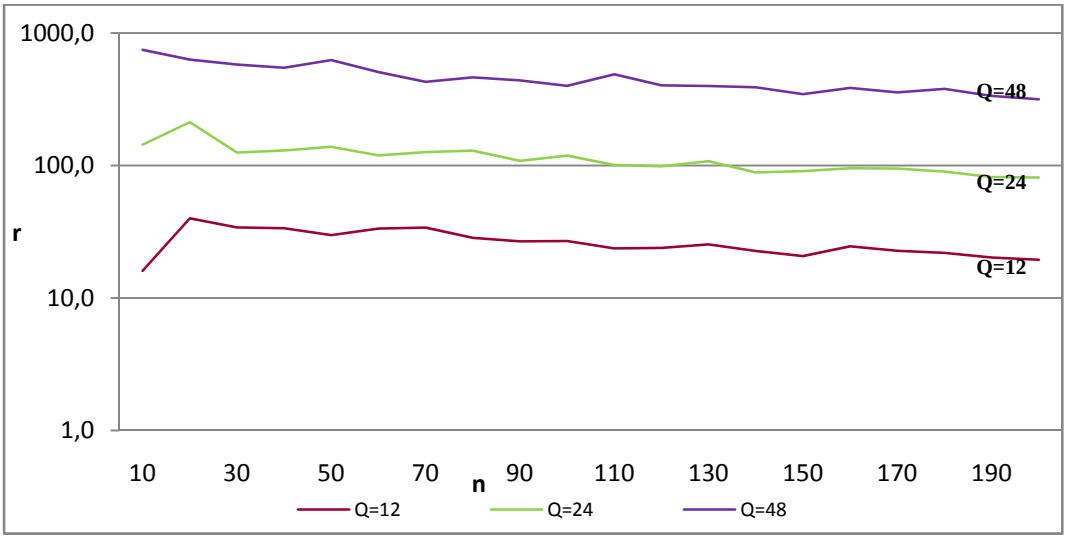

Figure 4.6: The ratio of the average computational times of DP over LA

To further examine the performance of LA in solving the LU problem, we applied in this case the stronger dominance criteria of the UU problem – Eqs. (3.5) and (3.6).These criteria do not preserve optimality, but lead to enhanced pruning. Regarding the quality of the solutions obtained by applying the sub-optimal dominance criteria, in 21% of the cases, the optimal solution has been found, while in all other cases the average deviation from the optimal solution was found to be 1.27%. Figure 4.7 shows the ratio ρ of the computational times of LA when the (optimal) dominance criteria of LU are used vs. the (suboptimal in this case) dominance criteria of UU. The results indicate that the more stringent dominance criteria can transform LA to a fast and effective heuristic for the LU problem.

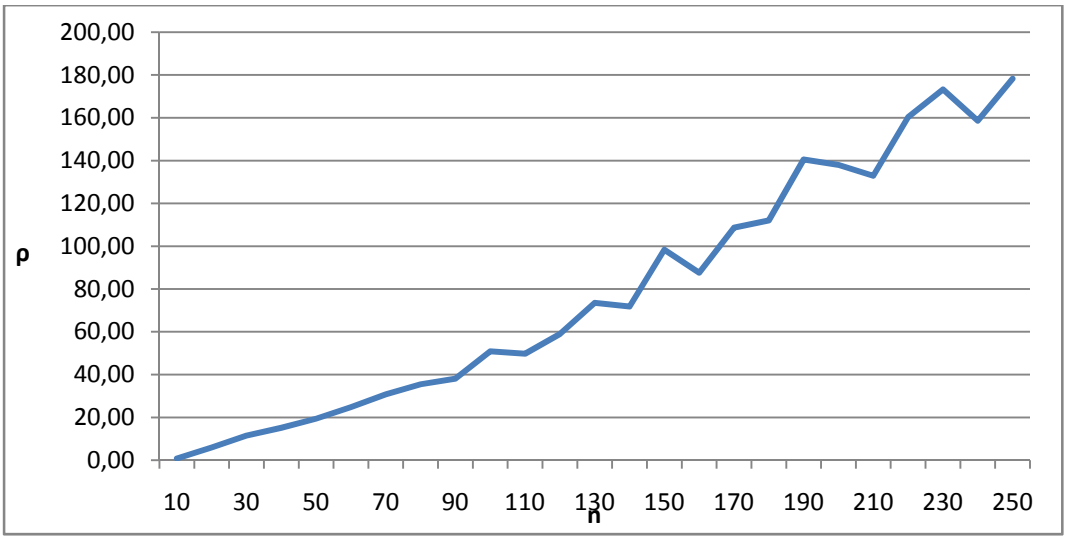

Figure 4.7: Problem UL: Average computational time of LA applying optimal vs. suboptimal dominance criteria

4.3. Problem LL

In this much more complex problem, the DP algorithm reaches the optimal solution, while the LA provides close to optimal solutions in a much more efficient manner. Note that the complexity of DP is even higher ($O(n^6)$), since the computational time increases with: a) the number of clients, b) the capacity of the vehicle, c) the inventory in Depot 1, d) the inventory in Depot 2, e) the quantity of products loaded onto the vehicle while it visits Depot 1, and f) the quantity of products loaded onto the vehicle while it visits Depot 2. Hence, the upper bound of the DP complexity is$O(n \times Q \times P_i^1 \times d_i^1 P_i^2 \times d_i^2) \sim O(n^6)$.

The experimental investigation for LL used the following test parameters: $n \in [10, 100]$and $Q = 6$.In terms of the quality of the solutions obtained, the optimal solution is found in 18% of the cases while in all other cases the average deviation from the optimal solution was found to be 1.85%.In terms of computational complexity, Figure 4.9 shows the average computational times of DP and LA over the 10 problem instances, as well as their ratio r, with increasing number of clients n. From these test results, it is clear that LA is much more efficient that DP; the ratio r(DP vs. LA) is almost steady over the range of n, and LA is almost three orders of magnitude faster than DP.

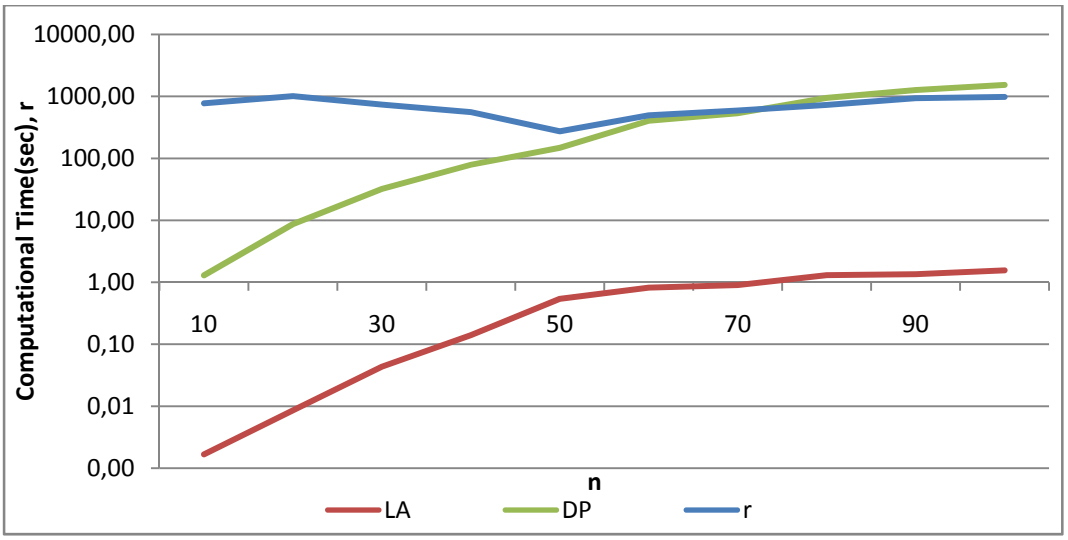

Figure 4.8: Computational times of dynamic of the DP and LA algorithms for problem LL, and their ratio *r*

Chapter 5. Conclusions

In this thesis we investigated the single vehicle routing problem with a predefined customer sequence and two replenishment depots. Three problems were modeled, solved and analyzed, depending on the inventory levels of the two depots. For each problem, two solution approaches were developed: A Dynamic Programming (DP) algorithm and a Labeling algorithm (LA).Both were evaluated through a large number of randomly generated problem instances. DP solves all three problems to optimality, while LA solves the first two (unlimited inventory in both depots -UU, and limited inventory in one depot unlimited in the other -LU), while it provides suboptimal, but good, solutions for the third (limited inventory in both depots -LL).In terms of computational time, the LA algorithm proved to be significantly more efficient than DP computational time for all cases; that is,1-3 orders of magnitude faster in problems UU to LL, respectively.

An interesting topic of future research is to study the efficiency of the LA algorithm to problems with stochastic client demand. Furthermore, since the studied problem is a partitioning problem, the proposed algorithm can be applied to other partitioning problems outside the field of vehicle routing, such as the database allocation problem (see Wang and Jea, 2009), in which the optimal data partition must be determined to minimize the access time.

References

1. Bianchi, L., Birattari, M., Chiarandini, M., Manfrin, M., Mastrolilli, M., Paquete, L., et al. (2006). Hybrid metaheuristics for the vehicle routing problem with stochastic demands. *Journal of Mathematical Modelling and Algorithms , 5*, 91-110.

2. Clarke, G., & Wright, J. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research, 12* (4), 568-581.

3. Desrochers, M., Desrosiers, J., & Solomon, M. (1992). A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research, 40*, 342-354.

4. Dijkstra, W. E. (1959). A note on two problems in Connexion with Graphs. *Numerische Mathematik, 1*, 269-271.

5. Feillet, D., Dejax, P., Gendreau, M., & Gueguen, C. (2004). An Exact Algorithm for the Elementary Shortest Path Problem with Resource Constraints: Application to Some Vehicle Routing Problems. *Networks, 44*, 216-229.

6. Fisher, L., Jormsten, K., & Madsen, O. (1997). Vehicle Routing with Time Windows: Two Optimization Algorithms. *Operations Research, 45* (3), 488-492.

7. Gendreau, M., Hertz, A., & Laport, G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science, 40* (10), 1276-1290.

8. Golden, B., Raghavan, S., and Wasil, E. (2008). The vehicle routing problem: latest advances and new challenges. Operations research/Computer science interfaces series, 43. Springer

9. Graves, G., McBride, R., Gershkoff, I., & Anderson, D. (1993). Flight crew scheduling. *Management Science, 39*, 657-682.

10. Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, 671-680.

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

12. Laporte, G., Nobert, Y., & Desrochers, M. (1985). Optimal routing under capacity and distance restrictions. *Operations Research, 33*, 1050-1073.

13. Lavoie, S., Minoux, M., & Odier, E. (1988). A new approach for crew pairing problems by column generation with an application to air transportation. *European Journal of Operation Research, 35*, 45-58.

14. Li, J.-Q., Mirchandani, P., & Borenstein, D. (2009a). A Lagrangian heuristic for the real-time vehicle rescheduling problem. *Transportation Research Part E: Logistics and Transportation Review, 45* (3), 419-433.

15. Li, J.-Q., Mirchandanib, P. B., & Borenstein, D. (2009b). Real-time vehicle rerouting problems with time windows. *European Journal of Operational Research, 194* (3), 711-727.

16. Osman, I. (1993). Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problems. *Annals of Operations Research, 41*, 421-452.

17. Rego, C., & Roucairol, C. (1996). A parallel tabu search algorithm using ejection chains for the vehicle routeing problem. *Meta-Heuristics: Theory and Applications*, 661-675.

18. Reinmann, M., & Doerner, K. (2004). D-Ants: Savings Based Ants divide and conquer the vehicle routing problem. *Computers & Operations Research*, 563-591.

19. Tatarakis, A., & Minis, I. (2009). Stohastic single vehicle routing with a predefined customer sequence and multi depot returns. *European Journal of Operation Research* (197), 557-571.

20. Toth, P., & Vigo, D. (2002). *The vehicle routing problem.* Philadelphia: SIAM.

21. Tsirimpas, P., Tatarakis, A., Minis, I., & Kyriakidis, E. (2008). Single vehicle routing with a predifined customer sequence and multiple depot returns. *European Journal of Operation Research* (187), 483-495.

22. Wang, J. Y., & Jea, K. F. (2009). A near-optimal database allocation for reducing the average waiting time in the grid computing environment. *Information Sciences, 179*, 3772-3790.

23. Yang, W., Mathur, K., & Ballou, R. (2000). Stochastic Vehicle Routing Problem with Restocking. *Transportation Science* (34), 99-112.