136746

# Integrated Vehicle Routing And Warehouse Location Problem

by

Arda Gezdur

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Industrial Engineering

Koç University

July, 2003

Koç Üniversitesi
Fen Bilimleri Enstitüsü

Yüksek Lisans Tez Sınavı Tutanağı

31-07-2003

Fen Bilimleri Enstitüsü yüksek lisans öğrencilerinden *20010657* numaralı *Arda Gezdur'un* sözlü tez jüri sınavı 31-07-2003 tarihinde yapılmış ve adı geçen öğrencinin "*Integrated Warehouse Location and Vehicle Routing Problem*" başlıklı yüksek lisans tezi başarılı bulunarak jüri tarafından kabul edilmiştir.

Tez jüri üyeleri

| | |
|---|---|
| Assistant Prof. Dr. Can Akkan | ........[illegible]........ |
| Prof. Dr. İ. Kuban Altınel | ........[illegible]........ |
| Assistant Prof. Dr. Fikri Karaesmen | ........[illegible]........ |
| Assistant Prof. Dr. Selçuk Savaş | ........[illegible]........ |
| Assistant Prof. Dr. Metin Türkay | ........[illegible]........ |

*To my mother and father*

# ABSTRACT

This thesis proposes an exact algorithm for the integrated vehicle routing and warehouse location problem, known as WLRP. WLRP helps the supply chain managers to make strategic level decisions, and may be used in a supply chain environment at different industrial sectors. Previous work on WLRP emphasized heuristic and metaheuristic solution approaches that do not guarantee optimality. In search of optimality, other related and well-studied problems are investigated and it is concluded that there is a close relationship between the vehicle routing problem with time windows (VRPTW) and WLRP. Therefore, first the VRPTW is studied extensively as a foundation to WLRP. Recent studies on VRPTW call attention to column generation methods together with the branch and bound algorithm. The column generation algorithm with dynamic programming is used and in this thesis a new dynamic programming formulation is presented. Furthermore, for heterogeneous fleet problems, a new variant of vehicle routing problem (VRP) is formulated which is named as the vehicle routing problem with time windows and discrete capacities (VRPTWDC); and the dynamic program is modified for this problem. The WLRP is formulated as a set partitioning problem and column generation technique is applied. The dynamic program is extensively modified to handle the problem more effectively, the bounds are tightened using 2-path cuts and the subtours are eliminated using a separation algorithm. Finally, branch and bound method is applied to solve WLRP optimally. All benchmark problems in WLRP literature are solved and it is illustrated that the proposed algorithm yields better solutions compared to algorithms reported in the literature.

# ACKNOWLEDGEMENTS

I would like to thank Asst. Prof. Metin Türkay and Asst. Prof. Selçuk Savaş, who have been my advisors throughout my graduate study, for all their help and guidance. I would like to especially express my sincere gratitude to Metin Türkay for making my attendance to INFORMS San Jose Conference possible and letting me to gain an important experience in an international organization. I would also like to thank Prof. Uğur Akman from my undergraduate university, for introducing me to linear programming and optimization methods and helping me with my senior year project. I would also like to thank Prof. Öner Hortaçsu from Boğaziçi University who taught us what engineering is all about.

I am also very grateful to Emrah Nikerel, Ömer Topçu and my other friends for their loyal participations to the social events to keep my mind fresh. I would also like to thank my colleague and dear friend Esen Mestan, for taking her time to listen and solve my problems since high school years. I also thank to Seden Mestan for her delicious coffee breaks during our long hours of studies.

Finally I would like to thank Ayşe Gezdur and Selçuk Gezdur for helping, motivating and directing me throughout my graduate study and my entire life.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# NOMENCLATURE

| | |
|---|---|
| WLRP | Warehouse Location Routing Problem |
| VRPTW | Vehicle Routing Problem with Time Windows |
| VRPTWDC | Vehicle Routing Problem with Time Windows and Discrete Capacities |
| MWLRP | Modified Warehouse Location Routing Problem |
| MDVDP | Multi Depot Vehicle Dispatch Problem |
| WLAP | Warehouse Location Allocation Problem |
| MDRAP | Multi Depot Routing Allocation Problem |
| LAP | Location Allocation Problem |
| VRP | Vehicle Routing Problem |
| SA | Simulated Annealing |
| TS | Tabu Search |
| GA | Genetic Algorithm |
| PFIH | Push Forward Insertion Heuristic |
| MINLP | Mixed Integer Non Linear Programming |
| SPPTW | Shortest Path Problem with Time Windows |
| ESPPTW | Elementary Shortest Path Problem with Time Windows |
| TSPTW | Traveling Salesman Problem with Time Windows |
| SPP | Shortest Path Problem |
| TSP | Traveling Salesman Problem |

# Chapter 1

# INTRODUCTION

The definition of logistics management according to The Council of Logistics Management is: "The process of planning, implementing and controlling the efficient and cost-effective flow and storage of raw-materials, in-process inventory finished goods and related information from point of origin to point of consumption for the purpose of conforming to customer requirements"[1]. While flow of material and information between the suppliers and manufacturing unit forms the Inbound Logistics Function, the same flow between the customers and the manufacturing units forms the Outbound Logistics Function. The flow of materials and information within the manufacturing unit constitutes the Internal Logistics Function. The Outbound Logistics functions can further be classified according to the level of decisions. Among these levels, routing of raw materials or products and location of production or inventory holding facilities both correspond to the strategic level decisions. Detailed information on the logistics management issues is given in Ghosh and Raychaudhuri[2].

The logistics operations show highly dynamic behavior due to changing demand patterns that require the optimization of the location of the facilities and routes the customers are served. The warehouse location problem alone is not sufficient for strategic level decisions in supply chain; because the customers are assigned on a straight-and-back basis that is each customer is served directly by a vehicle from the warehouse. This assumption in the warehouse location problem neglects routing and therefore even the optimal solutions to this problem can be improved by routing. On the other hand, the classical vehicle routing problem assumes that the sites of the warehouses are known. For this reason, even if the minimum cost is achieved, a better solution can still be achieved by changing the locations of the warehouses. These important drawbacks of addressing the warehouse location and vehicle routing problems separately force the researchers and practitioners to study the integration of these two problems effectively in order to achieve better solutions.

## 1.1 Literature Survey

Early studies on the warehouse location and vehicle routing problem (WLRP) include the work by Perl[3] and Perl and Daskin[4]. WLRP is first formulated in [3,4] as a mixed integer linear program with capacity and maximum route distance constraints for a three level supply chain where there are the suppliers, warehouses, and customers. The WLRP is modified to include only the warehouses and customers by eliminating the suppliers from the supply chain in (MWLRP)[4]. The MWLRP contains subtour elimination constraints to remove cycles that are added for every possible combination of customers. The prohibitively large number of subtour elimination constraints, even for small-scale problems, makes the MWLRP impossible to be solved within acceptable computing times. Thus, Perl and Daskin[4] developed a heuristic method to solve MWLRP by decomposing the problem into three phases: multi-depot vehicle dispatch problem (MDVDP), warehouse location-allocation problem (WLAP), and multi-depot routing-allocation problem (MDRAP). This method solves each problem iteratively and generates good solutions; however, it does not guarantee optimality. Hansen et al.[5] studied the WLRP and proposed a new formulation to handle the subtour elimination constraints by employing continuous variables. Although this formulation can be solved optimally, the lower bound for the problem is so low that only small problems can be solved in reasonable time. Hansen et al.[5] modified the heuristic method of Perl and Daskin and improved solutions to benchmark problems significantly. A very recent contribution to WLRP is done by Wu et al.[6]. In this paper, a different heuristic method was proposed in which WLRP is decomposed into a location-allocation problem (LAP) and a vehicle routing problem (VRP). New search methods are selected for the LAPs and VRPs using simulated annealing (SA).

The complexity of the problem has often forced the researchers to use approximation methods through repetitive problem solving. One of the earliest studies is done by Burness and White[7], who defined the traveling salesman location problem to locate a single new facility. Or and Pierskalla[8], Jacobsen and Madsen[9] proposed new formulations and algorithms using some side constraints such as capacity limit and maximum cost/tour-length restriction. Laporte and Nobert[10] proposed an integer programming problem to minimize routing and operating costs and solved the problem by relaxing the constraints. Nambiar et al.[11] and Nambiar et al.[12] studied the problem of improving the collection, processing and marketing of the Malaysian rubber industry and solved this location routing problem with heuristics. Laporte et al.[13] studied the Hamiltonian cycles and the subtour constraints for the routing-location problem. Madsen[14] and Min et al.[15] made extensive literature survey on the solution of the routing-location problems. Drezner et al.[16] concentrated on locating facilities using the rectilinear distances. Bookbinder and Reece[17] formulated a multi-commodity, capacitated distribution planning model as a non-linear mixed integer program and solved the problem using a generalized assignment type problem based on Benders decomposition. Laporte et al.[18] imposed a branch and bound algorithm on the multi-depot vehicle routing and location-routing problems after modifying the problems with graph representation and graph extension. Laporte et al.[19] and Chan et al.[20] worked on stochastic programming of the location-routing problems by decomposing the problems into location-allocation type problems. Srivastava and Benton[21] worked on the physical distribution system design of location-routing problems and investigated the effects of the ratio of location to routing costs. The location routing model of hazardous materials was addressed by Revelle et al.[22]. Srivastava[23] proposed three new location-routing problems and solved these problems by savings based heuristics. Tuzun and Burke[24] applied the tabu search method for the location-routing problems. The applications of the routing location problem included flow-interception problems, location-routing problems with uncertainty, environmental impacts of simultaneous location and routing, and logistics of hazardous materials. Detailed review of these applications together with analysis of the warehouse location problem can be found in Drezner[25].

Efficient routing and scheduling of vehicles can result in important savings for government agencies and industries at different sectors. The vehicle routing problem with time windows (VRPTW) arises in a wide range of practical decision-making problems, such as school bus routing, mail delivery, newspaper delivery, fuel oil delivery, and municipal waste collection.

VRPTW has become one of the most popular problems in logistics for two decades. Tan et al.[26] made a literature survey on VRPTW. In this work, the use of customer interchange method to improve solutions using local search algorithm as well as hybrid simulated annealing (SA) and tabu search (TS) is investigated. Furthermore, a hybrid genetic algorithm (GA) and local search method is also proposed. For all these heuristics, the initial solution of VRPTW is obtained from push forward insertion heuristic (PFIH). This heuristic is a straightforward and constructive algorithm to obtain initial solutions for all the problems presented in Solomon[27]. Previous work of Hax and Candea[28] involves routing a fleet of vehicles, with limited capacities, from a central depot to a set of geographically dispersed customers with known demands and predefined time window constraints. The route cost of a vehicle is the total distance it traveled, and the objective is to minimize the total cost of all routes with minimum number of vehicles without violating any constraints. A detailed information on the metaheuristics for the vehicle routing problem with time windows can be found in Gendreau et al.[29].

Survey on classifications and applications of VRPTW can be found in Osman[30], Laporte[31] and Bodin et al.[32]. Heuristic methods often produce near optimal solutions in a reasonable amount of computational time. Many researchers currently working on new heuristic methods for solving the VRPTW include Kontoravdis and Bard[33], Kohl and Madsen[34], Thompson and Psaraftis[35] or Kolen et al.[36]. Kolen et al.[36] presented the a heuristic branch and bound method, which is among the first optimization algorithms for VRPTW. The method calculates lower bounds using dynamic programming and state space relaxation. Branching decisions are taken on route-customer allocations. Lau et al.[37] developed a heuristic method by defining an upper bound on the number of vehicles to be used to solve the VRPTWDC. In this study, an upper bound on the number of vehicles is employed and heuristic solution methods to solve also heterogeneous fleet VRPTW are used.

The optimal solution to the VRPTW is investigated by various researchers, which frequently employed decomposition methods. Kohl and Madsen[34] used Lagrangean relaxation, Desrochers, Desrosiers and Solomon[38] investigated the Dantzig-Wolfe decomposition and Halse[39] has studied the variable splitting.

Most recent studies in the VRPTW include the work of Kohl, Desrosiers, Madsen, Solomon and Soumis[40]. In this work, several decomposition algorithms are reviewed and Dantzig–Wolfe decomposition algorithm is used. Valid inequalities are inserted using the 2-path cut techniques together with the decomposition algorithm. The proposed MINLP formulation finds the subsets of customers that violate $k$-path inequalities using the results of the relaxed problems tightening the bounds and reducing the solution time. Two different methods are also presented to find the subsets those have been proposed; the first is a heuristic, which is proved to find all the subsets whereas the other is an optimal separation algorithm. Another recent study is by Rich[41] that used 2-path cuts as well as 3-path cuts. Furthermore, a set partitioning model is formulated and an effective separation routine for *k*-path cuts is presented. Detailed information on the column generation approach to the VRPTW can be found in Larsen[42]. In this work, column generation approach is used by defining the master problem as a set partitioning problem and the subproblem as a shortest path problem with time windows. In the subproblem, feasible routes with negative reduced costs are generated using dynamic programming following the basic principles of Dijkstra's algorithm. A similar method with slight variations in the dynamic programming algorithm is used in this thesis. One of the most recent contributions to the VRPTW include the work by Irnich et al.[43], the Dantzig-Wolfe decomposition algorithm was used and several methods to improve the dynamic programming algorithm was investigated. The work Bazaraa et al.[44] contains detailed information and introduction to the Dantzig-Wolfe decomposition. The work of Cordeau et al[45] is useful in learning more on the application of the column generation algorithm on VRPTW. Furthermore, detailed information on the theory of valid inequalities can be found in Nemhauser and Wolsey[46].

## *1.2 Outline of the Thesis*

The aim of the thesis is to solve the vehicle routing and warehouse location problem WLRP. In order to solve the WLRP effectively, a good routing optimization algorithm is needed. Therefore, the vehicle routing problem with time windows is also addressed. The contributions to the VRPTW are as follows:

i. A new mixed integer formulation is presented with significantly reducing the number of variables and number of constraints. Previous formulations of the VRPTW model the flow constraints using three indices. In this thesis, the flow constraints are modeled using two different variables each with two indices that reduce the number of integer variables significantly. Furthermore, constraint aggregation is employed to the constraint sets which link two or more different types of integer variables.

ii. Three preprocessing methods are applied to tighten the time windows and eliminate several arcs prior to optimization.

iii. Column generation algorithm is applied to the VRPTW and the dynamic programming algorithm for the subproblem is changed. For each label in the dynamic program, information on the previous customer nodes is recorded as a vector. This recording process provides the elimination of k-cycles and therefore, increases the lower bound of the master problem and decreases the number of branch and bound nodes.

iv. The optimality gap is reduced due to the elimination of *k*-cycles, in many benchmark problems.

The VRPTW assumes that each vehicle has the same capacity. However, in many real problems heterogeneous fleet of vehicles is present. In order to solve heterogeneous fleet problems a new problem is formulated; the vehicle routing problem with time windows and discrete vehicle capacities (VRPTWDC). The contributions to the VRPTWDC are as follows:

i. A new two-index integer formulation is presented. This formulation reduces the number of constraints and variables as in the case of VRPTW.

ii. The new column generation algorithm for the VRPTW is modified for the VRPTWDC that includes fixed capacities of the vehicles in the route costs. Furthermore, at each iteration of the column generation, the dynamic problem is solved for each vehicle capacity and the results for each capacity are combined prior to the next iteration. The dynamic program eliminates *k*-cycles as in the case of VRPTW.

iii. The separation algorithm that is used for the VRPTW is modified for the VRPTWDC and is applied for each vehicle capacity.

iv. The branching priorities are set as in the VRPTW.

After obtaining efficient optimization algorithms for both homogenous and heterogeneous fleet routing problems, the WLRP is studied. The contributions to the WLRP are as follows:

i. A new two-index formulation is presented. Previous formulations modeled both the flow variables and the load variables using three indices. The flow variables are replaced with two two-index variables as in VRPTW and the load variables are replaced with one-index variables by adding several constraints. Although, several constraints are added the number of variables constraints are also reduced. The number of constraints is further reduced by constraint aggregation.

ii. Column generation algorithm is applied. The master problem is formulated as a set partitioning problem with side constraints and allocation variables for the warehouses. A new dynamic programming algorithm is proposed which uses different dual variables from the master problem at each run. The proposed dynamic program is run for each warehouse and vehicle type and it eliminates the *k*-cycles.

iii. The separation algorithm for the 2-path and subtour cuts is modified for the WLRP. The algorithm is run for each vehicle type.

iv. The branching priorities are set in order to reduce the number of nodes in the branch and bound tree.

v. Optimal solutions for benchmark problems are found.

In this thesis, the VRPTW and its slight modification, vehicle routing problem with time windows and discrete capacities (VRPTWDC) are considered as the first steps to generate exact methods to solve the WLRP. In Chapter 2, two-index formulations for the VRPTW and VRPTWDC are introduced. Furthermore, the sequential algorithm, namely, iterative solution of column generation, 2-path cuts generation, and branch and bound algorithms are reconsidered and a new dynamic programming algorithm for the column generation is presented in Chapter 2. A two index model for the WLRP is formulated using disjunctive programming and propositional logic in Chapter 3. The sequential algorithm for VRPTW is modified for the WLRP and presented in Chapter 4. The results of benchmark problems and an illustrative example are presented in Chapter 5. Computational complexity of the proposed algorithm is discussed, the conclusions are summarized and the possible future directions on WLRP are elaborated in Chapter 6.

Chapter 2

# VEHICLE ROUTING WITH TIME WINDOWS AND DISCRETE VEHICLE CAPACITIES

## *2.1 Problem Definition*

The vehicle routing problem with time windows (VRPTW) is a special case of the vehicle routing problem with time windows and discrete vehicle capacities (VRPTWDC). When only a single vehicle capacity is considered VRPTWDC is reduced to VRPTW with a single warehouse and multiple customers. The locations (or the Euclidean distances between) of the customers and the warehouse are given. The orders given by the customers include quantities and time limitations such as the service times (loading and unloading), the latest and earliest arrival times. Furthermore, the capacities for all vehicle types are given.

There are several assumptions in the routing. These can be listed as follows:

i. A customer can only be allocated on a single route.
ii. A vehicle cannot be used for two different routes.
iii. The velocity of the trucks is constant at 60 km/hr.
iv. There are no capacity constraints for the warehouse.
v. The fleet contains infinite number of vehicles.
vi. Early and late times are defined on the basis of work hours

The problem is defined on a graph as in Fig. 2.1:

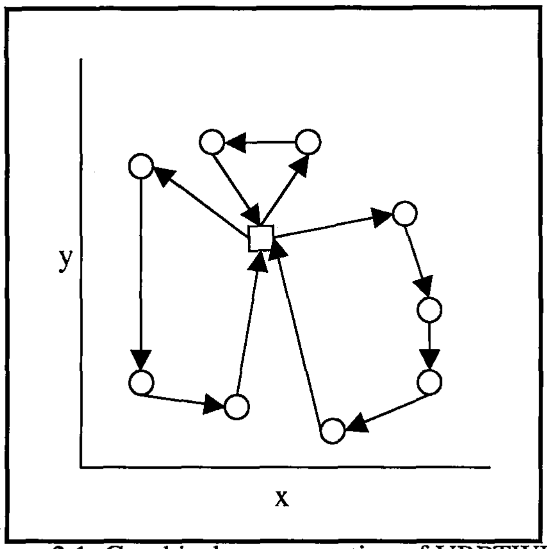


Figure 2.1: Graphical representation of VRPTWDC

In Fig. 2.1 the circles represent the customers (the demand points) and the square represents the warehouse. Using the locations in the *x-y* graph, the distance matrix is obtained and the problem is defined as a network representation for eliminating subtours and cycles.

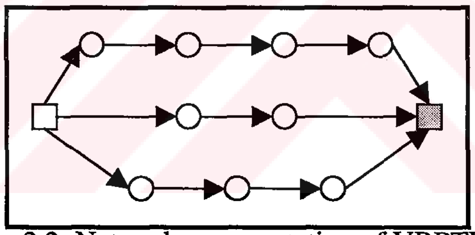


Figure 2.2: Network representation of VRPTWDC

In the network representation, the warehouse is represented by two squares. The white square is numbered as 0 and all routes originate from this point. Furthermore, the gray square is numbered as *N*+1 and all routes end at this point. For consistency in the representation, the warehouse (both the origin and the final destination) and the customers are named as nodes and all the routes as arcs.

The mathematical formulation of the VRPTWDC is given as follows:

Indices

| | |
|---|---|
| $i$ | Nodes (Customers and the warehouse) $0,1,2,K,N,N+1$ |
| $j$ | Nodes (Customers and the warehouse) $0,1,2,K,N,N+1$ |
| $k$ | Vehicles $1,2,K,K$ |
| $v$ | Vehicle types $1,2,K,V$ |

Variables

| | |
|---|---|
| $T_i$ | Arrival time at node $i$ |
| $W_i$ | Waiting time at node $i$ |
| $X_{ij}$ | $\begin{cases} 1 & \text{If node } i \text{ precedes } j \\ 0 & \text{otherwise} \end{cases}$ |
| $Y_k$ | $\begin{cases} 1 & \text{If vehicle } k \text{ is used} \\ 0 & \text{otherwise} \end{cases}$ |
| $Z_{ik}$ | $\begin{cases} 1 & \text{If customer } i \text{ is assigned to vehicle } k \\ 0 & \text{otherwise} \end{cases}$ |

Parameters

| | |
|---|---|
| $N$ | Total number of customers |
| $K$ | Total number of vehicles |
| $V$ | Total number of vehicle types |
| | |
| $d_{ij}$ | Euclidean distance between node $i$ and $j$ |
| $m_i$ | Demand at node $i$ |
| $q_v$ | Capacity of vehicle type $v$ |
| $e_i$ | Earliest arrival time at node $i$ |
| $l_i$ | Latest arrival time at node $i$ |
| $f_i$ | Service time at node $i$ |
| $r_k$ | Maximum work time allowed for vehicle $k$ |

| | |
|---|---|
| $vel$ | Velocity of vehicle |
| $fc_v$ | Fixed cost of using vehicle type $v$ |
| $n_{kv}$ | Equals to 1 if vehicle $k$ is of type $v$, otherwise it equals 0 |
| $tc$ | Traveling cost |
| $\overline{M_1}$ | A large number |

Minimize
$$\sum_{i=0}^{N+1}\sum_{j=0,j\neq i}^{N+1} tc\cdot d_{ij}\cdot X_{ij}+\sum_{v=1}^{V} fc_v\cdot\left(\sum_{k=1}^{K} n_{kv}\cdot Y_k\right) \qquad (2.1)$$

Subject to:

$$\sum_{i=0}^{N+1} X_{ii}=0 \qquad (2.2)$$

$$\sum_{j=0,j\neq i}^{N+1} X_{ij}=1 \qquad i\in\{1,\mathrm{K},N\} \qquad (2.3)$$

$$\sum_{j=0,j\neq i}^{N+1} X_{ji}=1 \qquad i\in\{1,\mathrm{K},N\} \qquad (2.4)$$

$$\sum_{k=1}^{K} Z_{ik}=1 \qquad i\in\{1,\mathrm{K},N\} \qquad (2.5)$$

$$\sum_{i=1}^{N} m_i\cdot Z_{ik}\leq\sum_{v=1}^{V} Y_k\cdot n_{kv}\cdot q_v \qquad \forall k \qquad (2.6)$$

$$\sum_{i=1}^{N} m_i\cdot Z_{ik}\geq\sum_{v=1}^{V} Y_k\cdot n_{kv}\cdot q_{v-1} \qquad \forall k \qquad (2.7)$$

$$T_j-T_i-W_i-f_i-\frac{d_{ij}}{vel}-\overline{M_1}X_{ij}\geq-\overline{M_1} \qquad \forall i,\forall j, j\neq 0 \qquad (2.8)$$

$$X_{ij}+Z_{ik}-Z_{jk}\leq 1 \qquad i,j\in\{1,\mathrm{K},N\},\forall k \qquad (2.9)$$

$$X_{0j}+Z_{jk}-Z_{0k}\leq 1 \qquad \forall j,\forall k \qquad (2.10)$$

$$X_{iN+1}+Z_{ik}-Z_{N+1k}\leq 1 \qquad \forall i,\forall k \qquad (2.11)$$

$$Z_{0k}+Z_{N+1k}-2Y_k=0 \qquad \forall k \qquad (2.12)$$

$$\sum_{i=0}^{N+1} Z_{ik} \leq N \cdot Y_k \qquad \forall k \qquad (2.13)$$

$$\sum_{i=0}^{N+1} Z_{ik} \geq Y_k \qquad \forall k \qquad (2.14)$$

$$T_i + W_i \leq l_i \qquad \forall i \qquad (2.15)$$

$$T_i + W_i \geq e_i \qquad \forall i \qquad (2.16)$$

$$\sum_{j=1}^{N} X_{0j} = \sum_{k=1}^{K} Y_k \qquad (2.17)$$

$$\sum_{i=0}^{N+1} X_{i0} = 0 \qquad (2.18)$$

$$\sum_{j=0}^{N+1} X_{N+1\,j} = 0 \qquad (2.19)$$

$$X_{0N+1} = 0 \qquad (2.20)$$

$$q_0 = 0 \qquad (2.21)$$

$$X_{ij} \in \{0,1\} \qquad \forall i, \forall j \qquad (2.22)$$

$$Y_k \in \{0,1\} \qquad \forall k \qquad (2.23)$$

$$Z_{ik} \in \{0,1\} \qquad \forall i, \forall k \qquad (2.24)$$

$$T_i \geq 0 \ , \ W_i \geq 0 \qquad \forall i \qquad (2.25)$$

The indices 0 and $N+1$ in $i$ and $j$ represent the warehouse. Constraints (2.2) state that a node cannot precede itself. According to the constraints (2.3) one and only one node can be visited after any customer $i$. Constraints (2.4) restrict the problem in such a way that a customer $i$ can only be visited by one node. Constraints (2.5) state that each customer is assigned to a single vehicle. Constraints (2.6) and (2.7)[1] state that the demands of the customers at any route must be between the capacities of vehicle types *v* and *v-1,* if that route is used. Constraints (2.8) give the time balance around the customer. The time balance around a customer is given by the following propositional logic in Fig. 2.3.

[1] *This constraint works if and only if the vehicle type capacities are listed in an ascending order*

If customer $j$ follows $i$, then $T_j - T_i - W_i - f_i - \dfrac{d_{ij}}{vel} \geq 0$,

otherwise $T_j, T_i$ and $W_i$ can be any positive number

$$X_{ij} \Rightarrow T_j - T_i - W_i - f_i - \frac{d_{ij}}{vel} \geq 0 \quad \wedge \quad \neg X_{ij} \Rightarrow T_j - T_i - W_i - f_i - \frac{d_{ij}}{vel} \geq -\left(\sum_{i=0}^{N}\sum_{j=0}^{N} d_{ij} + l_0\right)$$

$$T_j - T_i - W_i - f_i - \frac{d_{ij}}{vel} - X_{ij}\left(\sum_{u=0}^{N}\sum_{s=0}^{N}(d_{us}) + l_0\right) \geq -\sum_{u=0}^{N}\sum_{s=0}^{N}(d_{us}) - l_0$$

Figure 2.3: Logical description of the time balance constraints

where $\overline{M_1} = \sum_{u=0}^{N}\sum_{s=0}^{N}(d_{us}) + l_0$ is a large enough number.

The constraints (2.9) are important since they link the two binary variables. The propositional logic that is used in the constraint is as follows:

$$\left.\begin{array}{l}\text{If customer } j \text{ follows customer } i\\ \qquad\quad\wedge\\ \text{If customer } i \text{ is allocated on route } k\end{array}\right\}\ \begin{array}{l}\text{customer } j \text{ must also be allocated on}\\ \text{that same route}\end{array}$$

$$\begin{array}{l}(X_{ij}\wedge Z_{ik})\Rightarrow Z_{jk}\\ \neg(X_{ij}\wedge Z_{ik})\vee Z_{jk}\\ \neg X_{ij}\vee\neg Z_{ik}\vee Z_{jk}\\ 1-X_{ij}+1-Z_{ik}+Z_{jk}\geq 1\end{array}$$

$$X_{ij}+Z_{ik}-Z_{jk}\leq 1 \qquad\qquad \forall i, i\neq 0, \forall j, j\neq 0, \forall k$$

An alternative formulation for this constraint can be obtained if the sum over the vehicles is taken:

$$X_{ij}\sum_{k=1}^{K}\sum_{v=1}^{V}n_{kv}+\sum_{k=1}^{K}k(Z_{ik}-Z_{jk})\leq\sum_{k=1}^{K}\sum_{v=1}^{V}n_{kv} \qquad\qquad \forall i, i\neq 0, \forall j, j\neq 0, \forall k$$

where $\sum_{k=1}^{K}\sum_{v=1}^{V}n_{kv}$ is a large enough number.

Figure 2.4: Logical description of constraints linking $X_{ij}$, $Z_{ik}$ and $Z_{jk}$

Constraints (2.10) and (2.11) are the modified cases of the previous constraint for the warehouse. The constraints (2.12) specify that any route leaving the warehouse must end in the same warehouse. The following constraints (2.13) and (2.14) relate the two binary variables, $Z_{ik}$ and $Y_k$, to each other. They are formulated using the following propositional logic:

$$\sum_{i=0}^{N+1} Z_{ik} \Leftrightarrow Y_k$$

$$\sum_{i=0}^{N+1} Z_{ik} \Rightarrow Y_k \qquad\qquad Y_k \Rightarrow \sum_{i=0}^{N+1} Z_{ik}$$

$$\neg\sum_{i=0}^{N+1} Z_{ik} \vee Y_k \qquad\qquad \neg Y_k \vee \sum_{i=0}^{N+1} Z_{ik}$$

$$1-\sum_{i=0}^{N+1} Z_{ik} + N \cdot Y_k \geq 1 \qquad\qquad 1-Y_k+\sum_{i=0}^{N+1} Z_{ik} \geq 1$$

$$\sum_{i=0}^{N+1} Z_{ik} \leq N \cdot Y_k \qquad\qquad \sum_{i=0}^{N+1} Z_{ik} \geq Y_k$$

Figure 2.5: Logical description of constraints linking $Z_{ik}$ and $Y_k$

These linking constraints state that if a vehicle is not used then no nodes can be assigned to that vehicle. On the other hand, if a vehicle is used there must be at least one node assigned to that vehicle. Constraints (2.15) and (2.16) specify the earliest and latest time limitations for the deliveries. The constraints (2.17) state that the number of arcs originating from the warehouse equal the number of vehicles used. Constraints (2.18), (2.19), (2.20) and (2.21) set the limitations of the variables for the warehouse and the remaining constraints (2.22), (2.23), (2.24) and (2.25) specify the type of the variables.

The formulation of the VRPTW can be obtained from this representation when the fixed costs in the objective function are removed and constraints (2.6) and (2.7) are replaced with the following constraints:

$$\sum_{i=1}^{N} m_i \cdot Z_{ik} \leq q_1 \cdot Y_k \qquad \forall k \qquad (2.6')$$

where $q_1$ is the capacity of the single vehicle type.

## 2.2 *Solution Method for VRPTW*

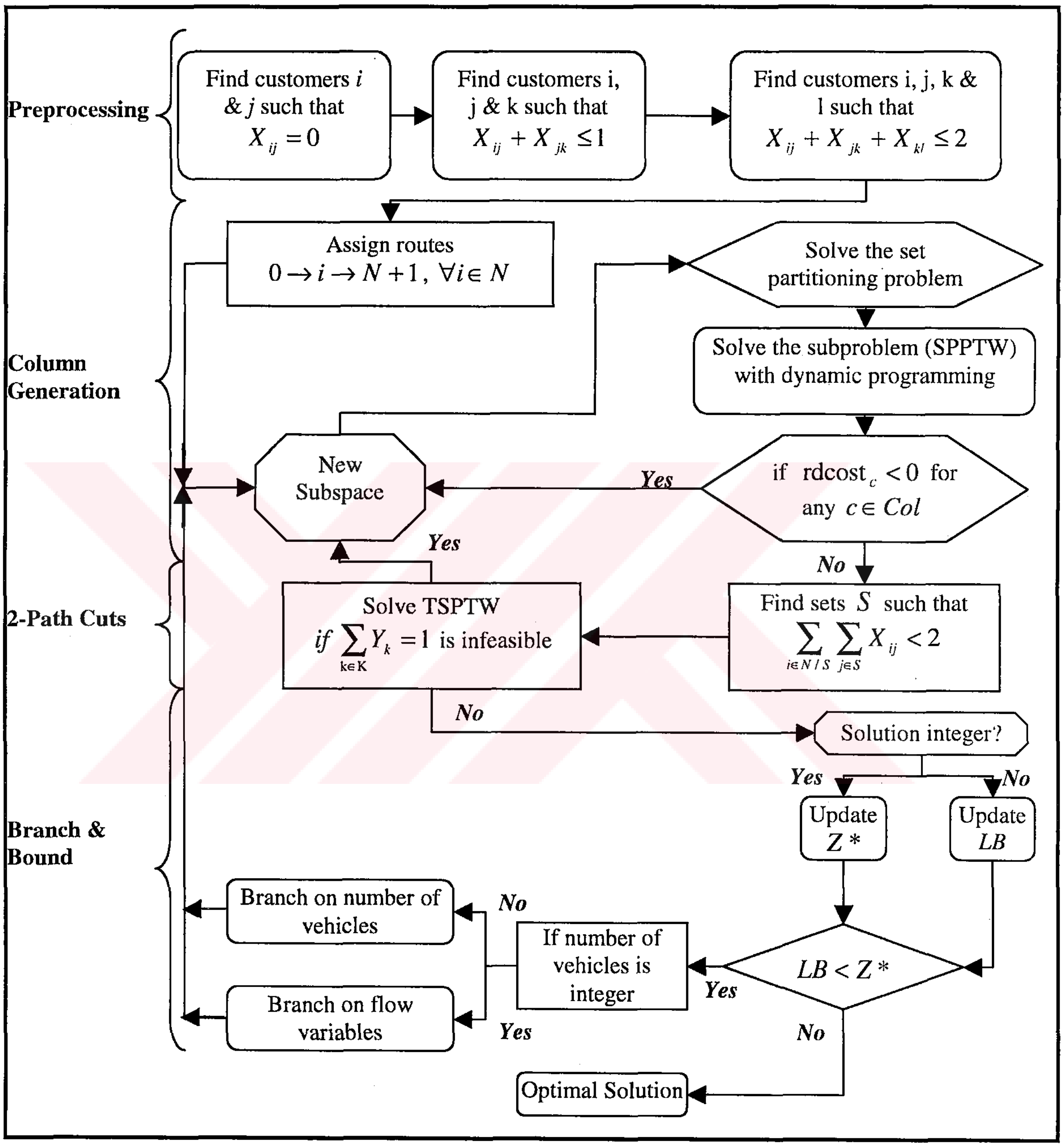

Preprocessing
Find customers $i$ & $j$ such that $X_{ij} = 0$
Find customers i, j & k such that $X_{ij} + X_{jk} \le 1$
Find customers i, j, k & l such that $X_{ij} + X_{jk} + X_{kl} \le 2$
Column Generation
Assign routes $0 \to i \to N+1, \forall i \in N$
Solve the set partitioning problem
Solve the subproblem (SPPTW) with dynamic programming
New Subspace
Yes
if $\text{rdcost}_c < 0$ for any $c \in Col$
Yes
No
2-Path Cuts
Solve TSPTW $if \sum_{k \in K} Y_k = 1$ is infeasible
Find sets $S$ such that $\sum_{i \in N/S} \sum_{j \in S} X_{ij} < 2$
No
Solution integer?
Yes
No
Update $Z^*$
Update $LB$
Branch & Bound
Branch on number of vehicles
No
If number of vehicles is integer
$LB < Z^*$
Branch on flow variables
Yes
Yes
No
Optimal Solution

Figure 2.6: The solution procedure for VRPTW

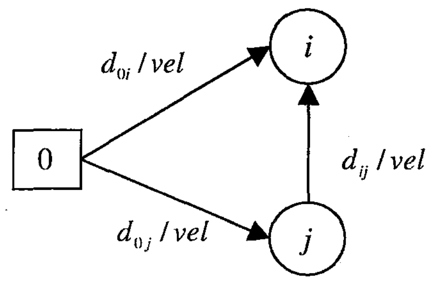
Although an effective integer formulation is presented in the previous section, the VRPTW and VRPTWDC still require major computation time for solving even small problems. The number of variables quickly increases and that makes it impossible to solve this problem within a reasonable time. Therefore, the following exact algorithm is used.

The solution procedure is shown in Fig. 2.4. The problem starts with preprocessing where some information is gathered before solving the optimization problem. Following the preprocessing, column generation, subtour and 2-path cuts generation, and branch and bound algorithms are employed using an iterative procedure. First, the relaxed problem is solved. Second, the subsets that violate the inequalities are found. After the iterative procedure, the MILP formulation is solved using branch and bound.

### *2.2.1 Preprocessing*

Preprocessing is very crucial in VRPTW and VRPTWDC that enables the reduction in the number of variables prior to the solution of the optimization problem. First the time windows are reduced by defining new limits for each customer according to possible earliest arrival and latest departure times. The method decribed in Desrosiers et al [46] is used in reducing the time windows. Then, three different preprocessing methods are introduced to fix certain variables. The aim of first method is to identify the customers that are visited directly after the warehouse.

**Preprocessing #1**



Travel time: $d_{ij}/vel$

Earliest acceptance time at $i$ = $e_i$

Latest acceptance time at $i$ = $l_i$

Service time at node $j$ = $f_j$

Earliest arrival time to $i$ by $j$ = $\frac{d_{0j}}{vel} + f_j + \frac{d_{ij}}{vel}$

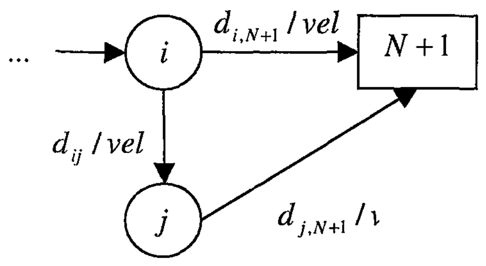
For any $i$, if $\frac{d_{0j}}{vel} + f_j + \frac{d_{ij}}{vel} > l_i \quad \forall j \Rightarrow X_{0i} = 1$

Figure 2.7: Preprocessing 1

For any customer $i$ there are two possibilities; either it is visited directly after the warehouse or it is visited after other customer(s). If only customer $j$ is visited before $i$, the earliest arrival time to the customer $i$ is $\frac{d_{0j}}{vel} + f_j + \frac{d_{ij}}{vel}$. This earliest arrival time, which only considers the traveling times and the service time at the customer $j$, does not include the waiting times. Therefore, if this earliest arrival time is greater than the latest acceptance time at customer $i$, it can be concluded that customer $j$ is not visited in between customer $i$ and the warehouse. Furthermore, if this is true for all $j$, then it is deduced that customer $i$ is visited directly after the warehouse.

The second preprocessing method is indeed the reverse of the first method. As shown in Fig. 2.8.

**Preprocessing #2**



Travel time: $d_{ij}/vel$

Earliest departure time at $i$: $e_i + f_i$

Latest departure time at $i$: $l_i + f_i$

Service time at node $j$: $f_j$

Earliest arrival time to the warehouse by *j*:

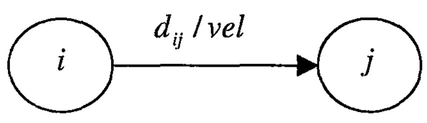
$$e_i + f_i + \frac{d_{ij}}{vel} + f_j + \frac{d_{j,N+1}}{vel}$$

For any $i$, if $e_i + f_i + \frac{d_{ij}}{vel} + f_j + \frac{d_{j,N+1}}{vel} > l_{N+1} \quad \forall j \quad \Rightarrow X_{i,N+1} = 0$

Figure 2.8: Preprocessing 2

In this method, it is observed whether *i* is the last customer on the route or not. There are two possibilities for the routes which depart from $i$. First, the vehicle can go directly to the warehouse and second, another customer $j$ may be visited in between. If any customer $j$ is visited, the earliest arrival time to the warehouse from customer $i$ is $e_i + f_i + \frac{d_{ij}}{vel} + f_j + \frac{d_{j,N+1}}{vel}$. Thus, if this time is greater than the latest acceptance time of the warehouse for all customers ( $j$ 's), it can be stated that customer $i$ is the last customer on a route.

The final preprocessing method is shown in the Fig. 2.9.



Earliest departure time from $i$ : $e_i + f_i$

Latest acceptance time at $j$ : $l_j$

If $e_i + f_i + \frac{d_{ij}}{vel} > l_j \quad \Rightarrow x_{ij} = 0$

Figure 2.9: Preprocessing # 3

This method is applied to any two customers. The earliest departure time from customer $i$ is $e_i + f_i$ and the latest acceptance time at customer $j$ is $l_j$. Thus, if the minimum traveling time from customer $i$ to $j$ is greater than the time difference $l_j - e_i$, it can be concluded that customer $j$ does not follow customer $i$. This last method is called the arc elimination. Although this arc elimination is performed between two customers, it can also be applied for 3 or more customers can also be done using the same logic. For example, if it is proved that three specific customers 1, 2 and 3 cannot follow each other, it can be modeled as: $X_{12} + X_{23} \leq 1$.

### 2.2.2 *Column Generation – The Master Problem*

Although preprocessing helps to remove significant number of arcs, there is still a need for better lower bounds for the problem. Larsen[42] and Kohl[40], showed that column generation proved to be successful in generating feasible paths and therefore in obtaining a tight lower bound. Column generation technique is employed on linear programming models with large number of variables. Here, the formulation for the VRPTW can be modeled as a set partitioning problem:

Indices

| | |
|---|---|
| $i$ | Customers $1,2,\mathrm{K}\ ,N$ |
| $r$ | Paths $1,2,\mathrm{K}\ ,R$ |

Variables

| | |
|---|---|
| $XR_r$ | 1 if path $r$ is used, 0 otherwise |

Parameters

| | |
|---|---|
| $N$ | Total number of customers |
| $R$ | Total number of feasible paths |
| $\delta_{ir}$ | 1 if customer $i$ is on path $r$ |
| $c_r$ | Total cost of the path $r$ |

Minimize $\sum_{r=1}^{R} c_r XR_r$ (2.26)

Subject to

$$\sum_{r=1}^{R} \delta_{ir} XR_r = 1 \qquad \forall i \in \{1,\mathrm{K}\ ,N\} \qquad (2.27)$$

$$XR_r = \{0,1\} \qquad \forall r \in R \qquad (2.28)$$

This formulation of the VRPTW as a set partitioning model depends on the generation of the set $\boldsymbol{R_i}$. This formulation finds the optimal cost of the problem if all feasible sets are generated. In this formulation, the sequence of customers is not represented; however, the cost of the path $c_r$ is calculated according to the sequence of customers. The number of feasible paths is usually large in many of the problems, which forces to relax the integrality constraints, and solve the problem as an LP. Thus, the set partitioning problem is defined as the *master problem*, and the set of feasible paths are to be generated in the *subproblem* for the column generation process.

The master problem for the VRPTW can be changed to VRPTWDC by changing the route costs only which now include the fixed costs of the vehicles.

The procedure starts with the initialization phase where $depot \rightarrow i \rightarrow depot$ routes are formed for every customer. Thus, the initial set of ***R*** contains *N* routes, and this gives the upper bound to the original problem. As the master problem is solved, the dual variables associated with the constraints (2.27), $\lambda_j^1$ are stored for the subproblem to generate feasible routes.

### *2.2.3 Column Generation - The Subproblem*

As the assignment type constraints are handled in the master problem, the subproblem is reduced to a shortest path problem with time windows with capacity constraints. The formulation of the subproblem is as follows:

Minimize $$\sum_{i=0}^{N+1} \sum_{j=0, j\neq i}^{N+1} \hat{d}_{ij} \cdot X_{ij} \qquad (2.29)$$

Subject to:

$$\sum_{i=0}^{N+1} X_{ii} = 0 \qquad (2.30)$$

$$\sum_{i=1}^{N} m_i \cdot \sum_{j=0}^{N+1} X_{ij} \leq q_v \qquad (2.31)$$

$$T_j - T_i - W_i - f_i - \frac{d_{ij}}{vel} - \overline{M_1} X_{ij} \geq -\overline{M_1} \qquad \forall i, \forall j, j \neq 0 \qquad (2.32)$$

$$T_i + W_i \leq l_i \qquad \forall i \qquad (2.33)$$

$$T_i + W_i \geq e_i \qquad \forall i \qquad (2.34)$$

$$\sum_{i=0}^{N+1} X_{i0} = 0 \qquad (2.35)$$

$$\sum_{j=0}^{N+1} X_{N+1\,j} = 0 \qquad (2.36)$$

$$X_{0N+1} = 0 \qquad (2.37)$$

$$X_{ij} = 0,1 \qquad \forall i, \forall j \qquad (2.38)$$

$$T_i \geq 0 \quad , W_i \geq 0 \qquad \forall i \qquad (2.39)$$

It can be seen from the constraint (2.31), this formulation is valid for a single vehicle type. Therefore, in VRPTW $q_v = q$, since there is only a single vehicle type, whereas for VRPTWDC this problem must be solved for each vehicle type independently at each call to the subproblems. The objective function of the subproblem is to minimize the reduced cost of the master problem and therefore $\hat{d}_{ij} = d_{ij} - \lambda_j^1$ is the modified distance. The parameter $\lambda_j^1$ is associated with the customer *j* in constraints (2.27). Solving this problem gives a short path with the most negative reduced cost for the master problem and requires a large computing time. An effective dynamic programming algorithm was presented in Larsen[42], which follows the Dijkstra's Algorithm. In this work, a slightly different algorithm is proposed for solving the VRPTW which reduces the computation time.

The dynamic program is formulated using the basic concepts of Dijkstra's algorithm. The following information is known for each label:

i. the current node *(DYN$_n$)*
ii. the modified cost *(DYN$_c$)*
iii. arrival time to the current node *(DYN$_t$)*
iv. predecessors of the current node in vector form *(DYN$_p$)*
v. current load on the vehicle *(DYN$_d$)*
vi. type of the label *(DYN$_y$)*

A label can only be strongly dominant *(DYN$_y$=1)* or weakly dominant *(DYN$_y$=2)*, where for each weakly dominant label the information on possible successors is kept in vector form *(DYN$_x$)*. Then the labels can be represented as follows:

For VRPTWDC , the initial label for a vehicle type *v* are:

$$DYN_n[0] = 0 \qquad (2.40)$$

$$DYN_c[0] = fc_v, \tag{2.41}$$

$$DYN_t[0] = 0 \tag{2.42}$$

$$DYN_d[0] = 0 \tag{2.43}$$

$$DYN_y[0] = 1 \tag{2.44}$$

$$DYN_p[0] = \{\mathbf{1000}\}, \tag{2.45}$$

$$DYN_x[0] = \{\mathbf{1000}\} \tag{2.46}$$

Other labels are generated using the following recursive formula:

$$DYN_n[lb] = j \tag{2.47}$$

$$DYN_c[lb] = DYN_c[lb'] + \hat{d}_{DYN_n[l'],j} \tag{2.48}$$

$$DYN_t[lb] = \max\left\{DYN_t[lb'] + \frac{d_{DYN_n[lb'],j}}{vel}, e_j\right\} \tag{2.49}$$

$$DYN_d[lb] = DYN_d[lb'] + m_j \tag{2.50}$$

$$DYN_p[lb] = \left\{\overline{DYN}_p[lb'] \quad DYN_n[lb'] \quad \mathbf{1000}\right\} \tag{2.51}$$

The vector $\{\mathbf{1000}\}$ contains elements that are equal to 1000, and $\overline{DYN}_p[lb']$ represents the vector of elements of $DYN_p[lb']$that are less than 1000. Note that $DYN_y[lb]$ and $DYN_x[lb]$ have not been explained yet. For each label a type is assigned that determines whether the label is strongly or weakly dominant. The dominance means: for any two labels $lb'$ and $lb$, if:

$$[DYN_n[lb'] = DYN_n[lb]] \;\&\; [DYN_t[lb'] \le DYN_t[lb]] \;\&\; [DYN_d[lb'] \le DYN_d[lb]] \;\&$$
$$[DYN_c[lb'] \le DYN_c[lb]] \tag{2.52}$$

Then it can be said that label $lb'$ dominates label $lb$. The domination property is very important because some of the dominated labels can be discarded and this speeds up the column generation significantly. The reason is that any successor of the dominated label is dominated by the same successor of the dominating label. The labels are discarded using the knowledge on elimination of 2-cycles which are well studied in Irnich[23]. If $lb'$ dominates label $lb$, $lb$ can be discarded if:

1) If $lb'$ is a weakly dominated label
2) If it is not possible to revisit the predecessor node of $lb'$ due to either
    a. time windows or
    b. capacity constraints
3) If it is possible to revisit the predecessor node of $lb'$ and
    a. If the predecessor node of $lb'$ is not accessible from $lb$ by either,
        i. time windows constraints or
        ii. capacity constraints or
        iii. if the predecessor node of $lb'$ is already visited by $lb$
    b. If $lb$ is dominated by other labels that have different predecessor nodes than $lb'$

If a label is dominated and not discarded then it becomes a weakly dominant label and it can only be extended to the intersection of the predecessors of the dominating labels and the elements of this intersection set are placed in $DYN_x[lb]$. The code written in OPL Studio 3.5 for the dynamic programming problem is given in Fig. 2.10.

```
Bestn:=0;Bestc:=FC[v];Bestt:=0;Bestd:=0;Besty:=1;
forall(j2 in 1..N){Bestp[j2]:=1000;Bestx[j2]:=1000;}
forall(i in [1..N+1]){cntlb[i]:=0;}start:=0;

while ((sum(i in [1..N+1])(cntlb[i])>0) ∨ start=0) do {start:=1;
  forall(i1 in 1..N+1){
    if(i1<>Bestp[1..N] ∧
      (Besty=2 ∧ i1=Bestx[1..N)then{
      if(Bestn<>i1 &
         Bestt+d[Bestn,i1]/vel<=l[i1]&
New      Bestd+m[i1] <= q[v]) then {  //Label is Feasible//
Labels   cntlb[i1]:=cntlb[i1]+1;
         newdynt:= max(Bestt+d[Bestn,i1]/vel,e[i1]);
          //Place the label in an ascending order according to demands//
          forall(ii1 in 1..cntlb[i1]:newdynt>=dynt[i1,ii1]){
             place:=ii1;break;}
         dynn[i1,place]:=i1;
         dync[i1,place]:=Bestc+d[Bestn,i1]-dual[i1];
         dynt[i1,place]:=max(Bestt+d[Bestn,i1]/vel,e[i1]);
         dynd[i1,place]:=Bestd+m[i1];
         dyny[i1,place]:=1;
         dynp[i1,place,1..N]:=Bestp[1..N] ∪ Bestn;
  }}}
  forall(i in [1..N+1]){
  forall(i10 in 1..cntlb[i]){
    forall(i11 in 1..cntlb[i]  ∧  i11<i10){
      if(dynn[i,i10]=dynn[i11]   ∧  dync[i,i10]<=dync[i,i10]  ∧
         dynt[i,i10]<=dynt[i,i10]  ∧  dynt[i,i10]<=dynt[i,i10])
         then{dyny[i,i11]:=2;
          if(dyny[i,i10]=2  ∨
              dynp[i,i10,1..N] ∪ dynp[i,i11,1..N]=dynp[i,i10,1..N]  &
Dominance     dynp[i,i101..N] ∪ dynp[i,i11,1..N]=dynp[i,i10,1..N])then{
Criterion        cntlb[i]:=cntlb[i]-1; //Discard label i11//
              else
                 dynx[i,i11,1..N]:=dynp[i,i10,1..N]
  }}}}}
   forall(i20 in [1..N+1]){
      if(dynt[i20,1] = min(i21 in 1..N+1)(dynt[i21,1])) then {
         Bestn:=i1;Bestc:=dync[i20,1];
Best     Bestt:=dynt[i20,1];Bestd:=dynd[i20,1];
Label    Besty:=dyny[i20,1];
         Bestp[1..N]:=dynp[i20,1];
         Bestx[1..N]:=dynx[i20,1];
         cntlb[i20]:=cntlb[i20]-1; //Discard Label i20//}}}
```

Figure 2.10: Dynamic Programming code for solving the SPPTW

### *2.2.4 2-Path Cuts*

Insertion of valid inequalities to VRPTWDC reduces the solution time significantly. Previous work on the insertion of valid inequalities includes 2-path cuts method that aims to find some sets of customers that require a minimum of two vehicles whereas the solution to the relaxed problem states that the customers are satisfied with less than two customers. Therefore, the purpose of the 2-path cuts is the insertion of the following constraint:

$$\sum_{i \in N/S} \sum_{j \in S} X_{ij} \geq 2 \tag{2.53}$$

where, the flow into the set is found to be less than 2 in the relaxed problem. The solution approach of the 2-path cuts is an iterative procedure; the relaxed problem is solved first, the results of the relax problem is evaluated next. Note that the 2-path cuts algorithm is only executed at the root node for practical reasons.

The next step is to find the sets where $\sum_{i \in N/S} \sum_{j \in S} X_{ij} < 2$. These sets are found using an iterative procedure named as the 2-path cut separation algorithm. In the previous work of Kohl[40], a heuristic procedure is applied to find these sets. However, it is more efficient to consider the intra-route sets, since for the customers that are distributed in different routes the flow into the set is always greater than or equal to 2. The separation algorithm evaluates the customers according to nearest neighbor basis and continues to add customers into a subset until is $\sum_{i \in N/S} \sum_{j \in S} X_{ij} < 2$ violated, and the last customer is removed from the set.

Following this step, these sets are examined whether their TSPTW problem is infeasible or not with a single vehicle. Thus, the following property is investigated:

$$\sum_{i \in N/S} \sum_{j \in S} X_{ij} > 1 \tag{2.54}$$

If the TSPTW problem for the subset is infeasible with one vehicle, then it can be deduced that this set is suitable for 2-path cuts.

After the cuts are found using the separation algorithm they are added to the master problem. This addition is done by employing new constraints (4.23) and (4.24) in the master problem. These constraints are explained in detail in section 4.2.

### 2.2.5 *Branch and Bound*

The final step is the optimization of the MILP that is formulated in section 2. Previous steps in the proposed method enabled the bounds to be tightened and therefore, the solution time of the branch and bound algorithm will be reduced. Although there are several branching decisions that can be applied to branch and bound, probably the most effective one is the best bound search. After the column generation and 2-path cuts the solution is usually fractional. It is shown that branching on the total number of vehicles is the most successful method. Therefore, two child nodes are created and the constraints $\sum_{r=1}^{R} XR_r = \lfloor k \rfloor$ and $\sum_{r=1}^{R} XR_r = \lceil k \rceil$ are added to the two child nodes, respectively. The new solution is then returned to the column generator and iterations continue until $LB = Z^*$. In the case where the total number of vehicles is integer, the branching continues with the flow variables.

## 2.3 *Illustrative Example*

It is best to describe the results with an example for the VRPTWDC. The following example is R205 with 25 customers from Solomon's benchmark problems (Solomon[27]). The upper bound on the number of vehicles is set at 3 for each vehicle type. The customer data for the problem is given in detail in Appendix A. Using the locations, 26x26 distance matrix is formed by using Euclidean distance measure[2]. The vehicle data and the other parameters are as follows:

Table 2.1: Other parameters for R205

| | |
|---|---|
| Number of Vehicle Types | 3 |
| Average velocity (km/h) | 60 |

[2] Euclidean distance: Distance $= \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$ where X and Y are the coordinates

Table 2.2: Vehicle Data for VRPTWDC

| Vehicle Type | Capacity | Fixed Cost |
|---|---|---|
| 1 | 150 | 50 |
| 2 | 200 | 75 |
| 3 | 250 | 100 |

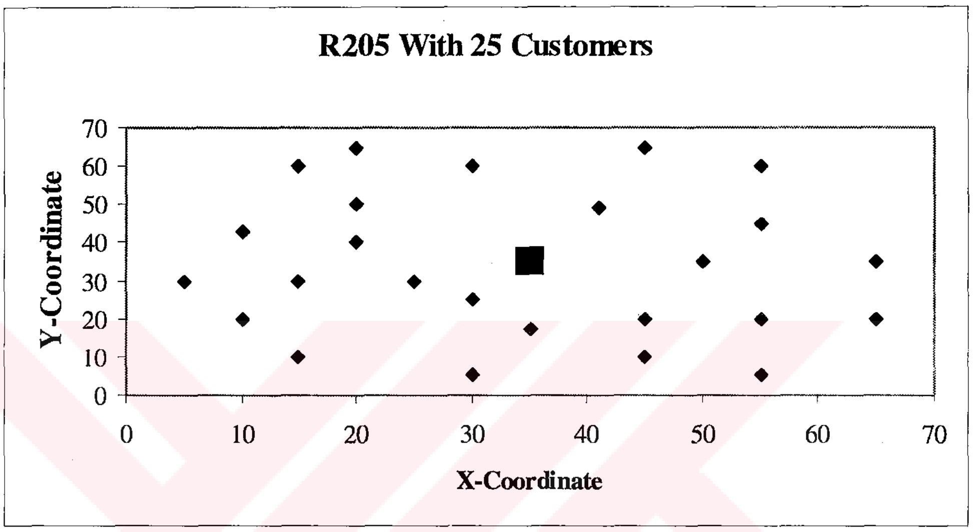


Figure 2.11: Warehouse and demand point locations of the illustrative example

The VRPTW only considers a single type of vehicle with a capacity of 200. The solution for the VRPTW is shown in Fig. 2.12.

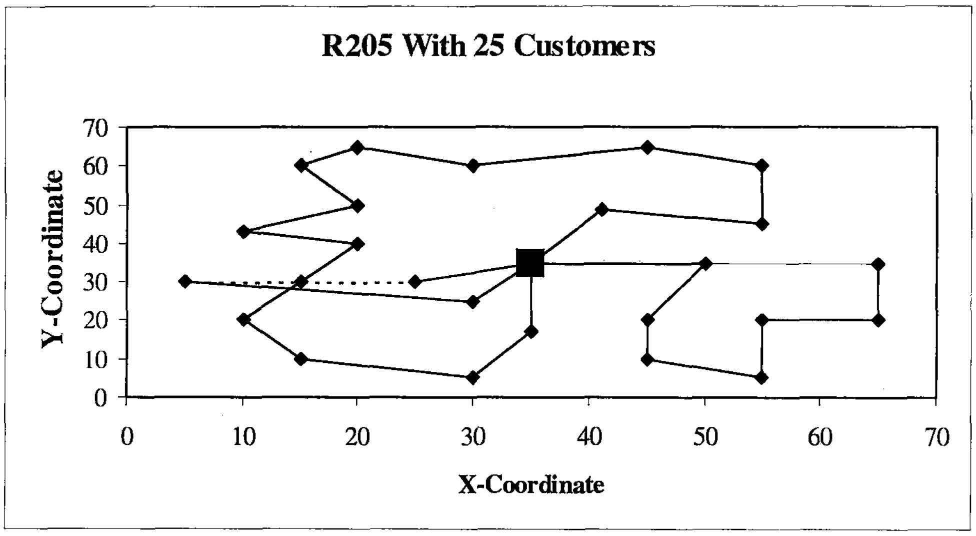


Figure 2.12: VRPTW solution of R205.025

In this solution three vehicles are used and the distance cost sums up to 393.00. Only one of the vehicles has a capacity greater than 150, hence the objective function of this solution according to VRPTWDC equals 393.00+50.00+50.00+75.00 = 568.00. However, this is not the optimal value for the VRPTWDC problem. The dotted line in Fig. 2.12 is an arc linking the two customers, and the customer on the center is not visited by this arc. The optimal solution for the problem is given in Fig. 2.13.

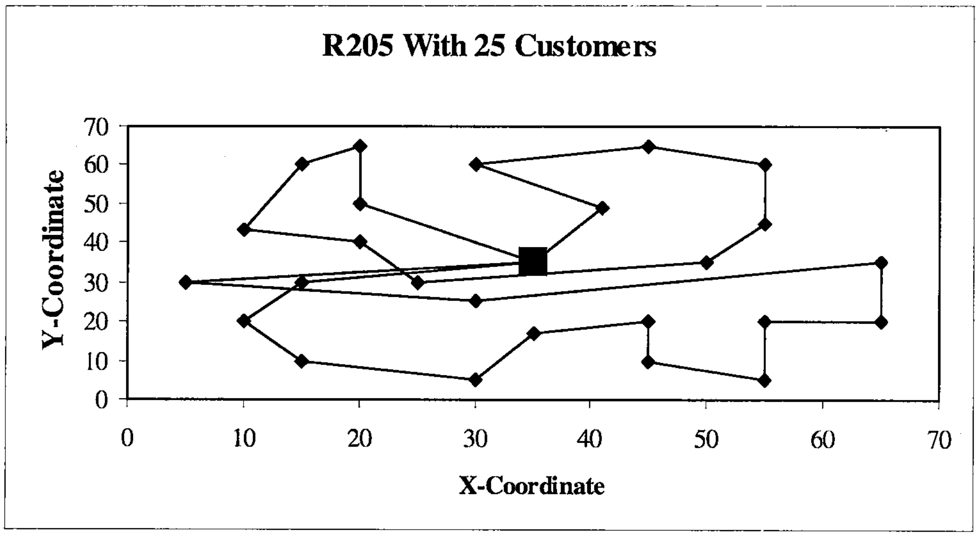


Figure 2.13: VRPTWDC solution of problem R205

The objective function of this problem is 551.80 which is less than the solution obtained by the VRPTW. Two vehicles are used in this solution with vehicle capacities 150 and 200. Thus, it is shown that the VRPTWDC problem can be used in the cases where the fleet is heterogeneous. The comparison of the results is as follows:

Table 2.3: Comparison of the results for R205 with VRPTWDC cost function

| **Problem** | **$LB_1$** | **$LB_2$** | **IP** | **Opt gap** | **Veh** | **Node** | **VI** | **Sub** |
|---|---|---|---|---|---|---|---|---|
| VRPTW | 568.00 | 568.00 | 568.00 | 0.0 | 3 | 1 | 0 | 89 |
| VRPTWDC | 527.08 | 536.27 | 551.80 | 0.05 | 2 | 11 | 9 | 112 |

In the VRPTW case the solution is obtained at the root node without generating the cuts. However, the lower bound of the VRPTWDC solution is lower than the optimal solution. In order to reduce the branch and bound nodes, 2-path cuts are examined at the root node. The total number of valid inequalities at the root node sums up to 9, and some of the optimality gap is closed prior to branch and bound. In the final solution of the master problem the problem size is represented as follows:

Table 2.4: Problem size for R205

| | |
|---|---|
| **Number of Variables** | 1956 |
| **Number of Constraints** | 2114 |
| **Number of Nodes** | 11 |

### *2.4 Computational Results*

The problems solved for the VRPTWDC are modified from the Solomon benchmark problems, Solomon[27]. Since VRPTWDC can also be used for strategic level decisions higher solution times are tolerable. Furthermore, the original Solomon Benchmark Problems are used for the VRPTW problems. These problem sets are classified into three groups. The R-problems contain customers that are randomly located in a certain region. The C-problems contain customers which are located in clusters. Finally, in the RC-problems the customers are located both randomly and in clusters, indeed, a mixture of the two distributions. The results of some of the problems are shown in Table 2.5.

Table 2.5: Results of some VRPTW problems

| **Problem** | **$LB_1$** | **$LB_2$** | **IP** | **Opt gap** | **Veh** | **Node** | **VI** | **Sub** |
|---|---|---|---|---|---|---|---|---|
| R101.025 | 617.10 | 617.10 | 617.10 | 0.0 | 8 | 1 | 0 | 7 |
| R101.050 | 1043.37 | 1044.00 | 1044.00 | 0.1 | 12 | 1 | 8 | 20 |
| R101.100 | 1631.15 | 1633.85 | 1637.70 | 0.4 | 20 | 23 | 11 | 124 |
| R104.025 | 416.90 | 416.90 | 416.90 | 0.0 | 4 | 1 | 0 | 13 |
| R107.025 | 424.30 | 424.30 | 424.30 | 0.0 | 4 | 1 | 0 | 27 |
| C101.025 | 191.30 | 191.30 | 191.30 | 0.0 | 3 | 1 | 0 | 16 |
| C101.050 | 362.40 | 362.40 | 362.40 | 0.0 | 5 | 1 | 0 | 41 |
| C101.100 | 827.30 | 827.30 | 827.30 | 0.0 | 10 | 1 | 0 | 53 |
| C103.025 | 190.30 | 190.30 | 190.30 | 0.0 | 3 | 1 | 0 | 35 |
| C103.050 | 361.40 | 361.40 | 361.40 | 0.0 | 5 | 1 | 0 | 77 |
| C107.025 | 191.30 | 191.30 | 191.30 | 0.0 | 3 | 1 | 0 | 25 |

| Problem | LB$_1$ | LB$_2$ | IP | Opt gap | Veh | Node | VI | Sub |
|---|---|---|---|---|---|---|---|---|
| C107.050 | 362.40 | 362.40 | 362.40 | 0.0 | 5 | 1 | 0 | 51 |
| C107.100 | 827.30 | 827.30 | 827.30 | 0.0 | 10 | 1 | 0 | 130 |
| RC101.025 | 406.63 | 452.08 | 461.10 | 13.4 | 4 | 7 | 11 | 29 |
| RC101.050 | 892.61 | 931.11 | 944.00 | 6.2 | 8 | 19 | 49 | 60 |
| RC101.100 | 1584.09 | 1610.82 | 1619.80 | 2.3 | 15 | 40 | 74 | 158 |
| RC105.025 | 410.95 | 410.95 | 411.30 | 0.1 | 4 | 3 | 0 | 21 |
| RC105.050 | 754.44 | 850.41 | 855.30 | 13.4 | 8 | 21 | 17 | 159 |
| RC106.025 | 345.50 | 345.50 | 345.50 | 0.0 | 3 | 1 | 0 | 71 |
| R205.025 | 393.00 | 393.00 | 393.00 | 0.0 | 3 | 1 | 0 | 89 |

In this table, LB$_1$ shows the objective function obtained at the root node, LB$_2$ shows the improved objective after the addition of the cuts at the root node, IP shows the integer optimal values, opt gap states the optimality gap between LB$_1$ and IP as a fraction, Veh stands for the number of vehicles used in the problem, Node shows the number of nodes in the branch and bound tree, VI shows the number of valid inequalities inserted and Sub shows the number of subproblems solved in the solution. For some of the problems, the LB$_1$ is obtained as the as the IP solutions. This is the most important contribution of this work. This improvement is due to the elimination of the 3-cycles which are explained in the next section.

Table 2.6: Results of some VRPTWDC problems

| Problem | q$_v$ | LB$_1$ | LB$_2$ | IP | Opt gap | Veh | Node | VI | Sub |
|---|---|---|---|---|---|---|---|---|---|
| R101.025 | 150-200-250 | 1017.00 | 1017.00 | 1017.00 | 0.00 | 8 | 1 | 0 | 29 |
| R104.025 | 150-200-250 | 616.90 | 616.90 | 616.90 | 0.00 | 4 | 1 | 0 | 74 |
| R107.025 | 150-200-250 | 624.30 | 624.30 | 624.30 | 0.00 | 4 | 1 | 0 | 86 |
| C101.025 | 150-200-250 | 337.19 | 338.50 | 341.30 | 0.02 | 3 | 11 | 4 | 80 |
| C103.025 | 150-200-250 | 340.30 | 340.30 | 340.30 | 0.00 | 3 | 1 | 0 | 66 |
| C107.025 | 150-200-250 | 340.67 | 341.30 | 341.30 | 0.01 | 3 | 1 | 8 | 39 |
| RC101.025 | 150-200-250 | 636.63 | 661.10 | 661.10 | 0.04 | 4 | 1 | 37 | 44 |

| **Problem** | $q_v$ | **LB$_1$** | **LB$_2$** | **IP** | **Opt gap** | **Veh** | **Node** | **VI** | **Sub** |
|---|---|---|---|---|---|---|---|---|---|
| RC105.025 | 150-200-250 | 602.82 | 604.15 | 611.30 | 0.02 | 4 | 27 | 6 | 107 |
| RC106.025 | 150-200-250 | 487.29 | 487.29 | 495.50 | 0.01 | 3 | 36 | 0 | 90 |
| R205.025 | 150-200-250 | 527.08 | 536.27 | 551.80 | 0.05 | 2 | 11 | 9 | 112 |

## *2.5 Conclusions*

In this section two routing problems are considered in depth. The formulated VRPTWDC problem is an extended version of the previous vehicle routing problems and contains specific vehicle capacity constraints which differentiate VRPTWDC from VRPTW. It considers the time windows and it also provides operational level decisions on the vehicle capacities for routing. Decision on the vehicle capacities has a positive effect on the problems and they prove to find better solutions than the traditional VRPTW problems when the fixed costs of vehicles are considered.

The formulation of the problem contains several assumptions. The first assumption is very common in the vehicle routing problems that any customer is located on a single vehicle (route). This assumption avoids multi travels to a customer and thus, the order can be traveled in a single routing. Moreover, it is logical in the cases where the warehouse is located far away from the customers and where the customers are clustered. The reason is that it is intuitively feasible to satisfy the customer need in a single routing rather than replenishing more than once. Another assumption is that any vehicle can be used only once. This assumption is also logical if the customer orders are evaluated at the decision center daily. The reason is that usually when the trucks return to the warehouse there is not enough time for it to make another delivery in day times. The velocity of the trucks is taken as 60km/hr. This is indeed an average value for trucks which is logical. However, for better definitions it would be better of having different velocities, since the average speed can increase to 80-90km/hr in highways or it can decrease up to 30-40km/hr in narrow country roads. The final assumption is that the warehouse contains unlimited unit and unlimited vehicles of each vehicle types. The problem VRPTWDC does not take into account the supplier and/or the manufacturer side and therefore that any order can be satisfied from the warehouse at any given time.

Using these assumptions the mathematical model of the problem is presented in section 2. Considerable effort is performed in forming the constraints and the logical expressions; thus, the resulting formulation is an MILP formulation.

The solution procedure is composed of the preprocessing part, column generation, 2-path cuts generation, and branch and bound. The preprocessing methods help to identify the first and the last customers on a route. Furthermore, it can also be deduced from the preprocessing that certain pairs of customers cannot be adjacent. Thus, by gathering this information the number of the variables is reduced significantly. The preprocessing proves to be more effective if the time windows is tight that is, if the constraints on the time limitations are strict, the preprocessing reduces more variables.

After the preprocessing and arc elimination, the column generation method is applied. Here, the master problem is defined as a set partitioning problem and the subproblem is defined as an elementary shortest path problem with time windows. The master problem is solved using LP and the subproblem is solved by a dynamic programming algorithm. This algorithm differs from the previous programs in the literature, in the way that it does not allow *k*-cycles that mean eliminating visits to a node twice in a specific route at the *i* th and *k+i* th positions. An example of a 2-cycle is $i_1 \rightarrow i_2 \rightarrow i_1$. In the previous studies, researchers could find a way to eliminate 2-cycles by employing types to each label and by reconsidering the dominance criteria. However, 3-cycle or more cycles cannot be eliminated. The presence of such routes decreases the lower bound at the column generation and thus increases the optimality gap. In the proposed method the predecessors of each node are kept as a vector and by using the dominance rules the *k*-cycles are not allowed in the dynamic programming algorithm. This proved to be very successful such that in some of the problems the optimality gap is reduced without increasing the computational complexity. There are several important points in the discussion of the column generation and the dynamic programming algorithm. The dynamic programming algorithm usually takes long CPU times in the solutions of the problems and several parameters that are described in Chapter 6, are used to stop the algorithm when certain conditions are satisfied.

The next step is to determine valid inequalities. The theory of valid inequalities is handled by employing 2-path cuts. Generation of 2-path cuts prove to be successful when certain sets of customers are found. The flow into these sets of customers must be less than two from the results of the relaxed problem and also the set must be infeasible when its TSPTW problem is solved. As shown in the illustrative example, the 2-path cuts strengthen the bounds significantly.

Finally the branch and bound algorithm is applied. According to the branching priorities, first the number of vehicles is branched. After several iterations, when the total number of vehicles becomes integer, branching is applied on the flow variables.

Another contribution is the two index formulation of the VRPTW. The model of the integer program is formulated using propositional logic. All the previous models on the VRPTW uses the adjacency variable as $X_{ijk}$. However, in this work the adjacency variable is formulated by employing two binary variables $X_{ij}$ and $Z_{ik}$, and by adding a new constraint which logically relates these two variables. Thus, for a 100 customer problem the variables due to $X$ reduces from 100,000 to 11,000 where 10,000 of the variables are from $X_{ij}$ and 1000 variables from $Z_{ik}$. New constraints are added for relating the two integer variables, $X_{ij}$ and $Z_{ik}$. Two formulations are presented for this relation by the Eqns. (2.9), (2.10) and Fig. 2.4. Although, the formulation that includes Eqn. (2.9) generates more constraints, it covers the convex hull of the problems more efficiently. Therefore, the solution time is reduced by employing the initial formulation. Another important property of the model is that it contains only linear constraints. In the previous work by Tan[26] and Kohl[40], the problem is presented as nonlinear programming problem due to the time balance around the customers. The TSPTW problem that is used in the 2-path cuts is the same model without the capacity constraints.

Chapter 3

# DESCRIPTION OF WLRP

The definition of the WLRP is first given in Perl[3], this formulation is changed to MWLRP in Perl and Daskin[4]. Hansen et al.[5] proposed a new formulation by discarding the subtour constraints by using a flow variable and finally Wu et al.[6] formulated the same problem with auxiliary variables. All of the above formulations use three-index variables. Hansen's[5] formulation is given below for comparison with the formulation given in this thesis:

Indices

| | |
|---|---|
| $i$ | Customers $1,2,\ldots,N$ |
| $j$ | Depots $N+1,N+2,\ldots,N+M$ |
| $g$ | Nodes $1,2,\ldots,N+M$ |
| $h$ | Nodes $1,2,\ldots,N+M$ |
| $k$ | Vehicles $1,2,\ldots,K$ |

Variables

$$X_{ghk} \quad \begin{cases} 1 & \text{If node } g \text{ precedes } h \text{ on route } k \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{ij} \quad \begin{cases} 1 & \text{If customer } i \text{ is allocated to depot } j \\ 0 & \text{otherwise} \end{cases}$$

$$Z_{j} \quad \begin{cases} 1 & \text{If depot } j \text{ is established} \\ 0 & \text{otherwise} \end{cases}$$

$$ZZ_{hk} \quad \begin{cases} 1 & \text{If node } h \text{ is assignted to route } k \\ 0 & \text{otherwise} \end{cases}$$

$F_{gik}$ The truck load between node $g$ and customer $i$ on route $k$

Parameters

| | |
|---|---|
| $N$ | Number of customers |
| $M$ | Number of potential depots |
| $K$ | Allowed number of vehicles |
| | |
| $d_{gh}$ | Distance between node *g* and node *h* |
| $q_g$ | Demand of node *g* |
| $fc_j$ | Fixed cost of establishing depot *j* |
| $vc_j$ | Variable cost per unit throughput at depot *j* |
| $t_j$ | Maximum throughput at depot *j* |
| $c_k$ | Capacity of vehicle *k* |
| $cm$ | Cost per mile |

Min $$\sum_{j=N+1}^{N+M}(fc_j\cdot Z_j)+\sum_{j=N+1}^{N+M}\sum_{i=1}^{N}(vc_j\cdot q_i\cdot Y_{ij})+\sum_{k=1}^{K}\sum_{g=1}^{N+M}\sum_{h=1}^{N+M}(cm\cdot d_{gh}\cdot X_{ghk}) \quad (3.1)$$

Subject to:

$$\sum_{k=1}^{K}\sum_{h=1,h\neq i}^{N+M}X_{ihk}=1 \qquad \forall i \quad (3.2)$$

$$\sum_{i=1}^{N}q_i\cdot ZZ_{ik}\le c_k \qquad \forall k \quad (3.3)$$

$$\sum_{g=1,g\neq h}^{N+M}X_{hgk}-\sum_{g=1,g\neq h}^{N+M}X_{ghk}=0 \qquad \forall h,\forall k \quad (3.4)$$

$$\sum_{i=1}^{N}q_i\cdot Y_{ij}-t_j\cdot Z_j\le 0 \qquad \forall j \quad (3.5)$$

$$ZZ_{ik}+ZZ_{jk}-Y_{ij}\le 1 \qquad \forall i,\forall j,\forall k \quad (3.6)$$

$$Y_{ij}-Z_j\le 0 \qquad \forall i,\forall j \quad (3.7)$$

$$ZZ_{hk}-\sum_{g=1,g\neq h}^{N+M}X_{ghk}=0 \qquad \forall h,\forall k \quad (3.8)$$

$$\sum_{j=N+1}^{N+M}Y_{ij}=1 \qquad \forall i \quad (3.9)$$

$$ZZ_{jk}-z_j\le 0 \qquad \forall j,\forall k \quad (3.10)$$

$$\sum_{g=1,g\neq i}^{N+M} F_{gik} - \sum_{r=1,r\neq i}^{N} F_{irk} - ZZ_{ik}\cdot q_i = 0 \qquad \forall i, \forall k \qquad (3.11)$$

$$F_{gik} - X_{gik}\cdot(c_k - q_g) \leq 0 \qquad \forall i, \forall g, \forall k, g \neq i \qquad (3.12)$$

$$\sum_{i=1}^{N}\sum_{j=N+1}^{N+M} Y_{ij} = N \qquad (3.13)$$

$$X_{ghk} = 0,1 \qquad \forall g, \forall h, \forall k \qquad (3.14)$$

$$Y_{ij} = 0,1 \qquad \forall i, \forall j \qquad (3.15)$$

$$Z_j = 0,1 \qquad \forall j \qquad (3.16)$$

$$ZZ_{hk} = 0,1 \qquad \forall h, \forall k \qquad (3.17)$$

$$F_{gik} \geq 0 \qquad \forall g, \forall i, \forall k, g \neq i \qquad (3.18)$$

This WLRP formulation is efficient only for small problems. The subtour elimination constraints in Perl's[3] formulation are handled wisely by implementing the truck load variables $F_{ghk}$. Since this formulation contains a lot of redundant constraints and variables, a new formulation that eliminates these redundancies is developed in this chapter.

The first redundancy appears in the new variables $F_{ghk}$ such that these variables contain information about the destination and the route of the truck; however, this information is already available in variables $X_{ghk}$. Thus, it is proposed to model $F_{ghk}$ as $F_g$ and use the destination and the route information of $X_{ghk}$ in the constraints. This change would lead to a decrease in the number of variables from $K(N+M)^2$ to $N+M$.

Another redundancy is seen in the allocation variables $Y_{ij}$ that are required in Hansen's formulation for modeling the capacity constraints for each warehouse in Eqn. (3.5). However, $Y_{ij}$ is obtained directly from the flow variables $X_{ghk}$ and hence it is proposed to remove $Y_{ij}$ and use a combination of $X_{ghk}$ and $F_g$ to model the capacity constraints of the warehouses. However, it is seen that it is impossible to model the truck load out from each warehouses with a single index and the variables $F_g$ are changed to $F_i$ and $FW_{jk}$. Together with this change, the variables $Y_{ij}$ are removed.

The final redundancy is seen in the flow variables $X_{ghk}$ that contain allocation information for each pair of customers. However the allocation variables $ZZ_{gk}$ are already present in Hansen's formulation, containing the allocation information for each node, and thus it is proposed to discard the last index in $X_{ghk}$ and model it as $X_{gh}$. Thus, the proposed formulation becomes a two-index formulation for the WLRP, and the number of variables are reduced from

$2K(N+M)^2+NM+K(N+M)+M$ to $(N+M)^2+M+K(N+M)+N+KM$ where for the 85-customer, 7-warehouse and 15-vehicle problem this decrease is from 204842 to 9744.

The proposed formulation is as follows:

Indices

| | |
|---|---|
| $i$ | Customers *1,2,...,N* |
| $j_1$ | Departure depots *N+1,N+2,...,N+M* |
| $j_2$ | Arrival depots *N+M+1,N+M+2,...,N+2M* |
| $g$ | Nodes *1,2,...,N+2M* |
| $h$ | Nodes *1,2,...,N+2M* |
| $k$ | Vehicles *1,2,...,K* |

Variables

$$X_{gh}\begin{cases}1 & \text{If node } g \text{ precedes } h\\ 0 & \text{otherwise}\end{cases}$$

$$Z_{j_1}\begin{cases}1 & \text{If depot } j_1 \text{ is established}\\ 0 & \text{otherwise}\end{cases}$$

$$ZZ_{hk}\begin{cases}1 & \text{If node } h \text{ is assignted to route } k\\ 0 & \text{otherwise}\end{cases}$$

| | |
|---|---|
| $F_i$ | The truck load carried out from customer $i$ |
| $FW_{j_1k}$ | The truck load carried out from depot $j_1$ on route $k$ |

Parameters

| | |
|---|---|
| $N$ | Number of customers |
| $M$ | Number of potential depots |
| $K$ | Allowed number of vehicles |
| $d_{gh}$ | Distance between node $g$ and node $h$ |
| $q_g$ | Demand of node $g$ |
| $fc_{j_1}$ | Fixed cost of establishing depot $j_1$ |
| $vc_{j_1}$ | Variable cost per unit throughput at depot $j_1$ |
| $t_{j_1}$ | Maximum throughput at depot $j_1$ |
| $c_k$ | Capacity of vehicle $k$ |

| | |
|---|---|
| $cm$ | Cost per mile |

min $$\sum_{j=N+1}^{N+M}\left(fc_j \cdot Z_j\right)+\sum_{j=N+1}^{N+M}\sum_{k=1}^{K}\left(vc_j \cdot FW_{jk}\right)+\sum_{g=1}^{N+M}\sum_{h=1}^{N+M}\left(cm \cdot d_{gh} \cdot X_{gh}\right) \quad (3.19)$$

subject to

$$\sum_{h=1,h\neq i}^{N+2M} X_{ih}=1 \qquad \forall i \qquad (3.20)$$

$$\sum_{i=1}^{N} q_i \cdot ZZ_{ik} \leq c_k \qquad \forall k \qquad (3.21)$$

$$\sum_{g=1,g\neq h}^{N+2M} X_{hg} - \sum_{g=1,g\neq h}^{N+M} X_{gh}=0 \qquad h\in\{1,\mathrm{K},N+M\} \qquad (3.22)$$

$$\sum_{j=N+1}^{N+2M} ZZ_{jk} \leq 1 \qquad \forall k \qquad (3.23)$$

$$K \cdot X_{ih}+\sum_{k=1}^{K} k \cdot \left(ZZ_{ik}-ZZ_{hk}\right)\leq K \qquad \forall i, \forall h \qquad (3.24)$$

$$K \cdot X_{hi}+\sum_{k=1}^{K} k \cdot \left(ZZ_{ik}-ZZ_{hk}\right)\leq K \qquad \forall i, \forall h \qquad (3.25)$$

$$\sum_{i=1}^{N} X_{ji}-\sum_{k=1}^{K} ZZ_{jk}=0 \qquad j\in\{N+1,\mathrm{K},N+2M\} \qquad (3.26)$$

$$\sum_{k=1}^{K} ZZ_{j_1k}-K \cdot Z_{j_1}\leq 0 \qquad \forall j_1 \qquad (3.27)$$

$$\sum_{k=1}^{K} ZZ_{j_1k}-Z_{j_1}\geq 0 \qquad \forall j_1 \qquad (3.28)$$

$$\sum_{k=1}^{K} ZZ_{ik}=1 \qquad \forall i \qquad (3.29)$$

$$f_{i_1}-f_{i_2}-q_{i_2}X_{i_1i_2}-X_{i_1i_2}\cdot\sum_{u=1}^{N} q_u \geq -\sum_{u=1}^{N} q_u \qquad \forall i_1, \forall i_2 \qquad (3.30)$$

$$\left(\sum_{j_2=N+M+1}^{N+2M} X_{ij}\right)\cdot\left(\sum_{u=1}^{N} q_u\right)+f_i \leq \sum_{u=1}^{N} q_u \qquad \forall i \qquad (3.31)$$

$$\sum_{j_1=N+1}^{N+M} FW_{j_1k}-\sum_{i=1}^{N} q_i \cdot ZZ_{ik}=0 \qquad \forall k \qquad (3.32)$$

$$FW_{j_1k}-\sum_{u=1}^{U} q_u \cdot ZZ_{j_1k}\leq 0 \qquad \forall j_1, \forall k \qquad (3.33)$$

$$\sum_{k=1}^{K} FW_{j_1 k} - t_{j_1} \cdot Z_{j_1} \leq 0 \qquad \forall j_1 \qquad (3.34)$$

$$ZZ_{j_1 k} - ZZ_{M+j_1,k} = 0 \qquad \forall j_1 \qquad (3.35)$$

$$\sum_{g=1}^{N+2M} \sum_{j_1=N+1}^{N+M} X_{g j_1} = 0 \qquad (3.36)$$

$$\sum_{g=1}^{N+2M} \sum_{j_2=N+M+1}^{N+2M} X_{j_2 g} = 0 \qquad (3.37)$$

$$X_{gh} = 0,1 \qquad \forall g, \forall h \qquad (3.38)$$

$$Z_{j_1} = 0,1 \qquad \forall j_1 \qquad (3.39)$$

$$ZZ_{hk} = 0,1 \qquad \forall h, \forall k \qquad (3.40)$$

$$F_i \geq 0 \qquad \forall i \qquad (3.41)$$

$$FW_{j_1 k} \geq 0 \qquad \forall j_1, \forall k \qquad (3.42)$$

Constraints (3.20) state that each customer must be followed by exactly one node. The sum of the capacities of customers in each route must be less than or equal to the vehicle capacity, and this is handled in constraints (3.21). Constraints (3.22) restrict that every node entered by the vehicle should be left by the same vehicle. Constraints (3.23) require that for every route there can be at most one warehouse assigned. Constraints (3.24) and (3.25) link the allocation and routing components using the propositional logic given in Figs. 3.1, 3.2 and 3.3.

$$ZZ_{ik} \wedge ZZ_{j_1 k} \Rightarrow Y_{ij_1}$$

$$ZZ_{ik} \wedge ZZ_{j_1 k} \Rightarrow Y_{ij_1}$$

$$\neg(ZZ_{ik} \wedge Z_{j_1 k}) \vee Y_{ij_1}$$

$$\neg ZZ_{ik} \vee \neg Z_{j_1 k} \vee Y_{ij_1}$$

$$(1 - ZZ_{ik}) + (1 - ZZ_{j_1 k}) + Y_{ij_1} \geq 1$$

$$ZZ_{ik} + ZZ_{j_1 k} - Y_{ij_1} \leq 1$$

Figure 3.1: Logical description of the constraints linking the variables $ZZ_{ik}$, $ZZ_{j_1 k}$ and $Y_{ij_1}$

However, in the proposed formulation the variables $Y_{ij_1}$ are removed in order to reduce the number of variables. Therefore the allocation variables $ZZ_{ik}$ are tied with the flow variables $X_{gh}$, following the logic:

$$
\boxed{
\begin{array}{l}
(X_{ih} \vee X_{hi}) \wedge ZZ_{ik} \Rightarrow ZZ_{hk} \\
\neg[(X_{ih} \vee X_{hi}) \wedge ZZ_{ik}] \vee ZZ_{hk} \\
\neg(X_{ih} \vee X_{hi}) \vee \neg ZZ_{ik} \vee ZZ_{hk} \\
(\neg X_{ih} \wedge \neg X_{hi}) \vee (\neg ZZ_{ik} \vee ZZ_{hk}) \\
(\neg X_{ih} \vee \neg ZZ_{ik} \vee ZZ_{hk}) \wedge (\neg X_{hi} \vee \neg ZZ_{ik} \vee ZZ_{hk}) \\
(1 - X_{ih}) + (1 - ZZ_{ik}) + ZZ_{hk} \geq 1 \quad \& \quad (1 - X_{hi}) + (1 - ZZ_{ik}) + ZZ_{hk} \geq 1 \\
X_{ih} + ZZ_{ik} - ZZ_{hk} \leq 1 \quad \& \quad X_{hi} + ZZ_{ik} - ZZ_{hk} \leq 1 \qquad\qquad \forall i, \forall h, \forall k
\end{array}
}
$$

Figure 3.2: Logical description of the constraints linking the variables $ZZ_{ik}$, $ZZ_{hk}$ and $X_{ih}$

However, if these constraints are added to the problem the number of constraints would increase by $2N(N+M)K$. This increase would lead to large computing times in the relaxation phase of the branch and bound, therefore these constraints are aggregated over the vehicles to form the constraints (3.24) and (3.25). For example, consider two customers $i_1$ and $i_2$ such that $i_2$ follows $i_1$. In this case the Boolean variable $x_{i_1 i_2}$ must have a value of Yes.

$$Kx_{i_1i_2} + k_1 zz_{i_1k_1} - k_2 zz_{i_2k_2} \le K$$

$$Kx_{i_2i_1} + k_2 zz_{i_2k_2} - k_1 zz_{i_1k_1} \le K$$

$$Kx_{i_1i_2} + k_2 zz_{i_2k_2} - k_1 zz_{i_1k_1} \le K$$

$$Kx_{i_2i_1} + k_1 zz_{i_1k_1} - k_2 zz_{i_2k_2} \le K$$

leading to

$$k_1 zz_{i_1k_1} - k_2 zz_{i_2k_2} \le 0$$

$$k_1 zz_{i_1k_1} - k_2 zz_{i_2k_2} \ge 0$$

Therefore, customers $i_1$ and $i_2$ can only be placed at the same vehicle (meaning that $k_1$ and $k_2$ must be equal).

Figure 3.3: Validation of the constraints (3.24) and (3.25)

Constraints (3.26), (3.27) and (3.28) tie the allocation and routing variables for the warehouses, stating that every warehouse that is used must be assigned to a route. Although constraints (3.26) seem to be redundant for the IP, it provides better bounds in the LP relaxation phase of the branch and bound. Constraints (3.29) state that each customer must be assigned to a route. Constraints (3.30) specify the load distribution between two customers $i_1$ and $i_2$ such that if customer $i_2$ follows $i_1$ then the sum of the outflow from $i_2$ and $i_2$'s demand equals to the outflow from $i_1$. Constraints (3.31) state that the outflow from the last customer in the route must be zero. Constraints (3.32) require that the outflow from the warehouse must be equal to the total load on the vehicle for each vehicle. Constraints (3.33) tie the allocation and outflow variables such that if a vehicle is not used then the outflow from the warehouse on that route must be zero. The constraints (3.34) state that the total outflow from each warehouse must be less than the warehouse capacity. The constraints (3.35) link the departure and arrival depots such that each route starts and ends at the same depot. Constraint (3.36) stipulates that there can be no arcs to a departure depot and constraint (3.37) states that there can be no arcs originating from the arrival depots.

Although this WLRP formulation reduces the problem size for integer programming, it is still impossible to solve the problem directly. An efficient exact algorithm is presented in the next chapter.

Chapter 4

# SOLUTION MEHOD FOR WLRP

The solution method for the WLRP is given below:

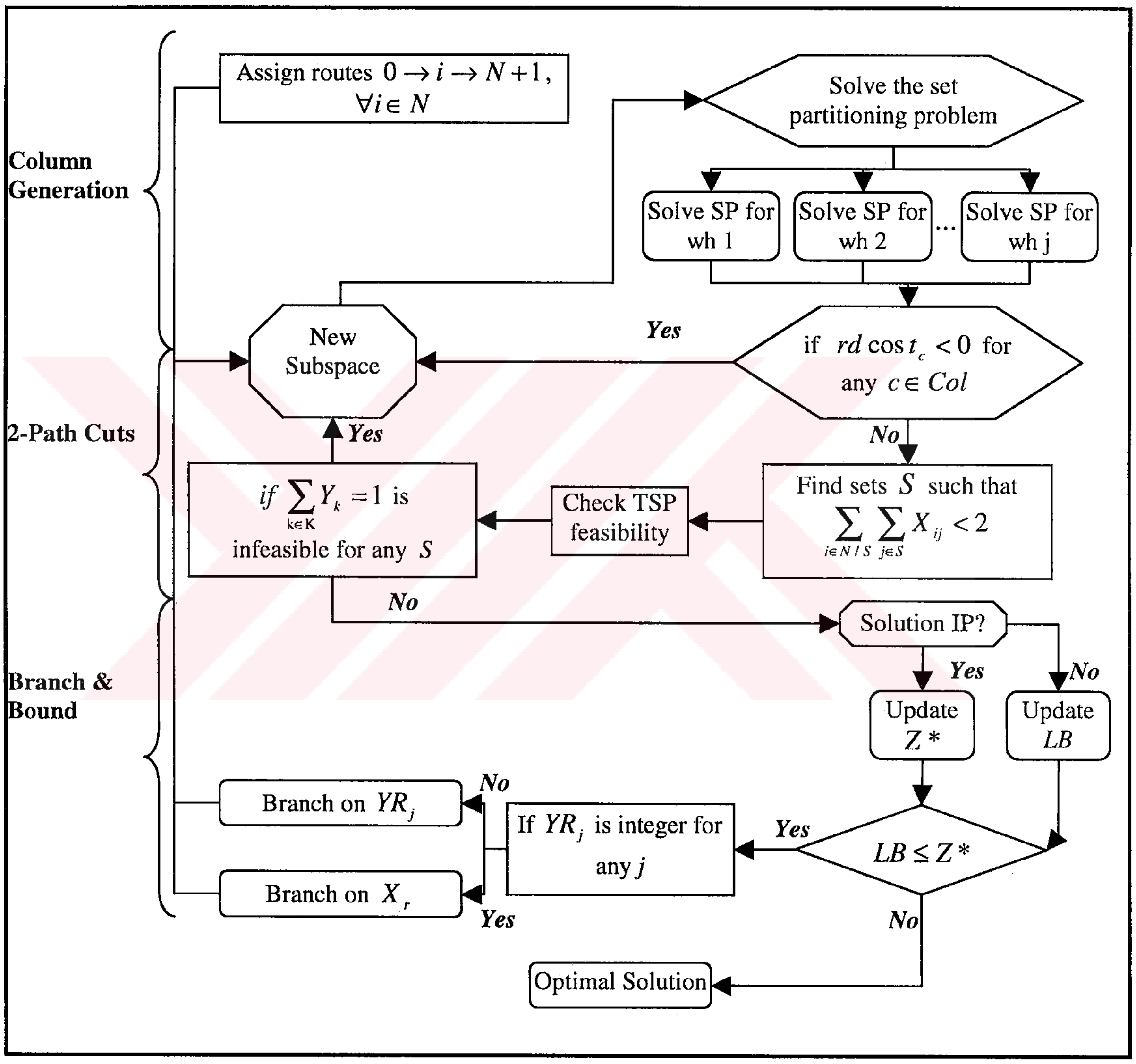

Column Generation
2-Path Cuts
Branch & Bound
Assign routes $0 \to i \to N+1$, $\forall i \in N$
Solve the set partitioning problem
Solve SP for wh 1
Solve SP for wh 2
Solve SP for wh j
New Subspace
Yes
if $rd\cos t_c < 0$ for any $c \in Col$
No
Find sets $S$ such that $\sum_{i \in N/S} \sum_{j \in S} X_{ij} < 2$
Check TSP feasibility
if $\sum_{k \in K} Y_k = 1$ is infeasible for any $S$
Yes
No
Solution IP?
Yes
No
Update $Z^*$
Update $LB$
$LB \le Z^*$
Yes
No
If $YR_j$ is integer for any $j$
No
Yes
Branch on $YR_j$
Branch on $X_r$
Optimal Solution

Figure 4.1: Solution method for WLRP

The algorithm initializes itself by assigning a single customer to a route that starts and ends in the same depot (i.e. $depot_j \rightarrow i \rightarrow depot_j$). Next, the master problem is solved and using the values of the dual variables of the master problem, the subproblem is solved for each warehouse independently. The column generation scheme continues until no more feasible routes are found and the objective value of the master problem at this stage specifies the lower bound for the problem. At this point, the 2-path cuts and subtour cuts are generated and the solution is again returned to the column generation algorithm. If the solution to the master problem is not integer, then branch and bound algorithm is employed to find integer solutions. The priority in the selection of branching variables is given to the warehouse allocation variables since these variables may result in larger variation in the objective function values. If all allocation variables are integer then the total number of vehicles is branched and finally, flow variables $XR_r$ are selected as branches.

## *4.1 Column Generation*

Like VRPTW, WLRP can be solved using the column generation method. In order to implement the column generation the WLRP must be decomposed effectively to a master problem and a subproblem. When the proposed formulation is examined it can be seen that there are two types of assignment variables: variables for assignment of node pairs $X_{gh}$ and variables for assignment of nodes to vehicles $ZZ_{gk}$. The constraints (3.22), (3.24), (3.25) and (3.26) satisfy the assignment constraints for the WLRP and they are independent from the other constraints.

### *4.1.1 The Master Problem*

Using the independence of the assignment constraints and flow constraints, the WLRP can be decomposed into an assignment problem and a shortest path problem. The assignment problem can be formulated as a set partitioning problem and it becomes the master problem. The formulation is as follows:

Indices

| | |
|---|---|
| $i$ | Customers $1,2,\dots,N$ |
| $j$ | Depots $N+1, N+2,\dots,N+M$ |
| $r$ | Paths $1,2,..,R$ |

Variables

| | |
|---|---|
| $XR_r$ | 1 if path $r$ is used, 0 otherwise |
| $YR_j$ | 1 if depot $j$ is established |

Parameters

| | |
|---|---|
| $N$ | Total number of customers |
| $M$ | Total number of warehouses |
| $R$ | Total number of feasible paths |
| $\delta_{ir}$ | 1 if customer $i$ is on path $r$ |
| $\mu_{jr}$ | 1 if depot $j$ is on path $r$ |
| $c_r$ | Total cost of the path $r$ |
| $fc_j$ | Fixed cost of establishing depot $j$ |
| $t_j$ | Maximum throughput at depot $j$ |
| $lw$ | Trivial lower bound on the number of depots |

Minimize $$\sum_{r=1}^{R} c_r XR_r + \sum_{j=N+1}^{N+M} fc_j YR_j \qquad (4.1)$$

Subject to

$$\sum_{r=1}^{R} \delta_{ir} XR_r = 1 \qquad \forall i \in N \qquad (4.2)$$

$$(N+M)YR_j - \sum_{r=1}^{R} \mu_{jr} XR_r \geq 0 \qquad \forall j \in \{N+1, N+M\} \qquad (4.3)$$

$$YR_j - \sum_{r=1}^{R} \mu_{jr} XR_r \leq 0 \qquad \forall j \in \{N+1, N+M\} \qquad (4.4)$$

$$\sum_{j=N+1}^{N+M} YR_j \geq lw \qquad (4.5)$$

$$\sum_{j=N+1}^{N+M} \mu_{jr} YR_j - XR_r \geq 0 \qquad \forall r \in R \qquad (4.6)$$

$$\sum_{r=1}^{R} \mu_{jr} \cdot \hat{q}_r \cdot XR_r \le t_j \cdot YR_j \qquad \forall j \in \{N+1, N+M\} \qquad (4.7)$$

$$XR_r = \{0,1\} \qquad \forall r \in R \qquad (4.8)$$

$$YR_j = \{0,1\} \qquad \forall j \in \{N+1, N+M\} \qquad (4.9)$$

The objective function (4.1) represents the objective function of the original WLRP. The fixed costs of establishing depots $fc_j$ are exactly the same in the formulation of WLRP. The costs $c_r$ include the traveling cost of the route $r$ and the variable cost of assigning route $r$ to depot $j$ which depends on the amount of load carried by the vehicle that is assigned to this route. The constraints (4.2) stipulate that each customer is visited exactly once. Constraints (4.3) and (4.4) link the variables $YR_j$ and $XR_r$ such that if $XR_r$ is 1 (meaning that if the route $r$ is used), then the depot $j$ that is assigned to the route in the subproblem must be used (i.e. $YR_j = 1$). Constraint (4.5) imposes a lower bound for the number of warehouses used depending on the capacities of the vehicles. Constraints (4.6) further link the variables $YR_j$ and $XR_r$ . Although constraints (4.6) seem redundant they are added to the problem because they provide better lower bounds in the relaxation phase. Constraints (4.7) specify that the sum of the capacities of the routes assigned to a specific depot $j$ must be less than or equal to the warehouse capacity.

This set partitioning formulation of the WLRP, again, depends on the generation of the set ***R***. The difference between the VRPTW and the WLRP is that the costs $c_r$ include the variable costs for the warehouses in the WLRP. The logic behind the column generation follows the logic presented in section 2.2.3 such that if all feasible sets are generated this formulation finds the optimal cost of the problem. However, due to the large number of feasible paths this problem cannot be solved as an integer problem and thus the integrality constraints (4.8) and (4.9) are relaxed.

The algorithm starts by assigning every customer to a different route that starts and ends from every depots similar to the initialization of the proposed algorithm for VRPTW. Therefore, there are $N \times M$ routes initially in the set ***R***, and this gives the upper bound to the original problem. As the master problem is solved, the dual variables associated with the constraints (4.2), (4.3), (4.4) and

(4.7) are stored for the subproblem to generate feasible routes and are named as $dual_i^2$, $dual_j^3$, $dual_j^4$ and $dual_j^7$ respectively.

### *4.1.2 The Subproblem*

The master problem handles the assignment constraints and the warehouse capacity constraints. Other capacity constraints remain in the subproblem and reducing the subproblem to a shortest path problem with capacity constraints. Formulation of the subproblem for any warehouse *j* is as follows:

Minimize $$\sum_{g=1}^{N\cup j} \sum_{h=1,h\neq g}^{N\cup j} \hat{d}_{gh} \cdot X_{gh} \qquad (4.8)$$

Subject to:

$$\sum_{g=1}^{N\cup j} X_{gg} = 0 \qquad (4.9)$$

$$\sum_{g=1}^{N\cup j} q_g \cdot \sum_{h=1}^{N\cup j} X_{gh} \leq c \qquad (4.10)$$

$$X_{gh} = 0,1 \qquad \forall g \in \{N \cup j\}, \forall h \in \{N \cup j\} \qquad (4.11)$$

The nodes in this formulation include only a single warehouse *j*, and therefore this problem must be solved for each warehouse. Moreover, this problem is for a single vehicle type as in VRPTW. The objective function of the subproblem is to minimize the reduced cost of the master problem and $\hat{d}_{gh} = d_{gh} - dual_h^2$ is the modified distance. Solving this problem gives a single short path with the most negative reduced cost for the master problem in a significantly large computing time. Another dynamic programming algorithm is formulated for this problem similar to the dynamic programming formulation in section 2.2.3. The subproblem is solved using Dijkstra's Algorithm without the time constraints. Since the time constraints are not included in this problem, several issues appear that must be handled properly.

For each label the following information are known: the current node *($DYN_n$)*, its modified cost *($DYN_c$)*, its predecessors in vector form *($DYN_p$)*, current load on the vehicle *($DYN_d$)* and the type of the label *($DYN_y$)*. A label can only be strongly dominant *($DYN_y=1$)* or weakly dominant *($DYN_y=2$)*. as in the VRPTW. The information on possible successors is kept in vector form *($DYN_x$)* for each weakly dominant label. Then the labels can be represented as follows:

For WLRP , the initial label for a warehouse *j* are:

$$DYN_n[0]=0 \tag{4.12}$$

$$DYN_c[0]=dual_j^3+dual_j^4, \tag{4.13}$$

$$DYN_d[0]=0 \tag{4.14}$$

$$DYN_y[0]=1 \tag{4.15}$$

$$DYN_p[0]=\{\mathbf{1000}\}, \tag{4.16}$$

$$DYN_x[0]=\{\mathbf{1000}\} \tag{4.17}$$

Other labels are generated using these recursive formulas

$$DYN_n[lb]=g \tag{4.18}$$

$$DYN_c[lb]=DYN_c[lb']+\hat{d}_{DYN_n[l'],g}+VC_j\cdot q_i-dual_j^7\cdot q_i \tag{4.19}$$

$$DYN_d[lb]=DYN_d[lb']+q_g \tag{4.20}$$

$$DYN_p[lb]=\{\overline{DYN_p}[lb'] \quad DYN_n[lb'] \quad \mathbf{1000}\} \tag{4.21}$$

The vector $\{\mathbf{1000}\}$ contains elements that are equal to 100, and $\overline{DYN_p}[lb']$ represents the vector of elements of $DYN_p[lb']$ that are less than 1000. $DYN_y[lb]$ and $DYN_x[lb]$ carry the same information as in the VRPTW case. Each label has a type and it determines whether the label is strongly or weakly dominant. The dominance rules for any two labels $lb'$ and $lb$ are slightly different from the VRPTW case. If:

$$DYN_n[lb'] = DYN_n[lb] \ \& \ DYN_d[lb'] \leq DYN_d[lb] \ \& \ DYN_c[lb'] \leq DYN_c[lb] \qquad (4.22)$$

Then label $lb'$ dominates label $lb$. As in VRPTW, the domination property is very crucial because some of the dominated labels can be discarded and this speeds up the column generation significantly. Any successor of the dominated label is dominated by the same successor of the dominating label as in the VRPTW case. The conditions for discarding is slightly different from the VRPTW. If $lb'$ dominates label $lb$, $lb$ can be discarded if:

1) If $lb'$ is a weakly dominated label
2) If it is not possible to revisit the predecessor node of $lb'$ due to capacity constraints
3) If it is possible to revisit the predecessor node of $lb'$ and
    a. If the predecessor node of $lb'$ is not accessible from $lb$ by either,
        i. capacity constraints or
        ii. if the predecessor node of $lb'$ is already visited by $lb$
    b. If $lb$ is dominated by other labels that have different predecessor nodes than $lb'$.

If a label is dominated and not discarded then it becomes a weakly dominant label and it can only be extended to the intersection of the predecessors of the dominating labels which is represented by the vector $DYN_x[lb]$.

A label is said to be processed in the dynamic program if new labels are generated from this label. Among the unprocessed labels, the label with the minimum demand is chosen to be processed. This is named as the best label algorithm and it differs from the algorithm in the VRPTW since the unprocessed label with the minimum time is chosen in VRPTW.

For practical reasons all the labels are classified in terms of their current nodes and they are ordered lexicographically by their demands in an ascending fashion. Therefore, a it is impossible for a label with a higher label number to dominate a label with a lower number.

The code of the dynamic programming can be given in Fig. 4.2 as follows:

```
Bestn:=0;Bestc:=dual3[j]+dual4[j];Bestd:=0;

Besty:=1;cntbest:=0;start:=0;
forall(j2 in 1..N){Bestp[j2]:=1000;Bestx[j2]:=1000;}
forall(i in [1..N+M]){cntlb[i]:=0;}

while ((sum(i in [1..N+M])(cntlb[i])>0) ∨ start=0) do {start:=1;
  forall(i1 in 1..N+M:i1<=N  ∨  i1=j){
    if(i1<>Bestp[1..N] ∧  (Besty=2 ∧ i1=Bestx[1..N])then{
        if(Bestn<>i1 & Bestd+q[i1] <= C) then {
          cntlb[i1]:=cntlb[i1]+1;
New       //Place the label in an ascending order according to demands//
Labels    forall(ii1 in 1..cntlb[i1]:Bestd+q[i1]>=dynd[i1,ii1]){
             place:=ii1;break;}
          dynn[i1,place]:=i1;
          dync[i1,place]:=Bestc+d[Bestn,i1]-
                           dual[i1]+q[Bestn]*(VC[j]-dual7[j]);
          dynd[i1,place]:=Bestd+q[i1];
          dyny[i1,place]:=1;
          dynp[i1,place,1..N]:=Bestp[1..N] ∪ Bestn;
 }}}
   forall(i in [1..N+M]){
   forall(i10 in 1..cntlb[i]){
      forall(i11 in 1..cntlb[i]  ∧  i11<i10){
         if(dynn[i,i10]=dynn[i,i11]   ∧  dync[i,i10]<=dync[i,i10]  ∧
            dynd[i,i10]<=dynd[i,i10]) then{
            dyny[i,i11]:=2;
            if(dyny[i,i10]=2  ∨
Dominance      (dynp[i,i10,1..N] ∪ dynp[i,i11,1..N] = dynp[i,i10,1..N]  ∧
Criterion      dynp[i,i10,1..N] ∪ dynp[i,i11,1..N]=dynp[i,i10,1..N]))then{
               cntlb[i]:=cntlb[i]-1; //Discard label i11//
            else
               dynx[i,i11,1..N]:=dynp[i,i10,1..N]
 }}}}}
   forall(i20 in [1..N+M]){
      if(dynd[i20,1] = min(i21 in [1..N+M])(dynd[i21,1])) then {
         cntbest:=cntbest+1;Bestn[cntbest]:=i1;
         Bestc:=dync[i20,1];
Best     Bestd:=dynd[i20,1];
Label    Besty:=dyny[i20,1];
         Bestp[1..N]:=dynp[i20,1];
         Bestx[1..N]:=dynx[i20,1];
         cntlb[i20]:=cntlb[i20]-1; //Discard Label i20//
}}}
```

Figure 4.2: Dynamic Programming code for solving SPP

## *4.2 2-Path Cuts*

Finding and implementing valid inequalities reduce the solution time significantly, as in VRPTWDC. The 2- path cuts and subtour cuts are the most important strong valid inequalities. Description of the 2-path cuts can be found in detail in section 2.2.3. The separation algorithm for the WLRP is the same algorithm that is used for VRPTW and VRPTWDC. However, there is no need to solve TSP problem since there are no time windows; therefore only the capacity constraints remain. Thus, if sets satisfying $\sum_{i\in N/S}\sum_{j\in S} X_{ij} < 2$ are found and if the total demand of the customers of the set exceeds the vehicle capacity, then it is concluded that a 2-path cut is generated. After the 2-path cuts and the subtours are found the following constraints are added to the master problem:

$$\sum_{r=1}^{R} \eta_{s_1 r} XR_r \geq 1 \qquad \forall s_1 \in S_1 \tag{4.23}$$

$$\sum_{r=1}^{R} \eta_{s_2 r} XR_r \geq 2 \qquad \forall s_2 \in S_2 \tag{4.24}$$

where the set $S_1$ represents the routes that contribute to the subtour sets, $S_2$ represents the routes that contribute to the 2-path cut sets and $\eta_{sr}$ takes an integer value which specifies the number of arcs that enter the set $s$ in route *r*. For example, consider that the following set is found using the separation algorithm (without paying attention to the sequence of nodes):

$\{2 \quad 6 \quad 11 \quad 24 \quad 30 \quad 53\}$

Furthermore, consider these four routes with the given sequence of nodes:

| | |
|---|---|
| Route 186: | $[58 \quad 19 \quad 20 \quad 3 \quad 44 \quad 32 \quad 36 \quad 58]$ |
| Route 477: | $[59 \quad 16 \quad 24 \quad 28 \quad 59]$ |
| Route 510: | $[58 \quad 15 \quad 30 \quad 31 \quad 46 \quad 47 \quad 53 \quad 51 \quad 58]$ |
| Route 581: | $[61 \quad 37 \quad 11 \quad 2 \quad 49 \quad 21 \quad 23 \quad 61]$ |

Route 186 does not contain any of the nodes in the cut, therefore $\eta_{s186} = 0$. Route 477 contains only node 24 and therefore $\eta_{s477} = 1$. Route 510 contains nodes 30 and 53, and the arcs to both of these originate from nodes outside the set, for this reason, $\eta_{s510} = 2$. Finally, although the route 581 contains two of the nodes in the cut (2 and 11), there is only a single arc that originates from nodes outside the cut (from 37), hence $\eta_{s581} = 1$.

The cut generation is only performed in the root node for computational reasons and the following column generation iterations include the constraints (4.23) and (4.24). The dual variables associated with these constraints are $dual^{23}_{s_1}$ and $dual^{24}_{s_2}$. However, for the dynamic programming $\eta_{sr}$ cannot be used directly and therefore $\alpha_{gh}$ is defined as the following:

$$\alpha_{gh} = \sum_{s_1=1}^{S_1} \sum_{r=1,(g,h)\in sol_r}^{R} \eta_{s_1 r} \cdot dual^{23}_{s_1} + \sum_{s_2=1}^{S_2} \sum_{r=1,(g,h)\in sol_r}^{R} \eta_{s_2 r} \cdot dual^{24}_{s_2} \tag{4.25}$$

This means that if node *h* follows node *g* in any route *r*, then the sum of the dual variables of every cut that includes the route *r* becomes the dual variable that corresponds to $\alpha_{gh}$. Therefore, the dynamic programming changes and the calculation of the cost becomes:

$$\hat{d}_{gh} = d_{gh} - dual^{2}_{h} - \alpha_{gh} \tag{4.26}$$

## 4.3 Branch and Bound

The final step is the optimization of the MILP is the branch and bound algorithm as in VRPTW. The solution is usually fractional after the column generation and 2-path cuts. Thus, good branching decisions must be made. In this work, it is observed that branching priorities on the variables that model warehouse locations, namely the variables $YR_j$, proved to be successful. The branching variable is selected to eliminate the largest infeasibility, that is, the variable $YR_j$ that has the closest value to 0.5. After branching on the warehouse location variables, branching is done on the total number of vehicles is the most successful method. Therefore, two child nodes are created and the constraints

$$\sum_{r=1}^{R} XR_r = \lfloor k \rfloor \text{ and } \sum_{r=1}^{R} XR_r = \lceil k \rceil \tag{4.27}$$

are added to the two child nodes, respectively. As this constraint is added to the master problem, the dual variable associated with this constraint $dual^{27}$ is considered in the subproblem. Therefore the initialization becomes:

$$DYN_c[0] = dual_j^3 + dual_j^4 - dual^{27}, \tag{4.13'}$$

The new solution is then returned to the column generator and iterations continue until no more columns are generated. Finally, the branching decisions are specified on the flow variables. The logic behind the implementation of the constraints for these branches are similar to the 2-path cuts. When the current solution is examined there are several fractional arcs $(g,h)$ which are found by summing up the values of the routes that contain this arc. Since none of the routes contain a specific arc twice, the following constraint is added to the master problem:

$$\sum_{r=1,(g,h)\in solp[r]}^{R} XR_r = 1 \text{ and } \sum_{r=1,(g,h)\in solp[r]}^{R} XR_r = 0 \tag{4.28}$$

Two child nodes are obtained after adding these constraints. The dual variables associated with this constraint $dual^{28}$ is subtracted each time in the dynamic programming algorithm the label is extended from node *g* to node *h*.

Chapter 5

# RESULTS OF WLRP PROBLEMS

Although there are several studies on the WLRP, there are only a few benchmark problems. Perl[3] generated three problems and these are the only problems that are solved by the following studies. Thus, in this work these three problems are studied and the results are compared with the previous studies.

## 5.1 *Illustrative example*

As an illustrative example Perl's test problem with 12 customers and 2 possible warehouse sites is selected. The locations of the warehouses and the customers are shown in Fig. 5.1:

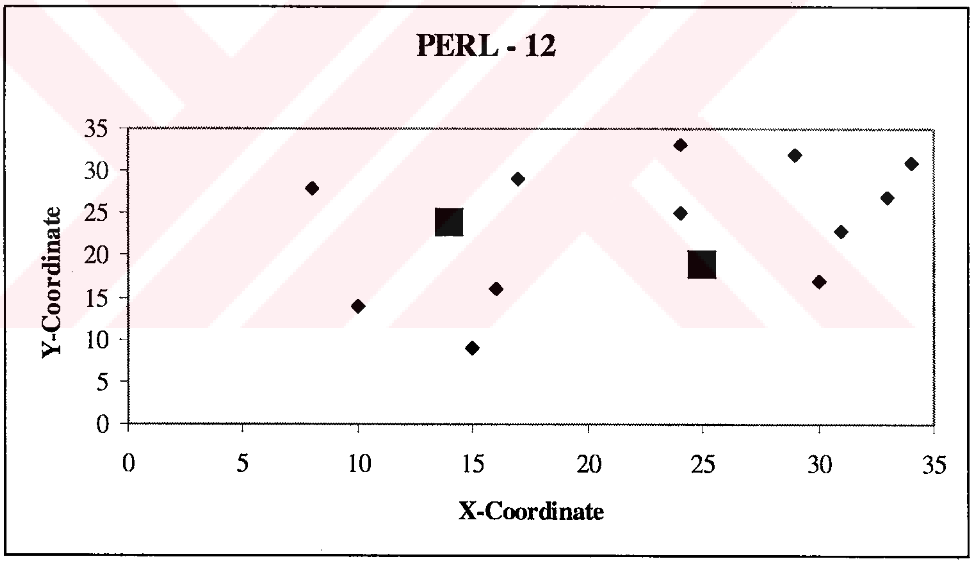


Figure 5.1: Location data for the small test problem Perl-12

The full data of this problem can be found in Appendix A. After the column generation at the root node the fractional solution of the problem can be represented in a flowchart as given in Fig. 5.2:

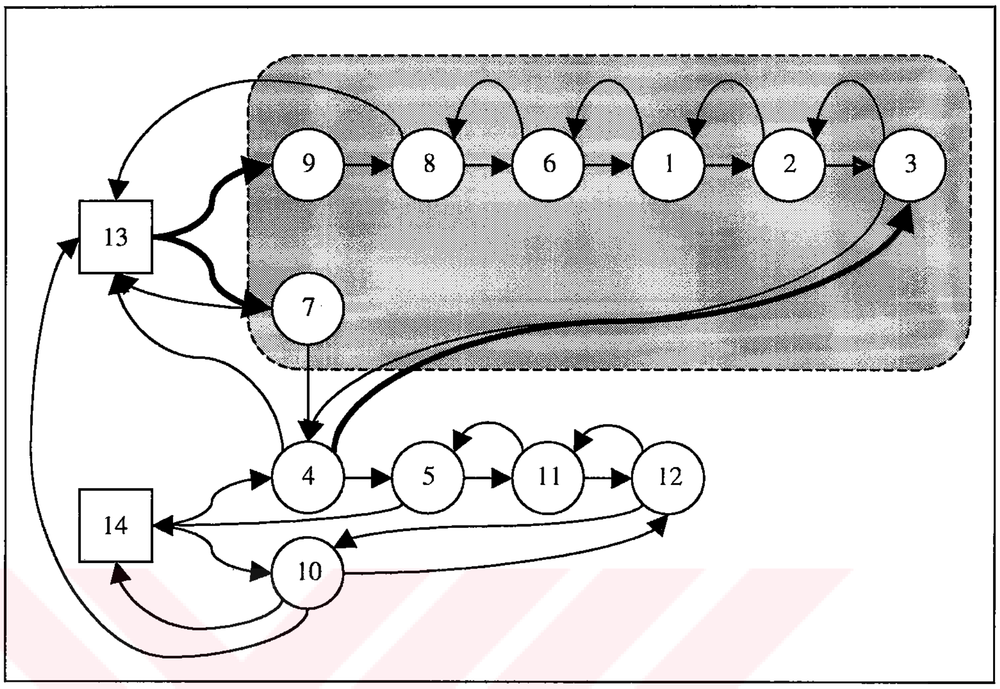


Figure 5.2: Solution obtained after column generation at the root node

The objective value of this solution is 351.5446. The first part of the algorithm ends after obtaining this result. The next step is to apply the separation algorithm to find subtours and 2-path cuts. The separation algorithm finds only one 2-path cut which is the set enclosed by the shaded rectangle shown in Fig. 5.2. There are three arcs entering the set and the sum of the fractional solutions of these arcs is less than 2. However, the sum of the demands of the nodes in the set exceeds the twice of the vehicle capacity. Therefore, this set requires more than 2 vehicles and the routes that contain any arc entering the set must be added in the cut with suitable coefficients. The values of these coefficients can be calculated using the knowledge in section 4.2.

After the cuts generation at the root node, branch and bound algorithm is employed. According to the branching priorities, first branching is done on the warehouses and $YR_{13} = 1$. Setting this variable, the problem is again solved using column generation and the optimal solution is found immediately after also setting $YR_{14} = 0$. The solution of the problem is given in Fig. 5.3:

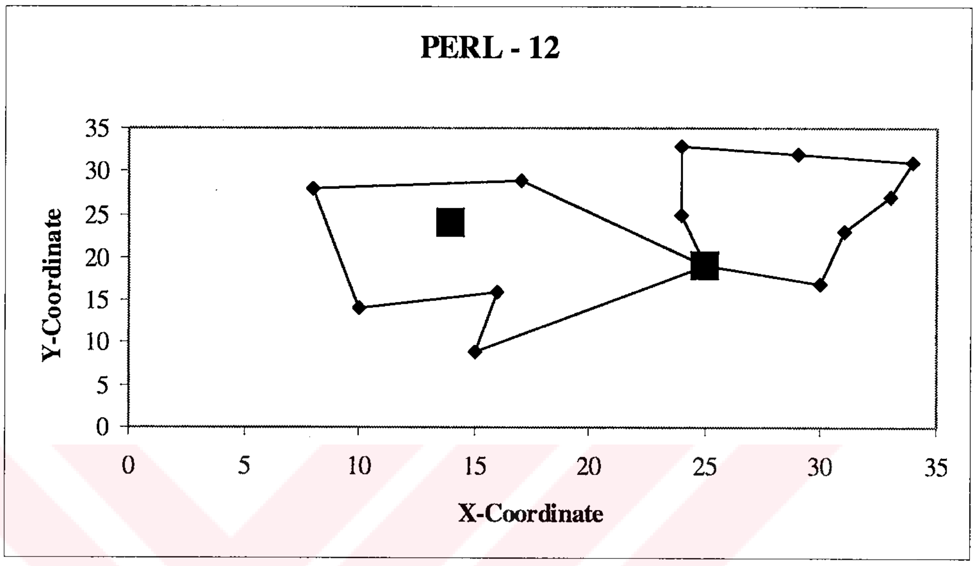


Figure 5.3: Integer solution to the small test problem Perl-12

According to this solution, only one of the warehouses is used with 2 routes. The nodes that are used in these routes are given in Table 5.2

## 5.2 *Results for benchmark Problems*

Results of the benchmark problems are given in Table 5.1. For the problem with 12 customers, the lower bound found at the root node is increased by finding a valid inequality. The solution is obtained at the first branching where two child nodes are obtained. The dynamic program is executed 14 times to obtain the solution. For the second problem, the fractional optimality gap is less than the first problem and 12 valid inequalities are generated at the root node. The optimal solution of the problem uses only three of the fifteen possible warehouse sites and the nodes that are used in the problem are shown in Table 5.3. The fractional optimality gap is the largest in the third problem and therefore, the problem can be solved after 76 nodes are generated in the branch and bound tree. The nodes that are used in the routes are shown in Fig. 5.4.

Table 5.1: Results for benchmark problems

| **Problem** | **LB$_1$** | **LB$_2$** | **IP** | **Opt gap** | **Wh** | **Veh** | **Node** | **VI** | **Sub** |
|---|---|---|---|---|---|---|---|---|---|
| Perl – 12 | 351.5446 | 352.3629 | 355.5825 | 0.020 | 1 | 2 | 2 | 1 | 14 |
| Perl – 55 | 5438.6451 | 5450.8122 | 5478.700 | 0.010 | 3 | 10 | 40 | 12 | 78 |
| Perl – 85 | 7422.2786 | 7448.1002 | 7517.6000 | 0.029 | 3 | 11 | 76 | 23 | 117 |

Table 5.2: Route information of the 12 customer problem

| Perl –12 | |
|---|---|
| Route 1 | 13-9-8-6-1-2-3-7-13 |
| Route 2 | 13-4-5-11-10-12-13 |

Table 5.3 Route information of the 55 customer problem

| Perl-55 | |
|---|---|
| Route 1 | 60-30-19-31-7-3-8-60 |
| Route 2 | 65-9-4-42-2-1-13-65 |
| Route 3 | 65-11-5-65 |
| Route 4 | 67-36-35-24-26-18-15-67 |
| Route 5 | 67-25-48-49-20-41-67 |
| Route 6 | 60-34-43-55-39-54-38-60 |
| Route 7 | 65-47-37-51-21-10-6-65 |
| Route 8 | 60-29-23-28-12-17-33-60 |
| Route 9 | 60-45-32-16-27-14-22-60 |
| Route 10 | 65-53-50-52-40-46-44-65 |

Table 5.4 Route information of the 85 customer problem

| Perl-85 | |
|---|---|
| Route 1 | 89-5-1-44-46-43-34-3-8-89 |
| Route 2 | 87-32-38-2-45-30-33-87 |
| Route 3 | 87-22-17-28-75-23-29-19-87 |
| Route 4 | 87-63-14-70-71-73-74-12-72-87 |
| Route 5 | 91-49-76-51-79-13-67-83-82-91 |
| Route 6 | 89-9-6-15-85-31-7-42-4-89 |
| Route 7 | 91-48-35-26-84-18-36-25-41-91 |
| Route 8 | 89-47-53-50-81-80-52-64-11-89 |
| Route 9 | 91-60-10-77-37-78-21-20-24-91 |
| Route 10 | 87-62-61-65-66-68-69-27-16-87 |
| Route 11 | 87-54-59-58-39-56-57-40-55-87 |

As it can be seen that the optimal values are found for each of these benchmark problems. There are no other benchmark problems that are solved by researchers and therefore, different location routing problems are generated in all studies usually using uniform distribution. Furthermore, these problems are listed without giving the detailed data of the problems. Since there is no basis for comparison no further problems are solved in this thesis. The previous results on these problems are as follows:

Table 5.5: Benchmark problems on WLRP

| | **Perl[3]** | | | **Hansen[5]** | | | **Wu[6]** | | | **Proposed** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Wh** | **Rt** | **Cost** | **Wh** | **Rt** | **Cost** | **Wh** | **Rt** | **Cost** | **Wh** | **Rt** | **Cost** |
| **Perl 12** | 1 | 2 | 355.58 | 1 | 2 | 355.58 | 1 | 2 | 355.58 | 1 | 2 | 355.58 |
| **Perl 55** | 3 | 10 | 5795.62 | 4 | 10 | 5617.67 | 3 | 10 | 5532.28 | 3 | 10 | 5517.25 |
| **Perl 85** | 3 | 11 | 7789.96 | 3 | 11 | 7551.61 | 3 | 12 | 7781.28 | 3 | 11 | 7517.6 |

For the 55 customer problem the optimal solution found in this thesis is lower than all the previous results. The optimal solution required 10 vehicles as in Wu's[6] work, however, the cost is lower due to the routing. The optimal solution is also less for the problem with 85 customers. The optimal solution contained only eleven routes as in Hansen's[5] work, however the routing algorithm in this thesis helps to find better routes than Hansen's[5] heuristics.

# Chapter 6 CONCLUSIONS

WLRP is a strategic level decision problem aiming to satisfy the demands of a given number of customers. The location decisions are as important as the routing decisions and thus, forced the researchers to emphasize on the integration of the two well known problems, the vehicle routing problem and the warehouse location problem. The need for exact methods expanded the scope of this thesis and the vehicle routing problem with time windows is studied. In this thesis there are significant contributions to VRPTW, VRPTWDC and WLRP. New formulations are presented. For all these problems Furthermore, the dynamic algorithm is modified for the VRPTW and VRPTWDC. The method that is used for the VRPTW and VRPTWDC is also applied to WLRP and important results are obtained.

## *6.1 Integer Formulations*

For both VRPTW and WLRP there are valuable studies and both of the problems are formulated as integer programs. Although it is known that these formulations cannot be solved directly by traditional mixed-integer programming algorithms such as the branch and bound, they require attention such that there are many redundant constraints and variables to be reconsidered. For both of the problems, the programs are changed from three-index formulation to two-index formulation. Logically this can be stated as in the previous works, the flow variables are represented as $X_{i_1 i_2 k}$, meaning that for each node $i_1$, the successor node $i_2$ and the vehicle k are known. Furthermore, it is also known from $X_{i_2 i_3 k}$ that the successor of node $i_2$ and the vehicle are known. However, this second variable is redundant in a way that if node $i_2$ follows $i_1$ and if node $i_1$ is assigned to vehicle $k$, then it is also known that $i_2$ is assigned to vehicle $k$. Thus, the formulation of $X_{i_1 i_2 k}$ is replaced with $X_{i_1 i_2}$, $Z_{i_1 k}$ and $Z_{i_2 k}$ (for WLRP these variables are $X_{i_1 i_2}$, $ZZ_{i_1 k}$ and $ZZ_{i_2 k}$). This replacement requires new constraints that tie these two integer variables which are also aggregated over the vehicles using propositional logic. Thus, the number of variables in each problem is reduced significantly and the number of constraints is also reduced by constraint aggregation.

Another variable $Y_{ij}$ is removed from the WLRP formulation, since the new variables $ZZ_{ik}$ are able carry the information of $Y_{ij}$.

Direct implementation of these formulations could only solve small problems, and the number of variables quickly enlarges as the problem size increases. Before applying the column generation algorithm, the WLRP is tried to be solved by relaxation of the formulation. This is done in an iterative manner such that first the problem is relaxed and then the separation algorithm for the 2-path cuts is run. Although there are several cuts that can be generated at the initial relaxation, the number of 2-path cuts and the subtour cuts quickly decreases in the next iterations. For these reasons, the bound of the problem does not rise significantly and the problem again becomes hard to solve.

## *6.2 Column Generation – The Master Problem*

The studies on the VRPTW started by examining an exact solution method, which is also named as the Dantzig-Wolfe decomposition technique. The column generation algorithm for VRPTW is examined in order to apply an exact algorithm for the WLRP. According to the Dantzig-Wolfe decomposition method the VRPTW is decomposed into two subproblems; the master problem is defined as a set partitioning problem which is an allocation type problem and the subproblem becomes the elementary shortest path problem with time windows. Although the master problem is defined as an integer program, the integrality constraints are relaxed and the set partitioning problem is solved as a linear problem.

The column generation algorithm is in the core of this exact algorithm since most of the computational resources are required in this step. The linear master problem is solved easily as a linear programming problem using CPLEX solver that is called from OPL Studio version 3.5. As the number of feasible routes increase the solution time required for the solution time of the master problem increases; however, it is almost always negligible compared to the column generation step.

The reduction in the computing time of the master problem is accomplished by removing some of the unused columns at every iteration as discussed in section 6.2.

The master problem for the VRPTW includes the constraints for the customers alone initially, whereas for the WLRP, the problem contains a different variable $YR_j$ and therefore the logical constraints that tie these variables with $XR_r$. Moreover, the WLRP contains the capacity constraints for the warehouses. The objective of the master problem is to generate the dual variables for each customer (and for each warehouse in the WLRP), so that the subproblem can be solved to find feasible routes.

There are also other constraints added to the problem as the procedure enters the cuts generation and branch and bound phases. The cuts and the branches are applied in special constraints that are explained in previous sections.

## *6.3 Column Generation - The Subproblem*

The subproblem is used for generating feasible columns, i.e. routes, by using the dual variables associated with the constraints in the master problem. The subproblem is defined as a SPP (for VRPTW the subproblem is SPPTW) and formulated as an integer program. The solution of the subproblem by the pure integer formulation is again NP-Hard and thus requires another exact method. Furthermore, the solution of the integer problem would give only the "best" route with the minimum reduced cost. However, usually a lot of "good" routes are required for fast convergence to the lower bound of the selected subspace. Therefore, a dynamic programming algorithm is used for the generation of routes.

The algorithm starts from the warehouse node and continues to the other nodes without violating the capacity constraints (for the VRPTW there also the time constraints that must be satisfied). This extension to the nodes is executed by using labels. Each label contains specific information on the current node. In the previous works the labels only contained information on the current node, current demand and the predecessor of the node. Moreover, in order to speed up the column generation, dominance rules are generated which eliminate 2-cycles only. That is, these rules do not allow the labels to return to their predecessors when certain conditions are satisfied. However, there is a possibility for $k \geq 3$ $k$-cycles, and the presence of these cycles decrease the lower bound. This reduction in the lower bound causes to have many nodes in the branch and bound at later phases of the algorithm.

In this thesis, not only the predecessor but all the preceding nodes are recorded with the same dominance rules. Thus, the sequence of the previous nodes is stated in a predecessor vector for each label. This application of the predecessor vector not only eliminates all $k$-cycles but also does not affect the computational complexity of the problem.

The solution of the subproblem may take a considerable computing time if certain parameters are not handled correctly. These parameters usually stop the program when certain conditions are satisfied.

### *6.3.1 The maxcol parameter*

There are many important issues that require attention in the dynamic programming algorithm. The algorithm is usually stopped before the best route is found. This is because usually a given number of columns are sufficient for improvement in the objective function of the master problem. For this application a parameter is used for the maximum number of columns generated, *maxcol.* In all problems *maxcol* is set to 20, and thus at the initial iterations of the column generation, the subproblems are solved very quickly. After certain iterations, the *maxcol* parameter no longer stops algorithm since there are usually less columns generated.

### *6.3.2 The maxbestlb parameter*

Furthermore, the generation of labels is stopped after a given number of best labels are generated. These best labels are the labels to be extended to the next nodes. This procedure is stated as the forced early stop in the previous works. The *maxbestlb* parameter is useless in the starting iterations of the column generation because the problem is usually stopped by the *maxcol* parameter. However, as the change in objective function value of the master problem starts to slow down, the *maxbestlb* parameter becomes important. However, there is no unique *maxbestlb* parameter that works well for each problem. As the problem size increases *maxbestlb* must also be increased proportionally.

However, at the final iteration, the *maxbestlb* parameter is always removed so that every feasible route is detected.

### *6.3.3 The remcol parameter*

Another important factor is the removal of columns. This is established in such a way that at each iteration of the column generation, the columns that are not used in the previous *remcol* iterations are removed. The reason for this is that many routes that are generated at the initial iterations, do not enter the basis in the latter iterations of the column generation. Moreover, each new route is checked with the previous routes in order to detect and eliminate if these routes are found before. Thus, by removing the columns that are not used in consecutively *remcol* iterations, the solution time of the dynamic programming decreases. Although some of the removed columns are regenerated at the later iterations, the number of such routes is negligible.

## *6.4 Cuts Generation*

Although, the new dynamic programming algorithm reduces the optimality gap for many problems, non-integer solutions are encountered at the root node of the branch and bound tree for some problems. Therefore, cuts are employed in order to reduce the branch and bound nodes. These cuts are detected using a separation algorithm that checks every possible set of nodes and tries to find the total flow into these node sets. If the total flow into any set is less than 1, then these sets are added to the master problem as subtour constraints. For 2-path cuts, the sets with a total inflow of less than 2 are required. However, these sets are only added if their TSP (for VRPTW, TSPTW is solved) problem is infeasible for a single vehicle. The addition of the cuts increases the lower bound significantly for most of the problems and the number of branch and bound nodes decrease.

## *6.5 Branch and Bound*

Branch and bound algorithm is applied for the problems that cannot be solved at the root node. The priorities set for the branching directions proved to have good results in the previous work by

Larsen[42]. The first priority is among the warehouse establishment variables, $YR_j$. Afterwards, the branching decisions are taken on the total number of vehicles and finally on the flow variables. Within each prioritized group, the branches are selected according to the best bound method. For the WLRP, branching on the warehouses affects the solution significantly, thus, this prioritization is quite efficient. The bound generated in the branch and bound are added to the master problem using the constraints explained in previous sections and the dual variables associated with each of these new constraints are handled in the dynamic algorithm of the subproblem.

## *6.6 Contributions*

The contributions to the problems can be listed.

i. New MILP models are formulated for VRPTW, VRPTWDC and WLRP
   a. Variable and constraint redundancies are eliminated
   b. Two-index variables are used
ii. New dynamic programming algorithm for VRPTW is proposed
   a. Predecessor nodes are stored in arrays
   b. *k*-cycles are eliminated
iii. A set partitioning problem is formulated for VRPTWDC
   a. Costs include the fixed costs of vehicles
iv. New dynamic programming algorithm for VRPTWDC is proposed
   a. Fixed costs of vehicles are included in cost calculations
   b. Predecessor nodes are stored in arrays
   c. *k*-cycles are eliminated
   d. Executed for every vehicle type
v. Branching priorities are specified for VRPTWDC
vi. A new set partitioning problem is formulated for WLRP
   a. Warehouse establishment variables are modeled
   b. Warehouse capacity constraints are modeled

   c. Variable and fixed costs of establishing warehouses are considered in the objective function

vii. A new dynamic programming algorithm for WLRP is proposed

   a. Fixed and variable costs of warehouses are considered in cost calculations
   b. Dual variables for the constraints in the master problem are handled
   c. Predecessor nodes are stored in arrays
   d. *k*-cycles are eliminated

viii. Branching priorities are specified for WLRP

## *6.7 Future Work*

The proposed method for the WLRP is quite effective and guarantees optimality. However, the computational complexities faced in the column generation may sometimes lead to changes in the parameters.

Furthermore, the WLRP can be enriched with several other side constraints. One such constraint would be the addition of maximum route length. This constraint can be handled in the dynamic programming algorithm and, indeed, it can reduce the computation time since it forces the program to stop at a time limit.

The WLRP proposed in this problem considers only the warehouses and the customers. This two-level supply chain can be enlarged to a three-level supply chain by adding the locations of the plants. This problem can decide on which warehouse to be supplied by which plant and thus, gains a more important role in the strategic level decisions.

The method for the VRPTW is solved using parallelization techniques in Larsen[42]. This property in VRPTW can also be applied to WLRP. This is an important property since the dynamic programming algorithm can easily solved as independent programs. In the proposed method the dynamic programming is solved for each warehouse (and if the vehicle types would be different, for each vehicle type). Thus, at each iteration of the column generation several independent problems need to be solved. Each problem can be solved by a computer and their results can be collected in a master computer at each iteration. The problem faced with this method is that quickly one of the slave computers become bottleneck and this is an area of future progress.

# BIBLIOGRAPHY

[1] http://www.CLM1.org , *Council of Logistics Management*, Presented at the annual business meeting, Anaheim, CA, October, 1998.

[2] S. Ghosh and S. Raychaudhuri, “Decision Support System for Production and Distribution of Fertilizers,” *PhD Dissertation, Indian Institute of Technology,* Kharagpur, 2002.

[3] J. Perl, “A unified warehouse location-routing analysis,” *UMI Dissertation Information Service*, June 1983.

[4] J. Perl and M.S. Daskin, “A warehouse location routing problem,” *Transportation Research Quarterly*, vol. 19B, pp. 381-396, 1985

[5] P.H. Hansen, B. Hegedahl, S. Hjortkjær and B. Obel, “A heuristic solution to the warehouse location-routing problem,” *European Journal of Operational Research*, vol. 76, pp. 111-127, 1994.

[6] T.-H. Wu, C. Low and J.-W. Bai, “Heuristic solutions to multi-depot location-routing problems,” *Computers & Operations Research*, vol. 29, pp. 1393-1415, 2002.

[7] R.C. Burness and J.A. White, “The traveling salesman location problem,” *Transportation Science*, vol. 10, no. 4, pp. 348-360, 1976..

[8] I. Or, and W.P. Pierskalla, “A transportation location-allocation model for regional blood banking,” *AIIE Transactions*, vol. 11, pp. 86-95, 1979

[9] S.K. Jacobsen and O.B. Madsen, “A comparative study of heuristics for a two-level routing-location problem,” *European Journal of Operational Research*, vol. 5, pp. 378-387, 1980.

[10] G. Laporte and Y. Nobert, “An exact algorithm for minimizing routing and operating costs in depot location,” *European Journal of Operational Research*, vol. 6, pp. 224-226, 1981.

[11] J.M. Nambiar, L.F. Gelders and L.N. Van Wassenhove, "A large scale location-allocation problem in natural rubber industry," *European Journal of Operational Research*, vol. 6, pp. 183-189, 1981.

[12] J.M. Nambiar, L.F. Gelders and L.N. Van Wassenhove, "Plant location and vehicle routing in the Malaysian rubber smallholder sector: A case study," *European Journal of Operational Research*, vol. 38, pp. 14-26, 1989.

[13] G. Laporte, Y. Nobert and P. Pelletier, "Hamiltonian location problems," *European Journal of Operational Research*, vol. 12, pp. 82-89, 1983.

[14] O.B.G. Madsen, "Methods for solving combined two-level location-routing problems of realistic dimensions," *European Journal of Operational Research*, vol. 12, pp. 295-301, 1983.

[15] H. Min, V. Jayaraman and R. Srivastava, "Combined location routing problems: A synthesis and future research directions," *European Journal of Operational Research*, vol. 108, pp. 1-15, 1998.

[16] Z. Drezner, G. Steiner and G.O. Wesolowsky, "One-facility location with rectilinear tour distances," *Naval Research Logistics Quarterly*, vol. 32, pp. 391-405, 1985.

[17] J.H. Bookbinder and K.E. Reece, "Vehicle routing considerations in distribution system design," *European Journal of Operational Research*, vol. 37, pp. 204-213, 1988.

[18] G. Laporte, Y. Nobert and S. Taillefer, "Solving a family of multi-depot vehicle routing and location-routing problems," *Transportation Science*, vol. 22, no. 3, pp. 161-172, August, 1988.

[19] G. Laporte, F. Louveaux and H. Mercure, "Models and exact solutions for a class of stochastic location-routing problems," *European Journal of Operational Research*, vol. 39, pp. 71-78, 1989.

[20] Y. Chan, W.B. Carter and M.D. Burness, "A multiple-depot, multiple-vehicle, location routing problem with stochastically processed demands," *Computers & Operations Research*, vol. 28(8), pp. 803-826, 2001.

[21] R. Srivastava and W.C. Benton, "The location-routing problem: Considerations in physical distribution system design," *Computers & Operations Research*, vol. 17, no. 5, pp. 427-435, 1990.

[22] C. Revelle, J. Cohon and D. Shobrys, "Simultaneous siting and routing in the disposal of hazardous wastes," *Transportation Science*, vol. 25, no. 2, pp. 138-145, May, 1991.

[23] R. Srivastava, "Alternate solution procedures for the location-routing problem," *Omega Int. J. of Mgmt. Sci.*, vol. 21, no. 4, pp. 497-506, 1993.

[24] D. Tuzun and L.I. Burke, "A two-phase tabu search approach to the location routing problem," *European Journal of Operational Research*, vol. 116, pp. 87-99, 1999.

[25] Z. Drezner, "Facility location: A survey of applications and methods," *Springer Series in Operations Research*, New York, 1995.

[26] K.C. Tan, L.H. Lee, Q.L. Zhu and K. Ou, "Heuristic methods for vehicle routing problem with time windows," *Artificial Intelligence in Engineering*, vol. 15, pp. 281-295, 2001.

[27] M. M. Solomon, "Algorithms for vehicle routing and scheduling problems with time window constraints," *Operational Research*, vol. 35(2), pp. 254-265, 1987.

[28] A.C. Hax and D. Candea, *Production and Inventory Management*, Prentice-Hall, Englewood Cliffs, NJ, 1984.

[29] M. Gendreau, G. Laporte and J.-Y. Potvin, "Metaheuristics for the capacitated VRP," *in The Vehicle Routing Problem by P. Toth and D. Vigo*, pp. 129-154, Siam Monographs on Discrete Mathematics and Applications, Bologna, Italy, 2001.

[30] I.H. Osman, "Vehicle routing and scheduling: application, algorithms and developments," *Proceedings of the International Conference on Industrial Logistics*, Rennet, France, 1993.

[31] G. Laporte, "The vehicle routing problem: an overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, pp. 345 –358, 1992.

[32] L. Bodin, B. Golden, A. Assay and M. Ball, "The state of the art in the routing and scheduling of vehicles and crews," *Computers & Operations Research,* vol. 10 (2), pp. 63 –211, 1983.

[33] G. Kontoravdis, J.F. Bard, "Improved heuristics for the vehicle routing problem with time windows," *INFORMS Journal on Computing*, 1994.

[34] N. Kohl, O.B.G. Madsen, "An optimization algorithm for the vehicle routing problem with time windows based on Lagrangean relaxation," *Operations Research*, vol. 45 (3), pp. 395 –406, 1997.

[35] P.M. Thompson and H. Psaraftis, "Cyclic transfer algorithms for multi-vehicle routing and scheduling problems," *Operations Research*, vol. 41, pp. 935 –946, 1993.

[36] A.W.J. Kolen, A.H.G. Reno Kaman and H.W.J.M. Trienekens "Vehicle routing with time windows," *Operations Research*, vol. 35 (2), pp. 266 –331 , 1987.

[37] H.C. Lau, M. Sims and K.M. Toe, "Vehicle routing problem with time windows and a limited number of vehicles," *European Journal of Operational Research*, vol. 148, pp. 559-569, 2003.

[38] M. Desrochers, J. Desrosiers and M. Solomon, "A new optimization algorithm for the vehicle routing problem with time windows", *Operations Research*, vol. 40(2), pp. 340-354, 1992.

[39] K. Halse, "Modeling and solving complex vehicle routing problems," *PhD dissertation no:60, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark*, DK-2800 Lyngby, Denmark, 1992.

[40] N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon and F. Soumis, "2-Path cuts for the vehicle routing problem with time windows", *Transportation Science*, vol. 33, no. 1, pp. 101-116, February, 1999.

[41] J.L. Rich, "A computational study of vehicle routing applications", *PhD. Thesis, Institute of Computational and Applied Mathematics, Rice University*, Houston, Texas, 1999.

[42] J. Larsen, "Parallelization of the vehicle routing problem with time windows," *PhD dissertation no:62, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark*, Lyngby, 1999.

[43] Irnich S., Villeneuve D. "The shortest path problem with k-cycle elimination $(k \geq 3)$: Improving a branch-and-price algorithm for the VRPTW," working paper, *Rheinisch-Westfälische Technische Hochschule (RWTH)*, Aachen, Germany, 2003

[44] M.S. Bazaraa, J.J. Jarvis and H.D. Sherali, "Linear programming and network flows," John Wiley & Sons, second edition, pp. 320-370, United States of America, 1990.

[45] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon and F. Soumis, "VRP with time windows," *in The Vehicle Routing Problem by P. Toth and D. Vigo*, pp. 157-194, Siam Monographs on Discrete Mathematics and Applications, Bologna, Italy, 2001.

[46] G.L. Nemhauser and L.A. Wolsey, "Integer and combinatorial optimization," *Wiley Interscience Series in Discrete Mathematics and Optimization*, United States of America, 1988.

[47] J. Desrosiers, Y. Dumas, M. Solomon, F. Soumis , "Time Constrained Routing and Scheduling," *in Handbooks in Operations Research and Management Science 8: Network Routing, M.O. Ball, T.L. Magnanti, C.L. Monma, G. L. Nemhauser*, pp. 35 – 139, North Holland, Amsterdam, The Netherlands, 1995.

APPENDIX A

# DETAILED DATA FOR ILLUSTRATIVE EXAMPLES

Table A.1: Customer data for R205

| Cust. No | Xcoord | Ycoord | Demand | Ready Time | Due Time | Service Time |
|---|---|---|---|---|---|---|
| 0 | 35 | 35 | 0 | 0 | 1000 | 0 |
| 1 | 41 | 49 | 10 | 658 | 898 | 10 |
| 2 | 35 | 17 | 7 | 93 | 333 | 10 |
| 3 | 55 | 45 | 13 | 436 | 676 | 10 |
| 4 | 55 | 20 | 19 | 620 | 860 | 10 |
| 5 | 15 | 30 | 26 | 20 | 260 | 10 |
| 6 | 25 | 30 | 3 | 345 | 585 | 10 |
| 7 | 20 | 50 | 5 | 251 | 491 | 10 |
| 8 | 10 | 43 | 9 | 323 | 563 | 10 |
| 9 | 55 | 60 | 16 | 329 | 569 | 10 |
| 10 | 30 | 60 | 16 | 485 | 725 | 10 |
| 11 | 20 | 65 | 12 | 146 | 386 | 10 |
| 12 | 50 | 35 | 19 | 167 | 407 | 10 |
| 13 | 30 | 25 | 23 | 639 | 879 | 10 |
| 14 | 15 | 10 | 20 | 32 | 272 | 10 |
| 15 | 30 | 5 | 8 | 118 | 358 | 10 |
| 16 | 10 | 20 | 19 | 203 | 443 | 10 |
| 17 | 5 | 30 | 2 | 682 | 922 | 10 |
| 18 | 20 | 40 | 12 | 286 | 526 | 10 |
| 19 | 15 | 60 | 17 | 204 | 444 | 10 |
| 20 | 45 | 65 | 9 | 504 | 744 | 10 |
| 21 | 45 | 20 | 11 | 153 | 393 | 10 |
| 22 | 45 | 10 | 18 | 332 | 572 | 10 |
| 23 | 55 | 5 | 29 | 146 | 386 | 10 |
| 24 | 65 | 35 | 3 | 656 | 896 | 10 |
| 25 | 65 | 20 | 6 | 716 | 956 | 10 |

Table A.2: Customer data for the small test problem Perl-12

| Cust. No | Xcoord | Ycoord | Demand |
|---|---|---|---|
| 1 | 34 | 31 | 20 |
| 2 | 29 | 32 | 20 |
| 3 | 24 | 33 | 20 |
| 4 | 17 | 29 | 20 |
| 5 | 8 | 28 | 20 |
| 6 | 33 | 27 | 20 |
| 7 | 24 | 25 | 20 |
| 8 | 31 | 23 | 20 |
| 9 | 30 | 17 | 20 |
| 10 | 16 | 16 | 20 |
| 11 | 10 | 14 | 20 |
| 12 | 15 | 9 | 20 |

Table A.3: Warehouse data for the small test problem Perl-12

| WH No. | Xcoord | Ycoord | Capacity | Fixed Cost | Variable Cost |
|---|---|---|---|---|---|
| 13 | 25 | 19 | 280 | 100 | 0.74 |
| 14 | 14 | 24 | 280 | 100 | 0.74 |

Table A.4: Other parameters for the small test problem Perl-12

| | |
|---|---|
| **Vehicle Capacity** | 140 |
| **Cost per mile** | 0.75 |