Abstract:

Labiad, Noureddine: Scheduling and routing of vehicles for a Transportation company. (Under the direction of Professor Russell E King)

This work addresses the problem of finding the minimum number of vehicles required to service weekly a set of customers subject to time windows. The specific structure of the problem requires that a full loaded vehicle serve only one customer each time it leaves the depot. As a consequence, all feasible schedules involve the same total distance traveled. The fleet is homogeneous and is located in a common depot. Vehicle capacity is finite, weekly mileage is limited, and split service is not permitted.

Four heuristics are developed to obtain feasible solutions. Results are reported for 1000 randomly generated problems with up to 150 deliveries. It is shown that the new heuristics outperform the actual algorithm used by the Transportation Company in terms of computation time and the quality of solution.

To gauge the quality of the solutions, three lower bounding procedures are developed. The first considers the 'bin packing aspect of the problem' with regard to the maximum weekly mileage. The second exploits the time windows constraints while the third lower bounding method uses a network flow formulation.

SCHEDULING AND ROUTING OF VEHICLES FOR A TRANSPORTATION COMPANY

By

NOUREDDINE LABIAD

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

INDUSTRIAL ENGINEERING

Raleigh

2002

APPROVED BY:

[illegible]

[illegible]

Chair of Advisory Committee

[illegible]

Co-Chair

Biography

Noureddine Labiad was born on August 3rd, 1975 in Casablanca, Morocco. He completed his undergraduate degree in Electrical Engineering from Ecole Mohammadia des Ingénieurs, Rabat, Morocco.

Right after he graduated, he started his career at La Banque Centrale Populaire as a Business Analyst. After a working experience of 18 months, he joined the multinational Corporation Italcementi Group as a Project Engineer. This last experience grew up his interest in Production and Manufacturing Engineering and his will to pursue his graduate studies in this area.

In 2001, he joined North Carolina State University to pursue his Masters degree in Industrial Engineering. He worked as a Teaching Assistant and Research Assistant and was elected to the Honor Society of Phi Kappa Phi. During his studies he worked as Industrial Engineering intern with ABB Inc., Raleigh, NC.

Acknowledgment

I would like to express my deepest appreciation to my advisor Dr. Russell E King for his invaluable guidance and help throughout all my thesis work at North Carolina State University. I would like to thank him for his continuous support, advice and guidance during my Master's research.

I would like also to thank my committee members, Dr Hodgson and Dr Chao for their constructive comments.

My special thanks to my parents, my brothers and sister for their endless support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1. Background

The Vehicle Routing Problem has been addressed by many researchers since the 1950's and continues to be an active research area. From the theoretical standpoint, it contains some of the most challenging problems in combinatorial optimization. From the practical viewpoint, there are still potential cost savings to be achieved across land, air and water transportation.

The VRP is among the most difficult problems in the area of logistics. The ability to understand the structure of the problem and develop optimal or near–optimal solutions remains limited especially for large-scale problems.

Operation researchers have investigated different variants of the Vehicle Routing Problem and many solution methods have been proposed to solve them. These methods range from heuristics algorithms, to optimization algorithms and metaheuristics methods to mathematical based heuristics.

In the next section the problem statement is discussed. Literature review and organization of the thesis are treated in the last section.

1.2. Statement of the problem

The purpose of this work is to propose weekly truck routes and schedules for a transportation company "TLS". In what follows we focus on the services provided to one of their major customers.

The basic need of this customer consists of shipping, on a weekly basis, products from its National Distribution Center (NDC) to different Forward Distribution Centers (FDCs). Each FDC delivery (henceforth referred to as a job) requires a full truckload shipment such that a truck can serve one and only one FDC in each round trip from the NDC. A given FDC may require more than one delivery per week. Based upon the distances involved, an assumed travel speed of the trucks, and given load and unload times, the processing time for each job can be determined. The processing time is used to define relative start and end times for the delivery.

We classify the jobs in view of the nature of the delivery due date into two categories:

- Fixed Jobs: jobs whose delivery must be made at a specific time on a specified day of the week.
- Time window jobs: jobs whose delivery can be made anytime within a defined time window during the week.

As the trucks are fully unloaded once at the FDC, it is not financially attractive to return to the NDC with the trailers empty. The following options may then be considered:

- Totes used to transport goods to the FDC must be transported from the FDC back to the NDC when sufficient quantities have accumulated.
- Inbound material from the customer's vendors may be picked up by a truck during its return trip to the NDC (referred to as a back haul).
- A mixed strategy may be adopted.

Back hauls are by far the most attractive option since they use the slack time between successive deliveries yielding an efficient use of the trucks and the elimination of deadhead miles. This work will address only the Vehicle Routing Problem With Time Windows. The Backhaul Scenario will be addressed in future research.

In an attempt to improve the actual scheduling system, we consider the following objectives:

1. minimize the number of tractors and trailers needed to deliver the customer products from the NDC to the FDC(s). A weekly route that is assigned to a vehicle consists of a number of legs. Each leg is a round trip between the NDC and an FDC.
2. uniformly utilize the tractors during the week yielding acceptable weekly mileage for every driver since their pay is based upon miles driven.

The development of the schedule should comply with the following constraints:

1. The tractors cannot run more than 5,500 miles per week, and
2. Trailer loads should not exceed 42,500lb.

1.3. Literature Review

Survey papers on heuristics and exact methods for Vehicle Routing Problems have been published by Bodin (1983), Golden and Assad (1986), Laporte and Nobert (1987), Desrochers (1988), Fisher (1995), and Laporte *et al.* (2000).

The Vehicle Routing Problem with Time Windows is the one that has most resemblance to our problem. The VRPTW is a NP-Hard Problem. The computational challenge involved makes it difficult, even impossible for large scale problems, to solve this type of problem to optimality in a timely way.

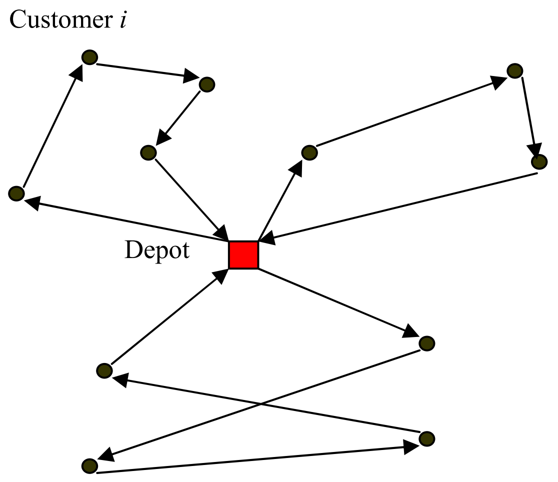


Figure 1: Spatial Representation of VRPTW routes

The Single Depot Vehicle Routing Problem with Time Windows (Figure 1) may be defined as a directed graph. Let V be the set of nodes and A is the set of all feasible arcs. A time window $[a_i, b_i]$ is associated with each node where a_i is the earliest feasible start time and b_i the latest completion time. Each arc $(i, j) \in A$ is characterized by a real cost c_{ij} and a positive duration t_{ij}, and it satisfies the feasibility condition:

$$a_i + t_{ij} \leq b_j$$

It is assumed that the service time at node i is included in the time values t_{ij} and that waiting is allowed before the start of a time window. The set of nodes V is composed of $N \cup \{o, d\}$, where N consists of nodes that can be visited on paths originating at the source node o and finishing at the sink node d.

Early efforts on VRPTW were directed at the development and analysis of heuristics capable of solving realistic size problems. A number of route construction and improvement procedures have been proposed.

Route construction algorithms build feasible routes by inserting unrouted customers into current partial routes. Difference is made at this point between sequential methods and parallel methods where several routes are built simultaneously. These algorithms have been applied first to VRP then generalized to VRPTW (Solomon, 1987).

Route improvement routines can be described in terms of Lin's and Kernighan (1973) *k*-interchanges mechanism, where at most *k* edges are exchanged between pairs of routes sets. Large number of exchange procedures has also been proposed by several authors (Osman, 1993; Taillard, 1993, Baker and Schaffer, 1986).

Another class of heuristics is the cluster first, route second algorithms, that solve a generalized assignment problem (Fisher and Jaikumar, 1981). This approach was extended by (Koskosidis, Powell and Solomon, 1992).

The 1990s have seen research focusing on applying general-purpose metaheuristics to the VRP (including simulated annealing, genetic algorithms, neural networks and tabu search). Traditional local improvement methods explore the neighborhood of a current solution and only select new solutions that strictly decrease the total distance. Tabu search and simulated annealing can select new solutions that increase the total distance and can avoid being trapped by poor local optima.

Researchers have proposed many implementations for metaheuristics. Tabu search and simulated annealing have been the most widely implemented. Without being exhaustive we list for simulated annealing: Osman, (1993), and for Tabu search: Osman, (1993), Potvin *et al., (*1996) and Cordeau and Laporte, (2001).

Exact methods that have been used to solve VRPTW problems are based on three techniques: set partitioning, lagrangian relaxation, or cutting planes. In the set partitioning formulation, the objective is to select a minimum cost set of feasible routes such that the customer is included in some route. The solving of this formulation uses the column generation approach where a fraction of the feasible routes is enumerated, then the linear relaxation with this partial route set is solved. The solution found for the relaxed problem is then used as a lower bound for the original problem. Desrochers *et al.*

(1992) were able to solve optimally 100-customer problems using the column generation approach combined with a branch and bound scheme to solve the integer set partitioning formulation.

The cutting plane method generates a set of constraints that are appended to the original problem. Then, the relaxed MILP with the added constraints is solved. If the solution is integer the algorithm terminates with the optimal solution to the original problem. If the solution is fractional, new constraints are added and the procedure repeated. The solution found can then be used a lower bound in a branch and bound scheme. Heuristics developed by Bard *et al.* (2002), were able to solve 50 and 100 customer problems using a branch-and-cut procedure. The objective was to minimize the number of tours needed to reach all customers within the time windows.

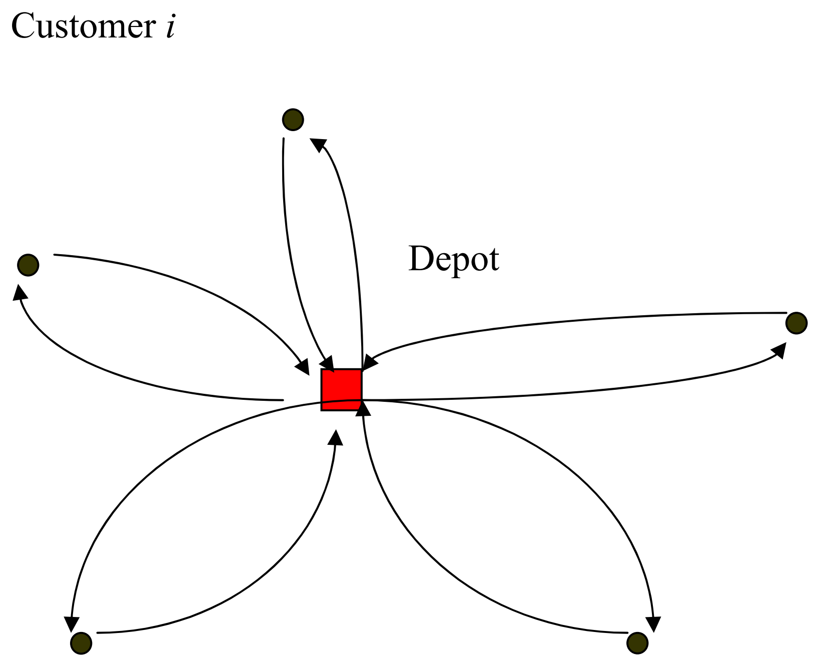


Figure 2: Spatial Representation of The thesis Problem routes

The problem that we are considering in this work (Figure: 2) was first addressed by Bombien (2001). It is related to some extent to the Vehicle Routing Problems that have been researched so far but it also has unique characteristics. The objective usually addressed in the literature is the total cost or distance. In our case this objective is insignificant, as the total mileage is the same for any feasible solution. Instead, we are seeking to minimize the number of vehicles. The other particularity of the problem is that the definition of a route is different from the classical one. As the customer demand is important compared to the vehicles capacity, the vehicle needs to make many round-trips between the depot to load the products and every single customer to unload them before serving any new customer. A route in this sense may correspond to many round-trips between the depot and the customers.

Chapter 2 Mathematical Model

In this chapter, we present a Mixed Integer Linear Program that models the problem.

2.1. Mixed Integer Linear Program

Let every round trip between an FDC and the NDC, which will be designated as a job in the remaining of this work, be denoted by a number $i \in V=\{1 \ldots n\}$. Let P_i be the time needed to perform round trip *I*, i.e. process job i. This time includes the round trip travel time (RT_i), the loading time at the NDC (LO_i), the unloading time at the FDC (UL_i) and the turnaround time (TA_i), i.e.

$$P_i = RT_i + LO_i + UL_i + TA_i \,.$$

For each job *i*, let $[a_i , b_i]$ be its time window. If a vehicle arrives to a node before the FDC operating window, it has to wait.

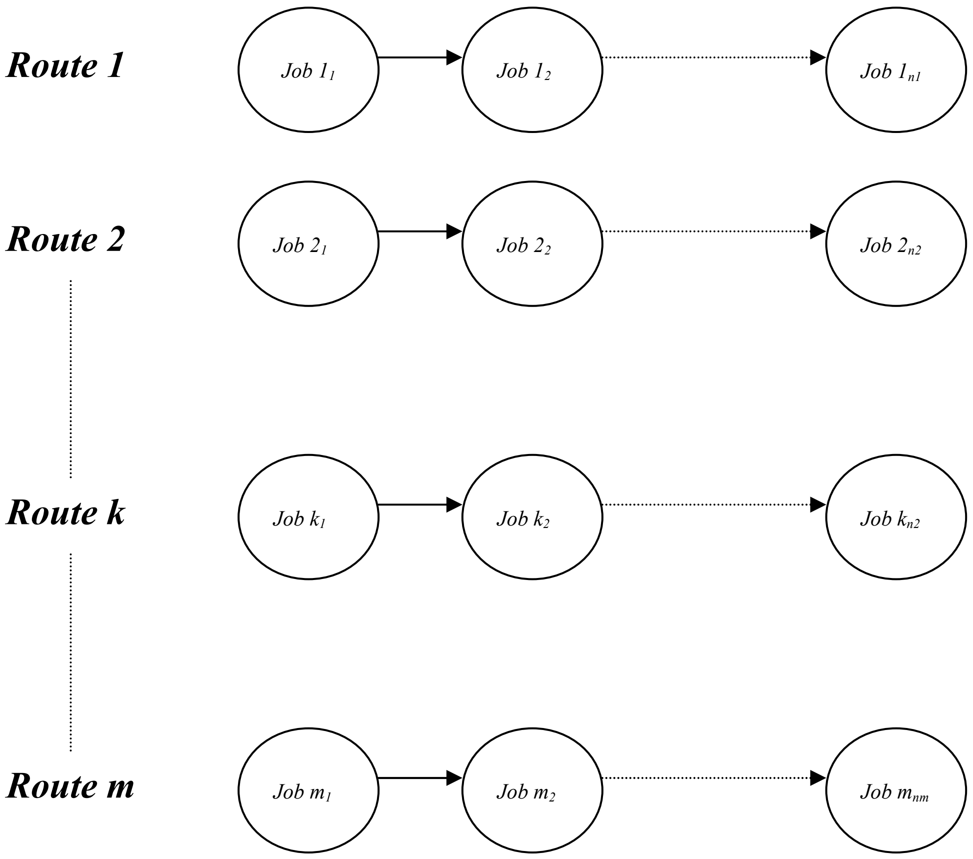


Figure 3: Representation of a feasible solution

This problem involves the following binary variables:

y_{ik}	$= 1$	if the i^{th} job is assigned to vehicle k
	$= 0$	otherwise
y_k	$= 1$	if vehicle k is used
	$= 0$	otherwise
δ_{ijk}	$= 1$	if job i is scheduled before job j and both are serviced by vehicle k
	$= 0$	otherwise

In addition, T_i represents the starting time of job $i \in V$.

Formally, we are seeking to:

$$Min \sum_{k=1}^{n} y_k$$

Subject to

$$\sum_{i=1}^{n} y_{ik} \le M y_k \quad k \in \{1...n\} \tag{1}$$

$$a_i \le T_i \le b_i \quad \forall i \in V \tag{2}$$

$$T_i + P_i - T_j \le (1-\delta_{ijk})M \quad \forall i,j \in V, k \in \{1 \ldots n\} \tag{3}$$

$$T_j + P_j - T_i \le \delta_{ijk} w + (1-\delta_{ijk})M \quad \forall i,j \in V, k \in \{1 \ldots n\}, w = 1 week \tag{4}$$

$$\sum_{i=1}^{n} RT_i y_{ik} \le \frac{L_{\max}}{V_{vehicle}} \quad \forall k \in \{1 \ldots n\} \tag{5}$$

$$\sum_{k=1}^{n} (\delta_{ijk} + \delta_{jik}) = \left\lfloor \frac{y_{ik} + y_{jk}}{2} \right\rfloor \quad \forall (i,j) \in V \ ; \ i < j \tag{6}$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x.

The first constraints (1) are linking constraints between the assignment variables y_{ik} and the variables y_k. If M is large then as long as at least one of the jobs is assigned to the vehicle k, y_k must be equal to one to ensure feasibility. The constraints (2) ensure that every job i starts within its time window. The constraints (3) express that in order to schedule a job i for service before another job j by a vehicle k, the completion time of the first is at maximum the start time of the second. Constraints (4) guarantee that the route length of every vehicle is less than one week. The constraint set (5) $L_{\max}$ is the maximum allowed weekly mileage and $V_{vehicle}$ the speed of the vehicles, thus their ratio is the maximum time per week that a vehicle can travel. Therefore, constraint set (5) ensures that route time for each vehicle does not exceed this time. Finally, constraint set (6) links the precedence constraints with the assignment constraints.

2.2. Computational complexity

The computational burden of the mixed integer programming formulation for problems with more jobs is evident from the fact that indices are function of cubic number of jobs $O(n^3)$.

By analyzing and adding up the set of constraints we can arrive at the following expression:

$$\textbf{Number of constraints} = 2n^3 + \frac{n^2}{2} + 5\frac{n}{2} = n\,(1) + n\,(2) + n^3\,(3) + n^3\,(4) + n\,(5) + \frac{n(n-1)}{2}\,(6)$$

It can also be shown that the number of mixed variables is

$$\textbf{Number of variables} = n^3 + n^2 + 2n = n^2 + n + n^3 + n$$

The following table gives an idea of the numbers discussed:

Number of jobs	Number of Constraints	Number of Variables
10	2,075	1,120
20	16,250	8,440
40	128,900	65,680
60	433,950	219,720
80	1,027,400	518,560
100	2,005,250	1,010,200

Table 1: Computational complexity for different problem sizes

At the present time with the computational facilities at our disposal, it is infeasible to solve real size problems (80-150 jobs) with the mixed integer linear programming formulation. Industrial Lingo has a limitation of 16,000 constraints and 3,200 variables.

In the next chapter, different heuristics approaches are proposed to solve our problem. Then, in the following chapter the performance of these heuristics is tested on randomly generated benchmark problems.

Chapter 3 Approach

In this section, four heuristics solutions procedures are defined. The following notation is used in the description of the heuristics.

$\boldsymbol{U}$ = Set of unrouted jobs

$\boldsymbol{a_u}$ = Earliest start time of job u

$\boldsymbol{b_u}$ = Latest start time of job u

$\boldsymbol{C_{ki}}$ = Completion time of the i^{th} job in route k

$\boldsymbol{S_{ki}}$ = Start time of the i^{th} job in route k

$\boldsymbol{M_u}$ = Mileage of job u

$\boldsymbol{P_u}$ = Processing Time of job u

3.1. Forward Heuristic

The vehicle routes are defined using a nearest neighborhood approach. The routes are constructed in a sequential way by adding unrouted jobs that are nearest in terms of a "distance" measure to the current route until it can not be extended due to the feasibility constraints. Then, a new route is begun and the process repeated until no more unrouted jobs are left. The distance criterion is based on two factors: the minimum waiting time of the vehicle and the job mileage or equivalently its processing time. The underlying idea behind choosing this criterion is mainly to construct routes by adding consecutively jobs that can be started closely after one another and with the longest possible processing time. This way, it is thought that the vehicles are more likely to be exploited to their full capacity and the number of vehicles needed closer to the optimal solution.

3.1.1. The algorithm

For a given route k that contains i jobs $\{k_1 \ldots k_i\}$, the $(i+1)^{th}$ job is added to this route using the following steps:

- The feasibility constraints are tested for the unrouted jobs in the set U, i.e.

- The latest start time of the unrouted job u must be greater than the completion time of the i^{th} job k_i
- The length of the route cannot exceed the maximum allowable mileage when adding the job u
- The time elapsed between the start time of the first job, k_1, and the completion time of the job u must not exceed one week.

- The following metric is calculated for all feasible unrouted jobs u

$$Dist(u) = C_1 \frac{\max(a_u - C_{ki}, 0)}{\#HoursPerweek} + C_2 \frac{MaxWeeklyMileage}{M_u}$$

- The $(i+1)^{th}$ job of route k, k_{i+1}, corresponds to the nearest unrouted job with respect to the above criterion

$$k_{i+1} = \underset{u \in U, u \text{ feasible}}{\arg\min} \{Dist(u)\}$$

- If no job is found a new route is started and the process repeated

As the solution found depends on the parameters C_1 and C_2. Different values for these parameters will be applied to the problem and the best solution selected.

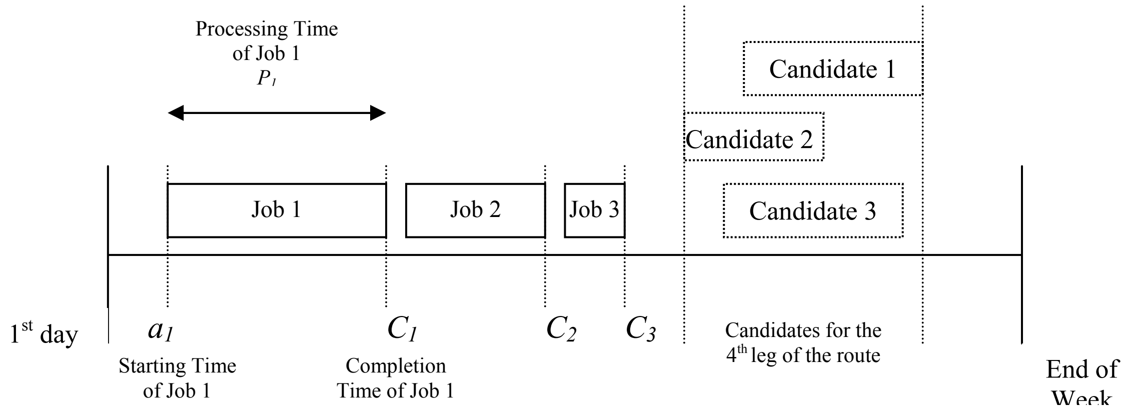


Figure 4: Forward Heuristic

3.2. Reverse heuristic

The reverse heuristic is similar in its principle to the forward case except that the routes are constructed beginning from the latest job in the routes. The consecutive jobs are added to the current routes from the latest to the earliest.

3.2.1. The Algorithm

A differentiation is made between the first step in the construction of each route and the consecutive steps:

1st Step

The last leg of the route is inserted first in the current route as follows:

- The following metric is calculated for all feasible unrouted jobs *u*

$$Dist(u) = C_1 \frac{(P_U + C_{ki})}{\#HoursPerweek} + C_2 \frac{M_u}{MaxWeeklyMileage}$$

- The last job of route *k*, corresponds to the nearest unrouted job with respect to the below criterion

$$k_1 = \underset{u \in U, u \text{ feasible}}{\arg\max} \{Dist(u)\}$$

ith Step

For a given route *k*, that contains *i* jobs $\{k_1 ... k_i\}$ and that are added in this order from the latest to the earliest, the $(i+1)^{th}$ job is added to this route using the following steps:

- The feasibility constraints are tested for the unrouted jobs in the set *U*, i.e.
  - The earliest completion time of the unrouted job *u* must be less than the start time of the i^{th} job k_i
  - The length of the route cannot exceed the maximum allowable mileage when adding the job *u*
  - The time elapsed between the start time of the job *u* and the completion time the first added job the start time, must not exceed one week.

- The following metric is calculated for all feasible unrouted jobs u

$$Dist(u) = C_1 \frac{\max(Ski_u - P_{ki} - b_u, 0)}{\#HoursPerweek} + C_2 \frac{MaxWeeklyMileage}{M_u}$$

- The $(i+1)^{th}$ job of route k, k_{i+1}, corresponds to the nearest unrouted job with respect to the above criterion

$$k_{i+1} = \underset{u \in U, u \text{ feasible}}{\arg\min} \{Dist(u)\}$$

- If no job is found a new route is started and the process repeated.

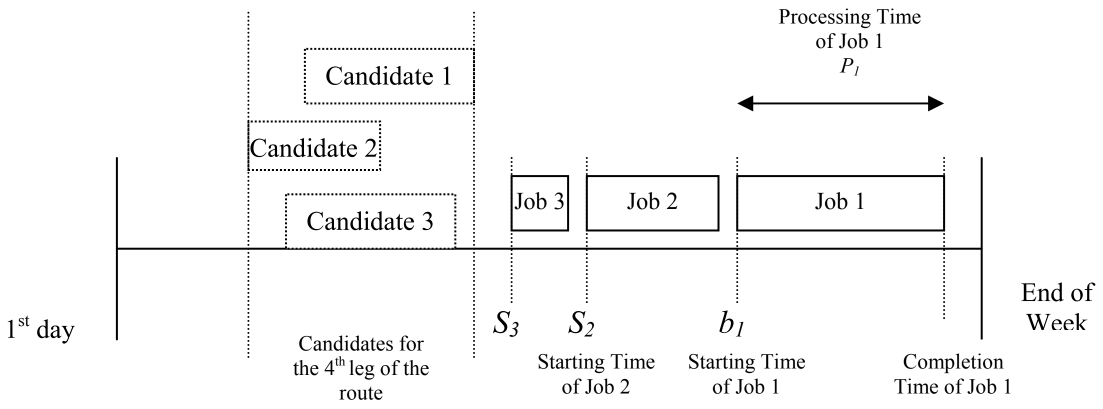


Figure 5: Reverse Heuristic

3.3. Randomized Forward Heuristic

This heuristic is an improved version of a heuristic developed by Bombien (2001). In this sequential heuristic, feasible solutions are constructed by selecting and assigning seed jobs to routes. Once the first job randomly assigned to the route, up to 4 jobs are added to the routes using the maximum processing time as selection criterion. By repeating the procedure several times, it can generate many good alternatives.

The improved heuristic described below, generalizes and extends the capabilities of this heuristic.

3.3.1. The algorithm

- The first job of each route is selected randomly from the set of unscheduled jobs U.
- For a given route k that contains i jobs $\{k_1 ... k_i\}$, the feasibility constraints are tested for the unrouted jobs in the set U, i.e.
    - The earliest completion time of the unrouted job u must be less than the starting time of the i^{th} job k_i
    - The length of the route cannot exceed the maximum allowable mileage when adding the job u
    - The time elapsed between the start time of the first job, k_1, and the completion time of the job u doesn't exceed one week.

- The following metric is calculated for all feasible unrouted jobs u

$$Dist(u) = C_1 \frac{\max(a_u - C_{ki}, 0)}{\# HoursPerweek} + C_2 \frac{MaxWeeklyMileage}{M_u}$$

- The $(i+1)^{th}$ job of route k, k_{i+1}, corresponds to the nearest unrouted job with respect to the above criterion

$$k_{i+1} = \underset{u \in U, u \text{ feasible}}{\arg\min} \{Dist(u)\}$$

- If no job is found a new route is started and the process repeated

3.4. Parallel Heuristic

In this heuristic, we construct feasible routes by first initializing a number of routes *r*, and then iteratively assigning one customer at a time in accordance with a set of rules. If at some step no assigned job can be inserted in a route, a new route is created. The number of initial routes can be anywhere between *1* and *n*. We set *r* to a predetermined lower bound. This will introduce the minimum number of routes, thus reducing the effort needed to reduce the number of routes afterward.

The initialization of the routes is done through the selection and assignment of seed jobs. We use an approach where the seed jobs are selected on the basis of both their mileage and their waiting time on existing routes. After initialization the main routing algorithm is called.

3.4.1. The algorithm

Initialization

- A lower bound *r* is determined for the number of routes needed. See section 4.2 for the bounds used.
- For $i \in \{1...r\}$
    - The following metric is calculated for all unrouted jobs *u*

$$Dist(u) = C_1 \frac{\max(a_u - C_{ki}, 0)}{\#HoursPerweek} + C_2 \frac{MaxWeeklyMileage}{M_u}$$

    - The job that minimizes the above criterion is the seed job for the i^{th} route.

Main routing algorithm

- For every existing route ρ we find the feasible unassigned job, if it exists, that minimizes the following metric:

$$Dist(u) = C_1 \frac{\max(a_u - C_{ki}, 0)}{\#HoursPerweek} + C_2 \frac{MaxWeeklyMileage}{M_u}$$

- This job is designed by $u_{\rho*}$.

- If there exists an unassigned job that can be inserted in the existing routes, the one that satisfies the relation, Min $Dist(u_{\rho*})$ $\rho \in$ existing routes, it is selected and added to the corresponding route
- If no unassigned can be added to the existing route a new route is created. Then, its seed is selected as in the initialization phase.

Chapter 4 Computational Experiments

4.1. Test Problems

The most widely used vehicle routing test problems are the 14-benchmark problems described by Christophides, Mingozzi and Toth 1979. As the problem addressed in this work is unique in its characteristics and these test problems are not suited for our case, it is important to generate a new set of benchmark problems to assess the performance of the newly developed heuristics.

To test our approach, a set of problems related to TLS Company operations has been generated. First, a cluster of deliveries is randomly generated and uniformly assigned to the forty-three FDCs served by the NDC. Then random time windows of 24 hours that start and end in the same day are randomly assigned to the generated deliveries. The algorithm used follows these steps:

1. Define # of jobs (deliveries) ***n***
2. Define maximum # jobs/FDC/week ***Maxjobs***
3. Generate a vector containing the identifiers of each of the FDCs, ***Maxjobs*** times.
4. Perform a random permutation on the initial vector
5. Select the first ***n*** elements of the permuted vector. They correspond to the required visits to the FDCs (jobs)
6. For each job
    - Generate randomly a time window
    - Assign the time window to the job

4.2. Lower Bounds

In order to evaluate the quality of the new heuristics, lower bounding procedures are developed and applied to the benchmark problems. We present in this section four different lower bounds on the number of vehicles.

We first formulate the Bin Packing Problem consisting of *n* jobs (items) and *n* vehicles (Bins). Let also w_j be the mileage of the job and *c* its maximum weekly mileage.

$$\min z = \sum_{i=1}^{n} y_i$$

Subject to

$$\sum_{i=1}^{n} x_{ij} = 1 \quad j \in V = \{1....n\}$$

$x_{ij} = 1$ if job *j* is assigned to vehicle *i*, $i,j \in V$

$= 0$ otherwise

$y_i = 1$ if the vehicle *j* is used, $i \in V$

$= 0$ otherwise

The first lower bound is obtained by solving the continuous relaxation of $0<=x_{ij}<=1$ and $0<=y_i<=1$. The solution of this relaxed problem is

$$z = \frac{\sum_{i=1}^{n} w_i}{c}$$

So a first lower bound is

$$L_1 = \left\lceil \frac{\sum_{i=1}^{n} w_i}{c} \right\rceil$$

The second lower bound to the bin-packing problem uses a more elaborate algorithm developed by Martello and Toth [1990]. The principle of this algorithm lies in the following theorem:

For any instance I of a Bin Packing Problem and any integer number α $0<=\alpha<=c/2$, let:

$$J_1 = \{ j \in V \mid w_j > c - \alpha \}$$

$$J_2 = \{ j \in V \mid c - \alpha \geq w_j > c/2 \}$$

$$J_3 = \{j \in V \,|\, c/2 \geq w_j \geq \alpha\}$$

and

$$L(\alpha) = |J_1| + |J_2| + \max(0, \left\lceil \frac{\sum_{j \in J_3} w_j - (|J_2|c - \sum_{j \in J_2} w_j)}{c} \right\rceil)$$

then

$$L_2 = \max\{L(\alpha) : 0 \leq \alpha \leq c/2, \alpha \text{ integer}\}$$

is a lower bound of the problem.

A third lower bound can be determined by considering the time a vehicle has to travel and wait to visit each customer *i* in an optimal schedule and then using the lower bound L_2 defined above. As the waiting time depends on the sequence of customers each vehicle is visiting, we introduce a new time

$$\Phi_i = \min(\max(a_j - b_i - P_i, 0)), \qquad j \in V \quad \textit{and} \quad a_i + P_i \leq b_j$$

The above equation determines the minimum amount of time a job has to wait before being processed. Let *UB* be some upper bound on the number of vehicles needed and *n* the number of jobs to be processed. At least *n-UB* jobs are subject to be waiting before their processing. Let Φ_{l1}, Φ_{l2}… Φ_{ln-UB} be the *n-UB* least values of Φ_I and define T=1week. As the optimal solution requires to consider more potential waiting timesΦ_I, a Lower bound on the number of vehicles is the minimum number of bins of capacity T for the following bin-packing problem of 2*n-UB* items of size: $P_1 \ldots P_n$, Φ_{l1}, Φ_{l2}… $\Phi_{ln-UB.}$

The last lower bound is based on the network flow algorithm proposed by Horn (1979). This algorithm minimizes the maximum lateness on m identical machines for n single operation Jobs while allowing preemption. Our approach consist of setting the starting number of machines to some lower bound on the number of vehicles, then increasing gradually the number of machines, and applying Horn's algorithm until the maximal maximum lateness becomes positive. Then the number of machine in the last iteration is a lower bound for the problem addressed.

4.3. Computational Results

The different heuristics described above were implemented using Matlab. Computation was performed on a 2 Ghz PC. The heuristics were applied to 1000-benchmark problems for different size problems: 20, 40, 60, 80, 100 jobs.

20 jobs case

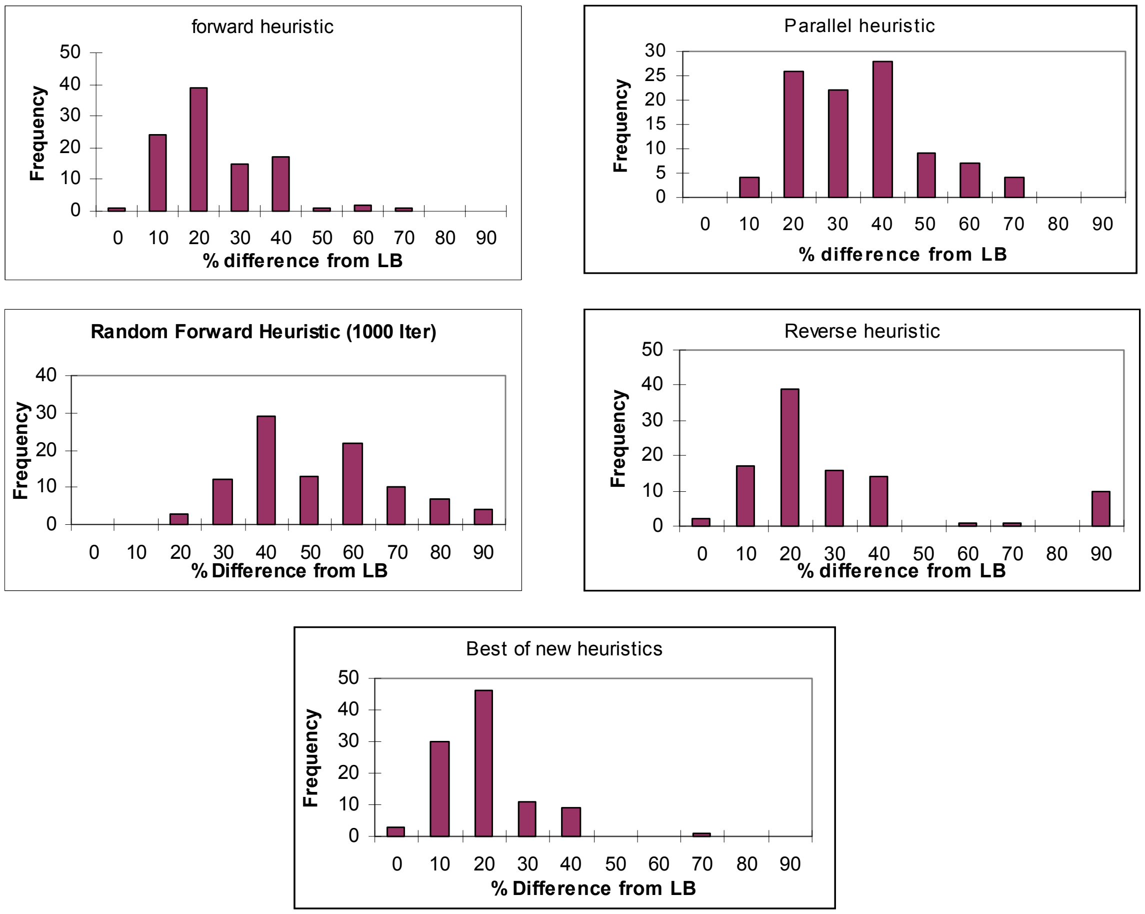
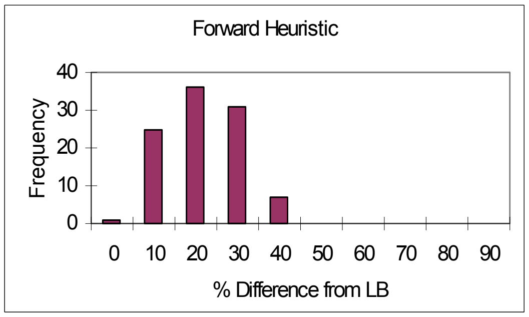
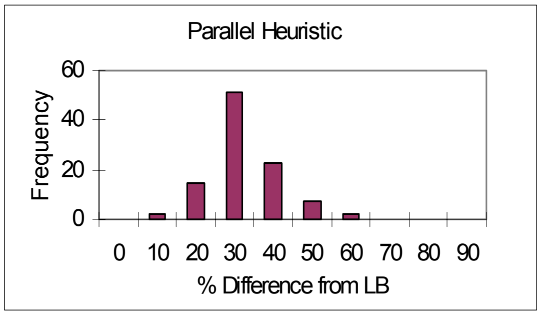
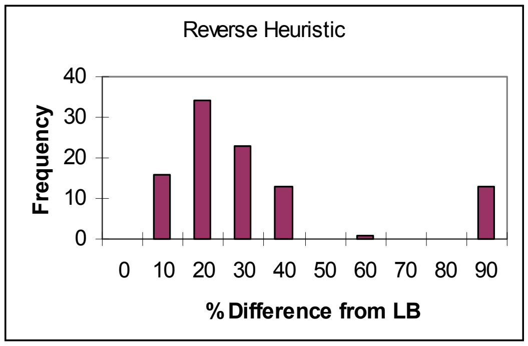


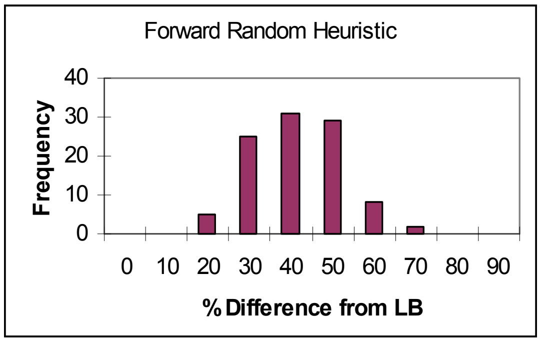
Figure 6: Histogram of the solutions obtained for the different heuristics for 1000 Benchmark problems (20 jobs)

40 jobs case









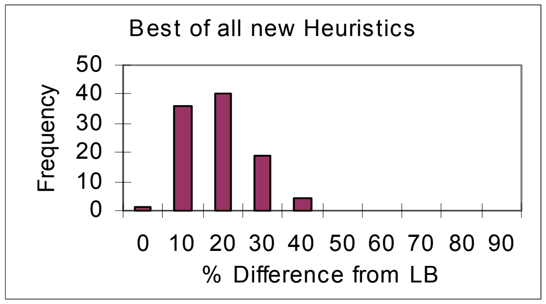


Figure 7: Histogram of the solutions obtained for the different heuristics for 1000 Benchmark problems (40 jobs)

60 jobs case

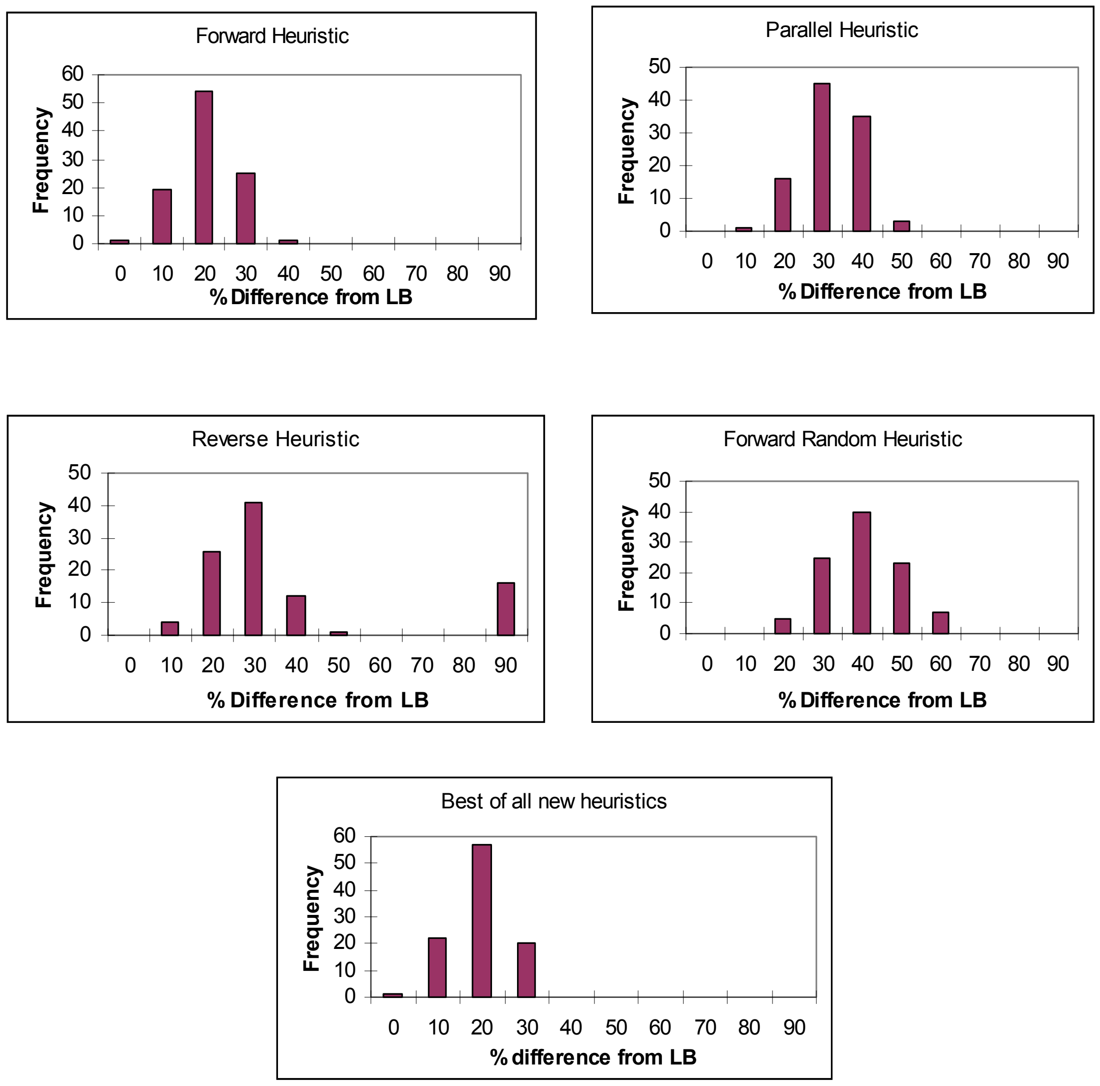


Figure 8: Histogram of the solutions obtained for the different heuristics for 1000 Benchmark problems (60 jobs)

80 jobs case

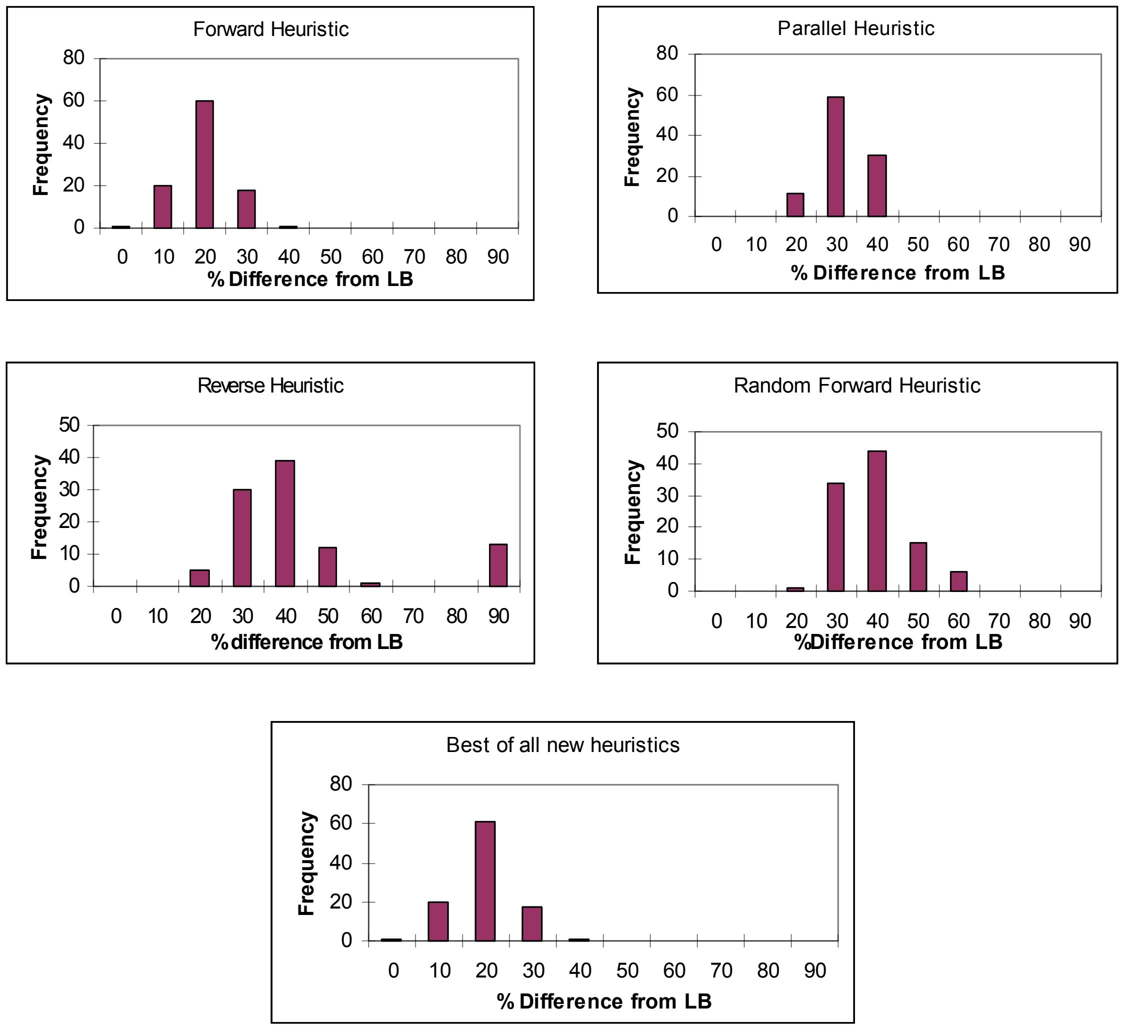


Figure 9: Histogram of the solutions obtained for the different heuristics for 1000 Benchmark problems (80 jobs)

The histograms of the solutions obtained from the heuristics for different size problems shows a better performance for the Forward Heuristic over the other methods especially for larger size problems (Figure 5- 9). The average performance is also stable relatively to the other methods (around 25% difference from the Lower Bound Table 2). It can also be noticed that the performance of the Random Forward Heuristic improves relatively faster compared to the other algorithms as the number of jobs increases (Figure 10) while the Reverse Heuristic showed an opposite trend. From all heuristics, the worst case performance of the Forward Heuristic is less than all the others and decreases for bigger size problems -around 30%-40%-. On the other hand, the worse case performance of the Reverse Heuristic ranged from 90 to 100% for all size problems.

In term of computation time, the Randomized Forward and Parallel Heuristic turn out to be the most demanding –100:1 the time required by the Forward Heuristic while the forward and Reverse Heuristic are the least consuming of computation time. The Forward Heuristic handles problem of realistic size relatively easily (Problems of 150 jobs are solved in approximately 5 seconds).

Concerning the stability of the heuristics, we can conclude from Table 2 that all heuristics except the Reverse Heuristic provide stable results. This is confirmed by the two modes in the solutions histogram of the Reverse Heuristic (Figure5-9).

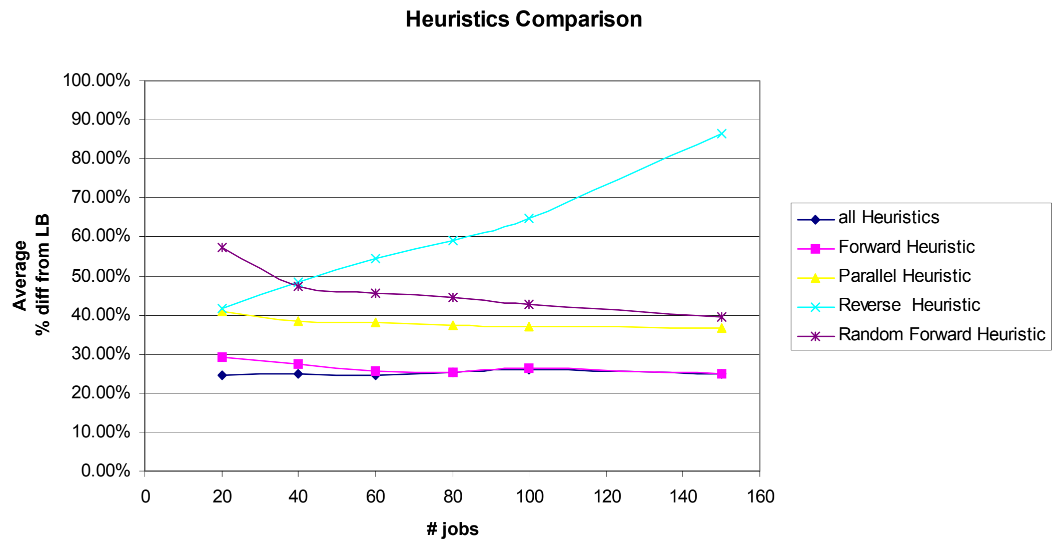


Figure 10: Heuristics Performance Comparison

	Forward Heuristic			**Parallel Heuristic**			**Reverse Heuristic**			**Rnd Fwd Heuristic (1000 Iterations)**		
# of jobs	Avg % Diff from LB	% Sol Stdev	CPU Time (Sec)	Avg % Diff from LB	% Sol Stdev	CPU Time	Avg % Diff from LB	% Sol Stdev	CPU Tim e	Avg % Diff from LB	% Sol Stdev	CPU Time
20	29.10	14.24	0.28	40.78	14.74	0.79	41.61	41.51	0.46	57.36	16.69	28.36
40	27.44	8.8	0.61	38.31	9.27	2.77	48.47	48.63	0.74	47.41	11.05	54.62
60	25.79	6.28	1.07	38.08	7.63	7.74	54.46	51.53	7.62	45.57	9.81	95.53
80	25.44	6.4	1.74	37.2	5.74	17.08	58.96	43.72	3.02	44.61	8.43	164.4
100	26.19	5.4	2.64	37.16	5.5	31.94	64.7	44.04	3.61	42.72	7.44	233.5
150	25.05	3.21	5.29	36.77	3.36	122.8	86.43	49.30	7.58	39.61	5.71	510.7

Table 2: Performance of the heuristics

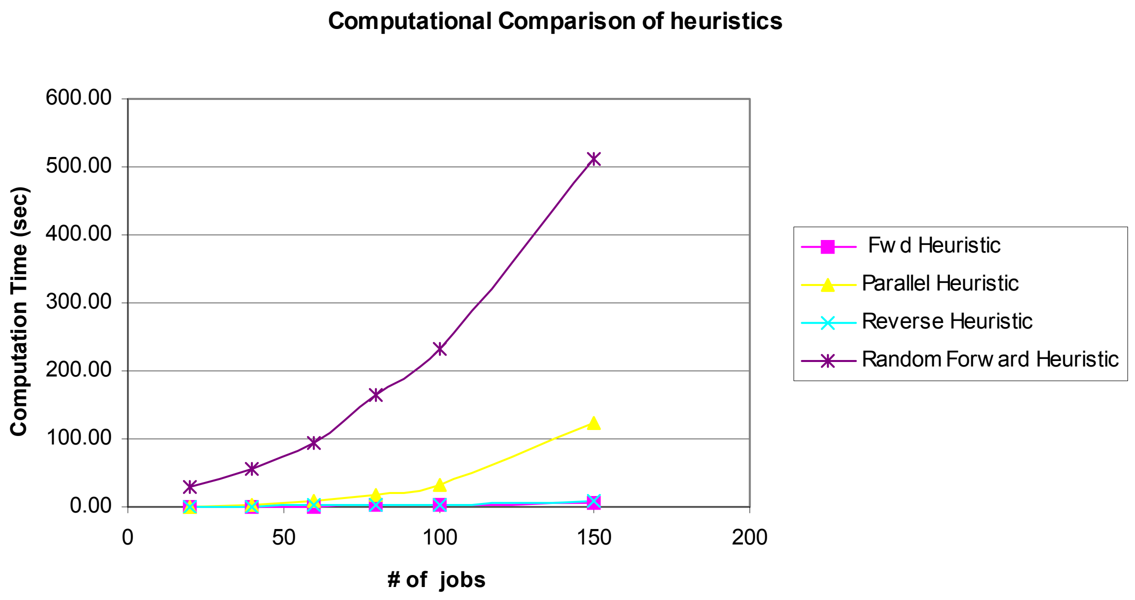


Figure 11: Computational performance of the heuristics

4.4 Evaluation of the lower bounding procedures

To assess objectively the performance of the developed heuristics, special attention needs to be given to the analysis of the quality of the lower bounds. Even though the ratio of the best heuristics solution to Lower bounds was in average of 25 %, this figure is only an upper bound of the real performance of the heuristics.

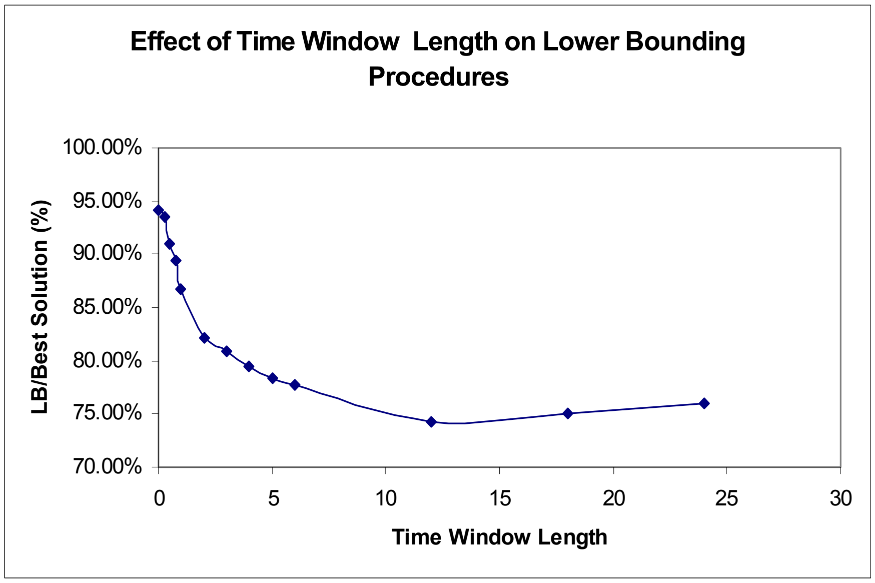


Figure 12: Effect of Time window length on the Lower Bounding procedures

The figure 12 shows the effect of the time windows on the quality of the lower bounds. For fixed job schedules the heuristic Solutions are at least 5% close to the optimal solution. If the time window length is increased slightly, the quality of the lower bounding procedure drops very quickly from 95% to 75%. This shows the inability of the actual lower bounding procedures to secure good lower bounds when time windows are introduced.

To further check this statement, 100 random benchmark problems of 60 jobs each have been run for different time window values to explore the distribution of the heuristic solutions. It can be noticed that the distribution of the solutions figure 14 is roughly the same for the different time window value and the heuristic is providing better solutions as the time window is increased. This leads to the conclusion that the lower bounding procedures are less performant as time windows are introduced.

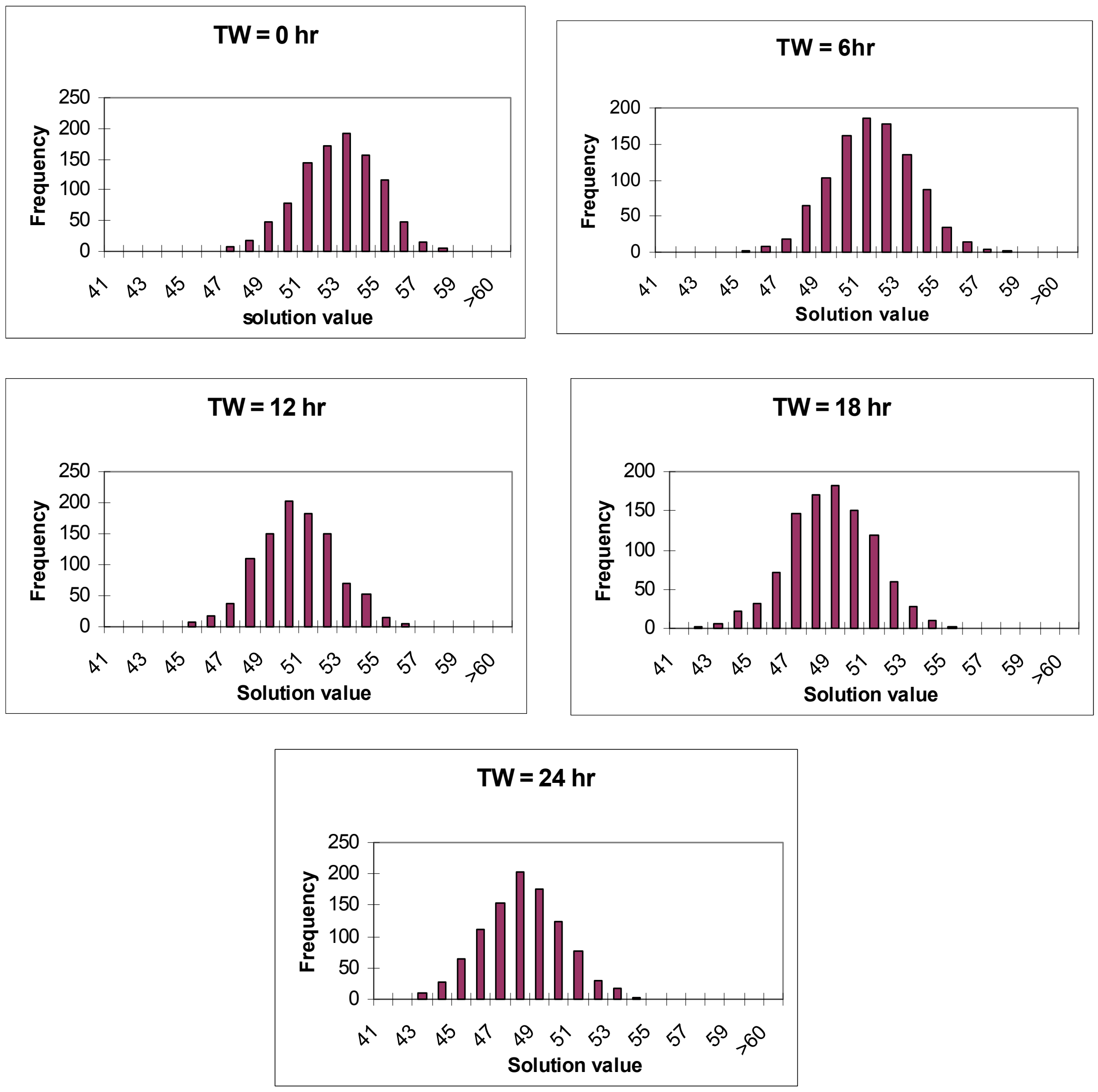


Figure 13: Histogram of the Heuristics solutions
For different Time Windows
Lower Bounding procedures

Chapter 5 Conclusions and Future Research

5.1 Conclusion

In this thesis, the difficult problem of scheduling and routing full load vehicles with the objective of minimizing fleet size was addressed. First, we proposed a Mixed Integer Linear Program model for the problem. We also showed the limitation of this approach in handling real-size problems using available software packages. Three lower bounding procedures were developed to secure lower bounds for the optimal solution. The two first methods are based on bin packing problem procedures while the third method is based on flow networks.

Since mathematical models proved to be a computationally expensive way to solve the problem, we developed four construction heuristics based both on sequential and parallel schemes.

To assess the proposed approach, the different heuristics were applied to randomly generated benchmark problems. The average performance of the best heuristic was around 25% higher than the best lower bound. It is conjectured that this is more a function of the weakness of the bounds.

5.2 Future Research

The following work can be further developed by investigating the following directions:

- The lower bounding procedures developed for these problems are not necessarily close to the optimal solution. An interesting problem would be to develop more effective bounding schemes. One approach is the use of statistical bounds based on extreme value theory and a technique proposed by Golden and Alt (1979). An example of this can be found in Wilson et al. (2002) in which a lower bounding scheme is developed for a two-stage flow line with dissimilar stage grouping.
- Better ways of applying randomization to the developed heuristics may be investigated.
- Improvement routines can be developed to improve the results obtained using the constructive heuristics.

- Meta-heuristics approaches such as Tabu-Search, Simulated-Annealing and Genetic Algorithms can be developed for the sake of comparison of effectiveness and computational performance with the heuristics developed in this work.

Bibliography

Baker, E. and J. Schaffer (1986) Computational experience with branch exchange heuristics for VRTPTW constraints. *Am J Math Management Science* 6, 261-300.

Bard, J.F., G. Kontoravdis, and G. Yu (2002) A branch and cut procedure for the vehicle routing problem with time windows. *Transportation Science* 36(2), 250-269.

Horn, W.A. (1979) Some simple scheduling algorithms

Bodin, L.D., B.L. Golden, A. Assad and M. Ball (1983) Routing and scheduling of vehicles and crews: The state of the art. *Comp Oper Res* 10, 69-211.

Bombien, R (2001) Generating Scheduled Routes for Trucking Services

Cordeau, J-F. and G. Laporte (2001) A tabu search algorithm for the site dependent vehicle routing problem with time windows. *Inform.* 29 (3), *292-297.*

Desrochers, M., J.K Lenstra, MWP Savelsberg, and F.Soumis (1988) Vehicle routing with time windows: optimization and approximation *Elsevier Scien. Publ* 65-83.

Desrochers, M., J. Desrosiers J., and M. Solomon (1992) A new Optimization Algorithm for the vehicle routing problem with time windows. *Operations Research* 40, 342-354.

Fisher, M. (1995) *Handbooks in OR & MS* 8 *Network Routing pp 1-33 Elsevier*.

Fisher, M. and R. Jaikumar (1981) A generalized assignment heuristic for vehicle routing. *Networks* 11, 109-124.

Golden, B.L., Alt, F.B., 1979. Interval estimation of a global optimum for large combinatorial problems. Naval Research Logistics Quarterly 26(1), 69 – 77.

Golden, B.L. and A.A. Assad (1986) Perspectives on vehicle routing: Exciting new developments. *Operations Research* 14, 803-810.

Koskosidis, Y.A., W.B Powell and M.M Solomon (1992) An optimization based heuristics for vehicle routing and scheduling with soft time windows constraints. *Transportation Science* 26, 69-85.

Laporte, G. and Y. Nobert (1987) Exact algorithms for the vehicle routing problem. *Annals of Discrete Math* 31, 147-184.

Laporte, G., M. Gendreau, J.Y. Potin, F. Semet (2000) Classical and modern heuristics for vehicle routing problem *Intl Trans in Oper Res* 7, 285-300.

Lin, S. and B. Kernighan (1973) An effective heuristic algorithm for the TSP. *Operations Research* 21, 498-516.

Martello, S. and P. Toth, (1990) Algorithms and computer implementation Wiley-Interscience series in discrete mathematics and optimization.

Osman, I.H. (1993) Meta strategy simulated annealing and tabu search algorithms for the VRP. *Annals of Operations Research* 41, 421-451.

Potvin, J.Y., T. Kervahut, B.L. Garcia, J.-M. Rousseau (1996) The VRPTW. *Informs Journal on Computing* 8, 158-164.

Solomon, M.M. (1987) Algorithms for the VR and scheduling problem with time constraints. *Operations Research* 35, 254-265.

Taillard, E.D. (1993) Parallel iterative search method for VRP, *Networks* 23, 661-673.

Wilson, A. D., R. E. King, and J. R. Wilson. 2002. Case study on statistically estimating minimum makespan for flow line scheduling problems. *European Journal of Operational Research*, in review.