CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Fleet-Sizing for Multi-Depot and Periodic Vehicle Routing Problems Using a Modular Heuristic Algorithm

Alireza Rahimi Vahed
Teodor Gabriel Crainic
Michel Gendreau
Walter Rei

September 2012

CIRRELT-2012-51

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

UNIVERSITÉ LAVAL · UQÀM · HEC MONTRÉAL · ÉCOLE POLYTECHNIQUE MONTRÉAL · Université de Montréal

Fleet-Sizing for Multi-Depot and Periodic Vehicle Routing Problems Using a Modular Heuristic Algorithm

Alireza Rahimi Vahed[1,2,*], Teodor Gabriel Crainic[1,3], Michel Gendreau[1,4], Walter Rei[1,3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-Ville, Montréal, Canada H3C 3J7

[3] Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

[4] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

Abstract. In this paper, we address the problem of determining the optimal fleet size for three vehicle routing problems, i.e., multi-depot VRP, periodic VRP and multidepot periodic VRP. In each of these problems, we consider three kinds of constraints that are often found in reality, i.e., vehicle capacity, route duration and budget constraints. To tackle the problems, we propose a new Modular Heuristic Algorithm (MHA) whose exploration and exploitation strategies enable the algorithm to produce promising results. Extensive computational experiments show that MHA performs impressively well, in terms of solution quality and computational time, for the three problem classes.

Keywords: Multi-depot periodic vehicle routing problem, fleet-sizing, modular heuristic algorithm.

Acknowledgements. Partial funding for this project has been provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), through its Industrial Research Chair, Collaborative Research and Development, and Discovery Grant programs, by the Fonds de Recherche du Québec - Nature et technologies (FRQNT) through its Team Research Project program and infrastructure grants, and by the EMME/2-STAN Royalty Research Funds (Dr. Heinz Spiess).

---

* Corresponding author: Alireza.RahimiVahed@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2012

1 Introduction

In the classical Vehicle Routing Problem (VRP), a homogeneous fleet of vehicles services a set of customers from a single distribution depot or terminal. Each vehicle has a fixed capacity that cannot be exceeded and each customer has a known demand that must be fully satisfied. Each customer must be serviced by exactly one visit of a single vehicle and each vehicle must depart from the depot and return to the depot (Dantzig and Ramser [1959]).

During the past five decades, the vehicle routing problem and its variations have been extensively studied. However, surveying the literature, one can perceive that not all VRP variants have been addressed with the same degree of attention. This is the case for the problem classes considered in this paper. On the other hand, most of the methodological developments target a special problem variant, the Capacitated VRP (CVRP) or the VRP with Time Windows (VRPTW), despite the fact that the majority of the problems encountered in real-life applications display more complicating attributes and constraints. This also applies to the problem addressed in this paper. Moreover, the literature survey underlines that, the problem classes studied in this study have been often set up with objective functions other than the minimization of the fleet size, despite the fact that there are many real-life applications in which it is more crucial to give more importance to the minimization of the fleet size in comparison with other existing objectives as the total traveled distance. This situation may occur when important factors such as high vehicle fixed costs exist.

Our objective is to contribute toward addressing the above three challenges. In this paper, we propose a modular heuristic algorithm capable of successfully dealing with three VRP variants: Multi-depot VRP, MDVRP, Periodic VRP, PVRP, and Multi-depot Periodic VRP, MDPVRP. In each of these problems, the goal is to determine the optimal fleet size where three practical constraints, i.e., vehicle capacity, maximum route duration and budget constraints, should be satisfied. The proposed heuristic algorithm incorporates different exploration and exploitation strategies to produce good results, in terms of solution quality and computational efficiency.

The remainder of this paper is organized as follows: Section 2 gives the problem statement. In Section 3, the literature survey relevant to the topic of this study is presented. Different aspects of the proposed heuristic algorithm are described in Section 4. The experimental results are given in Section 5. Finally, Section 6 provides conclusions and the evaluation of the work.

2 Problem statement

In this section, we formally state each of the problem classes, introducing the notations used throughout this paper.

MDVRP- Consider an undirected graph $G(V, E)$. The node set V is the union of two subsets $V = V_C \cup V_D$, where $V_C = \{C_1, ..., C_N\}$ represents the customers and $V_D = \{D_1, ..., D_M\}$ includes the depots. With each node $i \in V_C$ is associated a deterministic demand q_i. The edge set E contains an edge for each pair of customers and for each depot-customer combination. There are no edges between depots. With each edge $(v_i, v_j) \in E$ is associated a travel cost c_{ij}. The travel distance for arriving to node j from node i (d_{ij}) is considered equal to c_{ij}. Each vehicle performs only one route and each vehicle route must start and finish at the same depot. (Cordeau et al.

[1997]).

PVRP- In the PVRP, the undirected graph $G(V, E)$ is modified by fixing the value of M to one and by introducing a planning horizon of T periods. In such a graph, each customer i is characterized by a service frequency f_i, stating how often within these T periods the customer must be visited and a list Ω_i of possible visit-period combinations. Moreover, Ω is defined as the set of subsets of T, giving all the allowable patterns, that is: $\Omega = \cup_{i=1}^{N} \Omega_i$ (Vidal et al. [2012]).

MDPVRP- Finally, the Multi-Depot Periodic VRP (MDPVRP) combines the two above problem settings, asking for the selection of a depot and a visit pattern for each customer, with services in different periods to the same customer being required to originate at the same depot (Vidal et al. [2012]).

In each problem class, the goal is to determine the optimal fleet size subject to the following three practical constraints:

1. The vehicle capacity constraint: This constraint states that the total demand of the customers on any route should not exceed the vehicle capacity Q.
2. The route duration constraint: This constraint reveals this fact that the total duration of a route does not exceed a preset value D.
3. The budget constraint: In many logistical systems, one is usually faced with budgetary constraints that come from the fact that a limited investment budget is available for a certain area or a certain period of time. Although it is quite easy to understand the practical aspect of these constraints, budget considerations are almost always ignored when dealing with VRPs. In this paper, we consider a Travel-Distance Budget (TDB) constraint which imposes a threshold on the total distance traveled by vehicles for delivery operations. The TDB constraint is defined using two different rules, each realizing an important managerial challenge in real-life distribution and logistical systems. In the first rule (R_1), we set a bound on the total distance that vehicles are permitted to travel over the planning horizon. On the other hand, the second rule (R_2) aims to reflect the situations in which, due to geographical and operational constraints, the total distance traveled by vehicles assigned to a depot cannot exceed a limit in a period.

Depending on how we model the TDB constraint, each of the above problem classes can be expressed by one of the following mathematical programming models.

$$(R_1)\ \textit{min}\quad K \tag{1}$$

subject to

$$F(x, K) = b \tag{2}$$

$$\tau \leq \epsilon \tag{3}$$

In the above model, K is the total number of used vehicles (Fleet size) over the planning horizon which is to be minimized. Constraint (2) corresponds to the vehicle-capacity and route-duration restrictions described above. Constraint (3) imposes that the total traveled distance (τ) is limited by a positive value ϵ.

$$(R_2)\ \textit{min}\quad K \tag{4}$$

subject to

$$F(x, K) = b \tag{5}$$

$$\tau_{tj} \leq \epsilon_{tj} \qquad \forall t \in T, \forall j \in D; \qquad (6)$$

where τ_{tj} is the total distance traveled by the vehicles assigned to depot j in period t and ϵ_{tj} is a positive upper bound which is set on τ_{tj}.

Cordeau et al. [1997] showed that the formulation of a generalized PVRP includes the MDVRP as a special case by associating a different period to each depot, such that the ith customer has a frequency f_i=1 and can be visited in any period. Vidal et al. [2012] extended this result by proving that an MDPVRP with T periods and D depots can be transformed into a generalized PVRP by associating a period to each (period, depot) pair, such that the ith customer, having a list Ω_i of L patterns, is visited f_i times over the planning horizon using one of the $D \times L$ patterns. We rely on these two transformations in the development of the proposed modular heuristic algorithm.

3 Literature review

In this section, we focus on reviewing papers formerly published in the literature to solve the PVRP, the MDVRP and the MDPVRP. The aim of this review is first to present the most recently proposed heuristic and meta-heuristic algorithms for the considered problems and, to discern leading solution approaches which have been demonstrated to be impressive to address the three problem settings.

Several heuristics have been put forward for the MDVRP. Early heuristics, performing based on simple construction and improvement procedures, have been developed by Tillman [1969], Tillman and Hering [1971], Tillman and Cain [1972], Golden et al. [1977], and Raft [1982].

Cassidy and Bennett [1972] proposed an iterative heuristic for the multi-depot vehicle routing problem. The proposed method progressively improves the routing arrangements starting from an initial solution. An interesting feature of the algorithm is the method of data storage, which is designed to facilitate the alteration of route configurations. The suggested heuristic is divided in three main steps. In the first step, an initial solution is generated by assigning each customer to its nearest depot. In the second step, the initial solution obtained from the previous step is improved by taking each customer in turn and trying to fit it into another position. Finally in the last step, the algorithm examines all depots in the routes to see if any of them can be replaced by any of those still having enough capacity. Several years later, Chao et al. [1993] proposed a search procedure combining the record-to-record local search method for the reassignment of customers to different vehicle routes, followed by a 2-opt procedure for the improvement of individual routes. Salhi and Sari [1997] suggested a multi-level construction-based composite heuristic for solving a multi-depot fleet mix vehicle routing problem in which allocating customers to depots, finding the delivery routes and determining the vehicle fleet composition are simultaneously considered. The main purpose of that paper was to minimize the total traveled cost where both the vehicle capacity (the largest vehicle in case there are different types of vehicles) and the maximum distance traveled on any route must not be violated. The proposed heuristic consists of three levels. In the first level, a starting solution is found as follows: the vehicle fleet mix problem is first solved within each depot with certain customers left unassigned (borderline customers). Then each of these customers is inserted into an existing route or an empty route by using a selection-insertion procedure. In the second level, a composite heuristic, which attempts to improve on the solution found for

each depot when taken separately, is introduced. Finally in the third level, a composite heuristic which considers all depots is implemented.

Various meta-heuristics have also been developed to tackle the MDVRP. Renaud et al. [1996] described a tabu search heuristic in which an initial solution is built by first assigning every customer to its nearest depot. A petal algorithm is then used for the solution of the VRP associated with each depot. The algorithm is completed by an improvement phase using either a subset of the 4-opt exchanges to improve individual routes, swapping customers between routes from the same or different depots, or exchanging customers between three routes. The tabu search approach of Cordeau et al. [1997] is probably the best known algorithm for the MDVRP. An initial solution is obtained by assigning each customer to its nearest depot and a VRP solution is generated for each depot by means of a sweep algorithm. Improvements are performed by transferring a customer between two routes incident to the same depot, or by relocating a customer in a route incident to another depot. Reinsertions are performed by means of the GENI heuristic (Gendreau et al. [1992]). One of the main characteristics of this algorithm is that infeasible solutions are allowed throughout the search. Continuous diversification is achieved through the penalization of frequent moves. Dondo and Cerdà [2007] studied the multi-depot vehicle routing problem with time windows. To solve it, they presented a model-based large-scale neighbourhood search algorithm that steadily improves an initial solution generated through the three-phase cluster-based hybrid approach. At each iteration, a sequence of two evolutionary steps is executed. First, a neighbourhood around the starting solution is generated by using a mixed-integer linear problem that permits the algorithm to exchange multiple nodes between neighbouring trips. Next, a different neighbourhood is defined by just allowing relocations of nodes on the same tour. Lau et al. [2010] addressed an MDVRP in which the objective is to simultaneously optimize both the cost due to the total traveling distance and that due to the total traveling time. To solve the problem, a genetic algorithm with fuzzy logic adjusting the crossover rate and mutation rate after ten consecutive generations was proposed. Finally, Yu et al. [2011] designed a parallel ant colony optimization algorithm for the MDVRP. In the proposed algorithm, three improved strategies: the coarse-grain parallel strategy, the ant weight strategy and the local search strategy, were applied.

Solution algorithms proposed to solve the PVRP can be categorized into two main groups, i.e., classical heuristics, and meta-heuristics. Heuristics have been extensively studied to solve the PVRP. The majority of these heuristics are multi-phase optimization approaches which try to solve the problem at hand in a sequential manner. Russell and Gribbin [1991] presented a multi-phase approach to solve the PVRP. The first phase of the proposed method consists of a procedure which generates initial solutions by using a generalized network approximation method. The second phase involves an interchange heuristic that reduces the total traveled cost through a surrogate traveling salesman problem. In the third phase, the total traveled cost is further reduced by addressing the actual routes. Finally, a proposed 0-1 integer model is used to attempt further improvements. Chao et al. [1993] provided a two-phase heuristic. To obtain an initial solution they solve an integer linear program to assign visit day combinations to the customers. In a second phase, they use several improvement operators while they relax the capacity of the vehicles. When getting stuck, re-initializations are performed. Bertazzi et al. [2004] suggested a heuristic algorithm for a special case of the PVRP namely the periodic traveling salesman problem, in which a single vehicle is used in each period. The algorithm is a construction type with an embedded improvement procedure. At each iteration, a procedure selects a not yet processed city, assigns to it a

combination of visit days and, for each day of the combination day, inserts the city to the best position of the current partial tour. The iteration process is temporarily interrupted after a predefined number of iterations and an iterative improvement procedure tries to improve the current solution.

These early heuristics are outperformed by more recent meta-heuristic approaches, including tabu search, scatter search, and variable neighbourhood search. Cordeau et al. [1997] proposed a tabu search heuristic for the PVRP that can also be used to solve the Multi-Depot Vehicle Routing Problem and the Periodic Traveling Salesman Problem (PTSP). The neighbourhood consists of moving a customer from one route to another route of the same day or assigning a new visit combination to a customer. Insertions and removals of customers are performed using the GENI operator (Gendreau et al. [1992]). The tabu search algorithm allows for infeasible solutions during the search process using an adaptive penalty function. This paper presents an asynchronous parallel metaheuristic for the period vehicle routing problem (PVRP). Drummond et al. [2001] designed an island-based parallel meta-heuristic for the PVRP. The proposed algorithm was based on concepts used in parallel genetic algorithms and local search heuristics. Angelelli and Speranza [2002] presented a tabu search algorithm for an extension of the periodic vehicle routing problem where the homogeneous vehicles have the possibility of renewing their capacity at some intermediate facilities. The initial solution of the proposed tabu search is generated by using a procedure similar to the sweep algorithm (Gillett and Miller [1974]). Then, the initial solution is improved via an improvement procedure which consists of four move operators, i.e., relocation, changing the visiting schedule of a customer, redistribution, intersection. To enhance the performance of the proposed algorithm, the tabu search is permitted to search the solution space by using a tunneling strategy. Besides that, a diversification mechanism is also used. Recently, a scatter search procedure was developed by Alegre et al. [2007] for solving a problem of periodic pick-up of raw materials for a manufacturer of auto parts. They use a two-phase approach, that first assigns orders to days and then constructs the routes of each day. Alonso et al. [2008] proposed a tabu search for an extension of the periodic vehicle routing problem where each vehicle can service more than one route per day as long as the maximum delay operation time in not exceeded. Besides that, there exist some accessibility constraints of the vehicles to the customers in the sense that not every vehicle can visit every customer. The efficiency of the implemented tabu search is proved based on some existing and randomly generated test problems. Hemmelmayr et al. [2009] implemented a variable neighbourhood search for the periodic vehicle routing problem. First, for obtaining an initial solution each customer is randomly assigned a visit day combination. Routes are constructed by solving a vehicle routing problem for each day using Clarke and Wright savings algorithm (Clarke and Wright [1964]). Then, for the shaking phase two popular and effective neighbourhoods, i.e., move and cross-exchange, are proposed in order to enhance the quality of the starting solution in each iteration. Finally, the solution obtained through shaking is further improved by using 3-opt procedure as a local search. Pirkwieser and Raidl [2010] proposed a variable neighbourhood search for the periodic vehicle routing problem with time windows. In that paper, the authors claimed that using a random VNS often yielded significantly better results than a VNS using a reasonable fixed ordering of the shaking neighbourhoods. Furthermore, a selectively applied simple inter-route improvement procedure, 2-opt*, was shown to considerably improve both VNS variants at nearly no computational cost at all. Gulczynski et al. [2011] developed a new heuristic for the PVRP that combined integer programming and the record-to-record travel algorithm. The proposed heuristic produced very high-quality

results on standard benchmark instances. The authors also extended the heuristic to two new variants of the PVRP that involve reassigning customers to new routes and balancing the workload among drivers across routes.

The majority of solution methods, targeting the MDPVRP, are divided into two main groups: 1) Classical heuristics, which often solve the problem in a sequential manner, and 2) Sophisticated meta-heuristics and parallel algorithms, which tackle the problem by simultaneously optimizing all the involved attributes.

We are aware of three heuristics in the first group. Hadjiconstantinou and Baldacci [1998] formulated the problem of providing maintenance services to a set of customers as the MDPVRP with Time Windows (MDPVRPTW). The authors proposed a multi-phase optimization problem and solved it using a four-phase algorithm. In the developed algorithm, all customers are first assigned to particular depot. Then, customer visits are successively inserted among available periods to obtain feasible visit combinations. In the third phase, each of the depot-period VRP sub-problems is separately solved using a tabu search algorithm. Finally, in the last phase, solutions obtained during the optimization process are improved by modifying the period or depot assignments through a 3-opt procedure. Kang et al. [2005] designed a two-phase heuristic method to address the same problem. In the proposed method, all feasible schedules are first generated from each depot for each period and, then, the set of routes are determined through using the shortest path problem. Parthanadee and Logendran [2006] proposed a tabu search heuristic to tackle the problem considered by Hadjiconstantinou and Baldacci [1998]. In this algorithm, all the initial assignments are built by cheapest insertion, where each customer is assigned to its nearest depot and is given its most preferred visit pattern. In the improvement phase, a neighbourhood search is defined by depot and visit pattern interchanges.

We are also aware of two contributions belonging to the second group. The first contribution was the evolutionary meta-heuristic proposed by Vidal et al. [2012]. The authors developed a hybrid Genetic Algorithm (GA) to tackle the MDPVRP and two of its special cases, i.e., the Multi-depot VRP (MDVRP) and the Periodic VRP (PVRP). The most interesting feature of the proposed GA is a new population-diversity management mechanism which allows a broader access to reproduction, while preserving the memory of what characterizes good solutions represented by the elite individuals of the population. The second contribution was the cooperative parallel algorithm designed by Crainic et al. [2009]. The authors proposed a structured cooperative parallel search method, called Integrative Co-operative Search (ICS), to solve highly complex combinatorial optimization problems. The proposed ICS framework involves problem decomposition by decision sets, integration of elite partial solutions yielded by the sub-problems, and adaptive guiding mechanism. The authors used the MDPVRPTW to present the applicability of the developed methodology.

This brief review supports the general statements made in Section 1 that the problem classes of this study, especially the MDPVRP, are among the VRP variants which did not receive an adequate degree of attention and the solution algorithms proposed to solve them are scarce. Moreover, to the best of our knowledge, there is no significant contribution in the literature dealing with the minimization of the fleet size for the problem settings considered in this paper. To contribute toward addressing these three challenges, we develop a Modular Heuristic Algorithm (MHA) to efficiently address the PVRP, the MDVRP and the MDPVRP. The proposed MHA is described in the next section.

4 The proposed Modular Heuristic Algorithm (MHA)

In this section, we propose a new modular heuristic algorithm which solves each of the considered problems in three sequential phases, each targeting one special dimension of the problem. The general concepts and structure of the heuristic algorithm are first described in Section 4.1. Then, the main components of the MHA are explained in details in Sections 4.2-4-5.

4.1 The general structure of MHA

The Modular Heuristic Algorithm (MHA) that we propose is based on the sequential optimization paradigm, but it includes a number of advanced exploration and exploitation features which contribute to its high performance level, in terms of solution quality and computational efficiency.

The general scheme of the proposed heuristic algorithm is displayed in Algorithm 1. MHA consists of three different steps that sequentially address the decisions to be made. These decisions are: the visit pattern assignment, the depot assignment and the detailed route design. These three steps can be performed in different orders depending on the problem in question. Our experiments show that the order generating the best results, in terms of solution quality and computational time, is:

1. The visit pattern assignment: In this step, each of the customers is assigned to one of the possible visit patterns.
2. The depot assignment: In this step, customers of each period are assigned to depots.
3. The routing problem: Finally, in this step, the routes are established for each period and depot.

These three steps are iteratively repeated until MHA reaches its pre-defined stopping criteria. In this study, the following two stopping criteria are simultaneously considered:

- MHA is stopped if no improving solution is found for Ψ successive iterations. Ψ is a positive value which is determined at the beginning of the algorithm. Or,
- MHA is terminated if it passes a maximum allowable running time.

One of the most important characteristics of MHA is to use an elitism strategy which enables the algorithm not to lose good and diverse solutions obtained in the course of the optimization. Towards this end, MHA keeps all generated high-quality and diverse solutions in a list called the reference set. The reference set has the size equal to a predetermined positive value γ and consists of two different subsets. The first subset, B_1, preserves $\lceil 3 \times \gamma/4 \rceil$ high quality solutions, while the second subset, B_2, is made up of $\lfloor \gamma/4 \rfloor$ diverse solutions. The reference set is initially set as an empty list and is subject to be iteratively updated. Suppose a new solution, S_{new}, is obtained by the algorithm. The reference set is updated using the following two steps:

1. S_{new} is first investigated in terms of solution quality. In this case, S_{new} is directly added to B_1 if the number of solutions preserved in B_1 is less than $\lceil 3 \times \gamma/4 \rceil$; otherwise, if S_{new} is better than the worst existing solution in B_1, the latter is replaced by the former.

2. If none of the conditions mentioned in Step 1 are not met, S_{new} is assessed in terms of solution diversification. In this case, S_{new} is directly added to B_2 if the number of solutions existing in B_2 is less than $\lfloor \gamma/4 \rfloor$; otherwise, the following replacement strategy is implemented. We first define the contribution to diversity of solution *S* to the first subset of the reference set, $D(S, B_1)$, as the similarity between itself and its nearest neighbour in B_1, that is:

$$D(S, B_1) = \min_{X \in B_1, X \neq S} \Delta(S, X)$$

where $\Delta(S, X)$ is the Hamming distance. Moreover, let us define OF_S as the objective function value of solution *S*. The replacement strategy is implemented in three phases as follows: Firstly, the replacement strategy considers all the solutions of B_2 with poorer objective function values than S_{new} and finds the one, S_{max}, which maximizes the ratio of (*objective function value*)/(*contribution of diversity*) (Step 1). Then, the new generated solution, S_{new}, replaces S_{max} if the following inequality holds (Step 2):

$$\frac{OF_{S_{new}}}{D(S_{new}, B_1 - S_{max})} < \frac{OF_{S_{max}}}{D(S_{max}, B_1)} \tag{7}$$

In this way, we introduce into B_2 a solution with better objective function value and possibly higher contribution to diversity. If Inequality (7) does not hold, the worst solution of the set determined in the first step is replaced by S_{new} (Step 3).

Algorithm 1 Modular heuristic algorithm

```
-Initialize the search parameters.
-Determine the upper limit of the budget constraint.
-Set the initial reference set as an empty list.
while the termination criterion is not met do
    -Assign a possible visit pattern to each customer.
    -Assign each customer to a depot in each period of the
      selected visit pattern.
    -Design routes visiting customers assigned to the same
      depot using the three-phase heuristic.
    -Update the reference list.
end while
```

In the following sections, each of the steps used in MHA is described in details.

4.2 Solution representation

The first step in designing an algorithm for a particular problem is to devise a suitable solution representation scheme. In the proposed heuristic algorithm, the path representation proposed by Rahimi-Vahed et al. [2012] is used. The idea of this path representation is that the customers are listed in the order in which they are visited. In this kind of representation, a single row array of the size equal to N+1 is generated for each depot in each period, where N is the number of customers to be visited. The first position of the array (index 0) is related to the corresponding depot, while each of the other positions (index i; $1 \leq i \leq N$) represents a customer. The value assigned to a

position of the array represents which customer should be immediately visited after the customer or depot related to that position. In this path representation, negative values corresponds to the beginning of the next route index, 0 refers to the end of the routes and a vacant position reveals that the customer corresponding to that position is not served by the depot with which the array is associated. For a detailed description of the above solution representation, readers should refer to Rahimi-Vahed et al. [2012].

4.3 Visit pattern assignment

The modular heuristic algorithm, as depicted in Algorithm 1, first assigns, at each iteration, a possible visit pattern to each customer. There exist a very scarce number of contributions in the literature that use systematic and non-random methods to assign customers to visit patterns. For example, Tan and Beasley [1984] extended the generalized assignment problem of Fisher and Jaikumar [1981] to assign a customer to an allowable visit pattern. Christofides and Beasley [1984] developed a median problem to establish an initial assignment of customers to visit patterns that meets customer service requirements. Russell and Gribbin [1991] designed a generalized network approximation method to assign visit patterns to customers. The proposed method was represented by a tripartite transshipment graph with source nodes (customers), transshipment nodes (allowable visit patterns), and sink nodes (periods of the planning horizon).

In this study, we also propose a non-blind and non-random algorithm to systematically assign customers to their possible visit patterns. In the proposed algorithm, we first locate a single point, called reference point, for each period of the planning horizon and, then, using a new Integer Programming Model (IPM), customers are assigned to visit patterns so as to minimize the sum of customer-to-reference point distances. The proposed IPM, unlike the similar models existing in the literature, is governed by some parameters whose values are dynamically adjusted in the course of the optimization. This feature enables IPM to assign better visit patterns to customers as MHA evolves. The integer programming model is formulated as follows:

$$min \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{k \in \Omega_i} a_{kt} \{(x_i - x_t^{\nu})^2 + (y_i - y_t^{\nu})^2\} u_{ik} \tag{8}$$

subject to

$$\sum_{k \in \Omega_i} u_{ik} = 1 \qquad \forall i \in V; \tag{9}$$

$$LB_t^{\nu} \leq \sum_{i=1}^{N} \sum_{k \in \Omega_i} a_{kt} q_i u_{ik} \leq UB_t^{\nu} \qquad \forall t \in T; \tag{10}$$

$$u_{ik} \in \{0, 1\} \qquad \forall i \in V, \forall k \in \Omega_i; \tag{11}$$

In the above model, (x_i, y_i) is the location of customer i and (x_t^{ν}, y_t^{ν}) is the location of a reference point which is generated, in iteration ν of the algorithm, for period t. Moreover, the parameter a_{kt} equals 1 if period t is in pattern k, and 0 otherwise. The decision variable of this model is u_{ik}, which is equal to 1 if customer i is assigned to pattern k, and 0 otherwise. Finally, LB_t^{ν} and UB_t^{ν} are respectively lower and upper limits that bound the total number of demands which should be satisfied in period t and iteration ν. As it can be seen in this mathematical formulation, the aim is to assign possible visit patterns to customers so that the total squared Euclidean distance

between the customers and reference points is minimized. Constraints (9) impose that each customer is assigned to exactly one feasible pattern. Constraints (10) show that the demands serviced in a given period must be within an imposed interval.

One of the most crucial parameters affecting the strength of the above integer programming model is the reference points' locations. In this paper, the reference points' locations are generated using a memory-based algorithm as follows: We first enumerate all the customers that can be serviced in period t (t=1,2...T). That is: I_t=$\{i \in V_c \mid \exists k \in \Omega_i\text{: } a_{kt} = 1\}$. Then, the coordinates of the reference point is calculated using the two following formulas:

$$x_t^\nu = \frac{\sum_{i \in I_t} w_{it} x_i}{\sum_{i \in I_t} w_{it}} \qquad \forall \nu = 1, 2..., \forall t \in T; \tag{12}$$

$$y_t^\nu = \frac{\sum_{i \in I_t} w_{it} y_i}{\sum_{i \in I_t} w_{it}} \qquad \forall \nu = 1, 2..., \forall t \in T; \tag{13}$$

In the above equations, w_{it} is defined as a self-adjusting positive weight which reflects the desirability of visiting customer i in period t. The values of these weights are dynamically adjusted, at each iteration, based on the information gathered from the reference set. The adjusting procedure is summarized as follows: If customer i is visited in period t in more than θ% of the solutions existing in the reference set, the value of w_{it} is multiplied by 1+φ, otherwise it remains unchanged, where φ is a positive parameter. It should be noted that, this adjusting procedure starts after the modular heuristic passes a preliminary phase called Warming-up Stage (WS). In WS, the algorithm is repeated in λ successive iterations, $\lambda \geq \gamma$, so that, at each iteration, the weights involved in equations (12) and (13) are randomly generated in the interval (0,1].

The other important parameters having key roles in the integer programming model are the bounds considered in constraints (10). In this paper, LB_t^ν and UB_t^ν are considered as two parameters which are iteratively adjusted based on the information obtained from elite solutions kept in the reference set. Towards this end, LB_t^ν and UB_t^ν are respectively defined, in iteration ν, as the minimum and maximum required capacity for period t observed in elite solutions existing in the reference set. It should be noted that, in the Warming-up Stage described above, Constraints (10) are relaxed by respectively setting LB_t^ν and UB_t^ν to 0 and ∞.

4.4 Depot assignment

The proposed heuristic continues by assigning the customers of each period to depots. The assignment algorithm that we propose in this step belongs to a category of assignment problems which is called assignment by clusters. In this type of assignment problems, a cluster is defined as the set of points consisting of a depot and the customers assigned to it. The algorithms in this class try to build compact clusters of customers for each depot. When a customer is assigned to a cluster it means that this customer is assigned to that cluster's depot. In this study, the way in which customers are incorporated in a cluster is defined by an integer programming model which is mathematically expressed as follows:

$$\textit{min} \sum_{t=1}^{T} \sum_{i \in \Pi_t} \sum_{j=1}^{M} b_{ijt} \{(x_i - x_{D_j})^2 + (y_i - y_{D_j})^2\} \omega_{ijt} \tag{14}$$

subject to

$$\sum_{j=1}^{M} \omega_{ijt} = 1 \qquad \forall t \in T, \forall i \in \Pi_t; \tag{15}$$

$$\omega_{ijt} \in \{0,1\} \qquad \forall t \in T, \forall i \in \Pi_t, \forall j \in D; \tag{16}$$

In the above model, (x_{D_j}, y_{D_j}) is the location of depot j and Π_t is the set of customers to be visited in period t. The decision variable of this model is ω_{ijt}, which is equal to 1 if customer i is assigned to depot j in period t, and 0 otherwise. Moreover, the parameter b_{ijt} is defined as the penalty of assigning customer i to depot j in period t. In fact, the higher b_{ijt} is, the more desirable customer i is not assign to depot j in period t. The objective of the above integer programming model is to assign customers to depots so that the total weighted squared Euclidean distance between the customers and depots is minimized. Moreover, constraints (15) force that each customer to be assigned to exactly one depot in a period.

The strength of the integer programming model is dependent on the penalties involved in the objective function. In this paper, these penalties are considered as self-adjusting parameters which are iteratively updated based on the information obtained from the reference set. The adjusting procedure is summarized as follows: If customer i is assigned to depot j in period t in more than $\theta\%$ of the solutions kept in the reference set, the value of b_{ijt} is divided by $1+\mu$, otherwise it is multiplied by $1+\mu$, where μ is a positive parameter. This updating procedure enables the algorithm to assign customers to better depots as the heuristic gets closer to the termination criteria. It should again be noted that, in the Warming-up Stage, the penalties involved in the objective function are randomly generated in interval (0,1].

4.5 Route design

In this phase, customers assigned to the same depot, in each period, are divided into different routes. Towards this end, a heuristic algorithm, consisting of the following three phases, is implemented in a sequential manner:

1. **Construction phase**- In the first phase, customers of each depot are assigned to a giant tour using the GENIUS heuristic to solve the corresponding treveling salesman problem. GENIUS, proposed by Gendreau et al. [1992], consists of two phases that are implemented in a sequential manner. In the first phase, called GENI, a Hamiltonian cycle is progressively generated by inserting vertices (i.e., customers) one at a time. More precisely, GENI starts with a partial solution consisting of three arbitrarily chosen vertices and, at each iteration, it includes any given vertex between two of its p closest neighbors on the partial cycle. While making an insertion, GENI performs a local reoptimization of the partial cycle. Once all vertices have been inserted, the second phase, named US, is executed as a post-optimization heuristic which successively removes each vertex from the cycle, and reinserts it using GENI.

2. **Splitting phase**- In the second phase, using the optimal splitting procedure proposed by Prins [2004], each constructed giant tour is split into shorter routes, each satisfying the vehicle capacity constraint. In other words, by relaxing the route duration restriction and budget constraint, each giant tour is split to least possible number of shorter routes.

3. **Improving phase**- Finally, in the third phase, a heuristic consisting of two exchange procedures is implemented on each constructed route in order to reduce the route's length and, accordingly, to improve the total traveled distance. One of the main characteristics of the proposed heuristic method is that infeasible solutions are allowed throughout the search. Let us assume that *X* denotes the new solution generated by the search mechanism. Moreover, let $\tau(X)$ denote the total traveled distance of solution *X*, and let $A(X)$ and $B(X)$ denote the total violation of the route-length and ϵ- constraints, respectively. Solution *X* is evaluated by a function, $z(X) = \tau(X) + \alpha A(X) + \beta B(X)$, where α and β are self-adjusting positive parameters. By dynamically adjusting the values of these two parameters, this relaxation mechanism facilitates the exploration of the search space and is particularly useful for tightly constrained instances. Parameter α is adjusted as follows: if there is no violation of the route-length constraints, the value of α is divided by 1+Λ , otherwise it is multiplied by 1+Λ , where Λ is a positive parameter. A similar rule applies also to β with respect to route duration constraint. The two proposed exchange procedures are described as follows:

   - **SWAP procedure**: The first exchange procedure, called SWAP, is performed by choosing two distinct customers i and j, where i= 1,2...n_k-1 and j=i+1, i+2...n_k (n_k is the number of customers that are visited in the kth route), and then exchanging their positions within the route. If the swapping procedure results in a better solution according to the penalty function described above, the positions are exchanged, otherwise the solution remains unchanged. The procedure stops when no more exchanges that result in an improving solution are possible.
   - **INSERT procedure**: The second exchange procedure, called INSERT, is based on removing a customer from one route and inserting it into another route. Towards this end, we first randomly select a customer from the longest route of a depot whose length is defined as t_{max}. Then, the selected customer is removed from its current position and is re-inserted to a new route using the two following methods:
     (a) **Inter-depot method**: In this method, the removed customer is re-inserted into either one of the existing routes or a new route of the same depot from which the customer has been removed.
     (b) **Intra-depot method**: In this method, the depot to which the customer is concurrently assigned is changed to another one and the customer is re-inserted into either one of the existing routes or a new route of the new depot.

     Note that, in both cases, an insertion is called feasible if: 1) It does not violate the vehicle's capacity constraint, and 2) It does not produce a route longer than t_{max}. The position to which the customer is inserted is the one that satisfies the two above conditions and results in the best improvement in the penalty function. This procedure is repeated for a predetermined number of iterations, denoted by σ.

5 Experimental results

In this section, the performance of the proposed MHA is investigated based on three different sets of test problems. The first two sets, each including 10 problem instances,

have been developed by Cordeau et al. [1997] for the PVRP and the MDVRP, respectively, while the last set includes 10 different test problems which have been designed by Vidal et al. [2012] for the MDPVRP. Note that, in all the considered problem instances, the number of vehicles assigned to a depot, which was originally set as a limited and fixed parameter, is ignored. Detailed information on these sets are provided in Subsection 5.2.

The efficiency of the developed heuristic is tested in two different settings that are defined according to how the budget constraint is formulated: either using Rule R_1 or Rule R_2 defined in Section 2. Recall that, in Rule R_1, an upper limit is imposed on the total distance traveled over the planning horizon. As for Rule R_2, an upper bound is enforced on the total distance traveled by the vehicles assigned to each depot in each period.

In both of the above cases, values assigned as the upper bound of the budget constraint may have a major impact on the performance of the proposed modular heuristic algorithm. In this paper, the upper bounds of both rules (R_1 and R_2) are initially set based on the information extracted from the Best Known Solution (BKS) reported by Vidal et al. [2012] for the considered problems. More precisely, the values of ϵ, in Rule R_1, is set to the total traveled distance of the BKS, whereas the value of ϵ_{td}, in Rule R_2, is fixed to the total distance traveled by the vehicles assigned to depot d in period t of the BKS. Then, we systematically vary the values of the upper bound, set on the budget constraint, to investigate how the performance of MHA is affected by tightening or widening the budget constraint.

The proposed algorithm is run on each problem instance, for both budget constraint rules, and its efficiency, in terms of solution quality and computational time, is compared to the Unified Tabu Search (UTS) implemented in Lahrichi et al. [2011]. Note, however, that this algorithm was modified to handle the objective considered in this paper, i.e., minimization of the fleet size. Furthermore, the penalty function used in the algorithm was also modified to include budget constraint violations. Both algorithms have been coded in C++ and executed on a Pentium 4, 2.8 GHz, and Windows XP using 256 MB of RAM.

Different aspects of the experimental results are discussed as follows: In Section 5.1, we first use a well-structured algorithm to calibrate all the parameters involved in the heuristic algorithm. Then, in Section 5.2, computational results are given in details.

5.1 Parameter setting

Like most heuristic and meta-heuristic algorithms, the proposed heuristic method has several parameters that need to be tuned before it can reach good results. The problem then turns into finding best parameter setting for the heuristic to solve the considered problems efficiently and timely. Table 1 provides a summary of all the parameters involved in the algorithm.

There are various different methods in the literature to calibrate parameters used in a heuristic or meta-heuristic algorithm. Coy et al. [2000] designed a procedure based on statistical Design Of Experiments (DOE) that systematically selects high-quality parameter values. The parameter setting procedure has four steps that are implemented in a sequential manner. In the first step, a subset of problems to analyze is chosen from the entire set of problems. In the second step, computational experience is used to select the starting level of each parameter, the range over which each parameter will be varied, and the amount by which each parameter should be changed. In the third step, good parameter settings are selected for each problem in the analysis set using

Table 1: Parameters of the heuristic algorithm

Symbol	Description
γ	Maximum size of the reference set
θ	Threshold defined in Sections 4.3 and 4.4
φ	Factor involved in updating w_{it}
λ	Number of times that WS is repeated
μ	Factor involved in updating b_{ijt}
α, β	Self-adjusting parameters in the penalty function
Λ	Factor involved in updating α and β
σ	Number of times that INSERT is repeated
Ψ	Maximum allowable number of non-improving iterations

fractional factorial design and response surface optimization. Finally, in the fourth step, the parameter values obtained in the third step are averaged to obtain high-quality parameter values. The proposed approach does not use higher-order models (such as quadratic) since different response surfaces are averaged over all considered instances. The authors acknowledged that their method will perform poorly if the representative test problems are not chosen correctly or if the problem class is so broad that it requires very different parameter settings. For a detailed description, see Coy et al. [2000].

In this paper, we adopt the above four-step calibration method to tune the parameters used in the heuristic algorithm. The calibration results for each class, along with the final choice of parameter values, are reported in Table 2.

Table 2: Calibration results

Symbol	PVRP	MDVRP	MDPVRP	Final choice
γ	30	40	50	40
θ	0.2	0.3	0.4	0.3
φ	1	1	1	1
λ	$\lceil \frac{N*M*T}{5} \rceil$	$\lceil \frac{N*M*T}{5} \rceil$	$\lceil \frac{N*M*T}{5} \rceil$	$\lceil \frac{N*M*T}{5} \rceil$
μ	1	1	1	1
α, β	1, 1	1, 1	1, 1	1, 1
Λ	1	1	1	1
σ	N	N	N	N
Ψ	$N*M*T$	$N*M*T$	$N*M*T$	$N*M*T$

5.2 Computational results

We tested the proposed modular heuristic algorithm on the problem instances described at the beginning of this section using the two rules described in Section 2, Rules R_1 and R_2. Detailed computational results on each fold are given in the following subsections.

5.2.1 Rule R_1

In Rule R_1, both the modular heuristic and UTS are run 10 times on each problem instance. Moreover, the maximum running time of both algorithms for the PVRP and

MDVRP instances is set to 20 minutes, while for the MDPVRP instances, due to their greater difficulty, the maximum running time is fixed to 30 minutes.

Results on PVRP instances are presented in Table 3. The first four columns of this table respectively display instance identifier, number of customers, number of depots and number of periods. Moreover, in Column 5, different values of the upper bound, set on the budget constraint, are shown. In this column, ϵ^* refers to the total traveled distance of the BKS. The results of the proposed heuristic method are shown in Columns 6 and 7 as the average fleet size and computational time on 10 independent runs. We compare the performance of our heuristic algorithm to the results obtained with the modified version of the UTS implementation of Lahrichi et al. [2011] (UTS in Columns 8 and 9). Finally, the average percentage gap of the modular heuristic with respect to UTS, on each problem instance and for each value considered for ϵ, is reported in Column 10 (a negative value means a better performance of the modular heuristic).

As shown in Table 3, the proposed modular heuristic algorithm always produces either equivalent or better results compared to UTS. However, the average percentage gap between two algorithms clearly varies depending on values set as the upper bound of the budget constraint. In the case where the ϵ value is set to ϵ^*, the average percentage gap is -3.7% indicating that the modular heuristic performs significantly better than UTS. On the other hand, in the cases where the budget constraint is tightened by fixing the ϵ value to $0.8\epsilon^*$ and $0.9\epsilon^*$, both algorithms face a more challenging task to produce good results and, consequently, their performance slightly worsens. However, the results show that the tighten the budget constraint is, the more the modular heuristic shows its superiority to produce better results. These results reveal this fact that the proposed heuristic algorithm has better capability, compared to UTS, to solve PVRP instances, especially for those problems having more restricted search space. The average percentage gaps between the two algorithms respectively increase to -6.3% and -5.6%, for $0.8\epsilon^*$ and $0.9\epsilon^*$ cases. Contrary, increasing the upper bounds to $1.1\epsilon^*$ and $1.2\epsilon^*$ results in producing better solutions by both algorithms. In these cases, the average percentage gaps respectively change to -3.1% and -5.8%. Figure 1 schematically shows how the average percentage gap varies when changing the ϵ value.

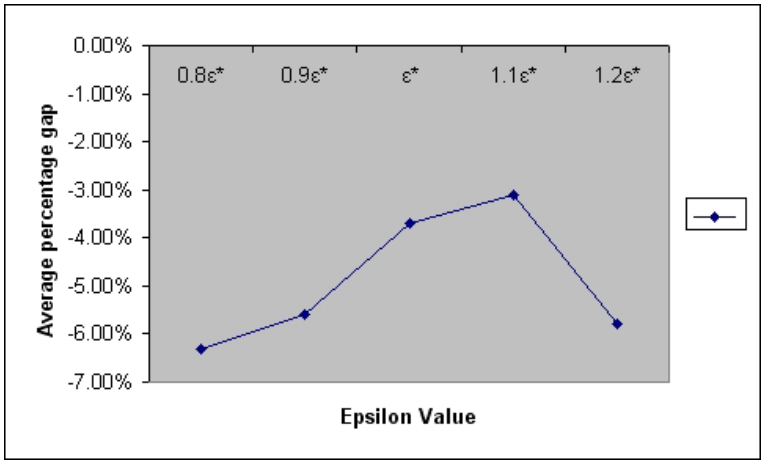

Figure 1: Average percentage gap for the PVRP instances

Results on MDVRP instances are displayed in Table 4, where, as shown in Table 3, Columns 2-4 represents respectively instance identifier, number of customers, number

Table 3: Results on PVRP instances with Rule R_1

Instance	N	M	T	ϵ	Heuristic (Ave.)		UTS (Ave.)		Gap (%)
					K	Time (min)	K	Time (min)	
pr01	48	4	4	$0.8\epsilon^*$	3	0.19	3	0.37	0
				$0.9\epsilon^*$	3	0.18	3	0.35	0
				ϵ^*	2	0.18	2	0.35	0
				$1.1\epsilon^*$	2	0.16	2	0.35	0
				$1.2\epsilon^*$	2	0.16	2	0.33	0
pr02	96	4	4	$0.8\epsilon^*$	5	0.52	5	0.94	0
				$0.9\epsilon^*$	5	0.48	5	0.85	0
				ϵ^*	4	0.46	4	0.80	0
				$1.1\epsilon^*$	4	0.42	4	0.75	0
				$1.2\epsilon^*$	4	0.39	4	0.71	0
pr03	144	4	4	$0.8\epsilon^*$	7	4.73	7	6.69	0
				$0.9\epsilon^*$	7	4.27	7	6.13	0
				ϵ^*	6	3.81	6	5.56	0
				$1.1\epsilon^*$	6	3.27	6	5.38	0
				$1.2\epsilon^*$	5	3.12	6	5.09	-17
pr04	192	4	4	$0.8\epsilon^*$	9	8.33	10	9.57	-10
				$0.9\epsilon^*$	9	8.13	9	9.12	0
				ϵ^*	7	7.42	8	8.19	-13
				$1.1\epsilon^*$	7	7.13	8	8.10	-13
				$1.2\epsilon^*$	7	6.75	8	7.92	-13
pr05	240	4	4	$0.8\epsilon^*$	12	15.80	14	19.91	-14
				$0.9\epsilon^*$	12	15.51	14	19.72	-14
				ϵ^*	10	14.73	10	18.22	0
				$1.1\epsilon^*$	10	14.23	10	17.71	0
				$1.2\epsilon^*$	9	13.79	10	17.24	-10
pr06	288	4	4	$0.8\epsilon^*$	14	16.45	16	19.36	-13
				$0.9\epsilon^*$	13	16.02	16	19.29	-19
				ϵ^*	12	15.51	14	19.11	-14
				$1.1\epsilon^*$	12	15.32	13	18.72	-8
				$1.2\epsilon^*$	12	15.27	13	18.33	-8
pr07	72	6	6	$0.8\epsilon^*$	4	1.29	4	1.89	0
				$0.9\epsilon^*$	4	1.21	4	1.84	0
				ϵ^*	3	1.15	3	1.72	0
				$1.1\epsilon^*$	3	1.11	3	1.65	0
				$1.2\epsilon^*$	3	1.06	3	1.57	0
pr08	144	6	6	$0.8\epsilon^*$	7	5.64	8	7.50	-13
				$0.9\epsilon^*$	7	5.12	7	7.39	0
				ϵ^*	6	4.76	6	7.23	0
				$1.1\epsilon^*$	6	4.55	6	7.14	0
				$1.2\epsilon^*$	6	4.31	6	7.11	0
pr09	216	6	6	$0.8\epsilon^*$	11	17.12	11	18.99	0
				$0.9\epsilon^*$	10	16.70	11	18.96	-10
				ϵ^*	10	16.22	11	18.89	-10
				$1.1\epsilon^*$	10	16.17	11	18.69	-10
				$1.2\epsilon^*$	10	16.09	11	18.44	-10
pr10	288	6	6	$0.8\epsilon^*$	14	19.02	16	19.54	-13
				$0.9\epsilon^*$	14	18.95	16	19.23	-13
				ϵ^*	12	18.88	12	19.01	0
				$1.1\epsilon^*$	12	18.75	12	18.86	0
				$1.2\epsilon^*$	12	18.43	12	18.77	0

of depots and number of periods. The results obtained by the heuristic algorithm are compared, once again, to the modified version of the unified tabu search of Lahrichi et al. [2011], in terms of solution quality and computational time.

The main conclusions derived from Table 4 are similar to those stated above for the PVRP. The modular heuristic clearly outperforms the unified tabu search on the majority of problem instances. Figure 2 represents how the average percentage gap of two algorithms changes when varying the ϵ value.

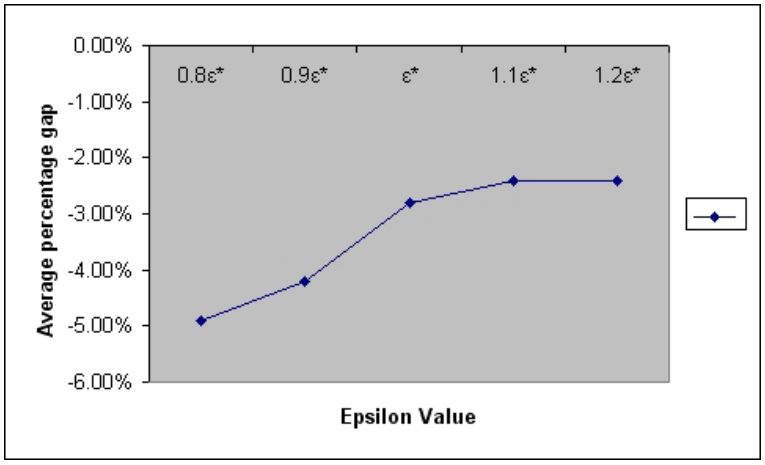

Figure 2: Average percentage gap for the MDVRP instances

Results on MDPVRP instances are finally summarized in Table 5 whose structure is similar to that of Tables 3 and 4. Once again, the results produced by the heuristic algorithm are compared to UTS in order to assess the efficiency of the algorithm, in terms of solution quality and computational time.

Table 5 show that the proposed modular heuristic algorithm is considered as a competitive solution methodology, able to produce high quality solutions in a reasonable time. As for the two previous problems, the relative performance of two algorithms, measured by the average percentage gap, depends upon the on values assigned to ϵ. Figure 3 depicts how different ϵ values affect the average percentage gap.

5.2.2 Rule R_2

In Rule R_2, we investigate how the algorithm reacts when an upper bound is imposed on the total distance that all vehicles assigned to each depot are permitted to travel in each period. The specifications of how the algorithms are applied in the case of Rule R_2 is exactly the same as in the case of Rule R_1 (i.e., the same number of runs is applied as well as identical maximum allotted run-times for the algorithms are considered for each problem).

Tables 6-8 display the results obtained by the heuristic algorithm on the PVRP, MDVRP and MDPVRP problems, respectively. Each of these tables use the same structure considered for the tables of the previous subsection. It should be noted that, in these tables, ϵ^* is a $T \times D$ matrix, in which ϵ_{td} corresponds to the total distance traveled by vehicles assigned to depot d in period t of the BKS. The performance of the heuristic algorithm is compared, on each set of test problems, to the modified version of the unified tabu search implementation of Lahrichi et al. [2011].

Table 4: Results on MDVRP instances with Rule R_1

Instance	N	M	T	ϵ	Heuristic (Ave.)		UTS (Ave.)		Gap (%)
					K	Time (min)	K	Time (min)	
pr01	48	4	4	$0.8\epsilon^*$	5	0.13	5	0.15	0
				$0.9\epsilon^*$	5	0.11	5	0.15	0
				ϵ^*	4	0.09	4	0.14	0
				$1.1\epsilon^*$	4	0.08	4	0.14	0
				$1.2\epsilon^*$	4	0.16	4	0.13	0
pr02	96	4	4	$0.8\epsilon^*$	8	0.41	9	0.81	-11
				$0.9\epsilon^*$	8	0.35	9	0.74	-11
				ϵ^*	7	0.31	8	0.69	-13
				$1.1\epsilon^*$	7	0.30	8	0.62	-13
				$1.2\epsilon^*$	7	0.29	8	0.60	-13
pr03	144	4	4	$0.8\epsilon^*$	13	1.06	14	1.52	-7
				$0.9\epsilon^*$	13	1.02	13	1.44	0
				ϵ^*	12	0.55	12	1.39	0
				$1.1\epsilon^*$	12	0.51	12	1.30	0
				$1.2\epsilon^*$	12	0.48	12	1.26	0
pr04	192	4	4	$0.8\epsilon^*$	16	3.88	17	4.68	-6
				$0.9\epsilon^*$	16	3.82	17	4.60	-6
				ϵ^*	15	3.77	16	4.51	-6
				$1.1\epsilon^*$	15	3.70	16	4.46	-6
				$1.2\epsilon^*$	15	3.64	16	4.39	-6
pr05	240	4	4	$0.8\epsilon^*$	21	7.50	22	9.51	-5
				$0.9\epsilon^*$	21	7.44	22	9.32	-5
				ϵ^*	20	7.29	20	9.16	0
				$1.1\epsilon^*$	20	7.22	20	9.09	0
				$1.2\epsilon^*$	20	7.16	20	9.03	0
pr06	288	4	4	$0.8\epsilon^*$	25	8.94	27	9.62	-7
				$0.9\epsilon^*$	25	8.90	27	9.56	-7
				ϵ^*	24	8.77	24	9.44	0
				$1.1\epsilon^*$	24	8.71	24	9.37	0
				$1.2\epsilon^*$	24	8.63	24	9.28	0
pr07	72	6	6	$0.8\epsilon^*$	7	0.30	7	0.55	0
				$0.9\epsilon^*$	7	0.28	7	0.54	0
				ϵ^*	6	0.26	6	0.51	0
				$1.1\epsilon^*$	6	0.24	6	0.47	0
				$1.2\epsilon^*$	6	0.22	6	0.41	0
pr08	144	6	6	$0.8\epsilon^*$	13	1.68	13	1.97	0
				$0.9\epsilon^*$	13	1.66	13	1.93	0
				ϵ^*	12	1.57	12	1.89	0
				$1.1\epsilon^*$	12	1.51	12	1.83	0
				$1.2\epsilon^*$	12	1.47	12	1.79	0
pr09	216	6	6	$0.8\epsilon^*$	20	7.77	22	8.84	-9
				$0.9\epsilon^*$	20	7.73	22	8.78	-9
				ϵ^*	19	7.68	20	8.71	-5
				$1.1\epsilon^*$	19	7.60	20	8.64	-5
				$1.2\epsilon^*$	19	7.55	20	8.59	-5
pr10	288	6	6	$0.8\epsilon^*$	25	8.80	26	9.73	-4
				$0.9\epsilon^*$	25	8.72	26	9.64	-4
				ϵ^*	24	8.65	24	9.57	0
				$1.1\epsilon^*$	24	8.60	24	9.51	0
				$1.2\epsilon^*$	24	8.53	24	9.46	0

Table 5: Results on MDPVRP instances with Rule R_1

Instance	N	M	T	ϵ	Heuristic (Ave.)		UTS (Ave.)		Gap (%)
					K	Time (min)	K	Time (min)	
pr01	48	4	4	$0.8\epsilon^*$	5	0.32	5	0.43	0
				$0.9\epsilon^*$	5	0.29	5	0.42	0
				ϵ^*	4	0.25	4	0.40	0
				$1.1\epsilon^*$	4	0.22	4	0.38	0
				$1.2\epsilon^*$	4	0.20	4	0.36	0
pr02	96	4	4	$0.8\epsilon^*$	5	0.86	5	1.39	0
				$0.9\epsilon^*$	5	0.81	5	1.34	0
				ϵ^*	4	0.73	4	1.28	0
				$1.1\epsilon^*$	4	0.69	4	1.26	0
				$1.2\epsilon^*$	4	0.64	4	1.19	0
pr03	144	4	4	$0.8\epsilon^*$	9	4.88	9	6.41	0
				$0.9\epsilon^*$	9	4.79	9	6.33	0
				ϵ^*	8	4.72	8	6.23	0
				$1.1\epsilon^*$	8	4.55	8	6.13	0
				$1.2\epsilon^*$	7	4.48	8	6.09	-13
pr04	192	4	4	$0.8\epsilon^*$	9	13.73	10	15.66	-10
				$0.9\epsilon^*$	8	13.62	9	15.58	-10
				ϵ^*	7	13.56	8	15.52	-13
				$1.1\epsilon^*$	7	13.50	8	15.44	-13
				$1.2\epsilon^*$	7	13.41	8	15.37	-13
pr05	240	4	4	$0.8\epsilon^*$	13	20.58	14	24.33	-7
				$0.9\epsilon^*$	13	20.53	14	24.17	-7
				ϵ^*	12	20.44	12	24.09	0
				$1.1\epsilon^*$	12	20.37	12	23.88	0
				$1.2\epsilon^*$	12	20.29	12	23.82	0
pr06	288	4	4	$0.8\epsilon^*$	14	17.41	16	21.45	-13
				$0.9\epsilon^*$	13	17.32	15	21.34	-13
				ϵ^*	12	17.22	14	21.25	-14
				$1.1\epsilon^*$	12	17.16	13	21.20	-8
				$1.2\epsilon^*$	12	17.08	13	21.11	-8
pr07	72	6	6	$0.8\epsilon^*$	7	1.46	7	2.06	0
				$0.9\epsilon^*$	7	1.39	7	1.98	0
				ϵ^*	6	1.35	6	1.96	0
				$1.1\epsilon^*$	6	1.30	6	1.91	0
				$1.2\epsilon^*$	6	1.24	6	1.84	0
pr08	144	6	6	$0.8\epsilon^*$	7	5.25	8	8.04	-13
				$0.9\epsilon^*$	7	5.14	7	7.97	0
				ϵ^*	6	5.06	6	7.91	0
				$1.1\epsilon^*$	6	5.03	6	7.88	0
				$1.2\epsilon^*$	6	4.98	6	7.82	0
pr09	216	6	6	$0.8\epsilon^*$	13	20.42	13	24.19	0
				$0.9\epsilon^*$	13	20.31	14	24.12	-7
				ϵ^*	12	20.19	13	23.96	-8
				$1.1\epsilon^*$	12	20.10	13	23.89	-8
				$1.2\epsilon^*$	12	19.89	13	23.81	-8
pr10	288	6	6	$0.8\epsilon^*$	19	23.14	20	26.79	-5
				$0.9\epsilon^*$	19	23.04	20	26.71	-5
				ϵ^*	18	22.88	18	26.53	0
				$1.1\epsilon^*$	18	22.76	18	26.49	0
				$1.2\epsilon^*$	18	22.70	18	26.41	0

Table 6: Results on PVRP instances with Rule R_2

Instance	N	M	T	ϵ	Heuristic (Ave.)		UTS (Ave.)		Gap (%)
					K	Time (min)	K	Time (min)	
pr01	48	4	4	$0.8\epsilon^*$	3	0.20	3	0.39	0
				$0.9\epsilon^*$	3	0.19	3	0.36	0
				ϵ^*	2	0.19	2	0.35	0
				$1.1\epsilon^*$	2	0.17	2	0.34	0
				$1.2\epsilon^*$	2	0.17	2	0.33	0
pr02	96	4	4	$0.8\epsilon^*$	5	0.54	5	0.94	0
				$0.9\epsilon^*$	5	0.49	5	0.85	0
				ϵ^*	4	0.47	4	0.82	0
				$1.1\epsilon^*$	4	0.44	4	0.77	0
				$1.2\epsilon^*$	4	0.40	4	0.73	0
pr03	144	4	4	$0.8\epsilon^*$	7	4.12	8	6.38	-13
				$0.9\epsilon^*$	7	3.92	8	6.16	-13
				ϵ^*	6	3.83	6	5.59	0
				$1.1\epsilon^*$	6	3.70	6	5.48	0
				$1.2\epsilon^*$	6	3.43	6	5.32	0
pr04	192	4	4	$0.8\epsilon^*$	9	8.39	10	9.61	-10
				$0.9\epsilon^*$	9	8.19	10	9.24	-10
				ϵ^*	8	7.44	8	8.25	0
				$1.1\epsilon^*$	7	7.31	8	8.19	-13
				$1.2\epsilon^*$	7	7.08	8	7.95	-13
pr05	240	4	4	$0.8\epsilon^*$	12	15.93	14	20.19	-14
				$0.9\epsilon^*$	12	15.24	14	18.95	-14
				ϵ^*	10	14.82	11	18.39	-9
				$1.1\epsilon^*$	10	14.73	11	18.11	-9
				$1.2\epsilon^*$	10	14.22	11	17.78	-9
pr06	288	4	4	$0.8\epsilon^*$	14	16.54	16	19.50	-13
				$0.9\epsilon^*$	14	16.09	16	19.42	-13
				ϵ^*	13	15.61	14	19.32	-7
				$1.1\epsilon^*$	13	15.45	14	18.80	-7
				$1.2\epsilon^*$	12	15.39	13	18.53	-7
pr07	72	6	6	$0.8\epsilon^*$	4	1.29	4	1.93	0
				$0.9\epsilon^*$	4	1.24	4	1.86	0
				ϵ^*	3	1.18	3	1.77	0
				$1.1\epsilon^*$	3	1.14	3	1.69	0
				$1.2\epsilon^*$	3	1.11	3	1.60	0
pr08	144	6	6	$0.8\epsilon^*$	7	5.79	8	7.53	-13
				$0.9\epsilon^*$	7	5.19	7	7.44	0
				ϵ^*	6	4.82	6	7.39	0
				$1.1\epsilon^*$	6	4.60	6	7.22	0
				$1.2\epsilon^*$	6	4.49	6	7.16	0
pr09	216	6	6	$0.8\epsilon^*$	11	17.19	12	19.12	0
				$0.9\epsilon^*$	10	16.76	12	19.04	-17
				ϵ^*	10	16.29	11	18.96	-10
				$1.1\epsilon^*$	10	16.20	11	18.78	-10
				$1.2\epsilon^*$	10	16.11	11	18.53	-10
pr10	288	6	6	$0.8\epsilon^*$	14	19.07	16	19.61	-13
				$0.9\epsilon^*$	14	18.99	16	19.44	-13
				ϵ^*	12	18.94	13	19.10	-8
				$1.1\epsilon^*$	12	18.83	13	18.97	-8
				$1.2\epsilon^*$	12	18.53	13	18.86	-8

Table 7: Results on MDVRP instances with Rule R_2

Instance	N	M	T	ϵ	Heuristic (Ave.)		UTS (Ave.)		Gap (%)
					K	Time (min)	K	Time (min)	
pr01	48	4	4	$0.8\epsilon^*$	5	0.14	5	0.17	0
				$0.9\epsilon^*$	5	0.12	5	0.17	0
				ϵ^*	4	0.10	4	0.16	0
				$1.1\epsilon^*$	4	0.09	4	0.14	0
				$1.2\epsilon^*$	4	0.07	4	0.14	0
pr02	96	4	4	$0.8\epsilon^*$	9	0.43	10	0.84	0
				$0.9\epsilon^*$	9	0.36	9	0.77	-10
				ϵ^*	8	0.33	8	0.72	0
				$1.1\epsilon^*$	8	0.31	8	0.66	0
				$1.2\epsilon^*$	8	0.30	8	0.62	0
pr03	144	4	4	$0.8\epsilon^*$	13	1.08	14	1.55	-7
				$0.9\epsilon^*$	13	1.04	13	1.45	0
				ϵ^*	12	0.59	12	1.44	0
				$1.1\epsilon^*$	12	0.59	12	1.35	0
				$1.2\epsilon^*$	12	0.48	12	1.29	0
pr04	192	4	4	$0.8\epsilon^*$	17	3.90	18	4.72	-6
				$0.9\epsilon^*$	17	3.85	18	4.66	-6
				ϵ^*	16	3.80	17	4.60	-6
				$1.1\epsilon^*$	16	3.74	17	4.52	-6
				$1.2\epsilon^*$	16	3.70	17	4.47	-6
pr05	240	4	4	$0.8\epsilon^*$	21	7.53	22	9.59	-5
				$0.9\epsilon^*$	21	7.49	22	9.35	-5
				ϵ^*	20	7.34	21	9.33	-5
				$1.1\epsilon^*$	20	7.29	21	9.19	-5
				$1.2\epsilon^*$	20	7.22	21	9.14	-5
pr06	288	4	4	$0.8\epsilon^*$	25	9.01	27	9.84	-7
				$0.9\epsilon^*$	25	8.97	27	9.60	-7
				ϵ^*	24	8.80	25	9.49	-4
				$1.1\epsilon^*$	24	8.75	25	9.40	-4
				$1.2\epsilon^*$	24	8.69	25	9.33	-4
pr07	72	6	6	$0.8\epsilon^*$	7	0.33	7	0.64	0
				$0.9\epsilon^*$	7	0.29	7	0.59	0
				ϵ^*	6	0.27	6	0.54	0
				$1.1\epsilon^*$	6	0.25	6	0.49	0
				$1.2\epsilon^*$	6	0.24	6	0.46	0
pr08	144	6	6	$0.8\epsilon^*$	13	1.73	13	2.04	0
				$0.9\epsilon^*$	13	1.69	13	1.99	0
				ϵ^*	12	1.62	12	1.95	0
				$1.1\epsilon^*$	12	1.56	12	1.88	0
				$1.2\epsilon^*$	12	1.50	12	1.84	0
pr09	216	6	6	$0.8\epsilon^*$	20	7.88	22	8.91	-9
				$0.9\epsilon^*$	20	7.79	22	8.84	-9
				ϵ^*	19	7.77	21	8.79	-10
				$1.1\epsilon^*$	19	7.66	21	8.74	-10
				$1.2\epsilon^*$	19	7.59	21	8.63	-10
pr10	288	6	6	$0.8\epsilon^*$	25	8.85	26	9.82	-4
				$0.9\epsilon^*$	25	8.80	26	9.72	-4
				ϵ^*	24	8.70	25	9.65	-4
				$1.1\epsilon^*$	24	8.64	25	9.55	-4
				$1.2\epsilon^*$	24	8.59	25	9.48	-4

Table 8: Results on MDPVRP instances with Rule R_2

Instance	N	M	T	ϵ	Heuristic (Ave.)		UTS (Ave.)		Gap (%)
					K	Time (min)	K	Time (min)	
pr01	48	4	4	$0.8\epsilon^*$	5	0.35	5	0.50	0
				$0.9\epsilon^*$	5	0.31	5	0.47	0
				ϵ^*	4	0.28	4	0.44	0
				$1.1\epsilon^*$	4	0.25	4	0.41	0
				$1.2\epsilon^*$	4	0.22	4	0.37	0
pr02	96	4	4	$0.8\epsilon^*$	5	0.89	5	1.43	0
				$0.9\epsilon^*$	5	0.84	5	1.38	0
				ϵ^*	4	0.77	4	1.31	0
				$1.1\epsilon^*$	4	0.71	4	1.28	0
				$1.2\epsilon^*$	4	0.67	4	1.22	0
pr03	144	4	4	$0.8\epsilon^*$	9	4.89	10	6.46	0
				$0.9\epsilon^*$	9	4.83	10	6.41	-10
				ϵ^*	8	4.75	9	6.29	-11
				$1.1\epsilon^*$	8	4.62	9	6.18	-11
				$1.2\epsilon^*$	8	4.53	9	6.11	-11
pr04	192	4	4	$0.8\epsilon^*$	9	13.76	9	15.70	0
				$0.9\epsilon^*$	9	13.69	9	15.66	0
				ϵ^*	8	13.62	8	15.64	0
				$1.1\epsilon^*$	7	13.56	8	15.49	-13
				$1.2\epsilon^*$	7	13.45	8	15.53	-13
pr05	240	4	4	$0.8\epsilon^*$	13	20.65	14	24.55	-7
				$0.9\epsilon^*$	13	20.59	14	24.48	-7
				ϵ^*	12	20.53	13	24.33	-8
				$1.1\epsilon^*$	12	20.42	13	23.94	-8
				$1.2\epsilon^*$	12	20.35	13	23.85	-8
pr06	288	4	4	$0.8\epsilon^*$	14	17.47	16	21.51	-13
				$0.9\epsilon^*$	14	17.35	16	21.40	-13
				ϵ^*	13	17.30	14	21.34	-7
				$1.1\epsilon^*$	13	17.41	14	21.29	-7
				$1.2\epsilon^*$	13	17.23	14	21.20	-7
pr07	72	6	6	$0.8\epsilon^*$	7	1.49	7	2.09	0
				$0.9\epsilon^*$	7	1.44	7	2.06	0
				ϵ^*	6	1.40	6	2.01	0
				$1.1\epsilon^*$	6	1.34	6	1.97	0
				$1.2\epsilon^*$	6	1.29	6	1.89	0
pr08	144	6	6	$0.8\epsilon^*$	7	5.28	8	8.09	-13
				$0.9\epsilon^*$	7	5.19	7	8.04	0
				ϵ^*	6	5.18	6	7.95	0
				$1.1\epsilon^*$	6	5.11	6	7.94	0
				$1.2\epsilon^*$	6	5.04	6	7.88	0
pr09	216	6	6	$0.8\epsilon^*$	14	20.45	15	24.22	-7
				$0.9\epsilon^*$	14	20.35	15	24.17	-7
				ϵ^*	13	20.31	13	24.12	0
				$1.1\epsilon^*$	13	20.23	13	23.95	0
				$1.2\epsilon^*$	13	20.07	13	23.87	0
pr10	288	6	6	$0.8\epsilon^*$	19	23.19	20	26.86	-5
				$0.9\epsilon^*$	19	23.12	19	26.76	0
				ϵ^*	18	22.94	18	26.69	0
				$1.1\epsilon^*$	18	22.83	18	26.57	0
				$1.2\epsilon^*$	18	22.77	18	26.46	0

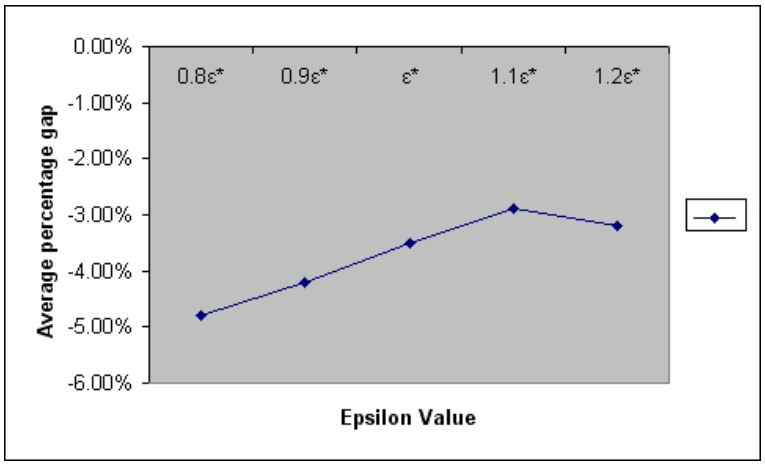

Figure 3: Average percentage gap for the MDPVRP instances

As shown in Tables 6-8, the proposed modular heuristic algorithm performs impressively well relative to the UTS, in terms of solution quality and computational time. For each of the problems considered, we investigated how the existing average percentage gap of two algorithms is affected by different ϵ values. The results obtained are shown by Figure 4.

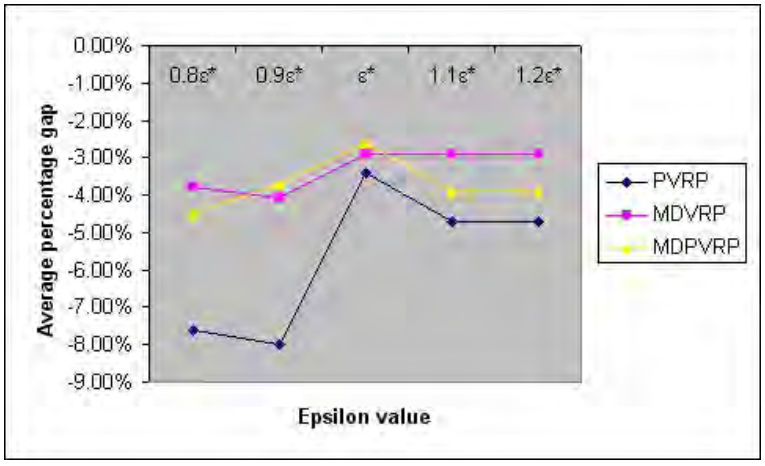

Figure 4: Average percentage gap

The results of Tables show that restricting the search space through bounding the total distance allowed to be traveled by vehicles assigned to a depot in each period may result in a decrease of the quality of the heuristic algorithm.

6 Conclusions

This paper presented a new modular heuristic algorithm for addressing several classes of multi-depot and periodic vehicle routing problems. In each of the considered prob-

lem classes, the goal is to determine the optimal fleet size when three constraints, i.e., vehicle capacity, route duration and budget constraints, are to be satisfied.

This paper introduced several methodological contributions, particularly, a self-learning mechanism that leads the algorithm to assign better visit patterns to customers, and also to assign customers to better depots as the solution process evolves. This learning mechanism, in addition to other components of the algorithm, provided the capability of the heuristic algorithm to reach high quality solutions.

To validate the efficiency of the proposed heuristic algorithm, different test problems, existing in the literature, were solved. The computational results showed that the proposed algorithm performs very well, for all problem instances.

Acknowledgements

Partial funding for this project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair, Collaborative Research and Development, and Discovery Grant programs, by the Fonds de Recherche du Québec - Nature et Technologies (FRQ-NT) through its Team Research Project program, and by the EMME/2-STAN Royalty Research Funds (Dr. Heinz Spiess). We also gratefully acknowledge the support of Fonds de recherche du Québec through their infrastructure grants.

References

J. Alegre, M. Laguna, and J. Pacheco. Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts. *European Journal of Operational Research*, 179 (3):736 – 746, 2007.

F. Alonso, M. J. Alvarez, and J. E. Beasley. A tabu search algorithm for the periodic vehicle routing problem with multiple vehicle trips and accessibility restrictions. *Journal of The Operational Research Society*, 59:963–976, 2008.

E. Angelelli and M. G. Speranza. The periodic vehicle routing problem with intermediate facilities. *European Journal of Operational Research*, 137(2):233 – 247, 2002.

L. Bertazzi, G. Paletta, and M. G. Speranza. An improved heuristic for the period traveling salesman problem. *Computers & Operations Research*, 31:1215–1222, 2004.

P. J. Cassidy and H. S. Bennett. TRAMPA Multi-depot Vehicle Scheduling System. *Journal of The Operational Research Society*, 23:151–163, 1972.

I.M Chao, B. L. Golden, and E. Wasil. A new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions. *American Journal of Mathematical and Management Sciences*, 13(3-4):371–406, 1993.

N. Christofides and J. E. Beasley. The period routing problem. *Networks*, 14(2):237–256, 1984.

G. Clarke and J. W. Wright. Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, 12:568–581, 1964.

J.F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30:105–119, 1997.

S. P. Coy, B. L. Golden, G. C. Runger, and E. A. Wasil. Using Experimental Design to Find Effective Parameter Settings for Heuristics. *Journal of Heuristics*, 7:77–97, 2000.

T. G. Crainic, G. C.Crisan, M. Gendreau, N. Lahrichi, and W. Rei. A concurrent evolutionary approach for rich combinatorial optimization. In *Genetic and Evolutionary Computation Conference*, pages 2017–2022, 2009.

G. B. Dantzig and J. H. Ramser. The Truck Dispatching Problem. *Management Science*, 6:80–91, 1959.

R. Dondo and J. Cerdà. A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows. *European Journal of Operational Research*, 176:1478–1507, 2007.

L. A. Drummond, L. S. Ochi, and D. S. Vianna. An asynchronous parallel metaheuristic for the period vehicle routing problem. *Future Generation Computer Systems*, 17: 379–386, 2001.

M. L. Fisher and R. Jaikumar. A generalized assignment heuristic for the vehicle routing problem. In *Conference on Computer Networks*, 1981.

M. Gendreau, A. Hertz, and G. Laporte. New Insertion and Postoptimization Procedures for the Traveling Salesman Problem. *Operations Research*, 40:1086–1094, 1992.

B. E. Gillett and L. R. Miller. A Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research*, 22:340–349, 1974.

B. L. Golden, T. L. Magnanti, and H. Q. Nguyen. Implementing vehicle routing algorithms. *Networks*, 7:113–148, 1977.

D. Gulczynski, B.L. Golden, and E.A. Wasil. The period vehicle routing problem: New heuristics and real-world variants. *Transportation Research Part E-logistics and Transportation Review*, 47:648–668, 2011.

E. Hadjiconstantinou and R. Baldacci. A multi-depot period vehicle routing problem arising in the utilities sector. *Journal of The Operational Research Society*, 49: 1239–1248, 1998.

V. C. Hemmelmayr, K. F. Doerner, and R. F. Hartl. A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195:791–802, 2009.

K.H. Kang, Y.H. Lee, and B.K. Lee. *An Exact Algorithm for Multi Depot and Multi Period Vehicle Scheduling Problem*. 2005.

N. Lahrichi, T.G. Crainic, M. Gendreau, W. Rei, and L.M. Rousseau. Solving the dairy problem for the province of Quebec. *CIRRELT*, 34, 2011.

H.C.W Lau, T.M. Chan, W.T. Tsui, and W.K. Pang. Application of genetic algorithms to solve the multidepot vehicle routing problem. *Automation Science and Engineering, IEEE Transactions on*, 7(2):383–392, 2010.

P. Parthanadee and R. Logendran. Periodic product distribution from multi-depots under limited supplies. *IIE Transactions*, 38:1009–1026, 2006.

S. Pirkwieser and G. R. Raidl. *Multilevel Variable Neighborhood Search for Periodic Routing Problems*. 2010.

C. Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31:1985–2002, 2004.

O. M. Raft. A modular algorithm for an extended vehicle scheduling problem. *European Journal of Operational Research*, 11:67–76, 1982.

A. Rahimi-Vahed, T.G. Crainic, M. Gendreau, and W. Rei. A path relinking algorithm for a multi-depot periodic vehicle routing problem. *CIRRELT*, 2012.

J. Renaud, G. Laporte, and F. F. Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research*, 23:229–235, 1996.

R. Russell and D. Gribbin. A multi-phase approach to the period routing problem. *Networks*, 21:747–765, 1991.

S. Salhi and M. Sari. A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, 103:95–112, 1997.

C. Tan and J. Beasley. A heuristic algorithm for the period vehicle routing problem. *Omega-international Journal of Management Science*, 12:497–504, 1984.

F. A. Tillman. The Multiple Terminal Delivery Problem with Probabilistic Demands. *Transportation Science*, 3:192–204, 1969.

F. A. Tillman and T.M. Cain. An upper bound algorithm for the single and multiple terminal delivery problem. *Management Science*, 18:664–682, 1972.

F. A. Tillman and R.W. Hering. A study of a look-ahead procedure for solving the multiterminal delivery problem. *Transportation Research*, 5:225–229, 1971.

T. Vidal, T.G. Crainic, M. Gendreau, N. Lahrichi, and W. Rei. A Hybrid Genetic Algorithm for Multi-Depots and Periodic Vehicle Routing Problems. *Operations Research*, 60:611–624, 2012.

B. Yu, Z.Z. Yang, and J.X. Xie. A parallel improved ant colony optimization for multi-depot vehicle routing problem. *Journal of the Operational Research Society*, 62: 183–188, 2011.