CIRRELT

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

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation

A Path Relinking Algorithm for a Multi-Depot Periodic Vehicle Routing Problem

Alireza Rahimi Vahed
Teodor Gabriel Crainic
Michel Gendreau
Walter Rei

September 2012

CIRRELT-2012-50

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

A Path Relinking Algorithm for a Multi-Depot Periodic Vehicle Routing Problem

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 consider a variant of vehicle routing problems which is characterized by the presence of a homogeneous fleet of vehicles, multiple depots, multiple periods and two kinds of constraints that are often found in reality, i.e., vehicle capacity and route duration constraints. The objective is to minimize total travel costs. Since the Vehicle Routing Problem has been proved to be NP-hard in the strong sense, an effective Path Relinking Algorithm (PRA) is designed for finding the best possible solutions to this problem. The proposed PRA incorporates several purposeful exploitation and exploration strategies that enable the algorithm to tackle the problem in two different settings: 1) As a stand-alone algorithm, and 2) As a part of a co-operative search algorithm called Integrative Co-operative Search (ICS). The performance of the proposed Path Relinking Algorithm is evaluated, in each of the above ways, based on various test problems. The computational results show that the developed PRA performs impressively, in both solution quality and computational efficiency.

Keywords: Vehicle routing problems, path relinking, integrative co-operative search.

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

The vehicle routing problem (VRP), introduced by Dantzig and Ramser (1959), is one of the most important and widely studied combinatorial optimization problems, with many real-life applications in distribution and transportation logistics. In the classical 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 (Toth and Vigo (2002)).

Several variations and specializations of the vehicle routing problem, each reflecting various real-life applications, exist. However, surveying the literature, one can notice that not all VRP variants have been studied with the same degree of attention in the past five decades. This is the case for the problem considered in this study. Moreover, 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.

Our objective is to contribute toward addressing these two challenges. In this paper, we address a variant of the VRP in which a daily plan is computed for a homogeneous fleet of vehicles that depart from different depots and must visit a set of customers for delivery operations in a planning horizon. In this VRP, we consider maximum route duration constraint and an upper limit of the quantity of goods that each vehicle can transport. Moreover, the cost of each vehicle route is computed through a system of fees depending on the distance that is traveled. This type of vehicle routing problem is typically called the Multi-depot Periodic Vehicle Routing Problem (MDPVRP).

To tackle the MDPVRP, we propose a new Path Relinking Algorithm, which incorporates exploitation and exploration strategies allowing the algorithm to solve the considered problem in two different manners: 1) As a stand-alone algorithm, and 2) As a part of a cooperative search method named as Integrative Cooperative Search (ICS).

The remainder of this paper is organized as follows. The problem statement is introduced in Section 2. The literature survey relevant to the topic of this paper is presented in Section 3. Section 4 deals with the proposed Path Relinking Algorithm. The experimental results are reported in Section 5. Finally, Section 6 provides conclusions and the evaluation of the work.

2 Problem statement

In this section, we formally state the MDPVRP, introducing the notations used throughout this paper. The MDPVRP can be defined as follows (Vidal et al. (2010)): Consider an undirected graph $G(V,E)$. The node set V is the union of two subsets $V = V_C \cup V_D$, where $V_C = \{v_1, ..., v_n\}$ represents the customers and $V_D = \{v_{n+1}, ..., v_{n+m}\}$ 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 time for arriving to node j from node i (t_{ij}) is considered equal to c_{ij}. A limited number (K) of homogeneous vehicles of known capacity Q is available at each depot. Moreover, the MDPVRP has a planning horizon, say *T* periods. Each customer *i* is characterized by a service frequency f_i, stating how often within these *T* periods this customer must be visited and a list L_i of possible visit-period combinations, called patterns. Each vehicle performs only one route per period and each vehicle route must start and finish at the same depot while the travel duration of the route should not exceed *D*. The MDPVRP aims to design a set of vehicle routes servicing all customers, such that vehicle-capacity and route-duration constraints are respected, and the total travel cost is minimized.

3 Literature review

In this section, we focus our attention on reviewing papers previously published in the literature to address the MDPVRP. The objective of this review is first to present what types of solution methodologies have been proposed to solve the considered problem and second, to distinguish leading solution approaches that have been proved to be efficient to tackle the MDPVRP.

By surveying the literature, one notices that the most common solution approach for solving this type of the VRP is to apply a successive-optimization approach which sequentially solves a series of particular cases instead of considering the problem as a whole. This procedure usually leads to suboptimal solutions. Solution algorithms, belonging to this category, can be divided into two groups, i.e., exact methods and heuristics. To the best of our knowledge, the only exact method used to solve the MDPVRP was the one designed by Mingozzi (2005). In the proposed method, first, an integer programming model which is an extension of the set partitioning formulation of the CVRP is described. Then, an exact method for solving the problem, which uses variable pricing in order to reduce the set of variables to more practical proportions, is proposed. The pricing model is based on the bounding procedure for finding near optimal solutions of the dual problem of the LP relaxation of the proposed integer programming model. The bounding procedure is an additive procedure that determines a lower bound on the MDPVRP as the sum of the dual solution costs obtained by a sequence of five different heuristics for solving the dual problem, where each heuristic explores a different structure of the MDPVRP. Three of these heuristics are based on relaxations, whereas the two others combine subgradient optimization with column generation. We also aware of three heuristic algorithms in this category. Hadjiconstantinou and Baldacci (1998) addressed the Multi-Depot Periodic VRP with Time Windows (MDPVRPTW). The authors proposed a multi-phase optimization problem and solved it using a four-phase algorithm. They developed a tabu search algorithm which solves the VRPTW and improved the solutions obtained during the optimization process using a 3-opt procedure. The last phase is the only one that modifies the depot and visit combination pattern assignments. Kang et al. (2005) studied the problem considered by Hadjiconstantinou and Baldacci (1998). The authors developed a two-phase solution method in which all feasible schedules are generated from each depot for each period and the set of routes are determined by solving the shortest path problem. Parthanadee and Logendran (2006) also solved the problem considered by Hadjiconstantinou and Baldacci (1998) using a tabu search. In this algorithm, all the initial assignments are built by cheapest insertion. At the improvement phase, depot and delivery pattern interchanges are used.

Another type of solution approaches more recently used to solve the MDPVRP target the problem as a whole by simultaneously considering all its characteristics. Crainic et al. (2009) proposed a well structured parallel cooperative search method, called

Integrative Co-operative Search (ICS), to solve combinatorial optimization problems. The proposed ICS framework relies on an attribute decomposition approach and its structure is similar to a self-adaptive evolutionary meta-heuristic evolving several independent populations, where one population corresponds to the solutions of the main problem whereas the others consist of the solutions addressing specific dimensions of the problem. The authors used the MDPVRP with time windows to illustrate the applicability of the developed methodology. Vidal et al. (2010) proposed a hybrid Genetic Algorithm (GA) to solve 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.

This brief review supports the general statement made in Section 1 that the MDPVRP is among the VRP variants which did not not receive an adequate degree of attention and the solution algorithms proposed to solve the MDPVRP are scarce. Moreover, solution methodologies which solve the MDPVRP as a whole by simultaneously considering all its characteristics are scarcer. To contribute toward addressing these two challenges, we develop a Path Relinking Algorithm to efficiently address the MDPVRP as a whole. The proposed Path Relinking Algorithm is described in the next section.

4 The Path Relinking Algorithm (PRA)

In recent years, meta-heuristic algorithms, especially population-based ones, have been applied with success to a variety of hard optimization problems. Among the population-based meta-heuristics, the PRA is known as a powerful solution methodology which solves a given problem using purposeful and non-random exploration and exploitation strategies (Glover and Laguna (2000)). The general concepts and principles of a Path Relinking are first described in Section 4.1. Then, the main components of the PRA proposed to solve the MDPVRP are explained in details in Section 4.1.

4.1 The Path Relinking Algorithm in general

The Path Relinking Algorithm has been suggested as an approach to integrate intensification and diversification strategies in the context of tabu search (Glover and Laguna (2000)). The PRA can be considered as an evolutionary algorithm where solutions are generated by combining elements from other solutions. Unlike other evolutionary methods, such as genetic algorithms, where randomness is a key factor in the creation of offsprings from parent solutions, Path Relinking systematically generates new solutions by exploring paths that connect elite solutions. To generate the desired paths, an initial solution and a guiding solution are selected from a so-called reference list of elite solutions to represent the starting and the ending points of the path. Attributes from the guiding solution are gradually introduced into the intermediate solutions, so that these solutions contain less characteristics from the initial solution and more from the guiding solution as one moves along the path.

Based on the description mentioned above, the main components of the general Path Relinking Algorithm are summarized as follows:

1. Rules for building the reference set
2. Rules for choosing the initial and guiding solutions

3. A neighbourhood structure for moving along paths

Algorithm 1 shows a simple Path Relinking procedure presenting how the above-mentioned different components interact.

Algorithm 1 The general Path Relinking Algorithm

1: Generate a starting set of solutions.
2: Designate a subset of solutions to be included in the reference list.
While the cardinality of the reference list > 1
- Select two solutions from the reference list;
- Identify the initial and guiding solutions;
- Remove the initial solution from the reference list;
- Move from the initial toward the guiding solution, generating intermediate solutions.
- Update the reference list;

3: Verify stopping criterion: Stop or go to 1.

To the best of our knowledge, the only Path Relinking Algorithm proposed to solve a VRP was developed by Ho and Gendreau (2006). The authors suggested a hybrid algorithm that uses tabu search and Path Relinking to solve the capacitated vehicle routing problem. The major novelty of this paper is to introduce an assignment problem to determine similarities and differences in the structure of initial and guiding solutions in the proposed Path Relinking. The assignment problem considered in this paper can be briefly described as follows. Once the initial and guiding solutions are selected, a matching of their routes is performed. The matching procedure amounts to solving an assignment problem on an auxiliary complete bipartite graph $G = (V, E)$, where $V = V_i \cup V_g$ and the vertices of V_i and V_g correspond to the routes of the initial and of the guiding solutions, respectively. To each edge $(k, l) \in E$ is associated a weight c_{k_l} , which is defined as the number of identical customers in routes k and l of the two solutions. The aim of this algorithm is to find a matching of the routes of the two solutions such that the number of identical customers in matched routes is maximized. In the Path Relinking phase, a moving mechanism ensures that the algorithm is making progress towards the guiding solution. Towards this end, the routes of the guiding solution are first relabeled in accordance with the matching of the routes determined previously, i.e., if the matching problem determines that route l of the guiding solution is matched by route k of the initial solution, label k is assigned to route l of the guiding solution. Then, a moving mechanism consisting of two neighbourhood search algorithms is used. The first neighbourhood is made up of all the potential solutions that can be reached from the current solution by moving customers from their current route to another while taking into account the structure of the guiding solution. Similarly, the second neighbourhood is defined as the set of all potential solutions that can be reached from the current solution by exchanging two customers between their respective routes while taking into account the structure of the guiding solution.

4.2 The proposed Path Relinking Algorithm

4.2.1 General idea

The Path Relinking Algorithm proposed in this paper relies on an easy-to-build and efficient reference list evolving several independent subsets, where one subset, called complete set, corresponds to elite solutions of the main problem while the others, named as partial sets, consist of elite solutions addressing specific dimensions of the problem. The cooperation between the sets of the reference list is set up by means of information exchange, through the searching mechanism of the PRA.

To construct such a reference list, the MDPVRP is first decomposed into two VRPs with fewer attributes, i.e., PVRP and MDVRP, by respectively fixing the attributes "multiple depots" and "multiple periods". Each of the constructed sub-problems is then solved by a dedicated solution algorithm which is called partial solver. The main advantage of applying such a decomposition procedure is that working on selected attribute subsets, instead of considering all attributes at a time, provides relatively high-quality solutions rapidly. Furthermore, well-known solution methodologies found in the literature may be used to solve sub-problems.

Elite solutions obtained by each partial solver are sent to a partial set of the reference list. The partial sets can be either kept unchanged in the course of the PRA or iteratively updated in order to include better solutions, in terms of both solution quality and diversification level, as the algorithm reaches the termination criterion. We respectively call these two possibilities as static and dynamic scenarios. Challenges, advantages and deficiencies of each scenario are thoroughly discussed in Section 5.

After constructing the initial partial sets, the proposed Path Relinking Algorithm starts to construct high-quality solutions of the main problem by exploring trajectories that connect solutions selected from the reference list. Towards this end, several selection strategies, each choosing initial and guiding solutions in a different manner, are first implemented. Then, for each selected initial and guiding solutions, a neighbourhood search generates a sequence of high-quality complete solutions using the information shared by the selected solutions. Elite solutions generated by the searching mechanism of the PRA are kept in the complete set of the reference list in order to avoid losing high-quality and diverse solutions.

Two special variants of the proposed Path Relinking Algorithm explained above can be obtained by respectively ignoring complete and partial sets. In the former case, the PRA generates complete solutions only based on the information gathered from partial solutions, while, in the latter case, the developed algorithm is converted to a general Path Relinking whose main characteristics have been described in Section 4.1.

Note that, throughout this paper, we use the term "partial" for the solutions obtained by the partial solvers only in order to distinguish between these solutions and the solutions generated in the Path Relinking Algorithm and it does not imply that these solutions are not complete and feasible for the main problem, i.e., MDPVRP. Different components of the proposed Path Relinking Algorithm are described in the following subsections.

4.2.2 Search space

It is well known in the meta-heuristic literature that allowing the search to enter infeasible regions may result in generating high-quality and diverse feasible solutions. One of the main characteristics of the proposed PRA is that infeasible solutions are allowed throughout the search. Let us assume that x denotes the new solution generated by the searching mechanism. Moreover, let $c(x)$ denote the travel cost of solution x, and let $q(x)$ and $t(x)$ denote the total violation of the load and duration constraints, respectively. Solution x is evaluated by a cost function $z(x) = c(x) + \alpha q(x) + \beta t(x)$, 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 capacity constraints, the value of α is divided by 1+γ , otherwise it is multiplied by 1+γ , where γ is a positive parameter. A similar rule applies also to β respect to route duration constraint.

4.2.3 Solution representation

One of the most important decisions in designing a meta-heuristic lies in deciding how to represent solutions and relate them in an efficient way to the searching space. Representation should be easy to decode to reduce the cost of the algorithm. In the proposed Path Relinking Algorithm, a path representation based on the method proposed by Kytöjoki et al. (2005) is used to encode the solution of the MDPVRP. The idea of the path representation is that the customers are listed in the order in which they are visited. To explain this solution representation, let us consider the following example: Suppose that there are four customers numbered 1-4 which have to be visited by two depots in two periods. Moreover, let us assume that the two first customers are served by the first depot, whereas the two last ones are visited by the second depot. Besides that, all customers need to be visited in each period. Figure 1 shows how a solution of the problem described above is represented.

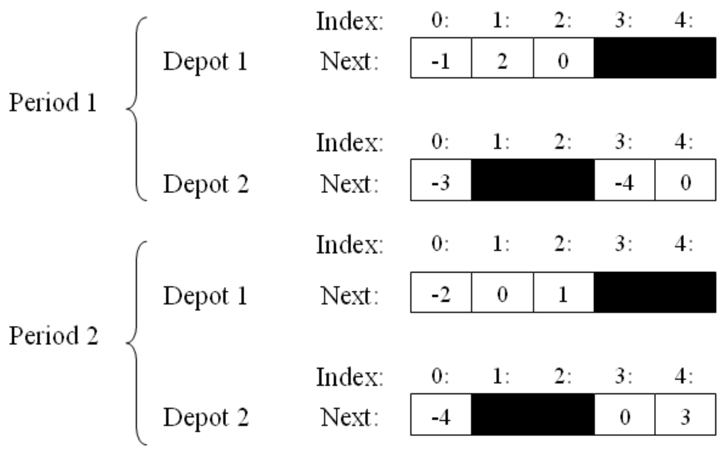

Figure 1: An example of the solution representation

As depicted in Figure 1, in this kind of representation, a single row array of the size equal to n+1 is generated for each depot in each period. Note that 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. For example, in Figure 1, the value "2" has been assigned to the second position (index 1) of the first array. It means that the second customer is immediately visited after the first customer by a vehicle departed from the first depot. In this path representation, negative values give the beginning of the next route index, 0 refers to the end of the routes and a vacant position (drawn as a black box in Figure 1) reveals that the customer corresponding to that position is not served by the depot to which the array belongs. Using this representation, changes to the solution can be performed very quickly. For example, the insertion of a new customer *k* between two adjacent customers a and b is done simply by changing the "next-values" of *k* to *b* and *a* to *k*. Similarly, one can delete a customer or reverse part of a route very quickly (Kytöjoki et al. (2005)).

4.2.4 Constructing the initial reference list

The reference list is a collection of high-quality solutions that are used to generate new solutions by way of applying the searching mechanism of the Path Relinking Algorithm. What solutions are included in the reference list, how good and how diversified they are, have a major impact on the quality of the new generated solutions (Ghamlouche et al. (2004)). Based on the descriptions mentioned in Section 4.2.1, the reference list implemented in the PRA consists of three different subsets where the first two subsets are the partial sets, each keeping elite partial solutions generated by a dedicated partial solver, while the last subset is the complete set consisting of elite solutions of the main problem. Note that, in the proposed algorithm, the maximum size of each subset is fixed to a predetermined value shown by R_{max}.

For the sake of the following descriptions, let us define first the following notations:

- Φ^i: the set of partial solutions added to the *i*th partial set of the reference list,
- Ψ^i: the set of whole partial solutions generated by the *i*th partial solver,
- ϕ^i_j: the *j*th partial solution of Φ^i,
- ψ^i_k: the *k*th solution of Ψ^i.

The construction of the initial reference list starts by adding R_{max} elite partial solutions existing in Ψ^i $(i = 1, 2)$ to the *i*th partial set of the reference list using the following strategy whose main aim is to ensure both the quality and diversity of the preserved solutions:

1. First, fill partially the *i*th partial set $(i = 1, 2)$ with $\lceil R_{max}/2 \rceil$ partial solutions of Ψ^i which have the best objective function values. Then, delete the added solutions from Ψ^i.
2. Define $\Delta(\phi^i_j, \psi^i_k)(\phi^i_j \in \Phi^i, \psi^i_k \in \Psi^i; \forall i, j, k)$ as the Hamming distance of the *j*th partial solution existing in Φ^i to the *k*th remaining partial solution of Ψ^i.
3. Calculate $d^{\Phi^i}_k = \min_{\phi^i_j \in \Phi^i} \Delta(\phi^i_j, \psi^i_k)(\forall \psi^i_k \in \Psi^i)$.
4. Sort the solutions of Ψ^i $(i = 1, 2)$ in descending order of $d^{\Phi^i}_k$.
5. Extend the *i*th partial set of the reference list $(i = 1, 2)$ with the first $\lfloor R_{max}/2 \rfloor$ solutions of Ψ^i.

Finally, the construction of the reference list is done by considering its last subset (the complete set) as an empty list which is gradually filled up by elite complete solutions generated during the Path Relinking Algorithm.

4.2.5 The reference list update method

The reference list constructed based on the principles described in the previous section is iteratively updated during the Path Relinking Algorithm. Unlike the general Path Relinking Algorithm in which the reference list is updated only when a new solution is generated, in the proposed PRA, two different kinds of updating method are independently applied as follows: The first type of updating method, called Internal Update Method (IUM), occurs whenever a high-quality complete solution is generated by the searching mechanism of the Path Relinking Algorithm. In IUM, once a feasible complete solution, S_{new}, is generated, it is directly added to the complete set of the reference list if the number of elite complete solutions preserved in this set is less than R_{max}; otherwise, the following replacement strategy is implemented. We first define the diversity contribution of the complete solution *S* to the complete set of the reference list shown by *P*, $D(S, P)$, as the similarity between itself and its nearest neighbour in the complete set, that is:

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

where $\Delta(S, X)$, as mentioned in the previous section, is the Hamming distance. Moreover, let us define OF_S as the objective function value of the complete solution *S*. The replacement strategy schematically shown by Figure 2 is implemented in three phases as follows: Firstly, the replacement strategy considers all the complete solutions of the complete set 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):

$$OF_{S_{new}}/D(S_{new}, P - S_{max}) < OF_{S_{max}}/D(S_{max}, P)$$

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

1) **Find** $S_{\max} = \max_{S \in I} OF_S / D(S, P)$, **where** $I = \{x \in P \mid f(S_{new})$ **is better than** $f(x)\}$.
2) **If** $OF_{S_{new}} / D(S_{new}, P - \{S_{\max}\}) < OF_{S_{\max}} / D(S_{\max}, P)$ **then replace** $S_{\max}$ **by** S_{new}.
3) **Else replace the worst solution of the set** I **by** S_{new}.

Figure 2: The proposed replacement strategy

On the other hand, the second type of the updating method, called External Update Method (EUM), occurs for the *i*th partial set of the reference list ($i = 1, 2$) whenever a new partial solution is obtained by the *i*th dedicated partial solver. As previously mentioned, the *i*th partial set of the reference list ($i = 1, 2$) consists of a set of high-quality solutions B_1 and a set of diverse solutions B_2. Suppose a new partial solution, x_{new}, is obtained by the *i*th partial solver. EUM updates the corresponding subset of the reference list as follows: First, x_{new} is examined in terms of solution quality. If it is better than the worst existing solution in B_1, the latter is replaced by the former. Otherwise, x_{new} is assessed in terms of solution diversification. In this case, x_{new} is added to the list if it increases the distance of B_2 to B_1. In other words, if the minimum Hamming distance of x_{new} to any solution in B_1 is greater than the minimum Hamming distance of the last existing solution in B_2 to any solution in B_1, the last solution in B_2 is replaced by x_{new}.

The main purpose of implementing two different update methods is to simultaneously maintain the elite partial and complete solutions generated respectively by the partial solvers and Path Relinking Algorithm.

4.2.6 Choosing the initial and guiding solutions

The performance of the Path Relinking Algorithm is highly dependent on how the initial and guiding solutions are selected from the reference list (Ghamlouche et al. (2004)). In the proposed Path Relinking Algorithm, four different strategies, each following a different purpose, are used to choose the initial and guiding solutions.

The first strategy called Partial Relinking Strategy (PRS) selects two partial solutions, each from a different partial set of the reference list, and sends them to the neighbourhood search phase to generate high-quality complete solutions. The main idea involved in implementing such a selection strategy is to produce complete solutions by integrating the best characteristics of the chosen partial solutions. Towards this end, the effect of four different sub-strategies, each generating R_{max} pairs of partial solutions, is investigated in order to choose the one having the most positive impact on the performance of the PRA. These four sub-strategies are described as follows:

PRS_1 : The *i*th pair of the first sub-strategy is constructed by defining the guiding and initial solutions as the *i*th best solution of the jth ($j = 1, 2$) and kth ($k = 1, 2$, $k \neq j$) partial sets, respectively. This sub-strategy is motivated by the idea that high-quality solutions share some common characteristics with optimum solutions. One then hopes that linking such solutions yields improved new ones.

PRS_2 : The *i*th pair of the second sub-strategy is generated by determining the guiding solution as the *i*th best solution of the jth ($j = 1, 2$) partial set, while the initial solution is defined as the *i*th worst solution of the kth ($k = 1, 2$, $k \neq j$) partial set. The purpose of this sub-strategy is to improve the worst partial solution of a partial set based on the appropriate characteristics of a high-quality partial solution of the other partial set.

PRS_3 : The *i*th pair of the third sub-strategy is constructed by randomly choosing the guiding and initial solutions from the jth ($j = 1, 2$) and kth ($k = 1, 2$, $k \neq j$) partial sets, respectively. The aim of this sub-strategy is simply to select the initial and guiding solutions in a random manner with the hope of choosing those pairs of elite partial solutions which are not selected using the other sub-strategies explained in this section.

PRS_4 : The *i*th pair of the fourth sub-strategy is generated by defining the guiding solution as the *i*th best solution of the jth ($j = 1, 2$) partial set, whereas the initial solution is chosen as the solution of the kth ($k = 1, 2$, $k \neq j$) partial set with maximum Hamming distance from the guiding solution. The aim of the fourth sub-strategy is to select the initial and guiding solutions not only according to the objective function value but also according to a diversity, or dissimilarity criterion.

On the other hand, in the second strategy called Complete Relinking Strategy (CRS), two different high-quality complete solutions are selected from the complete set of the reference list as the source for constructing a path of new solutions. In other words, in CRS, trajectories that connect complete solutions generated by the Path Relinking Algorithm are explored to obtain other high-quality complete solutions. The main purpose of this strategy is to prevent losing good complete solutions which can be obtained by searching paths constructed between other complete solutions previously generated by the algorithm. Suppose that the number of existing complete solutions in the complete set is equal to Ω ($\Omega \leq R_{max}$). In CRS, the effect of the following three sub-strategies, each generating Ω pairs of complete solutions, is investigated.

CRS_1 : The *i*th pair of the first sub-strategy is constructed by defining the guiding and initial solutions as the best and *i*th complete solutions of the complete set, respectively. The main idea involved in this sub-strategy is to improve each of the existing complete solution based on appropriate characteristics of the best complete solution found by the Path Relinking Algorithm.

CRS_2 : The *i*th pair of the second sub-strategy is generated by determining the guiding and initial solutions as the *i*th and (*i*+1)th best solutions of the complete set, respectively. The idea behind this sub-strategy is exactly the same as the idea of implementing the first sub-strategy of PRS.

CRS_3 : The *i*th pair of the last sub-strategy is generated as follows: The guiding solution is selected as the *i*th best solution of the complete set, whereas the initial solution is chosen as the solution of the same set with maximum Hamming distance from the selected guiding solution.

The third strategy called Mixed Strategy (MS) selects two distinct partial and complete solutions as the inputs of the moving mechanism phase. Using this selection strategy, we hope to improve the selected partial solution based on good features of the chosen complete solution. In MS, the effect of two different sub-strategies is investigated. These two sub-strategies are explained as follows:

MS_1 : The *i*th pair of the first sub-strategy is constructed by defining the guiding and initial solutions as the *i*th best solution of the *j*th ($j = 1, 2$) and complete sets of the reference list, respectively.

MS_2 : The *i*th pair of the second sub-strategy is generated as follows: The guiding solution is selected as the best solution of the complete set, whereas the initial solution is chosen as the solution of the *j*th ($j = 1, 2$) partial set of the reference list with maximum Hamming distance from the selected guiding solution.

The last strategy is called Ideal Point Strategy (IPS). For the sake of the following description, let us first consider the following definition:

Definition 1 Ideal Point (IP) is a virtual point whose *i*th coordinate ($i = 1, 2$) is made by the best partial solution preserved in the *i*th subset of the reference list ($i = 1, 2$).

IPS first selects two different guiding solutions so that the *i*th guiding solution is the solution kept in the *i*th coordinate of ideal point. Then, each of the solutions preserved in the reference list (partial or complete) serves respectively as the initial solution. The main purpose of choosing multiple guiding solutions is that promising regions may be searched more thoroughly in Path Relinking by simultaneously considering appropriate characteristics of multiple high-quality guiding solutions.

4.2.7 Neighbourhood structure and guiding attributes

In the proposed algorithm, unlike a general Path Relinking, two neighbourhood searches, each targeting a different goal, are implemented in parallel.

The first neighbourhood search is a memory-based searching mechanism which is done on each pair of partial solutions selected from the reference list using the partial relinking strategy. The aim of implementing such a neighbourhood search is to generate a sequence of high-quality complete solutions through integrating appropriate characteristics shared by the selected partial solutions.

As mentioned in Section 4.2.6, the partial relinking strategy selects a solution (A) from the first partial set of the reference list, as either initial or guiding solution, while the other solution (B) is chosen from the second partial set. Each of the selected partial solutions shares two important kinds of information: 1) A depot assignment pattern which shows that each customer is assigned to what depot, and 2) A visit pattern which reflects that each customer is serviced in what periods of the horizon. Without loss of generality, let us suppose that the first partial set of the reference list contains elite partial solutions of the MDVRP, whereas the second partial set is made up of elite partial solutions of the PVRP. Consequently, the selected solution (A) is a solution that the partial solver obtained by fixing the attribute "multiple periods" and by optimizing based on the attribute "multiple depots". Hence, it is reasonable to claim that in such a solution, each customer is assigned to a good depot, while there is no guarantee that the customers are visited based on good visit patterns. On the other hand, the chosen solution (B) is a solution that the other partial solver attained by fixing the attribute "multiple depots" and by optimizing based on the attribute "multiple periods". Therefore, each customer in this solution is visited through a good visit pattern, while there is no guarantee that the customers are served by good depots.

Based on the descriptions mentioned above, we can deduce that the good characteristic of the selected solution (A) is that each customer is served by a good depot, while the appropriate characteristic of the chosen solution (B) is that each customer is served based on a good visit pattern. The following definitions reveal the major idea involved in the proposed neighbourhood search:

Definition 2 A customer is called eligible if it is visited: 1) by the depot to which that customer is assigned in the solution selected from the first partial set, and 2) based on the visit pattern through which that customer is served in the solution chosen from the second partial set.

Definition 3 A good complete solution generated by the neighbourhood search is a solution in which all the customers are eligible.

Therefore, the main purpose of the neighbourhood search is to progressively introduce the properties mentioned in Definition 2 to all the customers of the selected initial solution. Towards this end, we define an algorithm which is repeated θ iterations where θ is a predetermined positive value. At the *i*th iteration of the algorithm, a customer of the initial solution is randomly selected and its eligibility is investigated based on the properties of Definition 2. Note that, depending on the partial set from which the initial solution is selected, one of the criteria mentioned in Definition 2 is always met. For example, if the initial solution is selected from the first partial set, each of the customers is assigned to a good depot. Consequently, the first property is always satisfied for all the customers and, thus, the second property should only be verified for the eligibility of the chosen customer. If the second property is not met and the selected customer is served by a visit pattern different from its corresponding visit pattern in the guiding solution, it is considered as an ineligible customer. The neighbourhood search follows then on of the following situations:

1. **Eligible customer**: If the customer is eligible, the following operators are successively applied:
   - **Intra-route relocate operator**- In this operator, the eligible customer is first removed, on each period of its visit pattern, from the route by which it is visited. It is then re-inserted to the best position, based on the penalty function described in Section 4.2.2, of the same route.

- **Inter-route relocate operator**- In this operator, the chosen customer is first removed, on each period of its visit pattern, from its current route and, then, it is re-inserted to the best position of the other routes assigned to the depot by which the customer is served.

2. **Ineligible customer**: If the selected customer is ineligible, a neighbourhood search, based on the relocate operator, is applied to the solution in order to overcome the ineligibility of the customer. To implement the relocate operator-based neighbourhood search, the following four steps are done in a sequential manner:
   (a) The depot to which the selected customer is currently assigned is changed to the depot by which that customer is served in the solution selected from the first partial set.
   (b) The current visit pattern of the selected customer is changed to the visit pattern through which the customer is visited in the solution chosen from the second partial set.
   (c) The customer is removed from the routes by which it is visited.
   (d) Finally, on each period of the new visit pattern, the removed customer is re-inserted to one of the routes assigned to the new depot. Once again, the position to which the customer is inserted is the position in which the described penalty function in Section 4.2.2 has the least value.

The neighbourhood search described above is equipped by a virtual memory whose aim is to enable the algorithm to search promising regions more thoroughly. Each element preserved in the implemented memory is represented by three indices (i, D^*, P^*), where *i* $(i = 1, 2...n)$ shows the customer's index, D^* and P^* represent, respectively, the depot and visit pattern based on which the *i*th customer is visited in the best solution generated so far by the Path Relinking. Suppose, in the course of the neighbourhood search, we select the *i*th customer which is an ineligible customer. To describe how the proposed memory works, let us consider the two following cases:

1. **The initial solution has been selected from the first partial set**: In this case, if the visit pattern through which the chosen customer is served is equal to P^*, the current visit pattern remains unchanged; otherwise, the visit pattern is changed to the one through which the customer is visited in the guiding solution.
2. **The initial solution has been chosen from the second partial set**: In this case, if the depot to which the selected customer is assigned is equal to D^*, the current depot is not changed; otherwise, the depot is changed to the one by which the customer is serviced in the guiding solution.

The main purpose of applying such a mechanism is to keep the structure of the selected solution as near as possible to the structure of the best solution obtained so far by the algorithm. This memory is updated when a new best solution is found and, for diversify search directions, the above rule is broken if the current best solution is not changed for ϵ iterations. Note that ϵ is a predetermined positive value.

The second neighbourhood search is another memory-based searching mechanism which explores trajectories connecting initial and guiding solutions selected through one of the other selection strategies, i.e., complete relinking, mixed or ideal point strategy. Like various neighbourhood searches implemented for the general Path Relinking Algorithm, the second neighbourhood search tries to gradually introduce best characteristics of either a single or multiple guiding solutions (based on the strategy used to select initial and guiding solutions) to new solutions obtained by moving away from the chosen initial solution. Similar to the neighbourhood search proposed above, the second neighbourhood is iterated θ times so that at each iteration, the eligibility of a randomly selected customer is investigated.

Definition of an eligible customer is different form what was given in the first neighbourhood search and is dependent on the strategy used to select initial and guiding solutions. Definition 4 represents the properties of an eligible customer in the cases where initial and single guiding solutions are selected using either partial relinking or mixed strategy.

Definition 4 A customer is called eligible if it is served based on the depot and visit pattern through which that customer is visited in the guiding solution.

On the other hand, Definition 5 shows the conditions under which a customer is called eligible if initial and multiple guiding solutions are chosen using ideal point strategy. Note that, in Definition 5, without loss of generality, we suppose that the first and second guiding solutions are respectively selected as the best solutions of the first and second partial sets.

Definition 5 A customer is called eligible if it is served: 1) by the depot to which that customer is assigned in the first guiding solution, and 2) based on the visit pattern through which that customer is served in the second guiding solution.

If the chosen customer is considered eligible, two operators described in the first neighbourhood search, i.e., inter- and intra-route relocate operators, are respectively implemented. Otherwise, to overcome the ineligibility of the chosen customer, a relocate operator-based neighbourhood search is applied. The proposed neighbourhood search removes first the customer from all the routes through which it is currently served. Then, one of the two following situations occurs:

- If the initial solution has been selected using either complete relinking or mixed strategy, the depot and visit pattern of the removed customer are respectively replaced by the depot and visit pattern based on which the customer is visited in the guiding solution.
- If the initial solution has been chosen using ideal point strategy, the depot and visit pattern of the removed customer are respectively changed to the depot and visit pattern of that customer in the first and second guiding solutions.

Finally, on each period of new visit pattern, the removed customer is inserted to one of the existing routes of new depot. Like the first neighbourhood search, the position to which the customer is inserted is the one in which the penalty function takes the least value.

4.2.8 Termination criterion

It is a condition that terminates the search process. In this paper, the two following stopping criteria are simultaneously considered:

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

4.2.9 Skeleton of the proposed PRA

Algorithm 3 represents the skeleton of the Path Relinking Algorithm proposed for the MDPVRP.

Algorithm 2 Path Relinking Algorithm

Initialize the search parameters.
Set υ=1, ρ=1.
Construct the initial reference list.
while the termination criterion is not met **do**
    Set α=1, β=1
    Update the reference list using the External Update Method (EUM).
    repeat
        Select one initial solution, S_i, and one or multiple guiding solutions according to one of the selection strategies.
        Set $x = S_i$.
        repeat
            Select randomly a customer of *x*.
            Verify the eligibility of the selected customer.
            Generate a solution $\bar{x}$ using the neighbourhood search corresponding to the chosen selection strategy.
            If $\bar{x}$ is feasible, update the reference list using the Internal Update Method (IUM).
            Compute *q*(.) and *t*(.) and update α and β.
            Set $x = \bar{x}$.
            Increment υ by 1.
        until $\upsilon \leq \theta$.
        Increment ρ by 1.
    until $\rho \leq R_{max}$.
end while

5 Experimental results

In this section, the performance of the proposed Path Relinking Algorithm is investigated based on different test problems. The only problem instances existing in the literature are those proposed by Vidal et al. (2010). The authors generated 10 problems whose characteristics are shown by Table 1.

To prove the efficiency of the proposed PRA, two different scenarios, each investigating one special aspect of the algorithm, are independently followed. In the first scenario, called static scenario, the partial sets of the reference list, initially filled up by the dedicated partial solvers, remain unchanged during the algorithm. In such a scenario, we aim to study how the PRA performs as a pure stand-alone algorithm without benefiting of the information that are shared by the partial solvers through updating the partial sets of the reference list. Towards this end, in each of the above problem instances, a feasible solution is first generated using the local search proposed by Vidal et al. (2010). Let us denote the constructed solution by *A*. Then, the problem in hand is decomposed into two vehicle routing problems with exactly one less attribute, i.e., PVRP and MDVRP. In the PVRP, the attribute ”multiple depots” is considered fixed

Table 1: Problem instances

Instance	n	K	m	T
pr01	48	1	4	4
pr02	96	1	4	4
pr03	144	2	4	4
pr04	192	2	4	4
pr05	240	3	4	4
pr06	288	3	4	4
pr07	72	1	6	6
pr08	144	1	6	6
pr09	216	2	6	6
pr10	288	3	6	6

by assigning each customer to the depot by which it is served in solution A. On the other hand, in the MDVRP, the other attribute ”multiple periods” is set to be fixed by allocating each customer to the visit pattern through which it is visited in solution A. Thereafter, each of the above sub-problems is solved using the hybrid genetic algorithm proposed by Vidal et al. (2010) to generate the required number of partial solutions. Finally, the obtained partial solutions are sent to the Path Relinking Algorithm in order to generate solutions of the main problem. Note that, the number of partial solutions fed to the PRA is set to 20 and the above procedure is repeated in 10 different runs for each of the problem instances.

On the other hand, in Scenario 2, called dynamic scenario, the partial sets of the reference list are updated in the course of the optimization by partial solutions generated through the Integrative Cooperative Search (ICS) method designed by Crainic et al. (2009). To more precisely understand how this scenario is built, let us briefly describe the solution methodology used in the ICS. In the ICS approach, three fundamental questions are carefully answered: how to decompose the problem at hand to define sub-problems; how to integrate partial solutions obtained from the decomposition phase to construct and improve solutions of the main problem and, finally, how to perform and guide the search. In the decomposition phase, the main problem is first decomposed into several sub-problems by fixing the values of given sets of attributes. The constructed sub-problems are then simultaneously solved by partial solvers which can be well-known constructive methods, heuristics, meta-heuristics or exact methods. The elite partial solutions obtained are sent to the central memory accompanied with context information (measures, indicators, and memories). Then, in order to construct whole solutions, integrators play their important role. integrators, which could be either exact methods or meta-heuristics, construct, and possibly improve, solutions to the main problem using solutions from the different partial solution sets. Finally, in order to repeatedly control the evolution of partial solvers and integrators implemented in the ICS approach, a guiding and controlling mechanism, namely global search coordinator, guides the global search by sending appropriate instructions to partial solvers and, eventually, integrators.

In the dynamic scenario, the proposed PRA, in fact, plays the same role as an integrator which works based on partial solutions generated during the optimization procedure of the ICS. Towards this end, a modified version of the ICS method proposed by Crainic et al. (2009) is executed on each problem instance in 10 different runs. In each of the runs, the ICS is interrupted in four different snapshots, i.e., 5, 10, 15 and 30 minutes, and partial solutions obtained at each snapshot are served in the PRA. The most distinguishable difference between these two scenarios is that, in the dynamic scenario, we examine how the quality of the proposed Path Relinking Algorithm is affected when better and more diversified partial solutions are eventually fed to the algorithm by the ICS solution methodology.

The proposed algorithm has 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 PRA, Then, in Section 5.2, we explore the impact of different combination of selection strategies, mentioned in Section 4.2.6, on the performance of the PRA. Finally, experimental results, on the two considered scenarios, are given in Section 5.3.

5.1 Parameter setting

Like the most meta-heuristic algorithms, the proposed PRA relies on a set of correlated parameters. Table 2 provides a summary of all the PRA parameters.

Table 2: Parameters of PRA

Symbol	Description
R_{max}	Maximum size of each subset of the reference list
α, β	Self-adjusting parameters in the penalty function
γ	Factor involved in updating the self-adjusting parameters
θ	Number of times that each neighbourhood search is iterated
ϵ	Number of iterations after which the memory rule is broken
μ	Maximum allowable number of non-improving iterations

There are various different methods in the literature to calibrate parameters used in a meta-heuristic. Recently, Smith and Eiben (2010) proposed a robust calibration method called Relevance Estimation and VAlue Calibration (REVAC) which is able to find good parameter values for a set of problems. In our paper, we adopted this method to tune the parameters used in the PRA. Note that, selection strategies PRS_4, CRS_3, MS_2 and IPS were used for this parameters tuning procedure. Table 3 represents the selected value of each parameter as follows:

Table 3: Calibration results

Symbol	Description
R_{max}	20
α, β	1,1
γ	1
θ	5*n
ϵ	10000
μ	400000

Table 4: Average improvement in the static scenario (%)

	PRS_1	PRS_2	PRS_3	PRS_4
(CRS_1, MS_1)	5.41	5.14	4.26	6.39
(CRS_1, MS_2)	5.50	5.24	4.51	6.64
(CRS_2, MS_1)	5.49	5.22	4.48	6.61
(CRS_2, MS_2)	5.52	5.25	4.59	6.71
(CRS_3, MS_1)	5.47	5.19	4.43	6.57
(CRS_3, MS_2)	5.61	5.33	4.89	7.12

Table 5: Average improvement in the dynamic scenario (%)

	PRS_1	PRS_2	PRS_3	PRS_4
(CRS_1, MS_1)	0.49	0.47	0.34	0.56
(CRS_1, MS_2)	0.62	0.54	0.49	0.69
(CRS_2, MS_1)	0.60	0.52	0.47	0.68
(CRS_2, MS_2)	0.64	0.56	0.51	0.72
(CRS_3, MS_1)	0.59	0.51	0.43	0.65
(CRS_3, MS_2)	0.71	0.63	0.59	0.81

5.2 Path Relinking selection strategies

We tested all combinations of selection strategies, mentioned in Section 4.2.6, in order to identify the best way to select initial and guiding solutions. The best combination is then used for the extensive experimental analysis of the Path Relinking Algorithm.

The same 10 problem instances used to calibrate the parameter settings are also used here. Moreover, each run is repeated 5 times. Thus, since there are 24 possible combinations of selection strategies (4 partial relinking strategies $\times$ 3 complete relinking strategies $\times$ 2 mixed strategies), a total of 1200 runs are performed. The performance of each combination of selection strategies is measured, in both the static and dynamic scenarios, as the average improvement in solution quality, compared to the best partial solution initially fed to the partial sets of the reference list. Note that, in the dynamic scenario, the best partial solution found at the first snapshot, 5 minutes, is used to compare the efficiency of all combinations. The comparative performances of all combinations of selection strategies, in the static and dynamic scenarios, are respectively presented in Tables 4 and 5.

Both of Tables 4 and 5 identify the combination of strategies PRS4, CRS 3 and MS2 as offering the best results. This set of selection strategies is therefore retained for our experimental analyses. The choice of this combination confirms the importance of selecting initial and guiding solutions non-randomly and also not only according to the objective function value but also according to a diversity criterion.

5.3 Results on MDPVRP instances

5.3.1 Static scenario

We tested the PRA on the problem instances described at the beginning of this section. For solve these problems, the maximum running time is set to 30 minutes. Table 6 summarizes the characteristics of partial solutions fed to the PRA using the hybrid genetic algorithm.

In Table 6, SP_1 and SP_2 represent partial solutions sets generated for the MDVRP and the PVRP, respectively. $SP_1 + SP_2$ is the union of all partial solutions obtained

Table 6: Characteristics of partial solutions in the static scenario

Instance	SP_1			SP_2			SP_1+SP_2			BKS
	Worst	Average	Best	Worst	Average	Best	Worst	Average	Best	
pr01	2152.38	2121.03	2118.84	2371.34	2253.45	2247.3	2371.34	2238.96	2118.84	2019.07
pr02	3784.13	3764.12	3747.75	4743.42	4742.1	4741.48	4743.42	4253.11	3747.75	3547.45
pr03	4943.99	4879.25	4856.39	6882.52	6439.64	6338.26	6882.52	5659.45	4856.39	4480.87
pr04	5714.78	5629.4	5575.78	8804.4	8799.12	8794.91	8804.4	7214.26	5575.78	5134.17
pr05	6059.12	6033.76	5998.22	8329.88	8324.96	8323.32	8329.88	7179.36	5998.22	5570.45
pr06	7196.13	7243.17	7130.2	8406.22	8402.05	8372.31	8406.22	7822.61	7130.2	6524.92
pr07	4820.72	4802.57	4788.51	5422.49	5421.88	5421.77	5422.49	5112.23	4788.51	4502.02
pr08	6620.41	6610.04	6594.6	8902.65	8733.29	8421.24	8902.65	7671.67	6594.6	6023.98
pr09	8750.52	8726.48	8689.47	11790.6	11779.4	11770.3	11790.6	10252.94	8689.47	8257.80
pr10	10678.9	10653.9	10617.8	14476.3	14465.1	14459.8	14476.3	12559.5	10617.8	9818.42

Table 7: Results on MDPVRP instances in the static scenario

	PRA				
	Worst	Average	Best	Computational	Gap to
Instance				time (sec)	BKS (%)
pr01	**2019.07**	**2019.07**	**2019.07**	37	0
pr02	**3547.45**	**3547.45**	**3547.45**	146	0
pr03	4533.47	4502.28	**4480.87**	487	0.47
pr04	5164.66	5160.17	5148.23	1096	0.45
pr05	5672.71	5619.72	5583.31	1612	0.97
pr06	6618.38	6571.32	6554.47	1782	0.11
pr07	4511.92	4504.77	**4502.02**	139	0.01
pr08	6038.44	6031.56	**6023.98**	428	0.14
pr09	8377.31	8353.94	8301.14	1800	1.19
pr10	10105.39	9982.63	9876.24	1800	1.70

by the hybrid genetic algorithm. Moreover, For each set, worse, average and best partial solutions on 10 runs are shown. Finally, the last column reveals the Best Known Solution (BKS) reported by Vidal et al. (2010).

In each of the problem instances, we carefully answer to the following questions: 1) What percentage of the gap is there between the PRA's output and the BKS?, and 2) How much is the PRA capable of improving the gap between partial solutions initially fed to the algorithm and the BKS?. Table 7 shows the results dealing with the first question. In this table, columns 2-4 respectively indicate the worst, average and best results obtained by the PRA on 10 runs for each instance. Moreover, the average computational time and the average error gap compared to BKS are shown in the last two columns. If the PRA gives a result (worst, average or best one) equal to the BKS, we indicate the corresponding value in boldface.

The average error gap to the BKS reported by Vidal et al. (2010) is +0.50% which is very reasonable considering the problem complexity. The PRA results vary clearly depending on the problem difficulty so that the average gap ranges from 0.00% to 1.70%. On two problems (pr01 and pr02), the algorithm seems to always converge toward the BKS, whereas problems pr08 to pr10, with larger number of depots and periods, seem particularly difficult to tackle. Generally, the proposed Path Relinking Algorithm performs well compared to the BKS even for more challenging instances including a larger number of customers, depots and periods.

On the other hand, Table 8 represents the results concerning with the second question. This table indicates what percentage of the gap between the BKS and worse, average and best partial solutions of the sets mentioned at the beginning of Section 6.2.1 (SP_1, SP_2 and $SP_1 + SP_2$) is improved using the PRA.

As shown in Table 8, The PRA is considerably powerful to decrease the gap existing between the BKS and partial solutions of all the sets. This fact, besides the results shown in Table 7, reveals that the proposed algorithm plays very well its role as a stand-alone algorithm to generate high-quality solutions of the considered MDPVRP.

5.3.2 Dynamic scenario

In the dynamic scenario, we try to properly answer to the same questions mentioned in the previous section. Table 9 indicates the main characteristics of partial solutions generated by the ICS in different snapshots. In each of the problem instances, the PRA is executed on partial solutions of each snapshot and the obtained results on 10 runs is

Table 8: Gap improvement to BKS in the static scenario (%)

	SP_1			SP_2			SP_1+SP_2		
Instance	Worst	Average	Best	Worst	Average	Best	Worst	Average	Best
pr01	6.60	5.05	4.94	17.45	11.61	11.30	17.45	8.08	4.94
pr02	6.67	6.11	5.65	33.71	33.68	33.66	33.71	19.89	5.65
pr03	9.16	8.42	8.38	52.42	43.24	41.45	52.42	25.83	8.38
pr04	10.71	9.19	8.33	70.89	70.93	71.21	70.89	40.06	8.33
pr05	6.94	7.35	7.45	47.70	48.48	49.19	47.70	27.92	7.45
pr06	8.85	9.90	8.82	27.40	27.66	27.86	27.40	13.46	8.82
pr07	6.86	6.58	6.36	20.22	20.33	20.43	20.22	13.45	6.36
pr08	9.66	9.59	9.47	47.54	44.83	39.80	47.54	27.21	9.47
pr09	4.52	4.48	4.70	41.33	41.45	9.47	41.33	22.96	4.70
pr10	5.84	6.81	7.55	44.52	45.63	46.68	44.52	26.12	7.55

reported in Table 10.

The average error gap to the BKS is +0.33%, +0.19%, +0.15% and +0.10% at 5, 10, 15 and 30-min snapshot, respectively. These average error gaps reveal that the quality of the proposed PRA increases by gradually feeding better and more diversified partial solutions by the ICS. On the other hand, in all the snapshots, the values of error gaps seem reasonable considering the problem difficulty. On four problem instance (pr01, pr02, pr07 and pr08), the PRA always trap, in all snapshots, on the best partial solution fed by the ICS. This phenomenon seems inevitable because, in each of these problems, there exists apparently no better solution than the BKS which is initially sent as a partial solution to the PRA by the ICS. On two problems (pr03 and pr10), the PRA obtained new best known solutions which are shown as boldface starred values in the table.

Table 11 reports the improvement percentage on the gap between the BKS and partial solutions initially sent to the PRA. Note that, pr01, pr02, pr07 and pr08 are ignored in Table 11 because, as mentioned above, the ICS always sends, in these problems, the BKS as a partial solution to the PRA.

By considering the results obtained in the static and dynamic Scenarios, we deduce that the proposed PRA is considerably a competitive structural method to generate high-quality solutions of the MDPVRP either as a stand-alone algorithm or as an integrator in the ICS solution methodology.

6 Conclusions

This paper presented a new Path Relinking Algorithm to efficiently tackle the multi-depot periodic vehicle routing problem, for which few efficient algorithms are currently available. The proposed algorithm was designed based on prominent exploration and exploitation strategies which enable the algorithm to solve the problem in two different ways: 1) As a pure stand alone algorithm, and 2) As an integrator in the ICS solution framework.

To validate the efficiency of PRA, different test problems, existing in the literature, were solved. The computational results revealed that the proposed Path Relinking Algorithm performs considerably well, in all the problem instances.

Table 9: Characteristics of partial solutions in the dynamic scenario

Instance	Snapshot	SP_1			SP_2			SP_1+SP_2		
		Worst	Average	Best	Worst	Average	Best	Worst	Average	Best
pr01	5 min.	2053.43	2028.14	2019.17	2112.51	2044.05	2019.17	2112.51	2036.09	2019.17
	10 min.	2043.01	2020.19	2019.17	2044.07	2026.42	2019.17	2044.07	2032.27	2019.17
	15 min.	2033.19	2026.38	2019.17	2027.48	2024.26	2019.17	2027.48	2025.61	2019.17
	30 min.	2022.88	2121.83	2019.17	2023.97	2021.75	2019.17	2023.97	2021.92	2019.17
pr02	5 min.	3608.77	3558.96	3547.45	3611.28	3562.31	3547.45	3611.28	3567.14	3547.45
	10 min.	3595.14	3552.16	3547.45	3595.14	3550.51	3547.45	3595.14	3553.22	3547.45
	15 min.	3588.42	3550.68	3547.45	3559.66	3549.14	3547.45	3588.42	3550.09	3547.45
	30 min.	3565.31	3549.66	3547.45	3554.04	3548.23	3547.45	3565.31	3549.53	3547.45
pr03	5 min.	4721.48	4537.04	4481.94	4853.33	4486.88	4481.94	4853.33	4507.53	4481.94
	10 min.	4700.62	4503.37	4481.94	4850.66	4483.23	4481.94	4850.66	4493.28	4481.94
	15 min.	4564.74	4523.19	4480.87	4486.41	4483.34	4480.87	4564.74	4503.14	4480.87
	30 min.	4538.53	4514.19	4480.87	4484.35	4482.45	4480.87	4538.53	4495.91	4480.87
pr04	5 min.	5201.69	5188.06	5172.76	5248.72	5195.26	5175.77	5248.72	5192.59	5172.76
	10 min.	5170.82	5164.58	5149.05	5162.32	5161.02	5149.05	5170.82	5162.81	5149.05
	15 min.	5177.83	5168.42	5149.05	5239.55	5178.44	5149.05	5239.55	5173.47	5149.05
	30 min.	5442.74	5239.41	5144.45	5152.96	5149.75	5144.45	5442.74	5199.34	5144.45
pr05	5 min.	5958.50	5788.24	5603.28	5958.50	5762.19	5682.16	5958.50	5768.22	5603.28
	10 min.	5720.83	5687.28	5642.99	5683.15	5664.23	5642.99	5720.83	5665.12	5642.99
	15 min.	5714.57	5681.45	5642.99	5958.50	5773.21	5642.99	5958.50	5728.35	5642.99
	30 min.	5706.81	5653.23	5604.95	5717.69	5664.12	5604.95	5717.69	5658.92	5604.95
pr06	5 min.	7085.21	6675.21	6608.98	7047.29	6663.29	6608.98	7085.21	6669.53	6608.98
	10 min.	6772.95	6659.41	6589.88	6735.38	6648.21	6589.88	6772.95	6653.12	6589.88
	15 min.	6744.39	6638.79	6589.81	6735.38	6626.47	6589.81	6744.39	6630.25	6589.81
	30 min.	6594.50	6574.25	6567.66	6590.36	6571.19	6567.66	6594.50	6572.22	6567.66
pr07	5 min.	4638.60	4578.29	4502.02	4778.42	4589.23	4502.02	4778.42	4584.18	4502.02
	10 min.	4577.91	4538.25	4502.02	4517.79	4506.77	4502.02	4577.91	4527.51	4502.02
	15 min.	4577.91	4528.84	4502.02	4509.36	4504.87	4502.02	4577.91	4520.36	4502.02
	30 min.	4509.97	4504.93	4502.02	4504.45	4503.33	4502.02	4509.97	4503.66	4502.02
pr08	5 min.	6246.78	6167.32	6024.24	6577.04	6321.87	6024.24	6577.04	6244.25	6024.24
	10 min.	6246.78	6097.44	6023.98	6485.56	6299.41	6023.98	6485.56	6226.09	6023.98
	15 min.	6246.78	6077.29	6023.98	6069.12	6044.65	6023.98	6246.78	6060.43	6023.98
	30 min.	6246.78	6054.38	6023.98	6025.21	6024.46	6023.98	6246.78	6044.61	6023.98
pr09	5 min.	8570.64	8433.12	8326.58	8531.55	8417.17	8316.95	8570.64	8425.62	8316.95
	10 min.	8312.40	8304.99	8296.42	8305.65	8301.44	8296.42	8312.40	8302.78	8296.42
	15 min.	8307.94	8301.45	8296.09	8305.65	8299.52	8296.09	8307.94	8300.27	8296.09
	30 min.	8424.49	8349.51	8293.33	8300.34	8297.44	8293.33	8424.49	8324.66	8293.33
pr10	5 min.	12626.90	10857.44	10128.8	13340.10	11190.51	10128.8	13340.10	13152.27	10128.8
	10 min.	10489.30	10227.44	9993.94	10402.10	10200.36	9993.94	10489.30	10214.77	9993.94
	15 min.	10169.20	10134.98	9993.94	10059.70	10032.46	9993.94	10169.20	10081.33	9993.94
	30 min.	10091.90	10049.77	9993.94	12192.90	10104.71	9993.94	12192.90	10070.14	9993.94

Table 10: Results on MDPVRP instances in the dynamic scenario

		PRA				
Instance	Snapshot	Worst	Average	Best	Computational time (sec)	Gap to BKS (%)
pr01	5 min.	**2019.07**	**2019.07**	**2019.07**	15	0
	10 min.	**2019.07**	**2019.07**	**2019.07**	15	0
	15 min.	**2019.07**	**2019.07**	**2019.07**	15	0
	30 min.	**2019.07**	**2019.07**	**2019.07**	15	0
pr02	5 min.	**3547.45**	**3547.45**	**3547.45**	70	0
	10 min.	**3547.45**	**3547.45**	**3547.45**	70	0
	15 min.	**3547.45**	**3547.45**	**3547.45**	70	0
	30 min.	**3547.45**	**3547.45**	**3547.45**	70	0
pr03	5 min.	**4480.87**	**4480.87**	**4480.87**	156	0
	10 min.	**4480.87**	**4480.87**	**4480.87**	156	0
	15 min.	**4480.87**	**4479.68***	**4472.22***	156	-0.061
	30 min.	**4480.87**	**4478.15***	**4472.22***	156	-0.12
pr04	5 min.	5155.32	5153.60	5150.73	257	0.36
	10 min.	5149.05	5148.95	5142.26	257	0.28
	15 min.	5149.05	5148.39	5142.26	257	0.27
	30 min.	5148.45	5147.92	5142.26	257	0.27
pr05	5 min.	5597.12	5595.78	5582.45	338	0.43
	10 min.	5603.28	5593.33	5581.10	338	0.37
	15 min.	5596.73	5589.14	5581.10	338	0.31
	30 min.	5594.94	5585.27	5581.10	338	0.23
pr06	5 min.	6573.29	6562.70	6542.33	459	0.53
	10 min.	6566.46	6547.39	6542.33	459	0.33
	15 min.	6566.46	6545.99	6538.91	459	0.30
	30 min.	6549.57	6542.55	6538.91	459	0.25
pr07	5 min.	**4502.02**	**4502.02**	**4502.02**	75	0
	10 min.	**4502.02**	**4502.02**	**4502.02**	75	0
	15 min.	**4502.02**	**4502.02**	**4502.02**	75	0
	30 min.	**4502.02**	**4502.02**	**4502.02**	75	0
pr08	5 min.	**6023.98**	**6023.98**	**6023.98**	184	0
	10 min.	**6023.98**	**6023.98**	**6023.98**	184	0
	15 min.	**6023.98**	**6023.98**	**6023.98**	184	0
	30 min.	**6023.98**	**6023.98**	**6023.98**	184	0
pr09	5 min.	8312.14	8305.74	8294.69	477	0.54
	10 min.	8292.23	8291.44	8287.14	477	0.37
	15 min.	8292.23	8287.14	8288.14	477	0.35
	30 min.	8286.01	8277.60	8274.38	477	0.24
pr10	5 min.	9995.29	9964.27	9859.12	732	1.49
	10 min.	9886.34	9876.51	9844.78	732	0.59
	15 min.	9886.34	9855.13	**9811.3***	732	0.35
	30 min.	9871.06	9835.78	**9811.3***	732	0.16

Table 11: Gap improvement to BKS in the dynamic scenario (%)

Instance	Snapshot	SP_1			SP_2			SP_1+SP_2		
		Worst	Average	Best	Worst	Average	Best	Worst	Average	Best
pr03	5 min.	5.37	1.25	0.02	8.31	0.13	0.02	8.31	0.59	0.02
	10 min.	4.90	0.50	0.02	8.25	0.05	0.02	8.25	0.28	0.02
	15 min.	1.87	1.01	0.19	0.12	0.12	0.19	1.87	0.56	0.19
	30 min.	1.29	0.84	0.19	0.08	0.13	0.19	1.29	0.43	0.19
pr04	5 min.	0.90	0.69	0.43	1.82	0.83	0.49	1.82	0.78	0.43
	10 min.	0.42	0.31	0.13	0.26	0.24	0.13	0.42	0.27	0.13
	15 min.	0.56	0.39	0.13	1.76	0.59	0.13	1.76	0.49	0.13
	30 min.	5.81	1.78	0.03	2.03	0.04	0.04	5.81	1	0.04
pr05	5 min.	6.49	3.48	0.37	6.49	3.01	1.79	6.49	3.12	0.37
	10 min.	2.11	1.73	1.11	1.43	1.31	1.11	2.11	1.33	1.11
	15 min.	2.11	1.68	1.11	6.49	3.32	1.11	6.49	2.52	1.11
	30 min.	2.01	1.26	0.43	2.20	1.45	0.43	2.20	1.36	0.43
pr06	5 min.	7.84	1.78	1.02	7.26	1.59	1.02	7.84	1.69	1.02
	10 min.	3.16	1.73	0.73	2.59	1.56	0.73	3.16	1.64	0.73
	15 min.	2.73	1.44	0.78	2.59	1.25	0.78	2.73	1.31	0.78
	30 min.	0.69	0.50	0.44	0.63	0.46	0.44	0.69	0.47	0.44
pr09	5 min.	3.13	1.58	0.39	2.66	1.39	0.27	3.13	1.49	0.27
	10 min.	0.24	0.20	0.11	0.17	0.16	0.11	0.24	0.15	0.11
	15 min.	0.19	0.17	0.13	0.16	0.15	0.13	0.19	0.16	0.13
	30 min.	1.68	0.87	0.26	0.17	0.24	0.26	1.68	0.45	0.26
pr10	5 min.	26.80	9.10	2.75	34.07	12.49	2.75	34.07	10.28	2.75
	10 min.	6.14	3.61	1.51	5.25	3.33	1.51	6.14	3.46	1.51
	15 min.	2.88	2.88	1.86	1.77	1.83	1.86	2.88	1.84	1.86
	30 min.	2.25	2.20	1.86	23.65	2.76	1.86	23.65	2.48	1.86

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 QuÃ©bec through their infrastructure grants

References

[1] Crainic, T.G., Crisan, G.C., Gendreau, M., Lahrichi, N., Rei, W., *A concurrent evolutionary approach for rich combinatorial optimization*. GECCO'09: Proceedings of the 11th annual conference companion on Genetic and evolutionary computation conference, 2009.

[2] Dantzig, G.B., Ramser, J.H., *The truck dispatching problem*. Management Science, 1959, 6, 80.

[3] Ghamlouche, I., Crainic, T.G., Gendreau, M., *Path Relinking, Cycle-Based Neighbourhoods and Capacitated Multicommodity Network Design*. Annals of Operations Research, 2004, 131(1-4), 109-133

[4] Glover, F., Laguna, M., *Fundamentals of Scatter Search and Path Relinking*. Control and Cybernetics, 2000, 29(3), 653-684.

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

[6] Ho, S.C., Gendreau, M., *Path Relinking for the vehicle routing problem*. Journal of Heuristics, 2006, 12, 55-72.

[7] Kang, K.H., Lee, Y.H., Lee, B.K., *An exact algorithm for multi depot and multi period vehicle scheduling problem*. Computational Science and Its Applications-ICCSA, 2005, 350-359.

[8] Kytöjoki, J., Nuortio, T., Bräysy, O., Gendreau, M., *An efficient variable neighborhood search heuristic for very large scale vehicle routing problems*. Computers & Operations Research, 2005, 47(2), 329-336.

[9] Mingozzi, A., *The Multi-depot Periodic Vehicle Routing Problem*. Lecture Notes in Computer Science, 2005, 3607, 347-350.

[10] Parthanadee, P., Logendran, R., *Periodic product distribution from multi-depots under limited supplies*. IIE Transactions, 2006, 38(11) 1009-1026.

[11] Smith, S.K., Eiben, A.E., *Parameter Tuning of Evolutionary Algorithms: Generalist and Specialist*. Applications of Evolutionary Computation, LNCS 6024, Springer, 2010, 542-551.

[12] Toth, P., Vigo, D., *The vehicle routing problem*. Society for Industrial and Applied Mathematics, 2002, Philadelphia, PA, USA.

[13] Vidal, T., Crainic, T.G., Gendreau, M., Lahrichi, N., Rei, W., *A Hybrid Genetic Algorithm for Multi-Depots and Periodic Vehicle Routing Problems*. CIRRELT, 2010, 34.