

A path relinking algorithm for a multi-depot periodic vehicle routing problem

Alireza Rahimi-Vahed · Teodor Gabriel Crainic ·
Michel Gendreau · Walter Rei

Received: 5 July 2012 / Revised: 1 April 2013 / Accepted: 14 April 2013 /
Published online: 28 April 2013
© Springer Science+Business Media New York 2013

Abstract In this paper, we consider a multi-depot periodic vehicle routing problem which is characterized by the presence of a homogeneous fleet of vehicles, multiple depots, multiple periods, and two types of constraints that are often found in reality, i.e., vehicle capacity and route duration constraints. The objective is to minimize total travel costs. To tackle the problem, we propose an efficient path relinking algorithm whose exploration and exploitation strategies enable the algorithm to address 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. The performance of the proposed path relinking algorithm is evaluated, in each of the above ways, based on standard benchmark instances. The computational results show that the developed PRA performs well, in both solution quality and computational efficiency.

Keywords Multi-depot periodic vehicle routing problem · Path relinking · Integrative co-operative search

A. Rahimi-Vahed (✉)

Département Informatique et recherche opérationnelle and Interuniversity Research Centre on Enterprise Networks, Logistics, and Transportation (CIRRELT), Université de Montréal, C.P. 8888, succ. Centre-ville, Montreal, QC H3C 3P8, Canada
e-mail: alireza.rahimi.vahed@umontreal.ca

T. G. Crainic · W. Rei

Département Management et technologie, ESG and Interuniversity Research Centre on Enterprise Networks, Logistics, and Transportation (CIRRELT), Université du Québec à Montréal, C.P. 8888, succ. Centre-ville, Montreal, QC H3C 3P8, Canada

M. Gendreau

Département de mathématiques et génie Industriel and Interuniversity Research Centre on Enterprise Networks, Logistics, and Transportation (CIRRELT), École Polytechnique, C.P. 6079, succ. Centre-ville, Montreal, QC H3C 3A7, Canada

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 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 which is the problem of designing, for an homogeneous fleet of vehicles, a set of routes for each period of a given planning horizon. Each vehicle performs only one route per period and each vehicle route must start and finish at the same depot. Each customer may require to be visited on different periods during the planning horizon and these visits may only occur in one of a given number of allowable visit patterns. 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 VRP is typically called the multi-depot periodic vehicle routing problem (MDPVRP).

To tackle the MDPVRP, we propose a new path relinking algorithm (PRA), 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 Sect. 2. The literature survey relevant to the topic of this paper is presented in Sect. 3. Section 4 introduces the proposed PRA. The experimental results are reported in Sect. 5. Finally, we conclude in Sect. 6.

2 Problem statement

In this section, we formally state the MDPVRP ([Cordeau et al. 1997](#)), introducing the notations used throughout this paper. The MDPVRP can be defined as follows ([Vidal et al. 2012](#)): consider an undirected graph $G(V, E)$. The node set V is the union of two subsets $V = V_C \cup V_D$, where $V_C = \{v_1, \dots, v_n\}$ represents the customers and

$V_D = \{v_{n+1}, \dots, v_{n+m}\}$ includes the depots. With each node $i \in V_C$ is associated a deterministic demand q_i , which is to be satisfied over a planning horizon of T periods. Each node $i \in V_C$ is also characterized by a service frequency f_i , stating how often within these T periods this node must be visited and a list L_i of possible visit-period combinations, called patterns. 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. 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 review papers previously published in the literature addressing the MDPVRP. The objective of this review is first to present the most recently proposed heuristic and meta-heuristic algorithms for the considered problem and second, to distinguish leading solution approaches that have been proved to be efficient to tackle the MDPVRP.

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 four heuristics in the first group. [Cordeau et al. \(1997\)](#) designed a tabu search heuristic for two special variants of the MDPVRP, i.e., the PVRP and the MDVRP. The proposed tabu search possesses some of the features of Taburoute ([Gendreau et al. 1994](#)), namely the consideration of intermediate infeasible solutions through the use of a generalized objective function containing self-adjusting coefficients, and the use of continuous diversification. Neighbour solutions are obtained by moving a vertex from its route between two of its closest neighbours in another route, by means of a generalized insertion (or GENI) (see [Gendreau et al. 1992](#)). [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. \(2005\)](#) 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.

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 Lahrichi et al. (2012). The authors proposed a structured cooperative parallel search method, called 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 statement made in Sect. 1 that the MDPVRP is among the VRP variants which did 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 PRA to efficiently address the MDPVRP as a whole. The proposed PRA is described in the next section.

4 The path relinking algorithm (PRA)

The PRA, as a population-based meta-heuristic, 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 Sect. 4.1. Then, the main components of PRA proposed to solve the MDPVRP are explained in details in Sect. 4.2.

4.1 The path relinking algorithm in general

The PRA has been suggested as an approach to integrate intensification and diversification strategies in the context of tabu search (Glover and Laguna 2000). PRA can be considered as an evolutionary algorithm where solutions are generated by combining elements from other solutions. Unlike other evolutionary methods, such as GAs, 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 PRA are:

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 these components interact.

Algorithm 1 The general PRA

- 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.
-

4.2 The proposed path relinking algorithm

4.2.1 General idea

The PRA 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 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 exchanges, through the search mechanism of 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 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 PRA or iteratively updated in order to include better solutions, in terms of both solution quality and diversification. We call these two possibilities as static and dynamic scenarios,

respectively. Challenges, advantages and deficiencies of each scenario are thoroughly discussed in Sect. 5.

After constructing the initial reference list, the proposed PRA starts to construct high-quality solutions of the main problem by exploring trajectories that connect solutions selected from different subsets, partial and/or complete, of the reference list. Towards this end, several selection strategies, each choosing initial and guiding solutions using a different strategy, are first implemented. Then, for each selected initial and guiding solutions, a neighbourhood search generates a sequence of complete solutions using the information shared by the selected solutions.

Two special variants of the proposed PRA can be obtained by ignoring complete and partial sets, respectively. In the former case, PRA generates complete solutions based on information only gathered from partial solutions, while, in the latter case, the proposed algorithm becomes to a general path relinking as described in Sect. 4.1.

Different components of the proposed PRA 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 search 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 $1 + \gamma$, otherwise it is multiplied by $1 + \gamma$, where γ is a positive parameter. A similar rule applies also to β with 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 search space. Representation should be easy to implement 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. (2007) 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 1–4, which have to be serviced from 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 serviced from the second depot. All customers need to be visited in each period. Figure 1 shows how a solution of this problem is represented.

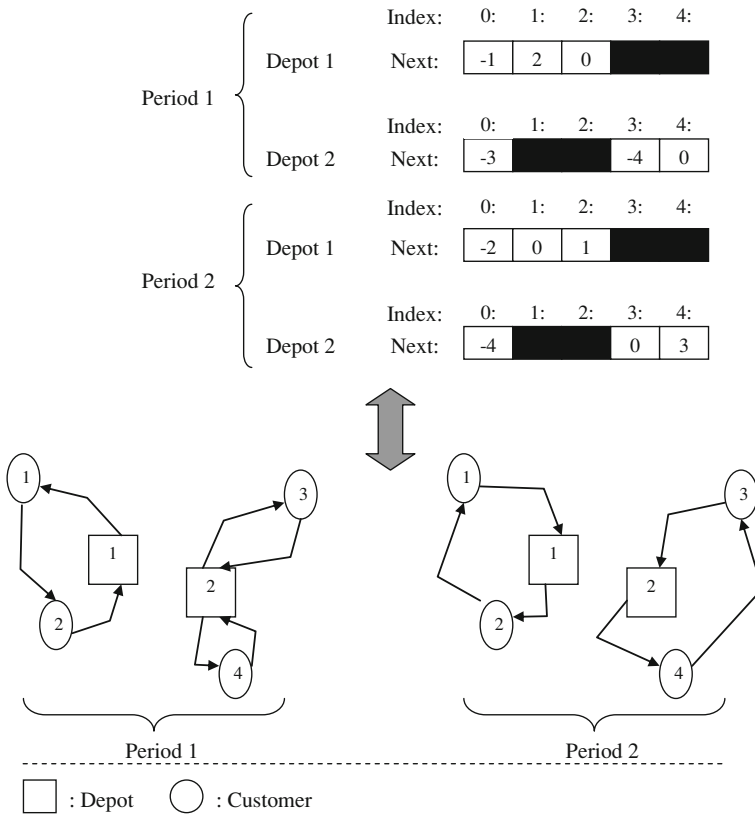


Fig. 1 An example of the solution representation

As depicted in Fig. 1, in this kind of representation, a single row array of size $n + 1$, where n is the number of customers to be visited, is generated for each depot in each period. 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 the customer to be immediately visited after the customer or depot related to that position. For example, in Fig. 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 leaving the first depot. In this path representation, negative values indicate the beginning of a new route, 0 refers to the end of the routes and a vacant position (drawn as a black box in Fig. 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. 2007).

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 applying the search mechanism of the PRA. 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 Sect. 4.2.1, the reference list implemented in 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 R_{max} .

Let us define 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_j^i : the j th partial solution of Φ^i ,
- ψ_k^i : 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 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 . We call this part of the partial set as B_1 .
2. Define $\Delta(\phi_j^i, \psi_k^i)(\phi_j^i \in \Phi^i, \psi_k^i \in \Psi^i; \forall i, j, k)$ 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 $d_k^{\Phi^i} = \min_{\phi_j^i \in \Phi^i} \Delta(\phi_j^i, \psi_k^i)(\forall \psi_k^i \in \Psi^i)$.
4. Sort the solutions of Ψ^i in descending order of $d_k^{\Phi^i}$.
5. Extend the i th partial set of the reference list with the first $\lfloor R_{max}/2 \rfloor$ solutions of Ψ^i . This part of the partial set is named as B_2 .

Note that, the construction of the reference list considers its last subset (the complete set) initially empty and then gradually filling it by elite complete solutions generated during the PRA.

4.2.5 The reference list update method

The reference list is iteratively updated during the PRA in order not to lose high-quality and diverse solutions. The general PRA updates the reference list when a new solution is generated. In the proposed PRA, this translates into two updating methods, which are independently applied as follows. The first method, called internal update method (IUM), occurs whenever a high-quality complete solution is generated by the search mechanism of the PRA. In IUM, once a feasible complete solution, S_{new} , is

- 1) Find $S_{max} = \max_{S \in I} OF_S / D(S, P)$, where $I = \{x \in P \mid f(S_{new}) \text{ is better than } f(x)\}$.
- 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} .

Fig. 2 The proposed replacement strategy

generated, it is directly added to the complete set of the reference list if the number of elite complete solutions kept in this set is less than R_{max} ; otherwise, the following replacement strategy is implemented. We first define the diversity contribution of the solution S_{new} to the complete set of the reference list shown by P , $D(S_{new}, P)$, as the minimum Hamming distance between S_{new} and the solutions in P , that is:

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

Moreover, let us define $OF_{S_{new}}$ as the objective function value of S_{new} . The replacement strategy schematically shown in Fig. 2 is implemented in three phases: 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*)/(*diversity contribution*) (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).

The second 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 diversity. 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 worst solution of B_2 to any solution in B_1 , the worst solution is replaced by x_{new} and all the existing solutions of B_2 , including x_{new} , are sorted again.

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 PRA.

4.2.6 Choosing the initial and guiding solutions

The performance of the PRA is highly dependent on how the initial and guiding solutions are selected from the reference list (Ghamlouche et al. 2004). In the proposed PRA, 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. 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, will be investigated in Sect. 5.2 in order to choose the one having the most positive impact on the performance of PRA. These four sub-strategies are:

- PRS*₁: The i th pair is constructed by defining the guiding and initial solutions as the i th best solution of the j th ($j = 1, 2$) and k th ($k = 1, 2, k \neq j$) 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 is generated by determining the guiding solution as the i th best solution of the j th ($j = 1, 2$) partial set, while the initial solution is defined as the i th worst solution of the k th ($k = 1, 2, k \neq j$) 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 is constructed by randomly choosing the guiding and initial solutions from the j th ($j = 1, 2$) and k th ($k = 1, 2, k \neq j$) 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 is generated by defining the guiding solution as the i th best solution of the j th ($j = 1, 2$) partial set, whereas the initial solution is chosen as the solution of the k th ($k = 1, 2, k \neq j$) 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.

In the second strategy, called complete relinking strategy (CRS), two different 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 PRA are explored to obtain other 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 selecting Ω pairs of complete solutions, will be investigated in Sect. 5.2.

- CRS₁*: The i th pair 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 PRA.
- CRS₂*: The i th pair 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.
- CRS₃*: 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 a partial and a complete solution as the inputs of the neighbourhood search phase. This selection strategy aims to improve the selected partial solution based on good features of the chosen complete solution. For MS, the effect of two different sub-strategies will be investigated in Sect. 5.2.

- MS₁*: The i th pair 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.
- MS₂*: The i th pair 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 j th ($j = 1, 2$) partial set of the reference list with maximum Hamming distance from the selected guiding solution.

The last strategy that we propose is called ideal point strategy (IPS), where the ideal point (IP) is defined as a virtual point whose i th coordinate ($i = 1, 2$) is made by the objective function value of the best partial solution of the i th partial set ($i = 1, 2$).

IPS first selects two different guiding solutions so that the i th guiding solution is the best solution of the i th coordinate of IP. 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, two neighbourhood searches, each targeting a different goal, are implemented in parallel.

First search strategy—The first neighbourhood search is a memory-based strategy operating on each pair of partial solutions selected from the reference list using the PRS. 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 Sect. 4.2.6, the PRS selects a solution (A) from the first partial set of the reference list, as either an 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 elements of information: (1) a depot assignment pattern that shows to which depot each customer is assigned, and (2) a visit pattern determining the periods of the planning horizon in which each customer is serviced. 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 “period” attribute and by optimizing based on the “depot” attribute. 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 “depot” attribute and by optimizing based on the “period” attribute. 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.

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

Definition 2 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 make eligible all the customers of the selected initial solution. The algorithm proceeds for 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 1. Note that, depending on the partial set from which the initial solution is selected, one of the criteria mentioned in Definition 1 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 only should 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 ineligible. The neighbourhood search then follows one of the following options:

1. **Eligible customer:** If the customer is eligible, the following operators are successively applied:
 - **Intra-route relocate operator**–The eligible customer is first removed, for each period of its visit pattern, from the route by which it is visited. It is then re-inserted into the best position, based on the penalty function described in Sect. 4.2.2, of the same route.
 - **Inter-route relocate operator**–The chosen customer is first removed, for each period of its visit pattern, from its current route and, then, it is re-inserted into the best position of the other routes assigned to the depot from 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 its ineligibility. 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 according 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, at each period of the new visit pattern, the removed customer is re-inserted into one of the routes assigned to the new depot. Once again, the position to which the customer is inserted is the position where the described penalty function in Sect. 4.2.2 has the least value.

The neighbourhood search described above is equipped with a memory whose aim is to enable the algorithm to search promising regions more thoroughly. Each element preserved in the memory is represented by three indices (i, D^*, P^*) , where i ($i = 1, 2, \dots, n$) 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 that 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 according to 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,

to diversify search directions, the above rule is not applied if the current best solution is not changed for ϵ iterations. Note that ϵ is a predetermined positive value.

Second search strategy—The second neighbourhood search is another memory-based strategy which explores trajectories connecting initial and guiding solutions selected through one of the other selection strategies, i.e., complete relinking, MS or IPS. Like various neighbourhood searches implemented for the general PRA, the second neighbourhood search tries to gradually introduce best characteristics of either a single or multiple guiding solutions (following 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.

The definition of an eligible customer is different from that of the first neighbourhood search and is dependent on the strategy used to select the initial and guiding solutions. Definition 3 represents the properties of an eligible customer in the cases where initial and single guiding solutions are selected using the partial relinking or MS.

Definition 3 A customer is called eligible if it is served based on the depot and visit pattern according to which that customer is visited in the guiding solution.

Definition 4 specifies the conditions under which a customer is called eligible if initial and multiple guiding solutions are chosen using the IPS. Note that, in Definition 4, 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 4 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 according to 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 applied. 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 the complete relinking or MS, 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 IPS, 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, in each period of the new visit pattern, the removed customer is reinserted into one of the existing routes assigned to the 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

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 2 represents the skeleton of the PRA proposed for the MDPVRP.

Algorithm 2 Path Relinking Algorithm

```

Initialize the search parameters.
Construct the initial reference list.
while the termination criterion is not met do
  Set  $\rho=0$ .
  Update the reference list using the EUM.
  for  $i=1 \dots \|C^*\|$  do
    Set  $j$  = The  $i$ th element of  $C^*$ .
    repeat
      Select one initial solution,  $S$ , and one or multiple guiding solutions according to
      selection strategy  $j$ .
      Set  $x=S$ .
      Set  $v=0$ .
      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  $q(\cdot)$  and  $t(\cdot)$  and update  $\alpha$  and  $\beta$ .
        Set  $x=\bar{x}$ .
        Increment  $v$  by 1.
      until  $v \leq \theta$ .
      Increment  $\rho$  by 1.
    until  $\rho \leq R_{max}$ .
  end for
end while

```

When considering Algorithm 2, the following remarks should be taken into account:

1. The way in which the partial sets of the reference list are updated depends on how and when partial solvers are called within each iteration of the proposed PRA. In the next section, we thoroughly describe when partial solvers are called and how they interact with the solutions found during PRA.
2. C^* is defined as a list representing the best order of selection strategies to choose initial and guiding solutions at a given iteration. For example, if C^* is made up as $PRS_2 \rightarrow CRS_3 \rightarrow MS_1 \rightarrow IPS$, it means that, at a given iteration, initial and guiding solutions should be selected using the following order:

- (a) The second partial relinking sub-strategy.
- (b) The third complete relinking sub-strategy.
- (c) The first mixed sub-strategy.
- (d) The IPS.

In Sect. 5.2, we explain how C^* is built.

5 Experimental results

In this section, the performance of the proposed PRA is investigated based on the standard benchmark instance proposed by Vidal et al. (2012). The authors generated 10 problems whose characteristics are shown by Table 1.

To explore the efficiency of the proposed PRA, two different scenarios, each investigating one special aspect of the algorithm, are investigated. 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 PRA performs as a pure stand-alone algorithm based on some initial partial information but without benefiting from information shared by partial solvers operating in parallel. Towards this end, a feasible solution is first generated using the local search proposed by Vidal et al. (2012) for each of the problem instances. Let us denote the constructed solution by A . Then, the problem is decomposed into two VRPs with exactly one less attribute, i.e., PVRP and MDVRP. In the PVRP, the attribute “multiple depots” is fixed 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 fixed by allocating each customer to the visit pattern according to which it is visited in solution A . Thereafter, each of the above sub-problems is solved using a dedicated partial solver. The dedicated partial solver can be any exact, heuristic or meta-heuristic method proposed in the literature to address the PVRP and MDVRP. In this paper, we use the hybrid GA proposed by Vidal et al. (2012) to rapidly generate fairly good (not necessarily high-quality) partial solutions. Finally, the obtained partial solutions are sent to the PRA in order to generate solutions of the main problem.

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

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 ICS method designed by [Lahrichi et al. \(2012\)](#). 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 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 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 role of an integrator working based on partial solutions generated during the optimization procedure of the ICS. Towards this end, a modified version of the ICS method is executed on each problem instance for 10 different runs. In each of the runs, the ICS is interrupted in four different snapshots, i.e., after 5, 10, 15 and 30 min, and partial solutions obtained at each snapshot are used to update the partial sets of the reference list using the External Update Method. Note that, in this scenario, each partial set remains unchanged between two successive snapshots. The most distinguishable difference between the two scenarios is that, in the dynamic scenario, we examine how the quality of the proposed PRA 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 Sect. 5.1, we first use a well-structured algorithm to calibrate all the parameters involved in PRA. Then, in Sect. 5.2, we explore the impact of different combination of selection strategies, introduced in Sect. 4.2.6, on the performance of PRA. Finally, experimental results on the two considered scenarios are given in Sect. 5.3.

5.1 Parameter setting

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

There are various methods in the literature to calibrate parameters used in heuristics and meta-heuristics. In this paper, we use the well-reported four-step calibration

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

Table 3 Calibration results of REVAC

Symbol	Starting level	Range	Changing step
R_{max}	20	[20,40]	5
α, β	1,1	[1,5], [1,5]	1,1
γ	1	[1,4]	1
θ	n	$[n, 5*n]$	$n/2$
ϵ	5,000	[5000,10000]	2,500
μ	200,000	[200000, 600000]	50,000

method of [Coy et al. \(2000\)](#) to tune the parameters used in the heuristic algorithm. [Coy et al. \(2000\)](#) proposed a fairly fast procedure based on statistical design of experiments and gradient descent to systematically calibrate parameter values. The proposed parameter setting procedure takes a small number of the problem instances from the entire problem set, finds high-quality parameter settings for each problem, and then combines the parameter settings to determine good parameter values for the entire set of instances. This procedure can be summarized in the following four steps that are implemented in a sequential manner:

- Step 1– A subset of instances to analyze (analysis set) is chosen from the entire set. The instances are selected so that most of the structural differences found in the problem set are represented in the analysis set. In this study, the analysis set is made up of five instances, i.e., pr02, pr04, pr06, pr09, and pr10.
- Step 2– 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. Towards this end, in this paper, a robust calibration method called relevance estimation and value calibration (REVAC) ([Smith and Eiben 2010](#)) is used. Technically, REVAC is a heuristic generate-and-test method that is iteratively searching for the set of parameter vectors of a given evolutionary algorithm (EA) with a maximum performance. For each iteration, a new parameter vector is generated and its performance is tested. Testing a parameter vector is done by executing the EA with the given parameter values and measuring the EA performance. A detailed explanation of REVAC can be found in [Smith and Eiben \(2010\)](#). [Table 3](#) summarizes the results obtained using REVAC.

Table 4 Calibration results

Symbol	Description
R_{max}	30
α, β	1, 3
γ	1
θ	$4*n$
ϵ	5,000
μ	500,000
Time (s)	13,826

- Step 3– Good parameter settings are selected for each instance in the analysis set using fractional factorial design and response surface optimization. In this paper, the half-factorial design is implemented for each selected analysis set.
- Step 4– Finally, in this step, the parameter values obtained in the third step are averaged to obtain high-quality parameter values. Table 4 represents the selected value of each parameter, along with the average computational time needed to calibrate the parameters. The obtained average computational time, reported in Table 4, seems reasonable considering this fact that each calibration method is done only once for each new typical application.

5.2 Path relinking selection strategies

We tested all combinations of selection strategies, defined in Sect. 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 PRA.

The same 5 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 ($4 \text{ PRS} \times 3 \text{ complete relinking strategies} \times 2 \text{ MS}$), a total of 600 runs were performed. The performance of each combination of selection strategies was 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 min, 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 presented in Tables 5 and 6, respectively.

Both of Tables 5 and 6 identify the combination of strategies PRS_4 (The fourth partial relinking sub-strategy), CRS_3 (The third complete relinking sub-strategy) and MS_2 (the second mixed sub-strategy) 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.

Table 5 Average improvement in the static scenario (%)

	PRS_1	PRS_2	PRS_3	PRS_4
(CRS_1, MS_1)	6.88	6.52	6.41	7.16
(CRS_1, MS_2)	7.50	7.31	7.12	7.82
(CRS_2, MS_1)	7.22	7.15	7.01	7.65
(CRS_2, MS_2)	7.71	7.44	7.29	8.07
(CRS_3, MS_1)	7.04	6.72	6.60	7.31
(CRS_3, MS_2)	7.64	7.50	7.34	8.49

Table 6 Average improvement in the dynamic scenario (%)

	PRS_1	PRS_2	PRS_3	PRS_4
(CRS_1, MS_1)	1.21	1.16	0.97	1.44
(CRS_1, MS_2)	1.60	1.39	1.30	1.79
(CRS_2, MS_1)	1.47	1.30	1.18	1.72
(CRS_2, MS_2)	1.71	1.58	1.41	1.90
(CRS_3, MS_1)	1.31	1.22	1.03	1.63
(CRS_3, MS_2)	1.80	1.59	1.44	2.12

Table 7 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

5.3 Results on MDPVRP instances

5.3.1 Static scenario

We tested PRA on the problem instances described at the beginning of this section. In this scenario, the maximum running time was set to 30 minutes. Table 7 summarizes the characteristics of partial solutions initially fed to PRA.

In Table 7, 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. Moreover,

Table 8 Average results on MDPVRP instances in the static scenario

Instance	PRA				HGA		Gap to HGA (%)	Gap to BKS (%)
	Worst	Average	Best	Time (s)	Average	Time (s)		
pr01	2019.07	2019.07	2019.07	30	2019.07	21	0	0
pr02	3547.45	3547.45	3547.45	124	3547.45	89	0	0
pr03	4480.87	4480.87	4480.87	495	4491.08	463.2	-0.22	0
pr04	5155.12	5149.64	5134.17	1,389	5151.73	1,326	-0.04	0.30
pr05	5672.71	5598.32	5579.43	1,800	5605.60	1,800	-0.12	0.50
pr06	6618.38	6568.79	6540.66	1,800	6570.28	1,800	-0.02	0.67
pr07	4502.02	4502.02	4502.02	203	4502.06	131	-0.01	0
pr08	6038.44	6027.51	6023.98	547	6029.58	478	-0.03	0.05
pr09	8341.09	8304.26	8268.88	1,800	8310.19	1,667	-0.07	0.56
pr10	9987.16	9963.55	9852.30	1,800	9972.35	1,800	-0.09	1.48
AV	5636.23	5616.15	5594.88	998.8	5619.94	957.5	-0.06	+0.36

for each set, worse, average and best partial solutions on 10 runs are shown. Finally, the last column reveals the best known solution (BKS) obtained by hybrid genetic algorithm (HGA) of Vidal et al. (2012) for the same problem. It should be noted that, the BKS do not represent the average performance of the HGA, but are rather the best solutions that were identified overall runs performed by HGA.

For each problem instance, we answer to the following questions:

1. What percentage of the gap is there between the PRA’s output and the BKS?
2. Can the proposed PRA compete with HGA of Vidal et al. (2012), which has been proved as one of the most powerful solution methodologies to address the MDPVRP?
3. How much is PRA capable of improving the gap between initial partial solutions and the BKS?

Table 8 shows the results dealing with the two first questions. In this table, the average results of 10 runs of PRA and HGA, in terms of solution quality and computational time, are reported in columns 2–6. Moreover, the average error gaps of PRA compared to the HGA and BKS are respectively shown in the last two columns. Finally, the last line (AV) provides the average value of each column. Note that, if each of the considered solution methods (PRA and HGA) give a result equal to the BKS, we indicate the corresponding value in boldface.

The results shown by Table 8 can be interpreted as follows:

1. The average error gap of PRA to the BKS is +0.36% which is very reasonable considering the problem complexity. PRA results vary according to the problem difficulty. The average gap ranges from 0.00 to 1.48%. On four problems (pr01, pr02, pr03 and pr04), 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 PRA performs well when compared to the BKS even on the more challenging instances.

Table 9 An example of the Friedman test table

Instance	The average results obtained by PRA	The average results obtained by HGA
pr01	X_{11}	X_{12}
pr02	X_{21}	X_{22}
pr03	X_{31}	X_{32}
⋮	⋮	⋮
⋮	⋮	⋮
⋮	⋮	⋮

2. The average standard deviation (ASD) per instance of PRA is 0.22 %. This value is less than the ASD obtained by HGA (0.26 %) and reveals this fact that PRA is very reliable to produce good results.
3. The average computational time of PRA is barely higher than the HGA, but it still seems reasonable considering the problem difficulty.
4. The average error gap existing between PRA and HGA is -0.06% which shows the superiority of PRA to produce better results. To statistically prove this superiority, the well-known Friedman test is used. The Friedman test is a non-parametric statistical test developed by the economist Milton Friedman. Similar to the parametric repeated measures ANOVA, it is used to detect differences in treatments across multiple test attempts. The procedure involves ranking each row (or block) together, then considering the values of ranks by columns. In this paper, the Friedman test table is presented as Table 9. In this table, the first column display instance identifier, while the two other columns show the average results respectively generated by PRA and HGA. The characteristics of Friedman test, implemented in this paper, are as follows:

– **Assumptions:**

- The results over problem instances are mutually independent (i.e., the results within one instance do not influence the results within other instance)
- Within each problem instance, the objective functions can be ranked.

– **Hypotheses:**

- H_0 : There is no significant difference between the outputs of PRA and HGA.
- H_1 : PRA performs better than HGA.

– **Procedure:**

- (a) Rank the results of both algorithms (PRA and HGA) within each problem instance, giving 1 to the best and 2 to the worst. Let us define $R(X_{ij})$ as the rank assigned to row i and column j of Table 9.
- (b) Calculate the total summation of squared ranks, A_2 , using the following formula:

$$A_2 = \sum_{i=1}^{10} \sum_{j=1}^2 [R(X_{ij})]^2$$

Table 10 Friedman test results

Friedman test's variable	Value
A_2	49
B_2	48.2
T_2	$36 > f_{0.99,1,9}$

Table 11 Average gap improvement to BKS in the static scenario (%)

Instance	SP_1			SP_2			$SP_1 + SP_2$		
	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	10.33	8.89	8.38	53.60	43.71	41.45	53.60	26.03	8.38
pr04	10.90	9.34	8.60	71.08	71.08	71.30	71.08	40.21	8.60
pr05	6.94	7.82	7.52	47.70	48.95	49.26	47.70	28.39	7.52
pr06	8.85	10.33	9.04	27.40	28.10	28.07	27.40	19.21	9.04
pr07	7.08	6.68	6.36	20.45	20.43	20.43	20.45	13.56	6.36
pr08	9.66	9.67	9.47	47.54	44.91	39.80	47.54	27.29	9.47
pr09	4.96	5.11	5.09	41.77	42.08	42.40	41.77	23.60	5.09
pr10	7.04	7.03	7.80	45.72	45.84	46.92	45.72	26.43	7.80

- (c) Compute the summation of the rank for each algorithm, $R_j = \sum_{i=1}^{10} R(X_{ij})$ for $j = 1, 2$ and calculate B_2 :

$$B_2 = \frac{1}{10} \sum_{j=1}^2 R_j^2$$

- (d) The test statistic is given by:

$$T_2 = \frac{9(B_2 - 45)}{A_2 - B_2}$$

- (e) Reject H_0 , at the level of significance 0.01, if T_2 is greater than the quantile of the F distribution with $K_1 = 1$ and $K_2 = 9$ degrees of freedom.

Table 10 summarizes the results of the above five-step procedure.

As shown in Table 10, the test static (T_2) is greater than $f_{0.99,1,9}$. This result justifies that PRA performs significantly better than HGA to produce good results, in terms of solution quality and computational time.

On the other hand, Table 11 represents the results concerning the third question. This table indicates how much PRA is able to improve the gap between partial solutions and the BKS.

As shown in Table 11, PRA is considerably powerful to decrease the gap existing between the BKS and partial solutions of all the sets. This fact, along with the results

shown in Table 8, 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 answer the same questions as mentioned in Sect. 5.3.1. Table 12 indicates the main characteristics of partial solutions generated by the ICS in different snapshots. In each of the problem instances, PRA is executed on partial solutions of each snapshot and the obtained results on 10 runs is reported in Table 13.

The average error gap to the BKS is +0.25, +0.20, +0.17 and +0.12% 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), PRA always traps, 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 PRA by the ICS. On two problems (pr03 and pr10), PRA obtained new best known solutions which are shown as boldface starred values in the table.

Moreover, the average standard deviation per instance produced by PRA is 0.22, 0.22, 0.20 and 0.19 at 5, 10, 15 and 30-min snapshot, respectively. These values, along with the obtained average computational time, reveal this fact that PRA, as a reliable algorithm, is able to generate good results in a reasonable time.

Finally, the average error gap between PRA and HGA is -0.16 , -0.21 , -0.25 and -0.30% at 5, 10, 15 and 30-min snapshot, respectively. These average error gaps prove, once again, better performance of PRA to generate promising results, in terms of solution quality and computational efficiency. Note that, in this scenario, the average result generated by PRA, for each problem instance and snapshot, is equal or better than the corresponding value in the static scenario. Consequently, we do not need to use the Friedman test and the significant difference between PRA and HGA is automatically proved.

Table 14 reports the improvement percentage on the gap between the BKS and partial solutions initially sent to PRA. Note that, pr01, pr02, pr07 and pr08 are ignored in Table 14 because, as mentioned above, the ICS always sends, in these problems, the BKS as a partial solution to PRA.

Studying the results obtained in the static and dynamic scenarios, we deduce that the proposed PRA can be used as a competitive solution method in the both of its considered settings, i.e., as a stand-alone algorithm and as an integrator in the ICS solution methodology.

6 Conclusions

This paper presented a new PRA to efficiently tackle the MDPVRP, for which few efficient algorithms are currently available. The proposed algorithm was designed

Table 12 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 13 Average results on MDPVRP instances in the dynamic scenario

Instance	Snapshot	PRA				Gap to HGA (%)	Gap to BKS (%)
		Worst	Average	Best	Time (s)		
pr01	5 min.	2019.07	2019.07	2019.07	15	0	0
	10 min.	2019.07	2019.07	2019.07	15	0	0
	15 min.	2019.07	2019.07	2019.07	15	0	0
	30 min.	2019.07	2019.07	2019.07	15	0	0
pr02	5 min.	3547.45	3547.45	3547.45	65	0	0
	10 min.	3547.45	3547.45	3547.45	65	0	0
	15 min.	3547.45	3547.45	3547.45	65	0	0
	30 min.	3547.45	3547.45	3547.45	65	0	0
pr03	5 min.	4480.87	4480.87	4480.87	242	-0.22	0
	10 min.	4480.87	4480.87	4480.87	242	-0.22	0
	15 min.	4480.87	4479.29*	4472.22*	242	-0.26	-0.03
	30 min.	4480.87	4478.12*	4472.22*	242	-0.29	-0.06
pr04	5 min.	5155.32	5148.42	5134.17	300	-0.07	0.27
	10 min.	5149.05	5148.09	5134.17	300	-0.07	0.27
	15 min.	5149.05	5147.81	5134.17	300	-0.07	0.27
	30 min.	5148.45	5147.75	5134.17	300	-0.07	0.26
pr05	5 min.	5597.12	5595.24	5581.10	300	-0.18	0.44
	10 min.	5603.28	5592.91	5581.10	300	-0.22	0.40
	15 min.	5596.73	5589.04	5581.10	343	-0.29	0.33
	30 min.	5594.94	5585.16	5581.10	343	-0.36	0.26
pr06	5 min.	6573.29	6560.44	6540.66	300	-0.14	0.54
	10 min.	6566.46	6547.08	6540.66	300	-0.35	0.33
	15 min.	6566.46	6546.19	6538.91	512	-0.37	0.32
	30 min.	6549.57	6541.80	6538.60	512	-0.43	0.25
pr07	5 min.	4502.02	4502.02	4502.02	101	-0.009	0
	10 min.	4502.02	4502.02	4502.02	101	-0.009	0
	15 min.	4502.02	4502.02	4502.02	101	-0.009	0
	30 min.	4502.02	4502.02	4502.02	101	-0.009	0
pr08	5 min.	6023.98	6023.98	6023.98	225	-0.09	0
	10 min.	6023.98	6023.98	6023.98	225	-0.09	0
	15 min.	6023.98	6023.98	6023.98	225	-0.09	0
	30 min.	6023.98	6023.98	6023.98	225	-0.09	0
pr09	5 min.	8312.14	8303.89	8268.88	300	-0.07	0.56
	10 min.	8292.23	8291.08	8268.88	300	-0.22	0.40
	15 min.	8292.23	8288.44	8268.88	494	-0.26	0.37
	30 min.	8286.01	8279.55	8268.88	494	-0.37	0.26
pr10	5 min.	9995.29	9890.22	9852.30	300	-0.82	0.73
	10 min.	9886.34	9875.78	9852.30	300	-0.97	0.58

Table 13 continued

Instance	Snapshot	PRA				Gap to HGA (%)	Gap to BKS (%)
		Worst	Average	Best	Time (s)		
AV	15 min.	9886.34	9856.30	9811.3*	750	-1.16	0.39
	30 min.	9871.06	9839.07	9811.3*	750	-1.36	0.21
	5 min.	5620.66	5607.16	5595.05	214.8	-0.16	+0.25
	10 min.	5607.08	5602.83	5595.05	214.8	-0.21	+0.20
	15 min.	5606.42	5599.96	5589.91	304.7	-0.25	+0.17
	30 min.	5602.34	5696.37	5589.88	304.7	-0.30	+0.12

Table 14 Average 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	0.98	0.19	0.12	0.09	0.19	1.87	0.53	0.19
	30 min.	1.29	0.80	0.19	0.08	0.09	0.19	1.29	0.44	0.19
pr04	5 min.	0.90	0.77	0.46	1.82	0.86	0.49	1.82	0.81	0.46
	10 min.	0.42	0.32	0.15	0.26	0.28	0.15	0.42	0.30	0.15
	15 min.	0.56	0.41	0.13	1.79	0.62	0.13	1.76	0.51	0.13
	30 min.	5.81	1.82	0.03	2.06	0.05	0.03	5.81	0.93	0.03
pr05	5 min.	6.49	3.50	0.38	6.49	3.02	1.80	6.49	3.26	0.38
	10 min.	2.11	1.74	1.12	1.43	1.33	1.12	2.11	1.53	1.12
	15 min.	2.11	1.69	1.12	6.49	3.34	1.12	6.49	2.51	1.12
	30 min.	2.01	1.28	0.44	2.20	1.46	0.44	2.20	1.37	0.44
pr06	5 min.	7.84	1.80	1.03	7.26	1.59	1.03	7.84	1.70	1.03
	10 min.	3.16	1.74	0.75	2.59	1.61	0.75	3.16	1.68	0.75
	15 min.	2.73	1.44	0.80	2.61	1.26	0.80	2.75	1.35	0.80
	30 min.	0.69	0.50	0.44	0.64	0.48	0.44	0.72	0.49	0.44
pr09	5 min.	3.13	1.60	0.40	2.66	1.41	0.28	3.13	1.50	0.28
	10 min.	0.24	0.21	0.10	0.17	0.18	0.10	0.24	0.20	0.10
	15 min.	0.19	0.17	0.12	0.16	0.15	0.14	0.18	0.16	0.12
	30 min.	1.68	0.87	0.24	0.16	0.24	0.28	1.65	0.56	0.24
pr10	5 min.	26.80	9.12	2.78	34.07	12.51	2.77	34.07	10.81	2.77
	10 min.	6.14	3.60	1.54	5.25	3.32	1.53	6.14	3.46	1.53
	15 min.	2.88	2.86	1.86	1.77	1.83	1.84	2.88	2.34	1.86
	30 min.	2.25	2.18	1.86	23.65	2.76	1.84	23.65	2.47	1.86

based on different exploration and exploitation strategies which enable the algorithm to address 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 PRA performs considerably well, for all the problem instances.

The proposed PRA is a general-purpose solver and opens the way to experiments and sensitivity analyses of local search and meta-heuristic components on a wide range of structurally different problems. Finally, the generalization of the method towards a wider variety of multi-objective and stochastic problems appears as an interesting research avenue.

Acknowledgments 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

- Cordeau, J.F., Gendreau, M., Laporte, G.: A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks* **30**, 105–119 (1997)
- Coy, S.P., Golden, B.L., Runger, G.C., Wasil, E.A.: Using experimental design to find effective parameter settings for heuristics. *J. Heuristics* **7**(1), 77–97 (2000)
- Dantzig, G.B., Ramser, J.H.: The truck dispatching problem. *Manag. Sci.* **6**, 80 (1959)
- Gendreau, M., Hertz, A., Laporte, G.: New insertion and postoptimization procedures for the traveling salesman problem. *Oper. Res.* **40**, 1086–1094 (1992)
- Gendreau, M., Hertz, A., Laporte, G.: A tabu search heuristic for the vehicle routing problem. *Manag. Sci.* **40**, 1276–1290 (1994)
- Ghamlouche, I., Crainic, T.G., Gendreau, M.: Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. *Ann. Oper. Res.* **131**(1–4), 109–133 (2004)
- Glover, F., Laguna, M.: Fundamentals of scatter search and path relinking. *Control Cybern.* **29**(3), 653–684 (2000)
- Hadjiconstantinou, E., Baldacci, R.: A multi-depot period vehicle routing problem arising in the utilities sector. *J. Oper. Res. Soc.* **49**, 1239–1248 (1998)
- Kang, K.H., Lee, Y.H., Lee, B.K.: An exact algorithm for multi depot and multi period vehicle scheduling problem. In: *Computational Science and Its Applications-ICCSA, Lecture Notes in Computer Science*, pp. 350–359. ICCSA, Springer-Verlag, Berlin, Heidelberg (2005)
- Kytöjoki, J., Nuortio, T., Bräysy, O., Gendreau, M.: An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Comp. Oper. Res.* **34**(9), 2743–2757 (2007)
- Lahrichi, N., Crainic, T.G., Gendreau, M., Rei, W., Crisan, G.C., Vidal, T.: An integrative cooperative search framework for multi-decision-attribute combinatorial optimization. Technical Report, CIRRELT, p. 42 (2012)
- Parthanadee, P., Logendran, R.: Periodic product distribution from multi-depots under limited supplies. *IIE Trans.* **38**(11), 1009–1026 (2006)
- Smith, S.K., Eiben, A.E.: Parameter Tuning of Evolutionary Algorithms: Generalist and Specialist. *Applications of Evolutionary Computation*, LNCS vol. 6024, pp. 542–551. Springer, Heidelberg (2010)
- Toth, P., Vigo, D.: *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, Philadelphia (2002)
- Vidal, T., Crainic, T.G., Gendreau, M., Lahrichi, N., Rei, W.: A hybrid genetic algorithm for multi-depots and periodic vehicle routing problems. *Oper. Res.* **60**(3), 611–624 (2012)