Published online ahead of print October 15, 2012

TRANSPORTATION SCIENCE
Articles in Advance, pp. 1–24
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.1120.0439

Adaptive Path Relinking for Vehicle Routing and Scheduling Problems with Product Returns

Christos D. Tarantilis, Afroditi K. Anagnostopoulou
Operations Research and Decision Systems Centre, Management Science Laboratory,
Department of Management Science and Technology, Athens University of Economics and Business,
Athens, Greece GR 10434
{tarantil@aueb.gr, aanag@aueb.gr}

Panagiotis P. Repoussis
Howe School of Technology Management, Stevens Institute of Technology, Hoboken, New Jersey 07030,
panagiotis.repoussis@stevens.edu

This paper deals with one-to-many-to-one vehicle routing and scheduling problems with pickups and deliveries and studies the effect of various backhauling strategies. Initially, focus is given on problem instances with clustered backhauls where all delivery customers must be visited before pickup customers. Afterward, operational settings with mixed backhauls and varying visiting sequence restrictions with respect to the capacity of the vehicles are examined. The proposed solution method evolves a set of reference solutions on the basis of a novel Adaptive Path Relinking framework. The latter encompasses an adaptive multisolution recombination procedure to generate provisional solutions based on the recurrence of particular solution attributes. On return, these solutions are used as guiding points for performing search trajectories from initial reference solutions via tunneling. Computational results on benchmark data sets of the literature illustrate the competitiveness and robustness of the proposed approach compared to state-of-the-art solution methods for well-known vehicle routing and scheduling problems. Finally, various experiments are also reported to demonstrate the economic effect of different mixing levels and densities of linehaul and backhaul customers.

Key words: reverse logistics; path relinking; vehicle routing; vehicle scheduling
History: Received: September 2010; revisions received: January 2012; accepted: June 2012. Published online in *Articles in Advance*.

1. Introduction

Managing the flows of spent or returned products has become a crucial concern for modern companies seeking to explore and integrate reverse logistics as a viable business activity. Depending on the nature of returned products, an alternative option is to design combined distribution-collection systems. In this case, the utilization of vehicles increases significantly when merging products brought to the customers as well as products brought back to the depot, and the vehicle routing and scheduling plans and the flows of freights get more effective and balanced. Improving productivity and utilization of the vehicle fleet through "backhauling" is a common practice that appears in many different sectors. Typical real-life paradigms can be found in parcel services (Anily 1996), in blood banks systems (Ganesh and Narendran 2007), and in food and grocery distribution-collection services (Dart 1983). Therefore, in practical terms, studying such problems definitely seems worthwhile because inefficient operational planning can limit the economic success of reprocessing end-of-life, used, recyclable, and/or other types of returning products (Sbihi and Eglese 2007).

Vehicle routing and scheduling problems deal with the optimum assignment and service sequence of a set of customer orders to a fleet of vehicles. They have a large number of real-life applications and come in many variants, depending on the type of operation, the time frame for decision making, the objective, and the types of constraint that must be adhered to (Braysy et al. 2008). Furthermore, they accent in the real-life context when temporal aspects (e.g., customer time windows) are considered in addition to the pure routing geographic counterpart (Repoussis, Tarantilis, and Ioannou 2009). Finally, the objective typically refers to the minimization of the total transportation cost, expressed mainly in terms of one-time (e.g., fleet size) and recurring costs (e.g., distance traveled).

The focus of this paper is given on one-to-many-to-one vehicle routing problems with both pickup and deliveries. The term "one-to-many-to-one" denotes that all delivery demands (shipment of products to linehaul customers) are initially located at the depot and correspondingly all pickup demands (collection of products from backhaul customers) are returned to the depot. On the basis of a set of predefined visiting sequence restrictions (also known as backhauling strategies), several models appear in the literature that embody the essence and characteristics of dealing with both linehaul and backhaul customers on the same vehicle routes. Among those considering customer time windows, the most well studied are the so-called Vehicle Routing Problem with Clustered Backhauls and Time Windows (VRPCBTW) (Gelinas et al. 1995) and the Vehicle Routing Problem with Mixed Backhauls and Time Windows (VRPMBTW) (Kontoravdis and Bard 1995). The aforementioned problems are NP-hard in the strong because they are natural generalizations of the well-known Vehicle Routing Problem with Time Windows (VRPTW) (Gendreau and Tarantilis 2010). Therefore, substantial computational effort is required for determining optimum or near optimum solutions even for medium size instances.

Given a homogeneous fleet of depot-returning capacitated vehicles, the goal is to design a set of vehicle routes in order to satisfy the delivery and collection requirements of a set of geographically scattered customers. Each customer has a known demand either for delivery (linehaul) or pickup (backhaul) and must be serviced within a predefined time window that models the earliest and the latest times during the day that service can take place. As such, vehicles must remain at the customer locations during the service, and there is a waiting time if a vehicle arrives before the customer's earliest time window. Finally, each customer must be visited only once by exactly one vehicle, and the load of a vehicle must not exceed vehicle's maximum capacity at any time along its route. The primary objective is to minimize the number of vehicles required to service all customers, and the secondary objective is to minimize the distance traveled.

There are two main backhauling strategies that fit the above described problem setting. Considering VRPCBTW instances, all linehaul customers of a route must be serviced before the vehicle starts visiting backhaul customers. Besides priority, access, security, and/or other reasons, the practical perspective behind this strategy is that the vehicle is normally loaded in a way that reflects the sequence of delivery customers to ensure efficient unloading and to eliminate additional rearrangements of carrying products during customer service. Although such a restriction eliminates potential inefficiencies, it also reduces the possible synergies of combining pickup and delivery customers, particularly if time windows for pickups are early and deliveries occur late during the planning horizon (Reimann and Ulrich 2006). Therefore, if no negative influence of picking return products occurs during the service of linehaul customers, then the best alternative strategy is to relax the restriction concerning the visiting sequence and to allow any ordering of pickup and delivery visits that satisfies the vehicle capacity constraint. This operational setting is captured by the VRPMBTW, in which no a priori visiting sequence restrictions are imposed. Aside from these two extremes, other alternatives—less studied in the literature—are to allow the controlled mixing of linehaul and backhaul customers on the basis of the density of backhaul customers and/or the tightness of time windows and/or the current loading of the vehicles during the customer service.

The aim of this paper is to develop an efficient and effective solution method for one-to-many-to-one vehicle routing and scheduling problems with clustered and mixed backhauls, including the VRPCBTW and the VRPMBTW, as well as to study the effect of alternative backhauling strategies based on the delivery load of vehicles. The proposed approach evolves a set of reference solutions on the basis of a novel Adaptive Path Relinking solution framework. Initiating from a set of diverse feasible solutions, subsets of intermediate solutions are produced iteratively via an enhanced path generation method. This method incorporates a multisolution recombination procedure to generate guiding provisional solutions based on the recurrence of particular solution attributes. On return, these provisional solutions are used as guiding points for performing search trajectories that initiate from elite reference solutions. The underlying relinking mechanism utilizes multiple edge-exchange neighborhood structures for variation and also benefits from tunneling through infeasible regions of the solution space assuming that capacity and time window constraints are relaxed. To that end, locally optimum intermediate solutions are selected and further improved via a local search improvement method. The latter treats both feasible and infeasible solutions on the basis of a penalized cost function and incorporates computationally efficient neighborhood evaluation methods.

For the evaluation of the proposed approach, computational experiments are performed on the benchmark data sets of Gelinas et al. (1995); Thangiah, Potvin, and Sun (1996); and Kontoravdis and Bard (1995). Compared to the current state-of-the-art methods for the VRPCBTW and the VRPMBTW, the proposed approach proved to be highly competitive, especially on large-scale instances where the total number of vehicles is significantly reduced. In most cases, the best reported cumulative and mean results are improved for most problem instances with fairly reasonable computational requirements. Furthermore, in an effort to demonstrate the robustness and generality of the proposed approach, computational experiments on the large-scale benchmark data sets of Gehring and Homberger (1999) for the VRPTW are reported. Overall, the proposed approach managed to produce high quality solutions for all groups of problem instances, and in several cases the best reported cumulative and mean results are also improved. Finally, a series of computational experiments is also performed to examine the cost profile of different backhauling strategies and of mixing levels of linehaul and backhaul customers, considering various capacity and loading restrictions.

The remainder of the paper is organized as follows: §2 provides an overview of the literature for one-to-many-to-one vehicle routing and scheduling problems. The problem definition and notations are given in §3. Then §4 discusses the proposed solution method and provides a detailed description of all algorithmic components and mechanisms. Computational experiments assessing the quality of the proposed approach along with a comparative performance analysis are presented in §5. Finally, in §6 conclusions are drawn and pointers for future research are provided.

2. Literature Review

Because of its wide applicability and high complexity, significant developments have been made toward the design of models and optimization methods for vehicle routing and scheduling problems with pickups and deliveries (Toth and Vigo 1997; Mingozzi, Giorgi, and Baldacci 1999; Parragh, Doerner, and Hartl 2008a, b). In particular, the literature for VRPCBTW and VRPMBTW instances includes both exact and approximate solution approaches. However, problem instances with more than 100 customers can be intractably hard to be solved to optimality. For this reason, the focus of most researchers is on the design and implementation of metaheuristic approaches capable of producing high quality solutions within reasonable computational time limits. However, there is much room for improvement, especially in terms of effectiveness for solving large-scale problem instances. Below, a brief overview of solution methods proposed for the VRPCBTW and the VRPMBTW is provided.

In the field of exact approaches for the VRPCBTW, Yano et al. (1987) introduced the first branch-and-bound algorithm based on real-life data. Later, Derigs and Metz (1992) examined a problem arising in the overnight express mail services of Federal Express with up to 80 customers. In particular, diverse formulations and relaxations are proposed, and a matching based solution approach is developed and evaluated on real-world data sets. More recently, Gelinas et al. (1995) introduced an exact branch-and-bound approach based on a set partitioning formulation. A key feature of the later approach is that resource variables (time and capacity) are branched instead of network flow variables. To this end, 45 problem instances with 25, 50, and 100 customers based on Solomon (1987) VRPTW benchmark data sets are generated, and more than half are solved to optimality.

Kontoravdis and Bard (1995) first introduced and implemented a Greedy Randomized Adaptive Search Procedure (GRASP) for the VRPMBTW. During the construction phase, a greedy randomized penalty-based parallel insertion construction heuristic is proposed, combined with a constant size restricted candidate list. On the other hand, the local search phase consist of a simple iterative improvement scheme on the basis of 2-Opt neighborhood structures. Computational experiments are reported on 27 appropriately modified longhaul problem instances of Solomon (1987) for the VRPTW with up 100 customers.

Thangiah, Potvin, and Sun (1996) proposed a local search heuristic approach for the VRPCBTW. For the construction of initial solutions, a push-forward sequential insertion heuristic based on Solomon's I1 construction heuristic (Solomon 1987) is introduced, combined with a feasibility technique proposed by Kontoravdis and Bard (1995) to confine time window violations during the customer insertion phase. On the other hand, the proposed iterative improvement local search heuristic incorporates λ-interchanges and 2-Opt edge-exchanges. Finally, new large-scale problem instances with 250 and 500 customers are also introduced.

Potvin, Duhamel, and Guertin (1996) designed a greedy route construction heuristic combined with a genetic algorithm for the VRPCBTW. The proposed construction heuristic operates on a sequential basis and myopically inserts customers one by one into the routes using a fixed a priori ordering of customers. In particular, customers' ordering is defined and used as the basis during route construction in order to select customers for insertion that minimize a weighted sum of distance increase and service delay. The genetic counterpart of the proposed scheme aims to identify orderings that will produce good routes. Later, Duhamel, Potvin, and Rousseau (1997) proposed a tabu search metaheuristic algorithm for the VRPCBTW, considering as secondary objective the minimization of the total schedule time (i.e., traveling times, service times, and waiting times). Initial solutions are generated via a modified savings-based (Clarke and Wright 1964) construction heuristic, and the solution space is explored on the basis of 2-Opt, Or-Opt, and Swap edge-exchange structures.

Reimann, Doerner, and Hartl (2002) proposed an insertion based ant colony optimization method for the VRPCBTW. Solutions are generated iteratively based on pheromone information, using a sequential insertion based construction heuristic, and the local search improvement phase consists of special 3-Opt and Swap operators. Pheromone information is updated depending on the trail persistence and on the number of elitist solutions. More recently, Reimann and Ulrich (2006) proposed a similar approach for solving the VRPCBTW and the so-called mixed VRPBTW. In particular, two parameters are introduced in order to associate some cost components with respect to the mixing of pickups and deliveries on the same vehicle route. The first represents a threshold and determines the percentage of free capacity required such that a pickup is allowed, whereas the second constitutes a penalty and deals with the increase in the service time for deliveries, once pickups have been made by a vehicle. It is also worth mentioning that comparisons among different backhauling strategies are also reported.

Hasama, Kokubugata, and Kawashima (1998) and Zhong and Cole (2005) proposed local search metaheuristic approaches for solving the VRPCBTW and the VRPMBTW. In particular, Hasama, Kokubugata, and Kawashima (1998) introduced a simulated annealing-like approach based on a string model. The latter expresses each solution as a sequence of characters (string) that implies the routing schedule for each vehicle. Zhong and Cole (2005) presented a Guided Local Search (GLS) approach. The main idea is to construct an initial infeasible solution and then apply GLS in an effort to restore feasibility as well as to improve the solution quality. During local search, several moves are applied cyclically, and a best-accept strategy is followed. The augmented objective function is based on constraint violations and a section planning technique that is used to divide each route into sections. The proposed penalty function depends on the distance between subsequent customers and the vehicle's waiting times.

Finally, Ropke and Pisinger (2006) presented a unified Large Neighborhood Search (LNS) metaheuristic algorithm for solving a wide range of vehicle routing problems with backhaul customers, including among others the VRPCBTW and the VRPMBTW. The proposed LNS framework consists of six removal and three insertion operators, which compete to modify the current solution. To that end, an adaptive layer controls stochastically the selection of operators with a bias toward its past performance, and at each iteration the new modified solution is evaluated and accepted according to a probabilistic criterion.

As mentioned earlier, one-to-many-to-one vehicle routing and scheduling problems generalize the VRPTW. The latter is one of the most intensively studied NP-hard combinatorial optimization problems. Many successful approaches for the VRPTW are population-based and typically involve local search improvement methods based on edge-exchange neighborhood structures. Furthermore, it is common to employ two distinct stages, dedicated to the minimization of the fleet size and then the distance traveling cost, respectively. However, few of the current state-of-the-art methods stand out in terms of simplicity and flexibility and there is an evident lack of efficient solution approaches with a wider applicability toward rich extensions of the VRPTW combining multiple features (Gendreau and Tarantilis 2010). Furthermore, some approaches are very intricate and largely rely on specific problem-tailored procedures and instance-specific neighborhood-evaluation procedures.

In an effort to contribute toward these issues and gaps, this paper presents a solution framework broadly applicable to a large variety of practical settings and different variants of one-to-many-to-one vehicle routing and scheduling problems. From the methodological viewpoint, the proposed approach introduces few user-defined parameters—compared to other approaches—and does not incorporate complex spatiotemporal decomposition schemes and heuristic restriction procedures to accelerate the neighborhood search. Although these mechanisms may have a strong impact on the efficiency and scalability towards large-scale problem instances, in many cases they are hard to implement for practical applications and heavily rely on the underlying structure of the problem instances (Gendreau and Tarantilis 2010). Finally, another aspect of the proposed framework is that we treat the fleet size as a decision variable, whereas other approaches set a fleet size limit and restart the search from scratch—after decrementing the number of vehicles—if no feasible solutions are found.

3. Problem Definition and Notation

Following the notation provided by Bent and van Hentenryck (2004) for the VRPTW, one-to-many-to-one vehicle routing and scheduling problems can be defined on a complete directed graph $G = (V, A)$ with a set of nodes $V := \{0\} \cup \{n+m+1\} \cup L \cup B$ and a set of arcs $A = \{(i, j) \in V \times V\colon i \neq j\}$, where the subsets $L = \{1, \ldots, n\}$ and $B = \{n+1, \ldots, n+m\}$ correspond to the linehaul and backhaul customer subsets. To that end, each node (customer) $i \in V \backslash \{0, n+m+1\}$ is associated with a demand d_i to be delivered or a demand p_i to be collected; a service time s_i; and a time window $[e_i, l_i]$, where e_i and l_i represent the earliest and latest allowable arrival times, respectively. Note that $d_i > 0$ and $p_i = 0$ for all customers $i \in L$, and similarly $d_j = 0$ and $p_j > 0$ for all customers $j \in B$. Nodes 0 and $n+m+1$ represent the depot (with fictitious demands and service times all equal to 0 as well as a time window $[e_0, l_0]$), where K identical vehicles with a given capacity Q are stationed. Each vehicle incurs a Euclidean travel cost $c_{ij} \in \mathbb{R}_+$ if it traverses the arc $(i, j) \in A$. Without loss of generality, we assume that the cost matrix $[c_{ij}]$ satisfies the triangle inequality (i.e., $c_{ij} + c_{ju} \geq c_{iu}$) and can be also used to measure the travel time t_{ij} from i to j (i.e., $c_{ij} = t_{ij}\ \forall (i, j) \in A$).

A vehicle route (or route for short) starts from the depot, visits a number of customers at most once, and returns to the depot. Let $\langle 0, v_1, \ldots, v_u, n + m + 1\rangle$ be a sequence of customers, where all v_i are different; the travel cost of this route r, denoted by t_r, is the cost of visiting all of its customers, i.e., $t_r = \sum_{i=0}^{u} c_{v_i v_{i+1}}$ (where $v_0 = 0, v_{u=1} = n + m + 1$), if the route is not empty and is 0 otherwise. To that end, a solution s (or routing plan) is a set of routes $\{r_1, \ldots, r_k\}$ $(k \leq K)$ visiting every customer exactly once; i.e., $\bigcup_{i=1}^{k} r_i = L \cup B$ and $r_i \cap r_j = \varnothing$ $(1 \leq i, j \leq k)$. Note that a routing plan assigns a unique successor i^+ and predecessor i^- to every customer i. Furthermore, for simplicity our definitions assume a single pair of the depot nodes 0 and $n + m + 1$; however, multiple copies are needed, one per vehicle route, to evaluate all subsequent properties.

A route is called feasible if both of the time window and capacity constraints are satisfied. Regarding the former, vehicles must arrive at customers before the end of the time window l_i. They may arrive early, but they have to wait until time e_i to be serviced. Given that e_0 represents the departure time of all vehicles from the depot, the departure time δ_i of customer i is defined recursively as follows:

$$\begin{cases} \delta_0 = 0, \\ \delta_i = \max(\delta_{i^-} + c_{i^- i}, e_i) + s_i \quad \forall\, i \in L \cup B. \end{cases} \tag{1}$$

The earliest service time of customer i, denoted by a_i, is defined as follows:

$$a_i = \max(\delta_{i^-} + c_{i^- i}, e_i) \quad \forall\, i \in L \cup B. \tag{2}$$

A routing plan satisfies the time window constraint for customer i if $a_i \leq l_i$. Similarly, the earliest arrival time $a(r)$ of a route is given by $\delta_{v_u} + c_{v_u v_{n+m+1}}$, where v_u is the last customer of the route. Therefore, a routing plan s satisfies the time window constraint for the depot if $a(r) \leq l_0\ \forall\, r \in s$. On this basis, the latest arrival time z_i for customer i can be defined recursively as follows:

$$\begin{cases} z_0 = l_0 \\ z_i = \min(z_{i^+} - c_{ii^+} - s_i, l_i) \quad \forall\, i \in V \backslash \{0\}. \end{cases} \tag{3}$$

Regarding capacity constraints, a route is capacity feasible if the accumulated load at any of its customers along the route does not exceed the vehicle's maximum capacity. Let h_i and g_i denote the delivery and pickup loads carried immediately after the service of customer i, respectively. Observe that h_0 and g_{n+m+1} represent the total delivery demand $\sum_{i=0}^{u} d_i$ and the total pickup demand $\sum_{i=0}^{u} p_i$ of route r. Given that a vehicle returns to the depot with zero delivery loads, h_i of customer i can be defined recursively as follows:

$$\begin{cases} h_{n+m+1} = 0, \\ h_i = h_{i^+} + d_{i^+} \quad \forall\, i \in V. \end{cases} \tag{4}$$

In a manner similar, a vehicle departs from the depot with zero pickup loads; therefore, g_i of customer i is defined recursively as

$$\begin{cases} g_0 = 0, \\ g_i = g_{i^-} + p_i \quad \forall\, i \in V. \end{cases} \tag{5}$$

The capacity constraint for customer i is satisfied if $h_i + g_i \leq Q$. However, in cases where restrictions on the mixing level of backhaul and linehaul customers are imposed, relevant to the delivery load of a vehicle, a routing plan satisfies the visiting sequence restrictions if $h_j < \xi Q\ \forall\, j \in B$, where the threshold parameter ξ $(0 \leq \xi \leq 1)$ enforces an upper bound on the delivery load such that a pickup is allowed. Observe that this restriction is redundant if $\xi = 1$ (case of mixed backhauls), whereas backhaul customers (if any) must be visited after linehaul customers (if any) along each route if $\xi = 0$ (case of clustered backhauls).

Let q_i denote the vehicle's delivery load slack of customer i. If a route r contains only linehaul customers, then q_i is $Q - h_0\ \forall\, i \in r$. However, if both linehaul and backhaul customers are present with predefined mixing restrictions, then q_i can be recursively defined with respect to the first encountered backhaul customer u along the route as follows:

$$q_i = \begin{cases} Q - h_0 & 0 \leq i < u, \\ \min(\xi Q - h_i, Q - h_0) & i = u, \\ \min(q_{i^-}, Q - h_i - g_i) & i > u. \end{cases} \tag{6}$$

Similarly, let b_i denote the pickup load slack of customer i. If a route contains only backhaul customers, then $b_i = Q - g_{n+m+1}\ \forall\, i \in r$. On the other hand, if a route contains both linehaul and backhaul customers with delivery load based mixing restrictions and given that $d_{n+m+1} = Q - g_{n+m+1}$, then b_i can be recursively defined as follows:

$$b_i = \begin{cases} \min(b_{i^+}, Q - h_i - g_i) & \\ \quad \text{if } h_i \leq \xi Q, & \forall\, i \in V. \\ 0 \quad \text{otherwise}, & \end{cases} \tag{7}$$

On the basis of the above, a routing plan $s = \{r_1, \ldots, r_k\}$ is feasible if the capacity and the time window constraints defined earlier are satisfied for each route. The total travel distance of s is defined as $f(s) = \sum_{r=1}^{k} t(r)$. Following earlier works of the literature, the goal is to find a feasible solution that minimizes the number of routes $|s|$ (primary objective) and, in case of ties, the total distance traveled $f(s)$ (secondary objective).

4. Adaptive Path Relinking

4.1. Solution Framework

This section describes the proposed Adaptive Path Relinking (APR) for solving one-to-many-to-one vehicle routing and scheduling problems with clustered and mixed backhauls. APR is a population-based approach that aims to evolve monotonically a so-called reference set of elite solutions by means of exploring trajectories between these solutions via Path Relinking (PR; Glover 1999). For this purpose, a fairly simple and flexible PR solution framework is developed that incorporates novel methods capable of tracking promising solution components, constructing combinations of multiple elite solutions, and improving the quality of intermediate solutions encountered during PR via local search.

Based on the expectation that high quality solutions have common characteristics (e.g., small inner customer distances) and features (e.g., identical permutation of customers; Tarantilis 2005), a path between two solutions in a neighborhood space will yield new intermediate solutions that share a significant amount of attributes with them in varying "mixes" (Glover 1999). In particular, upon starting from an initial solution, a necessary condition to generate the desired paths is to apply local moves that progressively introduce attributes contributed by a guiding solution. To that end, multiparent path generation possibilities also emerge and enhance the opportunity to exploit information contained in the union of elite solutions. For example, as described in Glover (1999) one may consider the combined attributes (possibly weighted) provided by a set of guiding solutions that will determine which local moves are given higher priority during the path generation process.

On this basis, the proposed APR framework departs from traditional PR implementations found in the literature and takes into account multiple solutions simultaneously as a foundation for creating combinations. The main effort is first to identify, select, and combine systematically promising solution components encountered during the search and second to extrapolate the search process beyond the regions spanned by the elite solutions maintained within the reference set. For this purpose, provisional solutions are generated at each iteration that are used as "guiding" points for performing search trajectories/paths from elite "initial" solutions of the reference set. This is done by ruining and reconstructing part of a "parent" reference solution using promising building blocks contained in the union of elite solutions maintained within the reference set. Clearly, one may expect that the paths generated via this multiparent combination scheme "relink" points in the solution space in ways not achieved in the previous search history.

The overall framework is displayed in Algorithm 1. Starting from an empty reference set R, the diversification generation method described in §4.2 is applied. The aim is to provide a good initial sampling of promising areas of the solution space as well as to ensure that the solutions forming the initial reference set are diversified. Subsequently, the core APR framework is triggered that manipulates R by means of exploring trajectories. At each iteration, a provisional "guiding" solution s_g is generated (one per reference solution s_t) via the adaptive recurrence-based ruin-and-recreate method presented in §4.3. Subsequently, an initial solution s_i is selected, and the path-relinking mechanism described in §4.5 is applied. Among the sequence of feasible and infeasible intermediate solutions s_u generated during the relinking process, a subset G of them is selected and further improved via the local search improvement method described in §4.6. On return, the best encountered feasible or infeasible solution competes to update the reference set. This process is controlled via the reference set update method presented in §4.7.

Algorithm 1 (Adaptive Path Relinking)
Input: μ, v, z, and p_c //User-defined parameters
Output: Best encountered feasible solution s_*

```
R ← Diversification Generation Method(μ, v, z)
while termination conditions do
    for all reference solutions s_t ∈ R do
        G ← ∅
        s_g ← Construction of Guiding
              Solution(s_t, p_c, R)
        s_i ← Selection of Initial Solution(R)
        G ← Path Generation Method(s_i, s_g, μ)
        for all intermediate solutions s_u ∈ G do
            s_m ← Local Search(s_u, v, z)
            if s_m is feasible and improves s_* then
                s_* ← s_m
            end if
            R ← Update Reference Set(s_m)
        end for
    end for
end while
```

The above described APR framework introduces four main user-defined parameters, namely the size μ

of the reference set R, a variation parameter p_c (see §4.3), and parameters v and z that are used to regulate the local search (see §4.6). The termination condition we adopt (Line 2 of Algorithm 1) enforces a maximum allowed CPU time consumption limit that varies with respect to the total number of customers. Other alternatives are to set a maximum number of inner APR iterations (Line 3 of Algorithm 1) and/or to break the execution if R has not been updated with the completion of a full inner iteration.

4.2. Diversification Generation Method

The goal is to initialize the reference set R with a collection of diverse solutions. For this purpose, a parallel insertion-based construction heuristic algorithm is utilized, coupled with a local search improvement method (see §4.6). The latter is applied for further improvement, and the new local optimum solution is added to R. The oscillations between new solution construction and local search are repeated until R is filled with μ solutions.

Given an initial set of one or more vehicle routes, an unrouted customer u is inserted between customers i and j of the current partial constructed route(s) that minimally increases the traveling distance; i.e., $c_{iu} + c_{uj} - c_{ij}$. Let $\pi_{ij,u}$ denote the insertion cost of u. For every feasible insertion position into a route r of a partial solution σ, the minimum insertion cost $\pi_{r,u} = \min_{i,j\in r} \pi_{ij,u}$ is found. The overall minimum insertion cost $\pi_{r',u}$ corresponds to the $\min_{r\in\sigma} \pi_{r,u}$ and denotes the best insertion position of u. The above solution construction process is repeated until all customers are assigned to vehicles. If at some iteration an unassigned customer cannot be inserted into any of the existing set of routes, a new "seed" customer is identified, and a route is initialized. The "seed" customers are determined such that the most time constrained are considered first (i.e., minimum time gap between the latest service time and the time needed to travel from the depot $l_i - c_{0i}$).

In an effort to ensure an adequate level of diversity, during the construction process we forbid the formation of consecutive triads of customers that appear in solutions added previously to R. More specifically, whenever the insertion of a customer u between customers i and j is considered, along with feasibility we also examine whether both arcs (i, u) and (u, j) are traversed—in this particular order and direction—in any of the reference solutions. If this is the case, we exclude from further consideration this insertion position for u. Note that this accessibility restriction is not applied during the local search improvement phase; however, we observed that a good sampling of the solution space is nevertheless achieved. Algorithm 2 provides an overview of the proposed Diversification Generation Method.

Algorithm 2 (Diversification Generation Method)
Input: μ, v, and z
Output: Initial reference set R

```
R ← ∅
while |R| < μ do
    s ← ∅
    while s is not complete do
        for all unrouted customers u do
            for all routes r ∈ s do
                for all accessible and feasible insertion
                    positions i, j ∈ r do
                    π_{ij,u} ← c_{iu} + c_{uj} − c_{ij}
                end for
                π_{r,u} ← min_{i,j∈r} π_{ij,u}
            end for
            if max_{r∈s} π_{r,u} = 0 then
                Initialize a route with a new "seed"
                    customer
            end if
        end for
        u' ← min_{u∉s, r∈s} π_{r,u}
        s ← s ∪ {u'}
    end while
    s_m ← Local Search(s, v, z)
    R ← R ∪ {s_m}
end while.
```

4.3. Construction of Provisional Guiding Solutions

All solutions of the reference set R are selected one by one, and provisional guiding solutions are produced (one per parent reference solution) via a novel ruin-and-recreate scheme. The main effort is to reconstruct part of a parent reference solution using for this purpose building blocks often encountered in other reference solutions. In particular, given a parent reference solution, a set of customers that seems to be "misplaced" is initially removed (see §4.3.1) using an adaptive threshold criterion in a probabilistic fashion. Next, the partially ruined solution is reconstructed by inserting the previously removed customers in "promising" high priority insertion positions with some probability (see §4.3.2). Observe that compared to traditional evolutionary algorithms, this approach encompasses both recombination and mutation mechanisms through the exchange of solution components from multiple solutions and the partial reconstruction of parent solutions, respectively.

4.3.1. Adaptive Threshold-Based Customer Removal Procedure. Various customer removal operators have been proposed in the literature, especially in the context of LNS approaches (Pisinger and Ropke 2006). Most of them take into account spatiotemporal (e.g., remove customers based on geographical proximity) and/or cost related criteria (e.g., remove customers that generate long detours), and some randomness during the selection of customers is also introduced. Inspired by earlier works of Repoussis, Tarantilis, and Ioannou (2009, 2010), a novel customer removal operator has been developed that takes into account the previous search history.

The proposed operator considers two main properties: the appearance frequency for each arc (pair of customers) and the diversity of the solutions making up R. Let $o_{i,j}$ denote the appearance frequency of an arc (i, j) within R and T denote a so-called acceptance threshold. The latter is used to indicate whether an arc can be considered as "promising" or not. The acceptance threshold T can be simply deduced from the cardinality of $|R|$ times the fraction of the average similarity $M_R^{s_*}$ of reference set solutions with respect to the best encountered feasible solution $s_* \in R$ over the total number of arcs $n+m+k$ contained in s_*; i.e.,

$$T = \frac{M_R^{s_*}}{n+m+k}|R|, \tag{8}$$

where $n+m$ is the total number of customers and k is the total number of vehicles. The average similarity of all solutions $s \in R - \{s_*\}$ with respect to a solution s_* can be calculated as follows (see also Ho and Gendreau 2006):

$$M_R^{s_*} = \frac{\sum_{s\in R-\{s_*\}} \sum_{(i,\,j)\in s,\, s_*} I_{ij}^{ss_*}}{|R|-1}, \tag{9}$$

where $I_{i,j}$ is an indication of whether two solutions s_a and s_b exhibit arc (i, j).

$$I_{i,j}^{s_a s_b} = \begin{cases} 1 & \text{if both solutions } s_a \text{ and } \\ & \quad s_b \text{ exhibit arc } (i, j), \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

An important feature of Equation (8) is that the values of T are self-adjusted with respect to the average similarity among reference solutions. For example, the larger the similarity is, the larger the threshold value becomes. Hence, at the early stages of the search process where large distances among solutions are observed, T is forced to small values in order to achieve convergence velocity. On the other hand, as reference solutions tend to converge during the course of evolution, T gradually increases in order to ensure convergence reliability.

On the basis of above, the proposed customer removal operator utilizes the values of T as a measure to indicate whether a node is "misplaced" or not. From the implementation viewpoint, all nodes of a given reference solution are candidates for removal with some probability. If the appearance frequencies of the connecting arcs of a node u between customers i and j, i.e., o_{iu} and o_{uj}, are both greater than or equal to T, then the corresponding node is removed from the solution with a probability $1-p_c$; otherwise, it is removed with a probability p_c. This procedure is repeated for all customer nodes in a sequential fashion. Note that in our experiments, p_c is set equal to 0.75. Observe that if p_c is set to 0.5, then the outcome is a purely random customer removal scheme.

On the other hand, if the starting reference solution is infeasible with respect to the customer's start time of service, then the customers with violated time windows (if any) are removed. Finally, an effort is also made to remove the surplus (if any) vehicle route(s) in order to favor solutions with minimal fleet size. In particular, if the total number of vehicle routes is greater than a precalculated lower bound (for details, see Lim and Zhang 2007, Kontoravdis and Bard 1995) or the known lowest number of vehicles, then the customers served by the vehicle route(s) with the smallest cardinality are also removed.

4.3.2. Probabilistic Recurrence-Based Solution Reconstruction Method. The removed customers are reinserted to the partially ruined solution via the insertion-based construction scheme described earlier. However, instead of looking myopically to minimize the insertion cost of unrouted customers, we also seek to maximize the sum of appearance frequencies of the corresponding connected edges between adjacent customers in a probabilistic fashion. On this basis, the proposed probabilistic recurrence-based reconstruction method can be described as follows: an unrouted customer u is inserted between routed customers i and j that either maximally increases the resulting sum of appearance frequencies, i.e., $o_{iu}+o_{uj}-o_{ij}$, with a probability p_c or minimally increases the distance traveled, i.e., $c_{iu}+c_{uj}-c_{ij}$, with a probability $1-p_c$. For all unrouted customers, all insertion positions (both feasible and infeasible) are examined, but in case of ties feasible insertion positions and/or insertion positions that cause the shortest detours are prioritized. Note that the fleet size remains fixed, and at the end of the reconstruction process the provisional guiding solution can be either feasible or infeasible in terms of time window and/or capacity constraints.

Below, Algorithm 3 provides an overview of the proposed method for generating provisional guiding solutions.

Algorithm 3 (Construction of Guiding Solution**)**
Input: s_t, p_c, and R
Output: Guiding provisional solution s_g
$T \leftarrow (M_R^{s_*}/(n+m+k))|R|$
$s_g \leftarrow s_t$
for all consecutive triads of nodes (i, u, j) exhibited in s_g **do**
  if $o_{iu} \geq T$ AND $o_{uj} \geq T$ **then**
    Remove u from s_g with a probability $1-p_c$
  else

```
        Remove u from s_g with a probability p_c
    end if
end for
Remove all customers with violated time
    windows
Remove all customers served by surplus
    vehicle routes
while s_g is not complete do
    for all unrouted customers u do
        for all routes r ∈ s_g do
            for all insertion positions i, j ∈ r do
                π_{ij,u} ← c_{iu} + c_{uj} − c_{ij} and
                    π'_{ij,u} ← o_{iu} + o_{uj} − o_{ij}
            end for
            π_{r,u} ← min_{i,j∈r} π_{ij,u} and
                π'_{r,u} ← max_{i,j∈r} π'_{ij,u}
        end for
    end for
    if rand(0,1) ≤ p_c then u ← max_{u'∉s_g, r∈s_g} π'_{r,u'}
    else u ← min_{u'∉s_g, r∈s_g} π_{r,u'}
    s_g ← s_g ∪ {u}
end while.
```

4.4. Selection of Initial Solutions

Having generated the provisional guiding solution, the next steps are to select an initial solution s_i from R and to trigger the relinking mechanism. Regarding the former, various selection strategies have been proposed in the literature, including among others random selection, selection of the fittest solutions and selection of distant solutions (see also Ho and Gendreau 2006). In the context of the proposed APR solution framework, a random selection scheme is applied, with equal selection probability for all reference solutions.

4.5. Path Generation Method

Staring from the initial solution s_i, the proposed path generation method selects and applies local moves that progressively introduce attributes contributed by the provisional guiding solution s_g, such that the Hamming distance H_{s_i,s_g} between s_i and s_g is reduced. To that end, a sequence of intermediate solutions $s_i = s_1, \ldots, s_{\mu-1}, s_\mu = s_g$ is produced that joins s_i and s_g. At each iteration, the solution s_μ is produced from $s_{\mu-1}$ by choosing either an inter-route 2-Opt or an intra-route 0–1 Relocate (equal selection probability) local move that decreases the most (best-accept strategy) the corresponding distance H_{s_μ, s_g}. In case of ties, the local move with the lowest distance traveling cost is applied.

There are various ways to express the Hamming distance $H_{x',x''}$ between two solutions x' and x''. Herein, we measure the amount of common arcs (also known as the broken pairs distance) that can be expressed as follows:

$$H_{x',x''} = \sum_{(i,j)\in x',x''} I_{ij}^{x'x''}, \tag{11}$$

where $I_{ij}^{x'x''}$ is the binary indicator defined in (10). To that end, given a provisional guiding solution s_g, the solution s with the minimum H_{s,s_g}, $\forall s \in R$ is selected as the initial solution.

Basing the relinking process on oscillations between different types of edge-exchange structures provides a useful variation. Furthermore, the proposed relinking mechanism also benefits from tunneling. In particular, time windows and capacity constraints are relaxed, and infeasible solutions are also accepted as intermediate solutions. Observe that tunneling may offer a chance to reach solutions that might otherwise be bypassed.

Finally, given the sequence of intermediate solutions, a subset G is selected for further improvement via local search. Overall, a total of $\mu/4$ intermediate solutions is selected, plus the guiding provisional solution, and added to the subset. To that end, both feasible and infeasible solutions are candidates for selection. For this purpose, we divide the generated path into equal sized sections (i.e., $4H_{s_i s_g}/\mu$), and we select the locally minimum solutions from each section. Among feasible and infeasible intermediate solutions, the feasible solution with the minimum number of vehicles and distance traveling cost is chosen, whereas among infeasible solutions the one that minimizes the penalized cost (as defined later in Equation (12)) with the least number of vehicles is selected.

4.6. Local Search Improvement Method

As mentioned earlier, a subset G of solutions generated during the path generation method is selected and further improved by means of a local search improvement method. For this purpose, a tabu search based short-term memory local search algorithm is employed. In broad terms, the proposed local search scheme seeks to explore the solution space $\mathcal{S}$ by moving at each iteration from a solution s to the best admissible solution s' in a subset $\Omega_y(s)$ of a neighborhood structure y. The short term memory records the most recently visited solutions and prevents revisiting them for a predefined number of iterations v (tabu tenure). The tabu status of a neighboring solution can be overridden only if predefined aspiration criteria are met. The overall procedure iterates until some termination conditions are met and the best encountered solution s_* is returned.

Local search is important for the fast progression toward high quality regions. However, it typically consumes more than 80% to 85% of the overall computational effort. Therefore, high computational efficiency is required. To that end, three aspects are decisive for the implementation as well as the performance of the above described local search scheme: the definition of the search space; the choice of neighborhood structures and evaluation techniques; and the definition of tabu list, admissible solutions, and aspiration conditions.

4.6.1. Search Space. Let a set of routes $\{r_1, \ldots, r_k\}$ make up solution s. If solution s is feasible, then the search space is defined with respect to the total distance traveled of all its routes, i.e., $f(s) = \sum_{r=1}^{k} t(r)$, and it is strictly confined over the feasible region. On the contrary, if the starting solution s is infeasible, then the search space is defined with respect to the total distance traveled plus the weighted sum of time window and capacity violations. Let ω^D and ω^Q represent the penalty coefficients, and let $P_D(s)$ and $P_Q(s)$ account for time window and capacity violations, respectively. To that end, a penalized cost $\phi(s)$ is defined as follows:

$$\phi(s) = f(s) + \omega^D P_D(s) + \omega^Q P_Q(s) \tag{12}$$

In the context of the so-called soft time windows, the amount of late arrivals can be measured as the excess compared to the latest allowable arrival times, e.g., $\max\{a_i - l_i, 0\}$. A more efficient approach—proposed recently by Nagata, Braysy, and Dullaert (2010) and further elaborated by Vidal et al. (2011)—is to pay a so-called "time-wrap" to reach the end of the time window, upon a late arrival to a customer. In particular, if a vehicle arrives late at a customer u (i.e., $a_u > l_u$), the vehicle travels back in time to l_u to start the service without delay, but at the expense of paying a penalty $a_u - l_u$. Observe that time wraps are symmetric to waiting times, although waiting times are not penalized (Vidal et al. 2011). To that end, when moving from customer i to $i+1$, the time-wrap $\zeta_{i,\,i+1}$ is given by $\max\{a_i + s_i + c_{ii+1} - a_{i+1}, 0\}$. Therefore, the time-wrap use $\zeta(r)$ of a route r—that serves a sequence of customers $\langle 0, v_1, \ldots, v_u, n+m+1\rangle$—can be defined as the sum of the penalties the vehicle must pay to service all customers and to arrive at the depot without delay; i.e., $\zeta(r) = \sum_{i=0}^{u} \zeta_{v_i v_{i+1}}$. To that end, $P_D(s)$ can be expressed as follows:

$$P_D(s) = \sum_{r=1}^{k} \zeta(r). \tag{13}$$

Regarding capacity constraints, various types of violations may occur because of loading and mixing level restrictions between linehaul and backhaul customers. At first, the total delivery and the total pickup demands of a route may exceed the vehicle capacity. Let $h(r)$ and $g(r)$ represent the total delivery demand $\sum_{i=0}^{u} d_i$ and the total pickup demand $\sum_{i=0}^{u} p_i$ of route r. The excess delivery and pickup demand of the route as a whole with respect to the vehicle capacity can be expressed as $\max\{0, h(r) - Q\}$ and $\max\{0, g(r) - Q\}$, respectively. Another type of capacity violation occurs when the vehicle's carrying load along the route, i.e., $h_i + g_i$, exceeds the vehicle's capacity. Let $d(r)$ denote the sum of the excess load during the customer service; i.e.,

$$d(r) = \sum_{i=1}^{u} \max\{h_i + g_i - Q, 0\}.$$

Finally, capacity violations may also occur because of mixing restrictions between linehaul and backhaul customers; i.e., $h_j > \xi Q$ during the service of backhaul customers along the route. Let us define $\theta(r)$ the sum of excess delivery load due to visiting sequence restrictions as $\theta(r) = \sum_{i \in B \cap r} \max\{h_i - \xi Q, 0\}$. To that end, $P_Q(s)$ can be defined as follows:

$$P_Q(s) = \sum_{r=1}^{k} \big\{ \big(\max\{0, h(r) - Q\} + \max\{0, g(r) - Q\} + d(r) + \theta(r) \big) \big\}. \tag{14}$$

Penalized cost functions of the above form can be applied for the controlled exploration of both feasible and infeasible regions. As reported by Vidal et al. (2011) and Hashimoto and Yagiura (2008), such an approach may enhance the performance of the search process. However, in the context of one-to-many-to-one vehicle routing and scheduling problems, the evaluation of penalties for capacity violations is more demanding compared to time window violations, and significant effort may be required to restore feasibility for problem instances with many backhaul customers and high rates of capacity utilization. For this reason, once the feasibility of an infeasible solution is restored, we don't allow the search to enter again the infeasible region. To that end, it is worth highlighting that the proposed penalized cost function is broadly defined and can be directly applied to problem variants with and without time window and route duration restrictions as well as with and without the presence of both pickup and delivery customers.

Finally, an important issue are the settings for the penalty coefficients ω^D and ω^Q. Nagata, Braysy, and Dullaert (2010) examined a range between 0.01 and 100 for both of them and conclude that values equal to 1 ensure a consistent performance. On the other hand, Vidal et al. (2011) initially select small values, and they consider two readjustment options by factors of 10 and 100 if a solution remains infeasible. In the proposed local search improvement method, we adopt a similar self-adjustment scheme that works as follows: initially, the penalty coefficients are globally set to 1. As long as the incumbent solution remains infeasible, and no improvement is observed with respect to the penalized cost, the coefficients are increased at each iteration by a factor of 10; however, they are reinitialized to 1 whenever a new local optimum is found during the local search process.

4.6.2. Neighborhood Structures, Feasibility Checks, and Evaluation Methods. As mentioned earlier, given the allowed set of neighbors $\Omega_y(s)$, the best admissible neighbor s' (i.e., $\min_{s' \in \Omega_y(s)}\{\phi(s')\}$) replaces the current solution s at each iteration of the local search. To that end, the neighborhood structures y used within the proposed implementation are based on traditional edge-exchange local moves, namely intra- and inter-route 2-Opt, 1–0 Relocate, and 1–1 Exchange (Kindervater and Savelsbergh 1998). Intra-route 2-Opt reverses the visiting sequence of a segment $(v_i^r, \ldots, v_j^r)$, whereas inter-route 2-Opt* swaps two segments $(v_i^r, \ldots, v_{n+m+1}^r)$ and $(v_{i'}^{r'}, \ldots, v_{n+m+1}^{r'})$, both involving the ending depot. On the other hand, 1–0 Relocate and 1–1 Exchange swap two disjoint segments $(v_i^r, \ldots, v_j^r)$ and $(v_{i'}^{r'}, \ldots, v_{j'}^{r'})$ that contain 0 and 1 customer visits, respectively.

The size of the above neighborhood structures is $O(n^2)$ and involves a constant number of edge-exchanges. During the local search, the oscillations among them are stochastic with equal selection probability. For their evaluation, a lexicographic ordering search is followed, coupled with feasibility as well as gain based pruning techniques. In our case, feasibility checks for both time window and capacity constraints can be performed in $O(1)$ time. For this purpose, we keep track of the vehicle's departure time δ_i, the latest allowable arrival time z_i, the delivery and pickup loads h_i and g_i, and the delivery and pickup slacks q_i and b_i for each node as well as for all consecutive sequences of nodes as dictated by the current solution. Observe that this information is sufficient to perform all feasibility checks in constant time.

Consider for example the insertion of a customer u between customers i and j of a route r. The arrival time at customer u is $\delta_i + c_{iu}$, and we first check if this is less than or equal to l_u. To that end, the new arrival time at j is $\max(\delta_i + c_{iu}, e_u) + s_u + c_{uj}$, and we check if the difference between the new and the old arrival time at j is less than or equal to z_j. Regarding capacity constraints, if u is a linehaul customer, then we check if $d_u \leq h(r)$, $h_i + g_i + d_u \leq Q$ and $d_u \leq q_i$; otherwise, if u is a backhaul customer, then we check if $p_u \leq g(r)$, $h_i + g_i + p_u \leq Q$, and $p_u \leq b_j$ in order to determine if the insertion of u is feasible.

When handling edge-exchange neighborhood structures for feasible solutions, the only necessity is to track the differences with respect to the arcs that change state (being deleted or added) in order to evaluate the distance traveling cost $f(s)$ of a neighboring solution s. However, when infeasible solutions are treated according to the penalized cost function (12), the changes of time window and capacity infeasibility must be also computed for the evaluation of local moves. Regarding time windows, we adopt the evaluation procedure and the reoptimization data structures proposed in Vidal et al. (2011) to compute the changes of time-wrap use (see Equation (13)) in $O(1)$ amortized time. On the other hand, the evaluation of capacity infeasibility is more demanding. In particular, it requires constant time to compute the first two terms of Equation (14) and $O(n+m)$ time to determine $d(r)$ and $\theta(r)$. However, observe that only a part of the route, starting from the first backhaul customer and up to the last linehaul customer, must be actually reevaluated using the data structures mentioned earlier. Thus, in practical terms it may take almost constant time to evaluate $d(r)$ and $\theta(r)$, at least for short-haul problem instances with few customers per route.

Finally, it is unnecessary to evaluate the whole neighborhood(s) at each iteration of the local search process because the effect of edge-exchanges is limited to the routes they modify, which are at most two in our case. Thus, once a local move is applied, only the modified routes must be reevaluated, whereas the values of unmodified routes are still valid (see also Repoussis, Tarantilis, and Ioannou 2009). Although this type of treatment increases the implementation complexity, notable reductions in the overall computational time can be observed.

4.6.3. Tabu List, Aspiration Conditions, and Restrictions. To avoid cycling, both the forward and reversal local move attributes, i.e., edges being added and deleted, of the corresponding neighborhood structure are stored within the tabu list, and we forbid the formation of these edges for a number of iterations v. The tabu status is ignored only if the incumbent solution is improved with respect to the feasible or infeasible search mode. Regarding the neighborhood structures, nonspatial or spatiotemporal heuristic restrictions are imposed in order to reduce the neighborhood size and to accelerate the neighborhood evaluation process because of the mixing restrictions between backhaul and linehaul customers. However, we forbid the reversal of customer segments with more than six nodes (intra-route 2-Opt), and also we do not allow the addition of customers (inter-route 1–0 Relocate and 2-Opt) to the vehicle route with the smallest cardinality, if the total number of vehicles is greater than the predetermined lower bound. On the other hand, if the incumbent solution is infeasible, we exclude during the neighborhood evaluation local moves between feasible routes, and we focus only on local moves between feasible and infeasible routes.

Below, Algorithm 4 provides an overview of the proposed local search improvement method. Regarding termination conditions, a maximum number of iterations z without observing any further improvement is considered.

Algorithm 4 (Local Search Improvement Method)
Input: s, v, and z
Output: Local optimum solution s_m

$s_m \leftarrow s$, $z_t \leftarrow 0$, $\omega^D \leftarrow 1$ and $\omega^Q \leftarrow 1$
Initialize Data Structures(s)
while $z_t < z$ **do**
  $y \leftarrow$ Random Selection()
  $\Omega_y(s) \leftarrow$ Neighborhood Evaluation(s, y)
  $s \leftarrow \min_{s' \in \Omega_y(s)} \phi(s')$
  Update Tabu List(s, y, v)
  if $\phi(s) < \phi(s_m)$ **then** $z_t \leftarrow 0$ and $s_m \leftarrow s$
  else $z_t \leftarrow z_t + 1$
  Update Penalty Coefficients(ω^D, ω^Q)
  Update Data Structures(s)
end while.

4.7. Reference Set Update Method

An important component of the proposed APR framework is the updating mechanism of the reference set because useful information about the structural form of optimal solutions is typically contained in a suitably diverse collection of elite solutions. Thus, there must be a balance between quality and diversity among the reference solutions in order to ensure the long term evolvability of the reference set and to guide the search process toward distant as well as promising regions. In our case, the reference set may contain both feasible and infeasible solutions; however, the number of feasible solutions is always maintained greater than or equal to the number of infeasible solutions.

The update criteria we adopt are deterministic and account attractiveness in terms of feasibility, number of vehicles, traveling cost, penalized cost, and level of similarity between solutions with respect to the current local optimum reference solution in a hierarchical order. Let b_f and w_f denote the best and the worst feasible solutions and b_{inf} and w_{inf} denote the best and the worst infeasible solutions of R, respectively. Depending on the feasibility status of the candidate solution s for insertion, the reference set update conditions are determined as (i) a feasible solution s replaces a feasible reference solution s', if $f(s) < f(s')$, $|s| \leq |s'|$ and $H_{s,b_f} > H_{s',b_f}$, whereas in the special case where s improves the current best, i.e., $f(s) < f(b_f)$ and $|s| \leq |b_f|$, then s replaces w_f; (ii) a feasible (or infeasible) solution s replaces a feasible (or infeasible) reference solution s' if $|s| < |s'|$; (iii) a feasible solution s always replaces an infeasible reference solution s' if $|s| \leq |s'|$; and, finally, (iv) an infeasible solution s replaces an infeasible reference solution s', if $|s| \leq |s'|$, $H_{s,b_{inf}} > H_{s',b_{inf}}$, and $\phi(s) < \phi(s')$, whereas in the special case where s improves the current infeasible best, i.e., $\phi(s) < \phi(b_{inf})$ and $|s| \leq |b_{inf}|$, then s replaces w_{inf}.

Finally, note that during the reference set update procedure, the penalty coefficients for the evaluation of the penalized cost are set equal to a large number (i.e., $1e+3$). The goal is to provide a common basis for comparison among reference and candidate for insertion solutions because the penalty coefficients vary during the local search process. Observe also this large value for the penalty coefficients aims to direct the search toward feasible solutions.

5. Computational Results

5.1. Benchmark Data Sets and Experimental Set Up

Computational results for vehicle routing and scheduling problems, including VRPTW, VRPCBTW, and VRPMBTW instances, are typically ranked according to a hierarchical objective function. The primary objective is to minimize the total number of vehicles, and for the same number of vehicles, the secondary objective is to minimize the total distance traveled by the vehicles. However, these two objectives can be either conflicting or complementary because the reduction of the total number of vehicles may either increase or reduce, respectively, the total traveling distance. Thus, in order for the competition between different algorithms to be fair, comparisons are valid only if the above described hierarchy of objectives is followed.

On the basis of the above, for the evaluation of the proposed solution approach several computational experiments are performed. Most state-of-the-art methods for the VRPCBTW adopt the Gelinas et al. (1995) data set. The set consists of 15 problem instances with up to 100 customers. In particular, there are five categories according to the density of customer time windows, and each category is further divided into three different groups (i.e., A, B, and C) according to the percentage of backhaul customers with respect to the total number of customers (i.e., 10%, 30%, and 50%). The Cartesian coordinates of customers are randomly generated from a uniform distribution. Finally, it is worth mentioning that these problem instances have tight time windows and short scheduling horizons.

The characteristics of the Thangiah, Potvin, and Sun (1996) large scale data set for VRPCBTW instances are similar. The set consists of 12 problems with 250 customers and 12 problems with 500 customers. The percentage of backhaul customers with respect to the total number of customers ranges from 10% to 50%.

Note also that the 250-node problem instances were generated choosing the first 250 customers from the corresponding 500-node problem instances.

Regarding the VRPMBTW, Kontoravdis and Bard (1995) modified the long-haul problem groups R2 and RC2 of the VRPTW Solomon (1987) data set. Each of these sets contains between 8 and 12 100-node problems over a service area defined on a 100 × 100 grid, similar to those of Gelinas et al. (1995). Linehaul and backhaul customers are selected randomly. The Cartesian coordinates of customers in group R2 are randomly generated from a uniform distribution. Group C2 has clustered customers, whereas group RC2 contains semi-clustered customers (i.e., a combination of clustered and randomly distributed customers). Contrary to Gelinas et al. (1995), these problem instances consider long scheduling horizons, and time windows are adjusted to allow the service of many customers by each vehicle.

The above described benchmark data sets can be used as a basis for evaluating both VRPCBTW and VRPMBTW instances. As such, useful comparisons can be made toward the effect of backhaul customers as well as the effect of scheduling horizons. For this purpose, computational experiments using both Gelinas et al. (1995) and Kontoravdis and Bard (1995) benchmark data sets are reported for problem instances with both clustered and mixed backhaul customers (see §§5.5 and 5.6).

Following the earlier work of Reimann and Ulrich (2006), we study the effect of mixing linehaul and backhaul customers on the same vehicle routes by enforcing additional constraints relevant to the remaining capacity of the vehicles. In particular, a threshold parameter ξ is introduced that determines an upper bound on the linehaul capacity (the remaining sum of delivery demands) such that a pickup is allowed. For example, if ξ is set to 0.25, then a pickup is allowed as long as the remaining load for deliveries of the vehicle is less than 25% with respect to the maximum capacity of the vehicle. To that end, computational experiments with different threshold levels ξ are also reported, and the cost dimensions of different mixing strategies are analyzed (see §5.5).

Finally, an effort is made to demonstrate the generality as well as the robustness of the proposed APR framework, and computational experiments on the large-scale VRPTW data sets of Gehring and Homberger (1999) are also performed, followed by a comparative performance analysis with state-of-the-art solution methods (see §5.4). These data sets consist of 300 problem instances divided into five groups, i.e., G02, G04, G06, G08, and G10. The features of Solomon's (1987) data set are maintained; however, the set of customers has a much larger cardinality, i.e., 200, 400, 600, 800, and 1,000 customers, respectively.

5.2. Parameters Settings and Termination Conditions

The proposed APR framework incorporates four user-defined parameters: namely, the size μ of the reference set R, the variation parameter p_c, the tabu tenure v, and the number of local search iterations z without observing improvement. Based on our computational experience, one can determine very well performing and robust parameter settings with modest effort within reasonable value ranges. In what follows, the effect of each parameter is discussed, and suitable value ranges are provided.

Two parameters must be defined for the local search improvement method. Regarding parameter v a range between 20 and 40 is mostly used by standard tabu search implementations of the literature for intensification search and seems to fit well for VRPCBTW and VRPMBTW instances. On the other hand, one may expect that large values of z (termination condition) will increase the efficiency of the proposed local search method. However, a balance between efficiency and effectiveness is needed because large values of z may result in excessive computational time consumption. In our case, a range between 200 and 400 was found to provide a good compromise for problem instance with up to 100 customers, whereas for larger problem instances values less than 100 seem to provide consistent and robust performance.

Critical to the global search capability of the proposed solution approach is the size μ of R, in terms of convergence reliability and velocity at the cost of the expense in computational time. Very small values of μ may not be able to handle the amount of information included into the reference solutions, whereas large values may enhance the performance of evolution, however, at the expense of convergence speed. Ho and Gendreau (2006) report a maximum size of 30 solutions, whereas Glover (1999) suggests values below 20. In our case, a range between 12 and 30 was found to provide a good compromise for both small- and large-scale problem instances.

Also important is the role of the variation parameter p_c. Based on our computational experiments, an appropriate value range is between 0.5 and 0.9. Values close to 0.9 enhance the exploitation capacity because provisional guiding solutions tend to adhere and incorporate more and more the solution attributes of the reference solutions. On the contrary, as values of p_c decrease this effect diminishes, and therefore diversification is favored. In the proposed implementation, relatively large values (greater than 0.7) seem to perform best, in combination with the relatively strict reference set update criteria we employ that take into account the dissimilarity among solutions.

Table 1 **Detailed Results for VRPCBTW 100-Customer Gelinas et al. (1995) Data Set**

Problem		BK			TPS		RDH		RP		HKK		RU		APR		
instance	%B	NV	TD	REF	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	CT
R101A	10	22	1,818.86	RP	24	1,842.3	22	1,831.68	22	1,818.86	23	1,811.58	22	1,818.86	22	1,818.86	16
R101B	30	23	1,959.56	RP	25	1,930.2	23	1,999.16	23	1,959.56	26	1,937.79	23	1,959.56	23	**1,959.52**	68
R101C	50	24	1,909.84	HKK	25	1,942.3	24	1,945.29	24	1,939.10	24	1,909.84	24	1,939.10	24	1,939.10	75
R102A	10	19	1,653.19	RP	20	1,670.4	19	1,677.62	19	1,653.19	21	1,698.79	19	1,653.19	19	**1,653.18**	16
R102B	30	21	1,764.30	TPS	21	1,777.4	22	1,754.43	22	1,750.70	22	1,730.02	22	1,750.70	22	1,752.28	150
R102C	50	21	1,745.70	TPS	21	1,746.7	22	1,782.21	22	1,775.76	23	1,772.28	22	1,775.76	22	1,775.76	25
R103A	10	15	1,371.60	TPS	15	1,371.6	16	1,348.41	15	1,387.57	16	1,316.20	15	1,387.57	15	1,385.38	24
R103B	30	15	1,390.33	RP	16	1,477.6	16	1,395.88	15	1,390.33	18	1,441.00	15	1,390.33	15	**1,390.32**	17
R103C	50	16	1,486.56	ZC	17	1,549.1	17	1,467.66	17	1,456.48	19	1,451.32	17	1,456.48	17	1,456.48	73
R104A	10	11	1,084.17	RP	13	1,220.3	11	1,205.78	11	1,084.17	12	1,093.58	11	1,084.17	10	**1,203.44**	195
R104B	30	11	1,154.84	RP	12	1,302.5	12	1,128.30	11	1,154.84	13	1,177.93	11	1,154.84	11	1,154.84	209
R104C	50	11	1,191.38	RP	13	1,346.6	12	1,208.46	11	1,191.38	13	1,140.46	11	1,191.38	11	1,194.73	203
R105A	10	15	1,561.28	RP	17	1,607.4	16	1,544.81	15	1,561.28	17	1,672.72	15	1,561.28	15	**1,560.15**	151
R105B	30	16	1,583.30	RP	18	1,643.0	16	1,592.23	16	1,583.30	19	1,673.25	16	1,583.30	16	1,583.30	1
R105C	50	16	1,710.19	RP	18	1,657.4	17	1,633.01	16	1,710.19	17	1,699.31	16	1,710.19	16	**1,709.66**	54
MNV/MTD		17.07	1,559.01		18.33	1,605.65	17.67	1,567.66	17.27	1,561.11	18.87	1,568.40	17.27	1,561.11	**17.20**	**15,69.13**	
CNV/CTD		256	23,385.10		275	24,084.8	265	23,514.93	259	23,416.71	283	23,526.07	259	23,416.71	**258**	**23,537.01**	
Machine						NMM 33 M		PIII 900 M		PIV 1.5 G		—		PIV 1.5 G		Intel X7900 2.8 G	
Runs						(5 versions)		10		10		—		5		3	
MCT						14		150		114		—		75		85.05	
Rel. speed						—		≈0.9		1		—		1		10.24	
Norm. time						—		1,350		1,140		—		375		2,614	

Given the above value ranges, several intuitively selected combinations were experimentally tested, and we chose the one that yielded the best average output. In particular, the experimental results reported in subsequent sections consider fixed parameters with the following settings: $\mu = 20$, $v = 40$, $z = 180$, and $p_c = 0.75$. In order to identify statistically significant differences, three simulation runs on an Intel® Core™ 2 Extreme X7900 at 2.80 GHz are performed for each problem instance, unless otherwise stated. Finally, the algorithm was coded in Standard x86 C++, and a maximum allowed CPU time consumption limit is considered, depending on the total number of customers.

5.3. Comparative Analysis with Existing Results for VRPCBTW and VRPMBTW Instances

Tables 1–3 summarize the results obtained on the data sets of Gelinas et al. (1995) and Thangiah, Potvin, and Sun (1996) for the VRPCBTW. The first line

Table 2 **Detailed Results for VRPCBTW 250-Customer Thangiah, Potvin, and Sun (1996) Data Set**

Problem		BK		TPS			RP			APR			
instance	%B	NV	TD	NV	TD	CT	NV	TD	CT	NV	TD	CT	%Dev
BHR1DO.10	10	46	4,843.9	49	5,160	265	46	4,844.8	571	46	**4,825.45**	136	−0.38
BHR1DO.30	30	45	5,062.7	48	5,243	254	45	5,062.7	512	45	**5,033.78**	698	−0.57
BHR1DO.50	50	49	5,107.1	52	5,403.1	406	49	5,107.1	531	49	**5,105.86**	345	−0.02
BHR1UP.10	10	31	4,056.9	39	4,278.6	513	32	4,056.9	503	31	**4,022.38**	452	−0.85
BHR1UP.30	30	34	4,549.7	41	4,715.2	561	34	4,427.8	474	34	**4,547.84**	501	−0.04
BHR1UP.50	50	35	4,618.4	43	4,937.4	361	36	4,618.4	473	35	4,649.50	321	0.67
BHRC1DO.10	10	32	4,211.6	39	4,613.4	395	32	4,310.4	500	32	**4,194.42**	524	−0.41
BHRC1DO.30	20	34	4,506.3	41	4,852.2	397	34	4,534.4	466	34	**4,461.34**	652	−1.00
BHRC1DO.50	50	34	4,513.9	41	4,329.4	560	34	4,513.9	458	34	**4,490.90**	610	−0.51
BHRC1UP.10	10	33	4,105.3	40	4,453.9	446	33	4,137	488	33	**4,103.47**	261	−0.04
BHRC1UP.30	30	35	4,538	43	4,722.4	372	35	4,538	459	35	**4,508.02**	435	−0.66
BHRC1UP.50	50	35	4,550.2	41	4,936.3	436	35	4,558.5	464	35	**4,522.18**	356	−0.62
CNV/CTD/CCT		443	54,664.00	517	57,644.90	4,966	445	54,709.90	5,899	**443**	**54,465.15**	5,291	
MNV/MTD/MCT		36.92	4,555.33	43.08	4,803.74	413.83	37.08	4,559.16	491.58	**36.92**	**4,538.76**	440.92	
Machine					NMM 33 M			PIV 1.5 G			Intel X7900 2.8 G		
Runs					(5 versions)			10			3		
Rel. speed					—			1			10.24		
Norm. time					—			4,916			13,546		

Table 3 Detailed Results for VRPCBTW 500-Customer Thangiah, Potvin, and Sun (1996) Data Set

Problem instance	%B	BK		TPS			RP			APR			
		NV	TD	NV	TD	CT	NV	TD	CT	NV	TD	CT	%Dev
BHR1DO.10	10	58	6,860.2	67	7,620.4	3,355	58	6,868	1,763	58	**6,819.66**	986	−0.59
BHR1DO.30	30	58	7,337.2	71	8,128.1	3,383.1	59	7,262.3	1,595	58	7,358.66	1,203	0.29
BHR1DO.50	50	60	7,294.7	76	8,376.5	3,515.1	60	7,294.7	1,584	60	7,311.57	1,233	0.23
BHR1UP.10	10	54	6,702.7	64	7,333.6	3,219	54	6,784.7	1,692	54	**6,683.05**	1,076	−0.29
BHR1UP.30	30	57	6,991	74	8,290.2	4,503.1	57	6,991	1,566	**56**	**7,078.67**	1,789	
BHR1UP.50	50	58	7,217.3	68	8,043.7	5,489.9	58	7,217.3	1,548	58	**7,158.02**	1,432	−0.82
BHRC1DO.10	10	52	6,313.3	61	7,099.4	4,320.1	52	6,313.3	1,658	52	**6,303.25**	1,201	−0.16
BHRC1DO.30	20	54	6,813.6	63	7,707.1	4,334.4	54	6,813.6	1,530	54	**6,771.81**	1,329	−0.61
BHRC1DO.50	50	54	6,896.5	65	7,771.6	4,321.9	54	6,896.5	1,520	54	**6,879.67**	1,002	−0.24
BHRC1UP.10	10	55	6,464.1	63	7,209.4	3,227.7	55	6,464.1	1,591	**54**	**6,496.58**	1,843	
BHRC1UP.30	30	57	7,028.3	63	7,967.1	5,247.3	57	7,028.3	1,500	**55**	**6,892.66**	1,674	
BHRC1UP.50	50	56	6,969.6	68	8,809.5	4,440.6	57	6,862.3	1,296	56	**6,850.48**	932	−1.71
CNV/CTD/CCT		673	82,888.50	803	94,356.60	49,357.2	675	82,796.10	18,843	**669**	**82,604.08**	15,700	
MNV/MTD/MCT		56.08	6,907.38	66.92	7,863.05	4,113.10	56.25	6,899.68	1,570.25	**55.75**	**6,883.67**	1,308.33	
Machine					NMM 33 M			PIV 1.5 G			Intel X7900 2.8 G		
Runs					(5 versions)			10			3		
Rel. speed					—			1			10.24		
Norm. time					—			15,702			40,192		

lists the authors using the following abbreviations: BK stands for the best known results; TPS stands for Thangiah, Potvin, and Sun (1996); RDH stands for Reimann, Doerner, and Hartl (2002); RU stands for Reimann and Ulrich (2006); HKK stands for Hasama, Kokubugata, and Kawashima (1998); ZC stands for Zhong and Cole (2005); RP stands for Ropke and Pisinger (2006); and APR stands for the proposed Adaptive Path Relinking method. The columns illustrate the total number of vehicles (NV), the total distance traveled (TD) and the computational time (CT) in seconds for each problem instance, and %B denotes the percentage of backhaul customers with respect to the total number of customers. At the right-hand side, (%Dev) stands for the percentage deviation with respect to the best known results. At the middle section, mean and cumulative results are reported. In particular, MNV, MTD, and MCT stand for mean number of vehicles, mean traveling distance, and mean computational time, and CNV, CTD, and CCT stand for cumulative number of vehicles, cumulative traveling distance, and cumulative computational time. In every case, boldface indicates new best solutions.

The bottom sections of all tables describe the machine used, the number of runs, the relative speed of the machine, and the normalized average computational time consumption. The relative speed of each machine is derived with respect to a Pentium IV 1.5 GHz, using the PassMark® CPU marks (http://www.cpubenchmark.net/cpu_list.php). To that end, the normalized computational time is calculated from the relative speed multiplied by the mean computational time (in seconds) and the number of runs. It is worth mentioning that this procedure provides an efficiency indication for each solution approach; however, it cannot be used as a basis for direct comparisons.

Based on the computational results reported in Table 1, APR seems to be the best performing approach compared to current state-of-the-art solution methods for VRPCBTW instances. In particular, six new best known solutions are obtained, and cost reductions up to 0.07% are reached with respect to the current best known results. Furthermore, APR improves the best known mean and cumulative number of vehicles over all problem instances, and the maximum deviation from the current best known solutions is less than 1.53% in terms of distance traveled. To that end, the average results obtained over multiple simulation runs revealed relatively small variations (close to 0.8% in the worst case) in terms of distance traveled and no variation in terms of the fleet size.

Tables 2 and 3 summarize the results obtained on the large scale VRPCBTW instances of Thangiah, Potvin, and Sun (1996) with 250 and 500 customers, respectively. Compared to the current best known solutions reported in the literature, 21 new best solutions are obtained out of 24 instances, with cost reductions up to 1.71%. The maximum deviation from the current best in the class approach of Ropke and Pisinger (2006) is less than 0.67%. Furthermore, APR improves the best known mean and cumulative number of vehicles over all problem instances, and significant reductions are obtained with respect to the total number of vehicles on large-scale problem instances with 500 customers. Overall, the aver-

Table 4 Detailed Results for VRPMBTW Kontoravdis and Bard (1995) Data Set

Problem Instance	BK		RP			ZC			APR			
	NV	TD	NV	TD	CT	NV	TD	CT	NV	TD	CT	%Dev
MR201	4	1,256.31	4	1,256.31	160	4	1,388.73	108	4	1,256.31	12	0.00
MR202	4	1,086.46	4	1,086.46	362	4	1,198.99	422	4	1,086.46	32	0.00
MR203	4	894.54	4	896.14	374	4	988.82	544	4	894.54	57	0.00
MR204	4	736.75	4	736.75	432	4	858.32	450	4	737.43	42	0.09
MR205	4	974.26	4	974.26	353	4	1,172.53	253	4	974.26	76	0.00
MR206	4	894.04	4	894.04	388	4	979.5	406	4	**893.67**	34	−0.04
MR207	4	800.79	4	800.79	422	4	912.69	451	4	800.79	21	0.00
MR208	4	716.28	4	716.28	431	4	764.52	408	4	716.28	13	0.00
MR209	4	879.63	4	881.6	361	4	978.82	516	4	**877.04**	21	−0.29
MR210	4	924.56	4	924.56	369	4	1,061.36	557	4	924.56	39	0.00
MR211	4	763.09	4	765.03	394	4	878.81	692	4	763.09	345	0.00
CNV/CTD/CCT	44	9,926.71	44	9,932.22	4,046	44	11,183.09	4,807	**44**	**9,924.42**	692	
MNV/MTD/MCT	4.00	902.43	4.00	902.93	367.82	4.00	1,016.64	437.00	**4.00**	**902.22**	62.91	
MRC201	5	1,346.3	5	1,355.63	236	5	1,498.9	73	5	**1,338.96**	257	−0.55
MRC202	4	1,230.24	4	1,230.24	171	4	1,539.41	493	4	**1,223.70**	158	−0.53
MRC203	4	995.63	4	995.63	176	4	1,303.48	713	4	**987,80**	183	−0.79
MRC204	4	833.6	4	836.89	187	4	932.48	472	4	833.60	76	0.00
MRC205	4	1,414.52	4	1,414.52	160	4	1,632.04	362	4	**1,411.19**	102	−0.24
MRC206	4	1,231.52	4	1,254.51	166	4	1,433.43	208	4	**1,221.74**	123	−0.79
MRC207	4	1,083.33	4	1,083.33	169	4	1,217.2	599	4	**1,066.24**	56	−1.58
MRC208	4	847.46	4	849.3	175	4	1,085.57	134	4	**843.58**	234	−0.46
CNV/CTD/CCT	33	8,982.60	33	9,020.05	1,440	33	10,642.51	3,054	**33**	**8,926.81**	1,189	
MNV/MTD/MCT	4.13	1,122.83	4.13	1,127.51	180.00	4.13	1,330.31	381.75	**4.13**	**1,115.85**	148,63	
Machine				PIV 1.5 G			P 450 M			Intel X7900 2.8 G		
Runs				10			1			3		
Rel. speed				1			≈0.1			10.24		
Norm. time				2,887			41			3,041		

age results obtained over multiple simulation runs revealed very small variations.

On the other hand, Table 4 summarizes the results obtained on the benchmark data sets of Kontoravdis and Bard (1995) for VRPMBTW instances. Note that the lower bound for the fleet size—as reported by Kontoravdis and Bard (1995)—is 4 for all problem instances, and the density of backhaul customers is 49%. The results obtained illustrate the performance of APR compared to earlier results. In particular, new best solutions are obtained for 9 out of 19 problem instances, with cost reductions up to 1.58%. Regarding the first group of problems (MR2), APR produces the lowest known cumulative distance traveled, with a 0.09% maximum deviation from the best known solutions. The figures are similar for the second group of problems (MRC2).

Regarding computational times, APR seems to be slower compared to other approaches for problem instances with clustered backhauls, whereas the computational efficiency is quite similar for problem instances with mixed backhauls. However, in all cases the reported computational times are reasonable for practical applications. Furthermore, the proposed APR approach seems to scale well for both medium- and large-scale problem instances.

5.4. Comparative Analysis with Existing Results for VRPTW Instances

Following the notation introduced earlier, Tables 5–9 summarize the results obtained for Gehring and Homberger's (1999) data sets. Each table is divided into three parts. The first part refers to the different classes R1, R2, C1, C2, RC1, and RC2, and reports the corresponding MNV and MTD values. The second part reports the cumulative results, i.e., CNV and CTD, over all problem instances. The last part describes the machine, the number of runs, and the average CPU time in minutes (MCT), along with the relative speed and the normalized time with respect to a PIV 2.0 GHz machine. The authors are listed using the following abbreviations: HY for Hashimoto and Yagiura (2008); MB for Mester and Braysy (2005); I for Ibaraki et al. (2008); LZ for Lim and Zhang (2007); PR for Pisinger and Ropke (2006); DPR for Prescott-Gagnon, Desaulniers, and Rousseau (2009); RTI for Repoussis, Tarantilis, and Ioannou (2009); VCGP for Vidal et al. (2011); and NBD for Nagata, Braysy, and Dullaert (2010). In all tables, boldface entries indicate best known results.

Compared to the best performing solution methods for the VRPTW, APR generates competitive results,

Table 5 Comparison of Solution Methods on Group G02 with 200 Customers

Class	APR	RTI	DPR	MB	I	LZ	PR	VCGP	NBD	HY
R1	18.20	18.20	18.20	18.20	18.20	18.20	18.20	18.20	**18.20**	18.20
	3,630.21	3,640.11	3,615.69	3,618.68	3,655.24	3,639.60	3,631.23	3,613.16	**3,612.36**	3,632.85
R2	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	**4.00**	4.00
	2,930.08	2,941.99	2,937.67	2,942.92	2,958.56	2,950.09	2,949.37	2,929.41	**2,929.41**	2,967.02
C1	18.90	18.90	18.90	**18.80**	18.90	18.90	18.90	18.90	18.90	18.90
	2,720.86	2,721.90	2,718.77	**2,717.21**	2,734.42	2,726.11	2,721.52	2,718.41	2,718.41	2,718.68
C2	6.00	6.00	**6.00**	6.00	6.00	6.00	6.00	6.00	6.00	6.00
	1,831.59	1,833.36	**1,831.59**	1,833.57	1,833.37	1,834.24	1,832.94	1,831.59	1,831.64	1,833.12
RC1	18.00	18.00	18.00	18.00	18.00	18.00	18.00	18.00	**18.00**	18.00
	3,213.08	3,224.63	3,192.56	3,221.34	3,275.38	3,205.51	3,212.28	3,180.48	**3,178.68**	3,196.86
RC2	4.30	4.30	4.30	4.40	4.30	4.30	4.30	**4.30**	4.30	4.30
	2,542.29	2,554.33	2,559.32	2,519.79	2,576.12	2,574.10	2,556.87	**2,536.20**	2,536.22	2,572.55
CNV	694	694	694	694	694	694	694	694	**694**	694
CTD	168,681	169,163	168,556	168,573	170,331	169,296	169,042	168,092	**168,067**	169,070
Machine	X 2.8 G	PIV 3 G	O 2.3 G	PIV 2 G	PIV 2.8 G	PIV 2.8 G	PIV 3 G	Xe 2.93 G	O 2.4 G	Xe 2.8 G
MCT	27	90	53	8	33.3	93.2	7.7	8.4	4.1	33
Runs	1	1	5	1	1	1	10	5	5	1
Rel. speed	7.13	1.99	2.26	1.00	1.68	1.68	1.99	22.03	2.53	2.11
Norm. time	193	179	599	8	56	157	153	925	52	69

Table 6 Comparison of Solution Methods on Group G04 with 400 Customers

Class	APR	RTI	DPR	MB	I	LZ	PR	VCGP	NBD	HY
R1	36.40	36.40	36.40	**36.30**	36.30	36.40	36.40	36.40	36.40	36.40
	8,464.14	8,514.11	8,420.52	**8,530.03**	8,788.54	8,489.53	8,540.04	8,402.57	8,403.24	8,544.80
R2	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	**8.00**	8.00
	6,193.75	6,258.82	6,213.48	6,209.94	6,251.54	6,271.57	6,241.72	6,152.92	**6,148.57**	6,262.43
C1	37.60	37.60	37.60	37.90	37.60	37.60	37.60	**37.60**	37.60	37.60
	7,201.30	7,273.90	7,182.75	7,148.27	7,302.50	7,229.04	7,290.16	**7,170.47**	7,175.72	7,213.21
C2	11.60	11.70	11.90	12.00	11.80	11.70	12.00	**11.60**	11.70	11.80
	3,996.13	3,941.70	3,874.58	3,840.85	3,985.21	3,942.93	3,844.69	**3,952.95**	3,899.00	3,910.11
RC1	36.00	36.00	36.00	36.00	36.00	36.00	36.00	**36.00**	36.00	36.00
	7,968.64	8,088.46	7,940.65	8,066.44	8,471.85	8,005.25	8,069.30	**7,907.14**	7,922.23	8,017.89
RC2	8.40	8.40	8.60	8.80	8.60	8.50	8.50	8.50	**8.40**	8.50
	5,398.33	5,516.59	5,269.09	5,243.06	5,328.84	5,431.15	5,335.09	5,215.21	**5,297.86**	5,326.87
CNV	**1,380**	1,381	1,385	1,389	1,384	1,382	1,385	1,381	1,381	1,383
CTD	**392,223**	395,936	389,011	390,386	401,285	393,695	393,210	388,013	388,466	392,507
Machine	X 2.8 G	PIV 3 G	O 2.3 G	PIV 2 G	PIV 2.8 G	PIV 2.8 G	PIV 3 G	Xe 2.93 G	O 2.4 G	Xe 2.8 G
MCT	82	180	89	17	66.6	295.9	15.8	34.1	16.2	67
Runs	1	1	5	1	1	1	10	5	5	1
Rel. speed	7.13	1.99	2.26	1.00	1.68	1.68	1.99	22.03	2.53	2.11
Norm. time	585	358	1,005	17	112	498	314	3,756	205	141

Table 7 Comparison of Solution Methods on Group G06 with 600 Customers

Class	APR	RTI	DPR	MB	I	LZ	PR	VCGP	NBD	HY
R1	54.50	54.50	54.50	54.50	54.50	54.50	54.50	**54.50**	54.50	54.50
	18,460.74	18,781.79	18,252.13	18,358.68	19,963.56	18,381.28	18,888.52	**18,023.18**	18,186.24	18,523.42
R2	11.00	11.00	11.00	11.00	11.00	11.00	11.00	11.00	**11.00**	11.00
	12,566.77	12,804.60	12,808.59	12,703.52	12,496.54	12,847.31	12,619.26	12,352.38	**12,330.49**	12,678.99
C1	**57.30**	57.30	57.40	57.80	57.50	57.40	57.50	57.40	57.40	57.40
	14,230.86	14,236.86	14,106.03	14,003.09	14,128.87	14,103.61	14,065.89	14,058.46	14,067.34	14,163.51
C2	17.40	17.40	17.50	17.80	17.40	17.40	17.50	**17.40**	17.40	17.40
	7,659.20	7,729.80	7,632.37	7,455.83	7,991.70	7,725.86	7,801.29	**7,594.41**	7,605.07	7,678.49
RC1	55.00	55.00	55.00	55.00	55.00	55.00	55.00	**55.00**	55.00	55.00
	16,833.66	16,767.72	16,266.14	16,418.63	17,395.51	16,274.17	16,594.94	**16,097.05**	16,183.95	16,352.56
RC2	11.40	11.40	11.70	12.10	11.60	11.50	11.60	11.50	**11.40**	11.50
	10,876.39	11,311.81	10,990.85	10,677.46	10,743.03	10,935.91	10,777.12	10,511.86	**105,886.14**	10,841.22

Table 7 **Continued**

Class	APR	RTI	DPR	MB	I	LZ	PR	VCGP	NBD	HY
CNV	**2,066**	2,066	2,071	2,082	2,070	2,068	2,071	2,068	2,067	2,068
CTD	**806,276**	816,326	800,797	796,172	827,192	802,681	807,470	786,373	789,592	800,982
Machine	X 2.8 G	PIV 3 G	O 2.3 G	PIV 2 G	PIV 2.8 G	PIV 2.8 G	PIV 3 G	Xe 2.93 G	O 2.4 G	Xe 2.8 G
MCT	134	270	105	40	100	646.9	18.3	99.4	25.3	100
Runs	1	1	5	1	1	1	10	5	5	1
Rel. speed	7.13	1.99	2.26	1.00	1.68	1.68	1.99	22.03	2.53	2.11
Norm. time	956	537	1,186	40	168	1,090	364	10,948	320	211

Table 8 **Comparison of Solution Methods on Group G08 with 800 Customers**

Class	APR	RTI	DPR	MB	I	LZ	PR	VCGP	NBD	HY
R1	72.80	72.80	72.80	72.80	72.80	72.80	72.80	**72.80**	72.80	72.80
	32,053.16	32,734.57	31,797.42	31,918.47	33,275.72	31,755.57	32,316.79	**31,311.38**	31,492.81	31,978.15
R2	15.00	15.00	15.00	15.00	15.00	15.00	15.00	15.00	**15.00**	15.00
	20,361.56	20,618.21	20,651.81	20,295.28	20,209.92	20,601.22	20,353.51	19,933.39	**19,914.97**	20,383.13
C1	75.20	75.20	75.40	76.20	75.70	75.40	75.60	75.40	**75.20**	75.20
	25,871.64	25,911.44	25,093.38	25,132.27	25,487.55	25,026.42	25,193.13	24,876.38	**25,151.83**	25,220.58
C2	23.40	23.40	23.50	23.70	23.40	23.40	23.70	**23.30**	23.40	23.30
	11,594.51	11,835.72	11,569.39	11,352.29	11,860.90	11,598.81	11,725.46	**11,475.05**	11,447.27	11,689.00
RC1	72.00	72.00	72.00	73.00	72.40	72.00	73.00	**72.00**	72.00	72.00
	33,518.25	33,795.61	33,170.01	30,731.07	34,621.63	31,267.84	29,478.30	**29,404.32**	31,278.28	31,343.54
RC2	15.40	15.50	15.80	15.80	15.70	15.60	15.70	15.40	**15.40**	15.40
	17,036.41	17,536.54	16,852.38	16,729.18	16,666.76	16,992.79	16,761.95	16,495.82	**16,484.31**	16,828.93
CNV	2, 738	2,739	2,745	2,765	2,750	2,742	2,758	2,739	2,738	**2,737**
CTD	1,404,355	1,424,321	1,391,344	1,361,586	1,421,225	1,372,427	1,358,291	1,334, 963	1,357,695	**1,367,971**
Machine	X 2.8 G	PIV 3 G	O 2.3 G	PIV 2 G	PIV 2.8 G	PIV 2.8 G	PIV 3 G	Xe 2.93 G	O 2.4 G	Xe 2.8 G
MCT	190	360	129	145	133.3	1,269.4	22.7	215	27.6	133
Runs	1	1	5	1	1	1	10	5	5	1
Rel. speed	7.13	1.99	2.26	1.00	1.68	1.68	1.99	22.03	2.53	2.11
Norm. time	1,355	716	1,457	145	225	2,138	451	23,680	349	280

Table 9 **Comparison of Solution Methods on Group G10 with 1,000 Customers**

Class	APR	RTI	DPR	MB	I	LZ	PR	VCGP	NBD	HY
R1	91.90	91.90	91.90	92.10	91.90	91.90	92.20	**91.90**	91.90	91.90
	50,284.56	51,414.26	49,702.32	49,281.48	53,366.10	48,827.23	50,751.25	**47,759.66**	48,287.98	49,650.82
R2	19.00	19.00	19.00	19.00	19.00	19.00	19.00	19.00	**19.00**	19.00
	29,895.31	30,804.79	30,495.26	29,860.32	29,546.19	30,164.60	29,780.82	29,076.45	**28,913.40**	29,806.86
C1	94.10	94.20	94.30	95.10	94.50	94.40	94.60	94.10	94.10	**94.00**
	42,450.88	43,111.60	41,783.27	41,569.67	42,459.35	41,699.32	41,877.00	41,572.86	41,683.29	**42,157.55**
C2	29.10	29.30	29.50	29.70	29.40	29.30	29.70	**28.80**	29.10	28.80
	16,675.10	16,810.22	16,657.06	16,639.54	16,986.46	16,589.74	16,840.37	**16,796.45**	16,498.61	17,152.20
RC1	90.00	90.00	90.00	90.00	90.00	90.00	90.00	**90.00**	90.00	90.00
	46,245.48	46,753.61	45,574.11	45,396.41	48,275.20	44,818.54	46,752.15	**44,333.40**	44,743.18	45,539.11
RC2	18.30	18.40	18.50	18.70	18.30	18.30	18.30	**18.20**	18.30	18.30
	25,075.59	25,588.52	25,470.33	25,063.51	24,904.08	25,064.88	25,090.88	**24,131.13**	23,939.62	24,696.91
CNV	3,424	3,428	3,432	3,446	3,431	3,429	3,438	**3,420**	3,424	3,420
CTD	2,106,269	2,144,830	2,096,823	2,078,110	2,155,374	2,071,643	2,110,925	**2,036,700**	2,045,720	2,085,125
Machine	X 2.8 G	PIV 3 G	O 2.3 G	PIV 2 G	PIV 2.8 G	PIV 2.8 G	PIV 3 G	Xe 2.93 G	O 2.4 G	Xe 2.8 G
MCT	240	450	162	600	166.7	1,865.4	26.6	349	35.3	167
Runs	1	1	5	1	1	1	10	5	5	1
Rel. speed	7.13	1.99	2.26	1.00	1.68	1.68	1.99	22.03	2.53	2.11
Norm. time	1,712	895	1,830	600	281	3,142	529	38,439	446	352

and produces the lowest known CNV for most problem groups with very reasonable computational times. For problem instances with 200 customers the same CNV is exhibited, and for problem instances with 400 and 600 customers the best known mean and cumulative number of vehicles is improved. Overall, the quality of the solutions produced by APR is in the worst case next to the best solutions reported by Nagata, Braysy, and Dullaert (2010) and Vidal et al. (2011) (see columns NBD and VCGP).

5.5. Effect of Backhauling Strategies and Mixing Levels

This section studies the effect and impact of different backhauling strategies. As mentioned earlier, the benchmark data sets for VRPCBTW and VRPMBTW instances can be used interchangeably for both cases. As such, useful comparisons can be made between the effect of backhaul customers and the effect of scheduling horizons (e.g., comparison between shorthaul and longhaul problem instances).

Tables 10 and 11 summarize the results obtained from the application of APR on Gelinas et al. (1995) and Kontoravdis and Bard (1995) data sets with different backhauling strategies. Both tables consist of five main columns that correspond to different mixing level restrictions by enforcing constraints relevant to the remaining delivery capacity of the vehicles. As mentioned earlier in §5.1, these mixing settings depend on the choice of a threshold parameter ξ that will determine a lower bound on the sum of remaining delivery demands such that a pickup is allowed. In particular, the values assumed for ξ are 0, 0.25, 0.50, 0.75, and 1, and each column reports the corresponding number of vehicles and traveling distance. Recall also that no a priori restrictions are imposed when ξ equals 1, whereas all linehaul customers are strictly visited before backhaul customers if ξ is set to 0 (see also Tables 1 and 4).

Because of the number of vehicle changes, an alternative evaluation measure of the form 100MNV + MTD is adopted in order for the competition to be fair. To that end, the bottom section of both tables reports the average modified cost (Cost) of each strategy and the associated gap (%Gap) with respect to the pure mixed backhauling strategy; i.e., $\xi = 1$. The latter can be seen as the additional cost one has to pay if visiting order restrictions are gradually imposed. Finally, the second row from the bottom reports the corresponding average results.

In all cases, the backhauling strategy that allows the mixed order of pickups and deliveries on the same vehicles is significantly more effective, and low cost vehicle routing plans are produced. More specifically, on the problem instances of Gelinas et al. (1995) with short scheduling horizons (see Table 10), the additional cost of routing backhaul customers after linehaul customers is 17.34%. On the other hand, the corresponding averages on Kontoravdis and Bard (1995) problem instances (see Table 11) with long scheduling horizons and relatively many customers per vehicle route are much higher. In particular, the average is close to 31% for problem instances with randomly scattered customers and more than 43% for problem instances with mixed random and clustered customers. Thus, it seems that as the number of customers per vehicle route increases (for a homogeneous fleet of capacitated vehicles), the cost of enforcing visiting order restrictions also increases, even

Table 10 Comparison of Different Mixing Levels on Gelinas et al. (1995) Data Set

Problem instance			$\xi = 1.00$		$\xi = 0.75$		$\xi = 0.50$		$\xi = 0.25$		$\xi = 0.00$	
Name	Group	%B	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
R101	A	10	19	1,650.80	19	1,650.80	19	1,650.80	19	1,667.72	22	1,818.86
	B	30	19	1,650.80	19	1,650.80	19	1,650.80	19	1,654.67	23	1,959.52
	C	50	19	1,650.80	19	1,650.80	19	1,650.80	19	1,655.45	24	1,939.1
	A	10	17	1,486.12	17	1,486.12	17	1,486.12	17	1,507.21	19	1,653.18
R102	B	30	17	1,501.84	17	1,503.77	17	1,519.49	18	1,621.22	22	1,752.28
	C	50	17	1,486.12	17	1,486.12	17	1,486.12	18	1,475.05	22	1,775.76
	A	10	13	1,292.68	13	1,292.68	13	1,292.68	14	1,247.59	15	1,385.38
R103	B	30	13	1,291.95	13	1,291.95	13	1,294.52	14	1,239.85	15	1,390.33
	C	50	13	1,291.95	13	1,291.95	13	1,291.95	14	1,228.75	17	1,456.48
	A	10	9	1,007.31	9	1,007.31	9	1,007.31	10	999.71	10	1,215.69
R104	B	30	9	1,007.31	9	1,007.31	9	1,018.76	10	1,027.12	11	1,154.84
	C	50	9	1,007.31	9	1,007.31	10	983.98	10	1,019.30	11	1,194.73
	A	10	14	1,377.11	14	1,377.11	14	1,377.11	14	1,410.18	15	1,560.15
R105	B	30	14	1,377.11	14	1,377.11	14	1,383.92	14	1,434.97	16	1,583.3
	C	50	14	1,377.11	14	1,377.11	14	1,383.92	14	1,412.69	16	1,709.66
MNV/MTD			14.40	1,363.75	14.40	1,363.88	14.47	1,365.22	14.93	1,373.43	17.20	1,569.95
Cost/%Gap			2,803.75	0.00	2,803.88	0.00	2,811.89	0.29	2,866.76	2.25	3,289.95	17.34

Table 11 Comparison of Different Mixing Levels on Kontoravdis and Bard (1995) Data Set

Problem instance	$\xi = 1.00$		$\xi = 0.75$		$\xi = 0.50$		$\xi = 0.25$		$\xi = 0.00$	
	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
MR201	4	1,256.31	4	1,260.43	4	1,350.22	5	1,623.31	5	1,723.92
MR202	4	1,086.46	4	1,087.24	4	1,117.40	4	1,356.95	5	1,464.50
MR203	4	894.54	4	894.54	4	910.30	4	1,036.66	4	1,413.32
MR204	4	737.43	4	741.02	4	773.88	4	819.62	4	977.95
MR205	4	974.26	4	983.54	4	1,025.13	4	1,193.33	5	1,286.89
MR206	4	893.67	4	900.93	4	926.37	4	1,023.84	5	1,167.21
MR207	4	800.79	4	810.81	4	830.84	4	883.78	4	1,085.30
MR208	4	716.28	4	720.28	4	740.07	4	777.23	4	912.63
MR209	4	877.04	4	882.47	4	937.13	4	1,042.94	4	1,449.45
MR210	4	924.56	4	928.36	4	947.39	4	1,061.99	4	1,360.79
MR211	4	763.09	4	764.86	4	795.03	4	901.72	4	1,076.16
MNV/MTD	4.00	902.22	4.00	906.77	4.00	941.25	4.09	1,065.58	4.36	1,265.28
Cost/%Gap	1,302.22	0.00	1,306.77	0.35	1,341.25	3.00	1,474.67	13.24	1,701.65	30.67
MRC201	5	1,338.96	5	1,345.01	5	1,446.65	6	1,595.27	6	2,012.92
MRC202	4	1,223.70	4	1,245.22	4	1,345.19	5	1,412.72	5	1,865.20
MRC203	4	987.80	4	1,008.86	4	1,029.35	4	1,265.24	5	1,413.17
MRC204	4	833.60	4	845.32	4	873.85	4	935.39	4	1,236.53
MRC205	4	1,411.19	4	1,429.48	4	1,730.03	5	1,580.42	6	1,769.78
MRC206	4	1,221.74	4	1,250.66	4	1,609.63	5	1,461.52	6	1,869.22
MRC207	4	1,066.24	4	1,096.77	4	1,233.62	5	1,238.97	5	1,574.65
MRC208	4	843.58	4	897.78	4	957.99	4	1,124.81	4	1,661.11
MNV/MTD	4.13	1,115.85	4.13	1,139.89	4.13	1,278.29	4.75	1,326.79	5.125	1,675.32
Cost/%Gap	1,528.35	0.00	1,552.39	1.57	1,690.79	10.63	1,801.79	17.89	2,187.82	43.15

if the total number of backhaul customers remains the same.

Along the same lines, the effect of relaxing the linehaul-backhaul precedence restrictions seems to be strong. Regarding longhaul problem instances, starting from the restrictive policy of $\xi = 0$, significant improvements are obtained as ξ gradually increases. The peak is observed at the early stages (region from 0 to 0.5) as ordering restrictions are relaxed (i.e., 3% to 13% improvements for $\xi = 0.25$, and 10% to 17% improvements for $\xi = 0.5$). However, the influence of ξ after this point gradually fades out (i.e., 0.3% to 2% improvements are obtained when moving from 0.75 to 1). On the contrary, small performance degradation for different levels of linehaul-backhaul precedence restrictions (less than 3%) are observed considering shorthaul problem instances. However, the difference in the region 0 to 0.25 is quite large (more than 14%).

Overall, it is apparent that the optimal policy is to allow the free mixing of linehaul and backhaul customers, and even a small increase in scheduling flexibility may result in significant improvements, at least for longhaul problem instances. On the other hand, it seems that in cases where time windows dictate to a large extent the vehicle routing plan, and relatively few customers are served per vehicle route, the effect of different mixing levels plays a rather minor role. It is also worth mentioning that the earlier work of Reimann and Ulrich (2006) arrives at similar conclusions.

5.6. Effect of the Density of Backhaul Customers

Another important aspect is the density of backhaul customers because they may significantly affect the cost of the resulting vehicle routing plans. From Table 10, it is observed that at least for VRPCBTW instances both fleet sizes and traveling distances increase as the total number of backhaul customers increases. Note that the opposite effect occurs when the percentage of backhaul customers increases beyond 50% (the extreme cases of 100% linehauls or 100% backhauls are equivalent). The figures are similar for the large scale VRPCBTW instances of Thangiah, Potvin, and Sun (1996) (see Tables 2 and 3). Figures 1–3 illustrate the effect of the number of backhaul customers on the number of vehicles and the distance traveled considering VRPCBTW instances with 100, 250, and 500 customers, respectively.

On the other hand, the effect of the density of backhaul customers with respect to the percentage of deviation among different backhaul strategies and mixing levels seems to be limited (see Table 10). However, one may expect that this is probably not the case for longhaul problem instances. Finally, it is worth mentioning that the worst results in terms of solution quality are observed with mixes of 50% linehauls and 50% backhauls (with very few exceptions).

5.7. Analysis of Method Components

This section analyzes the role of the main algorithmic components of the proposed APR solution framework. In particular, we measure the impact of the

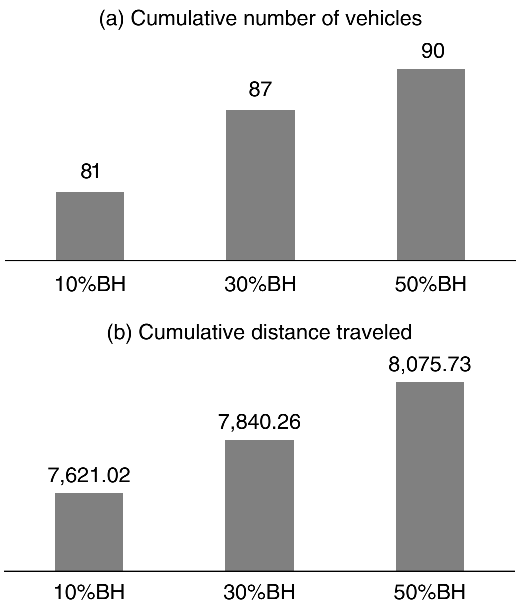

Figure 1 Effect of Backhauls Density on 100-Customer Gelinas et al. (1995) Data Set

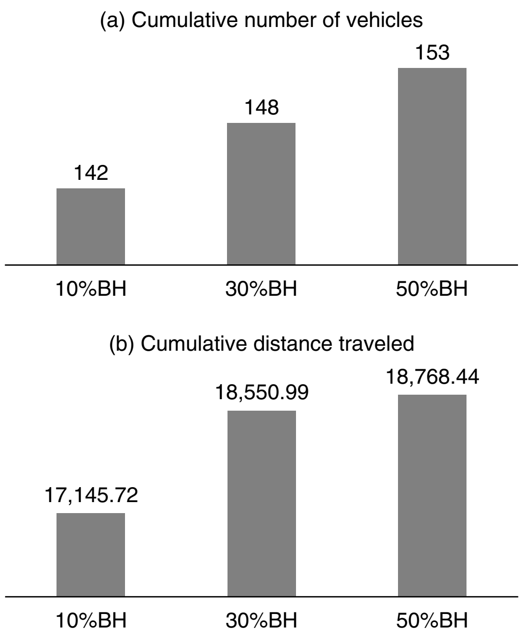

Figure 2 Effect of Backhauls Density on 250-Customer Thangiah, Potvin, and Sun (1996) Data Set

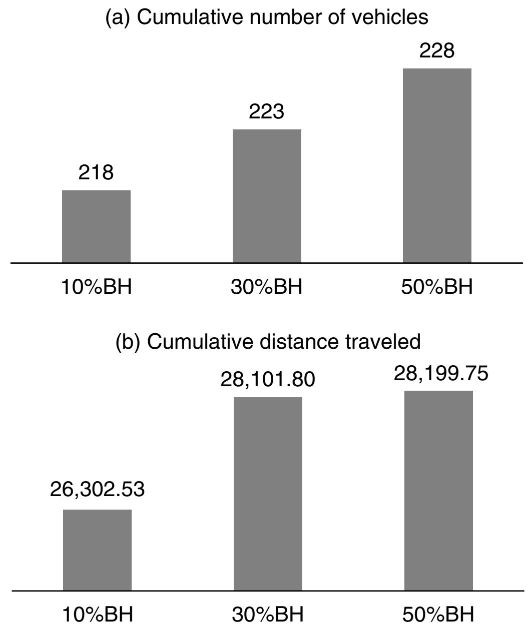

Figure 3 Effect of Backhauls Density on 500-Customer Thangiah, Potvin, and Sun (1996) Data Set

adaptive path relinking mechanism, the contribution of infeasible solutions, and the reference set update method. To that end, we examined three configurations by removing one element at a time from the method. The first (Config-1) does not incorporate the construction of provisional solutions and selects at random a guiding solution from the reference set. Observe that in this case the overall framework reduces to an ordinary path relinking solution approach. The second (Config-2) does not allow the acceptance of infeasible solutions within the reference set and relies exclusively on feasible solutions. In this configuration, infeasible solutions are only accepted as intermediate solutions for further improvement via local search. Apparently, this element cannot be avoided because of the fleet minimization objective. Finally, the third (Config-3) adopts an elitist scheme for updating the reference set and considers myopically the ordinary (or the penalized) solution cost.

Table 12 summarizes the results obtained from the above described algorithmic configurations on selected benchmark data sets. Each column reports the percentage deviation of each configuration with respect to the best average results of APR in terms of fleet size and traveling distance. Furthermore, the mean computational time is reported in minutes, assuming one simulation run. Clearly, all algorithmic components play an important role in the performance of APR, and they always have a positive contribution. Among them the most crucial seems to be the adaptive path relinking mechanism, followed by the reference set update method. Regarding the contribution of infeasible solutions, their role seems important only for large-scale problems, especially in fleet size minimization.

Overall, these experiments illustrate the pertinence of the APR solution framework for vehicle routing and scheduling problems with pickups and deliveries and underline the role of the proposed APR solution framework as well as the local search improvement method. Based on computational experience, the proposed adaptive multisolution path relinking mechanism exhibits two key properties. The first is that

Table 12 Analysis of APR Algorithmic Configurations

Benchmark	Config-1			Config-2			Config-3		
Data set	Fleet (%)	Distance (%)	Time	Fleet (%)	Distance (%)	Time	Fleet (%)	Distance (%)	Time
VRPCBTW (R1-100)	+0.00	+0.07	1.87	+0.00	+0.01	1.62	+0.00	+0.08	1.14
VRPCBTW (R1-250)	+0.23	+0.34	15.40	+0.00	+0.12	6.53	+0.00	+0.32	6.33
VRPCBTW (R1-500)	+0.75	+0.87	62.34	+0.15	+0.43	23.11	+0.90	+1.02	24.02
VRPMBTW (MR2 + MRC2-100)	+0.00	+0.03	3.65	+0.00	+0.00	3.02	+0.00	+0.00	3.67
VRPTW (G2-200)	+0.00	+1.16	41.02	+0.00	+0.32	24.34	+0.14	+1.35	22.45
VRPTW (G4-400)	+0.29	+3.64	198.01	+0.07	+0.96	85.06	+0.29	+3.12	87.09

during the early stages of solution manipulation the convergence velocity is high because the similarity between the intermediate solutions (produced via the path generation mechanism) and the reference solutions gradually increases. The second property is that during the late stages of the search process the probabilistic reconstruction of the reference solutions offers a very useful variation and increases the chances to escape from local optimum solutions.

6. Conclusions

This paper presented an Adaptive Path Relinking solution method that deals with one-to-many-to-one vehicle routing and scheduling problems with pickups and deliveries. The focus was given on problem settings with clustered and mixed backhauls, including the so-called VRPCBTW and VRPMBTW, and problem instances with different mixing levels and visiting order restrictions are also examined. Considering VRPCBTW instances, all linehaul customers of a route must be serviced before the vehicle starts visiting backhaul customers. Conversely, the mixing of linehaul and backhaul customers along the routes is allowed in VRPMBTW instances.

The proposed solution approach evolves a set of reference solutions by exploring search trajectories and combinations among multiple reference solutions. The main contribution lies in the integration of a large neighborhood search technique with novel diversity measures within the path relinking framework that is used to produce guiding provisional solutions. On return, these provisional solutions are used as guiding points for generating search trajectories from initial reference solutions. The suggested path generation procedure incorporates multiple edge-exchange structures for variation and also benefits from tunneling. To this end, a set of promising solutions is selected and further improved via a local search algorithm. The latter treats both feasible and infeasible solutions on the basis of a new penalized cost function and incorporates computationally efficient neighborhood structures and evaluation methods.

Experimental results on benchmark data sets of the literature demonstrated the competitiveness and robustness of the proposed approach. Compared to existing results for the VRPCBTW and VRPMBTW, it proved to be highly efficient and effective in improving the best reported cumulative and mean results over all sets with reasonable computational requirements and fixed parameter settings. Furthermore, it obtained the minimum published fleet size for all problem instances, and it matched or improved several best known solutions. On the other hand, it also obtains high quality solutions and comparable results with respect to the state-of-the-art methods for the VRPTW. Computational experiments on the effect of backhauling strategies suggested that as the number of customers per vehicle route increases, the cost of enforcing visiting order restrictions also increases, even if the total number of backhaul customers remains the same. Moreover, it is evident that even small increases in scheduling flexibility, considering the mixed order of linehaul and backhaul customers with respect to capacity restrictions, may result in significant cost reductions.

A research direction worth pursuing is toward the design of more sophisticated adaptive path relinking components. One option is to incorporate advanced pattern identification features instead of looking myopically at the recurrence of customer pairs. It is also important to design self adaptive procedures for tuning the variation parameter based on the search progress. Furthermore, the implementation of alternative selection and reference set update strategies may also render considerable improvements. Finally, as research moves toward more realistic and rich problems, the development of large-scale problem instances for one-to-many-to-one vehicle routing and scheduling problems that capture the essence of different backhauling strategies is of great interest.

Acknowledgments

This research was funded by the Research Center of the Athens University of Economics and Business. Support from the State Scholarships Foundation is also acknowledged. The authors wish to thank Prof. Stefan Røpke for kindly providing benchmark data sets. Finally, the authors wish to thank the editor-in-chief, the associate editor, and the two anonymous reviewers for their helpful comments and suggestions.

Appendix. Results on the Large-Scale VRPTW Data Sets

Table A.1 provides the detailed results obtained (i.e., number of vehicles and distance traveling cost) for each VRPTW instance of each class for every group. The table is divided into five parts each corresponding to a group. Each group contains six major columns each associated with a particular class, and the very first column refers to the index of the problem instance.

Table A.1 Results for the Large-Scale VRPTW Data Sets of Gehring and Homberger (1999)

	R1		R2		C1		C2		RC1		RC2	
Set #	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
Group G02—200 customers												
1	20	4,795.04	4	4,483.16	20	2,704.57	6	1,931.44	18	3,622.92	6	3,099.53
2	18	4,063.05	4	3,621.20	18	2,917.89	6	1,863.16	18	3,288.45	5	2,825.24
3	18	3,387.47	4	2,881.15	18	2,725.21	6	1,775.08	18	3,035.29	4	2,617.17
4	18	3,081.19	4	1,981.30	18	2,649.99	6	1,703.43	18	2,854.83	4	2,050.33
5	18	4,124.64	4	3,367.53	20	2,702.05	6	1,878.85	18	3,419.92	4	2,911.46
6	18	3,624.95	4	2,913.03	20	2,701.04	6	1,857.35	18	3,407.64	4	2,880.06
7	18	3,150.11	4	2,453.42	20	2,701.04	6	1,849.46	18	3,208.51	4	2,528.63
8	18	2,970.77	4	1,849.98	19	2,775.48	6	1,820.53	18	3,149.29	4	2,315.59
9	18	3,787.95	4	3,095.03	18	2,687.83	6	1,830.05	18	3,123.37	4	2,175.98
10	18	3,316.96	4	2,654.97	18	2,643.55	6	1,806.58	18	3,020.61	4	2,018.95
Group G04—400 customers												
1	40	10,373.64	8	9,216.35	40	7,152.06	12	4,116.14	36	8,752.91	11	6,687.61
2	36	9,082.67	8	7,609.87	36	7,696.16	12	3,930.25	36	7,997.41	9	6,370.90
3	36	7,964.91	8	6,045.94	36	7,073.44	11	4,118.36	36	7,657.84	8	5,014.53
4	36	7,381.05	8	4,321.92	36	6,866.80	11	3,918.68	36	7,379.95	8	3,663.38
5	36	9,318.96	8	7,153.91	40	7,152.06	12	3,938.69	36	8,305.70	8	6,982.28
6	36	8,485.69	8	6,127.60	40	7,153.45	12	3,875.94	36	8,267.99	8	5,960.64
7	36	7,691.27	8	5,131.11	39	7,443.42	12	3,894.16	36	8,037.72	8	5,468.80
8	36	7,294.60	8	4,039.85	37	7,465.39	11	4,354.87	36	7,803.08	8	4,858.98
9	36	8,797.66	8	6,437.22	36	7,138.36	12	3,877.34	36	7,777.35	8	4,645.50
10	36	8,250.98	8	5,853.75	36	6,871.90	11	3,936.91	36	7,706.41	8	4,330.69
Group G06—600 customers												
1	59	21,472.67	11	18,333.77	60	14,095.64	18	7,774.16	55	17,753.14	14	13,613.52
2	54	19,366.14	11	14,959.42	56	14,367.62	17	8,352.04	55	16,429.35	12	11,779.80
3	54	17,402.73	11	11,293.26	56	13,797.56	17	7,606.40	55	15,585.08	11	9,582.30
4	54	16,213.87	11	8,153.44	56	13,744.61	17	7,083.29	55	15,077.54	11	7,362.30
5	54	20,584.25	11	15,337.79	60	14,085.72	18	7,575.20	55	17,168.10	11	13,491.24
6	54	18,436.36	11	12,878.02	60	14,089.66	18	7,471.17	55	20,223.31	11	12,177.65
7	54	17,172.23	11	10,401.44	57	15,999.27	18	7,513.80	55	16,839.36	11	10,869.08
8	54	16,458.56	11	7,747.55	56	14,637.03	17	7,746.96	55	16,567.52	11	10,411.28
9	54	19,240.80	11	13,854.58	56	13,732.03	17	8,094.05	55	16,386.57	11	10,122.47
10	54	18,259.81	11	12,708.42	56	13,759.41	17	7,374.99	55	16,306.60	11	9,354.32
Group G08—800 customers												
1	80	37,170.53	15	28,325.52	80	25,184.38	24	11,662.08	72	35,950.20	18	21,498.09
2	72	34,008.26	15	23,125.57	72	27,870.48	23	12,498.57	72	33,205.32	16	18,600.27
3	72	30,393.14	15	18,298.92	72	24,699.80	23	11,489.60	72	30,658.39	15	15,019.63
4	72	29,291.67	15	13,852.47	72	24,374.65	23	11,101.30	72	29,103.57	15	11,552.61
5	72	34,474.27	15	24,720.22	80	25,166.28	24	11,434.03	72	35,456.12	15	19,905.72
6	72	31,755.55	15	21,012.84	80	25,160.85	24	11,348.43	72	35,745.20	15	18,823.76
7	72	29,748.81	15	17,215.72	78	25,945.74	24	11,378.45	72	34,253.13	15	17,350.00
8	72	29,076.39	15	13,188.23	74	26,044.71	23	11,623.01	72	33,859.45	15	16,605.35
9	72	33,173.37	15	22,825.91	72	28,232.27	23	12,265.59	72	33,340.68	15	15,907.30
10	72	31,439.64	15	21,050.21	72	26,037.26	23	11,144.08	72	33,610.43	15	15,101.32
Group G10—1,000 customers												
1	100	54,682.65	19	43,065.29	100	42,478.95	30	16,879.24	90	48,231.62	20	31,407.33
2	91	52,681.06	19	34,586.73	90	44,315.58	29	17,135.69	90	45,894.61	19	27,475.07
3	91	48,513.25	19	26,022.43	90	41,030.79	29	16,426.52	90	44,914.23	18	20,896.26
4	91	45,861.96	19	18,826.09	90	40,328.90	28	16,061.04	90	43,878.98	18	16,558.41
5	91	55,131.05	19	37,482.21	100	42,470.12	30	16,576.30	90	47,173.02	18	28,020.83
6	91	50,693.91	19	31,103.08	100	42,471.28	29	17,198.18	90	46,558.83	18	27,515.04
7	91	47,827.70	19	24,309.29	98	43,830.76	30	16,438.27	90	46,835.18	18	26,594.87
8	91	45,909.34	19	18,181.88	93	43,238.79	29	16,454.37	90	46,063.74	18	25,351.63
9	91	51,790.40	19	34,319.36	90	42,317.69	29	16,716.05	90	46,482.00	18	24,103.51
10	91	49,754.25	19	31,056.71	90	42,025.93	28	16,865.39	90	46,422.59	18	22,832.97

References

Anily S (1996) The vehicle routing problem with delivery and backhaul options. *Naval Res. Logist.* 43(3):415–434.

Bent R, van Hentenryck P (2004) A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Sci.* 38(4):515–530.

Bräysy O, Dullaert W, Hasle G, Mester D, Gendreau M (2008) An effective multirestart deterministic annealing metaheuristic for the fleet size and mix vehicle-routing problem with time windows. *Transportation Sci.* 42(3):371–386.

Clarke G, Wright JV (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Oper. Res.* 12(4):568–581.

Dart RC (1983) Success of food backhaul programs could pave way for major expansion. *Traffic World* 8:19–22.

Derigs U, Metz A (1992) A matching-based approach for solving a delivery/pickup vehicle routing problem with time constraints. *Oper. Res. Spektrum* 14(2):91–106.

Duhamel C, Potvin J-Y, Rousseau JM (1997) A tabu search heuristic for the vehicle routing problem with backhauls and time windows. *Transportation Sci.* 31(1):49–59.

Ganesh K, Narendran TT (2007) Cloves: A cluster-and-search heuristic to solve the vehicle routing problem with delivery and pickup. *Eur. J. Oper. Res.* 178(3):699–717.

Gehring H, Homberger J (1999) A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. *Proc. EUROGEN 99, University of Jyvaskyla, Finland*, 57–64.

Gelinas S, Desrochers M, Desrosiers J, Solomon MM (1995) A new branching strategy for time constrained routing problems with application to backhauling. *Ann. Oper. Res.* 61(1):91–100.

Gendreau M, Tarantilis CD (2010) Solving large-scale vehicle routing problems with time windows: The state-of-the-art. Technical report, Université de Motréal.

Glover F (1999) Scatter search and path relinking. Corne D, Dorigo M, Glover F, eds. *New Ideas in Optimisation* (Wiley Publishers), 297–316.

Hasama T, Kokubugata H, Kawashima H (1998) A heuristic approach based on the string model to solve vehicle routing problem with backhauls. *5th World Congress on Intelligent Transport Systems (ITS), Seoul*.

Hashimoto H, Yagiura M (2008) A path relinking approach with an adaptive mechanism to control parameters for the vehicle routing problem with time windows. *Lecture Notes Comput. Sci.* 4972:54–265.

Ho S, Gendreau M (2006) Path relinking for the vehicle routing problem. *J. Heuristics* 12(1-2):55–72.

Ibaraki T, Imahori S, Nonobe K, Sobue K, Uno T, Yagiura M (2008) Iterated local search algorithm for the vehicle routing problem with convex time penalty functions. *Discrete Appl. Math.* 156(11):2050–2069.

Kindervater GAP, Savelsbergh MWP (1998) Vehicle routing: Handling edge exchanges. Aarts E, Lenstra JK, eds. *Local Search in Combinatorial Optimization* (Wiley, London, UK), 337–360.

Kontoravdis G, Bard J (1995) A GRASP for the vehicle routing problem with time windows. *ORSA J. Comput.* 7(1):10–23.

Lim A, Zhang X (2007) A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. *INFORMS J. Comput.* 19(3):443–457.

Mester D, Braysy O (2005) Active guided evolution strategies for large scale vehicle routing problems with time windows. *Comput. Oper. Res.* 32(6):1593–1614.

Mingozzi A, Giorgi S, Baldacci R (1999) An exact method for the vehicle routing problem with backhauls. *Transportation Sci.* 33(3):315–329.

Nagata Y, Braysy O, Dullaert W (2010) A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. *Comput. Oper. Res.* 37(4):724–737.

Parragh S, Doerner KF, Hartl RF (2008a) A survey on pickup and delivery problems part I: Transportation between customers and depot. *J. Betriebswirtschaft* 58(1):21–51.

Parragh S, Doerner KF, Hartl RF (2008b) A survey on pickup and delivery problems, part II: Transportation between pickup and delivery locations. *J. Betriebswirtschaft* 58(2):81–117.

Pisinger D, Ropke S (2006) A general heuristic for vehicle routing problems. *Comput. Oper. Res.* 34(8):2403–2435.

Potvin J-Y, Duhamel C, Guertin F (1996) A genetic algorithm for vehicle routing with backhauling. *Appl. Intelligence* 6(4): 345–355.

Prescott-Gagnon E, Desaulniers G, Rousseau L-M (2009) A branch-and-price large neighborhood search algorithm for the vehicle routing problem with time windows. *Networks* 54(4):190–204.

Reimann M, Ulrich H (2006) Comparing backhauling strategies in vehiicle routing using ant colony optimization. *Central Eur. J. Oper. Res.* 14(2):105–123.

Reimann M, Doerner K, Hartl RF (2002) Insertion based ants for vehicle routing problems with backhauls and time windows. *Lecture Notes Comput. Sci.* 2463:135–148.

Repoussis PP, Tarantilis CD, Ioannou G (2009) Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *IEEE T Evolut. Comput.* 13(3):624–647.

Repoussis PP, Tarantilis CD, Ioannou G (2010) A hybrid evolution strategy for the open vehicle routing problem. *Comput. Oper. Res.* 37(3):443–455.

Ropke S, Pisinger D (2006) A unified heuristic for a large class of vehicle routing problems with backhauls. *Eur. J. Oper. Res.* 171(3):750–775.

Sbihi A, Eglese RW (2007) Combinatorial optimization and green logistics. *4OR: Quart. J. Oper. Res.* 5(2):99–116.

Solomon MM (1987) Algorithms for the vehicle routing problem with time window constraints. *Oper. Res.* 35(2):254–265.

Tarantilis CD (2005) Solving the vehicle routing problem with adaptive memory programming methodology. *Comput. Oper. Res.* 32(9):2309–2327.

Thangiah S, Potvin J-Y, Sun T (1996) Heuristic approaches to vehicle routing with backhauls and time windows. *Comput. Oper. Res.* 23(11):1043–1057.

Toth P, Vigo D (1997) An exact algorithm for the vehicle routing problem with backhauls. *Transportation Sci.* 31(4):372–385.

Vidal T, Crainic TG, Gendreau M, Prins C (2011) A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time windows. *Comput. Oper. Res.* 40(1):475–489.

Yano C, Chan T, Richter L, Cutler T, Murty K, Mcgettigan D (1987) Vehicle routing at Quality Stores. *Interfaces* 17(2):52–63.

Zhong Y, Cole M (2005) A vehicle routing problem with backhauls and time windows: A guided local search solution. *Transportation Res. E* 41(2):131–144.